Paquete de Seguridad de DoltSharp

1. Autenticación y Autorización

* Meta: Asegurar que solo los usuarios autenticados puedan acceder a las funcionalidades críticas del sistema.
* Implementación:
  + Usar hashing de contraseñas con algoritmos como bcrypt o PBKDF2 en lugar de almacenar las contraseñas en texto plano.
  + Las credenciales de los usuarios (nombre de usuario y contraseña) se almacenarán de forma segura en un archivo cifrado en JSON o XML.
  + Implementar autorización por roles si es necesario, permitiendo a ciertos usuarios realizar acciones específicas (como editar o eliminar eventos).

2. Protección de Datos Personales

* Meta: Proteger los datos sensibles de los usuarios (nombre, apellido, email, contraseñas, etc.) contra accesos no autorizados.
* Implementación:
  + Usar cifrado AES para los archivos que contienen información sensible, asegurando que solo el sistema pueda descifrar y leer estos datos.
  + Implementar mecanismos para validar la entrada del usuario (input validation), asegurando que los datos introducidos no contengan scripts maliciosos o inyecciones de código.
  + Asegurar que los archivos de configuración y datos no sean accesibles desde ubicaciones no autorizadas. Usar permisos del sistema de archivos para restringir el acceso.

3. Recuperación de Contraseña

* Meta: Implementar un proceso seguro para la recuperación de contraseñas.
* Implementación:
  + El usuario deberá ingresar su email registrado para solicitar un código de recuperación.
  + Generar un código temporal de recuperación, que será enviado al email del usuario. El código debe ser válido solo por un corto periodo (por ejemplo, 10-15 minutos).
  + El archivo de usuarios debe almacenar el código de recuperación cifrado junto con un timestamp para su expiración.

4. Protección contra Ataques Comunes

* Meta: Proteger la aplicación de ataques como fuerza bruta, inyección de código, y accesos no autorizados.
* Implementación:
  + Limitar el número de intentos fallidos de inicio de sesión. Tras varios intentos fallidos (por ejemplo, 5), bloquear el acceso temporalmente o requerir medidas adicionales como un código CAPTCHA.
  + Usar validación de datos de entrada para protegerse contra inyecciones de código y asegurar que la información ingresada por el usuario no cause fallos en el sistema.
  + Implementar protección contra Cross-Site Scripting (XSS) y inyección de comandos aunque la aplicación sea de escritorio, ya que podría existir algún nivel de comunicación con servicios externos en el futuro.

5. Seguridad en el Manejo de Archivos

* Meta: Asegurar que los archivos que almacenan los datos del sistema y los usuarios estén protegidos contra modificaciones no autorizadas y accesos indebidos.
* Implementación:
  + Establecer permisos de acceso a los archivos para que solo el sistema tenga permiso de lectura y escritura.
  + Usar bloqueo de archivos (file locking) para evitar accesos simultáneos que puedan corromper los archivos. Asegurarse de que los archivos no se queden bloqueados permanentemente en caso de fallo en el sistema.
  + Implementar copias de seguridad automáticas de los archivos de datos, de forma que si se corrompe un archivo, se pueda restaurar una versión anterior.

6. Cifrado de la Información Sensible

* Meta: Asegurar que toda la información sensible esté cifrada en el sistema.
* Implementación:
  + Usar cifrado AES para los datos almacenados en archivos, como contraseñas, emails, y otros datos personales. Las claves de cifrado deben estar bien protegidas y no embebidas directamente en el código.
  + Para datos temporales o no sensibles, como el estado de las tareas, se puede utilizar hashing simple o no cifrarlos si no es necesario.

7. Auditoría y Registro de Actividades

* Meta: Registrar las actividades clave del sistema para fines de auditoría y detección de comportamientos anómalos.
* Implementación:
  + Registrar los intentos de inicio de sesión, tanto exitosos como fallidos, en un archivo de registro.
  + Crear un sistema de logs para registrar actividades importantes como creación, modificación o eliminación de tareas y eventos. Esto también debe incluir el monitoreo de accesos a los archivos.
  + Utilizar herramientas de logging sencillas como Serilog para guardar estos registros en archivos de texto separados.

8. Manejo de Sesiones

* Meta: Mantener seguras las sesiones de los usuarios.
* Implementación:
  + Generar tokens de sesión únicos para cada usuario al iniciar sesión, y almacenarlos de forma segura. Los tokens deben ser validados cada vez que se realice una acción sensible.
  + Configurar una expiración de sesión automática después de un periodo de inactividad (por ejemplo, 30 minutos), forzando al usuario a iniciar sesión nuevamente si la sesión expira.

9. Manejo Seguro de Actualizaciones

* Meta: Proveer un proceso seguro para futuras actualizaciones del sistema.
* Implementación:
  + En caso de futuras versiones de la aplicación, asegurarse de que los archivos de actualización estén firmados digitalmente para garantizar que no han sido alterados.
  + Implementar un sistema que verifique la integridad de los archivos críticos de la aplicación antes de realizar cualquier actualización.