

📁 Exchange Project - Docker Setup

Guía para desplegar el stack completo (Backend + Frontend + Base de Datos) utilizando Docker Compose.

📁 Prerrequisitos

- [Docker](#) instalado y corriendo.
- [Git](#) instalado.

📁 Estructura del Proyecto

Antes de comenzar, asegúrate de que tu estructura de carpetas se vea así:

```
raiz/
├── back/           # Repositorio del Backend clonado
├── front/          # Repositorio del Frontend clonado
└── docker-compose.yml # Archivo de orquestación (se crea en el paso 3)
```

📁 Instalación y Configuración

1. Clonar Repositorios

Clona ambos proyectos dentro de la misma carpeta raíz:

```
# Ejemplo (sustituye con tus URLs reales)
git clone https://github.com/certi-devops-team/proyecto2-back ./back
git clone https://github.com/Andrezuu/proyecto-certi-react ./front
```

2. Configurar el Backend

Navega a la carpeta del backend y crea las variables de entorno necesarias.

1. Entra en la carpeta `back` :

```
cd back
```

2. Crea un archivo llamado `.env` y pega el siguiente contenido:

```
PORT=3000
NODE_ENV=development
DATABASE_URL=postgres://neondb_owner:[PASSWORD]@ep-jolly-haze-a46f3uhn-pooler.us-east-1.aws.neon.tech/neondb?sslmode=require
```

Nota: Solicitar la contraseña de la base de datos

3. Construcción individual:

```
docker build -t back:v1 .
```

3. Configurar el Frontend

1. Entra en la carpeta `front` :

```
cd ../front
```

2. Construcción individual con argumentos:

```
docker build -t front:v1 --build-arg VITE_API_URL=http://localhost:3000 .
```

4. Orquestación con Docker Compose

Vuelve a la carpeta raíz (donde están las carpetas `back` y `front`) y crea un archivo llamado `docker-compose.yml` .

Copia y pega el siguiente contenido. **Nota:** Se ha añadido el servicio `db` necesario para que `depends_on: - db` funcione correctamente.

```
version: '3.8'

services:
  backend:
    build: ./back
    ports:
      - "3000:3000"
    environment:
      - PORT=3000
    depends_on:
      - db
    networks:
      - exchange-network

  frontend:
    build:
      context: ./front
      args:
        - VITE_API_URL=http://localhost:3000
    ports:
      - "80:80"
    depends_on:
      - backend
    networks:
      - exchange-network

# Servicio de Base de Datos Local
db:
  image: postgres:15
  environment:
    POSTGRES_USER: postgres
    POSTGRES_PASSWORD: password
    POSTGRES_DB: exchange_db
  volumes:
    - postgres_data:/var/lib/postgresql/data
  networks:
    - exchange-network
  ports:
    - "5432:5432"

networks:
  exchange-network:
    driver: bridge

volumes:
  postgres_data:
```

► Ejecución

Para levantar todo el entorno, ejecuta el siguiente comando en la raíz:

```
docker compose up --build
```

Una vez que los contenedores estén corriendo, podrás acceder a:

- **Frontend:** <http://localhost:80> (o simplemente <http://localhost>)
- **Backend API:** <http://localhost:3000>
- **Base de Datos:** Puerto `5432`

Para detener los servicios, usa:

