FACULTAD DE INGENIERÍA

División de Ingeniería Eléctrica (DIE)

- Computo Móvil (674).

Proyecto Final

- ***Nombres***:
  - Jarquin López Daniel Eduardo
  - Pérez López Julio Armando
  - Sanchez Camacho Alan Rodrigo
- ***Profesor***:
  - Ing Marduk Pérez de Lara Domínguez

- ***Grupo***: 03
- **Semestre**: 2024-2
- ***Fecha de entrega:*** 24/05/2024
- ***Equipo***: 02

QuickSync

Objetivo:

El objetivo principal de QuickSync es proporcionar una solución eficiente y accesible para la transferencia inalámbrica de archivos entre dispositivos móviles y computadoras, así como entre dispositivos móviles entre sí. QuickSync busca eliminar las barreras comunes en la transferencia de archivos, como la necesidad de cables físicos o de registrarse en plataformas, ofreciendo una interfaz amigable y un proceso de transferencia directo que no requiere la creación de una cuenta. La aplicación está diseñada para ser intuitiva, permitiendo a los usuarios compartir documentos, fotos, videos y otros tipos de archivos de manera rápida y segura.

Justificación:

La necesidad de transferir información de manera rápida y segura entre dispositivos es más crítica que nunca en nuestro entorno digital actual. A medida que la cantidad de dispositivos por usuario crece y se convierte en una parte integral de la vida diaria, surge la necesidad de una herramienta que facilite una conexión fluida y sin complicaciones. QuickSync responde a esta demanda proporcionando una plataforma que no solo acelera el proceso de transferencia mediante el uso de tecnologías de comunicación avanzadas, sino que también lo hace accesible para cualquier usuario, sin la necesidad de pasar por procesos de registro o configuraciones complicadas. Esto es especialmente valioso en situaciones donde la rapidez y la facilidad de uso son cruciales, como en entornos profesionales, educativos o personales donde el tiempo y la simplicidad son esenciales.

Desarrollo:

Reglas de negocio

1. Transferencia inalámbrica de archivos: La principal regla de negocio es que la aplicación debe permitir la transferencia de archivos de manera inalámbrica entre diferentes dispositivos sin necesidad de cables físicos o conexión a internet.
2. Conexión directa entre dispositivos: QuickSync debe establecer una conexión directa y segura entre los dispositivos participantes, evitando el uso de servidores intermedios.
3. No requerir creación de cuenta: Para facilitar el acceso y uso de la aplicación, no se debe requerir que los usuarios creen una cuenta o se registren.

4. Seguridad y privacidad: Los datos transferidos deben estar encriptados para proteger la privacidad del usuario y la integridad de los archivos durante la transferencia.
5. Compatibilidad multiplataforma: La aplicación debe ser compatible con múltiples sistemas operativos, permitiendo transferencias entre dispositivos Android, iOS, Windows, y macOS.

Requerimientos funcionales

1. Transferencia de archivos:
    - Permitir a los usuarios enviar y recibir diversos tipos de archivos como documentos, imágenes, videos y música.
    - Soportar la transferencia de archivos grandes sin restricciones significativas.
2. Detección automática de dispositivos:
    - Utilizar tecnologías como Bluetooth y Wi-Fi Direct para detectar automáticamente dispositivos cercanos compatibles.
3. Interfaz de usuario:
    - Diseñar una interfaz intuitiva y fácil de usar, que permita la selección y envío de archivos con pocos toques o clics.
4. Grupo de transferencia:
    - Implementar la funcionalidad de enviar archivos a varios dispositivos simultáneamente.
5. Sin uso de datos de internet:
    - Las transferencias deben realizarse sin consumir el plan de datos del usuario.

Requerimientos no funcionales

1. Seguridad:
    - Utilizar encriptación avanzada para proteger los archivos durante la transferencia.
    - Asegurar que no se almacenen archivos en servidores externos.
2. Rendimiento:
    - Garantizar transferencias rápidas y eficientes utilizando las mejores tecnologías disponibles.
    - Minimizar el consumo de batería y recursos del dispositivo durante la transferencia.
3. Escalabilidad:
    - Diseñar la aplicación para que pueda manejar un número creciente de usuarios y transferencias simultáneas sin degradar el rendimiento.
4. Compatibilidad y actualización:

- Asegurar que la aplicación sea compatible con las versiones actuales y futuras de los sistemas operativos soportados.
- Proporcionar actualizaciones periódicas para mejorar la funcionalidad y la seguridad.

5. Usabilidad:
    - Realizar pruebas de usabilidad para asegurar que la aplicación sea accesible y fácil de usar para personas con diferentes niveles de habilidad tecnológica.
6. Mantenibilidad:
    - Estructurar el código de manera que sea fácil de mantener y extender.
    - Documentar adecuadamente el código y las funcionalidades para facilitar futuras mejoras y correcciones de errores.

Para asegurar un desarrollo enfocado y eficiente de QuickSync, es crucial determinar el alcance del proyecto y definir claramente el Producto Mínimo Viable (MVP). Esto permitirá ofrecer una versión funcional de la aplicación que satisfaga las necesidades básicas de los usuarios mientras se minimizan riesgos y se optimizan recursos.

Para definir nuestro MVP de QuickSync debemos establecer los límites y objetivos del desarrollo, abarcando las funcionalidades esenciales, las restricciones y los criterios de éxito.

Entre las funcionalidades esenciales, la aplicación debe permitir la transferencia de archivos de manera inalámbrica entre dispositivos, eliminando la necesidad de cables. Esta transferencia se realizará mediante una conexión directa y segura utilizando tecnologías como Bluetooth y Wi-Fi Direct. La aplicación también debe detectar automáticamente los dispositivos compatibles cercanos y ofrecer una interfaz de usuario intuitiva, facilitando la selección y envío de archivos con pocos toques.

Además, QuickSync debe ser compatible con múltiples sistemas operativos, asegurando su funcionamiento en dispositivos Android, iOS, Windows y macOS. La seguridad es primordial, por lo que se implementará encriptación de extremo a extremo para proteger los archivos durante la transferencia, y no se requerirá la creación de cuentas ni el almacenamiento de datos en servidores externos.

En cuanto a las restricciones, es importante considerar las limitaciones de hardware de los dispositivos más comunes para asegurar un rendimiento óptimo. Además, la aplicación debe cumplir con las normativas locales e internacionales sobre protección de datos y privacidad. El diseño también debe

tener en cuenta diferentes entornos de conectividad, incluyendo áreas con baja cobertura de internet.

El enfoque del MVP es proporcionar valor rápidamente a los usuarios y obtener retroalimentación que guíe el desarrollo futuro. Las características del MVP incluyen la capacidad de enviar y recibir archivos de manera inalámbrica entre dispositivos cercanos mediante conexiones directas usando Bluetooth y Wi-Fi Direct. La detección automática de dispositivos cercanos y una interfaz de usuario sencilla son también fundamentales, junto con la implementación de encriptación básica para proteger los archivos durante la transferencia.

Wireframes y Navegación de Pantallas

Pantalla de Bienvenida

Justificación de Viabilidad: La funcionalidad de bienvenida es completamente viable. No requiere tecnologías complejas y solo muestra información estática y un botón interactivo.

Funcionalidad Implementada: Introducción a la aplicación QuickSync.

Información y Tipo de Datos:

- Mensaje de bienvenida y descripción: Texto estático.
- Botón "Comenzar": Elemento interactivo.

Vigencia del Dato: Permanente, a menos que se actualice la aplicación.

Origen de los Datos: Almacenados localmente en el código de la aplicación.

Operaciones Esperadas: Pulsar el botón "Comenzar" lleva al usuario a la pantalla de solicitud de permisos.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native para el desarrollo de la interfaz.
- Justificación: Permite desarrollar aplicaciones nativas para múltiples plataformas utilizando una única base de código.

Solicitud de Permisos

Justificación de Viabilidad: La solicitud de permisos es esencial y completamente viable, ya que es una funcionalidad estándar en aplicaciones móviles modernas.

Funcionalidad Implementada: Solicitar acceso a ajustes de conectividad.

Información y Tipo de Datos:

- Mensaje de solicitud de permisos: Texto dinámico.
- Botones "Permitir" y "No permitir": Elementos interactivos.

Vigencia del Dato: Dinámico, se muestra cada vez que se necesita permiso.

Origen de los Datos: Generado por la aplicación según requerimientos del sistema operativo.

Operaciones Esperadas:

- "Permitir": Concede permisos y procede a la siguiente pantalla.
- "No permitir": Deniega permisos, limitando la funcionalidad de la aplicación.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native con permisos gestionados a través de las APIs nativas del sistema operativo.
- Justificación: React Native facilita el acceso a APIs nativas necesarias para gestionar permisos.

Introducción del Nombre del Usuario

Justificación de Viabilidad: La captura del nombre del usuario es viable y fácil de implementar.

Funcionalidad Implementada: Captura del nombre del usuario.

Información y Tipo de Datos:

- Nombre del usuario: Cadena de texto.
- Botón "Continuar": Elemento interactivo.

Vigencia del Dato: Hasta que el usuario decida cambiarlo en la configuración.

Origen de los Datos: Ingresado por el usuario.

Operaciones Esperadas:

- Entrada de texto: El usuario escribe su nombre.
- Pulsar el botón "Continuar" guarda el nombre y procede a la siguiente pantalla.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- Justificación: Permite una gestión sencilla de formularios y entradas de usuario.

Selección de Acción (Enviar o Recibir)

Justificación de Viabilidad: La selección de acción es una funcionalidad básica y viable.

Funcionalidad Implementada: Permitir al usuario elegir entre enviar o recibir archivos.

Información y Tipo de Datos:

- Opciones "Enviar" y "Recibir": Elementos interactivos.

Vigencia del Dato: Hasta que se realice otra selección.

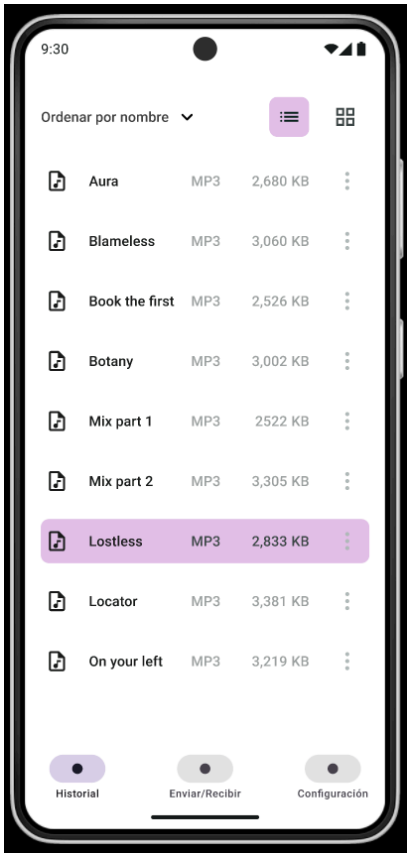
Origen de los Datos: Almacenados localmente en el código de la aplicación.

Operaciones Esperadas:

- Selección de "Enviar" o "Recibir" dirige al usuario a la correspondiente pantalla de acción.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- Justificación: Facilita la creación de interfaces de usuario interactivas y dinámicas.



Historial de Archivos Transferidos

Justificación de Viabilidad: Mostrar un historial de archivos transferidos es viable y una funcionalidad común.

Funcionalidad Implementada: Mostrar archivos que se han transferido y ahora están disponibles localmente.

Información y Tipo de Datos:

- Lista de archivos: Datos dinámicos.
- Opciones de ordenamiento: Elementos interactivos.

Vigencia del Dato: Dinámico, actualizado en tiempo real.

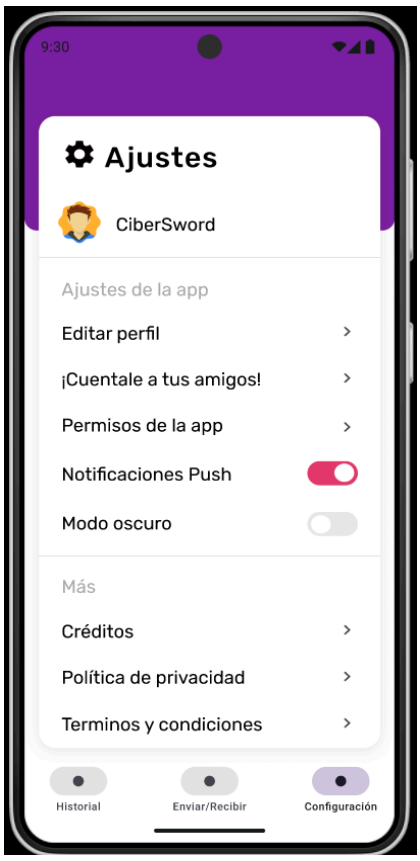
Origen de los Datos: Almacenados localmente en el dispositivo.

Operaciones Esperadas:

- Selección de archivos y opciones de ordenamiento.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native con integración de File System APIs.
- Justificación: Permite el acceso directo a los archivos almacenados en el dispositivo.



Ajustes

Justificación de Viabilidad: La funcionalidad de ajustes es viable y necesaria para cualquier aplicación.

Funcionalidad Implementada: Configuración de la aplicación.

Información y Tipo de Datos:

- Opciones de ajustes: Elementos interactivos.

Vigencia del Dato: Permanente.

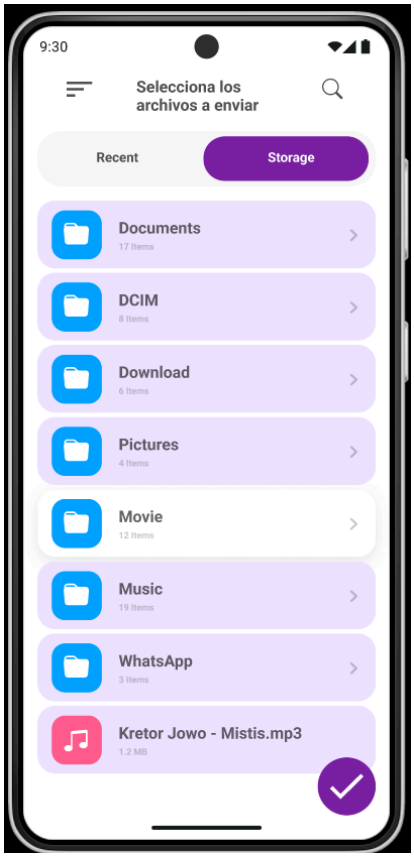
Origen de los Datos: Almacenados localmente en el código de la aplicación.

Operaciones Esperadas:

- Cambios en las configuraciones de perfil, notificaciones, permisos, etc.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- Justificación: Proporciona una forma sencilla de gestionar configuraciones de usuario.



Selección de Archivos para Enviar

Justificación de Viabilidad: Permitir al usuario seleccionar archivos es una funcionalidad básica y viable.

Funcionalidad Implementada: Permitir al usuario seleccionar archivos para enviar.

Información y Tipo de Datos:

- Lista de archivos y carpetas: Datos dinámicos.

Vigencia del Dato: Dinámico, actualizado en tiempo real.

Origen de los Datos: Almacenados localmente en el dispositivo.

Operaciones Esperadas:

- Selección de archivos para transferencia.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native con integración de File System APIs.
- Justificación: Facilita el acceso y la manipulación de archivos del sistema.

Búsqueda de Dispositivos Cercanos

Justificación de Viabilidad: La búsqueda y detección de dispositivos cercanos es viable utilizando tecnologías existentes como Bluetooth y Wi-Fi Direct.

Funcionalidad Implementada: Búsqueda y detección de dispositivos cercanos.

Información y Tipo de Datos:

- Lista de dispositivos detectados: Datos dinámicos.

Vigencia del Dato: Dinámico, mostrado en tiempo real.

Origen de los Datos: Generado por la aplicación en función de la proximidad de otros dispositivos.

Operaciones Esperadas:

- Selección de un dispositivo para iniciar la transferencia de archivos.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native con integración de Bluetooth y Wi-Fi Direct APIs.
- Justificación: Estas tecnologías permiten la detección y comunicación directa entre dispositivos cercanos.

Confirmación de Transferencia

Justificación de Viabilidad: La confirmación de transferencia es una funcionalidad crítica y viable.

Funcionalidad Implementada: Confirmar la transferencia de archivos.

Información y Tipo de Datos:

- Mensaje de confirmación: Texto dinámico.
- Botones para confirmar o cancelar: Elementos interactivos.

Vigencia del Dato: Dinámico, mostrado en tiempo real.

Origen de los Datos: Generado por la aplicación en función de la selección del usuario.

Operaciones Esperadas:

- Confirmación o cancelación de la transferencia.
- Confirmación para comprimir los archivos antes de enviar.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native y librerías de encriptación.
- Justificación: Permite una gestión eficaz de la interfaz de usuario y las interacciones.

Modo Descubrimiento

Justificación de Viabilidad: Activar el modo descubrimiento es viable y facilita la visibilidad de la aplicación a otros dispositivos.

Funcionalidad Implementada: Activar el modo descubrimiento para ser visible a otros dispositivos.

Información y Tipo de Datos:

- Mensaje de activación: Texto estático.

Vigencia del Dato: Permanente.

Origen de los Datos: Almacenados localmente en el código de la aplicación.

Operaciones Esperadas:

- Activación del modo descubrimiento.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native con integración de Bluetooth y Wi-Fi Direct APIs.
- Justificación: Estas tecnologías permiten la visibilidad y detección de dispositivos en proximidad.

Transferencia en Progreso

Justificación de Viabilidad: La funcionalidad de mostrar el progreso de la transferencia es viable y esencial para la transparencia y experiencia del usuario.

Funcionalidad Implementada: Mostrar el progreso de la transferencia de archivos.

Información y Tipo de Datos:

- Nombre del archivo: Texto dinámico.
- Progreso de transferencia: Porcentaje dinámico.
- Tiempo restante estimado: Texto dinámico.

Vigencia del Dato: Dinámico, actualizado en tiempo real durante la transferencia.

Origen de los Datos: Generado por la aplicación en función del estado de la transferencia.

Operaciones Esperadas:

- Visualización del progreso de la transferencia.
- Actualización del porcentaje completado y tiempo restante.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- APIs: Utilización de APIs nativas para la transferencia de archivos (Bluetooth y Wi-Fi Direct).
- Justificación: Proporciona actualizaciones en tiempo real y es compatible con las tecnologías de transferencia utilizadas.

Notificación de Transferencia

Justificación de Viabilidad: La funcionalidad de notificaciones es viable y mejora la usabilidad al mantener al usuario informado.

Funcionalidad Implementada: Mostrar notificación de progreso de transferencia en la barra de notificaciones.

Información y Tipo de Datos:

- Nombre del archivo: Texto dinámico.
- Progreso de transferencia: Porcentaje dinámico.

Vigencia del Dato: Dinámico, actualizado en tiempo real durante la transferencia.

Origen de los Datos: Generado por la aplicación en función del estado de la transferencia.

Operaciones Esperadas:

- Visualización del progreso de la transferencia en la barra de notificaciones.
- Actualización del porcentaje completado.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native con librerías como React Native Push Notification.
- APIs: APIs de notificaciones del sistema operativo.
- Justificación: Permite notificaciones nativas que mantienen al usuario informado incluso cuando la aplicación no está en primer plano.

Transferencia Exitosa

Justificación de Viabilidad: La confirmación de transferencia exitosa es viable y proporciona feedback positivo al usuario.

Funcionalidad Implementada: Confirmar que la transferencia de archivos fue exitosa.

Información y Tipo de Datos:

- Mensaje de éxito: Texto dinámico.
- Cantidad de archivos transferidos: Texto dinámico.

Vigencia del Dato: Dinámico, mostrado al finalizar la transferencia.

Origen de los Datos: Generado por la aplicación al completar la transferencia.

Operaciones Esperadas:

- Visualización de la confirmación de transferencia exitosa.
- Opción para iniciar otra transferencia.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- Justificación: Facilita la gestión de la interfaz de usuario y proporciona una experiencia coherente.

Error de Transferencia

Justificación de Viabilidad: La gestión de errores es viable y crucial para una buena experiencia del usuario.

Funcionalidad Implementada: Mostrar mensaje de error si la transferencia falla.

Información y Tipo de Datos:

- Mensaje de error: Texto estático.
- Botón de reintentar: Elemento interactivo.

Vigencia del Dato: Dinámico, mostrado al ocurrir un error.

Origen de los Datos: Generado por la aplicación en función del estado de la transferencia.

Operaciones Esperadas:

- Visualización del mensaje de error.
- Opción para reintentar la transferencia.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- Justificación: Permite manejar errores de manera eficiente y proporciona opciones para la recuperación de errores.

Compartir la App

Justificación de Viabilidad: La funcionalidad de compartir es viable y puede ayudar a aumentar la base de usuarios.

Funcionalidad Implementada: Permitir a los usuarios compartir la aplicación con otros.

Información y Tipo de Datos:

- Código QR: Imagen dinámica.
- Enlaces de descarga: Texto e imágenes estáticas.

Vigencia del Dato: Permanente.

Origen de los Datos: Generado por la aplicación y almacenado localmente.

Operaciones Esperadas:

- Escaneo del código QR para descargar la aplicación.
- Compartir los enlaces de descarga.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- Librerías: Generación de QR con librerías como react-native-qrcode-svg.
- Justificación: Facilita la creación y visualización de códigos QR y enlaces de descarga.

Créditos

Justificación de Viabilidad: La pantalla de créditos es viable y proporciona reconocimiento a los desarrolladores.

Funcionalidad Implementada: Mostrar información sobre los desarrolladores de la aplicación.

Información y Tipo de Datos:

Mensaje de créditos: Texto estático.

Versión de la aplicación: Texto estático.

Vigencia del Dato: Permanente.

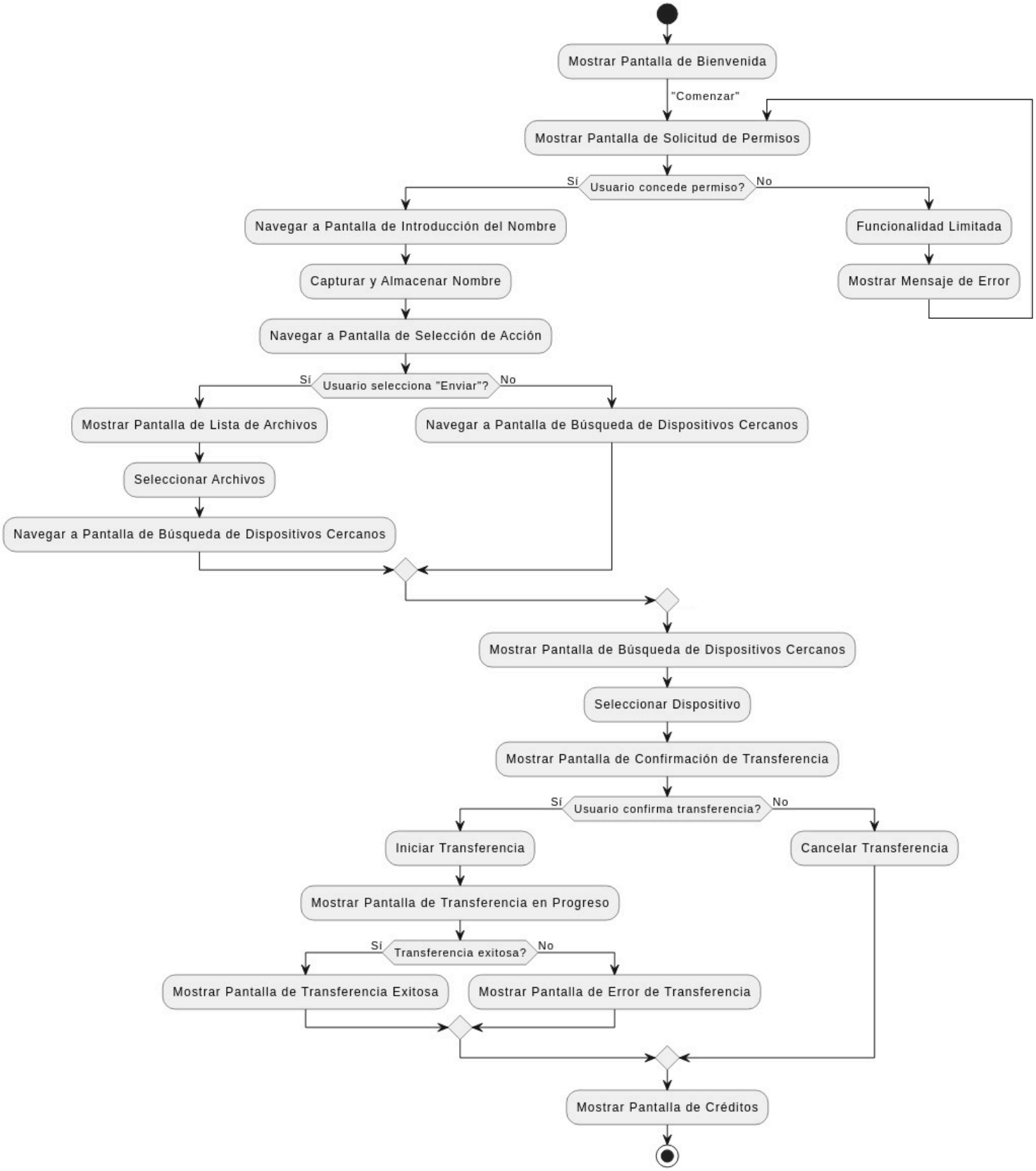
Origen de los Datos: Almacenados localmente en el código de la aplicación.

Operaciones Esperadas:

Visualización de los créditos de los desarrolladores.

Servicios, Frameworks y Kits de Desarrollo:

- Framework: React Native.
- Justificación: Facilita la creación de interfaces estáticas y la gestión de contenido estático.


Mostrar Pantalla de Bienvenida
"Comenzar"
Mostrar Pantalla de Solicitud de Permisos
Usuario concede permiso?
Sí
No
Navegar a Pantalla de Introducción del Nombre
Funcionalidad Limitada
Capturar y Almacenar Nombre
Mostrar Mensaje de Error
Navegar a Pantalla de Selección de Acción
Usuario selecciona "Enviar"?
Sí
No
Mostrar Pantalla de Lista de Archivos
Navegar a Pantalla de Búsqueda de Dispositivos Cercanos
Seleccionar Archivos
Navegar a Pantalla de Búsqueda de Dispositivos Cercanos
Mostrar Pantalla de Búsqueda de Dispositivos Cercanos
Seleccionar Dispositivo
Mostrar Pantalla de Confirmación de Transferencia
Usuario confirma transferencia?
Sí
No
Iniciar Transferencia
Cancelar Transferencia
Mostrar Pantalla de Transferencia en Progreso
Transferencia exitosa?
Sí
No
Mostrar Pantalla de Transferencia Exitosa
Mostrar Pantalla de Error de Transferencia
Mostrar Pantalla de Créditos

Conclusión:

Daniel Eduardo Jarquin López:

Para este proyecto final decidimos retomar el concepto trabajado en la tarea 3, ya que los fundamentos de la aplicación son bastante sólidos y cubren una necesidad de un público más amplio.

Debo reconocer que como equipo mejoramos mucho en el maquetado de las pantallas, siendo muy notable el cambio haciendo la comparación del diseño actual respecto al de la aplicación que presentamos para el Hackaton. En esta ocasión nos capacitamos en el uso de librerías y componentes típicos de kits de desarrollo móvil.

Decidimos utilizar React Native como tecnología principal de desarrollo, debido a que nos permitirá realizar la portabilidad de la aplicación a Android e iOS de manera más sencilla, lo que nos permitirá ahorrar tiempo en la portabilidad y utilizarlo para mejorar la app e implementar otras funciones.

Considero que el proyecto refleja adecuadamente el aprendizaje adquirido durante el curso.

Pérez López Julio Armando:

La fase de planeación y diseño de QuickSync ha sido una etapa crucial en el desarrollo del proyecto. A través de la creación de wireframes y la detallada planificación de la aplicación, hemos podido definir claramente las funcionalidades esenciales y la estructura de la interfaz de usuario. Este enfoque metodológico nos ha permitido anticipar posibles desafíos y planificar soluciones efectivas, asegurando que el desarrollo posterior se realice de manera eficiente y alineada con las necesidades del usuario. Aunque no hemos avanzado en la programación, el trabajo realizado hasta ahora nos proporciona una base sólida para la implementación técnica futura. Estoy convencido de que QuickSync, con su enfoque en la transferencia rápida y segura de archivos, tiene el potencial de ser una herramienta valiosa y ampliamente utilizada.

Sánchez Camacho Alan Rodrigo:

En esta fase del proyecto QuickSync, nos enfocamos en la planeación detallada y el diseño de la aplicación. Aunque no se desarrolló ni programó ninguna aplicación, la creación de wireframes y la definición de requerimientos funcionales y no funcionales nos permitió visualizar cómo debe funcionar QuickSync y qué características debe incluir. Esta planificación exhaustiva es crucial para asegurar que, cuando se inicie el desarrollo, tengamos una guía clara y precisa que facilite la implementación de las funcionalidades previstas. El análisis de viabilidad y la justificación de cada componente nos han ayudado a anticipar posibles desafíos y a establecer un marco sólido para el desarrollo futuro. En resumen, esta etapa ha sido fundamental para sentar las bases de QuickSync, asegurando que la aplicación cumpla con las necesidades de los usuarios de manera eficiente y segura.

Bibliografía:

- *Wi-Fi Direct | Wi-Fi Alliance*. (n.d.). https://www.wi-fi.org/discover-wi-fi/wi-fi-direct

- IONOS editorial team. (2020, August 10). *Multicast DNS: alternative name resolution on a small scale*. IONOS Digital Guide. https://www.ionos.com/digitalguide/server/know-how/multicast-dns/

- Cervera, A. (2024, 4 marzo). Las 11 mejores aplicaciones para pasar datos de Android a Android. Wondershare. https://mobiletrans.wondershare.com/es/phone-transfer/app-to-transfer-data-from-android-to-android.html

- Fernández, Y. (2022, December 9). AirDrop: qué es, cómo funciona y qué puedes compartir con el sistema inalámbrico de Apple. Xataka. https://www.xataka.com/basics/airdrop-que-como-funciona-que-puedes-compartir-sistema-inalambrico-apple