ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

APLICACIÓN PARA LA MEJORA DE LA ACCESIBILIDAD Y USO DE NUEVAS TECNOLOGÍAS

(Application for improving accessibility and use of new technologies)

Para acceder al Título de

Graduado en

Ingeniería de Tecnologías de Telecomunicación

Autor: María Carral Madrazo

Septiembre -2024

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: MARIA CARRAL MADRAZO
Director del TFG: ALBERTO ELOY GARCIA GUTIERREZ
Título: "APLICACIÓN PARA LA MEJORA DE LA ACCESIBILIDAD Y USO DE NUEVAS TECNOLOGÍAS"
Title: "APPLICATION FOR IMPROVING ACCESSIBILITY AND USE OF NEW TECHNOLOGIES"

Presentado a examen el día: 19 de Septiembre de 2024

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre): José Antonio Pereda Fernández antonio.pereda@unican.es
Secretario (Apellidos, Nombre): José Ángel Irastorza Teja angel.irastorza@unican.es
Vocal (Apellidos, Nombre): Roberto Sanz Gil roberto.sanz@unican.es

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo: El Presidente **Fdo: El Secretario**

Fdo: El Vocal **Fdo: El Director del TFG**
(sólo si es distinto del Secretario)

Vº Bº del Subdirector Trabajo Fin de Grado Nº
(a asignar por Secretaría)

Resumen

A lo largo de este trabajo se ha realizado un análisis general del estado actual de la accesibilidad en las nuevas tecnologías, identificando las principales barreras que enfrentan los usuarios que encuentran complicado su manejo. En respuesta a estos desafíos, se ha desarrollado una aplicación móvil diseñada para mejorar la accesibilidad y el uso de estas tecnologías. La aplicación ofrece una interfaz intuitiva y herramientas de asistencia que facilitan la navegación y la utilización de diversas funcionalidades tecnológicas. Este Trabajo de Fin de Grado aborda la necesidad de crear soluciones inclusivas que promuevan la igualdad de oportunidades en la era digital, proporcionando una herramienta que simplifique el uso de la tecnología y promueva la inclusión digital. La metodología empleada incluye un enfoque basado en el desarrollo ágil, con fases de investigación, diseño, desarrollo, pruebas y lanzamiento. Los resultados preliminares indican una mejora significativa en la usabilidad y accesibilidad para los usuarios objetivo.

Abstract

Throughout this work, a general analysis of the current state of accessibility in new technologies has been carried out, identifying the main barriers faced by users who find their handling complicated. In response to these challenges, a mobile application has been developed to improve accessibility and the use of these technologies. The application offers an intuitive interface and assistance tools that facilitate navigation and the use of various technological functionalities. This Final Degree Project addresses the need to create inclusive solutions that promote equal opportunities in the digital age, providing a tool that simplifies the use of technology and promotes digital inclusion. The methodology employed includes an agile development approach, with phases of research, design, development, testing, and deployment. Preliminary results indicate a significant improvement in usability and accessibility for the target users.

Agradecimientos

En primer lugar, deseo expresar mi más sincero agradecimiento a mi tutor, Alberto, por su constante apoyo, orientación y dedicación a lo largo del desarrollo de este trabajo. Sus valiosas sugerencias y su paciencia han sido fundamentales para la realización de este proyecto.

Quiero también agradecer a los profesores que, durante la carrera, me han animado a seguir adelante en los momentos más difíciles y me han brindado las herramientas necesarias para enfrentar cada reto académico. Su motivación y confianza en mi potencial han sido una inspiración constante.

Finalmente, a mi familia, que ha estado siempre a mi lado, brindándome su apoyo incondicional, paciencia y comprensión. Sin su respaldo, este trabajo no habría sido posible.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ACRÓNIMOS

API: Application Programming Interface

CWE: Common Weakness Enumeration

GDPR: Reglamento General de Protección de Datos

ISO: International Standard Organization

MVVM: Model-View-ViewModel

OWASP: Open Web Application Security Project

RGPD: Reglamento General de Protección de Datos

TTS: Text-To-Speech

WCAG: Web Content Accessibility Guidelines

1 INTRODUCCIÓN

Las nuevas tecnologías han transformado de manera rápida y profunda la forma en que interactuamos con el mundo, debido a esto, muchas personas enfrentan barreras a diario para acceder y usar estas tecnologías de manera efectiva. La accesibilidad digital se ha convertido en un aspecto crucial para garantizar que todas las personas, independientemente de sus capacidades, puedan beneficiarse de los avances tecnológicos.

Muchas personas, ya sea por falta de experiencia, conocimiento o por la complejidad de las interfaces, encuentran difícil adaptarse a las herramientas digitales modernas. Esta situación crea una brecha digital que limita la capacidad de estas personas para participar plenamente en la sociedad y aprovechar las oportunidades que ofrece la tecnología.

En respuesta a estos desafíos, se plantea el desarrollo de una aplicación móvil que facilite el acceso y uso de las nuevas tecnologías para todas aquellas personas que encuentran dificultades en su manejo. La motivación detrás de este proyecto surge de la necesidad de crear soluciones inclusivas que promuevan la igualdad de oportunidades en la era digital. Al desarrollar una aplicación que simplifique y facilite el uso de estas tecnologías, se busca no solo proporcionar una herramienta útil, sino también promover una mayor inclusión digital y reducir la brecha tecnológica.

Este Trabajo de Fin de Grado se centra en el desarrollo de una aplicación móvil diseñada específicamente para mejorar la accesibilidad y el uso de las nuevas tecnologías. La aplicación ofrecerá una interfaz intuitiva y herramientas de asistencia que permitirán a los usuarios navegar y utilizar tecnologías modernas de manera más eficiente y efectiva.

1.1 MOTIVACIÓN Y OBJETIVOS

Este proyecto surge de la creciente necesidad de soluciones inclusivas que promuevan la igualdad de oportunidades en la era digital. La velocidad acelerada de aparición de los avances tecnológicos está dejando a gente atrás, personas que simplemente no pueden adaptarse a los cambios tan rápido como estos aparecen. Esta disparidad ha dado lugar a una brecha digital significativa, que limita la capacidad y el acceso de los usuarios para beneficiarse plenamente de las innovaciones tecnológicas. Esta brecha no solo afecta a personas de edad avanzada sino también a los individuos con discapacidades, a aquellos con niveles de alfabetización digital bajos y a quienes enfrentan barreras económicas y sociales.

Este proyecto propone una respuesta a esta situación mediante el desarrollo de una aplicación móvil diseñada para facilitar el acceso y el uso de las nuevas tecnologías a aquellas personas que enfrentan a dificultades en su manejo. La premisa fundamental de esta aplicación es proporcionar una herramienta que simplifique la interacción con la tecnología, promoviendo así la inclusión digital y mitigando la brecha tecnológica que afecta a numerosos usuarios en la actualidad.

El objetivo principal del proyecto es crear una aplicación móvil que de soluciones que permitan mejorar la accesibilidad y usabilidad de las nuevas tecnologías, aportando atajos e indicaciones claras, que las haga más comprensibles y fáciles de usar para la mayoría de las personas independientemente de su nivel de competencia digital.

Para alcanzar este objetivo el proyecto se enfoca en tres áreas principales. Primero, se desarrollará una interfaz de usuario clara, intuitiva y de fácil navegación. Este diseño se orientará hacia la simplicidad y la eficacia, eliminando elementos innecesarios.

En segundo lugar, la aplicación integrará herramientas de asistencia como guías paso a paso. Un componente clave de estas herramientas será el uso de una ventana flotante que guiará a los usuarios a través de instrucciones, permitiendo a los usuarios recibir orientación continua sin tener que estar cambiando de aplicación para comprobar los pasos a seguir.

Finalmente se llevarán a cabo pruebas de usabilidad y se recogerá *feedback* detallado de los usuarios para evaluar el rendimiento y la funcionalidad de la aplicación. Estas evaluaciones permitirán identificar áreas de mejora y realizar ajustes continuos para optimizar la experiencia del usuario. La retroalimentación será fundamental para asegurar que la aplicación cumpla con sus objetivos de accesibilidad e inclusión, adaptándose a las necesidades cambiantes de los usuarios.

1.2 ORGANIZACIÓN DEL DOCUMENTO

Este documento está estructurado en cuatro capítulos principales que detallan el desarrollo y la implementación de una aplicación móvil destinada a mejorar la accesibilidad y el uso de las nuevas tecnologías para personas que encuentran su manejo complicado. En el primer apartado se analizan las necesidades del usuario, así como los requisitos para que una aplicación sea accesible, profundizando en sus características y las barreras tecnológicas existentes. En el segundo capítulo se realiza el desarrollo completo de la aplicación, poniendo énfasis en su estructura, sus funcionalidades sin olvidar el diseño. Por último, se sacarán conclusiones y se analizarán posibles líneas futuras de implementación. Para ello se analizan las respuestas de un grupo reducido de personas que han probado la aplicación.

2 MARCO TEÓRICO

El presente capitulo tiene como objetivo presentar las dificultades actuales presentes en el manejo de las nuevas tecnologías, así como profundizar en las características de accesibilidad que pueden hacer de una aplicación sencilla de usar, logrando así la autonomía de las personas a la hora de realizar tareas con dichas tecnologías.

2.1 BARRERAS TECNOLÓGICAS

Se comienza con una definición de las barreras tecnológicas para posteriormente diferenciar los tipos de barreras existentes, así como el impacto que tienen en el usuario final.

2.1.1 Definición y concepto

Las barreras tecnológicas se refieren a los obstáculos que dificultan o limitan la implementación y el uso de las tecnologías en diferentes contextos [1]. La conocida como brecha digital es la barrera tecnológica que muestra la diferencia de acceso, uso y aprovechamiento de las tecnologías entre diferentes grupos sociales, económicos y geográficos. Tanto esta, como otras muchas, pueden mostrarse a través de limitaciones en la estructura tecnológica, la falta de conocimiento técnico o la obsolescencia de ciertos componentes [2].

2.1.2 Tipos de barreras

Las barreras tecnológicas pueden clasificarse en las siguientes categorías:

2.1.2.1 Barreras de hardware

Dentro de este tipo de barreras se incluyen problemas como la incompatibilidad entre dispositivos, ya sean nuevos o existentes, las limitaciones en la capacidad de procesamiento o almacenamiento y la rápida obsolescencia de los equipos debido a los avances tecnológicos. Estos problemas pueden desembocar en gastos adicionales para los usuarios al tener estos que actualizar o reemplazar su hardware para poder seguir utilizando determinadas tecnologías. Un ejemplo claro de una barrera de hardware es cuando una aplicación requiere de gran capacidad de procesamiento para funcionar y que los dispositivos más antiguos no pueden ofrecer, lo cual obliga a los usuarios a invertir en nuevos equipos [3].

2.1.2.2 Barreras de software

Las barreras de software se describen como dificultades en la compatibilidad de los diferentes programas con los diferentes sistemas operativos, problemas derivados de la falta de actualizaciones o mantenimiento adecuado y dificultades resultantes de los desafíos en la integración con otros sistemas o software ya existentes. Estos problemas pueden hacer que un programa no funcione correctamente o que se vean reducidas sus capacidades. Por ejemplo, una aplicación puede no ser compatible con versiones más antiguas o diferentes a la versión de sistema operativo para la que está diseñada la aplicación [4].

2.1.2.3 Barreras de compatibilidad

Las barreras de compatibilidad comprenden desde la falta de estándares o protocolos que faciliten la compatibilidad tecnológica hasta los problemas de interoperabilidad entre sistemas y aplicaciones. Estas barreras pueden llevar a que diferentes componentes de un sistema no puedan trabajar juntos de manera eficiente. Un ejemplo de este tipo de barreras puede ser cuando una aplicación necesita interactuar con un sistema y, al no ser compatible con el mismo, su funcionalidad se ve limitada [5].

2.1.2.4 Barreras de conectividad

Este tipo de barreras comprenden los diferentes problemas de las infraestructuras de redes locales, así como las limitaciones en el acceso a internet que imposibilitan una conexión fiable y rápida. En cuanto a las repercusiones en el usuario final, esta limitación en la conectividad puede impedir el completo acceso a una aplicación viéndose limitadas algunas funcionalidades. Por ejemplo, en áreas con infraestructura de internet deficiente los usuarios pueden tener dificultades para utilizar aplicaciones que requieren una conexión a internet constante y rápida [6].

2.1.2.5 Barreras de seguridad

Las barreras de seguridad se basan en los riesgos asociados a la seguridad de la información y a la protección de datos. Este tipo de barreras puede desincentivar a los usuarios a utilizar una aplicación, al entender que sus datos se pueden ver

comprometidos o por tener preocupaciones con respecto al cumplimiento de regulaciones como el RGPD (reglamento General de Protección de Datos) [7].

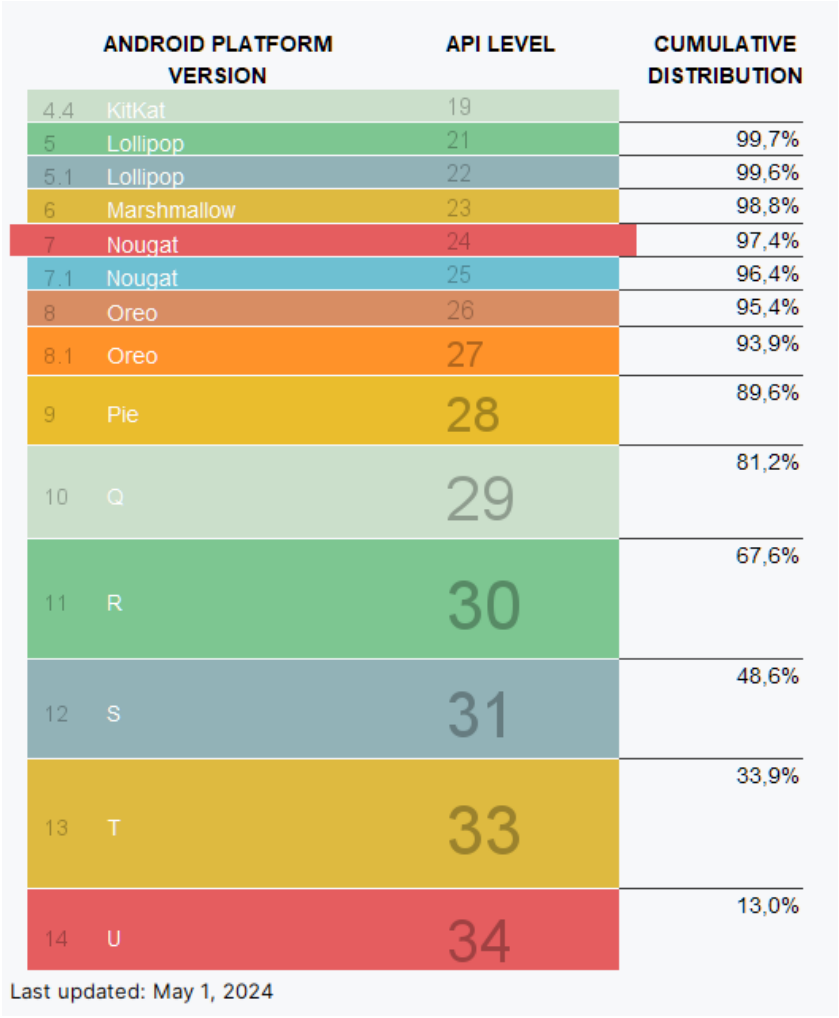
2.1.3 Relación entre Barreras Tecnológicas y Accesibilidad

Ahora que ya se han definido y clasificado las diferentes barreras tecnológicas se procede a relacionar estas barreras mostrando como afectan a la accesibilidad de las personas.

La accesibilidad, en este caso, es hacer que las aplicaciones puedan ser utilizadas por personas independientemente de sus capacidades o circunstancias, brindándoles igualdad de acceso a la información, el acceso en sí, y los servicios [8], intentando así superar todas las barreras anteriormente definidas.

Para superar la barrera de hardware es importante tener en cuenta que los dispositivos más antiguos tienen menos capacidad de cómputo en sus procesadores y sus capacidades se ven degradadas por el uso. En media, los móviles se reemplazan cada 33,6 meses, según un estudio echo entre 2013 y 2020 [9], pero hay que tener en cuenta que los teléfonos móviles, a menudo, se cambian por un modelo más nuevo, no por fallo, sino porque éste proporciona ventajas atractivas. Además, esta tasa de cambio varía entre países siendo, por ejemplo, en Estados Unidos cada 2 o 3 años [10].

En cuanto a las barreras de software lo importante es tener en cuenta cuántos dispositivos pueden acceder a la aplicación. Así, por ejemplo, desde la aplicación Android Studio, al crear el proyecto, se muestra una tabla comparativa como la mostrada en la Ilustración 1. En cuanto a las versiones de IOS, el sistema operativo de Apple, existe una tabla similar, como se muestra en la Ilustración 2 [11].



ANDROID PLATFORM VERSION		API LEVEL	CUMULATIVE DISTRIBUTION
4.4	KitKat	19	
5	Lollipop	21	99,7%
5.1	Lollipop	22	99,6%
6	Marshmallow	23	98,8%
7	Nougat	24	97,4%
7.1	Nougat	25	96,4%
8	Oreo	26	95,4%
8.1	Oreo	27	93,9%
9	Pie	28	89,6%
10	Q	29	81,2%
11	R	30	67,6%
12	S	31	48,6%
13	T	33	33,9%
14	U	34	13,0%

Last updated: May 1, 2024

Ilustración 1 Comparativa de las versiones de Android en relación con el nivel API y el porcentaje de posibles usuarios

Estos datos son importantes ya que las aplicaciones diseñadas para un sistema operativo no suelen funcionar en otro diferente, limitando así el número de posibles usuarios. Además de estas limitaciones, es necesario proporcionar una interfaz que sea amigable con el usuario, teniendo en cuenta las necesidades de este, así como sus posibles limitaciones, y haciendo sencillo su uso. Algunas características que son necesarias para que una aplicación se accesible son los estándares definidos dentro de las Pautas de Accesibilidad para el Contenido Web (WCAG). De estas directrices se destaca, por ejemplo, como el uso de diferentes colores puede hacer una aplicación imposible de usar, o como las indicaciones precisas son ideales para el correcto uso de la aplicación por parte de los usuarios [12].

Version	Released	Cumulative usage	Last iOS for
iOS 17	2023	70.5%	*Not applicable*
iOS 16	2022	87.2%	iPhone X, iPhone 8
iOS 15	2021	95.0%	iPhone 7, iPhone SE (gen 1), iPhone 6s
iOS 14	2020	96.4%	—
iOS 13	2019	97.0%	
iOS 12	2018	98.5%	iPhone 6, iPhone 5s
iOS 11	2017	99.1%	—
iOS 10	2016	99.5%	iPhone 5c, iPhone 5
iOS 9	2015	99.8%	iPhone 4s
iOS 8	2014	99.8%	—
iOS 7	2013	99.9%	iPhone 4
iOS 6	2012	99.9%	iPhone 3GS
iOS 5	2011	99.9%	—
iOS 4	2010	100.0%	iPhone 3G
iOS 3	2009	*No data*	iPhone (gen 1)
iOS 2	2008		—
iOS 1	2007		

These figures were last updated on **June 12, 2024** using data from Statcounter GlobalStats and this script.
You may update the figures yourself with a pull request.

Ilustración 2 Comparativa de las versiones de IOS en relación con el nivel API y el porcentaje de posibles usuarios

En lo relativo a las barreras de compatibilidad es esencial, al desarrollar una aplicación, el tener en cuenta los dispositivos en los que se va a usar. Para empezar, la mayor barrera de compatibilidad entre aplicaciones móviles es cuando una aplicación esta desarrollada para Android y no es posible que funcione en un dispositivo con sistema operativo de tipo IOS, y viceversa. Este hecho llega a ser tan importante al caer en la cuenta de que el 72,15% de los dispositivos móviles utiliza un sistema operativo basado en Android, frente a un 27,19% que presentan los basados en IOS. Hay que considerar también que los datos anteriores son a nivel mundial, y en determinados países los porcentajes son bien diferentes. A la hora de escoger para qué sistema operativo se va a programar la decisión depende mucho del público objetivo [13].

Cuando se habla de las barreras de conectividad va a depender mucho de en qué parte del mundo se esté observando, ya que no todos los países presentan una misma conectividad a internet. La conectividad puede variar entre diferentes localidades y hasta entre pueblos cercanos. Según datos de 2022, a nivel mundial el 63% de las personas tienen acceso a internet [14], pero esta conectividad no está asegurada en todo momento, ni a una velocidad que permita navegar por internet sin pausas o interrupciones del servicio. La velocidad media de internet varía tanto, que Emiratos

Árabes Unidos lidera con unos 303.21 Mbps en su acceso a internet mediante teléfonos móviles, mientras que en Cuba apenas se superan los 3,54 Mbps [15]. Por esto es necesario tener en cuenta el uso que se va a realizar de las diferentes aplicaciones, y si se va a ver afectada la capacidad de los usuarios de poder acceder a todas las funcionalidades de la aplicación.

Por último, las barreras de seguridad, al igual que las barreras anteriores, no son fijas y varían entre países en lo relativo, sobre todo, a la protección de datos. En la Unión Europea la protección de datos está regida mediante los RGPD que protegen a los usuarios mientras sus datos están siendo tratados por las aplicaciones. Por ello, y con el objetivo de operar dentro de la legalidad previo al desarrollo de las aplicaciones, es necesario conocer las regulaciones de seguridad a las que deben estar sometidas y cumplir sus parámetros.

Al igual que la protección de sus datos dentro de las aplicaciones es importante para los usuarios, también lo es que la aplicación sea segura y no suponga un riesgo para el teléfono móvil en su conjunto y todos los procesos que operan en él. Esta es la parte más complicada, ya que no existen pruebas concretas que puedan certificar que las aplicaciones son enteramente seguras e impenetrables. Simplemente existen buenas prácticas y marcos de seguridad, como el Top 10 de OWASP (Open Web Application Security Project) [16], el CWE (Common Weakness Enumeration) [17] o el estándar ISO 27001 [18]. En conjunto, todos estos factores pueden resultar en que el usuario sea renuente a instalar una aplicación, ya sea por miedo a cómo vayan a ser tratados sus datos o que, a una vez instalada, el usuario la desinstale por la elevada dificultad en el acceso a la misma. A su vez, esta elevada dificultad de acceso puede ser debida a un inicio de sesión con unas características inasumibles, o por un proceso de autentificado largo que frustra al usuario antes de haber siquiera accedido al contenido de la aplicación.

3 MARCO PRÁCTICO

A continuación, se expone el desarrollo práctico del proyecto, detallando las fases clave de implementación, las herramientas utilizadas y los métodos aplicados para alcanzar los objetivos propuestos. Esta sección describe en profundidad cómo se llevó a cabo la creación y evaluación de la aplicación, con un enfoque en los aspectos técnicos y operativos del proceso.

3.1 TECNOLOGÍAS UTILIZADAS

Para el desarrollo de la aplicación se ha utilizado *Android Studio*, el entorno de desarrollo integrado (IDE) oficial para aplicaciones Android. *Android Studio* proporciona una variedad de herramientas, como el emulador de Android, un depurador integrado y un editor de código avanzado, que facilitan la creación y prueba de aplicaciones móviles. La versión utilizada en este proyecto fue *Android Studio Koala* (2024.1.1), lo cual garantiza compatibilidad con las últimas versiones de Android.

La elección de *Java* como lenguaje de programación principal se debió a su fiabilidad, estabilidad y gran comunidad de desarrolladores, lo que asegura un soporte continuo y documentación extensa. Java es un lenguaje ampliamente utilizado en el desarrollo de aplicaciones móviles debido a su capacidad para manejar procesos en segundo plano y su perfecta integración con el *framework* de Android. Esto resulta crucial para implementar características como la interacción con servicios de geolocalización y el envío de datos en tiempo real, elementos esenciales para la funcionalidad principal de la aplicación.

Además de Android Studio y Java, se hizo uso de una serie de librerías nativas y de terceros que ayudaron a mejorar el rendimiento y la experiencia del usuario. Las librerías nativas proporcionadas por Android *Jetpack* fueron clave para garantizar que la aplicación fuera eficiente y mantenible. Las principales librerías utilizadas incluyen:

- **libs.appcompat**: Asegura la compatibilidad con versiones anteriores de Android, lo que permite que la aplicación funcione sin problemas en dispositivos con versiones antiguas del sistema operativo.
- **libs.material**: Esta librería de *Material Design* facilitó la implementación de componentes visualmente coherentes como botones flotantes, *snackbars*, *toolbars* y otros elementos interactivos, mejorando la experiencia del usuario.
- **libs.constraintlayout**: Utilizada como gestor de diseño, *ConstraintLayout* permite crear interfaces complejas de manera flexible y optimizada. Esto mejora el rendimiento de la aplicación al reducir la cantidad de niveles en la jerarquía de vistas.
- **libs.navigationFragment** y **libs.navigationUi**: Son fundamentales para la gestión de la navegación entre fragmentos, asegurando una experiencia fluida y coherente para el usuario.
- **libs.lifecycleLivedataKtx** y **libs.lifecycleViewmodel:** Estas librerías aseguran que los datos observables se mantengan incluso después de cambios en la configuración de la aplicación, como la rotación de pantalla.

Para la gestión del código y las pruebas, se utilizaron las siguientes herramientas:

- **libs.junit:** Librería utilizada para realizar pruebas unitarias, permitiendo verificar la funcionalidad de cada componente de la aplicación de forma individual.
- **libs.espressoCore**: Utilizada para las pruebas de la interfaz de usuario. Con *Espresso*, se simularon interacciones de los usuarios y se evaluó el comportamiento de la aplicación bajo diferentes escenarios de uso.

Para poder usar estas librerías primero se definieron en archivo *libs.versions.toml*, que gestiona las versiones de la librerias y los *plugins*, como se ve en la Ilustración 3 y posteriormente se añadieron al archivo *build.gradle*, que es donde se hace ya la implementación de las librerías, como se ve en la Ilustración 4.

```
[libraries]
appcompat = { module = "androidx.appcompat:appcompat", version.ref = "appcompat" }
material = { module = "com.google.android.material:material", version.ref = "material" }
constraintlayout = { module = "androidx.constraintlayout:constraintlayout", version.ref = "constraintlayout" }
navigationFragment = { module = "androidx.navigation:navigation-fragment", version.ref = "navigationFragment" }
navigationUi = { module = "androidx.navigation:navigation-ui", version.ref = "navigationUi" }
legacySupportV4 = { module = "androidx.legacy:legacy-support-v4", version = "1.0.0" }
lifecycleLivedataKtx = { module = "androidx.lifecycle:lifecycle-livedata-ktx", version.ref = "lifecycle" }
lifecycleViewmodel = { module = "androidx.lifecycle:lifecycle-viewmodel", version.ref = "lifecycle" }
fragment = { module = "androidx.fragment:fragment", version = "1.8.3" }
espressoCore = { module = "androidx.test.espresso:espresso-core", version.ref = "espressoCore" }
junit = { group = "junit", name = "junit", version.ref = "junitJunit" }
[plugins]
androidApplication = { id = "com.android.application", version.ref = "agp" }
```

Ilustración 3 Implementación de las librerías en el archivo libs.versions.toml

```
dependencies {
    implementation(libs.appcompat)
    implementation(libs.material)
    implementation(libs.constraintlayout)
    implementation(libs.navigationFragment)
    implementation(libs.navigationUi)
    implementation(libs.legacySupportV4)
    implementation(libs.lifecycleLivedataKtx)
    implementation(libs.lifecycleViewmodel)
    implementation(libs.fragment)
    testImplementation libs.junit
    androidTestImplementation(libs.espressoCore)
}
```

Ilustración 4 Implementación de las dependencias en el archivo build.gradle

3.2 LA ARQUITECTURA

La arquitectura de la aplicación está basada en el patrón **Model-View-ViewModel** (MVVM), que facilita una clara separación entre la lógica de negocio (*Model*), la interfaz de usuario (*View*) y la lógica de presentación (*ViewModel*) [19]. Este patrón fue elegido por su capacidad para mejorar la escalabilidad y el mantenimiento de la aplicación, asegurando que los cambios en la interfaz de usuario no afecten a la lógica de negocio y que los datos sean gestionados de forma eficiente.

3.2.1 Model

El *Model* se encarga de gestionar la lógica de negocio, incluidos los datos. En este caso almacena y organiza la información relacionada con los tutoriales de la aplicación como, por ejemplo, los pasos detallados que los usuarios deben seguir. Este modelo puede obtener datos desde una base de datos local o remota, y se mantiene independiente de la interfaz de usuario. Esto garantiza que los datos siempre estén disponibles, independientemente de cómo se presente la interfaz.

En esta aplicación, el *Model* incluye las clases que gestionan los datos de los tutoriales. Cada tutorial está compuesto por pasos detallados que son almacenados y recuperados desde el *Model*. Además, los servicios auxiliares como TTS (*Text-to-Speech*), que permite la lectura en voz alta de los tutoriales, también son tratados como parte del *Model*, ya que este servicio se encarga de realizar una tarea específica relacionada con la transformación de datos (de texto a voz) sin interacción directa con la interfaz de usuario.

3.2.2 View

La *View* es la interfaz de usuario, la parte visible de la aplicación que interactúa directamente con el usuario. En el patrón *MVVM*, la *View* debe ser lo más simple posible, mostrando solo los datos proporcionados por el *ViewModel*. En esta aplicación, la *View* se encarga de mostrar listas de tutoriales y los pasos correspondientes cuando el usuario

selecciona una opción, siguiendo las pautas de *Material Design* para mantener la coherencia visual y la accesibilidad. La *View* se suscribe a los cambios en los datos que proporciona el *ViewModel* a través de *LiveData*, asegurando que cualquier cambio en los datos se refleje automáticamente en la interfaz sin necesidad de interacción adicional por parte del usuario.

En este caso, las *Views* son las actividades y fragmentos que componen la interfaz de usuario, donde los usuarios pueden seleccionar tutoriales, ver los pasos de cada tutorial, o escuchar el contenido a través del servicio *TTS*. Además, la funcionalidad de la *Floating Window* (ventana flotante) se gestiona dentro de la *View*, ya que es un componente visual que permite al usuario seguir el tutorial mientras realiza otras tareas. Sin embargo, la lógica para mostrar o cerrar la ventana flotante está completamente gestionada por el *ViewModel*.

3.2.3 ViewModel

El *ViewModel* actúa como intermediario entre la *View* y el *Model*, tal como se muestra en la Ilustración 5. Se encarga de procesar los datos que provienen del *Model* y de proporcionar estos datos de manera eficiente a la *View*, utilizando mecanismos como *LiveData* para observar los cambios de datos en tiempo real. Una ventaja clave del *ViewModel* es que persiste durante los cambios de configuración, como la rotación de la pantalla, lo que significa que los datos no se pierden cuando la vista se recrea. Esto es especialmente útil en aplicaciones móviles, donde los cambios de orientación pueden causar problemas de rendimiento si los datos no se manejan correctamente.

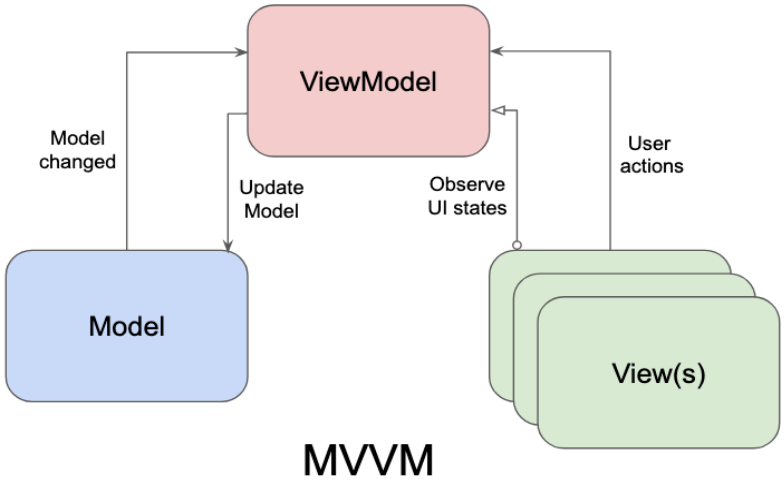


Ilustración 5 Arquitectura MVVM.
Imagen sacada de https://medium.com/@dheerubhadoria/android-mvvm-how-to-use-mvvm-in-android-example-7dec84a1fb73

En este caso, las *Views* son las actividades y fragmentos que componen la interfaz de usuario, donde los usuarios pueden seleccionar tutoriales, ver los pasos de cada tutorial, o escuchar el contenido a través del servicio *TTS*. Además, la funcionalidad de la *Floating Window* (ventana flotante) se gestiona dentro de la *View*, ya que es un componente

visual que permite al usuario seguir el tutorial mientras realiza otras tareas. Sin embargo, la lógica para mostrar o cerrar la ventana flotante está completamente gestionada por el *ViewModel*.

3.3 El diseño

En el diseño de la interfaz de usuario se siguieron las directrices de *Material Design* para asegurar una navegación fluida y coherente [20]. *Material Design*, creado por *Google*, es un sistema de diseño que proporciona una serie de principios y guías visuales que se utilizan para crear interfaces de usuario consistentes, intuitivas y accesibles. Este enfoque se basa en el concepto de capas físicas, utilizando sombras y movimientos sutiles para crear una experiencia de usuario que simula el mundo real. Además de su enfoque en la estética, *Material Design* se centra en la accesibilidad, asegurando que las aplicaciones sean utilizables para un público diverso, incluidas personas con discapacidades visuales o motoras [21].

Material Design hace énfasis en la claridad visual, la interactividad y la accesibilidad. Todo esto se implementó a través del uso de botones de formas y colores adecuados.

3.3.1 Botones redondeados con sombras sutiles

Los botones en la aplicación siguen el diseño característico de *Material Design*, cuyo ejemplo se muestra en la Ilustración 6, donde se utiliza una elevación (a través de sombras) para indicar que un botón es interactivo. Los bordes redondeados hacen que los botones sean más amigables y fáciles de identificar visualmente, mientras

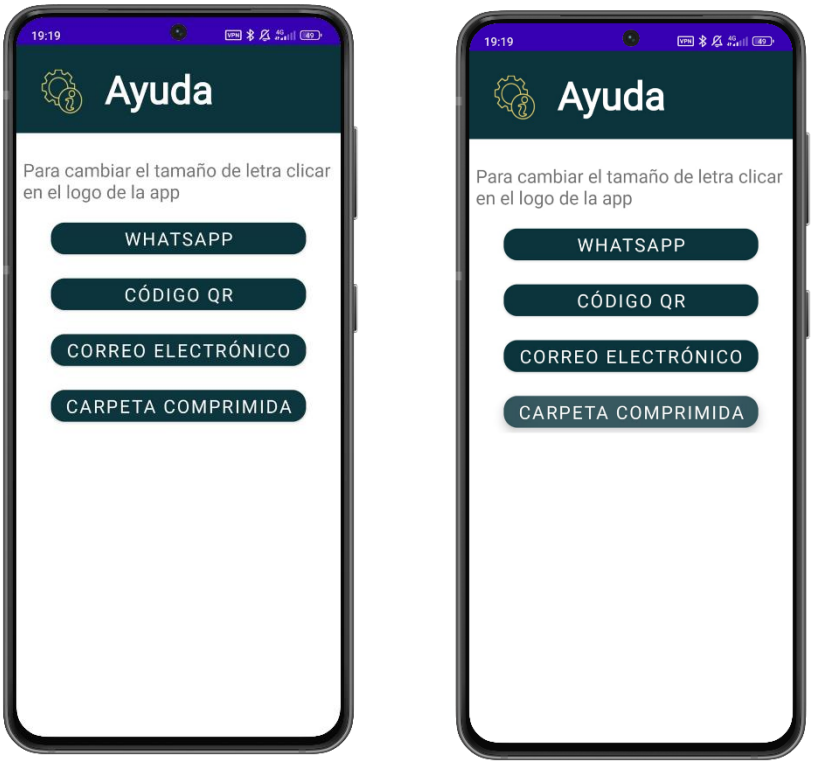


Ilustración 6 Muestra en la aplicación de los botones redondeados y presionables

que las sombras sutiles crean una sensación de profundidad, dando al usuario un indicativo claro de que esos elementos son "presionables". Esta retroalimentación visual es clave para que el usuario entienda de manera intuitiva que puede interactuar con esos elementos.

3.3.2 Uso adecuado de colores alto contraste

Se ha asegurado que los colores de la interfaz no solo sean estéticamente agradables, sino también accesibles. Por ejemplo, el texto blanco sobre un fondo verde oscuro en los botones asegura un contraste alto que mejora la legibilidad, especialmente en condiciones de poca luz o para usuarios con visión reducida. Este diseño sigue las recomendaciones de las pautas de accesibilidad WCAG, que sugieren un contraste mínimo de *4.5:1* y un contraste recomendable de *7:1* entre el texto y el fondo para garantizar que sea legible para todos los usuarios. Este contraste hace que los textos sean fácilmente distinguibles y que la navegación sea intuitiva, incluso para usuarios con discapacidades visuales leves o que utilicen dispositivos con pantallas de baja calidad.

En concreto en esta aplicación se han usado 4 colores, negro para las letras, blanco para fondos y para las letras de los botones, verde (*#0C343D*) para los botones y dorado(*#FFC600*) para mostrar cuando están pulsado determinados botones.

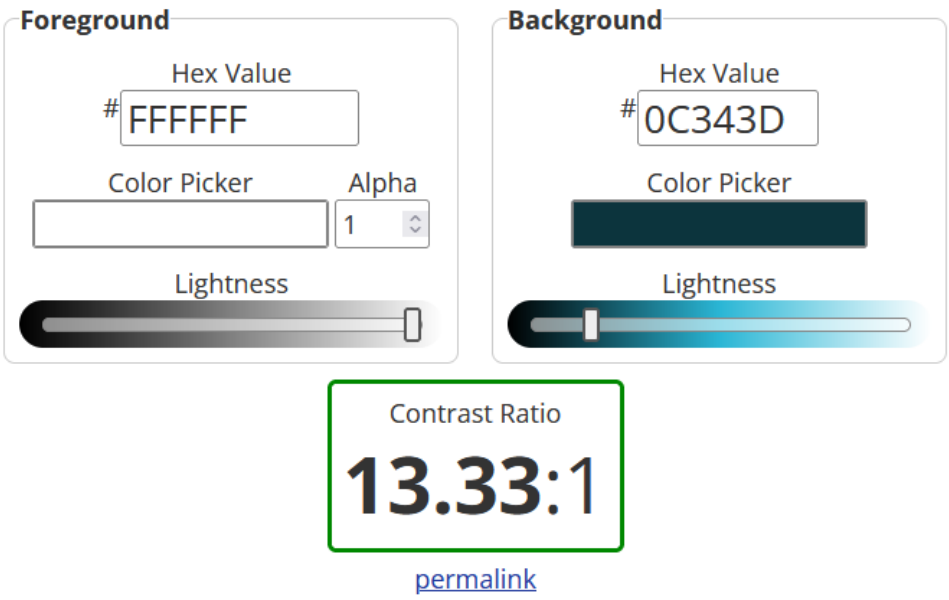


Ilustración 7 Comprobación del contraste de los colores de los botones. Imagen sacada de https://webaim.org/resources/contrastchecker/

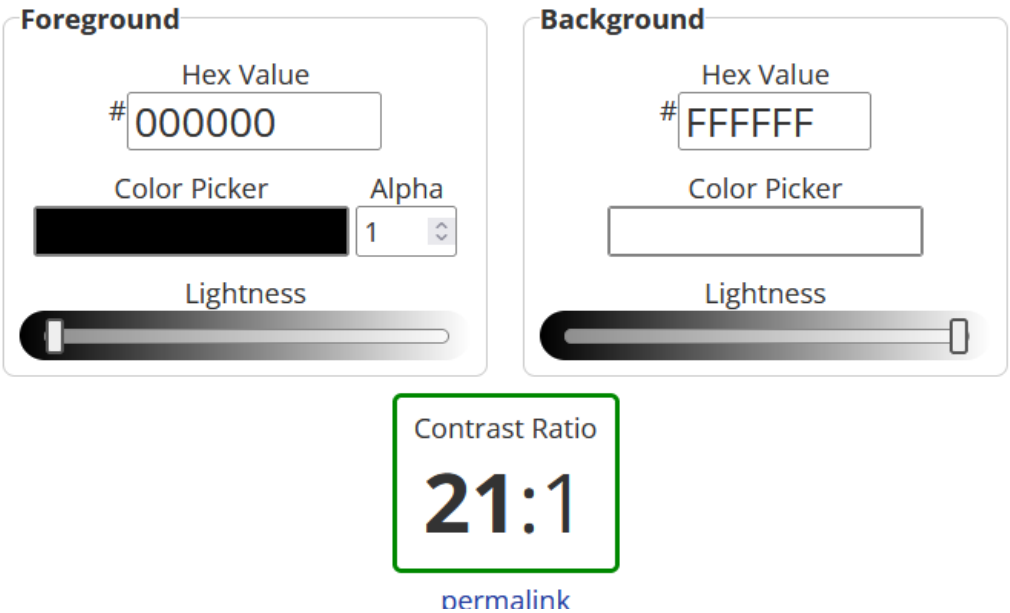


Ilustración 8 Comprobación del contraste de los colores de los textos. Imagen sacada de https://webaim.org/resources/contrastchecker/

Para comprobar si cumple el contraste mínimo se ha usado WebAIM, una organización sin ánimo de lucro líder en el campo de la accesibilidad web que tiene como objetivo mejorar la accesibilidad en la web para personas con discapacidades. El Comprobador del contraste de esta asociación permite asegurar si el contraste entre el texto y el fondo de una interfaz cumple con los estándares de accesibilidad WCAG [22]. Las Ilustraciones 7 y 8 muestran los resultados de este análisis sobre los colores seleccionados en el proyecto.

3.4 Implementación de la Accesibilidad

En el desarrollo de la aplicación, se ha prestado especial atención a la accesibilidad, con el objetivo de garantizar que todos los usuarios, incluidos aquellos con discapacidades visuales, auditivas o motoras, puedan interactuar eficazmente con la aplicación. La accesibilidad no solo mejora la experiencia del usuario, sino que también asegura el cumplimiento de las mejores prácticas recomendadas por la WCAG y las pautas de accesibilidad de Android.

3.4.1 Ventana flotante

La ventana flotante es una funcionalidad clave implementada en la aplicación para mejorar tanto la accesibilidad como la experiencia del usuario, tal como se muestra en la Ilustración 9. Su diseño permite que el contenido de los tutoriales esté disponible de manera persistente y accesible, incluso mientras el usuario interactúa con otras aplicaciones. Este acceso continuo es particularmente útil para personas que requieren más tiempo para seguir las instrucciones, ya que pueden realizar múltiples tareas sin perder de vista el tutorial.

Ilustración 9 Ejemplo de ventana flotante en la aplicación

La configuración de la ventana flotante se realiza mediante el servicio *WindowManager* de Android, lo que posibilita que el contenido se muestre por encima de otras aplicaciones sin interrumpirlas. Esto es especialmente beneficioso para usuarios que necesitan seguir instrucciones en tiempo real mientras realizan otras tareas. La ventana se ajusta automáticamente para ocupar el 80% del ancho de la pantalla, asegurando que la información sea fácilmente visible y accesible sin bloquear completamente la vista del usuario.

Además, la *ventana flotante* es completamente movible, permitiendo al usuario ajustar su posición en cualquier parte de la pantalla según sus preferencias o necesidades motoras. Esta flexibilidad mejora la accesibilidad, ya que permite a los usuarios organizar su espacio de trabajo digital de una manera que facilite su interacción con la interfaz, especialmente útil para personas con discapacidades motoras.

Una característica adicional importante es la integración con el servicio *Text-to-Speech* (TTS), que permite a los usuarios escuchar el contenido del tutorial mientras interactúan con la ventana flotante. Cada vez que se carga un nuevo fragmento en la ventana flotante, el servicio TTS lee automáticamente el contenido en voz alta, proporcionando una experiencia auditiva accesible para aquellos que prefieren o necesitan escuchar las instrucciones.

Los botones de navegación dentro de la ventana flotante permiten a los usuarios moverse entre los fragmentos del tutorial sin cerrar la ventana, manteniendo la continuidad del contenido tanto visual como auditivamente. Esto asegura que los usuarios puedan controlar el flujo de información de manera rápida y sencilla, lo que facilita el seguimiento de las instrucciones sin interrumpir su flujo de trabajo. La actualización de los fragmentos y la lectura de TTS se gestiona mediante el método *updateFloatingWindow()*, lo que garantiza que el contenido siempre esté sincronizado entre lo visual y lo auditivo.

3.4.2 Tamaño de letra

Una parte importante de la accesibilidad en aplicaciones móviles es la capacidad de ajustar el tamaño del texto, especialmente para usuarios con dificultades visuales. Para mejorar la legibilidad, la aplicación ofrece tres opciones de tamaño de letra, permitiendo a los usuarios elegir el que mejor se adapte a sus necesidades. Esta función sigue las pautas de accesibilidad recomendadas por Android y permite un mayor control sobre la apariencia del contenido visual en la interfaz.

Los tres tamaños de letra implementados son:

- Tamaño pequeño: Ideal para usuarios sin problemas visuales o aquellos que prefieran más contenido visible en la pantalla.
- Tamaño mediano: Proporciona un equilibrio entre legibilidad y cantidad de contenido visible.
- Tamaño grande: Pensado para usuarios que prefieren una visualización más cómoda.

La configuración de tamaño de letra se almacena y se adapta dinámicamente a toda la interfaz de usuario, asegurando que todos los textos y elementos relevantes se ajusten según la preferencia del usuario. Este ajuste es especialmente útil para tutoriales largos o con múltiples pasos, donde la fatiga visual puede ser un problema si el tamaño de la fuente es demasiado pequeño.

Para implementar la funcionalidad de cambio de tamaño de letra en la aplicación, se ha desarrollado un método que muestra un cuadro de diálogo de selección, permitiendo al usuario escoger entre tres opciones de tamaño de fuente: pequeño, mediano y grande. El método utiliza un *AlertDialog* que presenta estas opciones en una lista desplegable. Para cada una de ellas, se define un valor de escala que determina cómo se verá el texto en la interfaz de la aplicación: *1.0f* para el tamaño pequeño, *1.25f* para el tamaño mediano, y *1.5f* para el tamaño grande. Además, cada opción muestra una vista previa dinámica del tamaño de letra, permitiendo al usuario visualizar en tiempo real cómo afectará su elección a la apariencia del texto en la aplicación.

Una vez que el usuario selecciona una de las opciones y confirma su elección, el tamaño de fuente elegido se almacena de forma persistente en las *SharedPreferences* utilizando una clase *PreferenceManager*, lo que garantiza que el ajuste se mantenga incluso después de cerrar y reabrir la aplicación. Posteriormente, la actividad se recarga mediante el método *recreate()*, aplicando el nuevo tamaño de letra de manera inmediata en toda la interfaz de usuario. Este proceso asegura que los cambios se reflejen de forma global, adaptando todos los textos y elementos interactivos al tamaño de letra seleccionado. Al abrir por primera vez la aplicación se muestra un mensaje de estilo *pop-up* para escoger el tamaño de letra deseado, Ilustración 10 derecha. A posteriori es posible modificar el tamaño de letra pulsando sobre el icono de la aplicación en el menú principal que tras seleccionar "*Cambiar Tamaño de Letra*" muestra el mismo dialogo pop-up que al abrir por primera vez la aplicación, Ilustración 10.

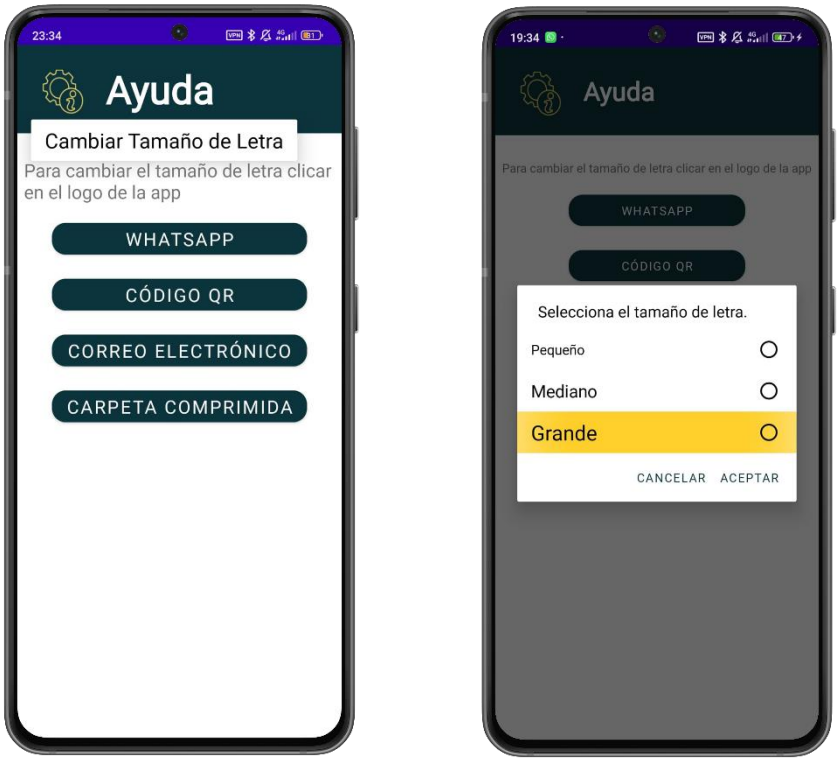


Ilustración 10 Cómo se muestra la opción de modificar de tamaño de letra

3.4.3 Lectura de textos

La capacidad de convertir el texto en audio, lo que resulta especialmente útil para usuarios con discapacidades visuales o aquellos que prefieren consumir información de manera auditiva. Para implementar esta funcionalidad, la aplicación incluye un servicio de *Text-to-Speech* (TTS) que permite leer en voz alta el contenido textual, como los pasos de los tutoriales o las instrucciones mostradas en la pantalla. Este servicio sigue las pautas de accesibilidad recomendadas por Android y asegura que la información esté disponible en un formato alternativo para quienes lo necesiten.

El servicio TTS se activa automáticamente cuando el usuario lo solicita, o bien cuando se abre una ventana flotante en la aplicación. La funcionalidad TTS se inicia utilizando el método *TextToSpeech.OnInitListener*, que se asegura de que el sistema esté correctamente inicializado antes de leer el texto en voz alta. En caso de que el TTS aún no esté listo para procesar la solicitud, el texto se almacena temporalmente para ser reproducido una vez que el servicio esté disponible.

Además, la aplicación ofrece soporte para distintos idiomas y voces, lo que permite al usuario seleccionar la voz y el idioma adecuados para su región o preferencia. Para garantizar que la experiencia de lectura sea coherente, el TTS está configurado por defecto para utilizar el idioma español (*es-ES*). Si este idioma no está disponible o no es soportado en el dispositivo, el sistema emplea una voz predeterminada que asegure la continuidad del servicio.

Una vez inicializado, el servicio TTS lee en voz alta cualquier texto que se le pase, con la opción de personalizar ciertas palabras o frases. Por ejemplo, términos como "*WhatsApp*" son adaptados para que suenen de forma más natural en el contexto del idioma. El texto se formatea adecuadamente para evitar problemas de pronunciación y se reproduce mediante el método *speak*, que utiliza la cola de reproducción para asegurar que no se pierda ninguna parte del contenido, incluso si se emiten varias solicitudes seguidas.

Para mejorar la experiencia de usuario, el TTS se integra con la ventana flotante, permitiendo a los usuarios escuchar las instrucciones mientras continúan interactuando con otras partes de la aplicación o del dispositivo. Esta ventana puede moverse libremente por la pantalla y permanece visible mientras se reproduce el contenido de voz, proporcionando una experiencia multitarea eficiente y accesible.

El servicio TTS se inicializa automáticamente cuando se abre la ventana flotante. Si el TTS ya está listo y configurado, el texto del fragmento que se está mostrando en ese momento se lee en voz alta de inmediato. Esto se gestiona mediante el método *speakOut*, que toma el texto del fragmento y lo convierte en audio. Si el servicio TTS no está completamente inicializado cuando se muestra el fragmento, el texto se almacena temporalmente en una variable llamada *pendingTextToRead*. Una vez que el servicio TTS esté listo para funcionar, el texto pendiente se reproduce automáticamente, garantizando que el contenido se lea en cuanto sea posible.

Cada vez que el usuario navega entre fragmentos (por ejemplo, avanzando o retrocediendo en un tutorial), el contenido de la ventana flotante se actualiza. Esta actualización es gestionada por el método *updateFloatingWindow*(), que carga el nuevo fragmento y su contenido en la ventana flotante. Después de la actualización, el texto del nuevo fragmento también se pasa al servicio TTS para ser leído en voz alta, asegurando que los usuarios siempre reciban la información más reciente de manera auditiva. Este proceso se repite cada vez que se navega entre fragmentos, lo que proporciona una experiencia continua.

3.4.4 Compatibilidad con lectores de pantalla

Para asegurar la accesibilidad total de la aplicación, se ha implementado compatibilidad completa con lectores de pantalla como *TalkBack*, el servicio nativo de Android. Esta funcionalidad es esencial para usuarios con discapacidades visuales, ya que permite la navegación e interacción con la aplicación mediante descripciones auditivas del contenido y los elementos interactivos.

Todos los elementos de la interfaz, incluyendo botones, menús y contenido textual, cuentan con etiquetas descriptivas (*contentDescription*), que son reconocidas y leídas por los lectores de pantalla. Esto permite a los usuarios recibir información sobre la funcionalidad de cada elemento y navegar por la aplicación de manera fluida sin necesidad de visualizar el contenido directamente.

Además, la integración con TTS complementa la experiencia del lector de pantalla, permitiendo que el contenido de los tutoriales se lea en voz alta de manera automatizada o bajo demanda. La combinación de estas dos tecnologías asegura que la aplicación sea totalmente accesible para personas con discapacidades visuales, ofreciendo una experiencia de usuario inclusiva y completa.

3.5 Pruebas y Evaluación

Una vez terminada la aplicación, se ha procedido a la realización de diferentes pruebas de uso, para lo cual se han empleado diferentes enfoques de prueba que garanticen la efectividad de la aplicación en un amplio rango de contextos. Los resultados de estas pruebas también han sido tenidos en cuenta.

3.5.1 Simulaciones de accesibilidad

Una de las principales prioridades del desarrollo de la aplicación fue garantizar que fuera accesible para personas con discapacidades visuales o motoras. Para verificar que los elementos de la interfaz fueran interpretados correctamente por lectores de pantalla, se realizaron simulaciones utilizando *TalkBack* (para Android), tal como se muestra en la Ilustración 11. Estas simulaciones permitieron evaluar cómo los usuarios recibirían la retroalimentación auditiva y asegurarse de que las etiquetas y descripciones en los botones y menús fueran claras y precisas.

Ilustración 11 Uso de TalkBack en la aplicación

Asimismo, se probaron escenarios de uso con el sistema de TTS para garantizar que las indicaciones en voz fueran correctas.

3.5.2 Pruebas de usabilidad

Se realizaron pruebas de usabilidad para garantizar que los usuarios pudieran interactuar con la aplicación de manera intuitiva. Estas pruebas se enfocaron en evaluar la facilidad de navegación, la comprensión de los menús y la capacidad de completar tareas comunes dentro de la aplicación, como acceder a los tutoriales o cambiar el tamaño de la fuente.

Además, se llevaron a cabo pruebas con usuarios que tenían poca experiencia en el manejo de tecnologías móviles, mediante el uso de diferentes tutoriales, como se muestra en la Ilustración 12. Estas pruebas revelaron que el diseño simplificado de la interfaz facilitaba la navegación, incluso para usuarios con dificultades tecnológicas. También se observó que la función de ventana flotante, que permite seguir las instrucciones sin cambiar de aplicación, fue particularmente bien recibida.

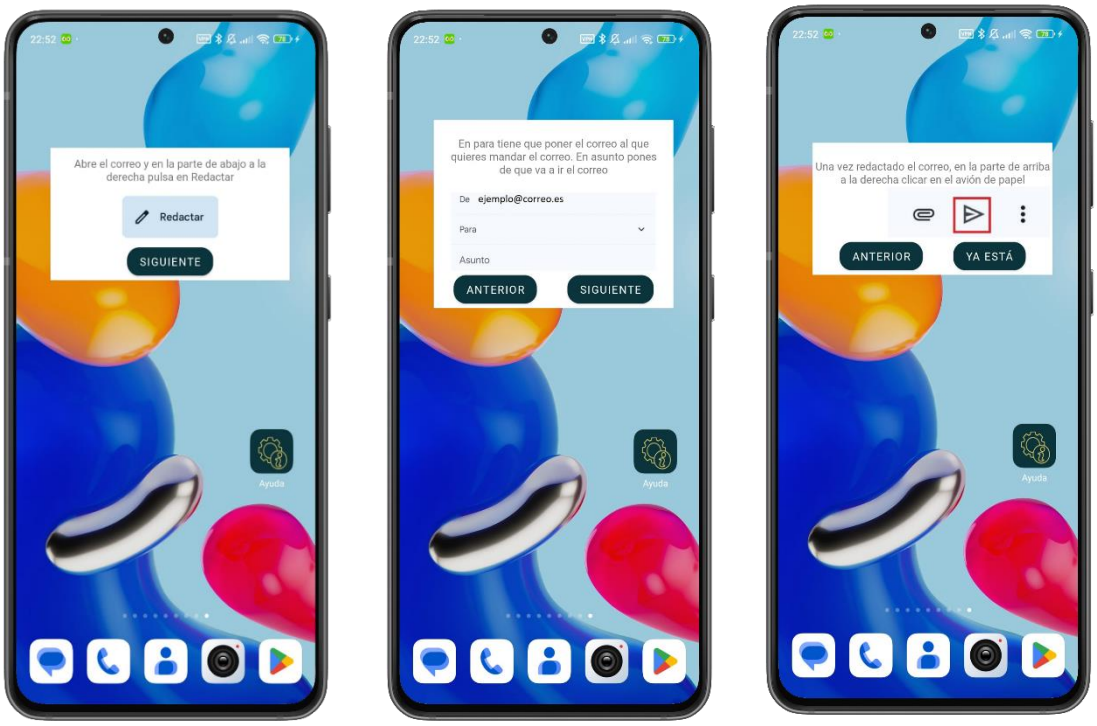


Ilustración 12 Ejemplo de tutorial

3.5.3 Análisis automatizados

Para validar la funcionalidad de los componentes clave de la aplicación y garantizar su correcto desempeño, se llevaron a cabo pruebas automatizadas utilizando herramientas como *Espresso* y *JUnit*. Estas pruebas simularon diversos escenarios de uso, verificando la estabilidad de la aplicación en situaciones como múltiples aperturas y cierres rápidos, cambios de orientación del dispositivo, y alternancia entre diferentes funciones. Además, se evaluó cómo se comportaba la aplicación bajo condiciones de estrés, asegurando que mantuviera su rendimiento sin fallos ni cierres inesperados. Este enfoque permitió detectar problemas potenciales antes de su implementación en entornos de producción, mejorando la confiabilidad del sistema.

Adicionalmente, se realizaron pruebas de carga para medir el tiempo de respuesta de la aplicación, centrándose en su capacidad de manejar tareas simultáneas y altas demandas de procesamiento. Con la herramienta *Perfetto Trace Processor,* se analizaron los recursos del sistema, como *CPU*, memoria e hilos, con el fin de identificar posibles cuellos de botella y asegurar un rendimiento óptimo.[23]

3.5.4 Evaluación de los tutoriales

El núcleo de esta aplicación son los tutoriales interactivos, diseñados para guiar a los usuarios paso a paso en tareas comunes como enviar correos, adjuntar archivos, abrir carpetas *ZIP*, usar el lector de códigos *QR* en dispositivos *Xiaomi*, y enviar ubicaciones o audios por *WhatsApp*. Estos tutoriales aparecen en ventanas flotantes durante momentos clave de la interacción, permitiendo que los usuarios aprendan sin

interrumpir su flujo de trabajo. Cada tutorial es claro y sencillo, con instrucciones visuales y textuales detalladas que facilitan el aprendizaje.

3.5.4.1 Enviar ubicación y audios por WhatsApp

WhatsApp es una de las aplicaciones de mensajería más utilizadas, y los tutoriales cubren funciones clave como enviar la ubicación en tiempo real y enviar mensajes de audio[24]. El tutorial para enviar la ubicación explica cómo acceder a la función de compartir ubicación en la aplicación, y cómo elegir entre compartir una ubicación fija o en tiempo real. En la Ilustración 13, se muestra un ejemplo de este tutorial, donde se guía al usuario para encontrar la opción dentro de *WhatsApp*. Otro tutorial similar muestra cómo grabar y enviar mensajes de audio, algo útil para usuarios con dificultades para escribir mensajes largos.

Ilustración 13 Tutorial de como enviar la ubicación por WhastApp

3.5.4.2 Uso del lector de códigos QR en dispositivos Xiaomi

Este tutorial está diseñado para guiar a los usuarios de dispositivos *Xiaomi* en cómo encontrar y utilizar el lector de códigos QR integrado. A partir de la versión *MIUI 12* y en *HyperOS*, el lector de códigos *QR* puede aparecer de dos formas: en el panel de notificaciones o en el centro de control, dependiendo de la configuración del sistema. Si el dispositivo del usuario tiene esta opción, se mostrarán ambos tutoriales para cubrir cada escenario.

El tutorial guía al usuario en función de cómo se muestre el lector en su dispositivo. Para aquellos usuarios que tienen la posibilidad de ver ambas opciones, se incluyen los dos tutoriales, uno para encontrar el lector de códigos *QR* en el centro de control y otro para acceder a él desde las notificaciones, tal y como se muestra en la Ilustración 14.

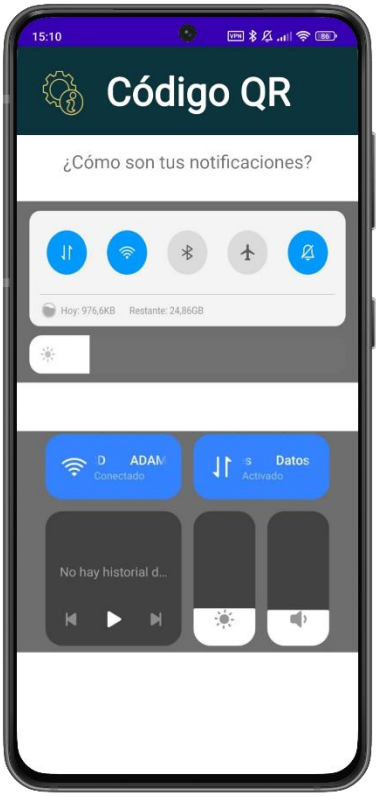


Ilustración 14 Ejemplo de los 2 tipos de notificaciones

En la Ilustración 15, se muestra cómo acceder al lector de códigos *QR* desde el panel de notificaciones, detallando todos los pasos. Dependiendo de la configuración, el botón puede estar visible de inmediato o puede ser necesario añadirlo manualmente a la vista general del panel de notificaciones. De igual manera, en la Ilustración 16, se detallan todos los pasos para acceder al lector desde el centro de control.

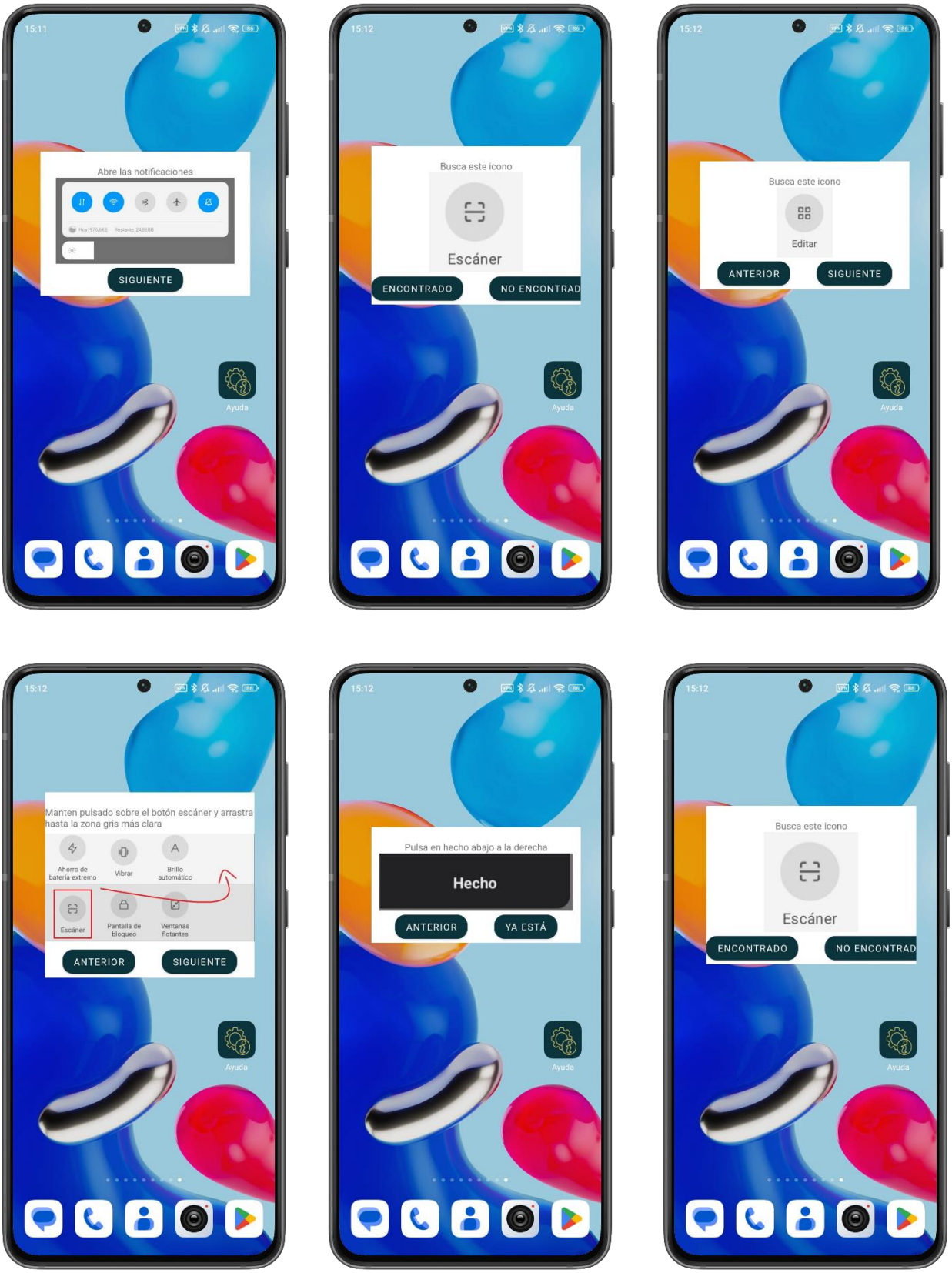


Ilustración 15 Tutorial de cómo encontrar el lector de códigos QR en el panel de notificaciones

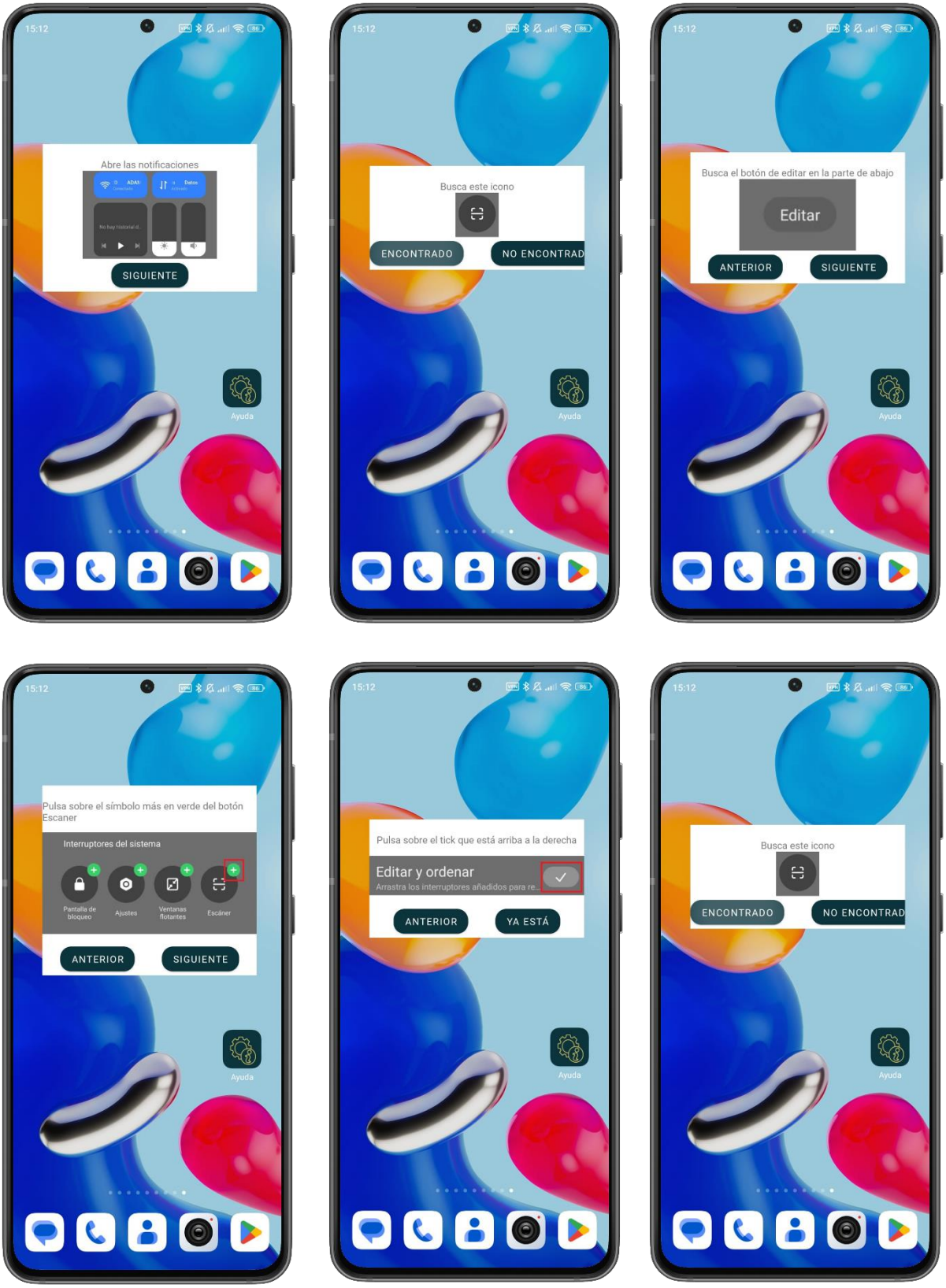


Ilustración 16 Tutorial de cómo encontrar el lector de códigos QR en el centro de control

Este enfoque asegura que los usuarios, sin importar la configuración de su dispositivo, puedan utilizar el lector de códigos *QR* de manera eficiente, siguiendo el tutorial adecuado según su versión de *MIUI* o *HyperOS*.

3.5.4.3 Envío de correos electrónicos y adjuntar archivos

Existen dos tutoriales principales relacionados con el envío de correos electrónicos: uno que guía a los usuarios en cómo redactar y enviar un correo, y otro que explica el proceso de adjuntar archivos. Estos tutoriales han sido diseñados para aplicarse a las aplicaciones de correo más utilizadas, como *Gmail*, *Outlook* y *Yahoo*. Se ha razonado ejemplificar estos procesos en estas plataformas, ya que, excluyendo *Apple Mail*, son los principales proveedores de correo electrónico a nivel global, con una amplia cuota de mercado[25].

El primer tutorial guía a los usuarios paso a paso en la redacción de un mensaje, desde la selección de destinatarios hasta el envío final del correo. El segundo tutorial enseña cómo adjuntar documentos o imágenes desde el almacenamiento del dispositivo. En la Ilustración 17, se muestra un ejemplo donde se explica cómo seleccionar adjuntar archivos a un correo en *Outlook*, un proceso que se adapta de manera similar a *Gmail* y *Yahoo*. Estos tutoriales son fundamentales para aquellos usuarios que necesitan enviar documentos, pero no están familiarizados con las opciones de adjuntar archivos en las diferentes plataformas de correo electrónico.

Ilustración 17 Tutorial de cómo mandar un correo con archivos adjuntos desde Outlook

3.5.4.4 Apertura de carpetas ZIP

Otro tutorial clave es el que muestra cómo abrir carpetas *ZIP*, una tarea que puede ser complicada para usuarios con menos experiencia tecnológica. El tutorial enseña a descomprimir archivos utilizando únicamente las herramientas integradas en el dispositivo, ya que todos los dispositivos Android vienen con esta funcionalidad por defecto. En las Ilustraciones 18 y 19, se muestra cómo se guía a los usuarios a través de los pasos para extraer archivos de una carpeta *ZIP* y acceder a su contenido de manera sencilla, sin necesidad de descargar aplicaciones de terceros.

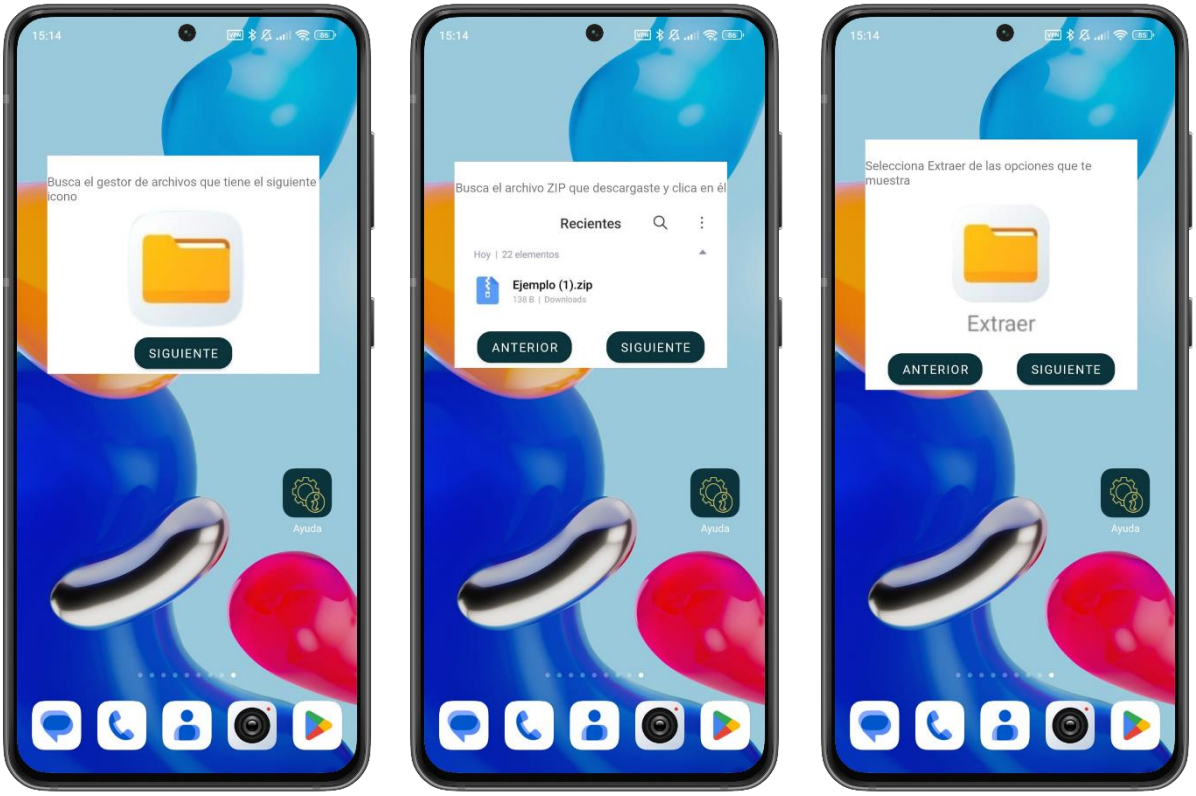


Ilustración 18 Parte 1 del tutorial de cómo descomprimir un archivo ZIP

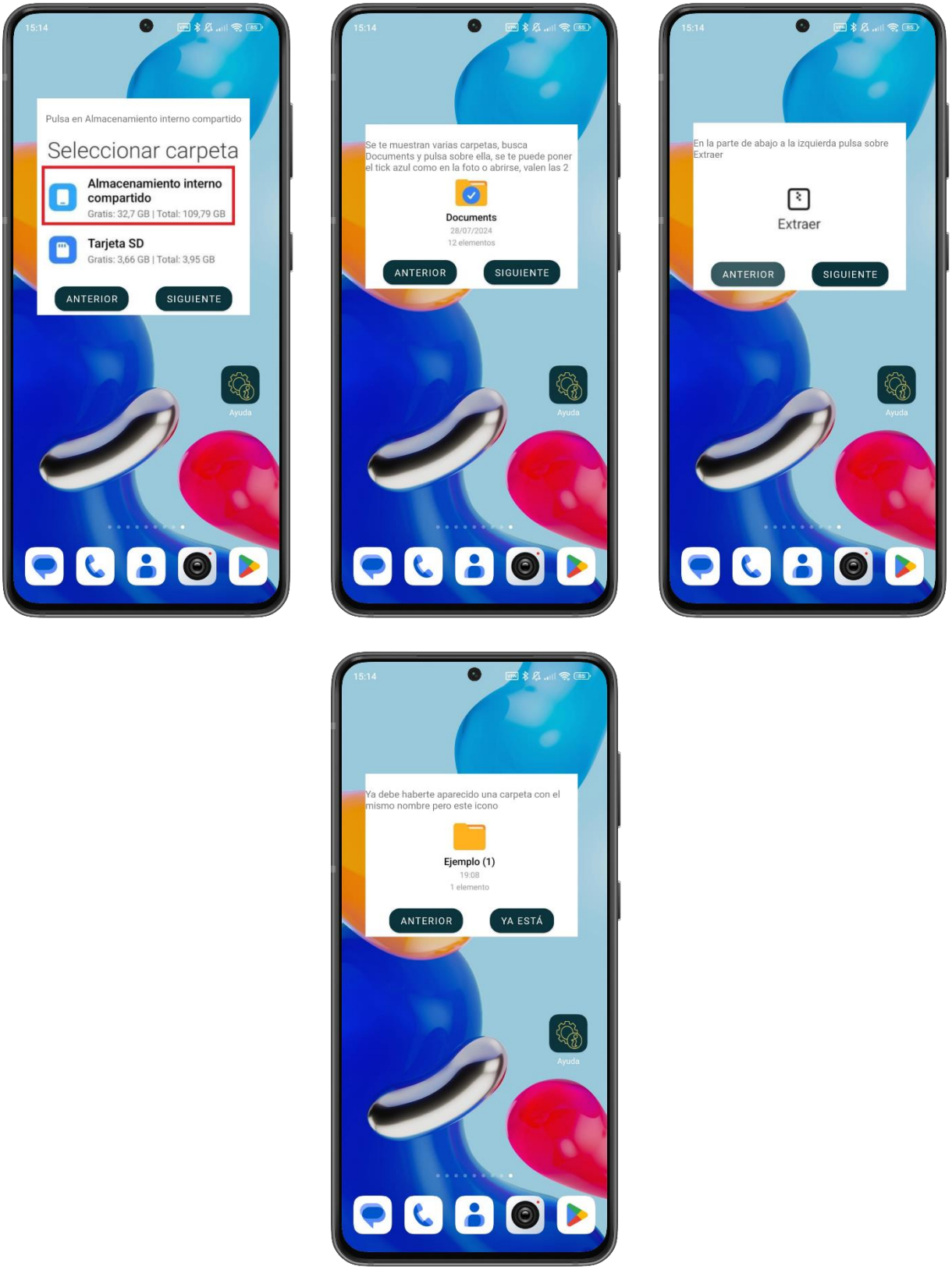


Ilustración 19 Parte 2 del tutorial de cómo descomprimir un archivo ZIP

3.6 RESULTADOS DE LAS PRUEBAS

Las pruebas realizadas sobre la aplicación se enfocaron en tres aspectos clave: accesibilidad, usabilidad, y rendimiento. Los resultados de estas pruebas confirmaron que la aplicación cumplió con los objetivos planteados en cada uno de estos ámbitos. A continuación, se presenta un análisis detallado de los hallazgos.

3.6.1 Accesibilidad

La accesibilidad fue un aspecto fundamental durante el desarrollo de la aplicación, especialmente para usuarios con discapacidades visuales. Las simulaciones realizadas con herramientas de accesibilidad y lectores de pantalla indicaron que los elementos de la interfaz de usuario estaban debidamente etiquetados, lo que permitió una correcta interpretación por parte de los sistemas de asistencia.

3.6.1.1 Lectores de pantalla

Los lectores de pantalla (*TalkBack* en Android) fueron capaces de identificar correctamente las etiquetas y descripciones de botones, campos de texto, y otros elementos interactivos, lo que facilitó la navegación para personas con discapacidad visual. Cada botón y componente tenía una etiqueta descriptiva adecuada que ayudaba a guiar al usuario.

3.6.1.2 Sistema de Texto a Voz (TTS)

El sistema *TTS* proporcionó una experiencia fluida, leyendo en voz alta los elementos de la interfaz sin pausas indebidas o repeticiones. Las pruebas confirmaron que los mensajes eran claros y comprensibles, proporcionando una navegación intuitiva para los usuarios que dependían de esta funcionalidad.

3.6.1.3 Contrastes de colores

La aplicación utilizaba combinaciones de colores con suficiente contraste, asegurando la legibilidad de los textos y la visibilidad de los elementos interactivos, incluso para usuarios con baja visión o daltonismo.

3.6.2 Usabilidad

El diseño de la interfaz de usuario de la aplicación fue simplificado para que los usuarios con poca experiencia tecnológica pudieran completar las tareas sin dificultades. Las pruebas de usabilidad, tanto con usuarios novatos como experimentados, reflejaron una buena experiencia general.

3.6.2.1 Simplicidad en la navegación

Las tareas más comunes, como enviar correos electrónicos o escanear códigos *QR*, se completaron de manera rápida y sencilla. Los usuarios señalaron que los botones y opciones estaban organizados de manera intuitiva, y no hubo confusión en cuanto a las funciones de cada sección.

3.6.2.2 Tiempos de respuesta

Los usuarios comentaron que las respuestas de la aplicación a sus interacciones fueron rápidas, lo que contribuyó a una experiencia de uso fluida. No se registraron retrasos perceptibles al realizar acciones como abrir la bandeja de entrada o cambiar entre secciones.

3.6.2.3 Tareas completadas con éxito

Todos los participantes pudieron completar las tareas asignadas sin requerir asistencia adicional. Incluso los usuarios con escasa familiaridad con dispositivos móviles lograron navegar por la aplicación tras una breve familiarización.

3.6.3 Rendimiento

Además de los aspectos de accesibilidad y usabilidad, el rendimiento fue otro foco clave de las pruebas. La aplicación fue sometida a simulaciones y pruebas automatizadas para evaluar su capacidad de manejar distintos escenarios de uso y carga de trabajo, incluso en dispositivos con recursos limitados.

3.6.3.1 Uso de la CPU

Se observó que la aplicación gestionaba eficientemente los recursos del dispositivo, con tiempos de ejecución de hilos cortos y estados de inactividad apropiados. Por ejemplo, el hilo *Studio:Socket* de la aplicación, encargado de las comunicaciones de red, tuvo una duración promedio de ejecución de *273 microsegundos*, lo que significa que las operaciones de red eran rápidas y no sobrecargaban el procesador.

3.6.3.2 Estados de hilos

Los hilos críticos de la aplicación entraban en estado de inactividad (*sleep*) tras completar sus tareas, optimizando así el uso de la CPU y ahorrando recursos. Esto fue especialmente útil para garantizar el buen rendimiento en dispositivos de gama baja.

3.6.3.3 Carga concurrente

La aplicación fue capaz de manejar múltiples tareas e hilos en paralelo sin comprometer la estabilidad. Incluso bajo escenarios de alta carga (como la ejecución de tareas de red mientras se escanea un código *QR*), no se detectaron ralentizaciones ni caídas del sistema.

3.6.3.4 Eficiencia energética

Gracias a la optimización en la gestión de los hilos y el bajo uso de la CPU en promedio, la aplicación mostró un consumo moderado de batería, lo cual es crucial para dispositivos móviles con recursos limitados.

3.6.4 Evaluación de los tutoriales

Los tutoriales, diseñados para guiar a los usuarios en tareas como el envío de correos, descompresión de archivos *ZIP*, uso del lector de códigos *QR* en *Xiaomi*, y envío de

ubicaciones o audios por *WhatsApp*, fueron probados internamente por el desarrollador, su tutor y miembros de la familia.

Las pruebas confirmaron que los tutoriales se activan, por parte de los usuarios, en los momentos adecuados, proporcionando instrucciones claras y fáciles de seguir. Además, se verificó que las ventanas flotantes no interrumpen el flujo de trabajo. Las pruebas en dispositivos con diferentes versiones de *MIUI* y *HyperOS* mostraron que los tutoriales cubren correctamente las variaciones de configuración.

Aunque las pruebas se realizaron en un grupo reducido, los resultados indicaron que los tutoriales mejoran la usabilidad y facilitan la navegación de la aplicación.

4 CONCLUSIONES, APLICACIONES Y LÍNEAS FUTURAS

A lo largo de este trabajo se ha desarrollado una aplicación móvil orientada a mejorar la accesibilidad y el uso de nuevas tecnologías para personas que se enfrentan a barreras tecnológicas. El desarrollo se ha basado en un enfoque inclusivo, considerando tanto a usuarios con discapacidades, como a aquellos con poca experiencia tecnológica. Esta sección recoge los principales hallazgos del trabajo, las aplicaciones prácticas del desarrollo y las posibles líneas de investigación y mejoras futuras.

4.1 RESUMEN DE HALLAZGOS

A lo largo del desarrollo de la aplicación, se han identificado varios hallazgos clave relacionados con la accesibilidad, usabilidad y rendimiento del sistema. Estos hallazgos reflejan los avances logrados en la superación de barreras tecnológicas y la implementación efectiva de soluciones orientadas a mejorar la experiencia del usuario, en particular para personas con discapacidades o con poca experiencia tecnológica. A continuación, se presenta un resumen de los principales descubrimientos y su impacto en el desarrollo de la aplicación.

- **Síntesis**

  El desarrollo de la aplicación ha permitido abordar de manera efectiva las barreras tecnológicas que limitan el acceso a las nuevas tecnologías para diferentes tipos de usuarios. Se ha conseguido implementar una solución

accesible que, mediante un diseño intuitivo y herramientas de asistencia como TTS, mejora significativamente la experiencia de usuario, ofreciendo una mayor autonomía a aquellos que encuentran dificultades en la interacción con dispositivos móviles.

- **Impacto de las barreras tecnológicas**

  El análisis de las barreras tecnológicas mostró que las limitaciones más comunes están relacionadas con la compatibilidad de software, la necesidad de hardware moderno y las barreras de accesibilidad. Durante el desarrollo, se superaron estas barreras mediante la implementación de tecnologías como el soporte para versiones anteriores de Android, el uso de librerías adaptativas para mejorar el rendimiento y la inclusión de funciones específicas para personas con discapacidades visuales o motoras.

- **Efectividad de las Medidas de Accesibilidad**

  Las medidas de accesibilidad implementadas, como la compatibilidad con lectores de pantalla, la capacidad de ajustar el tamaño de la fuente y la incorporación del TTS, demostraron ser efectivas durante las pruebas. Estas herramientas facilitaron la navegación y comprensión del contenido de la aplicación para usuarios con necesidades especiales, garantizando así un uso accesible e inclusivo.

4.2 CONCLUSIONES

El desarrollo de esta aplicación móvil ha logrado cumplir con los objetivos propuestos de mejorar la accesibilidad y la usabilidad de las nuevas tecnologías, especialmente para personas que enfrentan barreras tecnológicas o que tienen poca experiencia en el uso de dispositivos móviles. A lo largo del proceso, se ha conseguido implementar una solución accesible y eficiente, compatible con el 96% de los dispositivos Android, lo que garantiza su disponibilidad para una amplia base de usuarios.

La inclusión de funcionalidades como Text-to-Speech, el ajuste de tamaños de texto y la compatibilidad con lectores de pantalla ha demostrado ser efectiva en mejorar la experiencia de usuarios con discapacidades visuales o motoras. Asimismo, el enfoque en la simplicidad de la interfaz ha permitido que personas con poca experiencia tecnológica puedan navegar la aplicación de manera intuitiva y eficiente.

El análisis de las barreras tecnológicas, tanto a nivel de hardware como de software, permitió anticiparse a las dificultades que enfrentan los usuarios y desarrollar una solución que minimiza estos obstáculos. El éxito de este enfoque se evidenció en los resultados de las pruebas de usabilidad, donde se destacó la facilidad de uso y la accesibilidad de la aplicación.

Como línea futura, la expansión de la aplicación a *iOS* y la incorporación de soporte multilingüe serán pasos clave para ampliar su alcance global y ofrecer una solución aún más inclusiva. Además, se continuará optimizando la personalización de la interfaz para adaptarse mejor a las necesidades y preferencias de los usuarios.

En conclusión, la aplicación ha logrado reducir de manera significativa las barreras tecnológicas, ofreciendo una herramienta accesible y fácil de usar, con el potencial de seguir mejorando y evolucionando hacia una mayor inclusión digital en futuras versiones.

4.3 LÍNEAS FUTURAS

En cuanto a las líneas futuras de desarrollo, se recomienda avanzar con la publicación de la aplicación en Google Play Store, lo que permitirá una distribución más amplia y un acceso global a usuarios interesados. Este proceso no solo aumentará la visibilidad de la aplicación, sino que también proporcionará una plataforma donde los usuarios puedan proporcionar comentarios y sugerencias, facilitando así la mejora continua de la aplicación.

Pasos para la puesta en Google Play Store:

1. **Cumplimiento de los requisitos de Google Play**

   Adaptar la aplicación a las políticas de privacidad y seguridad de la plataforma, asegurando el cumplimiento de las regulaciones de Google, como la correcta gestión de permisos y el tratamiento de datos personales.

2. **Optimización del rendimiento**

   Realizar una última ronda de pruebas para garantizar que la aplicación funcione de manera fluida en una amplia gama de dispositivos, en especial aquellos con especificaciones más bajas.

3. **Creación de la ficha de la aplicación**

   Desarrollar una página en Google Play con descripciones detalladas, capturas de pantalla y vídeos explicativos sobre las principales funciones y beneficios de la aplicación.

4. **Despliegue inicial controlado**

   Implementar un lanzamiento progresivo, comenzando con una versión beta a través de *Google Play Console*, permitiendo identificar posibles errores antes de su acceso público general.

Además, sería recomendable realizar pruebas adicionales con un grupo más diverso de usuarios, incluyendo a personas con diferentes tipos de discapacidades y con niveles variados de competencia tecnológica. Estas pruebas contribuirían a identificar áreas de

mejora no detectadas previamente y a validar la efectividad de las soluciones implementadas.

Finalmente, explorar la integración de nuevas tecnologías, como inteligencia artificial, podría personalizar aún más la experiencia del usuario, adaptando dinámicamente la interfaz según sus necesidades particulares. Esto permitiría una interacción más fluida y ajustada a cada perfil de usuario.

4.3.1 Mejoras en la Aplicación

Aunque la aplicación ya es compatible con el 96% de los dispositivos Android, todavía hay margen para ampliar su accesibilidad y mejorar la experiencia del usuario. Algunas mejoras específicas que podrían implementarse en futuras versiones de la aplicación incluyen:

4.3.1.1 Soporte multilingüe

Expandir el sistema de TTS y la interfaz de la aplicación para ofrecer soporte en varios idiomas permitiría alcanzar una mayor cantidad de usuarios a nivel global. Incluir idiomas como inglés, francés, alemán o lenguas regionales beneficiaría a usuarios en diferentes partes del mundo. Esto podría incluir:

- **Integración de TTS multilingüe**

  Usar motores de TTS que soporten múltiples idiomas, asegurando que la pronunciación sea precisa en cada uno.

- **Traducción de la interfaz de usuario**

  Hay que asegurar que todos los textos, menús y mensajes de la aplicación estén traducidos y adaptados culturalmente para los usuarios de cada región, manteniendo una experiencia coherente en todos los idiomas.

4.3.1.2 Mejora de la interfaz de usuario

Para ofrecer una experiencia más personalizada y accesible, se podrían añadir opciones de personalización que permitan a los usuarios adaptar la interfaz según sus preferencias. Esto incluiría:

- **Opciones de color personalizables**

  Proporcionar esquemas de colores personalizables, que permitan a los usuarios elegir combinaciones de colores que mejoren la legibilidad, especialmente para personas con deficiencias visuales como el daltonismo.

- **Ajustes tipográficos avanzados**

  Ampliar las opciones de personalización de texto, permitiendo a los usuarios no solo ajustar el tamaño de la fuente, sino también elegir diferentes tipos de letra que se adapten mejor a sus necesidades visuales.

- **Reorganización de la interfaz**

  Implementar una interfaz más flexible que permita a los usuarios modificar la disposición de los elementos según sus necesidades o preferencias, facilitando la navegación para aquellos con menor habilidad tecnológica.

4.3.1.3 Expansión a plataformas iOS

Dado que la aplicación ya es compatible con la mayoría de los dispositivos Android, el siguiente paso lógico sería expandir su disponibilidad a usuarios de iOS. Esta expansión podría implicar:

- **Desarrollo para iOS**

  Adaptar la aplicación a la plataforma de Apple, reescribiendo partes del código para ajustarse a las pautas de diseño y accesibilidad de iOS. Esto aseguraría una experiencia similar para los usuarios de ambas plataformas.

- **Optimización para iOS**

  Garantizar que las funciones clave, como el TTS y la accesibilidad, funcionen de manera óptima en iOS, manteniendo la misma fluidez y nivel de personalización que en Android.

Con esta expansión, la aplicación tendría un alcance mucho mayor, permitiendo a los usuarios de ambas plataformas beneficiarse de sus características de accesibilidad y personalización.

5 BIBLIOGRAFÍA

[1] cas2017, «Principales Barreras y limitaciones del entorno digital», Conectados al Sur - Hablatam. Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://www.conectadosalsur.org/2017/10/13/principales-barreras-y-limitaciones-del-entorno-digital/

[2] «Qué es la brecha digital y cómo evitar que provoque desigualdad - Ahora». Accedido: 22 de julio de 2024. [En línea]. Disponible en: https://www2.cruzroja.es/web/ahora/brecha-digital

[3] «¿Cuáles son los desafíos de mantener la compatibilidad de hardware y software?» Accedido: 26 de julio de 2024. [En línea]. Disponible en: https://www.linkedin.com/advice/0/what-challenges-maintaining-hardware-software-knaic?lang=es&originalSubdomain=es

[4] «Los Desafíos De La Compatibilidad Entre Versiones De Software», FasterCapital. Accedido: 26 de julio de 2024. [En línea]. Disponible en: https://fastercapital.com/keyword/los-desafíos-de-la-compatibilidad-entre-versiones-de-software.html

[5] «Compatibilidad e integración de los sistemas —». Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://aceproject.org/ace-es/topics/et/etb/etb02/etb02e

[6] «5 barreras para los gobiernos digitales inclusivos». Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://www.caf.com/es/conocimiento/visiones/2022/03/5-barreras-para-los-gobiernos-digitales-inclusivos/

[7] «Descripción general de la seguridad de las apps», Apple Support. Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://support.apple.com/es-es/guide/security/sec35dd877d0/web

[8] «¿Qué es la accesibilidad? - Aprende desarrollo web | MDN». Accedido: 19 de julio de 2024. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Learn/Accessibility/What_is_accessibility

[9] «Smartphone replacement cycle worldwide 2013-2020», Statista. Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://www.statista.com/statistics/786876/replacement-cycle-length-of-smartphones-worldwide/

[10] J. McAllister, «How Long Do Smartphones Last? Understanding Phone Lifespan». Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://howlongdoesitlast.org/how-long-do-smartphones-last/

[11] E. Belinski, «iOS version usage — iOS Ref». Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://iosref.com/

[12] A. S. Navarro, «Cómo garantizar la inclusión y accesibilidad en apps», App Marketing News. Accedido: 19 de julio de 2024. [En línea]. Disponible en: https://appmarketingnews.io/como-garantizar-la-inclusion-y-accesibilidad-en-apps/

[13] «Infografía: El mapa mundial de Android e iOS», Statista Daily Data. Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://es.statista.com/grafico/29620/sistema-operativo-movil-con-la-mayor-cuota-de-mercado-por-pais

[14] «World Bank Open Data», World Bank Open Data. Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://data.worldbank.org

[15] «Velocidades de Internet por país 2024 - países-mundialmente.com», países-mundial.com. Accedido: 30 de julio de 2024. [En línea]. Disponible en: https://countries-worldwide.com/es/velocidades-de-internet-por-pais/

[16] «OWASP Top Ten | OWASP Foundation». Accedido: 31 de julio de 2024. [En línea]. Disponible en: https://owasp.org/www-project-top-ten/

[17] «CWE - Common Weakness Enumeration». Accedido: 31 de julio de 2024. [En línea]. Disponible en: https://cwe.mitre.org/

[18] «ISO 27001 - Seguridad de la información: norma ISO IEC 27001/27002», Normas ISO. Accedido: 31 de julio de 2024. [En línea]. Disponible en: https://www.normas-iso.com/iso-27001/

[19] «Modelo-Vista-Modelo de vista - .NET | Microsoft Learn». Accedido: 13 de septiembre de 2024. [En línea]. Disponible en: https://learn.microsoft.com/es-es/dotnet/architecture/maui/mvvm

[20] «Material Design», Material Design. Accedido: 13 de septiembre de 2024. [En línea]. Disponible en: https://m2.material.io/design/guidelines-overview

[21] F. Rodríguez, «¿Qué es y cómo usar Material Design?», Medium. Accedido: 13 de septiembre de 2024. [En línea]. Disponible en: https://medium.com/@franconrn/google-material-design-4b133625d33e

[22] «WebAIM: Contrast Checker». Accedido: 13 de septiembre de 2024. [En línea]. Disponible en: https://webaim.org/resources/contrastchecker/

[23] «Guía de rendimiento de la app | App quality», Android Developers. Accedido: 14 de septiembre de 2024. [En línea]. Disponible en: https://developer.android.com/topic/performance/overview?hl=es-419

[24] «Aplicaciones de mensajería: ranking según usuarios mensuales activos 2024», Statista. Accedido: 15 de septiembre de 2024. [En línea]. Disponible en: https://es.statista.com/estadisticas/599043/aplicaciones-de-mensajeria-mas-populares-a-nivel-mundial-de/

[25] «Los ocho proveedores de correo electrónico más populares | Mailchimp». Accedido: 15 de septiembre de 2024. [En línea]. Disponible en: https://mailchimp.com/es/resources/most-used-email-service-providers/