PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

Unidad Académica de Formación Técnica y Tecnológica – PUCE TEC

Sistema de Automatización para Auditoria de redes, puertos y vulnerabilidades en Windows

Proyecto de titulación previo a la obtención del título de:

Tecnólogo Superior en Desarrollo de Software

Autores:

Fausto Jonathan Navarrete Lascano

Juan Francisco Quishpe Piñaloza

Tutor:

Patricio Omar Alvear Granda

Quito, Ecuador

2025

Dedicatoria

A mis padres, Juan Quishpe y Martha Piñaloza,
quienes, con su esfuerzo, dedicación y valores inquebrantables,
me han dado las herramientas esenciales para forjar mi camino.

Su ejemplo ha sido mi mayor guía, y el apoyo constante que me han brindado me ha permitido alcanzar cada logro con convicción y realismo. Este trabajo es, en parte, el reflejo de todo lo que me han enseñado: la importancia del conocimiento, del trabajo arduo, y de mantenerse firme frente a la realidad de la vida.

Gracias por creer en mí, siempre.

A mis padres, Fausto Wilson Navarrete y Amparo Lascano,
por ser mi constante en un mundo de altibajos,
por estar a mi lado sin importar las decisiones que tomara ni las circunstancias que enfrentara.

Han sido mi refugio en los momentos más difíciles y mi impulso en cada paso adelante. Esta dedicatoria es un reconocimiento a su inquebrantable apoyo, que ha sido la base sobre la cual he construido mis logros y enfrentado mis desafíos con pragmatismo y determinación.

Gracias por estar siempre ahí, sin condiciones.

Tabla de contenidos

Lista de tablas

Lista de figuras

DECLARACIÓN y AUTORIZACIÓN

Yo, **FAUSTO JONATHAN NAVARRETE LASCANO** con C.I. 1722048814, autor del trabajo de titulación intitulado: "**SISTEMA DE AUTOMATIZACIÓN PARA AUDITORIA DE REDES, PUERTOS Y VULNERABILIDADES EN WINDOWS"**, previa a la obtención del título de **TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE** en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través de sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de Universidad.

Quito, febrero de 2025

Fausto Jonathan Navarrete Lascano

C.I. 1722048814

DECLARACIÓN y AUTORIZACIÓN

Yo, **Juan Francisco Quishpe Piñaloza** con C.I. 1752608271 respectivamente, autor del trabajo de titulación intitulado: "**SISTEMA DE AUTOMATIZACIÓN PARA AUDITORIA DE REDES, PUERTOS Y VULNERABILIDADES EN WINDOWS"**, previa a la obtención del título de **TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE** en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través de sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de Universidad.

Quito, febrero de 2025

Juan Francisco Quishpe Piñaloza

C.I. 17562608271

Agradecimientos

Queremos expresar nuestro más sincero agradecimiento a todos los profesores que nos hemos encontrado a lo largo de nuestra carrera. Su dedicación, paciencia y claridad en la enseñanza han sido fundamentales para que podamos asimilar los conocimientos y competencias que hoy poseemos. Cada clase, cada lección y cada consejo impartido nos ha preparado para afrontar con confianza los retos que enfrentaremos en nuestra vida profesional.

Asimismo, extendemos nuestro reconocimiento a la institución, cuyo compromiso con la educación y la formación continua ha sido un pilar importante en nuestro crecimiento académico. Gracias por ofrecer nuevas carreras que nos preparan para enfrentar los desafíos de un mundo cada vez más dominado por las nuevas tecnologías. Su ardua labor ha hecho posible que hoy estemos listos para contribuir con nuestras habilidades en un entorno profesional cambiante y demandante.

En especial, queremos expresar nuestro profundo agradecimiento a nuestros primeros profesores, Dianita, Jonathan y Patricio. Ustedes fueron los primeros rostros que vimos al iniciar este camino, y con su guía, motivación y entrega nos brindaron las ganas de seguir adelante. No solo los recordamos como nuestros primeros mentores, sino también como futuros colegas a quienes siempre llevaremos con gratitud y aprecio. Los momentos compartidos en las aulas y el conocimiento transmitido quedarán grabados en nuestra memoria como un valioso legado. Gracias, de corazón, por marcar el inicio de este viaje que ha transformado nuestras vidas.

Introducción

El presente proyecto de titulación tiene como objetivo la creación de un sistema automatizado para la gestión de permisos y la detección de vulnerabilidades en dispositivos que utilizan el sistema operativo Windows. La propuesta surge de la necesidad de contar con herramientas eficientes que permitan auditar y verificar el funcionamiento correcto de las computadoras a un nivel profundo, identificando posibles riesgos de seguridad que puedan comprometer tanto el rendimiento como la integridad de la información almacenada en los dispositivos.

El sistema desarrollado en este proyecto proporcionará una herramienta gratuita para el escaneo exhaustivo del sistema operativo, permitiendo detectar vulnerabilidades y software malintencionado que pueda estar presente en la computadora. A diferencia de otras soluciones que requieren conocimientos técnicos avanzados, esta herramienta ha sido diseñada para ser intuitiva y accesible, facilitando su uso para personas sin experiencia previa en el ámbito de la ciberseguridad o la administración de sistemas. De esta manera, los usuarios podrán verificar si sus computadoras están expuestas a amenazas, como la presencia de malware o accesos no autorizados que puedan aprovechar puertos abiertos para infiltrarse en el sistema.

La necesidad de este tipo de soluciones es cada vez más evidente, ya que la seguridad de la información se ha convertido en un componente esencial de la vida cotidiana. Con el constante avance hacia un entorno digital más complejo y conectado, conocido como Web 3.0, se hace indispensable contar con mecanismos que permitan prevenir ataques cibernéticos y garantizar la protección de los datos personales y corporativos. En este contexto, las herramientas gratuitas para la seguridad informática juegan un papel crucial, ya que ofrecen una primera línea de defensa para usuarios que de otra manera estarían expuestos a diversas formas de amenazas cibernéticas.

El proyecto también aborda la importancia de la ciberseguridad en la actualidad, destacando cómo la conectividad global ha incrementado tanto las oportunidades como los riesgos en el ámbito digital. La proliferación de dispositivos conectados y la creciente dependencia de la tecnología han dado lugar a nuevas formas de amenazas que exigen soluciones innovadoras y accesibles para todos. Este sistema automatizado no solo permitirá identificar vulnerabilidades presentes en el sistema operativo, sino que también contribuirá a la educación del usuario final, brindándole información clara y concisa sobre los problemas detectados y las medidas que pueden tomarse para mejorar la seguridad de sus dispositivos.

En conclusión, este proyecto no solo proporcionará una herramienta efectiva para el escaneo de sistemas, sino que también busca fomentar una mayor conciencia sobre la importancia de la ciberseguridad en la era digital. Al ofrecer una solución accesible y gratuita, se espera que más personas puedan proteger sus dispositivos sin necesidad de conocimientos técnicos avanzados, promoviendo un entorno digital más seguro y confiable para todos, para que a posterior de ser necesario se pueda adquirir una solución más robusta.

Capítulo I

Levantamiento de Requisitos y Diseño del Sistema

1.1. Requisitos Funcionales

1.1.1. Escaneo y Auditoría de Seguridad

Descripción: El usuario podrá observar una vista preliminar de cómo se encuentra su dispositivo, que al ser la primera vez no devolverá información relevante, una vez ya realizado la primera revisión, arrojará datos relevantes para su posible solución o auditoria si así lo amerita.
Entradas: • Fecha de Escaneo • Usuario que Realizó el Escaneo • Hora de Inicio de Escaneo • Hora de Finalización de Escaneo • Estado Inicial del Dispositivo • Estado Final del Dispositivo • Detalles Iniciales del Escaneo • Detalles Finales del Escaneo
Proceso: • El sistema verifica el estado inicial del dispositivo previo a ejecutar un scanner total, luego al presionar el botón para realizar el escáner, se ejecuta el proceso revisando todo el dispositivo para entregar un detalle de las posibles vulnerabilidades que posee la computadora y de necesitar ejecutar un cambio el usuario procederá a ejecutar el botón para arreglar mencionado problema.
Salidas: • Revisión realizada con éxito • Informe con datos de la revisión • Posibles Soluciones

Tabla 1 Escaneo y Auditoría de Seguridad

1.1.2. Interacción con el Sistema Operativo Windows

Descripción: El usuario a través de una interfaz intuitiva estará aplicando una serie de comandos y scripts que afectaran directamente al sistema operativo, para así poder realizar de forma automática las correcciones correspondientes encontradas en la auditoria o escaneo del sistema realizadas previamente
Entradas: • Información detallada de la Auditoria • Ajustes para realizar las correcciones • Verificación de Aplicación de correcciones • Estado del Dispositivo previo a aplicar las correcciones
Proceso: • El programa, una vez que finalizo el escaneo arroja datos necesarios para poder visualizarlos en la aplicación hasta que se cierre, estos datos podrán verse por el usuario admin, hasta que finalice el programa.
Salidas: • Corrección aplicada con éxito • Error al aplicar Solución • Mensaje de Confirmación de Ajustes

Tabla 2 Interacción con Sistema Operativo Windows

1.1.3. Gestión del Sistema

Descripción: El sistema requiere un módulo de gestión central que orqueste la comunicación entre los distintos componentes durante la ejecución del programa.
Entradas: • Estado de ejecución • Resultado del análisis de puertos • Respuestas de las APIs
Proceso: • El programa al realizar la revisión del sistema devolverá los problemas encontrados, así como también las posibles soluciones.
Salidas: • Interfaz actualizada con resultados en tiempo real. • Indicadores de progreso del sistema.

Tabla 3 Gestión del Sistema

1.1.4. Interfaz de Usuario

Descripción: El usuario admin mediante una interfaz gráfica podrá interactuar de manera rápida, fácil y eficiente con scripts y auditoria básica en su dispositivo Windows, para así aplicar de manera correcta soluciones a posibles vulnerabilidades de su sistema.
Entradas: • Datos de Usuario Admin • Validación de Usuario Admin • Interacción del Usuario
Proceso: • El Programa al iniciar presentará una pantalla de registro en la que el usuario deberá introducir datos admin para correr el programa.
Salidas: • Información almacenada dentro de la aplicación.

Tabla 4 Interfaz de Usuario

1.2. Requisitos No Funcionales

1.2.1. Rendimiento y Eficiencia:

- Optimizar los scripts para asegurar un rendimiento eficiente durante las auditorías, evitando la saturación del sistema.
- Asegurar tiempos de respuesta rápidos en la interfaz, facilitando la gestión y visualización de resultados.

1.2.2. Usabilidad y Accesibilidad:

- Implementar diseño modular para facilitar la futura integración de nuevos módulos o funcionalidades.

1.2.3. Seguridad:

- Garantizar que las herramientas del sistema no comprometan la seguridad de los dispositivos en los que se instalan, evitando la exposición de datos sensibles.
- Implementar medidas de seguridad que protejan los resultados de las auditorías y la información sensible de los usuarios.

1.3. Diseño del Sistema

1.3.1. Diseño de la Interfaz de Usuario (UI)

- **Elementos de la Interfaz:**
  - **Panel Principal:** Permite al usuario iniciar auditorías, gestionar configuraciones, y acceder a informes previos.

- **Vista de Resultados:** Presenta resultados detallados de las auditorías mediante gráficos, resúmenes visuales y listas detalladas para facilitar la interpretación de datos.
- **Opciones de Configuración:** Sección para ajustar parámetros de auditoría y personalizar la ejecución de scripts.
- **Botones de Acción:**
  - *"Iniciar Escaneo":* Ejecuta las auditorías de manera automatizada.
  - *"Información sobre módulo":* Accede a la documentación de cada módulo.
  - *"Configuraciones":* Permite modificar configuraciones y ajustes del sistema.

A continuación, se muestra un diseño preliminar de la aplicación

Ilustración 1. Módulo pantalla inicial

Ilustración 2. Pantalla de Inicio

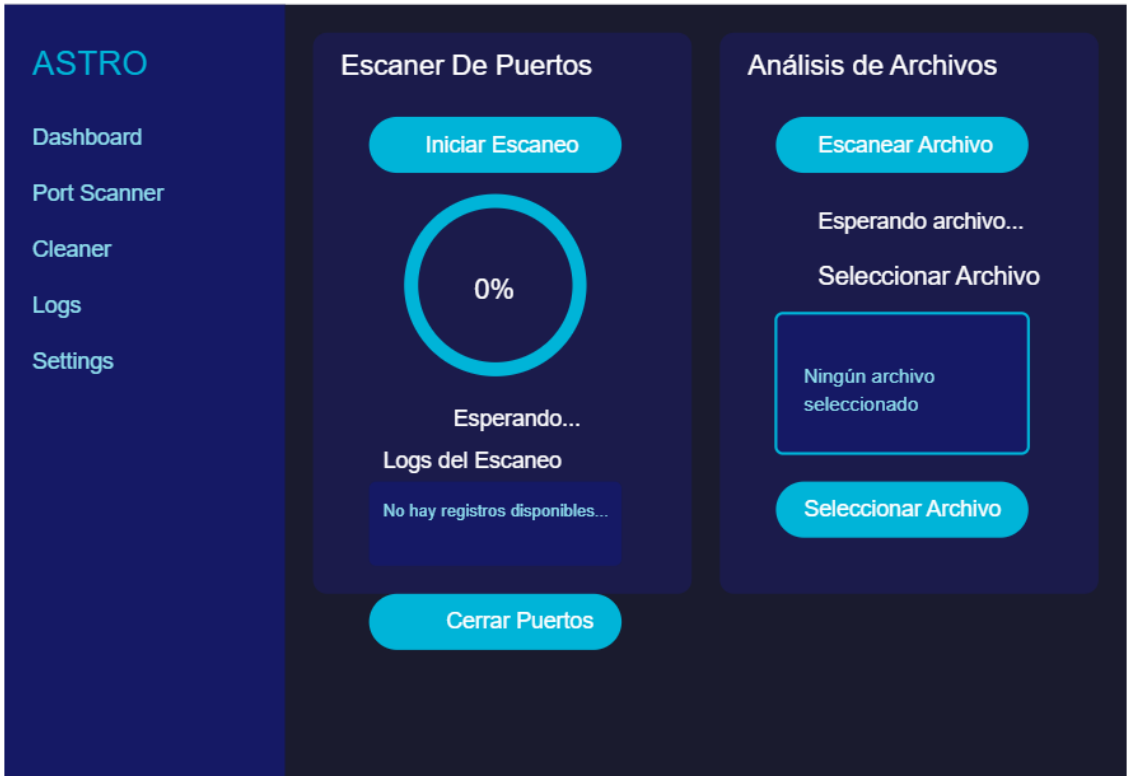

Ilustración 3. Pantalla Principal

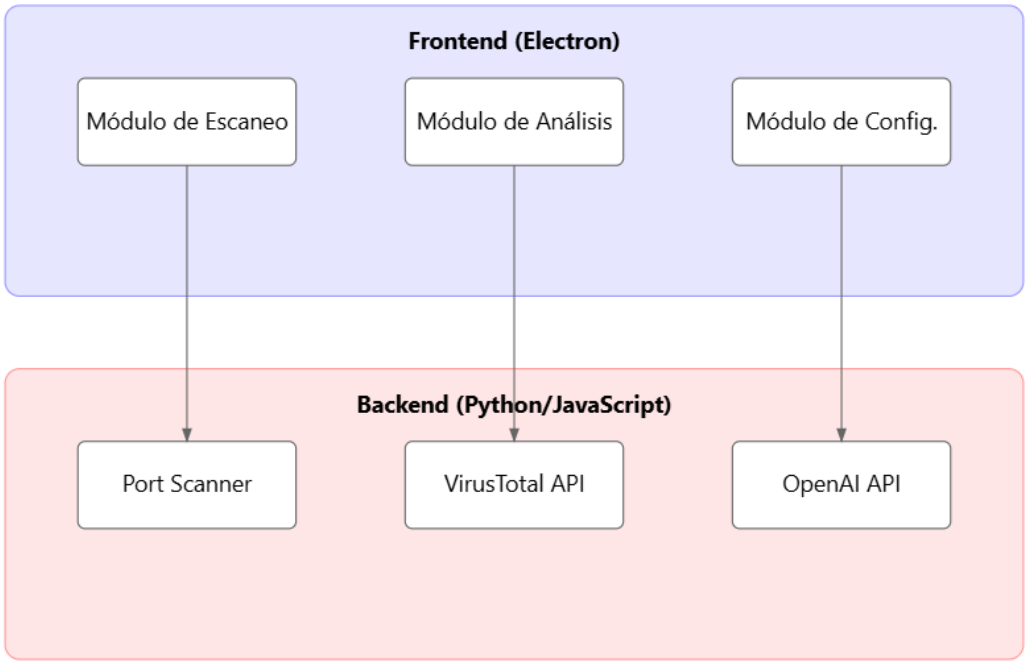

Ilustración 4. Sistema Auditoria de Seguridad

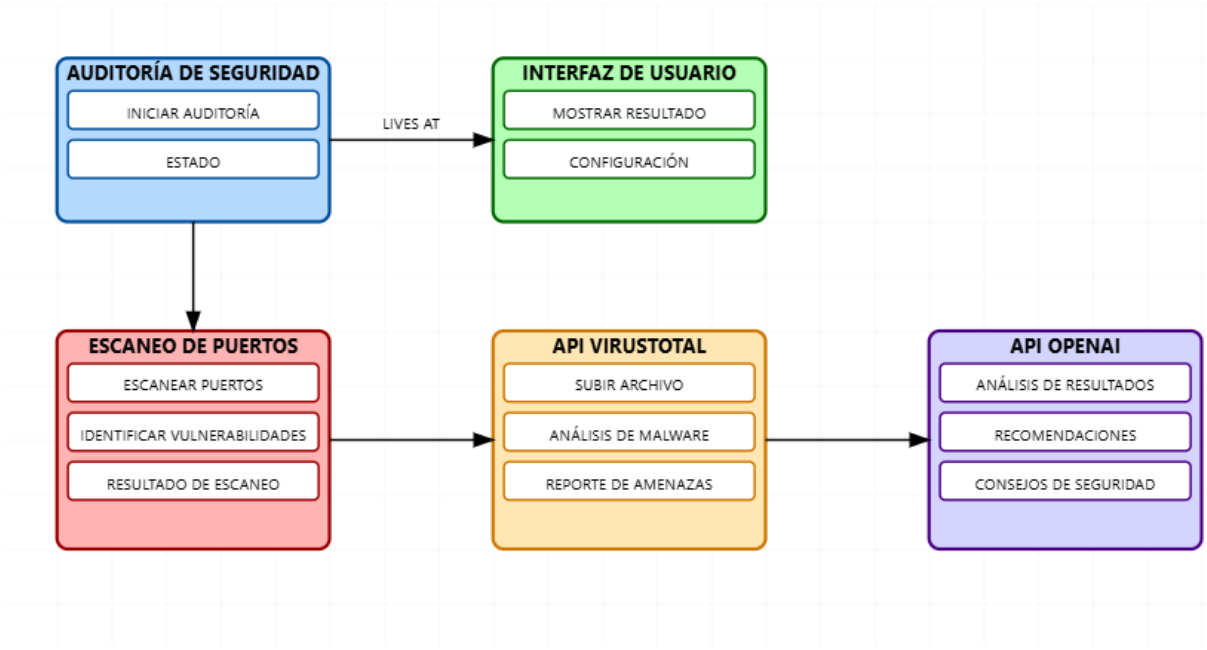

Ilustración 5. Diagrama del Sistema de Auditorias

Capítulo II

Construcción del Sistema

2.1 Construcción del Sistema Frontend

2.1,1. Estándares de Construcción

Como señalan Sommerville y Sawyer (2021), el desarrollo modular permite construir aplicaciones complejas mediante la integración de componentes más pequeños y manejables.

Para todo el proyecto todas las funciones, variables y clases de CSS serán escritas en inglés.

En el caso de las funciones estarán escritas en "camelCase".

En el caso de las variables y clases de CSS, estarán escritas en "snake_case"

2.1.2. Definición de Páginas

La aplicación se estructura en tres páginas principales. Cada una de ellas está organizada y codificada de manera modular para asegurar la eficiencia y el mantenimiento del código. Las páginas son:

- Principal Screen
- Port Scanner
- Virus Total Info

2.1.3. Codificación de las Páginas

Principal Screen

La pantalla principal está estructurada con el ScannerModule, que agrupa dos módulos secundarios: PortScanner y FileAnalyzer. Esta estructura permite gestionar y visualizar los componentes junto con sus estilos de manera eficiente.

ScannerModule

Este módulo contiene los módulos PortScanner y FileAnalyzer, que se encargan de las funciones principales del sistema.

PortScanner

En este módulo se manejan los datos y funciones relacionados con el escaneo de puertos. Utiliza las APIs proporcionadas por el backend para interactuar con el sistema. Las funciones principales son:

- Iniciar el escaneo de puertos del sistema.
- Visualizar el progreso en tiempo real del escaneo.
- Mostrar los resultados del escaneo.
- Cerrar puertos innecesarios.

Este módulo incluye los componentes:

ProgressCircle: Para mostrar el progreso del escaneo mediante un gráfico circular.

ScanLogs: Para visualizar los registros del escaneo.

PortControl: Para controlar los puertos en tiempo real.

FileAnalyzer

Este módulo maneja el análisis de archivos utilizando la API de VirusTotal. Se encarga de procesar y mostrar los resultados de la evaluación de archivos en tiempo real.

InformationModule

Este módulo contiene dos componentes principales que proporcionan documentación detallada:

- **Port Scanner Info**: Ofrece información técnica sobre los puertos y protocolos, la funcionalidad del escáner, medidas de seguridad recomendadas y guías de uso.
- **VirusTotal Info**: Presenta documentación sobre el proceso de análisis de archivos, la integración con VirusTotal, la interpretación de resultados y advertencias de seguridad.

SecurityAdvice

Este módulo implementa un sistema de consejos de seguridad. Las funciones principales incluyen:

- Botón flotante para activar el sistema.
- Ventana modal para mostrar los consejos de seguridad.
- Integración con IA para generar consejos de seguridad personalizados.
- Sistema de cierre interactivo para la ventana modal.

2.1.4. Codificación de los Componentes

SidebarComponent

El módulo de navegación lateral permite acceder a las diferentes secciones de la aplicación. Sus elementos principales incluyen:

- Logotipo del sistema "ASTRO".
- Menú de navegación con enlaces a las principales páginas de la aplicación.
- Indicadores visuales para la sección activa.

MainContentModule

Este módulo organiza y presenta la información en el área principal de trabajo. Los elementos principales son:

- **CardSystem**: Muestra información importante en tarjetas.
- **GridLayout**: Organiza los componentes en un diseño en cuadrícula.
- **ContentArea**: Define el área donde se presentan los resultados y las interacciones principales del usuario.

2.1.5. Funciones y Eventos

refreshProgress

Esta función actualiza visualmente el progreso del escaneo de puertos mediante la manipulación de elementos SVG.

Eventos

El evento ipcRenderer.on('scan-results') maneja los datos devueltos por el backend y actualiza dinámicamente el contenido en la interfaz.

2.1.6. Estilos Generales

Los estilos generales definen el aspecto de la aplicación y aseguran una presentación coherente en toda la plataforma. Estos son los principales estilos utilizados:

Resetbásico:

El reset básico elimina márgenes, paddings y establece un modelo de caja uniforme utilizando box-sizing: border-box.

Estilo del cuerpo:

- Fondo de color **#1E2A38**.
- Fuente Arial, colores de texto #FFF o #ccc.
- Altura mínima definida para asegurar que el contenido ocupe toda la pantalla.

2.1.7. Estilos Específicos de las Páginas

PrincipalScreenStyles

Los elementos principales incluyen:

Sidebar (.sidebar):

- Fondo de color #2B3A4A.
- Logotipo y secciones activas con color #3CD1C2.
- Transición de color en enlaces al pasar el cursor.

Main Content (.main-content):

- Distribución del contenido utilizando **flexbox**.

Cards (.card, .large-card, .medium-card):

- Bordes redondeados, sombras ligeras y espacio definido para el contenido.

Progress Circle (.progress-circle):

- Utiliza gráficos SVG para representar el progreso del escaneo.
- Cambios en stroke-dashoffset para animaciones de progreso.

Floating Button (#floatingButton):

- Botón flotante posicionado en la esquina inferior derecha.
- Escala y cambio de color al pasar el cursor.

PortScannerStyles

Los elementos principales incluyen:

Container (.container):

- Márgenes y paddings configurados para centrar el contenido.
- Fondo #2B3A4A con bordes redondeados y sombra.

Headers (h1, h2):

- Títulos con color **#3CD1C2**.
- Tamaños de fuente definidos para diferenciar los encabezados.

Buttons (.button, .button-secondary):

- Bordes redondeados y colores resaltados.
- Transiciones de color para eventos hover.

Warnings (.warning):

- Fondo en color **#B33A3A** con texto en blanco.

Lists (ul, li):

- Elementos de lista con viñetas estilizadas en **#3CD1C2**.

VirusTotalInfoStyles(virust_total_info_style.css)

Los elementos principales incluyen:

Container (.container):

- Ancho máximo de 1200px y fondo de color #2B3A4A.

Headers (h1, h2):

- Colores consistentes con los otros módulos y tamaños diferenciados para la jerarquía.

Buttons (.button, .regresar):

- Botones posicionados para navegación, con transiciones en eventos hover.

Warnings (.warning):

- Fondo y estilo de texto similar al módulo del escáner de puertos.

Construcción del Sistema Backend

2.2.1. Estándares de construcción

"Para la construcción del backend del proyecto Astro, se implementó una metodología de desarrollo modular. Esto permite la creación de componentes reutilizables que encapsulan funcionalidades específicas" (Martin, 2019, p. 157).

Todas las clases, funciones y variables están escritas en inglés para mantener consistencia en el código y facilitar su comprensión.

Se definieron los siguientes estándares:

- Las clases utilizan el formato de nomenclatura PascalCase.
- Las funciones y variables emplean snake_case.
- Las constantes están en UPPER_CASE.

2.2.2. Estructura y codificación

El backend del sistema Astro se encuentra estructurado en módulos especializados, cada uno con funciones definidas. A continuación, se describen los principales módulos y su funcionalidad.

Port Scanner Module

El módulo principal denominado port_scanner gestiona el escaneo y análisis de puertos en el sistema. Este módulo utiliza programación multihilo para mejorar el rendimiento en el análisis de redes.

El módulo incluye las siguientes funcionalidades:

1. Escaneo de puertos mediante la clase PortScanner, que permite identificar puertos abiertos en un rango definido por el usuario.
- Utiliza la clase Queue para administrar tareas en paralelo, incrementando la velocidad de escaneo mientras el sistema esté en funcionamiento.
- Almacena los puertos abiertos en una lista para su posterior procesamiento mientras este siga en abierto el sistema.

2. Interacción con el sistema mediante sockets, lo que permite la conexión y comprobación de cada puerto.
3. El cierre de puertos innecesarios se gestiona mediante la clase PortCloser, que utiliza comandos del sistema operativo para bloquear los puertos abiertos detectados durante el escaneo.

Port Closer Module

El módulo port_closer implementa un sistema de seguridad activa que permite bloquear puertos no esenciales en el sistema objetivo. La funcionalidad principal de este módulo incluye:

- Identificación de puertos críticos que deben permanecer abiertos, como los puertos 22 (SSH), 80 (HTTP) y 443 (HTTPS).
- Cierre de puertos no esenciales utilizando comandos del sistema operativo, implementados mediante subprocesos para ejecutar comandos en PowerShell.
- Gestión de errores durante el bloqueo de puertos, garantizando que el sistema mantenga su integridad.
- Generación de resultados en formato JSON, lo que permite su integración con otros módulos del sistema.

Virus Scanner Module

El módulo virus_scanner gestiona el análisis de archivos mediante la integración con la API de VirusTotal. Sus responsabilidades incluyen:

- Verificación de la existencia del archivo en la ruta proporcionada.
- Envío del archivo a la API de VirusTotal para su análisis.
- Consulta periódica del estado del análisis hasta que este sea completado.

- Procesamiento de los resultados del análisis, presentando estadísticas sobre detecciones de malware.
- Gestión de errores, como fallos en la conexión o respuesta inesperada de la API.

Cybersecurity Advice Module

El módulo cybersecurity_tips utiliza inteligencia artificial para proporcionar recomendaciones prácticas sobre ciberseguridad. La funcionalidad principal de este módulo incluye:

- Generación de consejos mediante la integración con el modelo GPT-3.5.
- Personalización de las respuestas para garantizar que los consejos sean claros y breves.
- Gestión de excepciones en caso de errores durante la comunicación con la API de OpenAI.
- Devolución de los resultados en formato JSON, facilitando su visualización en la interfaz del usuario.

Anexos:

https://github.com/Faustozd/proyectoastro

Capítulo III

Pruebas y Estabilización

3.1. Pruebas Frontend

N°: 1 **Caso de prueba FrontEnd Nombre Caso de Prueba:** Test Pop- Up IA

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar que cuando el usuario hace clic en el botón para obtener un consejo de IA, se muestra correctamente un pop-up con el contenido esperado.	
Precondiciones	• El usuario está dentro de la aplicación una vez ejecutado el programa • La funcionalidad de IA debe estar disponible y operativa. • El botón para obtener el consejo de IA debe estar visible en la interfaz. • El botón utiliza una iconografía llamativa que invita a ser presionada
Pasos para seguir	• Abrir la aplicación y navegar hasta la sección donde se encuentra el botón de obtención de consejo de IA. • Se hace clic en el botón con la iconografía llamativa para obtener el consejo. • Se verifica que se muestra un pop-up emergente. • Revisamos que el contenido del pop-up incluya el consejo generado por la IA. • Confirmamos que el pop-up se pueda cerrar correctamente.
Datos de Entrada	• Acción del usuario: Clic en el botón con la iconografía llamativa para obtener consejo.
Resultados Esperados	• Se muestra un pop-up con el consejo generado por la IA. • El diseño y formato del pop-up son consistentes con el resto de la aplicación. • Se incluye una opción clara para cerrar el pop-up. • No hay errores visuales ni fallos en la funcionalidad.
Criterios de Aceptación / Rechazo	**Aceptación:** • El pop-up aparece correctamente tras hacer clic en el botón. • El consejo de IA se muestra sin errores. • El usuario puede cerrar el pop-up sin problemas. **Rechazo:** • El pop-up no aparece o aparece con errores. • El consejo de IA no se muestra o muestra información incorrecta. • El pop-up no se puede cerrar o genera fallos en la aplicación.
Desarrollador Asignado	Fausto Navarrete

Tabla 5Test Pop- Up IA

N°: 2 **Caso de prueba FrontEnd Nombre Caso de Prueba:** Test Módulo VirusTotal

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar que el módulo permita cargar un archivo y realizar el escaneo en busca de virus correctamente desde el frontend.	
Precondiciones	• El usuario está dentro de la aplicación una vez ejecutado el programa • La funcionalidad de carga y análisis debe estar operativa. • Se debe contar con un archivo de prueba válido para la carga.
Pasos para seguir	• Nos Dirigimos a la sección del módulo de análisis de archivos. • Hacer clic en el botón **"Seleccionar archivo"** y escogemos un archivo válido. • Confirmamos que el archivo se carga correctamente y que aparece su nombre en la interfaz. • Hacer clic en el botón **"Escanear Archivo"** para iniciar el análisis. • Una vez finalizado el escaneo no mostrara un TextLabel con la información sobre la auditoria del archivo.
Datos de Entrada	• Archivo de prueba: Puede ser un archivo limpio y otro con malware de prueba (como EICAR). • Acción del usuario: Carga del archivo y clic en el botón de escaneo.
Resultados Esperados	• El usuario puede cargar un archivo sin errores. • El nombre del archivo aparece correctamente en la interfaz. • Al hacer clic en "Escanear Archivo", se inicia el proceso sin fallos. • El sistema devuelve un mensaje claro sobre si el archivo tiene virus o está limpio.
Criterios de Aceptación / Rechazo	**Aceptación:** • El archivo se carga correctamente y se muestra en la interfaz. • Al escanear, se obtiene un resultado confiable sin errores. • Se verifican correctamente los archivos con virus. **Rechazo:** • No es posible cargar el archivo o se muestra un mensaje de error. • El archivo se carga pero no se refleja en la interfaz. • El botón "Escanear Archivo" no funciona o no genera respuesta. • No se muestra el resultado del análisis o presenta errores.
Desarrollador Asignado	Fausto Navarrete

Tabla 6 Test Módulo VirusTotal

N°: 3 **Caso de prueba FrontEnd Nombre Caso de Prueba:** Test Iniciar Escaneo

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar que al hacer clic en el botón "Iniciar Escaneo", se active correctamente el proceso de escaneo de puertos y se muestre el círculo de carga junto con el mensaje "Esperando los logs del escaneo".	
Precondiciones	• El usuario debe estar dentro de la aplicación. • El botón "Iniciar Escaneo" debe estar visible y habilitado. • La funcionalidad del escáner debe estar operativa.
Pasos para seguir	• Abrir la aplicación y navegar a la sección del escáner de puertos. • Hacer clic en el botón "Iniciar Escaneo". • Verificar que: Se muestra el círculo de carga. Aparece el mensaje "Esperando los logs del escaneo". Los logs comienzan a generarse tras unos segundos.
Datos de Entrada	• Acción del usuario: clic en el botón "Iniciar Escaneo".
Resultados Esperados	• El escaneo comienza sin errores. • El círculo de carga y el mensaje aparecen correctamente. • Los logs se generan y actualizan dinámicamente en pantalla.
Criterios de Aceptación / Rechazo	**Aceptación:** • El escaneo comienza tras el clic. • El círculo de carga y los logs funcionan sin fallos. **Rechazo:** • El botón "Iniciar Escaneo" no responde. • No se muestra el círculo de carga o los logs. • El botón "Cerrar Puertos" está habilitado durante el escaneo.
Desarrollador Asignado	Fausto Navarrete

Tabla 7 Test Iniciar Escaneo

N°: 4 **Caso de prueba FrontEnd Nombre Caso de Prueba:** Test botón Cerrar Puertos

DESCRIPCIÓN DEL CASO DE PRUEBA Validar que al hacer clic en el botón "Cerrar Puertos", se generen las reglas de firewall necesarias y se actualice el log con la información correspondiente.	
Precondiciones	• El escaneo de puertos debe haberse completado. • El botón "Cerrar Puertos" debe estar habilitado.
Pasos para seguir	• Realizar un escaneo de puertos. • Verificar que el botón "Cerrar Puertos" se habilita al finalizar el escaneo. • Hacer clic en el botón "Cerrar Puertos". • Verificar que: Aparece un mensaje confirmando el cierre de puertos. Los logs se actualizan mostrando los puertos cerrados.
Datos de Entrada	• Acción del usuario: clic en el botón "Cerrar Puertos".
Resultados Esperados	• Los puertos detectados como abiertos son bloqueados correctamente. • Los logs muestran la confirmación de cierre de cada puerto. • No se generan errores visuales ni de funcionalidad.
Criterios de Aceptación / Rechazo	**Aceptación:** • Los puertos son bloqueados correctamente. • Los logs se actualizan con los detalles del cierre. **Rechazo:** • El botón no responde. • No se bloquean los puertos abiertos. • Los logs no se actualizan o muestran errores.
Desarrollador Asignado	Fausto Navarrete

Tabla 8 Test botón Cerrar Puertos

N°: 5 **Caso de prueba FrontEnd Nombre Caso de Prueba:** Carga y Actualizacion de Logs

DESCRIPCIÓN DEL CASO DE PRUEBA Comprobar que los logs del escaneo se generen y actualicen dinámicamente durante el proceso.	
Precondiciones	• El botón "Iniciar Escaneo" debe estar habilitado y funcional.
Pasos para seguir	• Iniciar un escaneo de puertos. • Verificar que los logs se actualizan en tiempo real mientras avanza el escaneo. • Confirmar que los logs muestran: Puertos escaneados. Estado (abierto/cerrado). Tiempo estimado de finalización del escaneo.
Datos de Entrada	• Acción del usuario: inicio del escaneo.
Resultados Esperados	• Los logs se generan en tiempo real. • Los mensajes de log son claros y sin errores de formato. • Al finalizar el escaneo, se muestra un resumen con los puertos abiertos.
Criterios de Aceptación / Rechazo	**Aceptación:** • **Los logs se actualizan dinámicamente sin interrupciones.** • **La información es clara y precisa.** **Rechazo:** • **Los logs no se generan o se muestran incompletos.** • **La información de los logs es incorrecta o confusa.**
Desarrollador Asignado	Fausto Navarrete

Tabla 9 Carga y Actualizacion de Logs

N°: 6 **Caso de prueba FrontEnd Nombre Caso de Prueba:** visualización de Disclaimer

DESCRIPCIÓN DEL CASO DE PRUEBA Comprobar que la información del panel izquierdo presente información relacionada con el uso de la aplicación y disclaimer	
Precondiciones	• Panel Izquierdo se Muestra Al Iniciar la aplicación claramente
Pasos para seguir	• Presionar en el panel izquierdo el Encabezado Información del Escáner de Puertos. • Proyección de la Información sobre disclaimer. • Confirmar que la información se presente sin errores visuales • Confirmamos que el botón nos siga a través de la imagen • Presionar este botón nos regresa al menú principal .
Datos de Entrada	• Acción del usuario: Presionar en el panel izquierdo el encabezado de Información del Escáner de puertos.
Resultados Esperados	• Información se muestra clara y precisa. • botón se presenta y sigue a través de la navegación • Al presionar el botón nos devuelve a la pantalla principal
Criterios de Aceptación / Rechazo	**Aceptación:** • Información se muestra correctamente. • Botón funciona y permite regresar a la pantalla principal **Rechazo:** • La información no aparece o tiene falla de interfaz • El botón no funciona
Desarrollador Asignado	Fausto Navarrete

Tabla 10 visualización de Disclaimer

3.2. Pruebas Backend

N°: 7 **Caso de prueba Backend Nombre Caso de Prueba:** Ejecución PortScanner

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar que el sistema cierre correctamente los puertos que no están en la lista de puertos necesarios.	
Precondiciones	• Sistema Windows con PowerShell habilitado • Usuario con permisos de administrador • Puertos abiertos en el sistema
Pasos para seguir	• Iniciar el aplicativo **(.exe)** • Navegar entre los módulos, al **primer módulo de análisis**. • Dar clic al botón de **Iniciar Escaneo**. • Observar la respuesta del programa. • Verificar los puertos abiertos. • Cerrar los puertos más vulnerables.
Datos de Entrada	• Objetivo: Localhost.
Resultados Esperados	• El script detectará correctamente los puertos abiertos. • El script cerrará correctamente los puertos que puedan ser cerrados. • Los puertos permanecerán en su estado original de no poder aplicar el script.
Criterios de Aceptación / Rechazo	**ACEPTACIÓN**: • No se produce **crash** del programa. • Se genera una **salida JSON** indicando que no se cerraron puertos. • El programa termina su ejecución de manera controlada. **RECHAZO:** • El script crashea sin manejar la excepción. • No se genera salida JSON. • El programa queda en un estado inconsistente.
Desarrollador Asignado	Juan Francisco Quishpe Piñaloza

Tabla 11 Ejecución PortScanner

N°: 8 **Caso de prueba Backend Nombre Caso de Prueba:** Escaneo de archivo de la API de Virus Total

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar que la API funcione correctamente al momento de realizar el escaneo de un archivo vulnerable	
Precondiciones	• Python 3.x instalado. • Módulo requests instalado. • API key de VirusTotal configurada. • Conexión a internet activa. • Archivo de prueba. • Se han realizado varias peticiones previas a la API
Pasos para seguir	• Preparar un archivo de prueba pequeño (< 32MB). • Utilizar la interfaz gráfica para cargar el archivo. • Presionamos el botón para ejecutar dentro del UI.
Datos de Entrada	• Archivo: **cualquiera tipo de archivo**. • Contenido: **cualquier tipo de archivo**. • Tamaño: < 32MB • API Key: La configurada en el script
Resultados Esperados	• El script intentará subir el archivo a VirusTotal • No se debe crashear el programa • Debe mantener la integridad del archivo original
Criterios de Aceptación / Rechazo	**ACEPTACIÓN:** • El script maneja el error de límite de API correctamente. • Se genera una salida JSON válida con el mensaje de error. • El archivo original permanece intacto. • El programa termina su ejecución de manera controlada. • Se muestra un mensaje de error claro sobre el límite de API. **RECHAZO:** • El script **crashea** sin manejar la excepción. • No se genera una salida JSON válida. • Se corrompe o modifica el archivo original.
Desarrollador Asignado	Juan Francisco Quishpe Piñaloza

Tabla 12 Escaneo de archivo de la API de Virus Total

N°: 9 **Caso de prueba Backend Nombre Caso de Prueba:** Ejecución de la API de OpenAI

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar que el sistema obtenga correctamente un consejo de seguridad desde la API de OpenAI	
Precondiciones	• Python 3.x instalado • **API key** de OpenAI configurada • Conexión a internet activa
Pasos para seguir	• Ejecutar el script Python • Verificar la respuesta del programa • Comprobar el formato **JSON** de la salida • Validar el contenido del consejo recibido
Datos de Entrada	• API Key: La configurada en el script • Modelo: gpt-3.5-turbo • Max **tokens**: 100
Resultados Esperados	• El **script** debe conectarse exitosamente a la **API.** • La respuesta debe estar en formato JSON. • Generar una respuesta para la auditoría.
Criterios de Aceptación / Rechazo	**ACEPTACIÓN**: • La conexión con la API es exitosa. • Se genera una salida JSON válida con el consejo. • El programa termina su ejecución de manera controlada. • El consejo recibido es coherente y útil. **RECHAZO:** • El script crashea sin manejar la excepción. • No se genera una salida JSON válida. • La API retorna un error de autenticación. • El consejo está vacío o es incoherente.
Desarrollador Asignado	Juan Francisco Quishpe Piñaloza

Tabla 13 Ejecución de la API de OpenAI

N°: 10 **Caso de prueba Backend Nombre Caso de Prueba:** Interrupción del escaneo de puertos

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar el comportamiento de la interfaz cuando el escaneo de puertos se interrumpe inesperadamente.	
Precondiciones	• Aplicación Electron iniciada • Interfaz cargada correctamente • Conexión a la red activa
Pasos para seguir	• Hacer clic en el botón 'startScanBtn' • Esperar que el progreso llegue al 50% • Simular una interrupción de la conexión • Observar el comportamiento de la interfaz • Verificar el estado de los elementos visuales
Datos de Entrada	• Click en botón de inicio • Evento de interrupción de red
Resultados Esperados	• El círculo de progreso debe detenerse • El texto de estado debe mostrar un mensaje de error • Los logs deben registrar la interrupción • El intervalo de progreso debe limpiarse • La interfaz debe quedar en un estado recuperable
Criterios de Aceptación / Rechazo	**ACEPTACIÓN**: • El intervalo se limpia correctamente • La interfaz muestra el error apropiadamente • Los elementos visuales reflejan el estado de error **RECHAZO:** • El progreso continúa después de la interrupción • La interfaz queda en estado inconsistente • No se muestra mensaje de error al usuario
Desarrollador Asignado	Juan Francisco Quishpe Piñaloza

Tabla 14 Interrupción del escaneo de puertos

N°: 11 **Caso de prueba Backend Nombre Caso de Prueba:** Manejo de elementos DOM inexistentes

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar el comportamiento del script cuando alguno de los elementos DOM referenciados no existe en la página.	
Precondiciones	• Página HTML cargada • Script JavaScript cargado • Consola del navegador abierta para monitoreo
Pasos para seguir	• Modificar el HTML eliminando el elemento 'logsLink' • Cargar la página • Verificar errores en consola • Intentar interactuar con otros elementos • Verificar si el resto de la funcionalidad sigue operativa
Datos de Entrada	• Interacciones con elementos existentes
Resultados Esperados	• No debe crashear la aplicación • Los event listeners deben funcionar • Debe registrar error en consola • Resto de funcionalidades intactas
Criterios de Aceptación / Rechazo	**ACEPTACIÓN**: • Error controlado en consola • Resto de funcionalidades operativas • No hay crash del script • Manejo de error apropiado **RECHAZO:** • Script crashea completamente • Errores no controlados • Event listeners no funcionan • Sin mensaje de error apropiado
Desarrollador Asignado	Juan Francisco Quishpe Piñaloza

Tabla 15 Manejo de elementos DOM inexistentes

N°: 12 **Caso de prueba Backend Nombre Caso de Prueba:** Selección múltiple de archivos API Virus total

DESCRIPCIÓN DEL CASO DE PRUEBA Verificar el comportamiento del fileInput cuando se intenta seleccionar múltiples archivos.	
Precondiciones	• Input de archivo presente en el DOM • Event listener de 'change' activo • Elemento fileName presente
Pasos para seguir	• Intentar seleccionar múltiples archivos • Verificar el comportamiento del event listener • Comprobar la actualización del texto • Verificar el manejo del array de files • Probar diferentes tipos de archivos
Datos de Entrada	• Múltiples archivos seleccionados • Diferentes tipos de archivos • Archivos con nombres especiales
Resultados Esperados	• Solo debe procesar el primer archivo • Se actualiza correctamente • No debe haber errores en consola • Debe manejar caracteres especiales • Debe mantener consistencia en el display
Criterios de Aceptación / Rechazo	**ACEPTACIÓN**: • Múltiples archivos procesados • Texto no actualizado • Errores en consola • Problemas con caracteres especiales • Display inconsistente **RECHAZO:** • Múltiples archivos no procesados • Texto no actualizado • Errores en consola • Problemas con caracteres especiales • Display inconsistente
Desarrollador Asignado	Juan Francisco Quishpe Piñaloza

Tabla 16 Selección múltiple de archivos API Virus total

3.3. Despliegue del Proyecto con Electron y Python

3.3.1. Empaquetamiento de la Aplicación

Para el empaquetamiento de la aplicación, utilicé **Electron**, ya que es un framework que permite la creación de aplicaciones de escritorio multiplataforma empaquetadas como ejecutables (.exe en Windows). Sin embargo, enfrentamos algunos problemas debido a que el lenguaje de desarrollo principal fue **Python**.

Uno de los principales inconvenientes fue que, al no estar Python instalado en todas las computadoras, el ejecutable de Electron no podía ejecutar directamente mis scripts **.py**. Para solucionar esto, convertí mis scripts de Python en ejecutables utilizando la librería **PyInstaller**.

Esto generó un archivo **.exe** que podía ejecutarse de manera independiente sin requerir una instalación previa de Python en el sistema del usuario.

3.3.2. Corrección de Problemas con Rutas Relativas y Absolutas

Otro desafío fue la gestión de las rutas relativas y absolutas en el módulo de **subida de archivos** para su análisis con la API de **VirusTotal**. Inicialmente, las rutas utilizadas en desarrollo no coincidían con las utilizadas tras la conversión a .exe, lo que provocó errores en la carga de archivos.

Pero se logró que el programa manejara correctamente los archivos subidos para su análisis con la API de **VirusTotal**, asegurando su funcionalidad después de la conversión a **.exe**.

3.3.3. Despliegue Final

Formato de Distribución

El despliegue de la aplicación se realizará como un **archivo .exe portable**, evitando la necesidad de un instalador para mayor facilidad de distribución y uso. Esto permite que los usuarios simplemente descarguen el ejecutable y lo ejecuten sin configuraciones adicionales.

3.3.4. Método de Distribución

Para facilitar el acceso y la descarga de la aplicación, se ha decidido subir el ejecutable a una **página web creada en Wix**. Wix fue elegido por su facilidad de uso y rapidez en el despliegue, permitiendo que en poco tiempo se pueda tener una página funcional con una interfaz sencilla pero efectiva.

La página incluirá:

- ✓ Capturas de la aplicación, detallando sus funcionalidades.
- ✓ Un botón de descarga directa del ejecutable.
- ✓ Información de acerca de nosotros los programadores.

Este método de distribución garantiza que los usuarios puedan acceder de manera rápida y sencilla a la aplicación sin complicaciones técnicas.

Conclusiones

- El desarrollo e implementación del sistema de automatización para auditoría de redes ha demostrado ser una solución efectiva para democratizar el acceso a herramientas de ciberseguridad, creando un puente entre la complejidad técnica y usuarios sin conocimientos especializados a través de una interfaz intuitiva y procesos automatizados.

- La arquitectura modular implementada, que combina un frontend en Electron con un backend en Python y JavaScript, ha probado ser una decisión técnica acertada, facilitando el desarrollo, las pruebas y estableciendo una base sólida para futuras expansiones. La integración de servicios externos como la API de VirusTotal y la API de OpenAI ha enriquecido significativamente las capacidades del sistema, permitiendo un análisis más completo que va más allá del simple escaneo de puertos, proporcionando información detallada sobre vulnerabilidades potenciales y recomendaciones de seguridad personalizadas.

- Las pruebas exhaustivas realizadas han validado la robustez del sistema en diversos escenarios de uso, demostrando su capacidad para manejar interrupciones y mantener la estabilidad operativa en condiciones variadas de carga y uso, incluyendo pruebas de estrés y simulaciones de diferentes tipos de amenazas comunes en entornos de red.

- La atención dedicada a la experiencia de usuario ha resultado en una interfaz que hace accesibles conceptos complejos de seguridad informática, permitiendo que usuarios sin experiencia técnica puedan realizar auditorías de seguridad efectivas, contribuyendo así a la promoción de mejores prácticas de ciberseguridad en un público más amplio.

- El sistema no solo cumple con su objetivo principal de proporcionar una herramienta accesible para la detección de vulnerabilidades, sino que también establece un nuevo estándar en la democratización de herramientas de seguridad informática, demostrando que es posible combinar funcionalidades avanzadas con una experiencia de usuario intuitiva y accesible.

Recomendaciones

- Para mantener la efectividad y relevancia del sistema a largo plazo, es fundamental establecer un proceso continuo de actualización de las firmas de seguridad y bases de datos de vulnerabilidades. Se recomienda implementar un sistema automatizado de actualizaciones que mantenga al día tanto las definiciones de seguridad como las mejoras funcionales del software, garantizando así su capacidad para detectar y responder a nuevas amenazas.
- La expansión de la compatibilidad del sistema a otras plataformas además de Windows representaría un paso significativo en su evolución. El desarrollo de versiones para sistemas operativos Linux y macOS permitiría alcanzar un público más amplio y diverso, maximizando el impacto positivo de la herramienta en la comunidad de usuarios.
- Se recomienda fortalecer los mecanismos de seguridad del sistema mediante la implementación de un sistema de autenticación más robusto que incluya autenticación de doble factor y gestión avanzada de permisos. Estas mejoras son especialmente relevantes considerando el potencial uso del sistema en entornos empresariales donde la seguridad y el control de acceso son críticos.
- Es aconsejable desarrollar un sistema más completo de generación de informes y seguimiento histórico. La capacidad de mantener registros detallados de las vulnerabilidades detectadas y las acciones correctivas tomadas no solo mejoraría la utilidad de la herramienta para fines de auditoría, sino que también facilitaría la identificación de patrones y tendencias en la seguridad del sistema.

- Para facilitar la adopción y uso correcto de la herramienta, se recomienda mantener una documentación exhaustiva y actualizada. Esto incluye la creación de guías de usuario detalladas, tutoriales en video y ejemplos de casos de uso que ayuden a los usuarios a aprovechar al máximo las capacidades del sistema.

Referencias bibliográficas

- Agrawal, S., & Agrawal, J. (2021). *Network Security and Cryptography*. Chapman and Hall/CRC.
- Bissell, K., LaSalle, R., & Dal Cin, P. (2023). *State of Cybersecurity Resilience 2023*. Accenture Security.
- Diogenes, Y., & Ozkaya, E. (2023). *Cybersecurity – Attack and Defense Strategies: Infrastructure security with Red Team and Blue Team tactics* (3rd ed.). Packt Publishing.
- Electron. (2024). *Build cross-platform desktop apps with JavaScript, HTML, and CSS*. https://www.electronjs.org/
- Gibson, D. (2023). *CompTIA Security+ Get Certified Get Ahead: SY0-601 Study Guide*. YCDA, LLC.
- Harris, S., & Maymi, F. (2023). *CISSP All-in-One Exam Guide* (9th ed.). McGraw-Hill Education.
- Kim, P. (2023). *The Hacker Playbook 3: Practical Guide To Penetration Testing*. Secure Planet LLC.
- Martin, R. C. (2019). *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Pearson Education.
- McClure, S., Scambray, J., & Kurtz, G. (2022). *Hacking Exposed 8: Network Security Secrets & Solutions* (8th ed.). McGraw-Hill Education.
- NIST. (2023). *Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1*. National Institute of Standards and Technology.
- OpenAI. (2024). *GPT-3.5 API Documentation*. https://platform.openai.com/docs/

- Sommerville, I., & Sawyer, P. (2021). *Software Engineering* (11th ed.). Pearson Education Limited.
- Stallings, W. (2023). *Network Security Essentials: Applications and Standards* (7th ed.). Pearson.
- VirusTotal. (2024). *VirusTotal API Documentation v3*. https://developers.virustotal.com/reference
- White, G., & Hewitt, B. (2023). *CompTIA Security+ Guide to Network Security Fundamentals* (7th ed.). Cengage Learning.
- Wix. (2024). *Wix Developer Documentation*. https://dev.wix.com/docs