Introducción

Esta API fue desarrollada con **Django REST Framework (DRF)** y permite gestionar vulnerabilidades en infraestructura Cloud, cruzando información con los **CVEs del NIST**.

Se implementaron endpoints para obtener, registrar y filtrar vulnerabilidades, así como un sistema de autenticación basado en **JWT**.

Tecnologías Utilizadas

- **Python 3.11**
- **Django 4.x**
- **Django REST Framework**
- **SQLite / PostgreSQL** (configurable)
- **Docker**
- **GitHub**

Configuración y Ejecución

1 Clonar el repositorio

```
git clone https://github.com/TU_USUARIO/NOMBRE_DEL_REPO.git
cd NOMBRE_DEL_REPO
```

2 Configurar el entorno virtual

```
python -m venv venv
source venv/bin/activate  # Mac/Linux
venv\Scripts\activate  # Windows
pip install -r requirements.txt
```

3 Configurar la base de datos

- **Por defecto:** usa **SQLite**.
- **Para PostgreSQL:** modificar `settings.py`:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'nombre_db',
        'USER': 'usuario',
        'PASSWORD': 'contraseña',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

4 Aplicar migraciones

```
python manage.py migrate
```

5️⃣ Ejecutar la aplicación

```
python manage.py runserver
```

La API estará disponible en: `http://localhost:8000/`

Endpoints de la API

🔹 Obtener todas las vulnerabilidades

GET /vulnerabilities/

```
[
  {"id": "CVE-2024-12345", "severity": "high", "status": "unfixed"},
  {"id": "CVE-2023-56789", "severity": "medium", "status": "fixed"}
]
```

🔹 Registrar una vulnerabilidad fixeada

POST /vulnerabilities/fixed/ Body:

```
{
  "id": "CVE-2024-12345",
  "status": "fixed"
}
```

🔹 Listar vulnerabilidades sin arreglar

GET /vulnerabilities/unfixed/

🔹 Resumen de vulnerabilidades por severidad

GET /vulnerabilities/summary/

```
{
  "critical": 5,
  "high": 10,
  "medium": 15,
  "low": 7
}
```

🔹 Obtener información del NIST

GET /vulnerabilities/fetch_nist/

Autenticación y Seguridad

La API usa **JSON Web Tokens (JWT)**. Para obtener un token:

```
POST /api/token/
{
  "username": "admin",
  "password": "admin123"
}
```

Respuesta:

```
{
  "access": "token_aqui",
  "refresh": "token_refresh"
}
```

Para usarlo en los endpoints:

```
Authorization: Bearer token_aqui
```

Docker: Ejecución en Contenedor

1 Construir la imagen

```
docker build -t django-api .
```

2 Ejecutar el contenedor

```
docker run -p 8000:8000 django-api
```

Pruebas Unitarias

Para ejecutar pruebas:

```
python manage.py test
```

Documentación Interactiva

Para acceder a la documentación generada automáticamente:

- Swagger: `http://localhost:8000/swagger/`
- ReDoc: `http://localhost:8000/redoc/`

Conclusión

Esta API permite gestionar vulnerabilidades en entornos Cloud, con integración de datos del **NIST**, autenticación JWT y despliegue en **Docker**. Está lista para ser ampliada con nuevas funcionalidades y seguridad avanzada.
