**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERÍA**

**ESCUELA DE INGENIERÍA DE SISTEMAS**

****

**Guía Práctica de Laboratorio**

**Laboratorio 02  
“Auditoria Móvil”**

**Que se presenta para el curso:**

“Auditoría de sistemas”

**LO**

**AUTOR(ES) : APELLIDOS Y NOMBRES**

**Docente:**

Dr. Oscar Juan Jimenez Flores

**TACNA – PERÚ**

**Índice General**

[Introducción 3](#_Toc206165197)

[1. Información sobre el evento práctico 4](#_Toc206165198)

[1.1. Título del evento práctico 4](#_Toc206165199)

[Laboratorio 02. Auditoria Móvil 4](#_Toc206165200)

[1.2. Objetivos 4](#_Toc206165201)

[1.3. Tiempo de duración (horas) 4](#_Toc206165202)

[1.4. Resultados de Aprendizaje (RA) 4](#_Toc206165203)

[1.5. Recursos (Equipos, materiales, programas y otros) 4](#_Toc206165204)

[2. Procedimiento o Metodología 5](#_Toc206165205)

[3. Conclusiones 15](#_Toc206165216)

[4. Referencias Bibliográficas 15](#_Toc206165217)

[5. Actividad 16](#_Toc206165218)

# Introducción

La auditoría de seguridad en Tecnologías de la Información y Comunicación (TIC) es un proceso sistemático y estructurado que evalúa la eficacia y la integridad de los controles de seguridad implementados en una organización. Su objetivo principal es identificar vulnerabilidades, asegurar el cumplimiento de políticas y normas, y verificar que los sistemas de TIC protejan adecuadamente la confidencialidad, integridad y disponibilidad de la información.

Durante una auditoría de seguridad en TIC, se examinan diversos aspectos, como la configuración de hardware y software, los controles de acceso, las políticas de seguridad, los procedimientos de respaldo y recuperación, y la gestión de incidentes. Además, se evalúan las prácticas de gestión de riesgos y el cumplimiento de normativas y estándares relevantes.

El objetivo de este laboratorio es realizar una auditoría de seguridad, analizar los posibles riesgos asociados y proponer controles.

Guía de Laboratorio Nº 02 “Auditoria de seguridad y hallazgos”

|  |
| --- |
| Información sobre el evento práctico |

### Título del evento práctico

### Laboratorio 02. Auditoria Móvil

### Objetivos

* Identificar los elementos que participan en la Auditoría de seguridad.
* Analizar los posibles riesgos asociados.
* Proponer controles efectivos para minimizar el riesgo asociado.

### Tiempo de duración (horas)

06 horas académicas

### Resultados de Aprendizaje (RA)

[AG-I02] Ética

[AG-I04] Comunicación

[AG-I07] Conocimientos de Ingeniería

[AG-I08] Análisis de Problemas

[AG-I09] Diseño y Desarrollo de Soluciones

[AG-I11] Uso de Herramientas

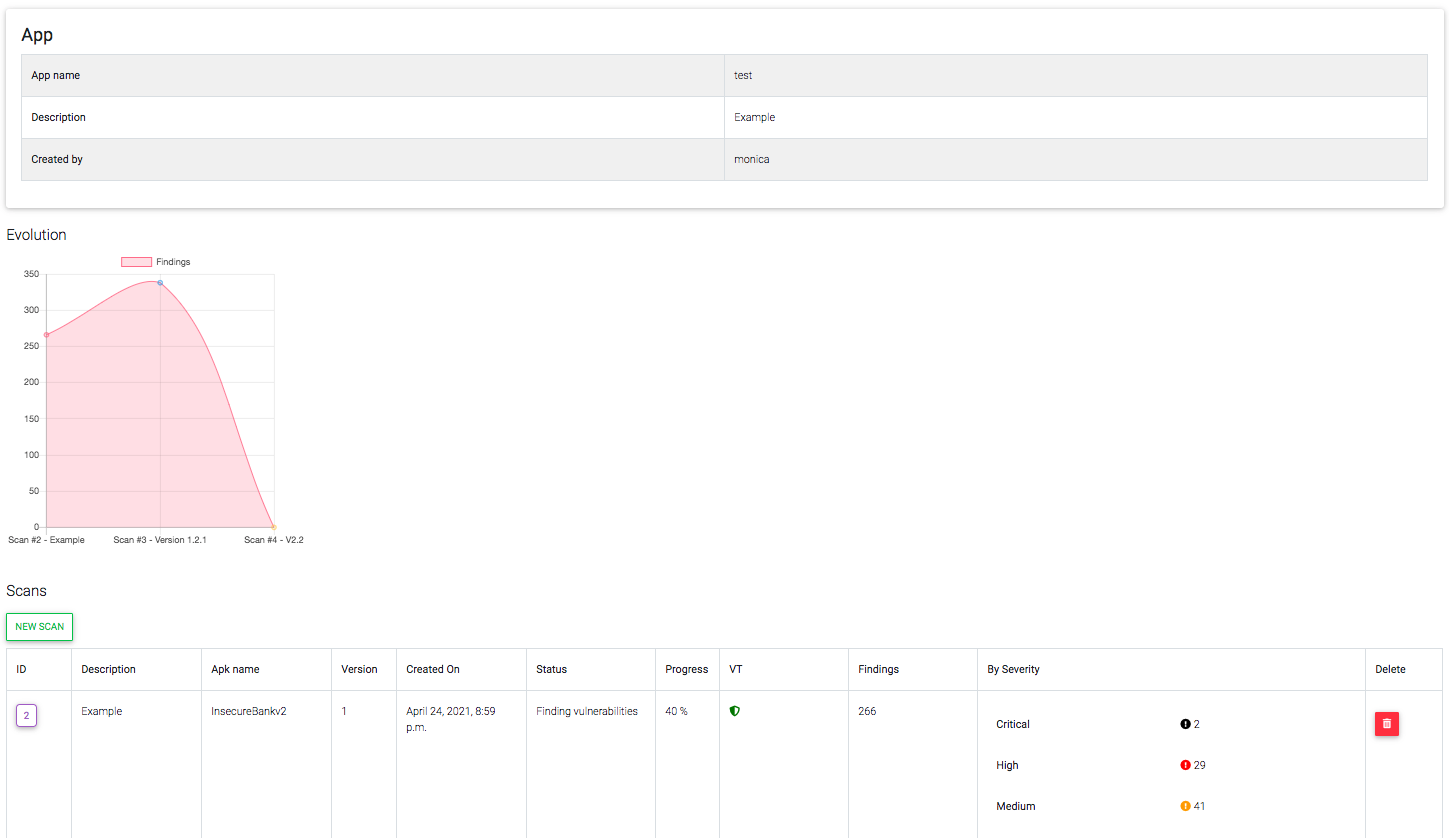
### Recursos (Equipos, materiales, programas y otros)

* Computador con S.O.
* Descargar e instalar repositorio GIT

<https://github.com/OscarJimenezFlores/CursoAuditoria/tree/main/AuditoriaMovil>

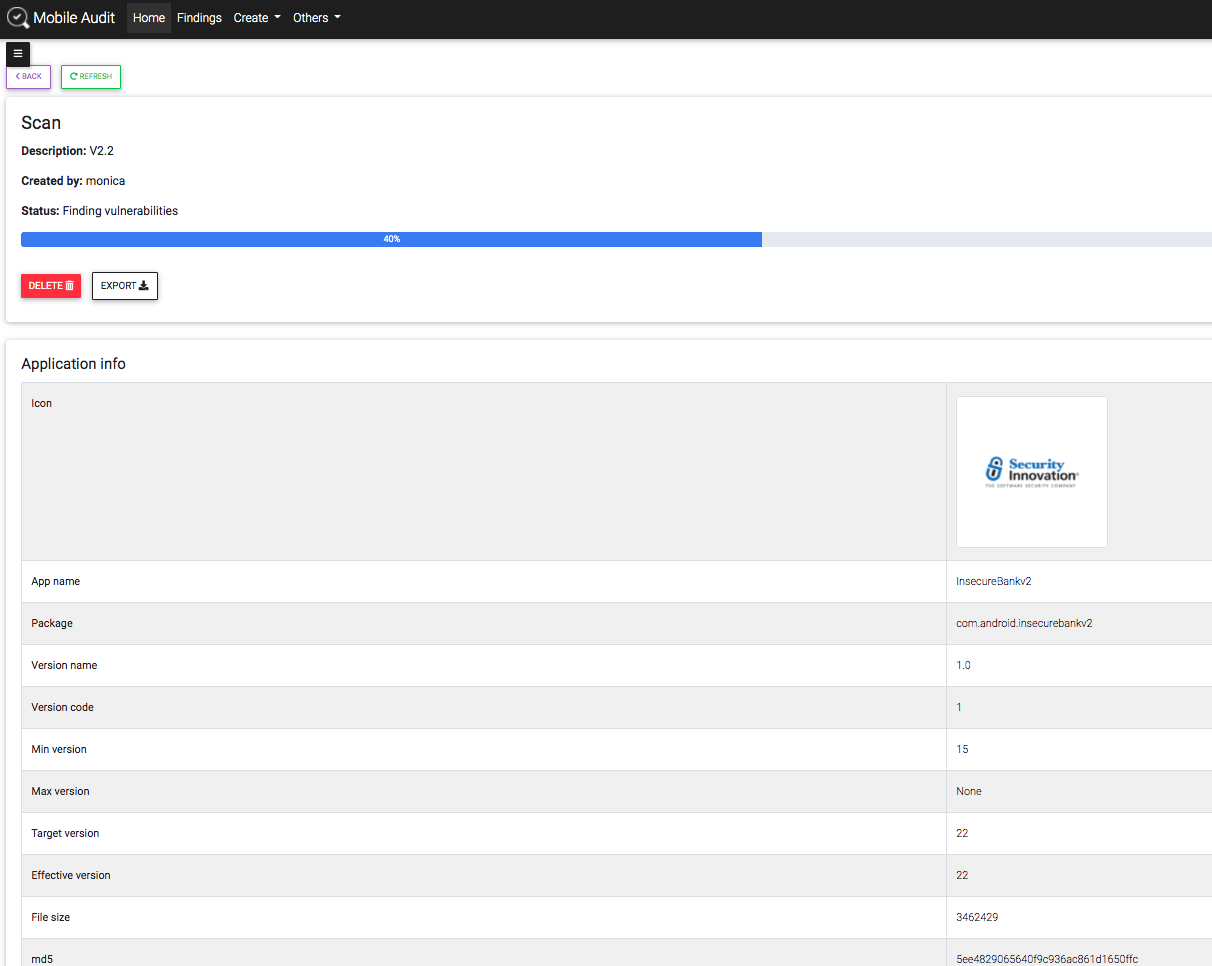
|  |
| --- |
| Procedimiento o Metodología |

**Auditoría Móvil** - Realiza el análisis estático de seguridad (SAST) y análisis de malware para archivos APK de Android. La aplicación web está desarrollada en Django para realizar análisis estático y detectar malware en archivos APK de Android.

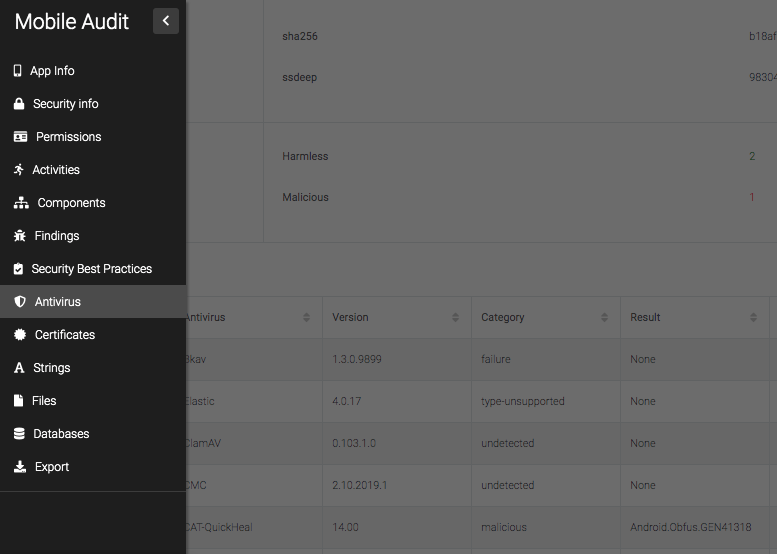
[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/app.png)

En cada uno de los escaneos, se muestra la siguiente información:

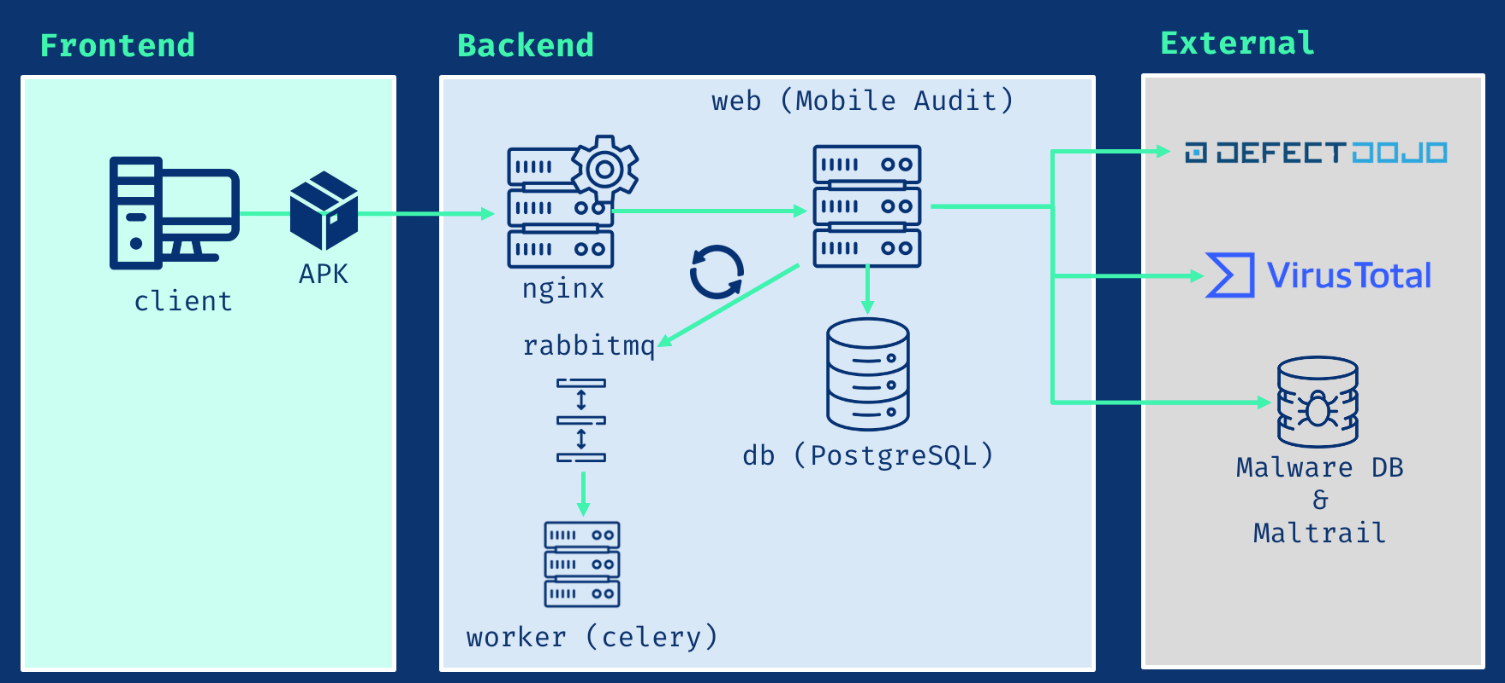
* Información de la aplicación
* Información de seguridad
* Componentes
* Hallazgos SAST
* Mejores prácticas implementadas
* Información de Virus Total
* Información de certificados
* Cadenas
* Bases de datos
* Archivos

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/scan.png)

Para un acceso fácil, hay una barra lateral en la página izquierda del escaneo:

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/menu.png)

### Componentes

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/architecture.png)

* **db**: PostgreSQL 3.11.5
* **nginx**: Nginx 1.23.3
* **rabbitmq**: RabbitMQ 3.11.5
* **worker**: Celery 5.2.2
* **web**: Aplicación Mobile Audit (Django 3.2.16)

### Imágenes base de Docker

La imagen está basada en python buster. Enlace a la [imagen de Docker Hub](https://hub.docker.com/repository/docker/mpast/mobile_audit)

| **Imagen** | **Etiquetas** | **Base** |
| --- | --- | --- |
| mpast/mobile\_audit | 3.0.0 | python:3.9.16-buster |
| mpast/mobile\_audit | 2.2.1 | python:3.9.7-buster |
| mpast/mobile\_audit | 1.3.8 | python:3.9.4-buster |
| mpast/mobile\_audit | 1.0.0 | python:3.9.0-buster |

### Características principales

* Usa Docker para un despliegue fácil en un entorno multiplataforma



* Extrae toda la información del archivo APK



* Analiza todo el código fuente en busca de vulnerabilidades



* Todos los hallazgos están categorizados y siguen los estándares **CWE**



* Todos los hallazgos están categorizados e incluyen el **Mobile Top 10 Risk**



* También resalta las **Mejores prácticas en la implementación segura de Android** en el APK



* Los hallazgos pueden ser editados y los **falsos positivos pueden ser clasificados y eliminados**



* Todos los resultados del escaneo pueden ser **exportados a PDF**



* Autenticación de usuario y gestión de usuarios



* API v1 con Swagger y ReDoc



* TLS



* Recarga dinámica de la página (en progreso)



* Integración con LDAP



* Exportar a Markdown



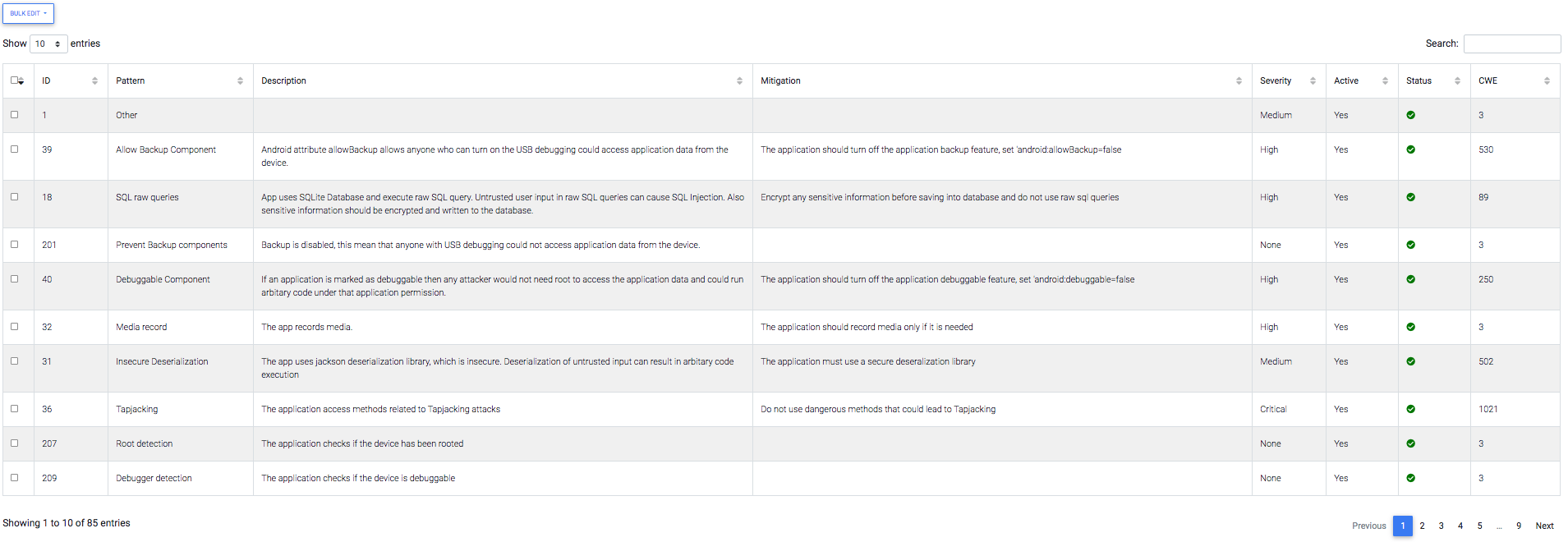
* Exportar a CSV



### Patrones

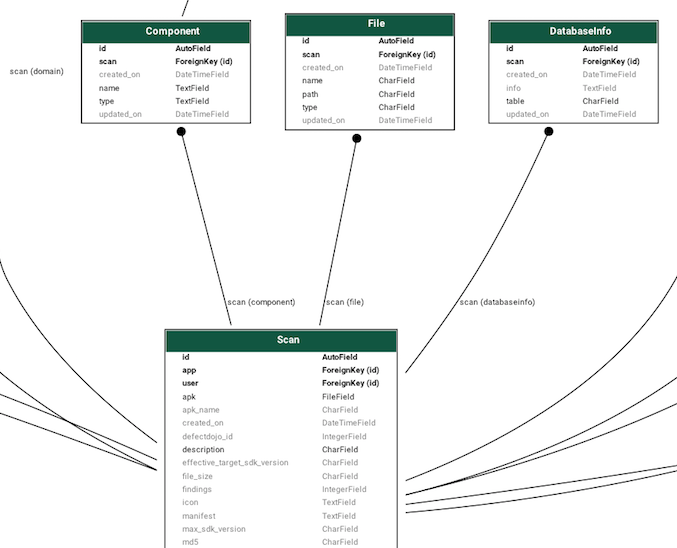
La aplicación tiene un motor con diferentes reglas y patrones que se utilizan durante la fase de escaneo para detectar vulnerabilidades y/o código malicioso en el APK.

Estos pueden ser activados y desactivados en /patterns

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/patterns.png)

### Modelos

La aplicación tiene modelos creados para cada una de las entidades de la información de los escaneos para poder crear relaciones y obtener las mejores conclusiones para cada uno de los APKs.

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/models_snippet.png)

Para ver el esquema completo de los modelos, ve a [modelos](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/models.png)

### Integraciones

#### Virus Total (API v3)

Verifica si se ha realizado un escaneo del APK y extrae toda su información. También, existe la posibilidad de subir el APK si se selecciona una propiedad en el entorno (Deshabilitado por defecto).

#### Defect Dojo (API v2)

Es posible subir los hallazgos al gestor de defectos.

#### MalwareDB & Maltrail

Verifica en la base de datos si existen URLs en el APK relacionadas con Malware.

### Instalación

Usando Docker-compose:

El archivo docker-compose.yml proporcionado permite ejecutar la aplicación localmente en desarrollo.

Para construir la imagen local y si hay cambios en el Dockerfile de la aplicación, puedes construir la imagen con:

docker-compose build

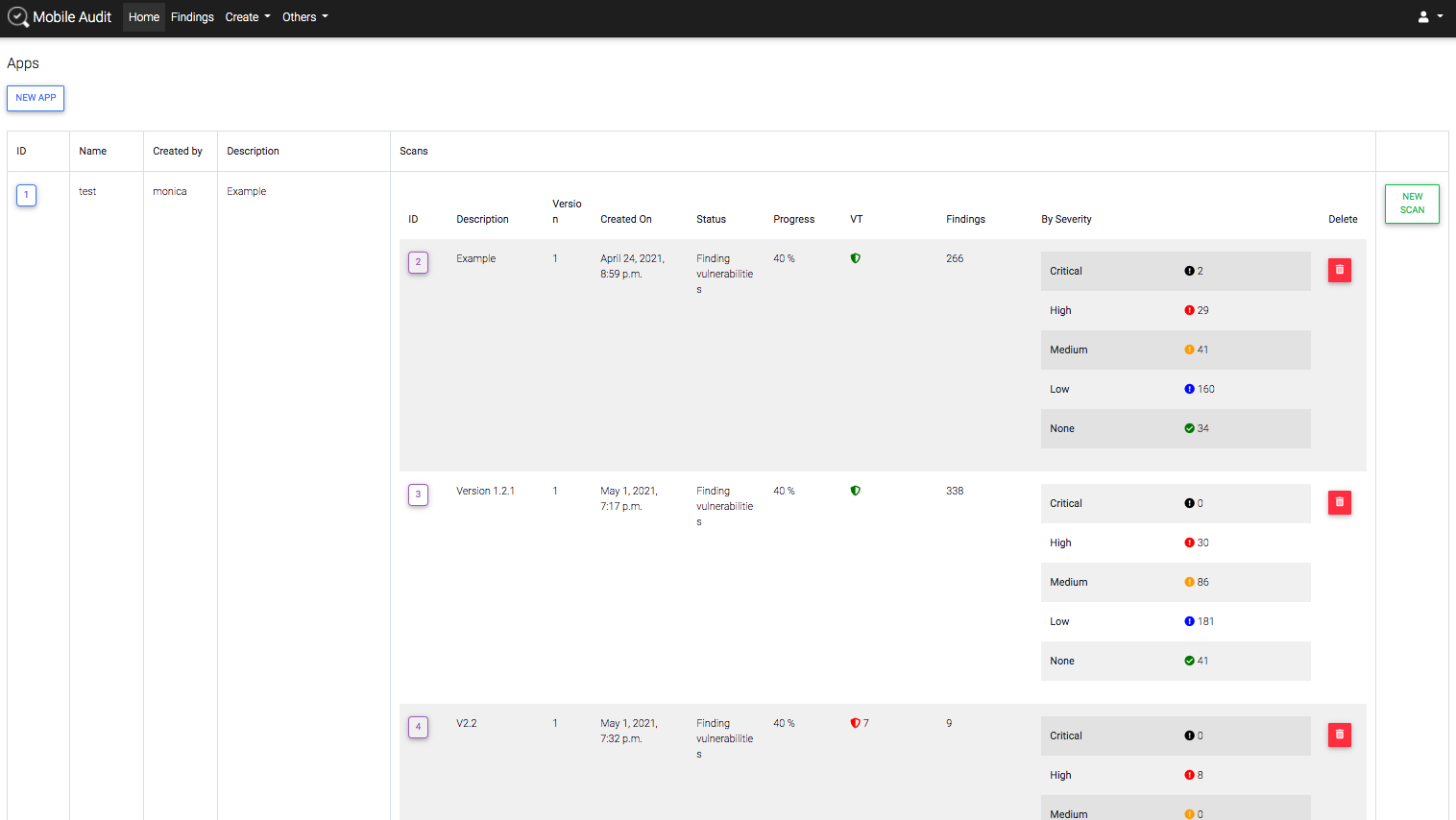
Luego, para iniciar el contenedor, ejecuta:

docker-compose up

Opcional: ejecuta en modo separado (sin ver los logs)

docker-compose up -d

Una vez que la aplicación esté en marcha, puedes probarla navegando a: <http://localhost:8888/> para acceder al tablero de control.

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/dashboard.png)

También hay una versión TLS usando docker-compose.prod.yaml que se ejecuta en el puerto 443.

Para usarlo, ejecuta:

docker-compose -f docker-compose.prod.yaml up

Luego, puedes probar la aplicación navegando a: <https://localhost/> para acceder al tablero de control.

Para más información, consulta [TLS](https://github.com/OscarJimenezFlores/CursoAuditoria#tls)

Para detener y eliminar los contenedores, ejecuta:

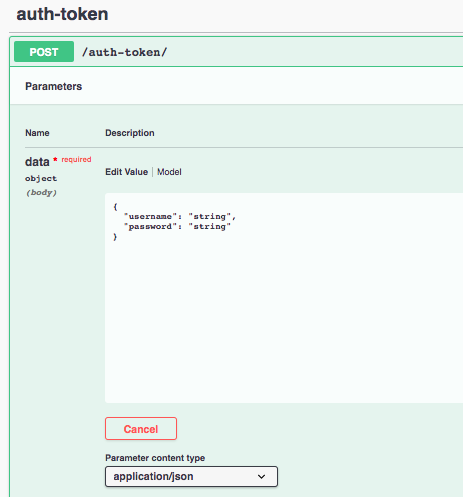
docker-compose down

### API v1

Integración REST API con Swagger y ReDoc.

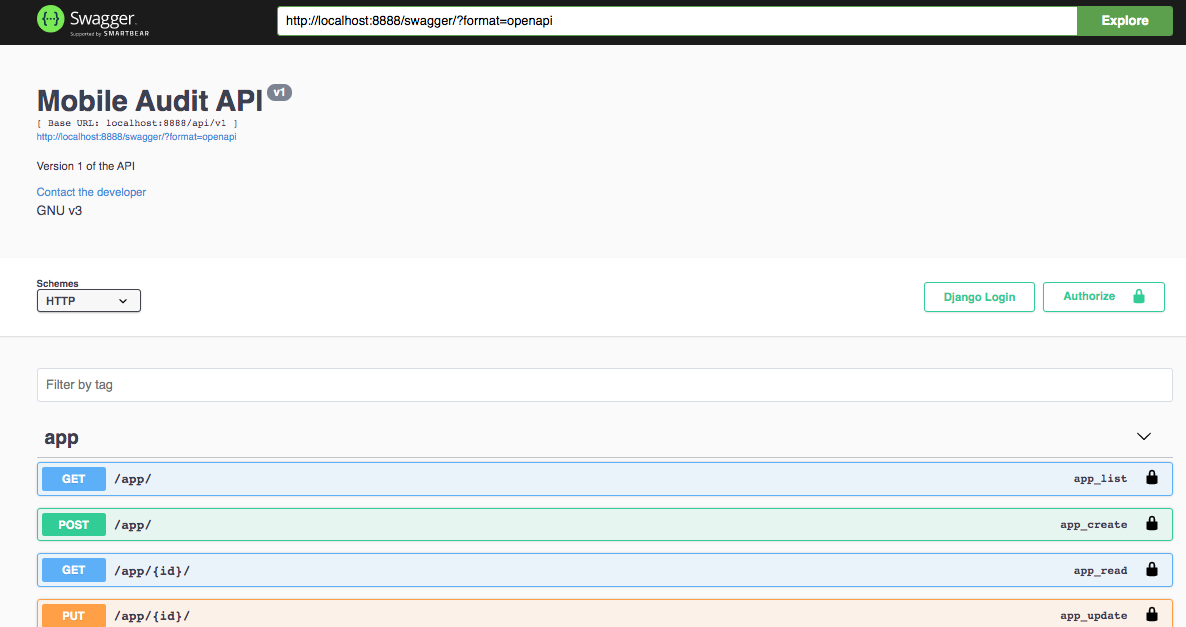
#### Usage

* Endpoint para autenticar y obtener el token: /api/v1/auth-token/

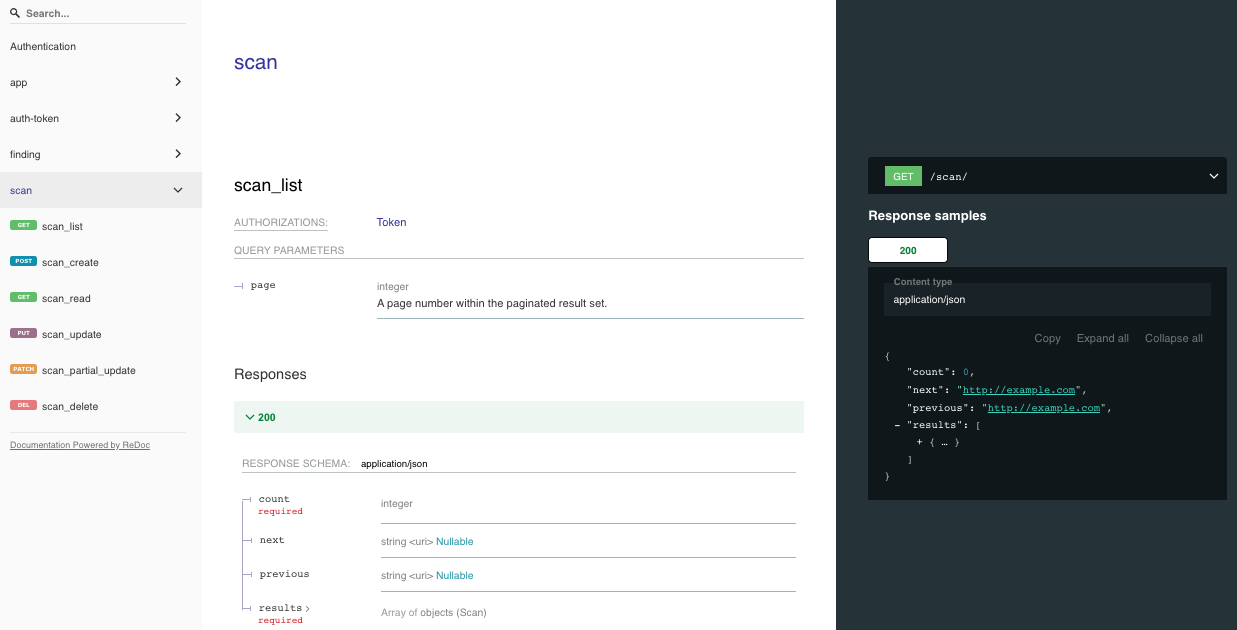
[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/auth_token.png)

* Una vez autenticado, usa el encabezado en todas las solicitudes: Authorization: Token <ApiKey>

#### Swagger

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/swagger.png)

#### ReDoc

[](https://github.com/OscarJimenezFlores/CursoAuditoria/blob/main/app/static/redoc.png)

#### Endpoints

* Una vista JSON de la especificación de la API en /swagger.json
* Una vista YAML de la especificación de la API en /swagger.yaml
* Una vista swagger-ui de la especificación de la API en /swagger/
* Una vista ReDoc de la especificación de la API en /redoc/

### TLS

#### Requisitos previos

* Agrega los certificados en nginx/ssl
* Para generar un certificado autofirmado:

openssl req -x509 -nodes -days 1 -newkey rsa:4096 -subj "/C=ES/ST=Madrid/L=Madrid/O=Example/OU=IT/CN=localhost" -keyout nginx/ssl/nginx.key -out nginx/ssl/nginx.crt

#### Configuración de Nginx

* TLS - port 443: nginx/app\_tls.conf
* Standard - port 8888: nginx/app.conf

#### Configuración de Docker

Por defecto, hay un volumen en docker-compose.yml con la configuración con el puerto 8888 disponible.

- ./nginx/app.conf:/etc/nginx/conf.d/app.conf

**En un entorno de producción**, usa docker-compose.prod.yaml con el puerto 443:

- ./nginx/app\_tls.conf:/etc/nginx/conf.d/app\_tls.conf

### Variables de entorno

Todas las variables de entorno están en un archivo .env, hay un .env.example con todas las variables necesarias. También están recopiladas en app/config/settings.py:

CWE\_URL = env('CWE\_URL', 'https://cwe.mitre.org/data/definitions/')

MALWARE\_ENABLED = env('MALWARE\_ENABLED', True)

MALWAREDB\_URL = env('MALWAREDB\_URL', 'https://www.malwaredomainlist.com/mdlcsv.php')

MALTRAILDB\_URL = env('MALTRAILDB\_URL', 'https://raw.githubusercontent.com/stamparm/aux/master/maltrail-malware-domains.txt')

VIRUSTOTAL\_ENABLED = env('VIRUSTOTAL\_ENABLED', False)

VIRUSTOTAL\_URL = env('VIRUSTOTAL\_URL', 'https://www.virustotal.com/')

VIRUSTOTAL\_FILE\_URL = env('VIRUSTOTAL\_FILE\_URL', 'https://www.virustotal.com/gui/file/')

VIRUSTOTAL\_API\_URL\_V3 = env('VIRUSTOTAL\_API\_URL\_V3', 'https://www.virustotal.com/api/v3/')

VIRUSTOTAL\_URL\_V2 = env('VIRUSTOTAL\_API\_URL\_V2', 'https://www.virustotal.com/vtapi/v2/file/')

VIRUSTOTAL\_API\_KEY = env('VIRUSTOTAL\_API\_KEY', '')

VIRUSTOTAL\_UPLOAD = env('VIRUSTOTAL\_UPLOAD', False)

DEFECTDOJO\_ENABLED = env('DEFECTDOJO\_ENABLED', False)

DEFECTDOJO\_URL = env('DEFECTDOJO\_URL', 'http://defectdojo:8080/finding/')

DEFECTDOJO\_API\_URL = env('DEFECTDOJO\_API\_URL', 'http://defectdojo:8080/api/v2/')

DEFECTDOJO\_API\_KEY = env('DEFECTDOJO\_API\_KEY', '')

|  |
| --- |
| Conclusiones |

En base al laboratorio desarrollado, sobre auditoría de seguridad, y particularmente en la extracción de evidencias u obtención de hallazgos, es impotante reconocer los riesgos asociados en torno al caso y determinar los controles necesarios para minimizar el riesgo de exfiltrado de documentos privados de la organización.

|  |
| --- |
| Referencias Bibliográficas |

* Calder, A., & Watkins, S. (2015). IT Governance: An International Guide to Data Security and ISO27001/ISO27002. Kogan Page.
* Orebaugh, A., Ramirez, D., Beale, J., & Wright, J. (2006). Wireshark & Ethereal Network Protocol Analyzer Toolkit. Syngress.
* Stallings, W., & Brown, L. (2018). Computer Security: Principles and Practice (4th ed.). Pearson.
* Weaver, A. C. (2013). Computer Security: A Hands-on Approach. CRC Press.

AlphaCloud

* Del Peso, E., Del Peso, M., & Piattini, M. (2008). Auditoría de tecnologías y sistemas de información. Rama. ISBN 9788499646039.

|  |
| --- |
| Actividad |

Desarrolla el laboratorio, recaba las evidencias del caso y presenta los resultados en un **informe PDF** con las siguientes características.

* Portada
* Resumen (descripción general de la actividad realizada con su resultado más importante)
* Materiales y métodos (recursos tecnológicos a utilizar y enlaces de descarga si corresponde)
* Resultados (incluye captura de imágenes)
* Conclusiones
* Anexos (opcional)

Rúbrica de evaluación

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ESCALA** | **DESCRIPCIÓN** | | | | |
| [E] Excelente | El criterio evaluado cumple a cabalidad lo esperado | | | | |
| [A] Aceptable | El criterio evaluado cumple parcialmente lo esperado | | | | |
| [D] Deficiente | El criterio evaluado no cumple lo esperado | | | | |
| [N] No desarrollado | El criterio no fue presentado | | | | |
| **CRITERIOS** | | **E** | **A** | **D** | **N** | |
| 1. Presenta portada y resumen | | 3 | 2 | 1 | 0 | |
| 1. Identifica los materiales y métodos a emplear | | 5 | 4 | 2 | 0 | |
| 1. Explica los resultados con sus evidencias (anexo) | | 7 | 6 | 3 | 0 | |
| 1. Desarrolla coherentemente sus conclusiones | | 5 | 3 | 2 | 0 | |
| **Puntajes** | | **20** | **15** | **8** | **0** | |

**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERÍA**

**ESCUELA DE INGENIERÍA DE SISTEMAS**

****

**Informe de Laboratorio**

**Laboratorio 02  
“Auditoria Móvil”**

**Que se presenta para el curso:**

“Auditoría de sistemas”

**LO**

**Integrante(s):**

* Apellidos, Nombres [código de estudiante]
* ……

**Docente:**

Dr. Oscar Juan Jimenez Flores

TACNA – PERÚ

202…..