

Universidad Tecnológica de Aguascalientes

*Ingeniería en Desarrollo y Gestión de Software*

**Desarrollo móvil integral**

9:10 practica práctica 3

**Desarrollo móvil integral**

**Juan Paulo Amezcua Mejía**

**Documento elaborado por:**

Ulises Martínez Olivares | 190624

**Grupo:** IDGS 10-A

Lugar: Aguascalientes, Ags. Fecha: 19 de octubre de 2023.

**Protección de datos personales**

**Cifrado de Datos en Reposo y en Tránsito**

Todos los datos almacenados en la base de datos SQL Server deben estar cifrados para protegerlos mientras están en reposo. Además, la comunicación entre la aplicación móvil y el servidor (.Net APIs) debe estar protegida mediante protocolos de seguridad como HTTPS para cifrar los datos en tránsito.

**Autenticación y Autorización Seguras**

Implementar un sistema sólido de autenticación y autorización para garantizar que solo los usuarios autorizados tengan acceso a los datos personales. Esto incluye medidas como contraseñas seguras (almacenadas de forma segura utilizando técnicas de hashing y salting) y tokens de acceso seguros.

**Protección contra Inyección de SQL**

Utilizar consultas parametrizadas o procedimientos almacenados para interactuar con la base de datos SQL Server. Para este proyecto se implementa un ORM (Entity Framework). Esto ayuda a prevenir ataques de inyección de SQL.

**Verificación y Validación de Datos de Usuario**

Validar y verificar cuidadosamente los datos de usuario antes de almacenarlos en la base de datos o mostrarlos en la aplicación. Esto ayuda a prevenir ataques de scripting y garantiza que los datos almacenados sean confiables y seguros.

**Gestión de Sesiones Seguras**

Implementar medidas de gestión de sesiones seguras, como tokens de sesión con tiempo de expiración y políticas de renovación, para evitar el acceso no autorizado a las cuentas de usuario.

**Auditoría y Registro de Acceso**

Registrar todas las actividades relacionadas con los datos, como el acceso a los registros y las modificaciones. Estos registros pueden ser útiles para la auditoría y pueden ayudar a identificar posibles brechas de seguridad.

**Consentimiento del Usuario**

Obtener el consentimiento claro y específico de los usuarios para recopilar y utilizar sus datos personales. Proporcionar a los usuarios opciones claras para administrar su consentimiento y permitirles retirarlo en cualquier momento.

**Cumplimiento Normativo**

Asegurarse de cumplir con las leyes y regulaciones de protección de datos en el área geográfica en la que opera la aplicación.

**Validación de Entradas**

Validar y sanitizar todas las entradas de usuario para prevenir ataques de inyección (como SQL Injection y XSS). Utilizar técnicas como parámetros preparados en las consultas SQL y escapar correctamente las salidas en las interfaces de usuario para prevenir ataques XSS.

**Gestión de Sesiones Seguras**

Implementar una gestión de sesiones seguras, utilizando tokens de sesión con tiempo de expiración adecuado y políticas de renovación. Asegurarse de que los tokens de sesión sean almacenados de forma segura y no se transmitan en URLs o en texto claro.

**Contraseñas Seguras**

Almacenar contraseñas de forma segura utilizando algoritmos de hash fuertes y únicos para cada usuario. Aplicar técnicas de "salting" para añadir una capa adicional de seguridad.

**Manejo correcto de pantallas de errores**

No revelar detalles sensibles en mensajes de error. Personalizar los mensajes de error para los usuarios y registrar los detalles completos del error para los administradores del sistema.

**Actualizaciones y Parches**

Mantener todas las dependencias y bibliotecas actualizadas para asegurar que se estén utilizando versiones sin vulnerabilidades conocidas. Regularmente aplicar parches de seguridad y actualizaciones tanto en el lado del cliente como en el lado del servidor.

**Validación en el Lado del Servidor**

No confiar en la validación del lado del cliente. Realizar todas las validaciones importantes en el lado del servidor para evitar posibles manipulaciones del cliente.

**Seguridad en las API**

Proteger las API contra ataques como CSRF (Cross-Site Request Forgery) y asegurarse de que todas las operaciones sensibles requieran autenticación y autorización adecuadas.

**Registros de Logs**

Implementar registros y auditorías para realizar un seguimiento de las actividades de los usuarios y del sistema. Estos registros pueden ser útiles para la detección temprana de actividades sospechosas.

**Puntos de vulnerabilidad en la aplicación móvil**

Por el momento, no se ha identificado ninguna vulnerabilidad en la aplicación. Sin embargo, una vez que la aplicación esté desarrollada, es importante considerar posibles puntos de vulnerabilidad que podrían surgir en el futuro. Algunos de estos posibles puntos de vulnerabilidad podrían incluir:

* Robo de información: Esta vulnerabilidad puede ocurrir cuando alguien ingresa a la aplicación haciéndose pasar por un proveedor de servicios falso y accede a los datos del usuario que requiere el servicio. Esto podría permitirle conocer información, como la dirección de domicilio u otros datos importantes del usuario.
* Vulnerabilidad de información incorrecta: Esta vulnerabilidad puede surgir cuando los proveedores de servicios cargan archivos que los usuarios deben revisar antes de contratarlos. En algunos casos, estos archivos pueden contener información falsa o ser desconocidos para los usuarios, lo que dificulta la evaluación de si el proveedor realmente posee la certificación requerida o no. Además, podría ocurrir que los proveedores suban archivos no relacionados para su certificación, lo que confunde a los usuarios que necesitan este servicio y dificulta verificar si están diciendo la verdad o si se trata de una estafa.
* El mal uso de datos personales puede surgir cuando los usuarios proporcionan sus datos y archivos personales para su verificación. Una vez que estos datos se verifican y se comparan con el perfil del usuario, si se detecta que la dirección u otros detalles no coinciden con los archivos proporcionados, se evidencia una vulnerabilidad en el sistema.
* Cuando se implemente el módulo de pago, estamos considerando cómo permitir que los trabajadores ingresen sus claves interbancarias para que nosotros, como administradores, podamos cobrar un cierto porcentaje de comisión. Sin embargo, existe la posibilidad de que ingresen una clave incorrecta, lo que podría dar lugar a un fraude y, potencialmente, causarnos problemas legales.

**Encriptación de datos**

**Contraseñas**

Para las contraseñas se implementa el cifrado de Sha256, con el cual se obtiene un cifrado en un solo sentido, el cual es muy complejo de descifrar, por lo que se obtiene un hash seguro, para este tipo de datos y poder almacenarlos en la base de datos.

**Datos de Tarjetas de Crédito y Pagos**

Los datos de tarjetas de crédito y otros detalles de pago deben ser encriptados durante la transmisión y también mientras se almacenan en la base de datos. Se puede utilizar el protocolo de seguridad SSL/TLS para cifrar la comunicación durante las transacciones. Para el almacenamiento seguro, se debe utilizar cifrado simétrico o asimétrico junto con algoritmos como AES (Advanced Encryption Standard) para cifrar los datos de la tarjeta de crédito antes de almacenarlos en la base de datos.

**Datos de Identificación Personal**

Información como números de seguro social, números de identificación nacional y direcciones pueden ser encriptados. Para esto, se pueden utilizar algoritmos de cifrado como AES con claves fuertes y únicas para cada conjunto de datos.

**Información de Geolocalización**

Si la aplicación recopila datos de ubicación, estos datos también deben ser encriptados para proteger la privacidad de los usuarios. La encriptación se puede lograr utilizando técnicas de cifrado como AES o algoritmos específicos para datos de ubicación.

**Tokens de Sesión y Cookies**

Los tokens de sesión y las cookies utilizadas para la autenticación deben ser encriptados para evitar la interceptación y manipulación. Se pueden utilizar algoritmos de cifrado simétrico como HMAC (Hash-based Message Authentication Code) para firmar y cifrar los tokens.

**Información de Perfil del Usuario**

Datos en el perfil del usuario, como nombre, dirección de correo electrónico y número de teléfono, también pueden ser encriptados. Se puede utilizar cifrado asimétrico con pares de claves públicas y privadas para cifrar estos datos durante el almacenamiento y la transmisión.

**SSL/TLS para Comunicación Segura**

Implementa SSL/TLS (Secure Sockets Layer/Transport Layer Security) para cifrar la comunicación entre la aplicación móvil y el servidor. Esto se logra configurando un certificado SSL/TLS en el servidor y asegurándose de que la aplicación móvil use HTTPS para las solicitudes a través de la capa de transporte segura.

**Uso de Protocolos de Seguridad**

Uso de protocolos seguros para el intercambio de datos, como HTTPS para la comunicación web y OAuth para la autenticación segura. Estos protocolos están diseñados para garantizar la seguridad de la información durante su transmisión y autenticar a las partes involucradas.

**Autenticación de Servidores**

Utiliza certificados digitales confiables para autenticar el servidor y asegúrate de que la aplicación móvil verifique la autenticidad del certificado del servidor durante el proceso de handshake SSL/TLS. Esto previene ataques de intermediarios (MITM).

**Validación de Certificados SSL/TLS**

Realiza una validación adecuada del certificado SSL/TLS en la aplicación móvil para evitar ataques de certificados falsos. La aplicación debe verificar que el certificado del servidor sea válido y esté firmado por una autoridad de certificación confiable.

**Servicio en la nube aplicaba al proyecto de desarrollo móvil**

En este apartado, utilizaremos la plataforma Cloudinary para gestionar las fotografías de los usuarios en sus perfiles. Con Cloudinary, seremos capaces de cargar, almacenar y optimizar las imágenes de los usuarios de manera eficiente, evitando así sobrecargar el almacenamiento de la base de datos. Dado que algunas de las imágenes pueden ocupar mucho espacio, también investigaremos otras plataformas y técnicas que nos permitan comprimir y almacenar las imágenes de manera más eficiente en la base de datos, asegurándonos de que ocupen el menor espacio posible sin comprometer la calidad visual. De esta manera, optimizaremos la gestión de imágenes en nuestros perfiles de usuario.

**Api´s de terceros implementadas**

Mailjet - plataforma de envío de correos electrónicos

Cloudinary – Servicio especializado en el almacenamiento, gestión y entrega de contenido multimedia, como imágenes y videos.

**Justificación de utilización de Apis de terceros**

Tener el servicio como lo es Cloudinary proporciona al usuario una experiencia más agradable permitiendo la personalización del perfil del usuario. Los envíos de correo electrónico proporcionan seguridad con los datos sensibles del usuario, además ahorramos tiempo y recursos al aprovechar soluciones ya existentes en lugar de desarrollar funcionalidades desde cero.   
**¿Cómo se implementarán las Api’s de terceros?**

Mailjet se implementa para atención al usuario, el olvido de contraseña de una cuenta tiene solución por él envió de un correo en el que contiene un token único con el Id del usuario en combinación con la generación del token.

Cloudinary se implementa para la foto de perfil del usuario, donde se tendrá la libertad de elegir si quiere imagen o no.