Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Ingeniería de Sistemas e Industrial
Curso: Ingeniería de Software I

CatatUNbo

Integrantes:

Felipe Rojas Marin ⚽
Santiago Alejandro Rojas 😇🙏
Juan Diego Rozo Álvarez 😡🔨
Miguel Angel Citarella Camargo ☠️

Proyecto - Spynet

Spynet es una aplicación de escritorio diseñada para asistir a empresas de auditoría informática en la realización, gestión y documentación de auditorías de seguridad técnica. Su objetivo principal es automatizar y centralizar procesos clave de la auditoría, tales como el escaneo de puertos y servicios, el análisis cualitativo de vulnerabilidades detectadas, la generación de informes técnicos, y el seguimiento estructurado de observaciones.

👉¿Qué buscamos con **Spynet**? 🥸

Con **Spynet** buscamos:

- ➔ **Simplificar la gestión de auditorías** mediante un sistema que permita registrar, visualizar y organizar auditorías en curso o finalizadas.
- ➔ **Automatizar tareas técnicas complejas** como escaneos de red con Nmap, integrando herramientas de análisis con inteligencia artificial para interpretar los resultados de forma comprensible y profesional.
- ➔ **Establecer trazabilidad y control de cambios**, mediante un historial inmutable de observaciones asociadas a cada auditoría.
- ➔ **Facilitar la comunicación con los clientes**, permitiendo que envíen solicitudes de auditoría directamente desde la plataforma.
- ➔ **Unificar toda la evidencia técnica y cualitativa** en un único informe PDF exportable, optimizando la presentación final de resultados a los clientes.
- ➔ **Integrar funcionalidades complementarias de análisis contextual**, como una sección donde se realiza *web scraping del comportamiento actual de criptomonedas* (BTC, ETH, etc.) desde Binance. Esta función permite observar correlaciones entre incidentes en activos digitales y posibles vulnerabilidades en sistemas conectados a servicios financieros o Web3.

✅Lista de requerimientos:

1. Consultar Auditorías
2. Registrar Auditoría
3. Historial de observaciones hecho
4. Escaneo con NMAP hecho
5. Análisis automático de resultados (IA o reglas) hecho
6. Exportación de resultados de Nmap hecho
7. Visualizar auditoría específica
8. Registro de Usuarios con roles
9. Consulta de Auditorías pasadas (admin)
10. Asignación de auditores
11. Registro de Hallazgos
12. Búsqueda y filtrado de auditorías
13. Módulo escalable
14. Compatibilidad de Escritorio

15. Tiempo de respuesta razonable
16. Tolerancia a errores controlados
17. Navegación clara y organizada
18. Canal de solicitudes del cliente
19. Proveer información sobre mercado de criptomonedas
20. Estado de auditoria con etiquetas
21. Visualización gráfica de auditorías futuras

Clasificación de Requerimientos:

FUNCIONALES (RF):	**NO FUNCIONALES (RNF):**
RF_01(RQ_1): Consulta de auditorías registradas en un panel general. RF_02(RQ_2): Registro de una nueva auditoría que contiene nombre del negocio, número del contacto, correo electrónico del contacto, teléfono, IPs del sistema web, observaciones y estado de la auditoría (No iniciado, en proceso, en pausa o finalizado). RF_03(RQ_3): Historial no modificable de observaciones visible para cada auditoría registrada con características como fecha, auditor quien escribió la observación, título de la misma y la descripción. RF_04(RQ_4): Escaneo agresivo de puertos de la red a partir de la IP otorgada en el registro de la auditoría con la herramienta NMAP. RF_05(RQ_5): Análisis de resultados generados por Nmap por medio de IA (Open AI) para devolver una bitácora cualitativa a partir de un prompt definido. RF_06(RQ_6): Exportación de resultados de NMAP a un formato PDF junto a la bitácora cualitativa generada por la IA de OpenAI. RF_07(RQ_7): Visualización de una auditoría específica donde se muestra toda la información suscrita al registro (RF_02). En este apartado se encuentra el historial de las observaciones (RF_03), el estado de la auditoría (RF_12) y la sección de escaneo con NMAP (RF_04, RF_05 y RF_06).	RNF_01(RQ_14): La aplicación deberá ser compatible con sistemas de escritorio, y todas sus funcionalidades se deben llevar a cabo en este entorno específico. RNF_02(RQ_13): El sistema debería poder ser escalable, es decir, tener una estructura que permita añadir más módulos con distintas funcionalidades en caso de necesitarlo. RNF_03(RQ_15): El sistema debería responder a las solicitudes de los usuarios en menos de 2 segundos, esto excluyendo a los escaneos exhaustivos y otro tipo de análisis que por su estructura necesitan de mayor tiempo para la respuesta. RNF_04(RQ_16): El sistema tendrá tolerancia a archivos corruptos y dañados sin dar errores, esto debido a que es un comportamiento esperable dentro del contexto del sistema. RNF_05(RQ_17): La interfaz debe tener una estructura jerárquica con menú lateral o pestañas para facilitar el acceso a cada módulo.

RF_08(RQ_8): Registro de nuevos usuarios con determinados roles (“Auditor”, “Inspector”, “Administrador”, “Cliente”). Cada rol debe tener determinados permisos sobre acciones en la aplicación. RF_09(RQ_12): Búsqueda de auditorías registradas en la base de datos a través del nombre del cliente (referente usualmente al nombre de la empresa), el responsable asignado, el estado de la auditoría y el tiempo límite de entrega de esta. RF_10(RQ_10): Asignación de auditorías a determinados roles con permisos de edición. Además, de la asignación de la auditoría a más de un rol, y la asignación a más de una persona dentro del rol. RF_11(RQ_9): Acceso al historial de auditorías terminadas, esto a través de información asociada a las auditorías, como lo es la fecha de finalización, el responsable, el estado final y el nombre del cliente. RF_12(RQ_20): Gestión del estado de la auditoría mediante etiquetas visuales como lo son “En curso”, ”Pendiente”, “Finalizado”, “Archivado”. Con posibilidad de actualización por medio de un rol con los permisos adecuados. RF_13(RQ_11): Registro manual de hallazgos relacionados con la auditoría, donde incluye datos como nombre del hallazgo, descripción, severidad (bajo, medio, alto, crítico) recomendación y evidencia RF_14(RQ_18): Sección donde un cliente manda una solicitud para la realización de una auditoría, que contiene los campos de información como nombre del negocio, número del contacto, correo electrónico del contacto, teléfono, IPs del sistema web, observaciones RF_15 (RQ_19): Apartado con el cuál se puede consultar a páginas de Binance, y realizar un resumen mediante herramientas visuales, sobre el estado de criptomonedas, como: Bitcoin (BTC), Ethereum (ETH), Cardano (ADA) y Solana (SOL). Se podrá	

hacer seguimiento de su precio en función del tiempo, volumen de ventas, valores máximos y mínimos y variabilidad. RF_16 (RQ_21): La aplicación debe ser capaz de organizar las auditorías pendientes por el prestador de servicios, en un calendario, en el que pueda consultar rápidamente el plazo para completar todo el ciclo de trabajo y la información básica para contextualizarse con respecto al trabajo pendiente, dependiendo de la metodología del prestador.	

Priorización MoSCoW

MUST:

- ➔ RNF_01: El sistema deberá funcionar con sistemas de escritorio, y todas sus funcionalidades se deben llevar a cabo en este entorno específico.

- ➔ RF_01: El usuario con rol de auditor y administrador debe poder consultar una lista de auditorías registradas en un panel general. Cada auditoría listada debe incluir los siguientes campos: identificación de la auditoría, nombre del cliente, fecha de creación de la auditoría, nombre del auditor responsable, estado actual de la auditoría (no iniciado, en proceso, en pausa, finalizado)

- ➔ RF_02: El usuario con rol de auditor y administrador debe poder registrar una nueva auditoría, la cual debe contener los siguientes campos obligatorios: nombre del negocio, nombre del contacto responsable, correo electrónico del contacto, número telefónico del contacto, una o varias direcciones IP objetivo del sistema web, campo de observaciones iniciales, y estado inicial de la auditoría seleccionado entre no iniciado, en proceso, en pausa, o finalizado. Al momento del registro, el sistema debe validar el formato de correo electrónico y el formato de las direcciones IP. La auditoría debe quedar almacenada en el sistema con un identificador único.

- ➔ RF_03: El usuario con rol de auditor y administrador deben poder comentar observaciones adicionales en una auditoría seleccionada. Cada observación debe quedar registrada en un historial visible y no modificable. El historial debe mostrar, por cada entrada, la fecha y hora de creación, el nombre del auditor que escribió la observación, un título corto descriptivo, y la descripción completa del comentario. Las observaciones deben ordenarse cronológicamente, y no deben poder editarse ni eliminarse por ningún rol una vez guardadas.

- ➔ RF_04: El usuario con rol de auditor y administrador debe poder ejecutar un escaneo de red agresivo utilizando la herramienta Nmap sobre las direcciones IP registradas previamente en una auditoría, por lo que deberá seleccionar una IP proveída en el registro. El escaneo debe realizarse de forma local desde la aplicación de escritorio mediante comandos del sistema operativo, y debe devolver los resultados de puertos abiertos, servicios activos, versiones detectadas, y posibles vulnerabilidades si están disponibles. El resultado del escaneo debe quedar asociado directamente a

la auditoría desde la cual se ejecutó. El sistema debe mostrar al usuario un mensaje claro sobre el progreso y el resultado final del escaneo.

➔ RF_07: El sistema debe permitir al usuario visualizar los detalles completos de una auditoría específica seleccionada desde el panel general. Esta vista debe mostrar toda la información registrada durante el proceso de creación de la auditoría (según lo definido en RF_02), incluyendo nombre del cliente, nombre del contacto, correo, teléfono, IPs objetivo, observaciones iniciales, y el estado actual de la auditoría. Adicionalmente, esta vista debe integrar:

- El historial de observaciones asociadas a la auditoría, en orden cronológico, con autor, fecha, título y descripción (según RF_03).
- La sección de escaneo con Nmap, mostrando los resultados obtenidos, la fecha de ejecución y la bitácora generada por IA si existe (según RF_04, RF_05, y RF_06).
- El estado actual de la auditoría, con indicación visual clara (según RF_12).

La interfaz debe organizar esta información de forma clara y segmentada, permitiendo al usuario navegar fácilmente entre secciones.

➔ RF_08: El sistema debe ser capaz de permitir la creación de cuentas que cumplan roles específicos dentro de la aplicación. Estos roles tendrán determinados permisos que cumpliran cierto objetivo, el primero y más importante será el administrador, aquel que tiene permisos sobre todas las auditorías tanto presentes como pasadas, es capaz de asignar por sí mismo roles a las demás cuentas y tiene control sobre la base de datos. El Inspector tiene la función de presentar los hallazgos, recomendar soluciones y asignar la severidad en los informes para los clientes. El Inspector es el encargado de asignar los casos a los auditores, además de ser el que tiene la autoridad para enviar el informe al cliente. Y por último, el cliente es aquel que necesita un informe, este sube la información necesaria.

➔ RF_09: El auditor e inspector deben ser capaces de encontrar determinados informes a través de una barra de búsquedas que discrimine entre el nombre del cliente, el estado de la auditoría, el tiempo límite de entrega y el responsable de la auditoría. Todo esto en función de los datos proporcionados y que están en la base de datos.

➔ RF_10: El inspector debe ser capaz de asignar auditorías a los auditores, esto a través de un apartado donde el inspector pueda ver la carga de trabajo y el nivel de experticia del auditor.

➔ RNF_05: La interfaz debe tener una estructura jerárquica con menú lateral o pestañas para facilitar el acceso a cada módulo.

SHOULD:

➔ RNF_02: El sistema debería poder ser escalable, es decir, tener una estructura que permita añadir más módulos con distintas funcionalidades en caso de necesitarlo.

➔ RNF_03: El sistema debería responder a las solicitudes de los usuarios en menos de 2 segundos, esto excluyendo a los escaneos exhaustivos y otro tipo de análisis que por su estructura necesitan de mayor tiempo para la respuesta.

- ➔ RF_05: El usuario con rol de auditor y administrador debe poder tomar como entrada la salida del escaneo Nmap de una auditoría, y generar automáticamente una bitácora cualitativa mediante el uso de la API de OpenAI. El análisis debe basarse en un prompt predefinido que incluya el resumen de puertos abiertos, servicios y versiones, y debe generar un texto técnico que interprete el resultado desde la perspectiva de seguridad informática. El análisis generado debe quedar asociado a la auditoría como una observación automática, incluyendo fecha de generación y referencia al uso de IA. Esta operación requiere conexión a internet activa.

- ➔ RF_11: El administrador debería ser capaz de ver las auditorías ya acabadas, como un tipo de backup para ver relaciones y paralelismos con las auditorías pendientes, este, al igual que el RF_09 manejaría una barra de búsqueda que discrimine según los datos que la auditoría tiene.
- ➔ RF_12: El auditor e inspector podrían añadir etiquetas en las auditorías que muestren gráficamente el estado del proceso de la misma, las etiquetas serían:
    - ◆ En curso: La auditoría ya ha sido designada por el inspector, y el auditor ya está trabajando en el caso
    - ◆ Pendiente: La auditoría no ha sido asignada
    - ◆ Finalizado: El auditor considera que la auditoría ha sido resuelta, así que la pone como finalizada para que el inspector la revise y dé el aval para entregarlo
    - ◆ Archivado: La auditoría ha sido entregada al cliente

COULD:

- ➔ RF_06: El usuario con rol de auditor y administrador debe poder exportar los resultados de una auditoría a un archivo en formato PDF. El archivo PDF debe incluir los siguientes elementos: datos del cliente (nombre, contacto, correo, teléfono), resumen del escaneo realizado con Nmap (puertos abiertos, servicios detectados, versiones), bitácora cualitativa generada por la IA, listado de observaciones registradas por los auditores, y el estado actual de la auditoría. El PDF debe guardarse en el sistema de archivos local con un nombre de archivo autogenerado que incluya el nombre del cliente y la fecha. La exportación debe ser accesible solo si el escaneo y el análisis ya han sido realizados.

- ➔ RF_14: El usuario con rol de cliente debe poder acceder a una sección exclusiva del sistema donde pueda enviar una solicitud formal para realizar una auditoría. El formulario de solicitud debe incluir los siguientes campos obligatorios: nombre del negocio, nombre del contacto responsable, correo electrónico, número telefónico, una o varias direcciones IP del sistema objetivo, y un campo de observaciones.
  El sistema debe validar el formato del correo y las IPs antes de permitir el envío.
  Una vez enviada la solicitud, esta debe quedar registrada internamente para revisión por parte del equipo auditor y marcada como “pendiente de aprobación”. Se debe notificar al equipo mediante el sistema (o correo si se desea extender) que hay una nueva solicitud de auditoría.

➔ RF_13: Después de realizar una auditoría informática, va a ser necesario guardar los detalles relevantes encontrados. Esto se puede lograr asignando a una auditoría un identificador, con el cuál se puede encontrar la descripción asociada, el auditor responsable de la revisión, la severidad de los hallazgos (con argumentos de apoyo), y evidencias en la aplicación de ello. Con esto se busca realizar una organización de los hallazgos, y planificar futuras decisiones.

WON'T:

➔ RNF_04: El sistema debe poder aceptar archivos corruptos y dañados sin dar errores, esto debido a que es un comportamiento esperable dentro del contexto del sistema.
➔ RF_15: La aplicación podría contar con un apartado en el cuál se pueda visualizar el estado de criptomonedas robustas, que reflejen el estado general del mercado, como: Bitcoin (BTC), Ethereum (ETH), Cardano (ADA) y Solana (SOL). Esto se podría tomar de la página de Binance, y visualizar con gráficas de precio promedio, máximo, mínimo y variabilidad contra el tiempo. Este apartado también contaría con filtros de tiempo, para que el usuario pueda navegar en el periodo de interés.
➔ RF_16: La aplicación cuenta con un apartado dedicado a la planificación de futuras auditorías. En esta sección se buscaría contextualizar gráficamente el plazo, información relevante y requerimientos de una auditoría pendiente. También, debería ser capaz de descartar aquellas ya completadas, y visualizar las siguientes. Con esto se buscaría organizar las operaciones futuras de los auditores, de modo que haya mayor control de plazo de trabajo y distribuciones de carga.

💅Estimación de Esfuerzo con Puntos de Fibonacci:

Requisito		Estimación	Argumento
RF_1 MUST	Consulta de auditorías registradas en un panel general.	2	Implica listar datos ya almacenados filtros básicos. Es sencillo de implem usando una tabla interactiva. No req lógica compleja ni integración externa
RF_2	Registro de una nueva auditoría que contiene nombre del negocio, número del contacto, correo electrónico del contacto, teléfono, IPs	2	Requiere validaciones de entrada, estructura de formulario, manejo de múltiples IPs y creación en la base de datos. Aunque básico, es más complejo que una simple consulta.

MUST	del sistema web, observaciones y estado de la auditoría (No iniciado, en proceso, en pausa o finalizado).		
RF_3 MUST	Historial no modificable de observaciones visible para cada auditoría registrada con características como fecha, auditor quien escribió la observación, título de la misma y la descripción.	3	Requiere almacenamiento de acciones en la base de datos, almacenando variables especiales como hora de inserción, (no edición/eliminación), orden cronológico, asociación con usuarios y persistencia de logs.
RF_4 MUST	Escaneo agresivo de puertos de la red a partir de la IP otorgada en el registro de la auditoría con la herramienta NMAP.	8	Involucra ejecutar procesos del sis desde Java, manejo de errores, pa de salida, y seguridad al eje comandos. También depende herramientas externas.
RF_5 SHOULD	Análisis de resultados generados por Nmap por medio de IA (Open AI) para devolver una bitácora cualitativa a partir de un prompt definido.	8	Involucra consumo de una API externa, autenticación con claves, envío de prompts y procesamiento de respuestas de lenguaje natural. Puede fallar si no hay conexión o por límites de uso.
RF_6 COULD	Exportación de resultados de NMAP a un formato PDF junto a la bitácora cualitativa generada por la IA de OpenAI.	5	Generar un PDF con contenido dinámico, estructura visual organizada, integración de datos de diferentes módulos y exportación local requiere una librería adicional, pero es manejable.
RF_7 MUST	Visualización de una auditoría específica donde se muestra toda la información suscrita al registro (RF_02). En este apartado se	3	Es compuesto. Centraliza la información ya obtenida en RF_02, RF_03, RF_04, RF_05 y RF_06. Aunque integra múltiples fuentes, no implica lógica compleja adicional.

	encuentra el historial de las observaciones (RF_03), el estado de la auditoría (RF_12) y la sección de escaneo con NMAP (RF_04, RF_05 y RF_06).		
RF_8 MUST	Registro de nuevos usuarios con determinados roles (“Auditor”, “Inspector”, “Administrador”, “Cliente”). Cada rol debe tener determinados permisos sobre acciones en la aplicación.	8	Aunque el concepto no es difícil, el realizar el login, generar la base de datos con los permisos necesarios y tener los test necesarios para que no hayan problemas de autenticación pueden ser demorados
RF_9 MUST	Búsqueda de auditorías registradas en la base de datos a través del nombre del cliente (referente usualmente al nombre de la empresa), el responsable asignado, el estado de la auditoría y el tiempo límite de entrega de esta.	3	Con la base de datos hecha, solo es llamarla con ciertos permisos, lo que no genera ninguna dificultad, sin embargo se gasta algo de tiempo en que tenga una interfaz intuitiva para el usuario
RF_10 MUST	Asignación de auditorías a determinados roles con permisos de edición. Además, de la asignación de la auditoría a más de un rol, y la asignación a más de una persona dentro del rol.	3	Es un cambio de estado simple en la base de datos, sin embargo contiene información que se estará moviendo constantemente por lo que se debe implementar seguridad e integridad a los datos
RF_11 SHOULD	Acceso al historial de auditorías terminadas, esto a través de información asociada a las auditorías, como lo es la fecha de finalización, el responsable, el estado final y el nombre del cliente.	1	Una consulta de datos pasados, que se guardan y que ya estaban en la base de datos, solo que con un estado determinado

RF_12 SHOULD	Gestión del estado de la auditoría mediante etiquetas visuales como lo son “En curso”, ”Pendiente”, “Finalizado”, “Archivado”. Con posibilidad de actualización por medio de un rol con los permisos adecuados.	5	La dificultad de este requisito recae en el diseño de las etiquetas, además de que sean fácilmente intercambiables en el caso de que haya un cambio de estado
RF_13 COULD	Registro manual de hallazgos relacionados con la auditoría, donde incluye datos como nombre del hallazgo, descripción, severidad (bajo, medio, alto, crítico) recomendación y evidencia	2	Para lograr este objetivo se podría realizar una interfaz gráfica de usuario, con la que se recoge información sobre la auditoría (el auditor, la descripción, la gravedad, evidencias y recomendaciones). Después, esta información se podría agregar a la base de datos que engloba este tipo de información, y complementar con la información relevante faltante, como la fecha en que se realizó y el identificador del hallazgo, para evitar colisiones con otros hallazgos.
RF_14 COULD	Sección donde un cliente manda una solicitud para la realización de una auditoría, que contiene los campos de información como nombre del negocio, número del contacto, correo electrónico del contacto, teléfono, IPs del sistema web, observaciones	2	Esta funcionalidad se puede materializar con una interfaz gráfica, con cuadros de texto en que se pueda ingresar la información del cliente necesaria. Posteriormente, la estructura de esta información es verificada, y notificada a los asesores, para que puedan evaluar la situación y contactar de vuelta al cliente si es necesario.
RF_15 WON’T	Consulta y resumen gráfico de criptomonedas en Binance	5	Esta tarea requeriría encontrar una forma de realizar consultas a un servidor. Un obstáculo podrían ser las políticas cambiantes de páginas que ofrecen este servicio. Una vez realizada esta consulta, se pueden realizar gráficas con las librerías pertinentes en java.

RF_16 WON'T	Visualización gráfica de auditorías futuras	3	Esta tarea requiere tomar de una base de datos con las auditorías pendientes, la información más relevante (como el cliente asociado y la situación a manejar), en un calendario gráfico, coherente con los plazos asociados a la información anterior. Esta sección también debería organizar estos plazos, de modo que todos sean visibles y distinguibles, mediante colores.
RNF_1 MUST	El sistema deberá funcionar con sistemas de escritorio, y todas sus funcionalidades se deben llevar a cabo en este entorno específico.	3	La compatibilidad de escritorio es algo que se trabajara en cierta medida en cada requisito, por lo tanto el trabajo específico para asegurar la compatibilidad de escritorio es bastante pequeño
RNF_2 SHOULD	El sistema debería poder ser escalable, es decir, tener una estructura que permita añadir más módulos con distintas funcionalidades en caso de necesitarlo.	3	Esta tarea es moderadamente sencilla ya que solo requerirá de tener un código escalable y de una interfaz modular que permita fácilmente añadir más módulos.
RNF_3 SHOULD	El sistema debería responder a las solicitudes de los usuarios en menos de 2 segundos, esto excluyendo a los escaneos exhaustivos y otro tipo de análisis que por su estructura necesitan de mayor tiempo para la respuesta.	5	Para lograr este requisito necesitaremos un considerable esfuerzo en hacer código limpio y en optimización de todos los sistemas de la aplicación.
RNF_4 WON´T	El sistema debe poder aceptar archivos corruptos y dañados sin dar errores, esto debido a que es un comportamiento esperable dentro del contexto del sistema.	13	Esto requiere de un amplio conocimiento sobre los binarios y demás conceptos de bajo nivel, que el equipo no tiene actualmente.
RNF_5	La interfaz debe tener una estructura	2	Esta tarea es sencilla ya que solo requiere de hacer un diseño de la UI

MUST	jerárquica con menú lateral o pestañas para facilitar el acceso a cada módulo.		acorde al requisito, y de la implementación de este menú lateral junto a la integración de este con el resto del sistema.

🤢Reglas de negocio:

Las **reglas de negocio** definen las políticas internas, restricciones y condiciones lógicas que deben cumplirse durante el funcionamiento del sistema Spynet. Estas reglas garantizan la integridad, consistencia y trazabilidad del proceso de auditoría informática.

1. **Las observaciones asociadas a una auditoría no pueden ser modificadas ni eliminadas una vez registradas.**
   Esto asegura la trazabilidad de los comentarios y evita manipulaciones posteriores al registro.

2. **Cada auditoría debe estar obligatoriamente vinculada a un cliente registrado.**
   No se pueden crear auditorías sin un conjunto mínimo de datos del cliente: nombre del negocio, correo de contacto, número de teléfono y al menos una IP objetivo.

3. **El estado de la auditoría debe seguir una de las siguientes etapas definidas: No iniciado, En proceso, En pausa, Finalizado.**
   Este flujo de estados estandariza el seguimiento y evita inconsistencias en la interpretación del progreso.

4. **Solo los usuarios con rol de auditor (empleado) pueden generar escaneos, registrar observaciones o ejecutar análisis con IA.**
   Esto restringe las acciones técnicas críticas a personal autorizado.

5. **Las solicitudes externas de auditoría (RF_14) no constituyen auditorías válidas hasta ser aprobadas por un usuario auditor.**
   Se almacenan en un estado de "pendiente de aprobación" hasta que se confirme su validez y se complete el registro completo.

6. **Cada escaneo con Nmap debe asociarse automáticamente a la auditoría que lo origina y almacenarse con marca de tiempo.**
   Esto permite auditorías repetidas con evidencias ordenadas cronológicamente.

7. **El sistema debe permitir múltiples escaneos por auditoría, pero solo el último podrá ser analizado por la IA.**

Esto optimiza el uso de la API de OpenAI y evita gastos innecesarios.

8. **La información de clientes y auditorías debe estar protegida bajo un modelo de acceso restringido por roles.**
   Usuarios no autenticados o clientes externos no pueden acceder al historial técnico interno.

9. **Las IPs registradas deben validarse para tener un formato IPv4 válido y evitar entradas erróneas.**
   Esto protege al sistema de errores de ejecución o escaneos inválidos.

🥵Alcance del Sistema

El sistema tiene que poder tener una funcionalidad básica que permita el registro, visualización y mantenimiento de auditorías. Así mismo el uso de módulos como nmap y su implementación con IA.