

Backend (Node + Express + TypeScript + Postgres)

Este documento explica cómo instalar, configurar y levantar el **backend** en modo desarrollo y producción.

Requisitos

- Node.js 18+ y npm
- PostgreSQL 13+ (local o Docker)
- Git (opcional)

1. Configurar variables de entorno

Crea el archivo **backend/.env**. Si tu proyecto incluye **.env.example**, úsalo como base.

Ejemplo típico (ajusta valores según tu entorno):

```
PORT=3001

DB_HOST=localhost
DB_PORT=5432
DB_USER=postgres
DB_PASSWORD=postgres
DB_NAME=periferia_social

JWT_SECRET=change_me
CORS_ORIGIN=http://localhost:5173
```

2. Instalar dependencias

Desde la carpeta del backend:

```
cd backend
npm install
```

3. Levantar en desarrollo (recomendado)

El proyecto incluye el script **dev** (TypeScript con recarga en caliente).

```
npm run dev
```

La API quedará disponible en el puerto definido por **PORT**.

4. (Opcional) Cargar datos de prueba (seed)

Si tu prueba requiere datos iniciales, ejecuta:

```
npm run seed
```

Nota: el seed normalmente asume que la base de datos ya existe y es accesible.

5. Compilar y ejecutar (producción/local)

Para compilar TypeScript y ejecutar el build:

```
npm run build
npm start
```

6. Ejecutar pruebas (si aplica)

```
npm test
```

PostgreSQL rápido con Docker (opcional)

Si no tienes Postgres instalado, puedes levantarlo con Docker Compose en la raíz del repo:

```
version: "3.9"
services:
  db:
    image: postgres:15
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: postgres
      POSTGRES_DB: periferia_social
    ports:
      - "5432:5432"
    volumes:
      - pgdata:/var/lib/postgresql/data
volumes:
  pgdata:
```

Luego:

```
docker compose up -d
```

Problemas comunes

No conecta a BD: revisa credenciales, puerto, y que la BD exista.

CORS: valida **CORS_ORIGIN** contra la URL del frontend.

Puerto ocupado: cambia **PORT** o libera el proceso.

Scripts disponibles (package.json)

El backend define los scripts: **dev**, **build**, **start**, **seed**, **test**.