Innovando la Seguridad de Contraseñas en la Era Digital

Descripción General

El proyecto consiste en desarrollar un gestor de contraseñas que permita a los usuarios crear, almacenar y acceder de manera segura a sus contraseñas para diferentes servicios en línea. El objetivo principal es brindar una solución que proteja la información sensible de los usuarios, al tiempo que les facilita el manejo de sus credenciales.

Objetivos del Proyecto

- Programar una aplicación segura y eficiente que permita codificar contraseñas, para mejorar la seguridad de la información personal a partir del lenguaje de programación Python.

- Establecer una aplicación de fácil acceso para el usuario, donde se implemente un algoritmo de cifrado con el lenguaje de Python, en el que cada usuario tenga una excelente interacción con sus contraseñas de manera práctica y precisa.

- Implementar un diseño gráfico para la aplicación que sea atractivo para el usuario, eficiente, práctico y preciso que genere un entorno confiable y un ecosistema de seguridad diferente al modelo estándar que se maneja en la actualidad.

Finalidad o Impacto Esperado

- Proteger la información personal y las credenciales de los usuarios, reduciendo el riesgo de accesos no autorizados y robos de identidad.

- Facilitar el manejo de contraseñas, permitiendo a los usuarios crear y recordar contraseñas únicas y seguras para cada servicio en línea.

- Contribuir a la seguridad digital de los usuarios, fomentando buenas prácticas en la gestión de contraseñas.

Análisis de Seguridad en la Aplicación

Identificación de Componentes Críticos

- Módulo de cifrado y almacenamiento de contraseñas.
- Generador de contraseñas aleatorias.
- Interfaz de usuario para la gestión de la gestión de contraseñas.
- Proceso de "salting" para el cifrado de contraseñas.

Posibles Amenazas y Vulnerabilidades

1. Ataques de fuerza bruta o diccionario para descifrar las contraseñas almacenadas.

2. Vulnerabilidades en el proceso de cifrado que permitan la exposición de las contraseñas.

3. Falta de validación de entrada en la interfaz de usuario, lo que podría llevar a ataques de inyección.

Medidas de Seguridad a Implementar

Para garantizar la seguridad de Alphaweb, es fundamental implementar herramientas de análisis estático y pruebas automatizadas. Las herramientas mencionadas (Bandit, Ruff y Pytest) proporcionan diferentes capas de verificación que, al combinarse, ofrecen una evaluación robusta de la seguridad.

Herramientas y su Aplicación

1. Bandit (Análisis de Vulnerabilidades)

Propósito:

- Detectar patrones de código inseguros comunes en Python.

- Identificar vulnerabilidades de seguridad específicas en el manejo de contraseñas.

- Aplicación en un codificador de contraseñas:

- Verificar que no se utilicen algoritmos de hash débiles.

- Detectar credenciales codificadas directamente en el código.

- Identificar posibles problemas en la generación de valores aleatorios para salts.

2. Ruff (Linter de Calidad de Código)

Propósito:

1. Mantener altos estándares de calidad en el código.

2. Detectar malas prácticas que podrían afectar indirectamente la seguridad.

3. Pytest (Pruebas Automatizadas)

Propósito:

- ➔ Verificar el comportamiento correcto del codificador.
- ➔ Asegurar que las implementaciones criptográficas funcionan como se espera.
- ➔ Realizar pruebas unitarias de funciones de hash.
- ➔ Verificar que contraseñas idénticas produzcan hashes diferentes al usar distintos salts.
- ➔ Ejecutar tests de rendimiento para prevenir ataques por fuerza bruta.

Flujo de Implementación Recomendado

Integración Continua:

Configurar un modulo para pruebas donde implemente Ruff para análisis de código en cada commit, Bandit en cada pull request, Pytest con una cobertura mínima del 90% para el código crítico.

Análisis de Resultados:

- Priorizar la corrección de problemas reportados por Bandit.
- Mantener el código limpio con Ruff.
- Asegurar una alta cobertura de pruebas en componentes críticos de seguridad.

Beneficios para el Proyecto

- ✔ Reducción de vulnerabilidades en la implementación criptográfica.
- ✔ Detección temprana de malas prácticas de seguridad.
- ✔ Código más mantenible y menos propenso a errores.
- ✔ Documentación implícita mediante pruebas automatizadas.
- ✔ Mayor confianza en la robustez del codificador de contraseñas.