🚀 Guía de Instalación y Configuración Manual - Thunderrol

Esta guía te permitirá ejecutar la aplicación Thunderrol de forma manual sin Docker, configurando cada componente por separado.

📋 Prerrequisitos del Sistema

1. Software Necesario

- **Node.js** 18+ y **npm/yarn**
- **Python** 3.11+
- **PostgreSQL** 15+
- **Git**

2. Verificar Instalaciones

```
# Verificar versiones
node --version    # debe ser 18+
npm --version
python3 --version # debe ser 3.11+
psql --version    # debe ser 15+
```

🗄️ Parte 1: Configuración de Base de Datos PostgreSQL

1. Instalar PostgreSQL (si no está instalado)

```
# Ubuntu/Debian
sudo apt update
sudo apt install postgresql postgresql-contrib

# macOS con Homebrew
brew install postgresql
brew services start postgresql

# Verificar que PostgreSQL esté corriendo
sudo systemctl status postgresql  # Linux
brew services list | grep postgresql  # macOS
```

2. Crear Base de Datos y Usuario

```
# Conectarse como usuario postgres
sudo -u postgres psql

# Dentro del prompt de PostgreSQL, ejecutar:
CREATE DATABASE thunderrol;
CREATE USER thunderrol WITH ENCRYPTED PASSWORD 'thunderrol123';
GRANT ALL PRIVILEGES ON DATABASE thunderrol TO thunderrol;
ALTER USER thunderrol CREATEDB;  -- Para migraciones
\q  -- Salir
```

3. Verificar Conexión

```
# Probar conexión con el usuario creado
psql -h localhost -U thunderrol -d thunderrol -W
# Ingresa la password: thunderrol123
# Si conecta exitosamente, escribir \q para salir
```

🔧 Parte 2: Configuración del Backend (FastAPI)

1. Navegar al directorio del backend

```
cd /home/ubuntu/thunderrol/backend
```

2. Crear y Activar Entorno Virtual

```
# Crear entorno virtual
python3 -m venv venv

# Activar entorno virtual
# En Linux/macOS:
source venv/bin/activate
# En Windows:
# venv\Scripts\activate

# Verificar que está activado (debe mostrar (venv) al inicio de la línea)
which python  # debe mostrar la ruta del venv
```

3. Instalar Dependencias

```
# Actualizar pip
pip install --upgrade pip

# Instalar dependencias
pip install -r requirements.txt

# Verificar instalación
pip list | grep fastapi
```

4. Configurar Variables de Entorno

```
# Crear archivo .env (ya existe, pero verificar contenido)
cat .env

# Debería contener:
# DATABASE_URL=postgresql://thunderrol:thunderrol123@localhost:5432/thunderrol
# SECRET_KEY=your-secret-key-here-thunderrol-2024
# ALGORITHM=HS256
# ACCESS_TOKEN_EXPIRE_MINUTES=30
# ENVIRONMENT=development
```

5. Ejecutar Migraciones de Base de Datos

```
# Verificar que alembic esté configurado
ls alembic.ini

# Ejecutar migraciones para crear las tablas
alembic upgrade head

# Verificar que las tablas se crearon
psql -h localhost -U thunderrol -d thunderrol -c "\dt"
```

6. Poblar Base de Datos con Datos Iniciales (Seeds)

```
# Ejecutar el script de seeders para crear usuarios demo
python -m app.db.seed

# Verificar que se crearon los usuarios
psql -h localhost -U thunderrol -d thunderrol -c "SELECT email, role FROM users;"
```

7. Iniciar el Servidor Backend

```
# Iniciar servidor en modo desarrollo
uvicorn app.main:app --reload --host 0.0.0.0 --port 8000

# Deberías ver algo como:
# INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
# INFO:     Started reloader process [xxxxx] using StatReload
```

8. Verificar Backend

```
# En otra terminal, probar la API
curl http://localhost:8000/health
# Debería responder: {"status": "ok", "timestamp": "..."}

# Ver documentación de la API
# Abrir en navegador: http://localhost:8000/docs
```

⚛️ Parte 3: Configuración del Frontend (Next.js)

1. Abrir Nueva Terminal y Navegar al Frontend

```
cd /home/ubuntu/thunderrol/frontend/app
```

2. Configurar Variables de Entorno

```
# Verificar archivo .env.local
cat .env.local

# Debería contener:
# NEXTAUTH_URL=http://localhost:3000
# NEXTAUTH_SECRET=super-secret-nextauth-key-change-in-production
# NEXT_PUBLIC_API_URL=http://localhost:8000
# DATABASE_URL=postgresql://thunderrol:thunderrol123@localhost:5432/thunderrol
```

3. Instalar Dependencias de Node.js

```
# Usar yarn (recomendado) o npm
yarn install
# O si prefieres npm:
# npm install

# Verificar instalación
yarn list | grep next
# O con npm:
# npm list | grep next
```

4. Configurar Prisma (para autenticación)

```
# Generar cliente de Prisma
yarn prisma generate

# Ejecutar migraciones de Prisma (crea tablas adicionales de NextAuth)
yarn prisma db push

# Ejecutar seeds (crear usuarios demo)
yarn prisma db seed
```

5. Iniciar el Servidor Frontend

```
# Iniciar en modo desarrollo
yarn dev
# O con npm:
# npm run dev

# Deberías ver algo como:
# ▲ Next.js 14.2.28
# - Local:        http://localhost:3000
# - ready started server on [::]:3000, url: http://localhost:3000
```

6. Verificar Frontend

```
# Abrir en navegador: http://localhost:3000
# Deberías ver la página de login de Thunderrol
```

🧪 Parte 4: Testing y Verificación

1. Probar Autenticación

```
# Credenciales de prueba disponibles:
# admin@thunderrol.com / admin123 (ADMIN)
# inventario@thunderrol.com / inv123 (MANAGER)
# taller@thunderrol.com / taller123 (OPERATOR)
# ventas@thunderrol.com / ventas123 (VIEWER)
```

2. Ejecutar Tests del Backend

```
# Navegar al directorio backend
cd /home/ubuntu/thunderrol/backend

# Activar entorno virtual (si no está activado)
source venv/bin/activate

# Instalar dependencias de testing
pip install pytest pytest-asyncio httpx

# Ejecutar tests
pytest tests/ -v

# Test específico
pytest tests/test_auth.py -v
```

3. Ejecutar Tests del Frontend

```
# Navegar al directorio frontend
cd /home/ubuntu/thunderrol/frontend/app

# Ejecutar linting
yarn lint

# Verificar tipos de TypeScript
yarn type-check
# O: npx tsc --noEmit

# Build de producción (para verificar que no hay errores)
yarn build
```

📊 Parte 5: Herramientas de Desarrollo

1. Acceso a Base de Datos

```
# Conectarse directamente a PostgreSQL
psql -h localhost -U thunderrol -d thunderrol

# Comandos útiles dentro de psql:
# \dt          - Listar tablas
# \d users     - Describir tabla users
# SELECT * FROM users; - Ver todos los usuarios
# \q           - Salir
```

2. Logs y Debugging

```
# Ver logs del backend (en la terminal donde corre uvicorn)
# Los logs aparecen automáticamente

# Para logs más detallados, editar app/core/config.py:
# LOG_LEVEL = "DEBUG"
```

3. Herramientas de API

```
# Documentación interactiva de la API
# http://localhost:8000/docs (Swagger UI)
# http://localhost:8000/redoc (ReDoc)
```

🔧 Parte 6: Scripts Útiles de Mantenimiento

1. Script de Inicio Completo

```
# Crear script para iniciar todo (thunderrol/start.sh)
#!/bin/bash
echo "🚀 Iniciando Thunderrol..."

# Verificar PostgreSQL
if ! pg_isready -h localhost -p 5432 -q; then
    echo "❌ PostgreSQL no está corriendo"
    exit 1
fi

# Iniciar backend
echo "🔧 Iniciando backend..."
cd backend
source venv/bin/activate
uvicorn app.main:app --reload --host 0.0.0.0 --port 8000 &
BACKEND_PID=$!

# Esperar un poco
sleep 3

# Iniciar frontend
echo "⚛️ Iniciando frontend..."
cd ../frontend/app
yarn dev &
FRONTEND_PID=$!

echo "✅ Thunderrol iniciado:"
echo "   Backend:  http://localhost:8000"
echo "   Frontend: http://localhost:3000"
echo "   API Docs: http://localhost:8000/docs"

# Función para limpiar procesos al salir
cleanup() {
    echo "🛑 Cerrando servicios..."
    kill $BACKEND_PID $FRONTEND_PID 2>/dev/null
    exit 0
}

trap cleanup SIGINT SIGTERM

# Mantener el script corriendo
wait
```

2. Script de Reset de Base de Datos

```
# Crear script para resetear DB (thunderrol/reset_db.sh)
#!/bin/bash
echo "🗄️ Reseteando base de datos..."

cd backend
source venv/bin/activate

# Eliminar todas las tablas
alembic downgrade base
# Recrear todas las tablas
alembic upgrade head
# Poblar con datos iniciales
python -m app.db.seed

echo "✅ Base de datos reseteada"
```

3. Hacer scripts ejecutables

```
chmod +x /home/ubuntu/thunderrol/start.sh
chmod +x /home/ubuntu/thunderrol/reset_db.sh
```

⚡ Parte 7: Comandos Rápidos de Desarrollo

Comandos del Backend

```
cd /home/ubuntu/thunderrol/backend
source venv/bin/activate

# Iniciar servidor
uvicorn app.main:app --reload --port 8000

# Crear nueva migración
alembic revision --autogenerate -m "descripción del cambio"

# Aplicar migraciones
alembic upgrade head

# Revertir última migración
alembic downgrade -1

# Ejecutar tests
pytest tests/ -v

# Formatear código
black app/

# Linting
ruff app/
```

Comandos del Frontend

```
cd /home/ubuntu/thunderrol/frontend/app

# Iniciar desarrollo
yarn dev

# Build de producción
yarn build

# Iniciar producción
yarn start

# Linting
yarn lint

# Verificar tipos
yarn type-check

# Regenerar Prisma client
yarn prisma generate

# Ver base de datos con Prisma Studio
yarn prisma studio
```

🚨 Troubleshooting

Problemas Comunes

1. Error de Conexión a PostgreSQL

```
# Verificar que PostgreSQL esté corriendo
sudo systemctl status postgresql

# Verificar puerto
sudo netstat -tlnp | grep 5432

# Verificar configuración en pg_hba.conf
sudo nano /etc/postgresql/15/main/pg_hba.conf
# Asegurar que hay una línea como:
# local   all             all                                     trust
```

2. Error “Module not found” en Backend

```
# Verificar que el entorno virtual esté activado
which python  # debe mostrar path del venv

# Reinstalar dependencias
pip install -r requirements.txt

# Verificar que el PYTHONPATH incluya el directorio actual
export PYTHONPATH=$PYTHONPATH:/home/ubuntu/thunderrol/backend
```

3. Error de Build en Frontend

```
# Limpiar cache
rm -rf .next node_modules yarn.lock
yarn install

# Verificar versión de Node
node --version  # debe ser 18+

# Regenerar Prisma
yarn prisma generate
```

4. Error de Migraciones

```
# Ver estado actual de migraciones
alembic current

# Ver historial
alembic history

# Forzar migración específica
alembic upgrade <revision_id>

# En caso de error grave, resetear completamente
alembic downgrade base
alembic upgrade head
```

📝 Notas Adicionales

1. Puertos Utilizados

- **Frontend**: http://localhost:3000
- **Backend**: http://localhost:8000
- **PostgreSQL**: localhost:5432

2. Archivos de Configuración Importantes

- `backend/.env` - Variables de entorno del backend
- `frontend/app/.env.local` - Variables de entorno del frontend
- `backend/alembic.ini` - Configuración de migraciones
- `frontend/app/prisma/schema.prisma` - Schema de la base de datos

3. Ubicaciones de Logs

- Backend: logs aparecen en la terminal donde ejecutas `uvicorn`
- Frontend: logs en la terminal donde ejecutas `yarn dev`
- PostgreSQL: `/var/log/postgresql/` (Linux)

4. Comandos para Parar los Servicios

```
# Parar con Ctrl+C en cada terminal donde corren los servicios
# O buscar y matar los procesos:
ps aux | grep uvicorn
ps aux | grep next-server
kill <PID>
```

¡Con esta guía deberías poder ejecutar Thunderrol completamente en modo manual! Si encuentras algún problema, revisa la sección de troubleshooting o verifica los logs en cada servicio.