Despliegue en contenedores. Dockerizando php

1. Se pide crear un fichero **docker-compose.yml** que permite el despliegue automático de los siguientes servicios:

    - Servicio de base de datos:

        - El servicio se llamará **db**
        - Usaremos un Dockerfile para descargar la imagen **mysql** y copiar en ella el script SQL con los datos. Se muestra captura de ejemplo:

            ```
            FROM mysql
            COPY pizzeria.sql /docker-entrypoint-initdb.d/
            ```

            De este modo el script SQL se ejecutará automáticamente en cuanto se inicie el servicio de mysql

        - Usaremos la opción **build** para que se contruya automaticament la nueva imagen a partir del Dockerfile anterior. Emplearemos la sintaxis:

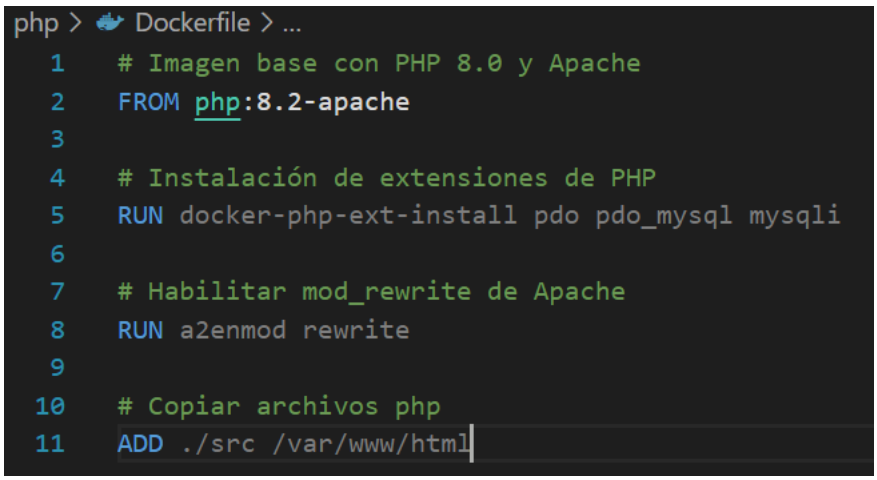
            ```
            build: .\db
            ```

            para indicar donde está el fichero Dockerfile anterior (se entiende que está en la carpeta db) en relación al docker-compose.yml

        - El contenedor creado llevara por nombre **base_datos**
        - Este servicio se reiniciará automáticamente
        - Llevará un volumen de Docker de nombre **datos_db** vinvulado al directorio **/var/lib/mysql** del contenedor que permitirá la persistencia de los datos generados en Mysql
        - Como variables de entorno se establecerá un valor para **MYSQL_ROOT_PASSWORD**
        - No es necesario exponer ningún puerto de la base de datos

    - Servicio backend php:

        - El servicio se llamará **back**
        - Indicaremos con **depends_on** que se debe arrancar primero el servicio db anterior
        - Crearemos un fichero Dockerfile para personalizar la imagen con nuestros datos. Se muestra ejemplo:



En este ejemplo se descarga la imagen de php 8.2, se instalan extensiones de php pdo y mysqli, se activa el modulo rewrite y se añade el contenido de la carpeta src al servidor Apache (se entiende que los scripts php deben ser copiados previamente a dicha carpeta src)

- Como en el caso anterior, construiremos la imagen nueva con la sintaxis:

  `build: .\php`

  Entendiendo que el fichero Dockerfile está ubicado dentro de la carpeta php

- Expondremos el puerto 80 del contenedor sobre el puerto 8080 de la maquina Windows (este puerto 8080 puede cambiarse si es necesario)

- Servicio phpmyadmin:

  - Emplearemos la imagen **phpmyadmin**
  - Usaremos la variable de entorno **PMA_HOST** para seleccionar el servicio donde está la base de datos
  - Usaremos **depends_on** para que este servicio arranque después del servicio de base de datos
  - Expondremos el puerto 80 del servicio Apache con el puerto 9000 de la maquina Windows (este puerto podemos modificarlo si es necesario)

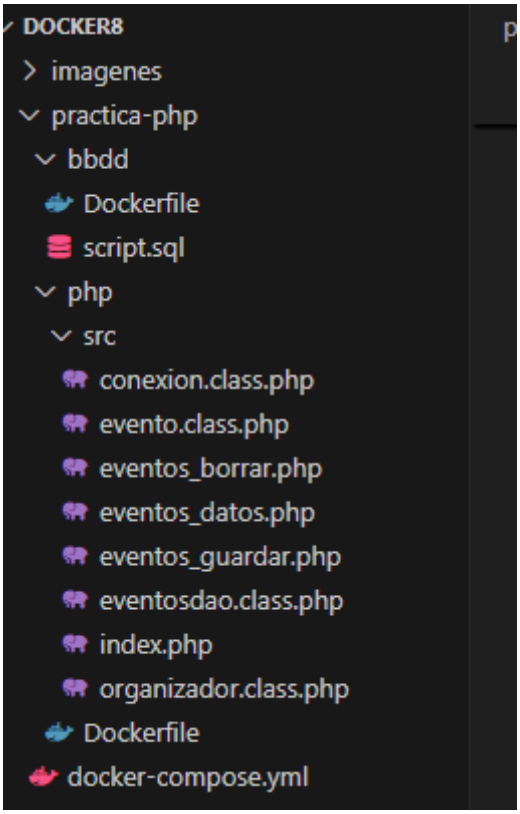
2. Se debe desplegar la aplicación web con el comando

   `docker-compose up -d`

y verificamos que la aplicación PHP es accesible y funciona correctamente, así como probar el acceso a la base de datos mediante la aplicación phpmyadmin

1.- Como en la actividad anterior ya levantamos una aplicación de php, voy a seguir la misma estructura, pero modificando el Dockerfile y el docker-compose.

La estructura es la siguiente:



El Dockerfile es el siguiente:

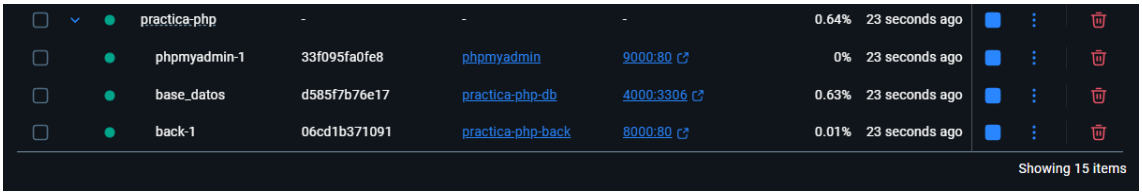
```
FROM php:8.4-apache
RUN docker-php-ext-install pdo pdo_mysql
#ADD ./src /var/www/html

RUN apt update
RUN apt install -y git
RUN git clone https://github.com/guillermosobrino01/DAW2B-DAW-Guillermo-Sobrino.git /var/www/html

EXPOSE 80

# Este puerto tiene que ser compatible con el del docker compose
```

El docker-compose es el siguiente:

practica-php > docker-compose.yml

```yaml
Run All Services
services:
  Run Service
  back:
    build: ./php
    ports:
      - 8000:80
      # Este puerto tiene que ser compatible con el del dockerfile (el 80)
    depends_on:
      - db
  Run Service
  db:
    build: ./bbdd
    container_name: base_datos
    ports:
      - 4000:3306
      # El puerto 3306 puede cambiar
    volumes:
      - datos_bd:/var/lib/mysql
    environment:
      - MYSQL_ROOT_PASSWORD=root
  Run Service
  phpmyadmin:
    image: phpmyadmin
    environment:
      - PMA_HOST=db
    ports:
      - 9000:80
    depends_on:
      - db
volumes:
  datos_bd:
```

Como se puede observar ya esta lanzado el docker:



Y funciona:

Inserte los datos del evento

Nombre del evento:*
Fecha:* dd/mm/aaaa
Ubicacion:*
Número de asistentes:*
Organizador: Eventos Corporativos S.L.
Guardar datos del evento
Mostrar los eventos guardados
Borrar eventos