

IES FRANCISCO DE GOYA

GymTracker

Proyecto fin de Ciclo Formativo

Daniel Marín Gómez
Álvaro Gallego Yáñez
David Martín Díaz



Manual de despliegue con Docker

1. Instalación de docker.....	3
1.1.Para linux:.....	3
1.2.Para Windows:.....	3
2. Preparación para el despliegue.....	6
3. Despliegue.....	11

1. Instalación de docker

1.1. Para linux:

Para instalar docker en linux abrimos una terminal y ejecutamos el siguiente comando:

sudo apt install docker.io

Y para poder usar docker sin sudo metemos nuestro usuario en el grupo de docker con este comando:

sudo usermod -aG docker NuestroUsuario

Y posteriormente reiniciamos nuestra máquina para que se aplique el cambio de grupo a nuestro usuario:

sudo reboot

1.2. Para Windows:

Para instalar docker en windows nos ayudamos de este tutorial:

<https://youtu.be/ZO4KWQfUBBc?si=ii53FktLvmKTIXQ8>

Siguiendo el tutorial vamos a descargar docker con wsl (Windows Subsystems for Linux). Para ello nos ayudaremos de las documentaciones oficiales:

- documentación oficial de docker:
<https://docs.docker.com/desktop/install/windows-install/>
- documentación oficial de microsoft:
[Install WSL | Microsoft Learn](https://aka.ms/wsl-learn)

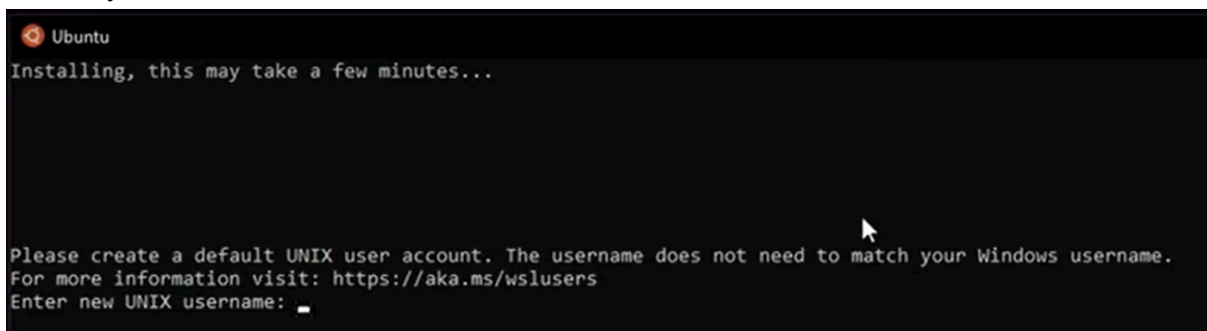
Comenzaremos por instalar WSL con el siguiente comando en una consola powershell:

wsl --install



```
PS C:\Users\damag> wsl --install
```

Reiniciamos y se nos abre la ventana de que se está instalando wsl y luego nos pide usuario y contraseña:

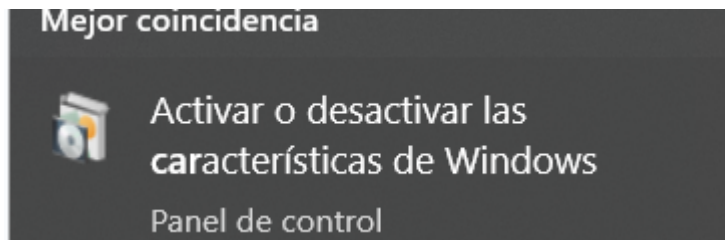


```
Ubuntu
Installing, this may take a few minutes...

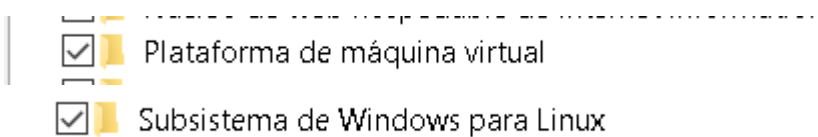
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: _
```

Rellenamos y ya tenemos wsl en nuestro windows.

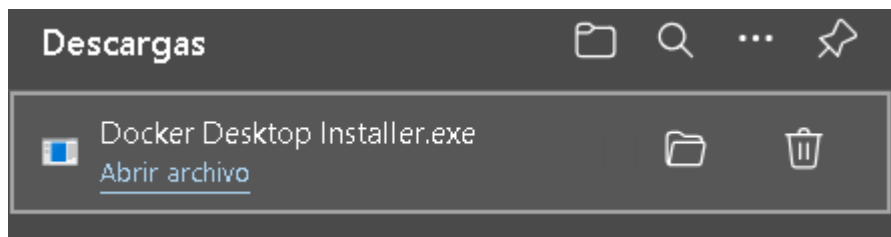
Después de esto seguimos los pasos de la documentación oficial y configuramos la virtualización. Para ello buscamos **features o características**:




Abrimos y buscamos las opciones de **virtual machine platform o plataforma de máquina virtual y Windows Subsystems for Linux o Subsistema de Windows para Linux** y activamos las opciones si no están activadas:



Después de esto ya podremos instalar docker desktop desde la página oficial descargamos el instalador:



Y lo ejecutamos:

 Installing Docker Desktop 4.30.0 (149282)

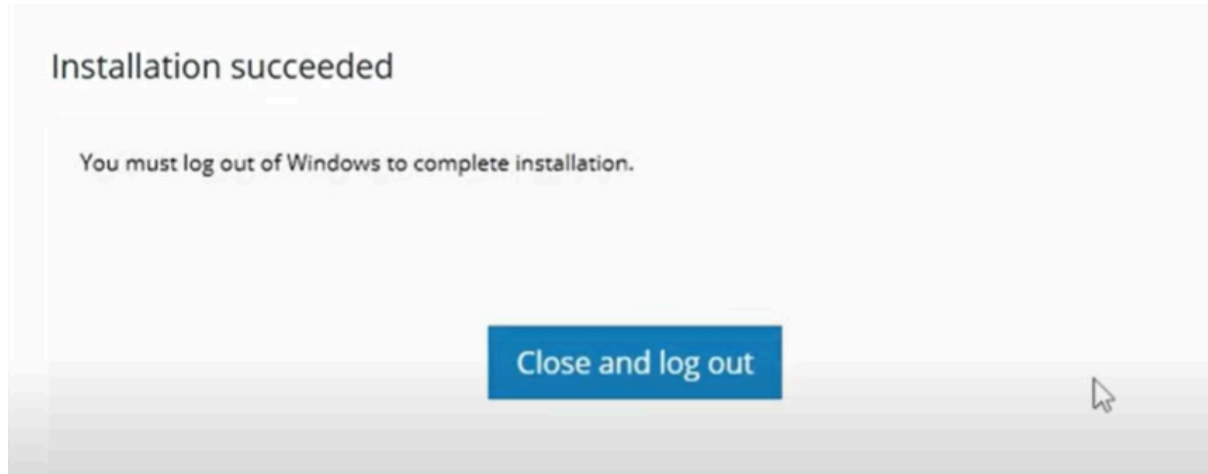
— □ ×

Docker Desktop 4.30.0

Unpacking files...

```
Unpacking file: frontend/libGLv2.dll
Unpacking file: frontend/libEGL.dll
Unpacking file: frontend/ffmpeg.dll
Unpacking file: frontend/Docker Desktop.exe
Unpacking file: frontend/d3dcompiler_47.dll
Unpacking file: DockerCli.exe
Unpacking file: Docker.Core.dll
Unpacking file: Docker.Backend.dll
Unpacking file: Docker Desktop.exe
Unpacking file: Docker Desktop Installer.exe
Unpacking file: courgette64.exe
Unpacking file: resources/wsl/wsl-data.tar
Unpacking file: resources/wsl/wsl-bootstrap.tar
Unpacking file: resources/wsl/docker-wsl-cli.iso.sha256
```

Nos saldrá esto al terminar:



Pulsamos en **Close and log out** y se reiniciará la máquina una vez se reinicie nos muestra los términos de docker desktop los aceptamos y ya tendremos docker desktop y podremos ejecutar comandos de docker en la terminal.

Para comprobarlo podemos poner el siguiente comando para ver la versión de docker que tenemos:

docker --version

```
C:\Users\damag>docker --version
Docker version 26.1.1, build 4cf5afa
```

2. Preparación para el despliegue

Para preparar el despliegue de nuestra aplicación en docker comenzamos por crear una carpeta para docker en la que pusimos lo siguiente:

- Un readme con comandos de interés de docker a la hora de desplegar llamado **readme_comandos_docker.md**.

Ejecutar Docker Compose para Generar o Iniciar los Contenedores

```
docker-compose -p contenedor-gymtracker up --build
```

Este comando genera los contenedores de GymTracker y los inicia. Si es la primera vez que se ejecuta este comando o si se realizaron cambios en la configuración, se reconstruirán las imágenes de Docker antes de iniciar los contenedores.

Parar Contenedores

Para detener los contenedores de GymTracker, presiona Ctrl + C en la terminal donde se están ejecutando.

Ver Contenedores

```
docker ps -a
```

Este comando muestra una lista de todos los contenedores, incluidos los que están detenidos.

Ver Volúmenes

```
docker volume ls
```

Este comando muestra una lista de todos los volúmenes Docker en el sistema.

Acceder a la Base de Datos del Contenedor

Para acceder a la base de datos del contenedor de la base de datos de GymTracker, primero accede al shell del contenedor y luego ingresa a MySQL:

```
docker exec -it contenedor-gymtracker_database_1 /bin/bash
mysql -u root -p
```

La contraseña predeterminada es "root_password".

Acceder a los Logs de los Contenedores de la app y la api

Puedes acceder a los logs de los contenedores de la aplicación y la API de GymTracker de la siguiente manera:

```
docker logs contenedor-gymtracker_app_1
docker logs contenedor-gymtracker_api_1
```

Borrar Contenedores de GymTracker

```
docker rm contenedor-gymtracker_database_1 contenedor-gymtracker_app_1 contenedor-gymtracker_api_1
```

Este comando elimina los contenedores específicos de GymTracker.

Borrar Volumen de GymTracker

```
docker volume rm contenedor-gymtracker_db_data
```

Este comando elimina el volumen llamado "contenedor-gymtracker_db_data", utilizado por la base de datos de GymTracker.

- **Los jars**, el jar de la api de ejercicios y el jar de la app de Gymtracker.
- **Los Dockerfile** para cada jar para crear una imagen a partir de la cuál crear el contenedor para la api y para la app.
 - **Dockerfile.api:**

```
1  # Dockerfile para la API
2  FROM openjdk:17-jdk-slim
3  COPY Api-Ejercicios-0.0.1-SNAPSHOT.jar /api/api.jar
4  WORKDIR /api
5  CMD ["java", "-jar", "api.jar"]
```

FROM openjdk:17-jdk-slim:

Esta línea especifica la imagen base que se utilizará para construir la nueva imagen. En este caso, se utiliza una imagen de OpenJDK 17 en su versión "slim", que es una versión más ligera y optimizada.

COPY Api-Ejercicios-0.0.1-SNAPSHOT.jar /api/api.jar:

Esta línea copia el archivo Api-Ejercicios-0.0.1-SNAPSHOT.jar desde el sistema de archivos del host (donde se encuentra el Dockerfile) a la ruta /api/api.jar dentro de la imagen Docker. Este archivo es el ejecutable de la API.

WORKDIR /api:

Esta línea establece el directorio de trabajo dentro del contenedor en /api. Esto significa que cualquier comando que se ejecute después de esta línea se ejecutará dentro de este directorio.

CMD ["java", "-jar", "api.jar"]:

Esta línea especifica el comando que se ejecutará cuando se inicie un contenedor basado en esta imagen. En este caso, se ejecuta el comando java -jar api.jar para iniciar la API.

- **Dockerfile.app:**

```
1  # Dockerfile para la aplicación
2  FROM openjdk:17-jdk-slim
3
4  # Copia el archivo JAR de la aplicación
5  COPY GymTracker-0.0.1-SNAPSHOT.jar /app/app.jar
6
7  # Copia el script wait-for-it
8  COPY wait-for-it.sh /app/wait-for-it.sh
9
10 # Para que el script wait-for-it.sh tenga permisos de ejecución
11 RUN chmod +x /app/wait-for-it.sh
12
13 # Establecer el directorio de trabajo
14 WORKDIR /app
15
16 # Comando para iniciar la aplicación, esperando a que la base de datos esté lista
17 CMD ["/app/wait-for-it.sh", "database:3306", "--", "java", "-jar", "app.jar"]
```

Este Dockerfile crea una imagen Docker para la aplicación Java GymTracker, asegurándose de que la aplicación espere hasta que la base de datos esté lista antes de iniciarse.

FROM openjdk:17-jdk-slim:

Usa una imagen base ligera de OpenJDK 17 para ejecutar la aplicación Java.

COPY GymTracker-0.0.1-SNAPSHOT.jar /app/app.jar:

Copia el archivo JAR de la aplicación al contenedor.

COPY wait-for-it.sh /app/wait-for-it.sh:

Copia el script wait-for-it.sh al contenedor para esperar a que la base de datos esté lista.

RUN chmod +x /app/wait-for-it.sh:

Hace que el script sea ejecutable.

WORKDIR /app:

Establece /app como el directorio de trabajo dentro del contenedor.

CMD ["/wait-for-it.sh", "database:3306", "--", "java", "-jar", "app.jar"]:

Usa wait-for-it.sh para esperar a que la base de datos esté disponible en database:3306 antes de iniciar la aplicación con java -jar app.jar.

Uso del script wait-for-it.sh

El script wait-for-it.sh asegura que la aplicación GymTracker no intente conectarse a la base de datos antes de que esta esté lista, evitando errores de conexión. Esto nos sirvió para que el Docker Compose pudiera ejecutar el contenedor de la base de datos y la aplicación simultáneamente.

- **Script wait-for-it:**

El script `wait-for-it` lo obtuvimos de GitHub, y puedes encontrarlo en el siguiente enlace: <https://github.com/vishnubob/wait-for-it>. Este script nos ayudó a solucionar el error que ocurría al hacer el `docker-compose up`, donde la aplicación no se iniciaba correctamente porque dependía de que la base de datos estuviera disponible primero. Utilizando `wait-for-it`, pudimos hacer que nuestra aplicación esperará hasta que la base de datos estuviera lista antes de intentar iniciarse, asegurando así un inicio ordenado y sin errores.

- Archivo para la inicialización de la base de datos **init.sql**.
- **docker-compose.yml** :

Este archivo docker-compose.yml lo hemos creado para desplegar la aplicación Gymtracker. Configura tres contenedores: una base de datos MySQL, la API de ejercicios y la aplicación principal. A continuación, una explicación breve de cada sección:

Versión:


```
version: '3.8'
```

- Especificamos la versión del formato de **docker-compose**.

Servicios

Base de datos (database)

```
services:
  database:
    image: mysql:8.0
    environment:
      MYSQL_ROOT_PASSWORD: root_password
      MYSQL_DATABASE: gymtracker
      MYSQL_USER: dad
      MYSQL_PASSWORD: padre
    ports:
      - "3307:3306"
    volumes:
      - db_data:/var/lib/mysql
      - ./init.sql:/docker-entrypoint-initdb.d/init.sql
    networks:
      - app-network
```

- Usamos MySQL 8.0 con configuraciones de entorno para la base de datos.
- Mapeamos el puerto 3306 del contenedor al 3307 del host.
- Definimos volúmenes para la persistencia de datos y la inicialización.
- Conectamos a la red app-network.

API (api)

```
api:
  build:
    context: .
    dockerfile: Dockerfile.api
  ports:
    - "8081:8081"
  networks:
    - app-network
```

- Construimos la imagen usando Dockerfile.api.
- Mapeamos el puerto 8081 del contenedor al 8081 del host.

- Conectamos a la red app-network.

Aplicación (app)

```
app:
  build:
    context: .
    dockerfile: Dockerfile.app
  depends_on:
    - database
    - api
  environment:
    SPRING_DATASOURCE_URL: jdbc:mysql://database:3306/gymtracker
    SPRING_DATASOURCE_USERNAME: dad
    SPRING_DATASOURCE_PASSWORD: padre
    SERVER_PORT: 443
  ports:
    - "443:443"
  networks:
    - app-network
```

- Construimos la imagen usando Dockerfile.app.
- Establecemos dependencias para que la aplicación espere a que database y api estén listas.
- Configuramos variables de entorno para la conexión a la base de datos.
- Mapeamos el puerto 443 del contenedor al 443 del host.
- Conectamos a la red app-network.

Redes y Volúmenes

```
networks:
  app-network:
    driver: bridge
volumes:
  db_data:
```

- Definimos una red app-network para la comunicación entre servicios.
- Definimos un volumen db_data para la persistencia de datos de la base de datos.

- Con este docker-compose.yml, aseguramos que la aplicación GymTracker espere a que la base de datos y la API estén listas antes de iniciar, previniendo errores de dependencias.

Una vez hemos hecho la carpeta ya está todo listo para el despliegue.

3. Despliegue

Tanto en linux como windows:

1. Entrar en el github de Gymtracker:
[deiivvv/GymTracker: TFG de Alvaro Gallego, Daniel Marin y David Martin \(github.com\)](https://github.com/deiivvv/GymTracker)
2. Localiza el archivo ZIP llamado despliegue_docker.zip en el repositorio y descargalo.
3. Descomprime el archivo ZIP en la ubicación deseada.
4. Abre la carpeta descomprimida y verás el archivo readme_comandos_docker.md en el que tienes los comandos para despliegue y mantenimiento del mismo.
5. Abre una terminal en la carpeta y ejecuta el siguiente comando:

docker-compose -p contenedor-gymtracker up --build

Y ya tendrás GymTracker listo para acceder en tu navegador desde la siguiente dirección:

https://localhost

