Despliegue automatizado de WordPress con MariaDB

02/02/26
CURSO:2ºDAW

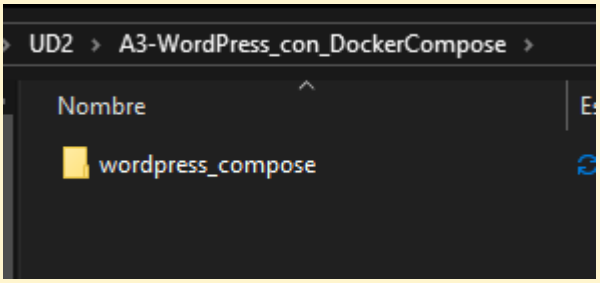
Despliegue
Celia Caravaca Vega

índice

Parte 1: Despliegue con volúmenes Docker

Tarea 1.1: Creación del archivo docker-compose.yml

1. **Crea un directorio: ~/wordpress_compose.**



2. **Investiga en la documentación de Docker Hub las imágenes de WordPress y MariaDB para identificar:**
   - **Variables de entorno necesarias para WordPress**

```
-e WORDPRESS_DB_HOST=mariadb:3306   (Host donde está la BD más el puerto)
-e WORDPRESS_DB_USER=wp_user        (Usuario de la BD)
-e WORDPRESS_DB_PASSWORD=wp_password(Contraseña del usuario)
-e WORDPRESS_DB_NAME=wordpress      (Nombre de la BD)
```

   - **Variables de entorno necesarias para MariaDB**

```
MARIADB_ROOT_PASSWORD=mi_password   (Contraseña del usuario)
MARIADB_DATABASE=wordpress          (Para crear la BD)
MARIADB_USER=wp_user                (Para nombre de usuario)
MARIADB_PASSWORD=wp_password        (Contraseña del usuario)
```

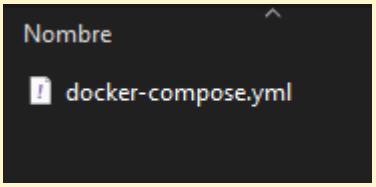
   - **Puertos que utilizan**
     MariaDB 3306
     WordPress 80

   - **Directorios para persistencia de datos**

     WordPress /var/www/html

     MariaDB /var/lib/mysql

3. **Crea un archivo docker-compose.yml que defina:**
   Servicio de WordPress:



   - **Imagen: wordpress**
   - **Puerto 80 del host mapeado al puerto del contenedor**

- Variables de entorno necesarias para conexión a base de datos
- Volumen Docker para el contenido de WordPress (/var/www/html/wp-content)
- Dependencia del servicio de base de datos
- Política de reinicio automático

4. Servicio de MariaDB:
   - Imagen: mariadb
   - Variables de entorno para configuración inicial (base de datos, usuario, contraseñas)
   - Volumen Docker para los datos de la base de datos (/var/lib/mysql)
   - Política de reinicio automático

5. Volúmenes: Define dos volúmenes Docker (uno para WordPress, otro para MariaDB)

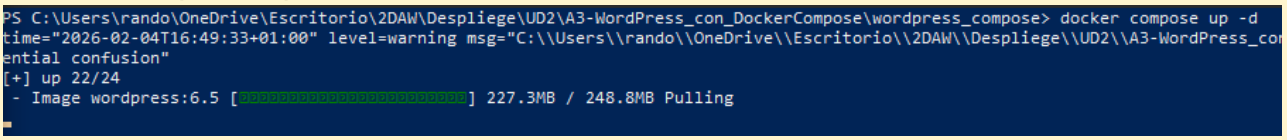
```yaml
version: "3.8"

services:
  wordpress:
    image: wordpress:6.0
    ports:
      - "80:80"
    environment:
      WORDPRESS_DB_HOST: mariadb:3306
      WORDPRESS_DB_USER: wp_user
      WORDPRESS_DB_PASSWORD: wp_password
      WORDPRESS_DB_NAME: wordpress
    volumes:
      - wordpress_data:/var/www/html/wp-content
    depends_on:
      - mariadb
    restart: always

  mariadb:
    image: mariadb:0.2
    environment:
      MARIADB_ROOT_PASSWORD: root_password
      MARIADB_DATABASE: wordpress
      MARIADB_USER: wp_user
      MARIADB_PASSWORD: wp_password
    volumes:
      - mariadb_data:/var/lib/mysql
    restart: always

volumes:
  wordpress_data:
```

```
  mariadb_data:
```

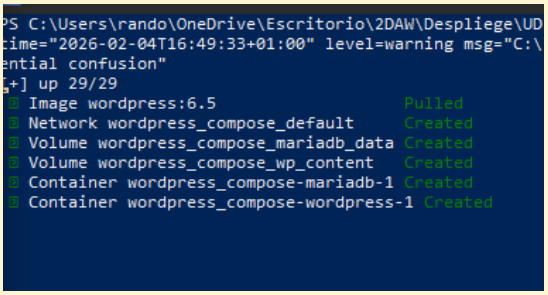
6. **Analiza y responde:**
   - **¿Por qué hay dos volúmenes diferentes?**
     Uno para los datos de wordpress como formularios, páginas…
     Y otro para los datos de la BD MariaDB
   - **¿Qué datos almacena cada volumen?**
     mariadb_data: almacena las bases de datos, tablas, registros.
     woedpress_data: almacena configuraciones, plugins, temas, archivos subidos por los usuarios.
   - **¿Por qué WordPress usa el nombre del servicio de base de datos como hostname?**
     Docker Compose crea una red interna, los servicios se comunican por el hostname y al poner el nombre de mariadb(docker) no necesitamos poner una IP fija.

Tarea 1.2: Despliegue y configuración

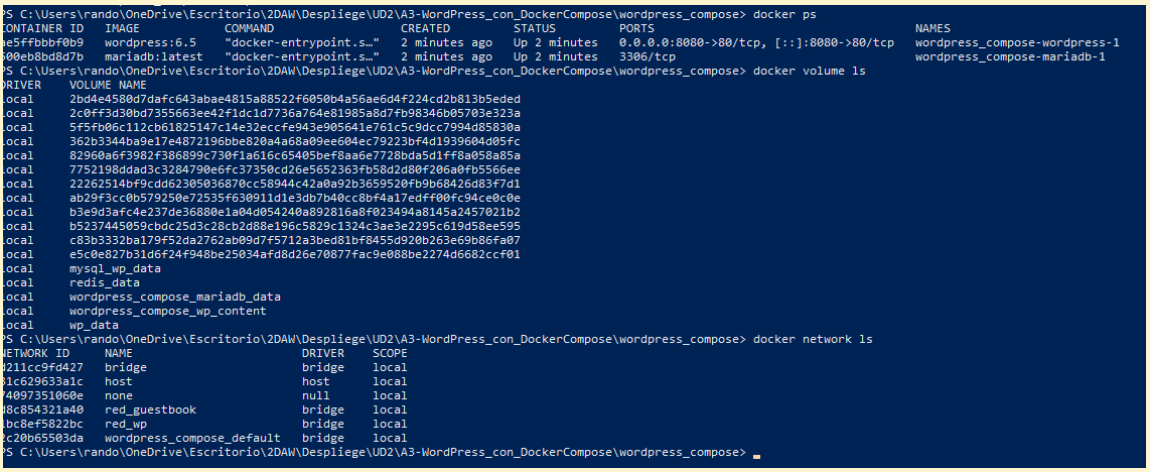
1. **Despliega el escenario con Docker Compose.**
   docker compose up -d



2. **Observa qué recursos se crean automáticamente.**



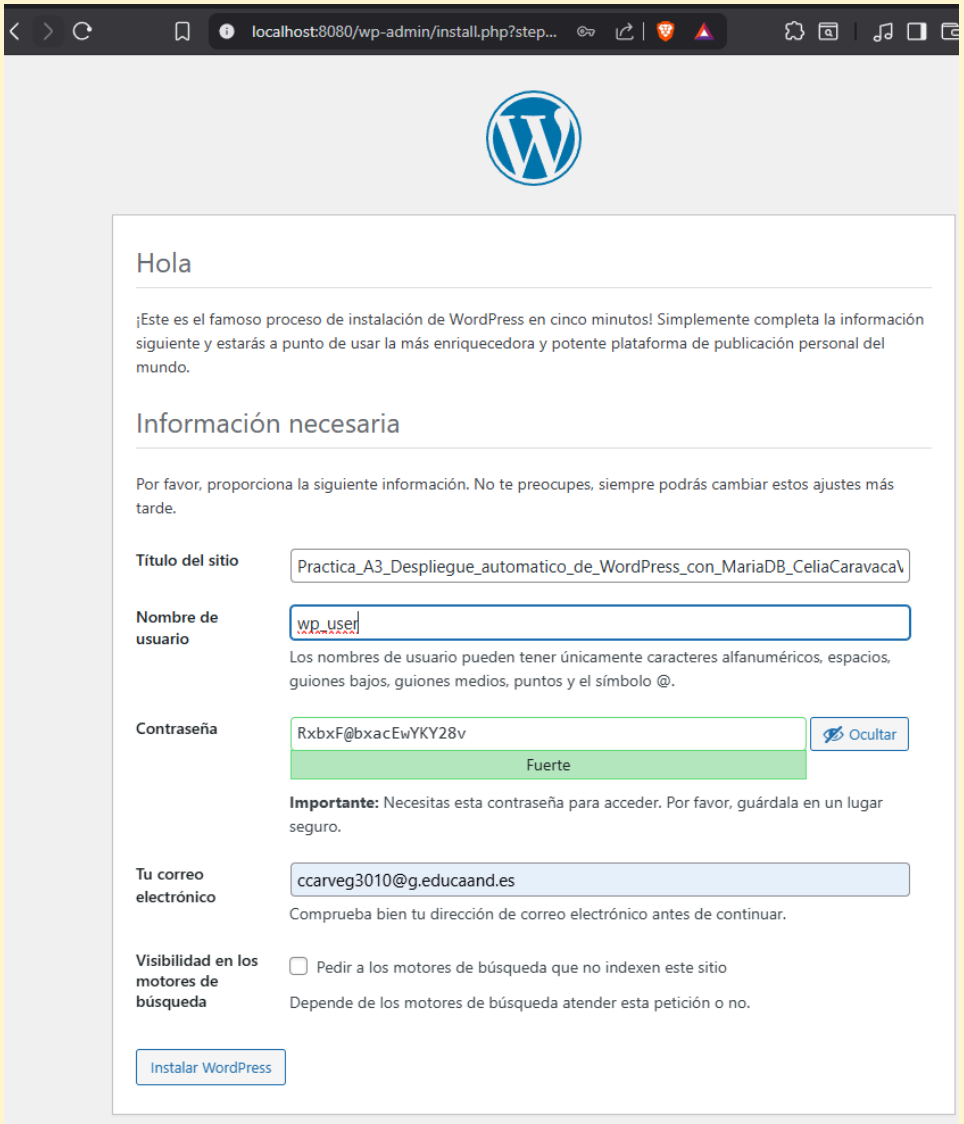
   docker ps
   docker volume ls
   docker network ls

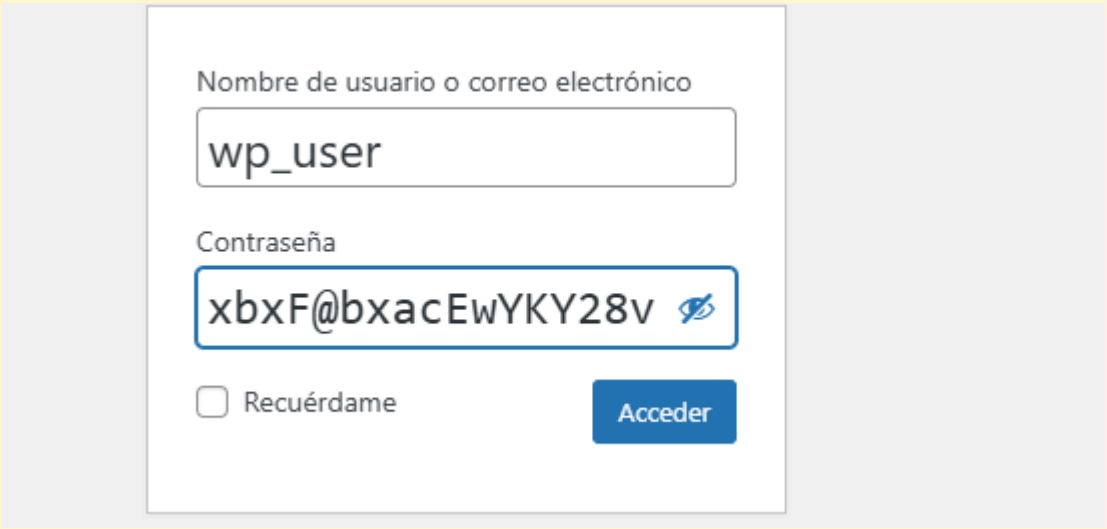

PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker ps
CONTAINER ID   IMAGE   COMMAND   CREATED   STATUS   PORTS   NAMES
ae5ffbbbf0b9   wordpress:6.5   "docker-entrypoint.s…"   2 minutes ago   Up 2 minutes   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   wordpress_compose-wordpress-1
600eb8bd8d7b   mariadb:latest "docker-entrypoint.s…"   2 minutes ago   Up 2 minutes   3306/tcp   wordpress_compose-mariadb-1
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker volume ls
DRIVER   VOLUME NAME
local   2bd4e4580d7dafc643abae4815a88522f6050b4a56ae6d4f224cd2b813b5eded
local   2c0ff3d30bd7355663ee42f1dc1d7736a764e81985a8d7fb98346b05703e323a
local   5f5fb06c112cb61825147c14e32eccfe943e905641e761c5c9dcc7994d85830a
local   362b3344ba9e17e4872196bbe820a4a68a09ee604ec79223bf4d1939604d05fc
local   82960a6f3982f386899c730f1a616c65405bef8aa6e7728bda5d1ff8a058a85a
local   7752198ddad3c3284790e6fc37350cd26e5652363fb58d2d80f206a0fb5566ee
local   22262514bf9cdd62305036870cc58944c42a0a92b3659520fb9b68426d83f7d1
local   ab29f3cc0b579250e72535f630911d1e3db7b40cc8bf4a17edff00fc94ce0c0e
local   b3e9d3afc4e237de36880e1a04d054240a892816a8f023494a8145a2457021b2
local   b5237445059cbdc25d3c28cb2d88e196c5829c1324c3ae3e2295c619d58ee595
local   c83b3332ba179f52da2762ab09d7f5712a3bed81bf8455d920b263e69b86fa07
local   e5c0e827b31d6f24f948be25034afd8d26e70877fac9e088be2274d6682ccf01
local   mysql_wp_data
local   redis_data
local   wordpress_compose_mariadb_data
local   wordpress_compose_wp_content
local   wp_data
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker network ls
NETWORK ID   NAME   DRIVER   SCOPE
4211cc9fd427   bridge   bridge   local
b1c629633a1c   host   host   local
7409735 1060e   none   null   local
48c854321a40   red_guestbook   bridge   local
1bc8ef5822bc   red_wp   bridge   local
2c20b65503da   wordpress_compose_default bridge   local
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>

3. **Verifica el estado de los servicios.**
   docker compose ps

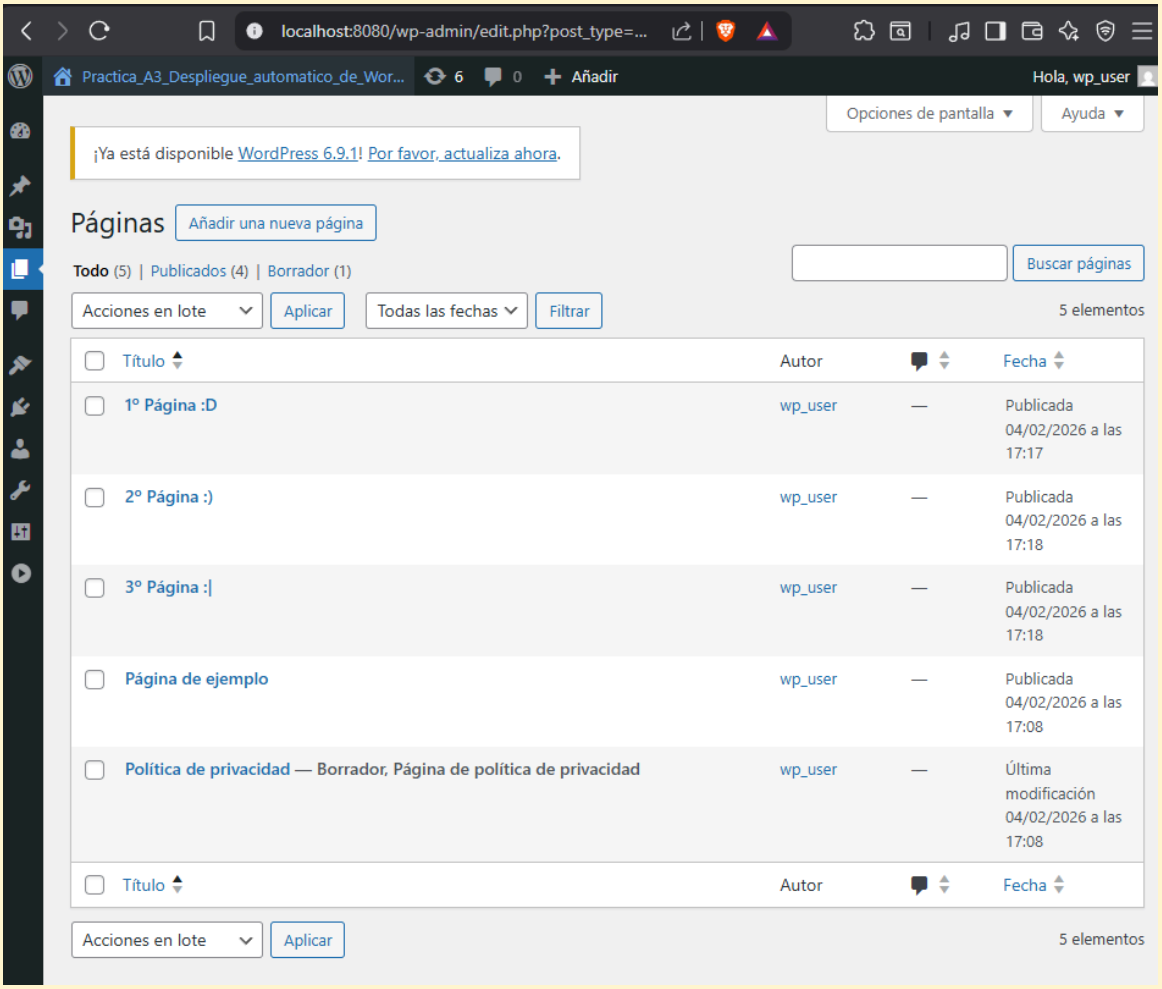
```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose ps
time="2026-02-04T16:59:45+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPres
ential confusion"
NAME                              IMAGE              COMMAND                  SERVICE     CREATED         STATUS         PORTS
wordpress_compose-mariadb-1       mariadb:latest     "docker-entrypoint.s…"   mariadb     8 minutes ago   Up 8 minutes   3306/tcp
wordpress_compose-wordpress-1     wordpress:6.5      "docker-entrypoint.s…"   wordpress   8 minutes ago   Up 8 minutes   0.0.0.0:
```

4. **Accede a WordPress (http://localhost) y completa la instalación:**
   - **Título del sitio**
   - **Usuario administrador**
   - **Contraseña**
   - **Email**

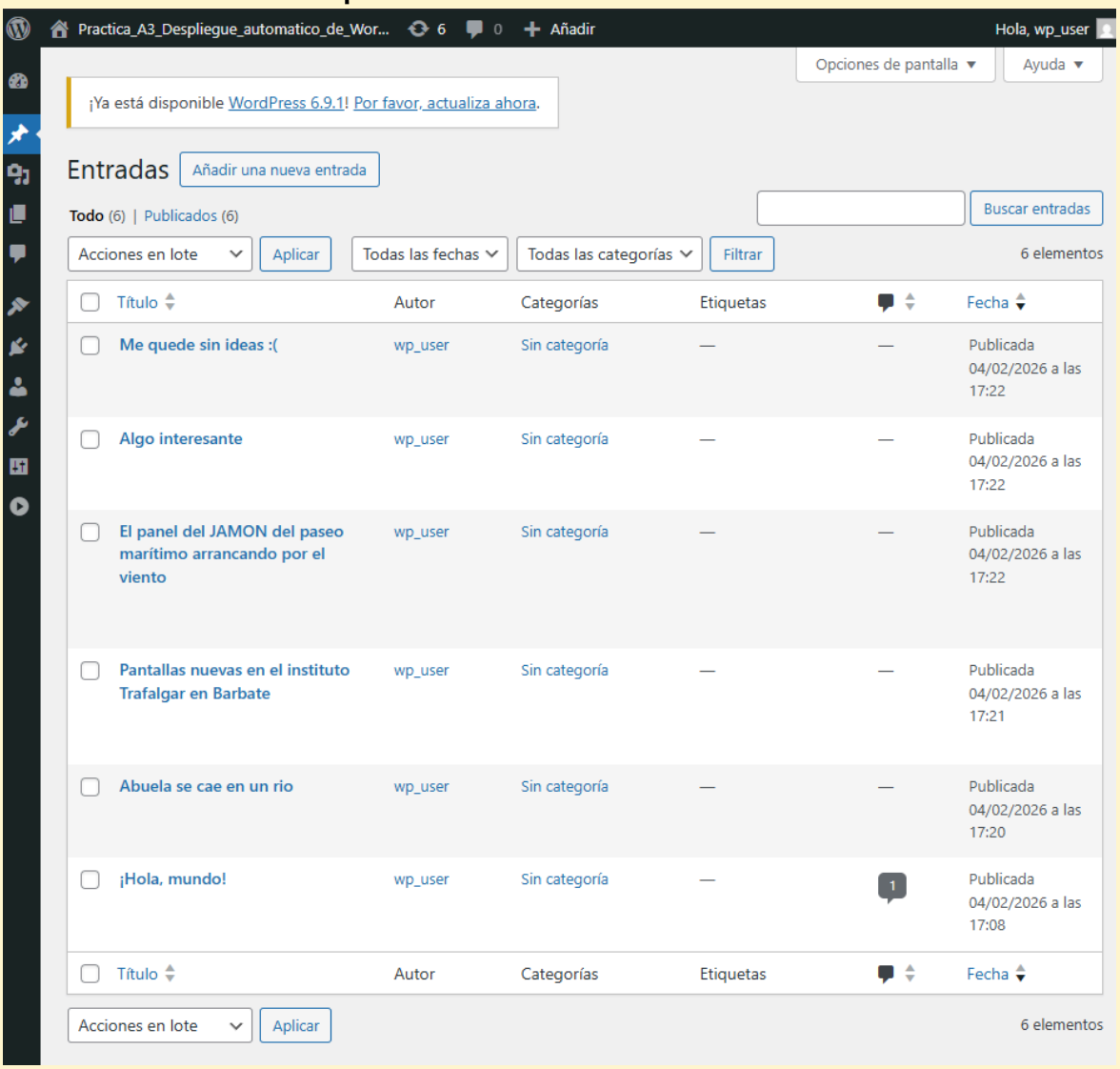




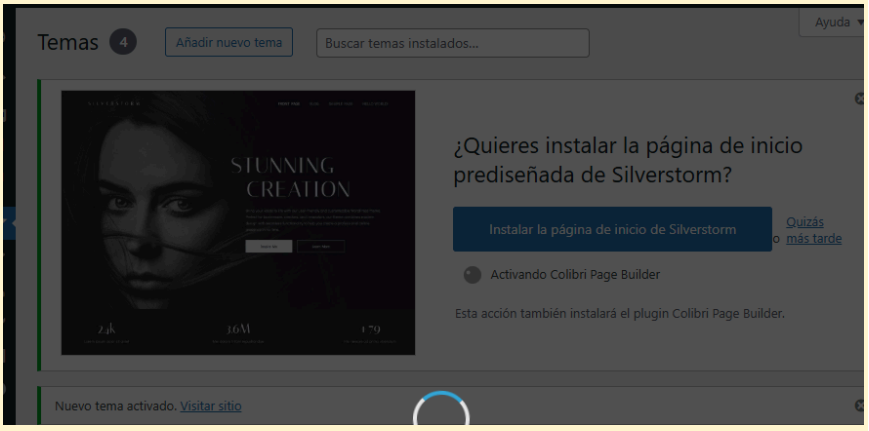
5. **Crea contenido:**
    - **Al menos 3 páginas**



- **Al menos 5 posts/entradas**



- **Instala y activa un tema**



- **Instala al menos 2 plugins**

Tarea 1.3: Gestión del escenario

1. **Detener servicios:**
   - **Detén ambos servicios**
     (hice down en vez de stop en resumen me los he cargado un momento)

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose down
time="2026-02-04T17:31:00+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_
ential confusion"
[+] down 3/3
 Container wordpress_compose-wordpress-1 Removed
 Container wordpress_compose-mariadb-1   Removed
 Network wordpress_compose_default       Removed
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>
```

Lo volvi a levantar ya que no se borraron los volúmenes por lo que tiene que estar bien.
(Tube un pequeño problema con los puertos por eso en mi yml puse el puerto

```
ports:
      - "8080:80"
```

para wordpress)
Al final lo volvi a hacer de 0 por que al volver a levantar los docker de maria y wordpress se pisaban los volúmenes.

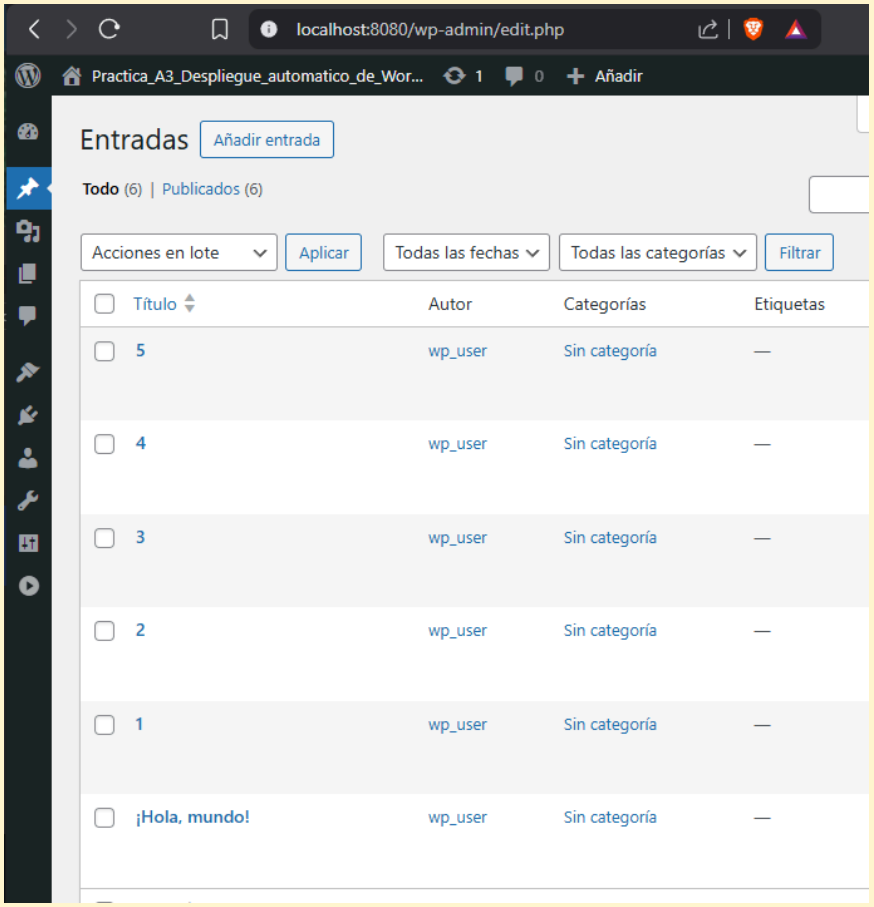
   - **Verifica que están detenidos pero no eliminados**

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose stop
time="2026-02-04T18:49:59+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_con_Doc
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] stop 2/2
 Container wordpress_compose-wordpress-1 Stopped
 Container wordpress_compose-mariadb-1   Stopped
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose ps
time="2026-02-04T18:50:24+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_con_Doc
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
NAME      IMAGE     COMMAND   SERVICE   CREATED   STATUS    PORTS
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose ps -a
time="2026-02-04T18:50:28+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_con_Doc
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
NAME                          IMAGE       COMMAND                  SERVICE     CREATED         STATUS                       PORTS
wordpress_compose-mariadb-1     mariadb     "docker-entrypoint.s…"   mariadb     4 minutes ago   Exited (137) 24 seconds ago
wordpress_compose-wordpress-1   wordpress   "docker-entrypoint.s…"   wordpress   4 minutes ago   Exited (0) 27 seconds ago
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>
```

2. **Reiniciar y verificar persistencia:**
   - **Arranca los servicios nuevamente**

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose sta
time="2026-02-04T18:51:23+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPres
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] start 2/2
 Container wordpress_compose-mariadb-1   Started
 Container wordpress_compose-wordpress-1 Started
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>
```

- Accede a WordPress
- Verifica que todo el contenido persiste



3. Eliminar contenedores (sin volúmenes):
   - Elimina el escenario manteniendo los volúmenes

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose down
time="2026-02-04T18:53:46+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_co
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] down 3/3
 ✔ Container wordpress_compose-wordpress-1 Removed
 ✔ Container wordpress_compose-mariadb-1   Removed
 ✔ Network wordpress_compose_default       Removed
```

(¬_¬) ...

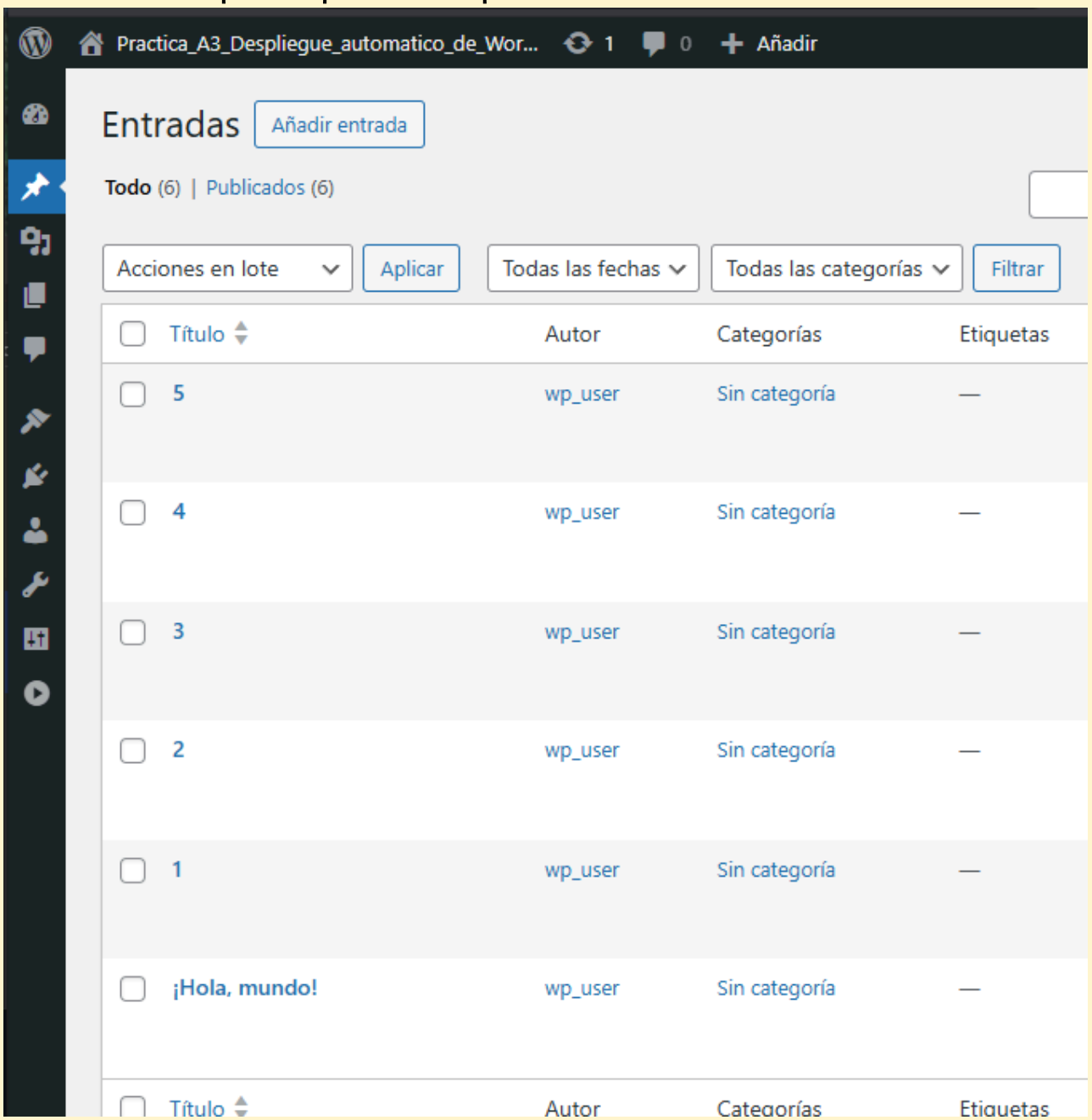
   - Verifica que los volúmenes siguen existiendo

```
local     mysql_wp_data
local     redis_data
local     wordpress_compose_mariadb_data
local     wordpress_compose_wordpress_data
local     wordpress_compose_wp_content
local     wp_data
```

   - Recrea el escenario

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose up -d
time="2026-02-04T18:54:59+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_co
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] up 3/3
 ✔ Network wordpress_compose_default       Created
 ✔ Container wordpress_compose-mariadb-1   Created
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>
```

- **Comprueba que los datos persisten**



(⊙_⊙)？(No se que hice antes que los datos no persistian)

4. **Eliminar todo (con volúmenes):**
   - **Elimina el escenario incluyendo volúmenes**
     docker compose down -v

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose down -v
time="2026-02-04T18:58:32+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_con
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] down 5/5
 ✔ Container wordpress_compose-wordpress-1 Removed
 ✔ Container wordpress_compose-mariadb-1   Removed
 ✔ Network wordpress_compose_default       Removed
 ✔ Volume wordpress_compose_mariadb_data   Removed
 ✔ Volume wordpress_compose_wordpress_data Removed
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>
```

- **Verifica que los volúmenes se han eliminado**
  docker volume ls

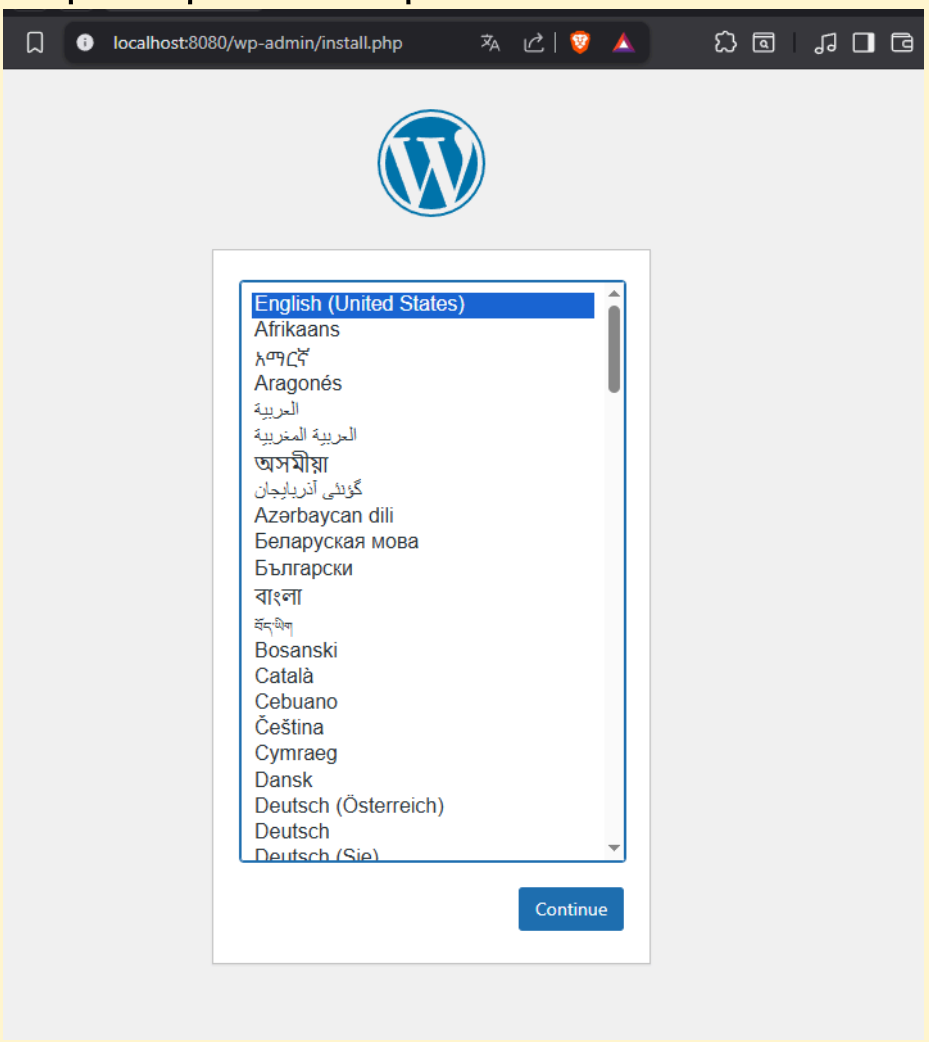
```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker volume ls
DRIVER    VOLUME NAME
local     2bd4e4580d7dafc643abae4815a88522f6050b4a56ae6d4f224cd2b813b5eded
local     2c0ff3d30bd7355663ee42f1dc1d7736a764e81985a8d7fb98346b05703e323a
local     5f5fb06c112cb61825147c14e32eccfe943e905641e761c5c9dcc7994d85830a
local     6ce108b301260d58118840a6ef3abf3be2db32c762a9ed0b06ba9e72c0da84d6
local     362b3344ba9e17e4872196bbe820a4a68a09ee604ec79223bf4d1939604d05fc
local     02283a1300db8b12f34cb73043e9da1e8cbc0319e34e577527e17a93efb6ea92
local     67802fb58bb02d9ec629f5ed3d2863c8c0d5ad84e1bc7f534a4cef907891a0aa
local     82960a6f3982f386899c730f1a616c65405bef8aa6e7728bda5d1ff8a058a85a
local     7752198ddad3c3284790e6fc37350cd26e5652363fb58d2d80f206a0fb5566ee
local     22262514bf9cdd62305036870cc58944c42a0a92b3659520fb9b68426d83f7d1
local     a7d6ace26a489cfb5ecfe8d3d7d5420a8f326e1d973aa644c08fcc0a72633d63
local     ab29f3cc0b579250e72535f630911d1e3db7b40cc8bf4a17edff00fc94ce0c0e
local     b3e9d3afc4e237de36880e1a04d054240a892816a8f023494a8145a2457021b2
local     b5237445059cbdc25d3c28cb2d88e196c5829c1324c3ae3e2295c619d58ee595
local     c83b3332ba179f52da2762ab09d7f5712a3bed81bf8455d920b263e69b86fa07
local     e5c0e827b31d6f24f948be25034afd8d26e70877fac9e088be2274d6682ccf01
local     e401be2462d9d54801f1ad654bc5747f481a066f0e8c4cd19ddff5faa8a9e47e
local     f7320bab46e2375632faa503d6eaff63a8951823b7a5647136201d1829e26a72
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>
```

Limpito

- **Recrea el escenario**
  docker compose up -d

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose> docker compose up -d
time="2026-02-04T19:02:24+01:00" level=warning msg="C:\\Users\\rando\\OneDrive\\Escritorio\\2DAW\\Despliege\\UD2\\A3-WordPress_con_DockerCompose\\wordpress_compose\\docker-
compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] up 5/5
 ✔ Network wordpress_compose_default       Created                                                                                                                0.2s
 ✔ Volume wordpress_compose_wordpress_data Created                                                                                                                0.1s
 ✔ Volume wordpress_compose_mariadb_data   Created                                                                                                                0.1s
 ✔ Container wordpress_compose-mariadb-1   Created                                                                                                                1.3s
 ✔ Container wordpress_compose-wordpress-1 Created                                                                                                                1.0s
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_compose>
```

- **Comprueba que WordPress pide instalación inicial**

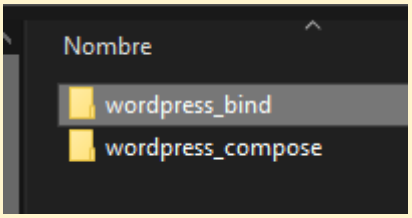


Hay que ser paciente a que cargue todo si no puede que me pase un buen tiempo buscando un error que no existe jeje.

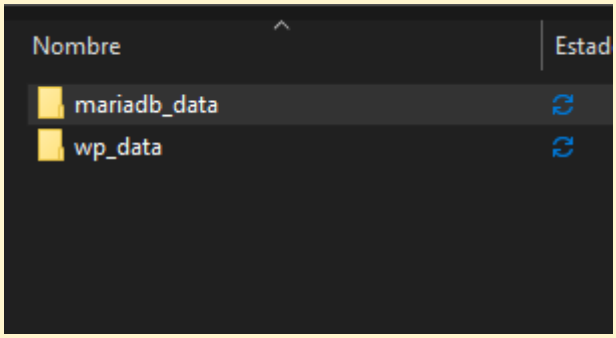
Parte 2: Despliegue con bind mounts

Tarea 2.1: Archivo docker-compose.yml con bind mount

1. Crea un nuevo directorio: ~/wordpress_bind.



2. Crea los directorios necesarios para los bind mounts:
   - Directorio para datos de WordPress
   - Directorio para datos de MySQL/MariaDB



3. Crea un nuevo archivo docker-compose.yml que use bind mounts en lugar de volúmenes Docker:
   - Investiga la sintaxis para bind mounts en Docker Compose
   - Usa rutas relativas (./directorio) para montar desde el host
   - Mantén la misma estructura de servicios que en la Parte 1
   - Cambia solo la definición de volúmenes por bind mounts

```yaml
version: "3.9"

services:
  wordpress:
    image: wordpress:6.4.3-php8.3-apache
    ports:
      - "8080:80"
    environment:
      WORDPRESS_DB_HOST: mariadb:3306
      WORDPRESS_DB_USER: wp_user
      WORDPRESS_DB_PASSWORD: wp_password
      WORDPRESS_DB_NAME: wordpress
    volumes:
      - ./wordpress_bind/wp_content:/var/www/html/wp-content
    depends_on:
      - mariadb
    restart: always
```

```
mariadb:
  image: mariadb:10.11
  environment:
    MARIADB_ROOT_PASSWORD: root_password
    MARIADB_DATABASE: wordpress
    MARIADB_USER: wp_user
    MARIADB_PASSWORD: wp_password
  volumes:
    - ./wordpress_bind/mysql_data:/var/lib/mysql
  restart: always
```

4. **Despliega la aplicación y configura WordPress nuevamente.**

Tarea 2.2: Comparación de enfoques

1. **Explora los directorios wordpress y mysql en el host.**
2. **Identifica qué archivos hay en cada directorio.**
   En el de wp_data tenemos:
   wp_data/
   ├── themes/ (los temas descargados)
   ├── plugins/ (plugins descargados)
   ├── uploads/ (imagenes y medios)
   ├── index.php (index por defecto)
   En mariaDb
   mariadb_data/
   ├── wordpress/ (Tablas de bbdd de wp)
   ├── mysql/ (usuarios y permisos)
   ├── performance_schema/
   ├── ibdata1 (archivos internos de InnoDB)
   ├── ib_logfile0 (archivos internos de InnoDB)

3. **Compara con los volúmenes Docker:**

Aspecto	Volúmenes Docker	Bind Mounts
Ubicación	Están dentro de una VM Linux (WSL2)	Directorio del sistema de archivos del host
Visibilidad desde host	Directorio del sistema de archivos del host	Alta, accesible directamente
Portabilidad	Alta (independiente del host)	Media (depende de rutas del host)

Backups	Media (depende de rutas del host)	Simple(copias la carpeta)
Permisos	Gestionados por Docker	Dependientes del sistema operativo

Parte 3: Configuración avanzada

Tarea 3.1: Variables de entorno desde archivo

1. Investiga cómo usar archivos .env con Docker Compose.
   Este archivo .env contiene las variables de entorno que se van a utilizar dentro de el yml solo se le indica la ruta del archivo env.
2. Crea un archivo .env que contenga todas las variables de configuración:
   - Contraseña root de MySQL
   - Nombre de la base de datos
   - Usuario de la base de datos
   - Contraseña del usuario
   - Variables correspondientes para WordPress
   - Puerto de WordPress

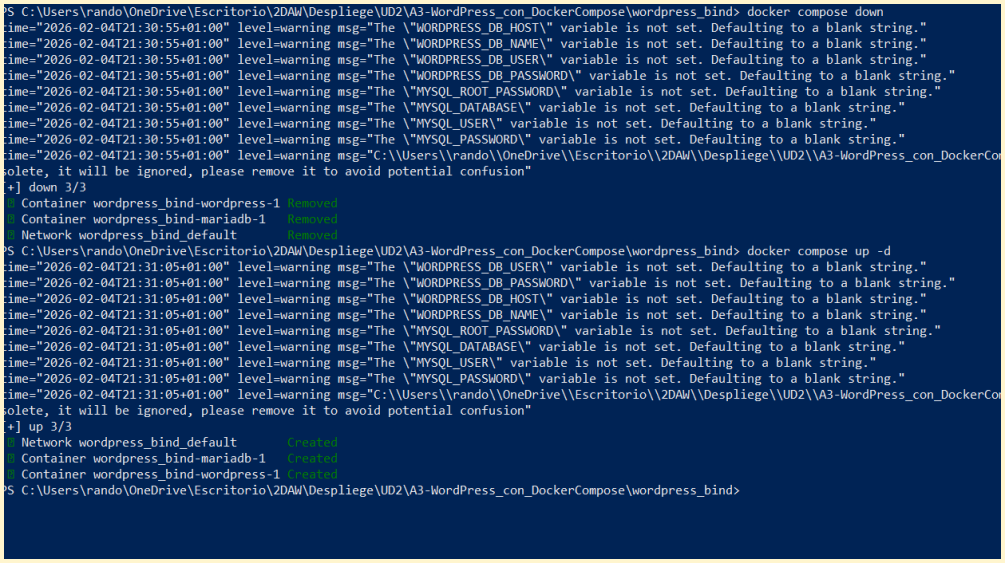
```
MYSQL_ROOT_PASSWORD=rootpass
MYSQL_DATABASE=wordpress
MYSQL_USER=wp_user
MYSQL_PASSWORD=wp_password

WORDPRESS_DB_HOST=mariadb
WORDPRESS_DB_NAME=wordpress
WORDPRESS_DB_USER=wp_user
WORDPRESS_DB_PASSWORD=wp_password

WORDPRESS_PORT=8080
```

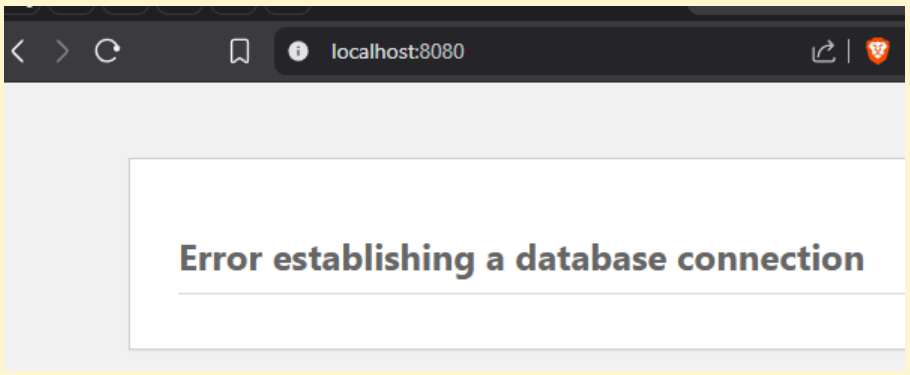
3. Modifica tu docker-compose.yml para usar variables del archivo .env con la sintaxis ${VARIABLE}.

```
version: "3.9"

services:
  wordpress:
    image: wordpress:6.4.3-php8.3-apache
    ports:
      - "8080:80"
    environment:
      WORDPRESS_DB_HOST: ${WORDPRESS_DB_HOST}
      WORDPRESS_DB_NAME: ${WORDPRESS_DB_NAME}
      WORDPRESS_DB_USER: ${WORDPRESS_DB_USER}
      WORDPRESS_DB_PASSWORD: ${WORDPRESS_DB_PASSWORD}
```

```
    volumes:
      - ./wp_data:/var/www/html/wp-content
    depends_on:
      - mariadb
    restart: always

  mariadb:
    image: mariadb:10.11
    environment:
      MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
      MYSQL_DATABASE: ${MYSQL_DATABASE}
      MYSQL_USER: ${MYSQL_USER}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD}

    volumes:
      - ./mariadb_data:/var/lib/mysql
    restart: always
```

4. **Despliega y verifica que funcione correctamente.**





(funciona lo que pasa es que tarda en arrancar)

5. **Reflexiona sobre las ventajas de seguridad de este enfoque.**
   El uso de archivos .env mejora la seguridad al evitar credenciales directamente en el docker-compose.yml y facilita la reutilización del archivo en distintos entornos.

Tarea 3.2: Configuración de red personalizada

1. **Investiga en la documentación cómo definir redes personalizadas en Docker Compose.**
   networks:
   wp_network:
   driver: bridge

2. **Modifica tu archivo docker-compose.yml para incluir:**
   - **Definición de una red personalizada tipo bridge**
   - **Conecta ambos servicios a esta red**
   - **Usa un nombre descriptivo para la red**

```yaml
version: "3.9"

services:
  wordpress:
    image: wordpress:6.4.3-php8.3-apache
    ports:
      - "8080:80"
    environment:
      WORDPRESS_DB_HOST: ${WORDPRESS_DB_HOST}
      WORDPRESS_DB_NAME: ${WORDPRESS_DB_NAME}
      WORDPRESS_DB_USER: ${WORDPRESS_DB_USER}
      WORDPRESS_DB_PASSWORD: ${WORDPRESS_DB_PASSWORD}

    volumes:
      - ./wp_data:/var/www/html/wp-content
    depends_on:
      - mariadb
    restart: always
  networks:
    - wp_network

  mariadb:
    image: mariadb:10.11
    environment:
      MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
      MYSQL_DATABASE: ${MYSQL_DATABASE}
      MYSQL_USER: ${MYSQL_USER}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD}

    volumes:
```

```
      - ./mariadb_data:/var/lib/mysql
    restart: always
    networks:
      - wp_network

networks:
  wp_network:
    driver: bridge
```

3. Verifica la configuración de red usando comandos de Docker Compose e inspección de red.

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind> docker network ls
NETWORK ID     NAME                       DRIVER    SCOPE
f475220f0373   bridge                     bridge    local
31c629633a1c   host                       host      local
74097351060e   none                       null      local
d8c854321a40   red_guestbook              bridge    local
301105d8e620   wordpress_bind_default      bridge    local
460189b1803e   wordpress_bind_wp_network   bridge    local
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind>
```

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind> docker network inspect wordpress_bind_wp_network
[
    {
        "Name": "wordpress_bind_wp_network",
        "Id": "460189b1803ef260db40f83bda54da26dc5c75d3dbd7c5c622352d924954b1d0",
        "Created": "2026-02-04T21:26:21.828887727Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv4": true,
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.20.0.0/16",
                    "IPRange": "",
                    "Gateway": "172.20.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Options": {
            "com.docker.network.enable_ipv4": "true",
            "com.docker.network.enable_ipv6": "false"
        },
        "Labels": {
            "com.docker.compose.config-hash": "568265de292dc0b7393a5ad996221f2a639db4239684db36dbd1f4312a55a1e2",
            "com.docker.compose.network": "wp_network",
            "com.docker.compose.project": "wordpress_bind",
            "com.docker.compose.version": "5.0.0"
        },
        "Containers": {
            "3f216e6b5f50dd6dc24eb529a96342fdf918f52bf86d4cfaf5e31e4b6ad09328": {
                "Name": "wordpress_bind-wordpress-1",
                "EndpointID": "b9de1349d9d495c194a5744e461ccad5b3228c1965b345a2e2d14e4051ff73b9",
                "MacAddress": "8e:c1:77:9d:9c:72",
                "IPv4Address": "172.20.0.3/16",
                "IPv6Address": ""
            },
            "5f6d34c2ac235fa5f4d8c59ea5ecf874c0a71d98351bcbf572569bf940cc3569": {
```

Tarea 3.3: Healthchecks y límites de recursos

1. Investiga la sintaxis de healthchecks y límites de recursos en Docker Compose.
2. Añade a tu archivo docker-compose.yml:
   Para WordPress:
   - Healthcheck que verifique la disponibilidad del puerto 80
   - Límites: 1 CPU y 512MB de memoria

- cpus: 1 para WordPress
- memoria: 512M para ambos
- test: curl -f http://localhost para WordPress
- interval: 30s
- timeout: 10s
- retries: 3

3. Para MariaDB:
   - Healthcheck que verifique la disponibilidad de MySQL
   - Límites: 0.5 CPU y 512MB de memoria
   - cpus: 0.5 para MariaDB
   - memoria: 512M para ambos
   - test: mysqladmin ping -h localhost para MariaDB
   - interval: 30s
   - timeout: 3s
   - retries: 3

```
services:
  wordpress:
    image: wordpress:6.4.3-php8.3-apache
    ports:
      - "${WORDPRESS_PORT}:80"
    environment:
      WORDPRESS_DB_HOST: ${WORDPRESS_DB_HOST}
      WORDPRESS_DB_NAME: ${WORDPRESS_DB_NAME}
      WORDPRESS_DB_USER: ${WORDPRESS_DB_USER}
      WORDPRESS_DB_PASSWORD: ${WORDPRESS_DB_PASSWORD}
    volumes:
      - ./wp_data:/var/www/html/wp-content
    depends_on:
      - mariadb
    networks:
      - wp_network
    restart: always

    deploy:
      resources:
        limits:
          cpus: "1"
          memory: 512M

    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost"]
      interval: 30s
      timeout: 10s
      retries: 3
```

```yaml
  mariadb:
    image: mariadb:10.11
    environment:
      MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
      MYSQL_DATABASE: ${MYSQL_DATABASE}
      MYSQL_USER: ${MYSQL_USER}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD}
    volumes:
      - ./mariadb_data:/var/lib/mysql
    networks:
      - wp_network
    restart: always

    deploy:
      resources:
        limits:
          cpus: "0.5"
          memory: 512M

    healthcheck:
      test: ["CMD", "mysqladmin", "ping", "-h", "localhost"]
      interval: 30s
      timeout: 3s
      retries: 3

networks:
  wp_network:
    driver: bridge
```

4. **Investiga qué comando usar para verificar el estado de salud de los servicios.**
   docker compose ps

```
NAME                        IMAGE                      COMMAND                  SERVICE     CREATED          STATUS
wordpress_bind-mariadb-1    mariadb:10.11              "docker-entrypoint.s…"   mariadb     51 seconds ago   Up 43 seconds (healthy)
wordpress_bind-wordpress-1  wordpress:6.4.3-php8.3-apache  "docker-entrypoint.s…"   wordpress   49 seconds ago   Up 42 seconds (health: starting)
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind>
```

5. **Verifica que los healthchecks funcionan correctamente.**

docker inspect <contenedor>

```
"Health": {
    "Status": "healthy",
    "FailingStreak": 0,
```

```
"Health": {
    "Status": "unhealthy",
    "FailingStreak": 8,
    "Log": [
```

(o no wordpress es estricto con el test)

Solución

```
    healthcheck:
      test: ["CMD-SHELL", "apachectl -t"]
      interval: 30s
      timeout: 10s
      retries: 3
```

y borrar y volver a up

```
  IMAGE                          COMMAND                  SERVICE     CREATED              STATUS
  mariadb:10.11                  "docker-entrypoint.s…"   mariadb     About a minute ago   Up About a minute (healthy)
  wordpress:6.4.3-php8.3-apache  "docker-entrypoint.s…"   wordpress   About a minute ago   Up About a minute (healthy)
\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind>
```

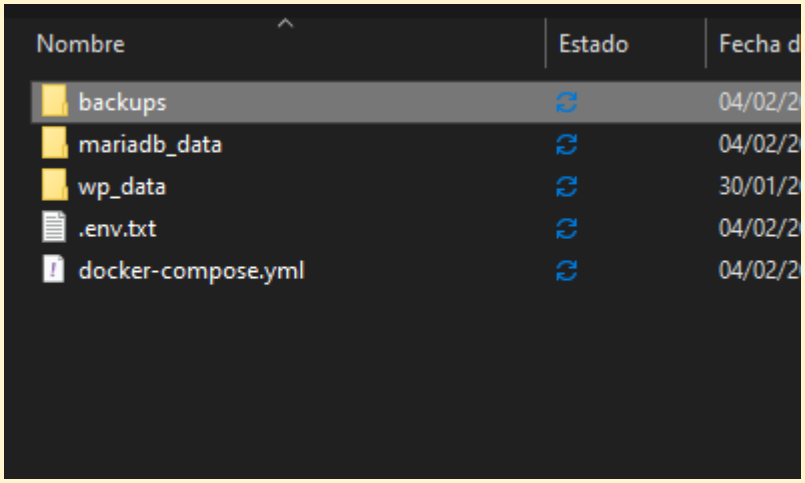
Ahora tiene buena pinta

```
"Health": {
    "Status": "healthy",
    "FailingStreak": 0,
    "Log": [
        {
            "Start": "2026-02-0
            "End": "2026-02-04T
```

Parte 4: Backup y restauración

Tarea 4.1: Backup con volúmenes Docker

1. Investiga estrategias para realizar backups de volúmenes Docker.
2. Crea un directorio para backups.



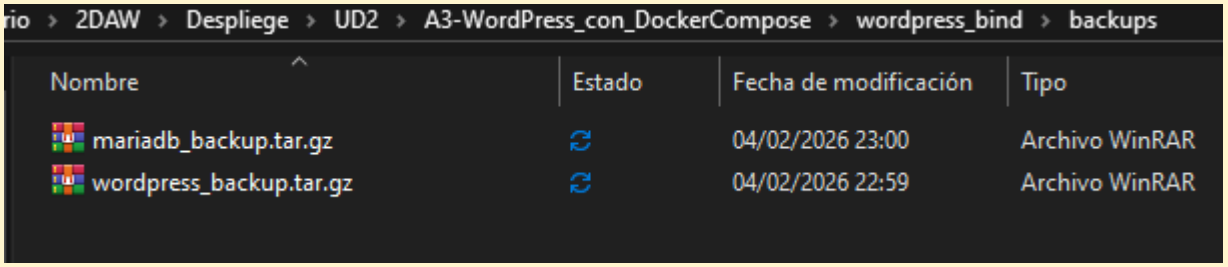
3. **Investiga y ejecuta comandos para:**
    - **Crear un backup del volumen de WordPress usando un contenedor temporal**
      docker run --rm -v wordpress_data:/data -v ${PWD}/backups:/backup alpine tar czf /backup/wordpress_backup.tar.gz -C /data .

    - **Crear un backup del volumen de MariaDB usando un contenedor temporal**
      docker run --rm -v mariadb_data:/data -v ${PWD}/backups:/backup alpine tar czf /backup/mariadb_backup.tar.gz -C /data .
    - **Comprimir los datos en archivos tar.gz**
    - **Almacenar los backups en el directorio del host**

4. **Verifica que los archivos de backup se han creado correctamente.**

Pista: Necesitarás usar contenedores temporales que monten el volumen y un directorio de backup.

```
PS C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind> ls backups


    Directorio: C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind\backups


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        04/02/2026     23:00             87 mariadb_backup.tar.gz
-a----        04/02/2026     22:59             87 wordpress_backup.tar.gz
```



Tarea 4.2: Restauración desde backup

1. **Elimina el escenario completo incluyendo los volúmenes.**
   docker compose down -v
2. **Vuelve a crear el escenario (se crearán volúmenes vacíos).**
   docker compose up -d
3. **Investiga y ejecuta comandos para restaurar los datos desde los backups:**
    - **Usa contenedores temporales para descomprimir y restaurar datos**

    - **Restaura el volumen de WordPress**
      docker run --rm -v wordpress_data:/data -v ${PWD}/backups:/backup alpine tar xzf /backup/wordpress_backup.tar.gz -C /data

    - **Restaura el volumen de MariaDB**
      docker run --rm -v mariadb_data:/data -v ${PWD}/backups:/backup alpine tar xzf /backup/mariadb_backup.tar.gz -C /data

4. **Arranca el escenario con Docker Compose.**
   docker compose up -d

5. **Verifica que todos los datos se han restaurado correctamente (contenido, configuración, base de datos).**

```
+] down 3/3
 Container wordpress_bind-wordpress-1 Removed
 Container wordpress_bind-mariadb-1   Removed
 Network wordpress_bind_wp_network    Removed
S C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind> docker compose up -d
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"WORDPRESS_PORT\" variable is not set. Defaulting to a blank string."
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"WORDPRESS_DB_HOST\" variable is not set. Defaulting to a blank string.
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"WORDPRESS_DB_NAME\" variable is not set. Defaulting to a blank string.
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"WORDPRESS_DB_USER\" variable is not set. Defaulting to a blank string.
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"WORDPRESS_DB_PASSWORD\" variable is not set. Defaulting to a blank str
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"MYSQL_DATABASE\" variable is not set. Defaulting to a blank string."
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"MYSQL_USER\" variable is not set. Defaulting to a blank string."
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"MYSQL_PASSWORD\" variable is not set. Defaulting to a blank string."
ime="2026-02-04T23:01:59+01:00" level=warning msg="The \"MYSQL_ROOT_PASSWORD\" variable is not set. Defaulting to a blank strin
+] up 3/3
 Network wordpress_bind_wp_network    Created
 Container wordpress_bind-mariadb-1   Created
 Container wordpress_bind-wordpress-1 Created
S C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind> docker run --rm   -v wor
ackup.tar.gz -C /data
S C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind> docker run --rm   -v mar
p.tar.gz -C /data
S C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind>
S C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind> docker compose up -d
ime="2026-02-04T23:04:19+01:00" level=warning msg="The \"WORDPRESS_DB_HOST\" variable is not set. Defaulting to a blank string.
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"WORDPRESS_DB_NAME\" variable is not set. Defaulting to a blank string.
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"WORDPRESS_DB_USER\" variable is not set. Defaulting to a blank string.
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"WORDPRESS_DB_PASSWORD\" variable is not set. Defaulting to a blank str
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"WORDPRESS_PORT\" variable is not set. Defaulting to a blank string."
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"MYSQL_ROOT_PASSWORD\" variable is not set. Defaulting to a blank strin
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"MYSQL_DATABASE\" variable is not set. Defaulting to a blank string."
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"MYSQL_USER\" variable is not set. Defaulting to a blank string."
ime="2026-02-04T23:04:20+01:00" level=warning msg="The \"MYSQL_PASSWORD\" variable is not set. Defaulting to a blank string."
+] up 2/2
 Container wordpress_bind-mariadb-1   Running
 Container wordpress_bind-wordpress-1 Running
S C:\Users\rando\OneDrive\Escritorio\2DAW\Despliege\UD2\A3-WordPress_con_DockerCompose\wordpress_bind>
```

Reflexión: ¿Por qué es importante probar regularmente los procesos de restauración?
Es importante probar las restauraciones porque un backup que no se ha probado puede no ser válido cuando se necesita.

Tarea 4.3: Backup con bind mounts

1. **Para el escenario con bind mounts, el backup es más directo.**
   cp -r wp_data backups/wp_data_backup
   cp -r mariadb_data backups/mariadb_data_backup

2. **Investiga qué comandos del sistema operativo puedes usar para:**
   - **Crear copias de seguridad de directorios completos**
   - **Comprimir los directorios en archivos tar.gz**
   - **Verificar la integridad de los backups**
3. **Realiza backups de ambos directorios (wordpress y mysql).**
   tar czf wordpress_bind_backup.tar.gz wp_data
   tar czf mariadb_bind_backup.tar.gz mariadb_data

4. **Compara las ventajas y desventajas del backup con bind mounts vs. volúmenes Docker:**

Aspecto	Volúmenes Docker	Bind Mounts
Facilidad	Media	Alta
Comandos Docker	Sí	No
Visibilidad	Baja	Alta
Portabilidad	Alta	Media

Parte 5: Análisis y documentación

Tarea 5.1: Preguntas de análisis

Responde en tu documentación:

1. **Volúmenes vs. bind mounts:**
   - **¿Cuándo preferirías volúmenes Docker?**
     En producción para que Docker gestione la persistencia
     Cuando quieres portabilidad y seguridad de datos
     Cuando no quieres preocuparte de rutas en el host
   - **¿Cuándo preferirías bind mounts?**
     Desarrollo local: quieres ver y editar archivos directamente
     Testing: fácil copiar, versionar y compartir directorios
     Debugging: inspeccionar logs y configuraciones
   - **¿Cuál es más fácil para backups?**
     Bind mounts: copias con comandos del SO
     Volúmenes Docker: requiere contenedores temporales
2. **Seguridad:**
   - **¿Es seguro tener las contraseñas en el archivo Compose?**
     No, cualquiera que tenga acceso al proyecto podrá verlas.
   - **¿Cómo mejorarías la seguridad usando .env?**
     Guardar contraseñas en .env
     Añadir .env a .gitignore
     Evitar subirlo a repositorios
   - **¿Qué otras medidas de seguridad aplicarías?**
     Usar usuarios limitados de DB, no root
     Limitar puertos expuestos
     Actualizar imágenes y contenedores
3. **Persistencia:**
   - **¿Qué pasaría si pierdes el volumen de WordPress?**
     Perderías contenido (themes, plugins, uploads)

- **¿Y si pierdes el volumen de MariaDB?**
  Perderías toda la base de datos: usuarios, posts, settings
- **¿Cuál es más crítico?**
  MariaDB, porque sin ella WordPress no puede funcionar

4. **Dependencias:**
   - **¿Por qué WordPress depende de la base de datos?**
     Toda la información dinámica (posts, usuarios, configuraciones) se guarda allí.
   - **¿Qué pasa si intentas iniciar WordPress sin MariaDB?**
     WordPress no puede arrancar y mostrará error “Error establishing a database connection”(o una pantalla blanca)
   - **¿depends_on garantiza que MariaDB esté lista?**
     Solo garantiza que el contenedor esté iniciado.
     No espera a que la DB acepte conexiones (por eso usamos healthchecks)
5. **Comparación con práctica anterior:**
   - **¿Cuántos comandos necesitabas en la Práctica 2.3?**
     Necesitabas varios docker run y docker exec para backup/restor
   - **¿Cuántos comandos necesitas con Docker Compose?**
     Solo docker compose up/down y contenedores temporales para backup
   - **¿Qué es más fácil de mantener?**
     Docker Compose + healthchecks + bind mounts/volúmenes
     es mucho más limpio

Tarea 5.2: Escenarios de uso

Describe cómo usarías cada enfoque en estos escenarios:

Escenario	Enfoque recomendado	Por qué
Desarrollo local	Bind mounts	Puedes editar archivos directamente y ver cambios inmediatos
Producción	Volúmenes Docker	Persistencia gestionada por Docker, más seguro y portable
Testing / CI	Bind mounts	Fácil de crear/descartar entornos, revisar logs y contenido