

FASE EJECUCION

BASE DOCUMENTAL

GA8-220501096-AA1-EV01

PRESENTADO POR:

JHON ALEXANDER GONZALEZ GALLEGO

CC: 1014309798

PRESENTADO A:

NELLY ISABELL RODRIGUEZ

FICHA: 2977390

SENA

TECNOLOGIA ANALISIS Y DESARROLLO DE SOFTWARE

CAICEDONIA VALLE

2025

CONTEXUALIZACION DEL PROYECTO

La creciente amenaza de ataques cibernéticos resalta la necesidad de implementar soluciones de seguridad eficientes. El desarrollo de un software completo y actualizado permite garantizar la integridad, confidencialidad y disponibilidad de la información de los usuarios, minimizando así riesgos asociados a malware, ransomware y otras amenazas digitales.

Por eso nace la creación del proyecto del software de seguridad con el fin de ser un software robusto y eficiente, garantizando la seguridad, integridad, confidencialidad de los datos de los usuarios

METODOLOGIA

Elegir la metodología juega un papel muy importante ya que esta nos dará todo el protocolo que debemos llevar a cabo para culminar exitosamente el proyecto propuesto, no es elegir por elegir hay que buscar la que mejor se adapte a la finalidad del proyecto.

La metodología seleccionada, la cual hace parte de las metodologías ágiles, fue la metodología Scrum. La metodología Scrum se centra en el aprendizaje continuo y en la adaptación dependiendo de los diferentes factores que afecten el proyecto y a los integrantes del equipo. Scrum se encuentra estructurado de tal forma, que logra ayudar a los equipos del proyecto a adaptarse de manera eficaz a los diferentes cambios que vayan saliendo en la creación del proyecto, haciendo claro que el equipo no lo sabe todo desde un principio y permite recibir sugerencias frente a los requisitos solicitados por cliente.

ARQUITECTURA

La arquitectura de software es fundamental en el desarrollo de sistemas, ya que define la estructura y organización de los componentes que lo conforman. Una buena arquitectura no solo nos facilita la escalabilidad y el mantenimiento del software, sino que también permite a los equipos de desarrollo trabajar de manera más eficiente.

Una buena arquitectura se logra mediante la comprensión profunda de los requisitos del usuario y los objetivos del negocio. Es esencial involucrar a todas las partes interesadas relevantes, incluyendo a los desarrolladores, los gerentes de proyecto y los usuarios finales. La modularidad, la flexibilidad y la capacidad para adaptarse a cambios son pilares de una arquitectura robusta.

Entender los Requisitos: Comprende completamente los requisitos del usuario y del negocio. Esto implica una comunicación clara y continua con los interesados para asegurarte de que todos los aspectos clave estén cubiertos.

Descomposición Modular: Divide el sistema en módulos o componentes lógicos. Cada módulo debe tener una responsabilidad específica y estar diseñado para ser independiente del resto del sistema. Esto favorece la modularidad y la reutilización del código.

Escalabilidad y Rendimiento: Diseña la arquitectura para ser escalable. Considera cómo el sistema manejará un aumento en la carga y asegúrate de que sea capaz de escalar horizontal o verticalmente según sea necesario.

Flexibilidad y Adaptabilidad: La arquitectura debe ser lo suficientemente flexible como para permitir futuras expansiones y modificaciones sin requerir una revisión completa del sistema. Adopta prácticas como la inyección de dependencias y el diseño basado en interfaces para facilitar los cambios.

Mantenibilidad: La arquitectura debe ser fácil de entender y mantener. Utiliza prácticas de codificación limpias y sigue principios de diseño sólidos. La documentación clara y el código bien comentado son fundamentales.

Seguridad: Considera las medidas de seguridad desde el principio. Identifica las vulnerabilidades potenciales y diseña capas de seguridad en el sistema para proteger los datos y las operaciones críticas.

Pruebas y Validación: Implementa pruebas unitarias, de integración y de sistema para validar la arquitectura. Las pruebas ayudan a identificar posibles problemas antes de que se conviertan en problemas mayores.

En resumen, una arquitectura de software bien diseñada proporciona una base sólida para el desarrollo, asegurando la calidad, eficiencia y capacidad de adaptación del sistema a lo largo de su ciclo de vida.

REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES

Es importante reconocer y solucionar las necesidades que requiere el software. Por esta razón se es necesario la verificación de los requerimientos funcionales y no funcionales del software de seguridad, poco a poco el software mostrara sus necesidades de acuerdo a la problemática que se vayan presentando, las aplicaciones siempre van a requerir actualizaciones y poder tener un buen funcionamiento.

REQUISITOS FUNCIONALES

- Registro de usuario
- Ingreso sesión con usuario
- Base de datos de productos
- Actualizaciones obligatorias periódicas y frecuentes
- Sugerencias para optimizar

REQUISITOS NO FUNCIONALES

- La app deberá mantener actualizada la información en todo momento.
- Se dará mantenimiento a la app lo más seguido posible.
- El tiempo de respuesta de la app no debe tardar mucho, seria inmediata.
- La aplicación funcionaria las 24 horas.
- Los protocolos de registro darán confiabilidad a la hora de ingresar a la aplicación.
- Un dispositivo solo se permite el ingreso de un usuario en tiempo real.

HISTORIAS DE USUARIO

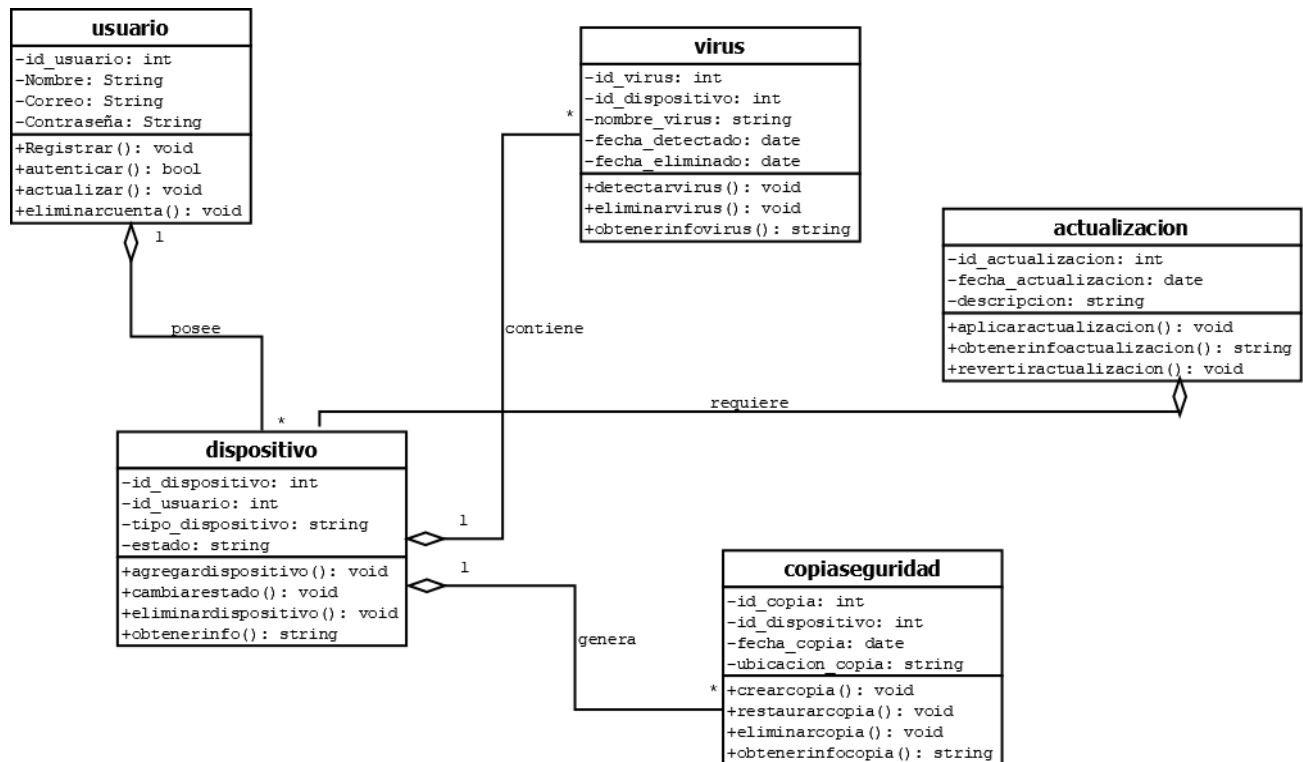
Historia de usuario	
Numero: # 0001	Nombre de la historia de usuario: Solicito restauración de contraseña
Usuario: Claro	
Prioridad: Media	Puntos estimados:1
Descripción: Restaurar contraseña	
Observaciones: El cliente solicita restaurar su contraseña, ya que se le olvido	
Criterios de aceptación: Se verifica correo del usuario, dándole validación para que pueda restaurar la contraseña	

Historia de usuario	
Numero: #0002	Nombre de la historia de usuario: No puedo registrarme
Usuario: Homecenter	
Prioridad: Alta	Puntos estimados:1
Descripción: El usuario manifiesta no poderse registrar	
Observaciones: Se validan datos ingresados por el cliente	
Criterios de aceptación: Se evidencia un error por parte de la página web	

Historia de usuario	
Numero: #0003	Nombre de la historia de usuario: No me deja ingresar
Usuario: Colpensiones	
Prioridad: Media	Puntos estimados:1
Descripción: El usuario manifiesta no poder ingresar correctamente	
Observaciones: Se evidencia que el usuario y la contraseña no coinciden	
Criterios de aceptación: Se le informa al usuario poner correctamente la clave, o en su defecto reestablecer contraseña	

Estas historias permitirán a los equipos de desarrollo y diseño comprender las necesidades y expectativas de los usuarios finales, asegurando que las funcionalidades implementadas sean coherentes con los requisitos del negocio.

DIAGRAMA DE CLASES

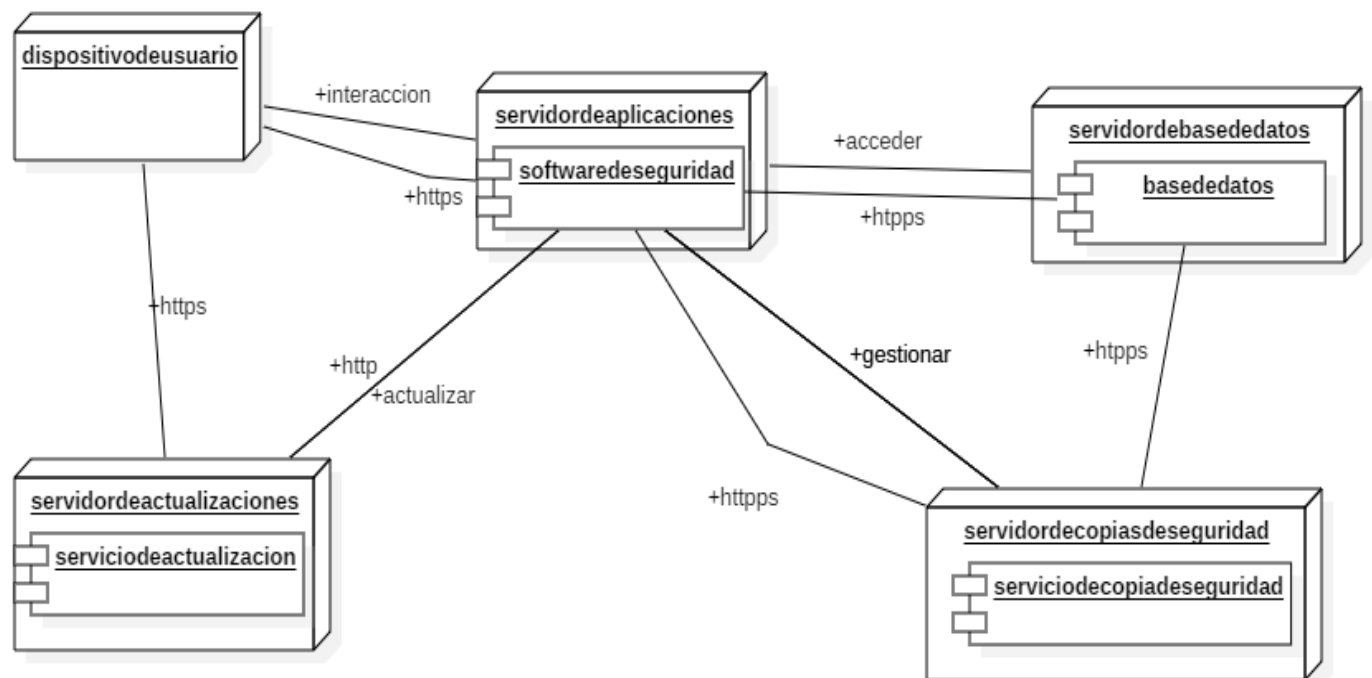


Este diagrama de clases representa cómo funcionaría un sistema de antivirus desde el punto de vista de su diseño.

En él, los usuarios pueden registrarse, iniciar sesión y administrar sus cuentas. Cada usuario posee uno o varios dispositivos, los cuales pueden estar expuestos a virus. El sistema tiene la capacidad de detectar y eliminar esas amenazas.

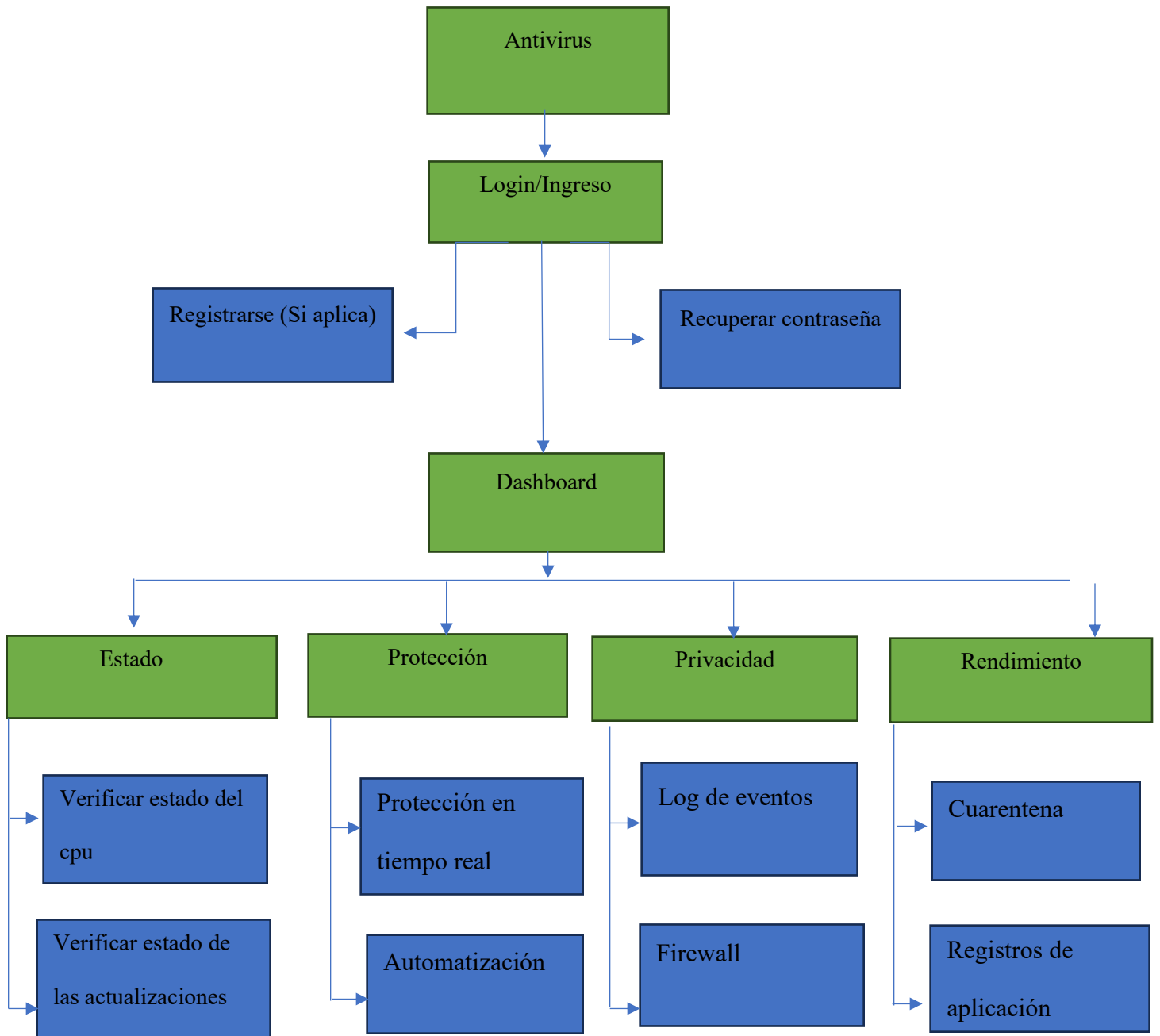
Además, los dispositivos requieren actualizaciones para mantenerse seguros, y pueden generar copias de seguridad para proteger la información en caso de fallos.

DIAGRAMA DE PAQUETES



En conclusión, este diagrama refleja un diseño descentralizado y escalable, donde cada componente cumple una función específica: proteger, actualizar, almacenar y respaldar. Esto asegura un sistema de antivirus robusto, seguro y confiable, en el que las responsabilidades están claramente separadas y bien organizadas.

MAPA DE NAVEGACION



Árbol jerárquico

└─ ****Login/Ingreso****

| └─ Registrarse (Si aplica)

| └─ Recuperar contraseña

|

└─ ****Dashboard (Panel Principal) ****

| └─ ****Estado****

| | └─ Verificar estado del CPU

| | └─ Verificar estado de las actualizaciones

| |

| └─ ****Protección****

| | └─ Protección en tiempo real

| | └─ Automatización (ejm: escaneos programados)

| |

| └─ ****Privacidad****

| | └─ Log de eventos (historial de acciones)

| | └─ Firewall (gestión de conexiones)

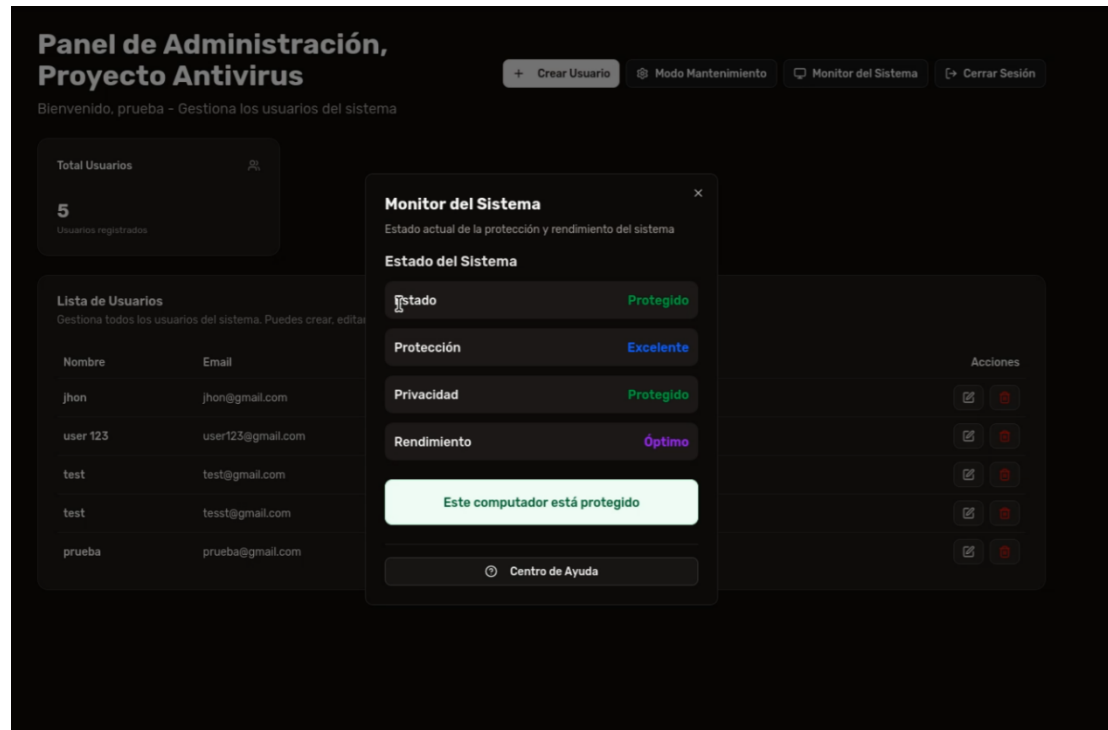
| |

| └─ ****Rendimiento****


| └─ Cuarentena (archivos aislados)

| └─ Registros de aplicación (logs técnicos)

PRUEBAS UNITARIAS




Lista de Usuarios			
Gestiona todos los usuarios del sistema. Puedes crear, editar y eliminar usuarios.			
Nombre	Email	ID	Acciones
jhon	jhon@gmail.com	68901edf2def3909d4359cbb	🔗 🔗
user 123	user123@gmail.com	68901ee72def3909d4359cbc	🔗 🔗
test	test@gmail.com	689d33ac36446f593a580b41	🔗 🔗
test	tesst@gmail.com	689d37ae036446f593a580b42	🔗 🔗
prueba	prueba@gmail.com	68a0964fa3bc3a136ecd0154	🔗 🔗




Sitio en Mantenimiento


Estamos realizando mejoras en nuestro sistema para brindarte una mejor experiencia.

**¿Qué está pasando?**


Nuestro equipo técnico está trabajando para resolver problemas de rendimiento y agregar nuevas funcionalidades.


**¿Qué puedes hacer?**

Te recomendamos intentar acceder nuevamente en unos minutos. El mantenimiento suele completarse rápidamente.


**Tiempo estimado**

Esperamos completar el mantenimiento en aproximadamente 15-30 minutos.

Intentar de nuevo


Ir a página principal

Si el problema persiste, contacta a nuestro equipo de soporte
Gracias por tu paciencia



Recuperar Contraseña


Ingresa tu email para restablecer tu contraseña




¡Contraseña Restablecida!

Tu nueva contraseña es: **password123**


Te recomendamos cambiar esta contraseña después de iniciar sesión.

Ir al Login

Restablecer otra contraseña

¿Recordaste tu contraseña? [Inicia sesión aquí](#)

¿No tienes una cuenta? [Regístrate aquí](#)



Crear Cuenta

Completa el formulario para crear tu cuenta

Nombre completo

Email

Contraseña

☐ Use a Securely Generated Password
 Zen will save this password for this website.

[Manage Passwords](#)

Crear Cuenta

[¿Olvidaste tu contraseña? Recupérala aquí](#)

LISTA DE CHEQUEO

Ítem	Cumple	No Cumple	Observaciones
1. La documentación incluye una descripción clara del objetivo del proyecto (desarrollo de un antivirus moderno).	x		El objetivo está bien definido en la introducción.
2. Se especifica la arquitectura de microservicios y cómo se integra en el proyecto.	x		La arquitectura se describe con diagramas claros.

3. La documentación detalla los requisitos funcionales y no funcionales del antivirus.		x	Faltan algunos requisitos no funcionales, como el rendimiento esperado.
4. Se incluyen diagramas que explican el funcionamiento del sistema.	x		Los diagramas son claros y están bien detallados.
5. La documentación describe las tecnologías y herramientas que se utilizarán.	x		Las tecnologías y herramientas están bien especificadas.
6. Se especifican los protocolos de seguridad que se implementarán.	x		Se detallan los protocolos de seguridad de manera suficiente.
7. La documentación incluye un plan de pruebas para validar el funcionamiento del antivirus.		x	El plan de pruebas falta estructurarlo de manera correcta.
8. La documentación está completa, organizada y es fácil de entender.	x		La documentación es clara y está bien organizada.

ENLACE REPOSITORIO

<https://github.com/JhonGonzalez99/BaseDocumental>