FASE 3: DISEÑO DEL PROYECTO "ENCRIPTACIÓN DE CONTRASEÑA SEGURA"

INTRODUCCIÓN

La fase de diseño es esencial para establecer cómo funcionará la herramienta a nivel lógico y visual, asegurando que cumpla con los requisitos definidos en la fase de análisis. En esta etapa, se crean modelos, diagramas y estructuras que servirán como guía para los desarrolladores, enfocándose tanto en la funcionalidad como en la experiencia del usuario (UX). Además, se establecen los mecanismos de seguridad que protegerán la información ingresada por los usuarios.

Objetivo del diseño

El objetivo principal es definir una arquitectura lógica y técnica para la herramienta, asegurando que sea segura, eficiente y fácil de usar. Esto incluye:

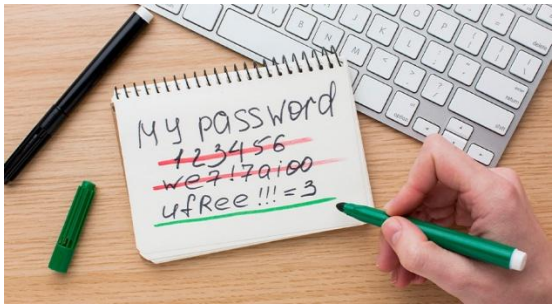
1. Diseñar la interfaz de usuario (UI) para garantizar una experiencia intuitiva.
2. Definir los componentes internos del sistema, como algoritmos y estructuras de datos.
3. Plantear medidas de seguridad para proteger la información sensible.
4. Establecer cómo interactuarán los módulos del sistema entre sí.

Componentes del diseño

1. Arquitectura del sistema

Se utilizará una arquitectura modular para facilitar el desarrollo, mantenimiento y escalabilidad del sistema. Los módulos principales son:

- **Módulo de evaluación:** Encargado de analizar la seguridad de la contraseña ingresada.
- **Módulo de generación:** Genera contraseñas seguras basadas en parámetros definidos.
- **Módulo de verificación:** Consulta bases de datos de contraseñas comprometidas.
- **Interfaz de usuario (UI):** Permite la interacción entre el usuario y la herramienta.



2. Diseño lógico

- **Flujo del sistema:**
  - El usuario ingresa una contraseña en la interfaz.
  - La contraseña es evaluada por el **módulo de evaluación** en tiempo real.
  - Si es débil, se genera una alternativa en el **módulo de generación**.
  - Finalmente, se informa al usuario del resultado con recomendaciones para mejorar su seguridad.

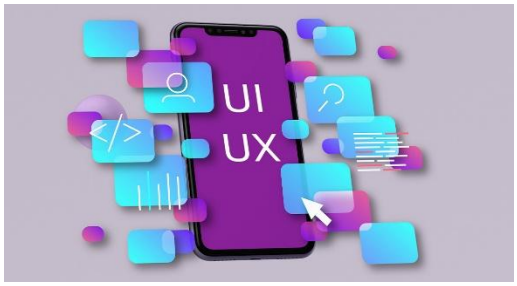
- **Diagrama de flujo:**
  El diseño debe incluir un diagrama de flujo que represente este proceso, destacando los puntos clave de decisión (por ejemplo, si la contraseña es segura o no).

3. Diseño físico

- **Frontend:** Utilizar tecnologías como HTML, CSS y JavaScript para crear una interfaz interactiva.
- **Backend:** Usar Python, Node.js u otro lenguaje para implementar la lógica de negocio.
- **Base de datos:** Si se almacenan listas de contraseñas comunes, usar una base de datos ligera como SQLite o integrarse con APIs externas como Have I Been Pwned.

4. Diseño de la interfaz de usuario (UI/UX)

- **Aspecto visual:**
  - Una interfaz limpia y minimalista que guíe al usuario paso a paso.
  - Colores y elementos visuales que destaquen la evaluación de contraseñas (por ejemplo, un semáforo de colores: verde para segura, amarillo para moderada, rojo para débil).
- **Interacción:**
  - Botones claros para ingresar contraseñas y generar alternativas.
  - Mensajes educativos que expliquen por qué una contraseña es débil o segura.



5. Algoritmos y lógica de negocio

- **Algoritmo de evaluación:**
  - Verificar longitud mínima (12 caracteres).
  - Verificar la presencia de mayúsculas, minúsculas, números y caracteres especiales.

    - Consultar si la contraseña aparece en listas de contraseñas comprometidas.
- **Algoritmo de generación:**
    - Generar cadenas aleatorias de alta entropía.
    - Incluir reglas personalizables (por ejemplo, longitud mínima, caracteres específicos).

6. Seguridad en el diseño

- **Encriptación:** Todas las contraseñas ingresadas deben procesarse localmente (en el cliente o backend) y nunca almacenarse.
- **Comunicación segura:** Usar HTTPS para proteger la transmisión de datos entre cliente y servidor.
- **Control de errores:** Implementar validaciones robustas para prevenir inyecciones u otros ataques.

Amenazas en la fase de diseño

1. **Errores en la arquitectura:** Una arquitectura mal planteada puede dificultar el mantenimiento y aumentar la posibilidad de vulnerabilidades.
2. **Diseño inseguro:** Si no se consideran las mejores prácticas de seguridad, la herramienta puede ser explotada por atacantes.
3. **Experiencia de usuario deficiente:** Una interfaz confusa podría desmotivar a los usuarios a utilizar la herramienta correctamente.

Soluciones a las amenazas

1. **Errores en la arquitectura:**
    - Crear diagramas UML (diagrama de clases, diagramas de casos de uso, etc.) para validar el diseño lógico.
    - Revisar el diseño con expertos en desarrollo y seguridad antes de avanzar al desarrollo.

2. **Diseño inseguro:**
    - Aplicar principios de **Secure by Design**, asegurando que la seguridad esté integrada desde el inicio.
    - Utilizar estándares de la industria para algoritmos de hash y generación de contraseñas.
3. **Experiencia de usuario deficiente:**
    - Realizar prototipos de la interfaz (usando herramientas como Figma o Adobe XD) y probarlos con usuarios reales.
    - Incluir mensajes claros y educativos para guiar al usuario durante todo el proceso.

Entregables del diseño

1. **Diagrama de flujo del sistema.**
2. **Diseño de interfaz (wireframes o prototipos).**
3. **Especificaciones de los algoritmos de evaluación y generación de contraseñas.**
4. **Plan de seguridad que detalle las medidas implementadas.**

Resultados esperados del diseño

- Tener una guía clara y detallada que permita al equipo de desarrollo implementar el sistema sin ambigüedades.
- Asegurar que la herramienta sea segura, funcional y fácil de usar.
- Reducir el riesgo de errores o vulnerabilidades durante la fase de desarrollo.