



UNIVERSIDAD NACIONAL DE CHIMBORAZO

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA

INFORMACIÓN

Aplicación móvil para la gestión de adopción de mascotas en la ciudad de

Riobamba mediante el uso del Framework Flutter

Trabajo de Titulación para optar al título de Ingeniero en Tecnologías de

la Información

Autor:

Naranjo Romero, Dayana Estefania

Tutor:

Ing. Ana E. Congacha, MSc.

Riobamba, Ecuador. 2024

DECLARATORIA DE AUTORÍA

Yo, Dayana Estefania Naranjo Romero, con cédula de ciudadanía 2101149363, autor (a) del trabajo de investigación titulado: Aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba mediante el uso del Framework Flutter, certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mi exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor (a) de la obra referida, será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, 10 de octubre de 2024.



Dayana Estefania Naranjo Romero
C.I: 2101149363

DICTAMEN FAVORABLE DEL PROFESOR TUTOR



Dirección
Académica
VICERRECTORADO ACADÉMICO



UNACH-RGF-01-04-08.11
VERSIÓN 01: 06-09-2021

ACTA FAVORABLE - INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN

En la Ciudad de Riobamba, a los 17 días del mes de septiembre de 2024, luego de haber revisado el Informe Final del Trabajo de Investigación presentado por el estudiante **DAYANA ESTEFANIA NARANJO ROMERO** con CC: **2101149363**, de la carrera **TECNOLOGÍAS DE LA INFORMACIÓN** y dando cumplimiento a los criterios metodológicos exigidos, se emite el **ACTA FAVORABLE DEL INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN** titulado **"APLICACIÓN MÓVIL PARA LA GESTIÓN DE ADOPCIÓN DE MASCOTAS EN LA CIUDAD DE RIOBAMBA MEDIANTE EL USO DEL FRAMEWORK FLUTTER"**, por lo tanto se autoriza la presentación del mismo para los trámites pertinentes.



Ing. Ana Elizabeth Congacha
TUTORA

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación Aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba mediante el uso del Framework Flutter, presentado por Dayana Estefanía Naranjo Romero, con cédula de identidad número 2101149363, bajo la tutoría de Mgs. Ana Elizabeth Congacha Aushay; certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba 10 de octubre de 2024.

Maria Isabel Uvidia, Mgs.
PRESIDENTE DEL TRIBUNAL DE GRADO



Fernando Molina, PhD.
MIEMBRO DEL TRIBUNAL DE GRADO



Milton López, Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO



CERTIFICADO ANTIPLAGIO



Dirección
Académica
VICERRECTORADO ACADÉMICO



UNACH-RGF-01-04-08.17
VERSIÓN 01: 06-09-2021

CERTIFICACIÓN

Que, **DAYANA ESTEFANIA NARANJO ROMERO** con CC: **2101149363**, estudiante de la Carrera de **INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN**, Facultad de **INGENIERÍA**; ha trabajado bajo mi tutoría el trabajo de investigación titulado "**APLICACIÓN MÓVIL PARA LA GESTIÓN DE ADOPCIÓN DE MASCOTAS EN LA CIUDAD DE RIOBAMBA MEDIANTE EL USO DEL FRAMEWORK FLUTTER**", cumple con el 3%, de acuerdo al reporte del sistema Anti plagio **TURNITIN**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 2 de octubre de 2024



Ing. Ana Elizabeth Congacha Aushay
TUTORA PROYECTO DE INVESTIGACIÓN

DEDICATORIA

A mi amado padre, **Luis Naranjo**, por ser mi ejemplo de perseverancia y sabiduría, por su constante aliento y apoyo incondicional a lo largo de mi vida y carrera académica, por sus regaños y consejos desde niña para que hoy llegue a cumplir todas mis metas.

A mi querida madre, **Rocio Romero**, cuya dedicación, amor infinito y sacrificios han sido la luz que guió cada paso de este camino, por enseñarme a ser una mujer fuerte y valiente, a nunca dejarme caer y ser constante.

A mis queridos hermanos y hermana, **Jefferson, Andrés y Andrea**, quienes estuvieron ahí para alentarme y compartir cada triunfo y desafío, por enseñarme la complicidad y las risas.

A mi amor, **Jordan Pilay**, quien ha sido más que mi pareja, mi mejor amigo incansable de mis berrinches y mi mayor motivación durante todo este tiempo, por creer siempre en mí y en lo nuestro incluso a través de la distancia que nos ha separado todos estos años. Aunque no siempre lo diga, eres mi inspiración y mi gran orgullo. Te amo.

A **One Direction**, mi boyband favorita, “los chicos normales pero terribles bailarines”, ustedes han estado ahí para mí desde *What Makes You Beautiful* hasta *History*, llenando mis días con *Little Things* que hicieron la diferencia en momentos que nunca olvidaré. Gracias por *Stole My Heart* y por ser *More Than This* para tantas directioners como yo. Aquí estaré, esperando cantar juntos *Walking in the wind* porque sé que “*It’s not the end, I’ll see your face again*”.

A mi compañero fiel, **Cocky**, que ha estado a mi lado en todas mis noches de vela cuando las ideas no fluían y por ser mi confidente silencioso en cada momento difícil. Por ti, decidí dedicar mi tesis a un tema que ayudará a que no haya más animales en las calles, con la esperanza de que otros puedan encontrar un compañero tan maravilloso como tú.

Este trabajo está dedicado a cada uno de ustedes, por ser mi motivación y mi fuerza en este camino hacia el conocimiento y la realización personal.

AGRADECIMIENTO

En primer lugar, agradezco a Dios por darme la fuerza, la sabiduría y la salud necesarias para llevar a cabo este proyecto.

A mis padres, Luis Naranjo y Rocio Romero quienes con su amor, apoyo incondicional y consejos sabios me han guiado en cada paso de mi vida académica y personal. Gracias por creer en mí y por ser mi pilar fundamental en todo momento.

Quiero agradecer también a mi novio, Jordan Pilay, quien sin su ayuda no hubiera culminado este trabajo, gracias por estar ahí para mí durante las largas noches y los códigos erróneos, por tu apoyo constante a pesar de la distancia y por ser mi gran sustento.

A mi querida tutora Ing. Anita Congacha, por su paciencia y conocimiento a lo largo de esta tesis, gracias por corregirme para bien y ayudarme a producir un excelente trabajo, por ser mi maestra desde tercer semestre y enseñarme a amar las bases de datos. Gracias por ser una verdadera maestra y profesional.

A los amigos que conocí durante mi carrera universitaria, quienes me brindaron su amistad, compañerismo y momentos inolvidables que alegraron mis días.

A todos mis respetados maestros, por su guía experta, paciencia y conocimiento compartido, que han enriquecido mi aprendizaje y mi visión del mundo.

Finalmente, agradezco a todos aquellos que de una u otra manera han contribuido a la realización de esta investigación. Sin su ayuda y apoyo, este logro no habría sido posible.

Con gratitud y aprecio,
Dayana.

ÍNDICE GENERAL

DECLARATORIA DE AUTORÍA

DICTAMEN FAVORABLE DEL PROFESOR TUTOR

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

CERTIFICADO ANTIPLAGIO

DEDICATORIA

AGRADECIMIENTO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

ABSTRACT

CAPÍTULO I. INTRODUCCIÓN.....	14
1.1 Planteamiento del problema	14
1.2 Justificación	15
1.3 Formulación del problema.....	15
1.4 Objetivos.....	15
CAPÍTULO II. MARCO TEÓRICO.....	16
2.1 Aplicación Móvil.....	16
2.2 Gestión de adopción de mascotas	16
2.3 Framework Flutter	17
2.3.1 Definición	17
2.3.2 Arquitectura de Flutter.....	17
2.3.3 Ventajas y desventajas de Flutter	19
2.3.4 Componentes de Flutter	20
2.4 Herramientas de Desarrollo	22
2.4.1 Visual Code	22
2.4.2 FlutterFlow	22
2.4.3 Google Firebase	23
2.4.4 Firebase Performance Monitoring.....	23
2.4.5 Firebase Test Lab.....	24
2.5 Modelos de calidad de software	24
2.5.1 Modelo FURPS	25
2.6 Metodología Kanban	25

CAPÍTULO III. METODOLOGIA	27
3.1 Tipo de Investigación	27
3.2 Diseño de Investigación.....	27
3.3 Población de estudio y tamaño de muestra.....	27
3.4 Técnicas de Recolección de Datos	27
3.5 Métodos de análisis y procesamiento de datos	27
3.6 Identificación de variables.....	28
3.6.1 Variable dependiente	28
3.6.2 Variable independiente	28
3.7 Operacionalización de variables	29
3.8 Metodología de Desarrollo	30
3.8.1 Análisis	30
3.8.2 Diseño del sistema	32
3.8.3 Implementación	36
3.8.4 Pruebas	38
CAPÍTULO IV. RESULTADOS Y DISCUSIÓN	41
4.1 Resultados.....	41
4.1.1 Evaluación del Tiempo de Respuesta.....	41
4.1.2 Evaluación de Velocidad de Procesamiento.....	41
4.1.3 Evaluación de Consumo de recursos	42
4.2 Discusión	42
CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES	44
5.1 Conclusiones.....	44
5.2 Recomendaciones	45
BIBLIOGRAFÍA	46

ÍNDICE DE TABLAS

Tabla 1. Ventajas y desventajas de Flutter	20
Tabla 2. Operacionalización de variables	29
Tabla 3. Diseño inicial Tablero Kanban	30
Tabla 4. Descripción de roles	31
Tabla 5. Requerimientos Funcionales.....	31
Tabla 6. Requerimientos no funcionales	32
Tabla 7. Tablero Kanban, fase Diseño.....	36
Tabla 8. Resultados de uso de recursos	42

ÍNDICE DE FIGURAS

Figura 1: Flutter multiplataforma	17
Figura 2: Arquitectura de Flutter.....	19
Figura 3: Frameworks usados globalmente por desarrolladores de software	19
Figura 4: Árbol de Widgets.....	21
Figura 5: Visual Studio Code para el desarrollo de Flutter	22
Figura 6: Interfaz de FlutterFlow	23
Figura 7: Pruebas con Firebase Test Lab	24
Figura 8: Secciones de la metodología Kanban	26
Figura 9: Metodologías ágiles más utilizadas en 2023	26
Figura 10: Diagrama de casos de uso.....	33
Figura 11: Arquitectura cliente-servidor.....	33
Figura 12: Esquema colección “Users”	34
Figura 13: Esquemas de colecciones de la base de datos NoSQL	34
Figura 14: Mapa de navegabilidad de Login y Registro de Usuarios	35
Figura 15: Mapa de navegabilidad de perfil y editar usuario.....	35
Figura 16: Mapa de navegabilidad del registro y lista de mascotas.....	36
Figura 17: Interfaces de Inicio de sesión.....	37
Figura 18: Base de datos implementada en Firebase	38
Figura 19: Codificación de la interfaz HomePage	38
Figura 20: Subir APK a Test Lab	39
Figura 21: Elección de dispositivos Firebase Test Lab.....	39
Figura 22: Pruebas del APK en Firebase Test Lab	40
Figura 23: Dashboard de Firebase Performance Monitoring de la aplicación móvil.....	40
Figura 24: Resultados tiempo de respuesta.....	41
Figura 25: Resultados Velocidad de procesamiento	41

RESUMEN

Esta investigación se centró en el desarrollo de una aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba-Ecuador, se utilizó el Framework Flutter y se exploró el potencial de esta tecnología para crear una plataforma digital que optimice los procedimientos de adopción. La metodología incluyó un análisis exhaustivo de Flutter y la implementación de las mejores prácticas de diseño y desarrollo específicas para este framework. Se emplearon FlutterFlow y Visual Studio Code junto con la metodología Kanban, para asegurar un proceso eficiente y estructurado. Las pruebas de rendimiento se realizaron utilizando el modelo de calidad FURPS con la ayuda de Firebase Performance Monitoring y Firebase Test Lab. Esta investigación reveló hallazgos importantes, donde se demostró que la aplicación presenta un tiempo de respuesta 1% más rápido al escalar de 1 a varios dispositivos a la vez. El procesamiento de la pantalla de inicio disminuyó en un 50% y no hubo fotogramas congelados, garantizando una interacción rápida y sin retrasos significativos para los usuarios. El consumo eficiente de recursos (CPU y memoria) garantizó un rendimiento óptimo de la aplicación, incluso en dispositivos con capacidades de hardware limitadas. Como cierre de este estudio se subraya la idoneidad de Flutter para proyectos de desarrollo móvil en el ámbito de la adopción de mascotas, ofreciendo un ejemplo concreto de su aplicación efectiva en este dominio. Este trabajo tuvo implicaciones significativas para futuras investigaciones y mejoras en la gestión de adopción de mascotas, destacando el potencial de Flutter en la innovación tecnológica y práctica en este campo.

Palabras claves: Aplicación móvil, Flutter, Desarrollo de software, Adopción de mascotas.

ABSTRACT

This research focused on developing a mobile application for managing pet adoptions in Riobamba, Ecuador. The Flutter Framework was used to explore the potential of this technology to create a digital platform that optimizes adoption procedures. The methodology included a thorough analysis of Flutter and the implementation of best design and development practices specific to this framework. FlutterFlow and Visual Studio Code were used alongside the Kanban methodology to ensure an efficient and structured process. Performance tests were conducted using the FURPS quality model with the help of Firebase Performance Monitoring and Firebase Test Lab. This research revealed important findings, demonstrating that the application shows a 1% faster response time when scaling from 1 to multiple devices. The processing of the home screen decreased by 50%, and there were no frozen frames, ensuring quick interactions without significant delays for users. Efficient resource consumption (CPU and memory) ensured optimal application performance, even on devices with limited hardware capabilities. This study concludes by highlighting the suitability of Flutter for mobile development projects in pet adoption, offering a concrete example of its practical application in this domain. This work has significant implications for future research and improvements in pet adoption management, highlighting Flutter's potential for technological and valuable innovation in this field.

Keywords: Mobile applications, Flutter, Software development, Pet adoption.



ANA ELIZABETH
MALDONADO LEON

Reviewed by:
Ms.C. Ana Maldonado León
ENGLISH PROFESSOR
C.I.0601975980

CAPÍTULO I. INTRODUCCIÓN

La creciente preocupación por el bienestar de las mascotas y la necesidad de mejorar los procesos de adopción de la fauna urbana en la ciudad de Riobamba han motivado la formulación de esta investigación. En este estudio, se explora el potencial favorable del uso de la tecnología Flutter en el desarrollo de una aplicación adaptable a diversos dispositivos móviles. Se busca así resolver la falta de una plataforma digital y mejorar la eficacia de los procedimientos existentes en este ámbito específico.

A nivel mundial, se estima que alrededor del 70% de los perros carecen de un hogar según un estudio de la Organización Mundial de la Salud [1]. En este contexto, resulta imperativo comprender los antecedentes del problema, tanto en el entorno sociocultural de la ciudad como en el ámbito tecnológico que rodea la gestión de adopciones.

La decisión de optar por Flutter se fundamenta en su capacidad para desarrollar aplicaciones multiplataforma con un diseño visualmente atractivo y en su potencial para mejorar la productividad en el proceso de desarrollo. Según los resultados obtenidos en la producción de la aplicación Google Classroom, se ha observado una reducción del 66% en el tamaño del código en comparación con otras tecnologías, lo que asegura una experiencia de usuario óptima [2].

Flutter, respaldado por Google, ofrece eficiencia y agilidad aprovechando su función de Hot Reload para un desarrollo rápido y preciso. Además, la aplicación se desarrollará con Dart como lenguaje de programación, el cual se enfoca en hacer los procesos de programación más cómodos y ágiles para desarrolladores [3]. Conjuntamente, la integración de Firebase brinda servicios esenciales como una base de datos en tiempo real, autenticación segura y notificaciones en tiempo real.

Esta investigación se presenta como un esfuerzo integral para abordar una problemática concreta en la gestión de adopciones de mascotas, fusionando la pasión por el bienestar animal con la capacidad tecnológica de Flutter para ofrecer una solución práctica y efectiva.

1.1 Planteamiento del problema

A pesar de la creciente conciencia social sobre la importancia de la adopción responsable de mascotas, persisten desafíos significativos en la gestión eficiente de este proceso. Según diario La Prensa la problemática de hallar animales abandonados en la ciudad de Riobamba ha generado un intenso debate en la sociedad [4].

La gestión de adopciones de mascotas en el país enfrenta desafíos significativos puesto que según un estudio realizado por la Universidad San Francisco de Quito hasta el 2018 existían 600 mil animales de compañía en las calles de la capital del Ecuador [5]. Aunque existen esfuerzos loables por parte de organizaciones y centros de adopción, la falta de una solución tecnológica integral ha generado una brecha en la gestión eficiente de este proceso.

Esta necesidad no satisfecha se evidencia en la carencia de una herramienta digital específica para Riobamba que simplifique y centralice la información relacionada con mascotas disponibles, adoptantes potenciales y procesos de adopción.

1.2 Justificación

Ante la falta de plataformas adecuadas para la gestión de adopción de mascotas en la ciudad de Riobamba, la investigación propone el desarrollo de una aplicación móvil multiplataforma utilizando Flutter. Esta aplicación abarcará diversas funcionalidades, entre las que se incluyen: el inicio de sesión de usuarios, el registro de animales rescatados, la visualización y búsqueda de mascotas disponibles para su adopción, la comunicación entre rescatistas y adoptantes a través de mensajes, así como el seguimiento de las mascotas adoptadas.

Esta investigación señalará la importancia de utilizar Flutter junto con Google Firebase en la aplicación “AcolitaPet”, puesto que el Framework Flutter incluye un conjunto de herramientas de desarrollo de software de interfaz de usuario de código abierto donde posibilita la creación de aplicaciones compiladas de manera nativa para dispositivos móviles, web y de escritorio a partir de un solo código base [6]. Además, la unión entre Flutter y Firebase potenciara la experiencia de “AcolitaPet” con un diseño interactivo y atractivo, asegurando una navegación sin obstáculos y una comunicación intuitiva

Se prevé que la aplicación esté disponible para todo el público, permitiendo a los usuarios acceder como individuos particulares o como representantes de organizaciones de protección animal. Sin embargo, para llevar a cabo el proceso de adopción o registro de un animal rescatado, será necesario que el usuario inicie sesión utilizando su correo electrónico y complete los formularios requeridos para el mencionado proceso de adopción o rescate.

1.3 Formulación del problema

¿El uso del Framework Flutter incidirá en el rendimiento de una aplicación móvil para la gestión de adopción de mascotas en Riobamba?

1.4 Objetivos

Objetivo General

Desarrollar una aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba mediante el uso del Framework Flutter.

Objetivos específicos

- Analizar la tecnología del Framework Flutter enfocado al desarrollo de aplicaciones móviles.
- Aplicar la tecnología Flutter en el desarrollo de la aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba.
- Evaluar el rendimiento de la aplicación móvil mediante el modelo de calidad FURPS.

CAPÍTULO II. MARCO TEÓRICO

2.1 Aplicación Móvil

Una aplicación móvil o abreviado usualmente como “App”, es un programa diseñado para operar en dispositivos como smartphones, tablets, entre otros. Estas apps suelen ofrecer servicios similares a los que se encuentran en las computadoras. Por lo general, son programas pequeños con funciones específicas. Inicialmente, este tipo de software fue introducido por Apple Inc [7].

Estas aplicaciones brindan al usuario la capacidad de realizar diversas tareas, simplificando las gestiones o actividades que necesitan realizar. Normalmente, están disponibles a través de plataformas de distribución específicas o directamente desde las empresas propietarias de los dispositivos móviles, como Android, iOS, BlackBerry OS y Windows Phone [8].

Las aplicaciones móviles han diversificado su utilidad, abarcando desde facilitar la comunicación hasta proveer entretenimiento y servicios especializados. En el contexto de la gestión de adopción de mascotas, estas aplicaciones desempeñan un papel crucial al proporcionar una plataforma centralizada donde los refugios y fundaciones pueden promover animales en busca de un hogar amoroso.

2.2 Gestión de adopción de mascotas

La gestión de adopción de mascotas es un proceso complejo que involucra a varios actores y tiene como objetivo final encontrar un hogar adecuado para cada animal. Se trata de un proceso que requiere organización, planificación y seguimiento para garantizar el bienestar de los animales y la satisfacción de los adoptantes [9].

En 2019, se promulgó la ordenanza municipal 004-2019 en la ciudad de Riobamba [10], la cual destaca la gestión de adopción de mascotas como un pilar fundamental dentro de las políticas de protección animal. Esta ordenanza enfatiza la importancia de garantizar procesos de adopción que aseguren el bienestar tanto de los animales como de las familias adoptantes, con el objetivo de reducir la población de animales en situación de calle y fomentar una convivencia armoniosa entre humanos y mascotas, ver Anexo 1.

Según se señala en [11], se realizó un análisis de tres aplicaciones de Latinoamérica dedicadas a la gestión de adopción de mascotas y sus funcionalidades, se observa que gracias a las más de 10 mil descargas registradas hasta el año 2019, estas aplicaciones han permitido a las fundaciones y refugios de animales alcanzar a una audiencia más amplia. A través de estas plataformas, pueden exhibir fotos y descripciones de las mascotas disponibles para adopción, lo que aumenta significativamente la visibilidad y las posibilidades de encontrarles un hogar adecuado.

2.3 Framework Flutter

2.3.1 Definición

Según el sitio oficial de la empresa, Flutter es un Framework de código abierto creado por Google para el desarrollo de aplicaciones móviles multiplataforma, el cual permite crear interfaces de usuario nativas para Android, iOS, Web y escritorio con una única base de código [12]. Este enfoque unificado simplifica el proceso de desarrollo y garantiza una consistencia visual en todas las plataformas de destino, lo que a su vez reduce la complejidad y los tiempos de desarrollo.

Con el uso del lenguaje de programación DART y su amplio conjunto de herramientas y widgets, Flutter se vuelve altamente adaptable para diseñar y construir aplicaciones móviles modernas y atractivas en iOS, Android y Web, como muestra la figura 1. La versatilidad de Flutter permite a los desarrolladores crear experiencias de usuario consistentes y fluidas en múltiples plataformas con un esfuerzo mínimo. Además, la amplia gama de widgets predefinidos y personalizables facilita la creación de interfaces de usuario ricas y dinámicas, lo que permite a los equipos de desarrollo centrarse en la innovación y la creatividad en lugar de en aspectos técnicos más tediosos [13].



Figura 1: Flutter multiplataforma
Fuente: [14]

2.3.2 Arquitectura de Flutter

Flutter se concibe como un sistema de capas extensible, donde cada capa consiste en bibliotecas independientes que se apoyan en la capa que está por debajo. Ninguna capa tiene privilegios especiales sobre la capa subyacente, y cada elemento del marco se estructura para ser flexible y permitir la sustitución [15], las capas de la arquitectura de Flutter en orden ascendente son las siguientes:

a. **Embedder (Incrustador)**

Un incrustador específico de cada plataforma sirve como punto de entrada, este se coordina con el sistema operativo subyacente para acceder a servicios esenciales como la

renderización de gráficos, la accesibilidad y la entrada del usuario, así como para gestionar el bucle de eventos del mensaje. El incrustador está desarrollado en un lenguaje adecuado para cada plataforma, como Java y C++ para Android, Objective-C/Objective-C++ para iOS y macOS, y C++ para Windows y Linux. A través del incrustador, el código Flutter puede integrarse en una aplicación existente como un módulo, o bien puede constituir el contenido completo de una nueva aplicación[15].

b. Flutter Engine (Motor Flutter)

En el núcleo de Flutter se encuentra el Motor Flutter, desarrollado principalmente en C++, que ofrece el soporte esencial para todas las aplicaciones creadas con Flutter. El motor de renderizado de Flutter está a cargo de convertir las escenas compuestas (composiciones de widgets y otros elementos de la interfaz de usuario) en píxeles en la pantalla del dispositivo. Proporciona la implementación de bajo nivel de la API principal de Flutter, que incluye gráficos, diseño de texto, E/S de archivos y red, soporte de accesibilidad, arquitectura de plugins, y un tiempo de ejecución y cadena de herramientas de Dart [7].

c. Flutter Framework (Marco de Flutter)

Por lo general, los desarrolladores interactúan con Flutter a través del marco de Flutter, que presenta un enfoque reactivo moderno desarrollado en Dart. Este marco incluye un amplio conjunto de bibliotecas esenciales, de diseño y de plataforma, organizadas en varias capas. Desde abajo hacia arriba, según la figura 2, las capas son las siguientes:

- Las clases fundamentales y los servicios básicos, como animación, pintura y gestos, proporcionan abstracciones comunes sobre la infraestructura subyacente.
- La capa de renderizado ofrece una abstracción para el manejo del diseño, permitiendo la construcción de un árbol de objetos renderizables. Esta capa permite manipular dinámicamente los objetos, con el árbol actualizando automáticamente el diseño para reflejar los cambios.
- La capa de widgets actúa como una abstracción de composición. Cada objeto renderizado en la capa Rendering tiene una clase correspondiente en la capa de widgets. Además, esta capa permite definir combinaciones de clases que pueden ser reutilizadas.
- Las bibliotecas Material y Cupertino ofrecen conjuntos completos de controles que emplean las primitivas de composición de la capa de widgets para implementar los lenguajes de diseño Material o iOS.

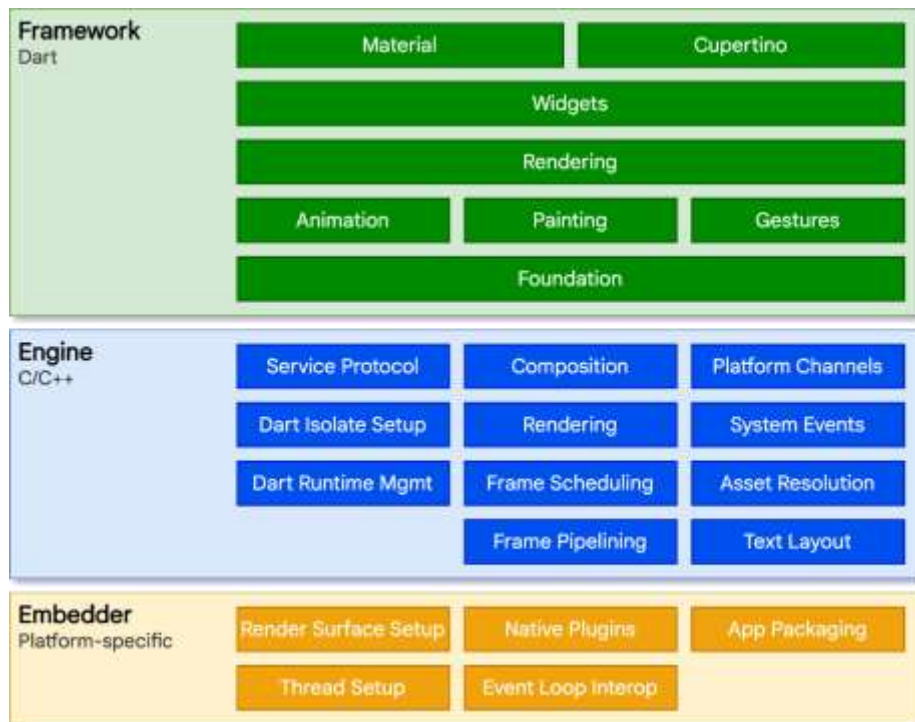


Figura 2: Arquitectura de Flutter
Fuente: [15]

2.3.3 Ventajas y desventajas de Flutter

En el dinámico mundo del desarrollo móvil, la búsqueda constante de herramientas eficientes y versátiles es una prioridad. En este sentido, el año 2022 marcó un hito significativo con la consolidación de Flutter como el principal framework de desarrollo móvil. Según el último informe de Statista, un portal alemán de estadísticas en línea, Flutter emergió como el indiscutible líder, demostrando su capacidad para impulsar la creación de aplicaciones móviles de alto rendimiento [16], como indica la figura 3.

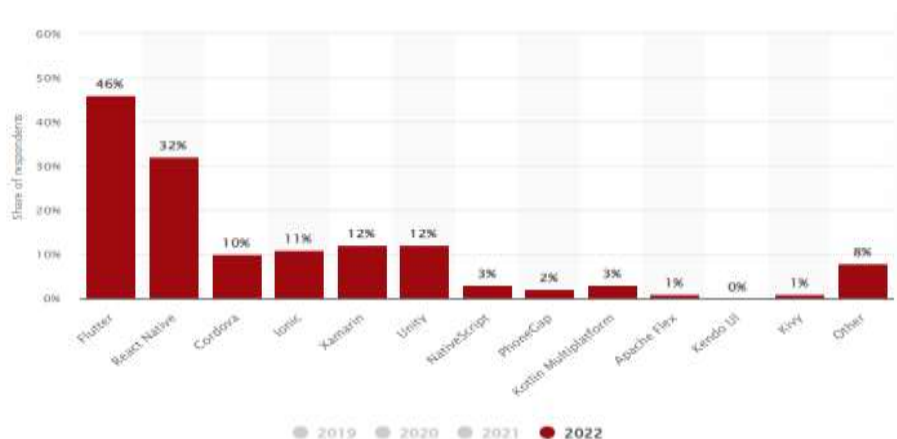


Figura 3: Frameworks usados globalmente por desarrolladores de software
Fuente: [16]

Para comprender mejor las razones detrás de esta preferencia, en la siguiente Tabla 1 se analiza detalladamente las ventajas y desventajas de Flutter en el panorama del desarrollo móvil.

Tabla 1. Ventajas y desventajas de Flutter

Ventajas	Desventajas
<p>Respuestas rápidas: Flutter facilita la actualización instantánea con recarga en caliente, lo que aumenta la eficiencia y posibilita correcciones al momento.</p>	<p>Aplicaciones extensas y pesadas: Pueden resultar en la creación de aplicaciones de considerable tamaño, lo que puede suponer un desafío al manejar archivos de gran envergadura.</p>
<p>Desarrollo multiplataforma: Ofrece crear aplicaciones compatibles con múltiples plataformas utilizando un único conjunto de código, lo que resulta en una notable reducción de tiempo y costos.</p>	<p>Limitaciones en bibliotecas y documentación: A pesar del crecimiento de la comunidad, puede existir una oferta restringida de bibliotecas y recursos disponibles.</p>
<p>Rendimiento nativo: Al utilizar Skia para la ejecución, garantiza un desarrollo eficiente y rápido, resultando en aplicaciones que se ejecutan de manera óptima y son prácticamente indistinguibles de las nativas.</p>	<p>Posibles desafíos con iOS: Las actualizaciones pueden demorar más en iOS en comparación con Android.</p>
<p>Ahorro de tiempo en el desarrollo: Menores exigencias y mayor rapidez en la creación de aplicaciones.</p>	<p>Escasa popularidad de Dart: A pesar de su facilidad de aprendizaje, Dart no ha logrado la misma popularidad que otros lenguajes como C#, Java, entre otros.</p>
<p>Aceleración en el lanzamiento al mercado: Flutter se destaca por su agilidad, ofreciendo un rápido tiempo de comercialización y compatibilidad multiplataforma.</p>	
<p>Motor de renderización exclusivo: Proporciona elementos visuales específicos para iOS y Android, garantizando una experiencia de usuario uniforme.</p>	
<p>Fácil aprendizaje: Dart, el lenguaje de programación de Flutter, es accesible para aprender, lo que brinda la oportunidad a desarrolladores novatos de crear prototipos y aplicaciones.</p>	
<p>“Hot-Reload”: La capacidad de efectuar modificaciones en tiempo real sin necesidad de reiniciar la aplicación, lo que agiliza el proceso de desarrollo.</p>	

Fuente: [17]

2.3.4 Componentes de Flutter

a. Widgets en Flutter

Dentro del framework de Flutter, los widgets representan las partes visuales de la interfaz de usuario de una aplicación. Estos componentes son esenciales para construir y personalizar la apariencia y la funcionalidad de la interfaz de usuario de manera flexible. Flutter ofrece una amplia gama de widgets integrados que facilitan la creación rápida de

interfaces de usuario, como botones, campos de texto, listas, menús desplegables, tablas e imágenes, entre otros. Además, brinda la capacidad de crear widgets personalizados para satisfacer las necesidades específicas de cada aplicación [18].

En Flutter, los widgets se disponen en una estructura de árbol, tal como se muestra en la figura 4, donde cada widget puede contener varios widgets secundarios. Esta disposición facilita la creación de interfaces de usuario elaboradas y cambiantes, ya que se actualizan automáticamente al modificar los datos que representan o el estado de la aplicación.

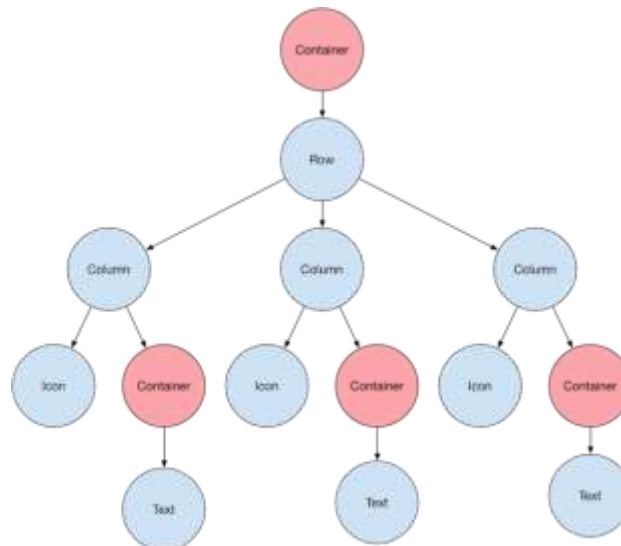


Figura 4: Árbol de Widgets
Fuente: [18]

b. Lenguaje de programación Dart

A diferencia de Android, en Flutter, Google optó por utilizar el lenguaje de programación Dart, el cual también fue desarrollado por Google en 2011 y alcanzó su primera versión estable en noviembre de 2013, casi dos años antes del debut de Flutter. Antes de la llegada de Flutter, Dart era poco conocido, pero gracias a esta plataforma, su popularidad y desarrollo han experimentado un rápido crecimiento [19].

Dart es un lenguaje de programación moderno, versátil y de alto rendimiento creado por Google. Se caracteriza por ser orientado a objetos, de tipado estático y compilado a código nativo. Dart se puede usar para desarrollar aplicaciones web, móviles, de escritorio y del lado del servidor. Algunas de las aplicaciones populares creadas con Dart incluyen Flutter, AngularDart, Firebase Admin SDK y Server-side Dart [20].

Además de su uso en desarrollo móvil con Flutter, Dart también puede utilizarse para crear aplicaciones web y servidores, gracias a su capacidad para ejecutarse tanto en el lado del cliente como en el servidor. Es un lenguaje flexible y versátil que ha ganado una base de usuarios dedicada y en crecimiento.

2.4 Herramientas de Desarrollo

2.4.1 Visual Code

Visual Studio Code es un potente editor de código fuente desarrollado por Microsoft para sistemas operativos como Windows, Linux y macOS. Entre sus características más destacadas se encuentra el sólido soporte para depuración, integración nativa con Git, resaltado de sintaxis, y una finalización inteligente de código, aunque en ocasiones requiere la instalación de paquetes adicionales. Además de estas funcionalidades, Visual Studio Code permite una alta personalización, brindando a los usuarios la capacidad de modificar temas, atajos de teclado y preferencias según sus necesidades [21].

Cabe destacar que este software es gratuito y de código abierto, convirtiéndolo en una herramienta accesible para desarrolladores de todo el mundo. Es ampliamente utilizado en la comunidad de desarrollo de Flutter debido a su versatilidad y eficiencia en la creación de aplicaciones multiplataforma, como muestra la figura 5.

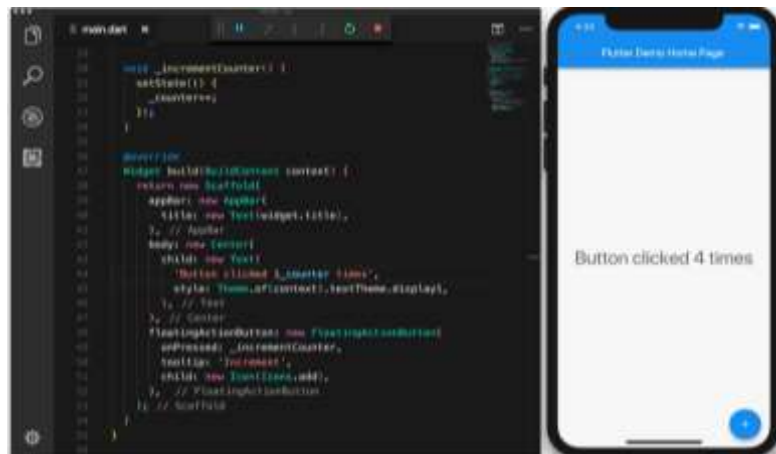


Figura 5: Visual Studio Code para el desarrollo de Flutter
Fuente: [22]

2.4.2 FlutterFlow

FlutterFlow es una plataforma de desarrollo de aplicaciones poderosa y fácil de usar que ayuda a reducir el tiempo de desarrollo al proporcionar widgets pre-creados y una interfaz visual para diseñar y construir aplicaciones, como indica la figura 6. Simplifica el proceso de desarrollo al generar automáticamente código para la interfaz de usuario de la aplicación, utilizando el framework Flutter. Esto permite a los desarrolladores centrarse en otros aspectos del proyecto mientras aseguran una aplicación receptiva y visualmente atractiva [23].

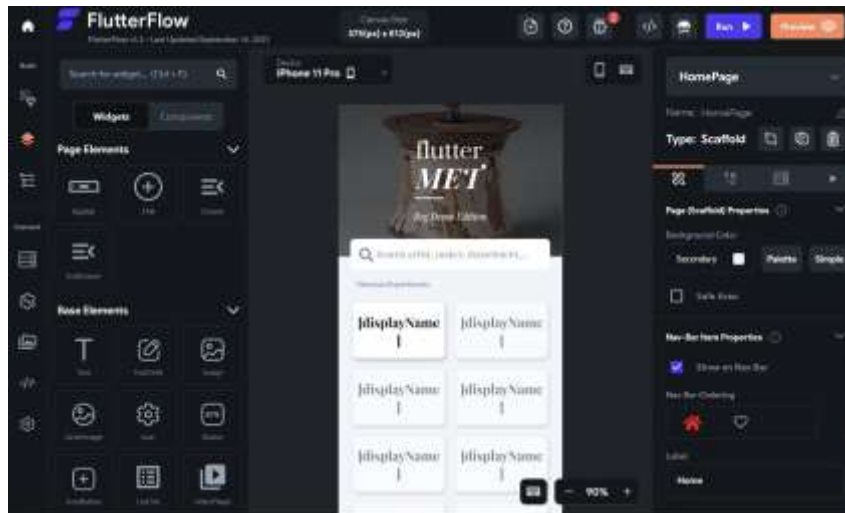


Figura 6: Interfaz de FlutterFlow
Fuente: [24]

2.4.3 Google Firebase

Firebase es una plataforma en la nube de Google que ofrece una gama de herramientas para el desarrollo de aplicaciones móviles y web [25]. Entre sus características principales se encuentran:

- Base de datos en tiempo real: Permite sincronizar datos entre dispositivos en tiempo real, ideal para aplicaciones como chats o juegos multijugador.
- Autenticación: Ofrece un sistema de autenticación flexible para que los usuarios puedan iniciar sesión en su aplicación.
- Almacenamiento: Permite almacenar archivos en la nube de Google de forma segura y escalable.
- Hosting: Ofrece un servicio de hosting para alojar aplicaciones web estáticas.
- Machine Learning: Integra herramientas de aprendizaje automático para agregar funcionalidades inteligentes a tu aplicación.
- Notificaciones push: Permite enviar notificaciones push a los usuarios de tu aplicación.
- Análisis: Ofrece herramientas para analizar el comportamiento de los usuarios en tu aplicación.

Firebase es una herramienta integral que proporciona a los desarrolladores una solución completa para desarrollar, mejorar y hacer crecer sus aplicaciones móviles y web de manera efectiva y eficiente.

2.4.4 Firebase Performance Monitoring

Firebase Performance Monitoring es una herramienta ofrecida por Firebase, una plataforma de desarrollo de aplicaciones móviles y web de Google [26]. Esta herramienta permite a los desarrolladores monitorear el rendimiento de sus aplicaciones en tiempo real y obtener información detallada sobre el rendimiento de la aplicación en diferentes dispositivos y ubicaciones geográficas.

Todos los datos recopilados por Firebase Performance se muestran en un panel de control en la consola de Firebase. Este panel permite filtrar los datos por versiones de la aplicación, países, y dispositivos, entre otros.

2.4.5 Firebase Test Lab

Firestore Test Lab es una infraestructura de prueba de aplicaciones basada en la nube que permite probar una aplicación en una amplia gama de dispositivos y configuraciones. Esto ayuda a tener una mejor idea de cómo funcionará la aplicación en las manos de usuarios reales [27].

Pruebas en dispositivos físicos y virtuales

- **Dispositivos Físicos:** Se puede ejecutar pruebas en una amplia gama de dispositivos físicos Android e iOS alojados en los centros de datos de Google.
- **Dispositivos Virtuales:** También hay la opción de realizar pruebas en emuladores y simuladores, lo que permite verificar el comportamiento de la aplicación en configuraciones específicas.

Informes Detallados

Al finalizar las pruebas, Test Lab genera informes detallados con registros, capturas de pantalla como muestra la figura 7, videos, y datos de rendimiento. Esto incluye métricas como el uso de la CPU, memoria, y batería, además de registros de fallos y errores [27] .

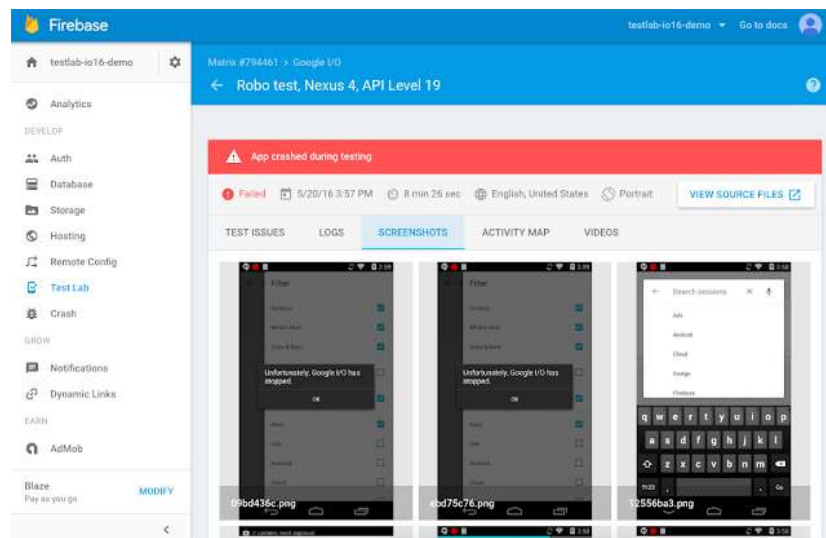


Figura 7: Pruebas con Firebase Test Lab

Fuente: [27]

2.5 Modelos de calidad de software

Los modelos de calidad de software son marcos o estructuras diseñados para evaluar y garantizar la calidad de un producto de software. Estos modelos establecen estándares y criterios que permiten medir diversos aspectos de calidad, como la funcionalidad, la fiabilidad, la eficiencia, la usabilidad y la mantenibilidad del software [28]. Además, proporcionan directrices y procesos para mejorar continuamente la calidad del software a lo

largo de su ciclo de vida, desde la planificación y el diseño hasta el desarrollo, la implementación y el mantenimiento.

2.5.1 Modelo FURPS

El modelo FURPS es un marco utilizado para evaluar los requisitos funcionales y no funcionales del software [29]. Su nombre es un acrónimo de las cinco características principales que evalúa:

- **Funcionalidad (Functionality):** Define lo que el software debe hacer en términos de características, capacidades y tareas que puede realizar.
- **Usabilidad (Usability):** Evalúa la facilidad de uso del software, considerando factores como la simplicidad de la interfaz, la claridad de la navegación y la coherencia de la información.
- **Confiabilidad (Reliability):** Mide la capacidad del software de funcionar correctamente sin fallar durante un período determinado. Incluye aspectos como la estabilidad, la tolerancia a errores y la recuperación de fallos.
- **Rendimiento (Performance):** Analiza la velocidad, el tiempo de respuesta y el uso de recursos del software bajo distintas cargas de trabajo.
- **Soporte (Supportability):** Evalúa la facilidad de mantener, modificar y actualizar el software a lo largo de su ciclo de vida. Incluye aspectos como la documentación, el modularidad y la testabilidad.

2.6 Metodología Kanban

Kanban es una metodología de gestión de proyectos que se basa en la visualización del flujo de trabajo y la mejora continua. Se originó en la industria automotriz japonesa en la década de 1940 y se ha adaptado para su uso en una amplia gama de sectores, como el desarrollo de software, la gestión de proyectos y la gestión del trabajo personal [30]. Se pueden identificar algunas secciones comunes en los tableros como muestra la figura 8, que suelen estar presentes en un sistema Kanban típico que son las siguientes:

- **Por hacer (To Do):** Esta es la columna donde se colocan las tareas o elementos de trabajo que aún no han comenzado. Representa el trabajo que se necesita hacer, pero que aún no se ha iniciado.
- **En progreso (In Progress):** En esta columna se encuentran las tareas que están en curso, es decir, aquellas en las que el equipo está trabajando actualmente. Estas tareas están en proceso de ser completadas.
- **En revisión (Review):** Algunos equipos incluyen una columna de revisión donde se colocan las tareas que han sido completadas pero que aún necesitan ser revisadas o aprobadas antes de ser consideradas terminadas.
- **Completado (Done):** Esta es la columna final donde se colocan las tareas una vez que han sido completadas satisfactoriamente. Representa el trabajo que ha sido finalizado y entregado con éxito.



Figura 8: Secciones de la metodología Kanban
Fuente: [31]

La prominencia de Kanban en el desarrollo de software se refleja en su posición como la segunda metodología ágil más utilizada, después de Scrum, según revela los datos obtenidos por Scrum en 2023 con la figura 9 [32] . Este alto grado de adopción destaca la eficacia de Kanban en el ámbito del desarrollo de software. Su enfoque en la gestión visual del flujo de trabajo, la limitación del trabajo en curso y la mejora continua resultan especialmente beneficiosos para los desarrolladores individuales.

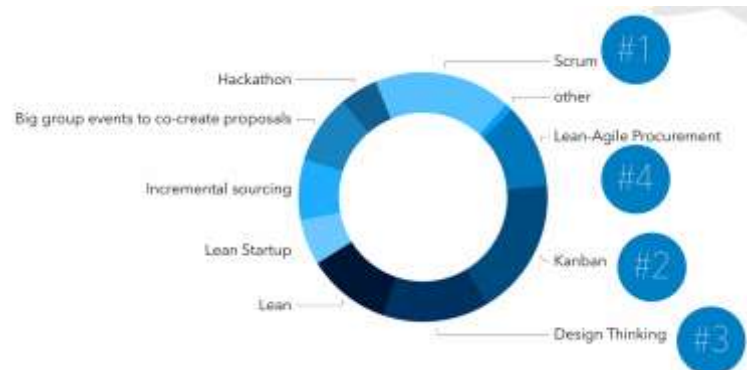


Figura 9: Metodologías ágiles más utilizadas en 2023
Fuente: [32]

Además, la capacidad de Kanban para facilitar la entrega continua y adaptarse rápidamente a los cambios del mercado y las necesidades de los clientes lo convierte en una elección especialmente ventajosa para aquellos que trabajan de forma independiente en el desarrollo de software. Según [33], sugiere que tanto Scrum como Kanban conducen al éxito en el desarrollo de proyectos; sin embargo, los resultados destacan que Kanban podría ser superior a Scrum en términos de gestión del cronograma del proyecto. Esta evidencia respalda aún más la preferencia por Kanban, puesto que ofrece una herramienta efectiva para gestionar el tiempo y los recursos en el desarrollo de software, lo que lo convierte en una opción atractiva para desarrolladores individuales.

CAPÍTULO III. METODOLOGIA

3.1 Tipo de Investigación

El presente proyecto de investigación empleó la investigación aplicada – descriptiva. Aplicada porque su objetivo principal es resolver un problema práctico. En este caso, el problema es la necesidad de facilitar el proceso de gestión de adopción de mascotas mediante el desarrollo de una aplicación móvil. La investigación se considera como descriptiva también, porque implica describir y comprender las características y comportamientos del rendimiento de la aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba.

3.2 Diseño de Investigación

Se optó por un diseño de investigación no experimental con enfoque cuantitativo y cualitativo, ya que este diseño no manipuló las variables independientes y permitió analizar y medir únicamente el rendimiento de la aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba. Con el enfoque cuantitativo el objetivo fue recopilar datos concretos y medibles sobre el rendimiento de la aplicación mediante el uso de Firebase Performance Monitoring.

3.3 Población de estudio y tamaño de muestra

De acuerdo con el tipo de investigación planteada, se consideró una población infinita, dado que se recopilaron datos de múltiples mediciones, empleando las herramientas Firebase Performance Monitoring y Firebase Test Lab para evaluar el rendimiento de la aplicación.

3.4 Técnicas de Recolección de Datos

Para recolectar datos sobre el rendimiento de la aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba, no se emplearon técnicas tradicionales, más bien se emplearon dos principales técnicas de recolección de datos. En primer lugar, se configuraron y ejecutaron pruebas automatizadas utilizando Firebase Test Lab. Estas pruebas permitieron simular el comportamiento de la aplicación en una variedad de dispositivos móviles y configuraciones de hardware, recopilando datos detallados sobre su rendimiento en cada caso. En segundo lugar, se implementó Firebase Performance Monitoring para monitorear continuamente el rendimiento de la aplicación en tiempo real mientras los usuarios la utilizaban. Esta herramienta proporcionó métricas en tiempo real sobre la tasa de rendimiento, el tiempo de respuesta, velocidad de procesamiento y consumo de recursos detallada de la aplicación durante su uso en situaciones reales.

3.5 Métodos de análisis y procesamiento de datos

Una vez recopilados los datos sobre el rendimiento de la aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba utilizando Firebase Performance Monitoring y Firebase Test Lab, se emplearon diversas técnicas de análisis y procesamiento de datos para comprender y extraer conclusiones significativas. En primer lugar, se realizaron análisis cuantitativos de los datos recopilados, calculando medidas estadísticas para evaluar el rendimiento general de la aplicación en diferentes dispositivos y condiciones.

Además, se utilizaron técnicas de visualización de datos, como gráficos y diagramas, para representar visualmente los resultados. Por último, se llevaron a cabo análisis cualitativos para interpretar los datos en el contexto más amplio de la experiencia del usuario y las necesidades de adopción de mascotas en Riobamba.

3.6 Identificación de variables

3.6.1 Variable dependiente

El rendimiento de la aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba mediante el uso del Framework Flutter.

3.6.2 Variable independiente

Aplicación móvil.

3.7 Operacionalización de variables

La presente tabla 2, muestra la operacionalización de variables.

Tabla 2. Operacionalización de variables

PROBLEMA	TEMA	OBJETIVOS	VARIABLES	CONCEPTUALIZACION	DIMENSION	INDICADORES
¿El uso del Framework Flutter incidirá en el rendimiento de una aplicación móvil para la gestión de adopción de mascotas en Riobamba?	Aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba mediante el uso del Framework Flutter	General	Independiente	Es un programa de software diseñado para ser utilizado en dispositivos móviles como teléfonos inteligentes y tabletas	Desarrollo software	<ul style="list-style-type: none"> • Compatibilidad con dispositivos • Número de funcionalidades • Tiempo de desarrollo
		Específicos	Dependiente	Flutter es un framework de código abierto desarrollado por Google que se utiliza para construir aplicaciones móviles para diferentes plataformas, incluyendo Android, iOS, web y escritorio, todo desde un único código base <ul style="list-style-type: none"> • El modelo FURPS se reconoce como un modelo de calidad fijo el cual es usado para realizar la evaluación de la calidad de un producto. • Rendimiento se refiere a la eficiencia y capacidad de respuesta de la aplicación en relación con la experiencia del usuario y la capacidad de manejar cargas de trabajo específicas. 	Rendimiento	Criterios de rendimiento mediante el modelo de calidad FURPS: <ul style="list-style-type: none"> • % Velocidad de procesamiento • % Tiempo de respuesta • % Consumo de recursos

3.8 Metodología de Desarrollo

En el desarrollo de la aplicación móvil para la gestión de adopción de mascotas en la ciudad de Riobamba utilizando el Framework Flutter, se implementó la metodología Kanban. Esta metodología emplea tableros donde las tareas se desplazan a medida que se completan. Para una mejor organización, el proyecto se dividió en cuatro fases: análisis, diseño, implementación y pruebas.

3.8.1 Análisis

Diseño del tablero Kanban

En el flujo de trabajo Kanban, las tareas se desplazan a través de varias secciones representadas por columnas. Comienza con "Por hacer (To Do)", donde las tareas pendientes esperan comenzar. Luego, pasan a "En progreso (In Progress)" mientras se trabaja en ellas. Una vez completadas, se mueven a "En revisión (Review)" para ser evaluadas antes de ser marcadas como "Completado (Done)", indicando su finalización exitosa, como lo muestra la Tabla 3 siguiente:

Tabla 3. Diseño inicial Tablero Kanban

Por hacer	En progreso	En revisión	Completado
Levantar requerimientos funcionales y no funcionales.	Realizar entrevistas al personal de PAE y CRIAR.	Estudiar las herramientas necesarias para el desarrollo de la aplicación.	
Definir roles	Realizar tablero Kanban		
Realizar diagrama de casos de uso			
Realizar diagrama de arquitectura			
Actualizar tablero Kanban			
Instalar Visual Studio Code y Flutter			
Instalar Emulador Android Studio			
Comprar licencia Flutterflow			
Configurar Google Cloud Platform y Firebase			
Diseñar y codificar la interfaz gráfica de la aplicación móvil			
Actualizar el tablero Kanban			
Pruebas de funcionalidad			
Corregir posibles errores			
Medir rendimiento de la aplicación móvil mediante Firebase Test Lab y Firebase Performance Monitoring			

Roles de la aplicación

A continuación, en la tabla 4 se presenta los roles de la aplicación.

Tabla 4. Descripción de roles

Rol		Descripción
Administrador	General	Administrar los usuarios Buscar y filtrar usuarios Administrar mascotas Resolver problemas
Usuario	No registrado	Explorar galería de animales disponibles para adopción. Buscar mascotas por categoría Acceder información general de mascotas
	Registrado	Explorar galería de animales disponibles para adopción. Buscar mascotas por categoría Acceder información de mascotas Gestionar información del animal adoptado Agregar mascotas para ser adoptadas y carga de archivos

Especificación de requerimientos funcionales

A continuación, se presenta la tabla 5, detalla los requerimientos funcionales para la aplicación de gestión de adopción de mascotas. Estos requerimientos describen las funciones específicas que la aplicación debe cumplir para satisfacer las necesidades de los usuarios y alcanzar los objetivos del proyecto. Además, los requerimientos se obtuvieron posterior a una entrevista con CRIAR y PAE, donde se recibió documentos como solicitud de adopción y acta de compromiso (ver anexo 3 y 4). Cada requerimiento está identificado con un ID único y se describe de manera concisa para facilitar su comprensión y gestión durante el desarrollo del sistema.

Tabla 5. Requerimientos Funcionales

ID	Rol	Prioridad	Funcionalidad	Descripción	Observación
RF01	Administrador, Usuario	Alta	Registro de usuarios	Los usuarios pueden crear cuentas con información básica como nombre, dirección de correo electrónico y contraseña.	
RF02	Administrador	Alta	Dashboard de Administrador	El administrador puede acceder al resumen de toda la información mediante gráficos y estadísticas.	
RF03	Administrador, Usuario registrado	Media	Gestión de perfiles de usuario	Los usuarios pueden editar y actualizar su información de perfil, incluidos detalles como la ubicación y la foto de perfil.	

RF04	Usuario registrado y no registrado	Alta	Búsqueda y filtrado de mascotas	Los usuarios pueden buscar y filtrar mascotas disponibles para adopción según su especie.
RF05	Usuario registrado	Alta	Visualización de detalles de mascotas	Los usuarios pueden ver información detallada sobre cada mascota, incluidas fotos, descripciones, historias y requisitos especiales, si los hubiera.
RF06	Usuario registrado y no registrado, Administrador	Alta	Gestión de mascotas rescatadas	Los usuarios y administradores pueden agregar información sobre mascotas rescatadas que necesiten ser adoptadas.
RF07	Usuario registrado	Media	Comunicación entre usuarios	Los usuarios pueden comunicarse entre sí a través de la aplicación.

Especificación de requerimientos no funcionales

La Tabla 6 presenta el formato empleado para describir los requisitos no funcionales en el proyecto. Para una visión completa de todos los requisitos, consulte el Anexo 2. Estos requisitos complementan los aspectos funcionales del sistema al abordar áreas como usabilidad, seguridad, tecnología, entre otros. Cada requerimiento no funcional proporciona directrices específicas que guiarán el desarrollo y la implementación de la aplicación para garantizar una experiencia de usuario óptima y un funcionamiento eficiente y confiable.

Tabla 6. Requerimientos no funcionales

ID	Categoría	Requerimiento	Descripción
RNF01	Usabilidad	Facilidad de Uso	La aplicación móvil debe ser diseñada de manera que resulte intuitiva y accesible para los usuarios finales, promoviendo una experiencia amigable y sin complicaciones.

3.8.2 Diseño del sistema

Diagrama de casos de uso

A continuación, en la figura 10 se puede observar el diagrama de casos de uso donde se ilustra las interacciones principales entre los usuarios y la aplicación.

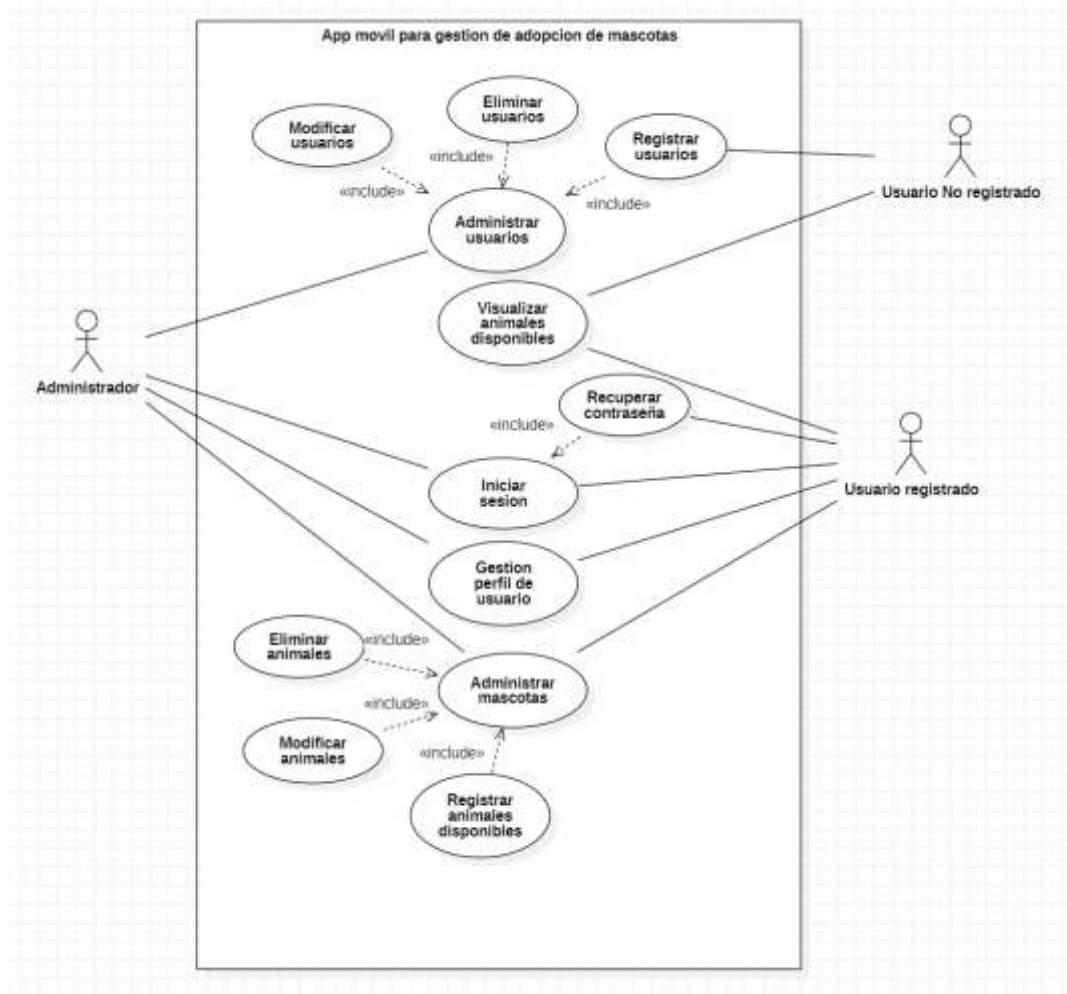


Figura 10: Diagrama de casos de uso

Arquitectura cliente servidor de la aplicación

La arquitectura empleada en el diseño de la aplicación se definió como cliente-servidor, donde todas las solicitudes realizadas a través de la aplicación son procesadas por el servidor. Para este proyecto, se utilizó Firebase Firestore como servidor, lo que permite un manejo eficiente y seguro de la base de datos en tiempo real. A continuación, en la figura 11 se presenta el diagrama de arquitectura que ilustra cómo se integran estos componentes en el sistema.

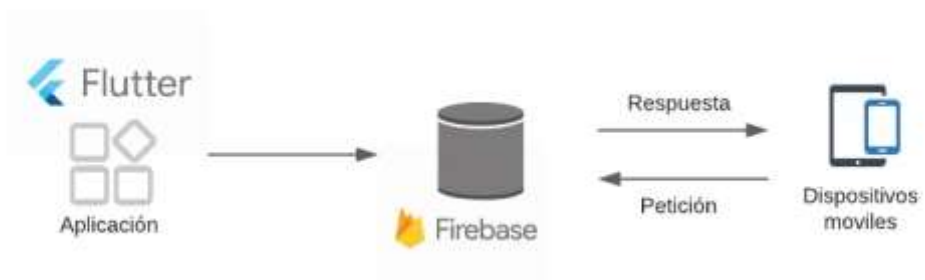


Figura 11: Arquitectura cliente-servidor

Esquema base de datos no relacional

Para desarrollar la aplicación móvil, se utilizó Firebase, una base de datos NoSQL en tiempo real. Esto proporciona una estructura de datos flexible que puede adaptarse y cambiar según las necesidades de la aplicación. Firebase almacena los datos en documentos y colecciones, como se ilustra en la figura 12, a diferencia de las bases de datos SQL que utilizan tablas para este propósito.

Colección Users	
uid	string
display_name	string
email	string
password	string
photo_url	Image Path
phone_number	String
address	String
cedula_number	String
birthday_date	DateTime
created_time	DateTime

Figura 12: Esquema colección “Users”

A continuación, en la figura 13 se muestra las colecciones Pets, Acuerdo_Adopcion, Chats, Chat_messages, utilizadas para el funcionamiento de la aplicación.

Colección Pets	
id_pet	string
name	string
specie	string
pets_photos	List < Image Path >
uid	string
created_time	DateTime
age_year_pet	string
age_month_pet	string
size_pet	string
breed_pet	string
sex_pet	string
description_pet	string

a) Esquema colección “Pets”

Colección Chats	
user_a	Doc Reference (Users)
user_b	Doc Reference (Users)
last_message	string
last_message_time	DateTime
user	Doc Reference (Users)
image	Image Path
message_seen	Boolean

b) Esquema colección “Chats”

Colección Chat_messages	
user	Doc Reference (Users)
chat_messages	Doc Reference (chats)
text	string
image	Image Path
timestamp	DateTime

c) Esquema colección “Chat_messages”

Figura 13: Esquemas de colecciones de la base de datos NoSQL

Mapa de navegabilidad

Para la creación de la aplicación móvil mediante el uso del framework Flutter, se utilizó la herramienta FlutterFlow en donde se diseñó las interfaces brindando un mapa de navegabilidad accesible. En la figura 14 se ilustra el mapa de Login y registro de usuarios.

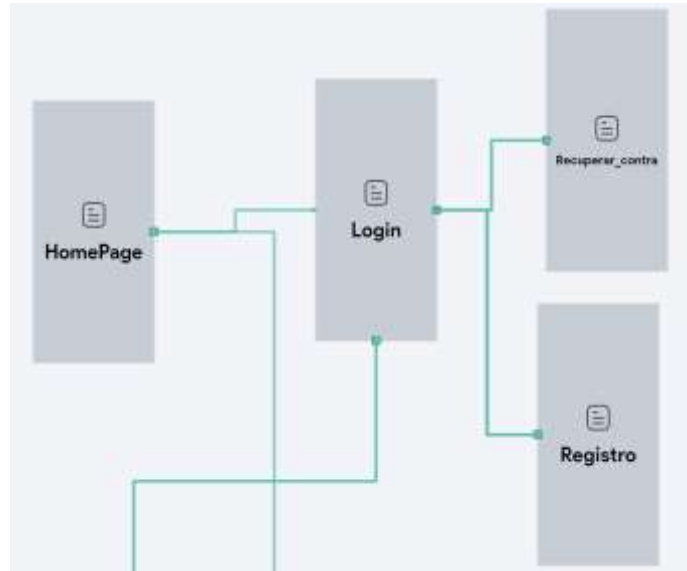


Figura 14: Mapa de navegabilidad de Login y Registro de Usuarios

La figura 15 muestra el diseño del mapa de navegabilidad del perfil de usuario donde se puede acceder y editar el perfil.

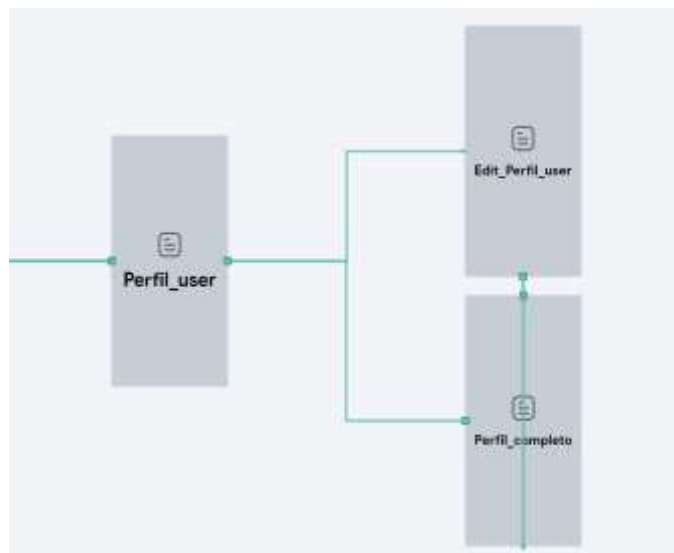


Figura 15: Mapa de navegabilidad de perfil y editar usuario

A continuación, en la figura 16 se puede visualizar el mapa de navegabilidad del registro y lista de mascotas a hacer adoptadas.

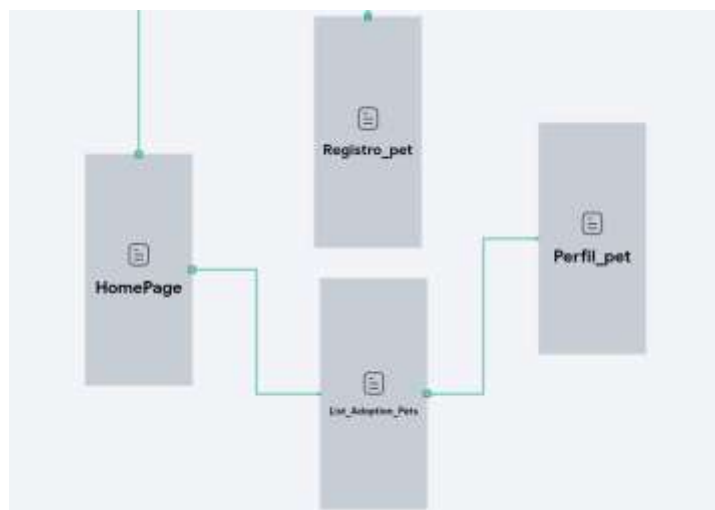


Figura 16: Mapa de navegabilidad del registro y lista de mascotas.

Actualización Tablero Kanban

Al finalizar la fase de Diseño del sistema, se realizó una actualización del tablero Kanban para visualizar el avance como se muestra en la tabla 7.

Tabla 7. Tablero Kanban, fase Diseño

Por hacer	En progreso	En revisión	Completado
Actualizar tablero Kanban	Diseñar y codificar la interfaz gráfica de la aplicación móvil	Instalar Visual Studio Code y Flutter	Estudiar las herramientas necesarias para el desarrollo de la aplicación.
Pruebas de funcionalidad		Instalar Emulador Android Studio	Realizar entrevistas al personal de PAE y CRIAR.
Corregir posibles errores		Comprar Flutterflow licencia	Levantar requerimientos funcionales y no funcionales.
Medir rendimiento de la aplicación móvil mediante Firebase Test Lab y Firebase Performance Monitoring		Configurar Google Cloud Platform y Firebase	Definir roles
			Realizar diagrama de casos de uso
			Realizar diagrama de arquitectura

3.8.3 Implementación

Interfaz grafica

Después de crear el mapa de navegación, se procedió al desarrollo de la aplicación móvil y sus interfaces utilizando Flutterflow, como se ilustra en la Figura 17, que muestra las interfaces de inicio de sesión. Para ver todas las interfaces, consulte el Anexo 5.



a) Home Page



b) Login



c) Crear cuenta



d) Recuperar contraseña

Figura 17: Interfaces de Inicio de sesión

Base de datos NoSQL Firebase

Al implementar la base de datos NoSQL se realizó mediante colecciones, a continuación, en la figura 18 se muestra hecha en Firebase.

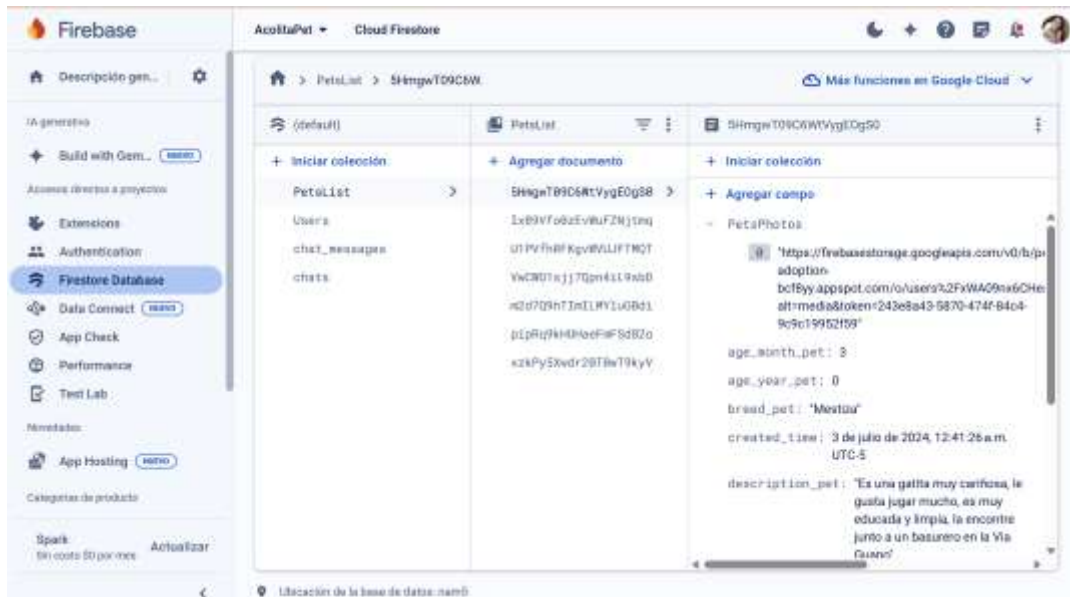


Figura 18: Base de datos implementada en Firebase

Codificación

La figura 19 muestra la estructura de la codificación mediante el uso del framework Flutter. A la izquierda, se visualizan las páginas que contiene el proyecto; en el centro, los widgets que se crean para dar como resultado lo que se visualiza a la derecha.

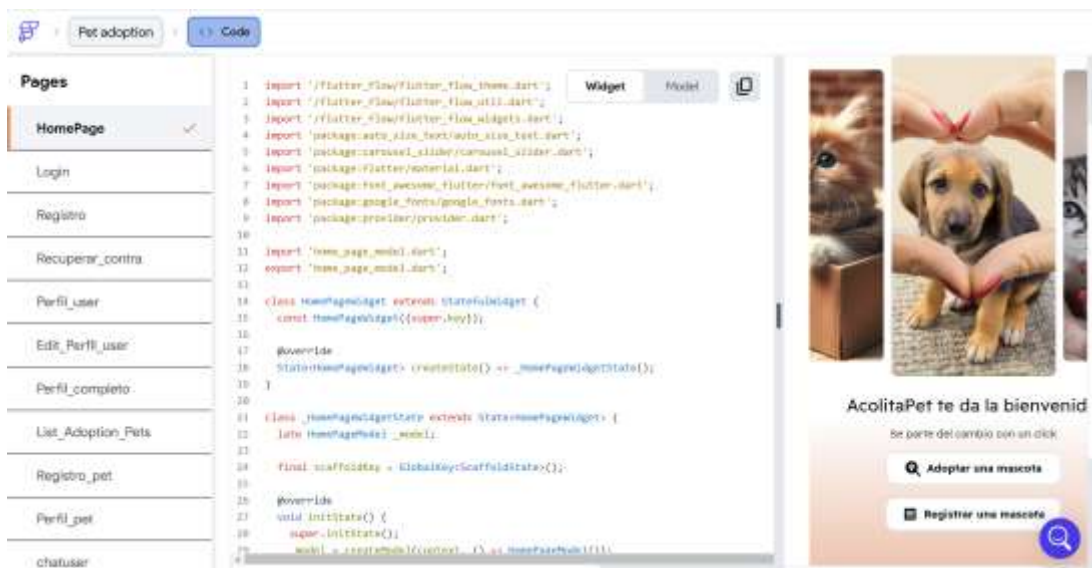


Figura 19: Codificación de la interfaz HomePage

3.8.4 Pruebas

Una vez completado el desarrollo de la aplicación, se procedió a cumplir con el último objetivo de este trabajo de investigación. Se llevaron a cabo pruebas para medir el rendimiento de la aplicación móvil destinada a la gestión de adopciones de mascotas. Para ello, se utilizó el Firebase Test Lab para simular diversos dispositivos y Firebase Performance Monitoring para obtener métricas detalladas, como el tiempo de respuesta, velocidad de procesamiento y el consumo de recursos, criterios de rendimiento según el modelo FURPS.

Configuración Firebase Test Lab

Inicialmente, se configuró Firebase Test Lab. Para esto, se generó el archivo APK de la aplicación y se procedió a subirlo, como se muestra en la figura 20.



Figura 20: Subir APK a Test Lab

Sucesivamente se seleccionó la prueba Robo que es una prueba automática y se eligió los diferentes dispositivos de gama alta como media para las pruebas, como muestra la figura 21. Por cada prueba se puede llegar a seleccionar de 1 a 15 dispositivos virtuales o físicos, sin embargo, a su vez se realizaron 8 veces las pruebas.



Figura 21: Elección de dispositivos Firebase Test Lab

También se pudo personalizar las pruebas y se seleccionó el tiempo en que se quería que los dispositivos utilicen la aplicación, mientras mayor sea el tiempo, mayor será el uso de todos los módulos que se prueban dentro de la aplicación.

Para este caso se tomó 10 minutos con los dispositivos físicos y virtuales ejecutando y al final se mostraron si se realizó con éxito en todos los dispositivos seleccionados, como se puede ver en la figura 22.

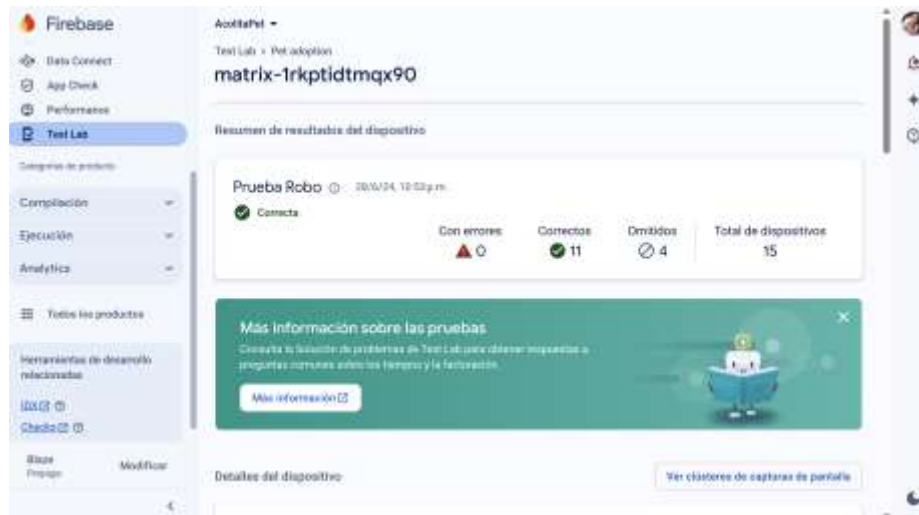


Figura 22: Pruebas del APK en Firebase Test Lab

Monitorear rendimiento mediante Firebase Performance Monitoring

Después de que las pruebas se ejecutaron exitosamente en Firebase Test Lab, se accedió a la consola de Firebase para revisar los informes de rendimiento. Firebase Performance Monitoring proporcionó datos detallados sobre la tasa de rendimiento, realizando infinitas pruebas a la vez, como muestra la figura 23, permitiendo así monitorear de manera efectiva el rendimiento de la aplicación.



Figura 23: Dashboard de Firebase Performance Monitoring de la aplicación móvil.

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

4.1 Resultados

Tras realizar las pruebas correspondientes, se cumplió el objetivo de evaluar el rendimiento de la aplicación móvil mostrando el tiempo de respuesta, velocidad de procesamiento y el consumo de recursos. A continuación, se presentan los resultados obtenidos con los indicadores según el modelo FURPS.

4.1.1 Evaluación del Tiempo de Respuesta

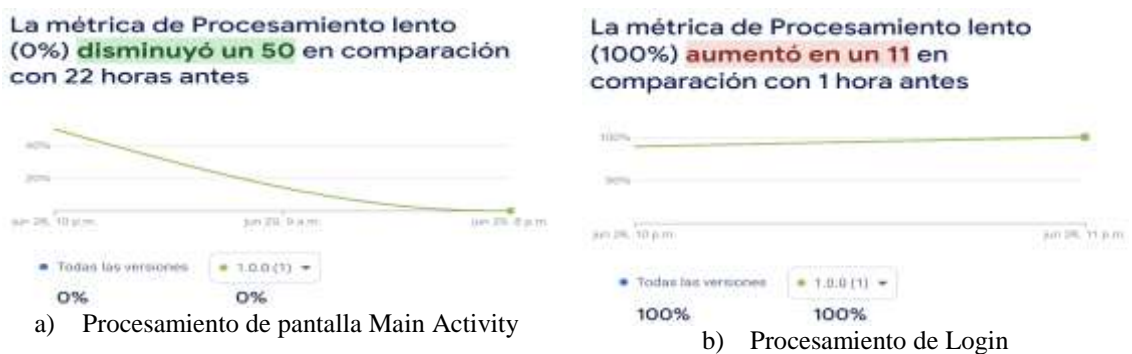
Para medir el tiempo de respuesta de la aplicación se hicieron varias pruebas, en donde Firebase Performance Monitoring dio los siguientes resultados mostrados en la figura 24.



Figura 24: Resultados tiempo de respuesta

4.1.2 Evaluación de Velocidad de Procesamiento

Para medir la velocidad de procesamiento de la misma manera se obtuvieron porcentajes de velocidad comparando de varios dispositivos a la vez, como se muestra en la figura 25.



c) Rendimiento de renderización

Figura 25: Resultados Velocidad de procesamiento

4.1.3 Evaluación de Consumo de recursos

Para medir el uso de recursos se evaluó en varios dispositivos de gama alta como gama media, dando como media los siguientes resultados en la tabla 8, además se puede observar las gráficas en el Anexo 5.

Tabla 8. Resultados de uso de recursos

	Dispositivos Gama Alta	Dispositivos Gama Media
CPU	14%	26%
Memoria RAM	400 KiB	100KiB

Según la Tabla 8 se puede observar que los procesadores de gama media pueden ser menos potentes, lo que requiere un mayor uso de recursos de CPU para realizar las mismas tareas. También se observa que los dispositivos de gama alta utilizan una cantidad significativamente mayor de memoria RAM en comparación con los de gama media, esto puede entenderse porque los dispositivos de gama alta suelen tener más RAM disponible, lo que les permite ejecutar aplicaciones más exigentes y tener más pestañas o aplicaciones abiertas al mismo tiempo.

4.2 Discusión

Para obtener los resultados anteriores, las pruebas se realizaron con la herramienta Firebase Test Lab y Firebase Performance Monitoring, las cuales brindaron métricas para evaluar el rendimiento, a su vez se realizaron comparativas con estudios similares de rendimiento en aplicaciones móviles, llegando a las siguientes discusiones de los resultados.

Tiempo de respuesta

La evaluación del tiempo de respuesta de la aplicación mostró resultados positivos, cumpliendo con las expectativas establecidas. Las pruebas realizadas revelaron los siguientes puntos clave.

- La aplicación demostró tiempos de inicio rápidos de 452ms, con un 1% más rápido en comparación de varios dispositivos interactuando a la vez por cada prueba robo, lo cual es crucial para la primera impresión del usuario. Incluso con un número creciente de dispositivos, el tiempo de inicio se mantuvo dentro de parámetros aceptables.
- Según un estudio a gran escala, en el cual se analizó el rendimiento de 215 aplicaciones móviles [34], se sugiere que el tiempo óptimo de respuesta debe ser inferior a 1 segundo para garantizar una experiencia fluida para el usuario. Al comparar estos resultados con los obtenidos en la aplicación desarrollada en esta investigación, se puede concluir que el rendimiento de la aplicación es favorable, puesto que cumple con este estándar y asegura una interacción rápida y eficiente.

Velocidad de procesamiento

- Procesamiento de la pantalla Home Page y Login, estas funciones, esenciales para la experiencia del usuario, mostraron tiempos de procesamiento eficientes, en donde se visualizó una disminución de 50% en el Home Page y el Login un aumento de 11%.
- En términos de rendimiento de renderización, la aplicación desarrollada en esta investigación presentó un 0% de fotogramas congelados, lo que garantiza una experiencia fluida y sin interrupciones, esencial para mantener la satisfacción del usuario. En comparación con el estudio [35], que sugiere que superar el 1% de fotogramas congelados (FPS) podría indicar problemas de renderización, el rendimiento de esta aplicación está en óptimas condiciones y cumple con los estándares recomendados.

Consumo de recursos

La medición del consumo de recursos en dispositivos de gama alta y media reveló diferencias significativas, como se muestra en la tabla 8, estas diferencias se deben principalmente a las capacidades de hardware de los dispositivos. Los dispositivos de gama media, con procesadores menos potentes, requieren un mayor uso de CPU para realizar las mismas tareas. Por otro lado, los dispositivos de gama alta utilizan más memoria RAM, lo cual es comprensible debido a su mayor disponibilidad de RAM, permitiendo ejecutar aplicaciones más exigentes y mantener múltiples aplicaciones abiertas simultáneamente.

Además, el estudio [34] señala que el uso de CPU en dispositivos de gama alta o media, mientras la aplicación está activa, es generalmente inferior al 30%, dependiendo del tipo de aplicación. Esto sugiere que el consumo de recursos es eficiente y adecuado para estos dispositivos.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

El análisis de Flutter ha demostrado que este framework no solo es una herramienta eficaz, sino también revolucionaria en el desarrollo de aplicaciones móviles. Su capacidad de generar aplicaciones multiplataforma desde una única base de código representa un avance crucial en la optimización del tiempo y los costos de desarrollo, permitiendo a las empresas estar presentes tanto en iOS como en Android sin comprometer la calidad. Además, Flutter compite con las soluciones específicas de cada plataforma, gracias a su combinación de un motor de renderizado rápido, compilación directa a código máquina, su función Hot-Reload, una arquitectura eficiente y una gran capacidad de optimización.

La implementación de Flutter en el desarrollo de la aplicación móvil para la gestión de adopciones de mascotas en Riobamba ha sido sumamente exitosa. El uso de este framework permitió la creación de una aplicación sólida, eficiente y visualmente atractiva en un tiempo considerablemente más corto que otros enfoques tradicionales. La aplicación no solo destaca por su interfaz intuitiva, sino también por proporcionar una experiencia de usuario que potencia la interacción y el compromiso. Además, la estructura de código único de Flutter simplifica de manera excepcional el mantenimiento y la incorporación de futuras mejoras, asegurando que la aplicación se mantenga al día y funcione de forma óptima con un esfuerzo mínimo.

La evaluación del rendimiento de la aplicación de adopción de mascotas ha sido altamente positiva. Con un tiempo de inicio de 452 ms, la aplicación ofrece una experiencia ágil desde el primer uso, superando el estándar recomendado de 1 segundo. La velocidad de procesamiento también fue eficiente, con mejoras del 50% en la Home Page y un 0% de fotogramas congelados, lo que garantiza una interacción fluida y sin interrupciones. El consumo de recursos se mantuvo óptimo, con un uso de CPU inferior al 30%, incluso en dispositivos de gama media. En conjunto, la aplicación cumple con los estándares de rendimiento, asegurando una experiencia rápida, estable y de alta calidad para los usuarios.

Finalmente, al haber desarrollado una aplicación móvil accesible para el público en general, se realizaron pruebas piloto en una organización dedicada a la adopción de mascotas PAE. La aplicación fue aceptada por esta organización, lo que demuestra su viabilidad y utilidad en un entorno real. Se adjunta en el anexo 7 imágenes que respaldan la implementación exitosa del proyecto.

5.2 Recomendaciones

Dada la eficacia del framework Flutter se recomienda continuar explorando y actualizándose con las últimas novedades y mejores prácticas de Flutter. Participar en comunidades de desarrolladores, asistir a conferencias y seguir la documentación oficial permitirá mantenerse al día con las nuevas funcionalidades y mejoras del framework, asegurando que se aprovechen al máximo sus capacidades para futuros proyectos.

Es importante mantener una práctica constante de revisión y optimización del código para asegurar la eficiencia y la sostenibilidad a largo plazo. Involucrar a los usuarios finales en el proceso de desarrollo mediante encuestas y pruebas de usabilidad periódicas puede proporcionar información valiosa para mejorar la aplicación. Además, considerar la implementación de nuevas funcionalidades basadas en las necesidades emergentes de los usuarios y avances tecnológicos.

Utilizar herramientas de análisis de rendimiento y realizar evaluaciones regulares basadas en el modelo FURPS garantizará que la aplicación siga cumpliendo con los estándares de calidad. Además, fomentar una cultura de mejora continua dentro del equipo de desarrollo para asegurarse de que la aplicación se mantenga eficiente y satisfactoria para los usuarios a lo largo del tiempo.

BIBLIOGRAFÍA

- [1] “Día Mundial del Perro: el 70% de los que hay en el mundo no tienen hogar - Diario La Provincia SJ”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://www.diariolaprovinciasj.com/elmundo/2021/7/21/dia-mundial-del-perro-el-70-de-los-que-hay-en-el-mundo-no-tienen-hogar-156853.html>
- [2] “Flutter Showcase | Google Classroom”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://flutter.dev/showcase/google-classroom>
- [3] P. De Titulación, I. Joffre, y V. L. Acurio, “UNIVERSIDAD TÉCNICA DE BABAHOYO FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA EXAMEN COMPLEXIVO DE GRADO O DE FIN DE CARRERA PRUEBA PRÁCTICA INGENIERÍA EN SISTEMAS”.
- [4] “¿Cómo abordar la problemática de los perros abandonados? - Diario La Prensa Riobamba”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://www.laprensa.com.ec/perros-abandonados-en-riobamba/>
- [5] “El abandono de perros en Quito se incrementó en 90% – Diario La Hora”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://www.lahora.com.ec/pais/el-abandono-de-perros-en-quito-se-incremento-en-90/>
- [6] S. Pan, T. Baranwal, T. Roy, P. Ghosh, K. Bhattacharjee, y A. Das, “Wind Chat: Robust & Dynamic Flutter Messaging Application”, *International Journal Of Scientific Research In Engineering & Technology*, pp. 116–121, abr. 2024, doi: 10.59256/IJSREAT.20240402018.
- [7] T. Fin De Grado, A. : David, y V. Pardo, “Crear y desarrollar una aplicación de alto rendimiento con bajo coste utilizando Flutter y Firebase”.
- [8] “Vista de Desarrollo Híbrido con Flutter”. Consultado: el 27 de abril de 2024. [En línea]. Disponible en: <https://www.ciencialatina.org/index.php/cienciala/article/view/2959/4350>
- [9] C. Romarís Caamaño, “Aplicación Web para la Gestión de Adopción y Localización de Mascotas Perdidas”.
- [10] “EL CONCEJO MUNICIPAL DEL CANTÓN RIOBAMBA CONSIDERANDO: EL CONCEJO DEL GOBIERNO AUTÓNOMO DESCENTRALIZADO MUNICIPAL DEL CANTÓN RIOBAMBA”.
- [11] D. N. Peña Quiroga, “Aplicación móvil para el servicio de adopción de mascotas en la fundación hogar de perros”, UNIVERSIDAD SANTO TOMÁS, Villavicencio, 2021. Consultado: el 4 de mayo de 2024. [En línea]. Disponible en: <https://repository.usta.edu.co/handle/11634/35288>
- [12] “Flutter - Crea aplicaciones para cualquier pantalla”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://esflutter.dev/>
- [13] L. Freth, R. Martínez, J. Guillermo, y F. Burgos, “INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN CONVENCIONAL Comité de Acreditación y Currículo Facultad de Ingenierías APLICACIÓN MÓVIL UTILIZANDO EL FRAMEWORK FLUTTER PARA LA CARNETIZACIÓN DE LOS ESTUDIANTES DE LA UNIVERSIDAD DE CÓRDOBA. UNIVERSIDAD DE CÓRDOBA FACULTAD DE INGENIERÍAS DEPARTAMENTO DE SISTEMAS MONTERÍA, CÓRDOBA Por una universidad con calidad, moderna e incluyente”, Consultado: el 27 de abril de 2024. [En línea]. Disponible en: www.unicordoba.edu.co
- [14] “Flutter para la Web. Hola a todos, les traigo un pequeño... | by Eduardo CQ | Comunidad Flutter | Medium”. Consultado: el 27 de abril de 2024. [En línea].

- Disponible en: <https://medium.com/comunidad-flutter/flutter-para-la-web-91d8b217ad35>
- [15] “Flutter architectural overview | Flutter”. Consultado: el 27 de abril de 2024. [En línea]. Disponible en: <https://docs.flutter.dev/resources/architectural-overview>
 - [16] “Cross-platform mobile frameworks used by global developers 2022 | Statista”. Consultado: el 23 de abril de 2024. [En línea]. Disponible en: <https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/?ref=itdo.com>
 - [17] R. Ibrahim Guardia, “Flutter en el Drone engineering ecosystem: desarrollo de tutorial y estudio sobre videostreaming”, feb. 2024, Consultado: el 27 de abril de 2024. [En línea]. Disponible en: <https://upcommons.upc.edu/handle/2117/403655>
 - [18] J. Ignacio, C. Carmona, y M. Rubio Sánchez, “APLICACIÓN WEB BASADA EN EL FRAMEWORK FLUTTER PARA LA ENSEÑANZA DEL DISEÑO DE ALGORITMOS RECURSIVOS DE MANERA VISUAL”, jun. 2023, Consultado: el 27 de abril de 2024. [En línea]. Disponible en: <https://eciencia.urjc.es/handle/10115/22704>
 - [19] M. da Silva, A. Henrique Tutor, y D. Víctor Manuel Rivas Santos, “Trabajo Fin de Máster”, 2021.
 - [20] “dart-lang/sdk: The Dart SDK, including the VM, dart2js, core libraries, and more.” Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://github.com/dart-lang/sdk>
 - [21] D. A. Romero, P. Tutor, : D Blas, y T. García, “Aplicación multiplataforma para Prevención de Riesgos Laborales con Flutter y Firebase”.
 - [22] “Visual Studio Code para el desarrollo de flutter | by Jaime Hernández | Medium”. Consultado: el 4 de mayo de 2024. [En línea]. Disponible en: <https://devjaime.medium.com/visual-studio-code-para-el-desarrollo-de-flutter-256dbd13b453>
 - [23] J. D. R. Colón y J. Duffany, “A Personal Health Monitoring and Emergency Assistance Mobile Application”.
 - [24] “What is FlutterFlow? A visual app builder for Flutter • The Register”. Consultado: el 4 de mayo de 2024. [En línea]. Disponible en: https://www.theregister.com/2021/09/22/an_easier_way_to_flutter/
 - [25] “Firebase | Google’s Mobile and Web App Development Platform”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://firebase.google.com/?hl=es-419>
 - [26] “Firebase Performance Monitoring”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://firebase.google.com/docs/perf-mon?hl=es-419>
 - [27] “Firebase Test Lab”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://firebase.google.com/docs/test-lab?hl=es-419>
 - [28] “Modelos de calidad de software: Una revisión sistemática de la literatura - PDF Free Download”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://docplayer.es/17951650-Modelos-de-calidad-de-software-una-revision-sistemica-de-la-literatura.html>
 - [29] M. Callejas-Cuervo, A. C. Alarcón-Aldana, y A. M. Álvarez-Carreño, “Modelos de calidad del software, un estado del arte”, *ENTRAMADO*, vol. 13, núm. 1, pp. 236–250, 2017, doi: 10.18041/ENTRAMADO.2017V13N1.25125.
 - [30] “¿Qué es la metodología Kanban y cómo funciona? [2024] • Asana”. Consultado: el 6 de abril de 2024. [En línea]. Disponible en: <https://asana.com/es/resources/what-is-kanban>

- [31] “Agile Software Development Lifecycle: A Comprehensive Guide | Agile”. Consultado: el 4 de mayo de 2024. [En línea]. Disponible en: <https://agilie.com/blog/agile-software-development-lifecycle>
- [32] “STATE OF AGILITY global annual report 2023 IN PROCUREMENT & SUPPLY with the focus topic”.
- [33] H. Lei, F. Ganjeizadeh, P. K. Jayachandran, y P. Ozcan, “A statistical analysis of the effects of Scrum and Kanban on software development projects”, *Robot Comput Integr Manuf*, vol. 43, pp. 59–67, feb. 2017, doi: 10.1016/J.RCIM.2015.12.001.
- [34] R. Rua y J. Saraiva, “A large-scale empirical study on mobile performance: energy, run-time and memory”, *Empir Softw Eng*, vol. 29, núm. 1, pp. 1–56, feb. 2024, doi: 10.1007/S10664-023-10391-Y/TABLES/14.
- [35] Ginny, C. Kumar, y K. Naik, “Smartphone processor architecture, operations, and functions: current state-of-the-art and future outlook: energy performance trade-off: Energy–performance trade-off for smartphone processors”, *Journal of Supercomputing*, vol. 77, núm. 2, pp. 1377–1454, feb. 2021, doi: 10.1007/S11227-020-03312-Z/TABLES/16.

ANEXOS

ANEXO 1

Artículo 1 de la Ordenanza Municipal 004-2019

Artículo 1.- Objeto.- La presente Ordenanza tiene por objeto establecer normas, mecanismos y herramientas para el bienestar animal, que permita regular la protección, tenencia y control de la fauna urbana en el cantón Riobamba, generando y promoviendo una convivencia armónica, pacífica y responsable entre los ciudadanos y los animales de compañía, oficio, consumo, entretenimiento y experimentación.

Nota: El Artículo 1 de la Ordenanza Municipal 004-2019 se presenta en formato imagen para preservar su presentación original.

ANEXO 2

Requerimientos no funcionales

ID	Categoría	Requerimiento	Descripción
RNF02	Seguridad	Seguridad de Acceso	Deberá existir un proceso adecuado de autenticación para los usuarios.
RNF03	Tecnología	Tecnología de Desarrollo	La aplicación móvil debe ser desarrollada utilizando el framework Flutter, asegurando así una estructura tecnológica sólida y coherente para su funcionamiento.
RNF04	Fiabilidad	Fiabilidad del Sistema	Es fundamental que el sistema cuente con manuales de usuario detallados, garantizando así la disponibilidad de recursos que faciliten la comprensión y el uso adecuado de la aplicación por parte de los usuarios
RNF05	Integración	Integración con Servicios Externos	Se posibilitará la utilización de los servicios proporcionados por Firebase, permitiendo una integración fluida y eficiente con funcionalidades adicionales que enriquezcan la experiencia del usuario.
RNF06	Operatividad	Operatividad y Navegación	La interfaz de usuario de las aplicaciones debe ser diseñada de manera que proporcione una navegación fácil e intuitiva, facilitando así la interacción de los usuarios con todas las funciones y características de la aplicación

ANEXO 3

Formato de solicitud de adopción proporcionado por PAE Riobamba



SOLICITUD DE ADOPCIÓN

SOLICITUD N°:	FECHA:	NEGADO	APROBADO
ID	ANIMAL ESCOGIDO	RASGOS CARACTERÍSTICOS	
<p><i>Nuestro principal compromiso es asegurar que el animal a ser adoptado llegue a un hogar que presente las mejores condiciones en cuanto a su bienestar animal. PAE Riobamba tiene el derecho absoluto sobre los animales y se reserva el derecho de proporcionar información sobre la aprobación o rechazo de la solicitud.</i></p>		<p>.....</p> <p>.....</p> <p>.....</p>	

Todos los animales de PAE Riobamba que salen en adopción, lo hacen **ESTERILIZADOS** como política interna y como parte de un manejo responsable de animales para evitar su irresponsable reproducción.

Esta solicitud será aprobada o rechazada en un plazo de 48 horas, para ello nosotros nos contactaremos con usted.

USTED DESEA ADOPTAR

CANINO				FELINO			
MACHO		ADULTO		MACHO		ADULTO	
HEMбра		CACHORRO		HEMбра		CACHORRO	
TAMAÑO:				TAMAÑO:			
SITUACIÓN PERSONAL							
NOMBRE COMPLETO:				CÉDULA:			
DIRECCIÓN:				TELÉFONO:			
EDAD:	E MAIL:			ESTADO CIVIL:			
TRABAJA:	NOMBRE DEL SITIO DE TRABAJO:			ACTIVIDAD:			
INGRESO MENSUAL APROXIMADO (OPCIONAL)				PERSONAS CON LAS QUE VIVE (OPCIONAL)			
0 - 200		501 - MÁS		Edad y parentesco:			
201 - 500		OTRO				
TIPO DE VIVIENDA				PERMANENCIA DEL ANIMAL			
PROPIO	CASA	CERRAMIENTO DE MALLA		ENCADENADO	TERRAZA		
ARRIENDO	DEPARTAMENTO	CERRAMIENTO DE PARED		PATIO/JARDIN	INTERIOR		

RELACIÓN CON LOS ANIMALES

EL ANIMAL ADOPTADO SERÁ PARA		EN CASO DE ENFERMEDAD UD	
COMPañIA	FAMILIAR O AMIGO	MEDICA USTED	ESPERA A QUE SE SANE
CUIDAR CASA	OTRO	LLEVA AL VETERINARIO	LLEVA AL CENTRO DE SALUD
DIETA QUE PROPORCIONARÁ A SU ANIMAL		EN CASO DE VIAJES LARGOS, SU ANIMAL	
BALANCEADO	RESTOS	VIAJA CON USTED	FAMILIAR O AMIGO
DIETA CASERA	OTROS	HOSPEDAJE	OTRO
LA ADOPCIÓN FUE DISCUTIDA CON SU FAMILIA?		LA FAMILIA ESTÁ	
	SI NO	TOTALMENTE DE ACUERDO	DESACUERDO
		LO ACEPTARON POR USTED	INDIFERENTE
		ATENDIDO POR:	
		OBSERVACIONES:	
		
Firma del Solicitante			

ANEXO 4

Formato de Acta de adopción proporcionado por PAE Riobamba



ACTA DE ADOPCIÓN

A los ___ días del mes de _____ de 20 ____, comparecen a la suscripción de la presente acta, por una parte la Fundación Protección Animal Ecuador - Capítulo Riobamba, debidamente representada por Silvia Ramos Samaniego, a quién en adelante se le denominará "PAE", y por otra parte el/la señor/a _____ portadora de la cédula de ciudadanía _____, a quien en adelante se le denominará simplemente "Adoptante" la misma que será Dueño/a y Responsable de la mascota objeto del presente instrumento.

"PAE" Capítulo Riobamba declara haber recogido y/o rescatado como lo faculta la Ordenanza de Tenencia Responsable de Mascotas un _____ con las siguientes características:

RAZA	SEXO	EDAD APROXIMADA		TAMAÑO
		AÑOS	MESES	

ID	RASGOS CARACTERÍSTICOS

COMPROMISO

El "Adoptante" domiciliado en las calles _____ de la ciudad de _____, con teléfono _____, declara tener condiciones suficientes para adoptar a la mascota objeto del convenio, por lo que "PAE", le entrega en ADOPCIÓN, y por el presente convenio el "Adoptante" se compromete a: Alimentarle adecuadamente, brindarle atención veterinaria, colocarle una plaquita de identificación, ubicarlo en su domicilio en un espacio adecuado, mantener contacto humano y cercano con el animal, respetar y limpiar adecuadamente sus necesidades físicas, demás necesidades propias del animal.

El "Adoptante" se compromete además a permitir la visita de un representante de "PAE" a fin de verificar, realizar seguimiento, y demás que considere pertinente de la mascota y el cumplimiento del presente instrumento.

El "Adoptante" queda impedido de entregar la mascota producto de esta acta a terceros sea por venta, donación o empréstito, cualquier tipo de acción para tenencia de la mascota deberá ser aprobada previamente por "PAE".

En caso de incumplimiento de esta acta o de la ordenanza de tenencia responsable de mascotas, el "Adoptante" faculta a "PAE" para retirar, rescatar o intervenir a la mascota con sujeción a la ley. "PAE" se reserva las acciones legales pertinentes.

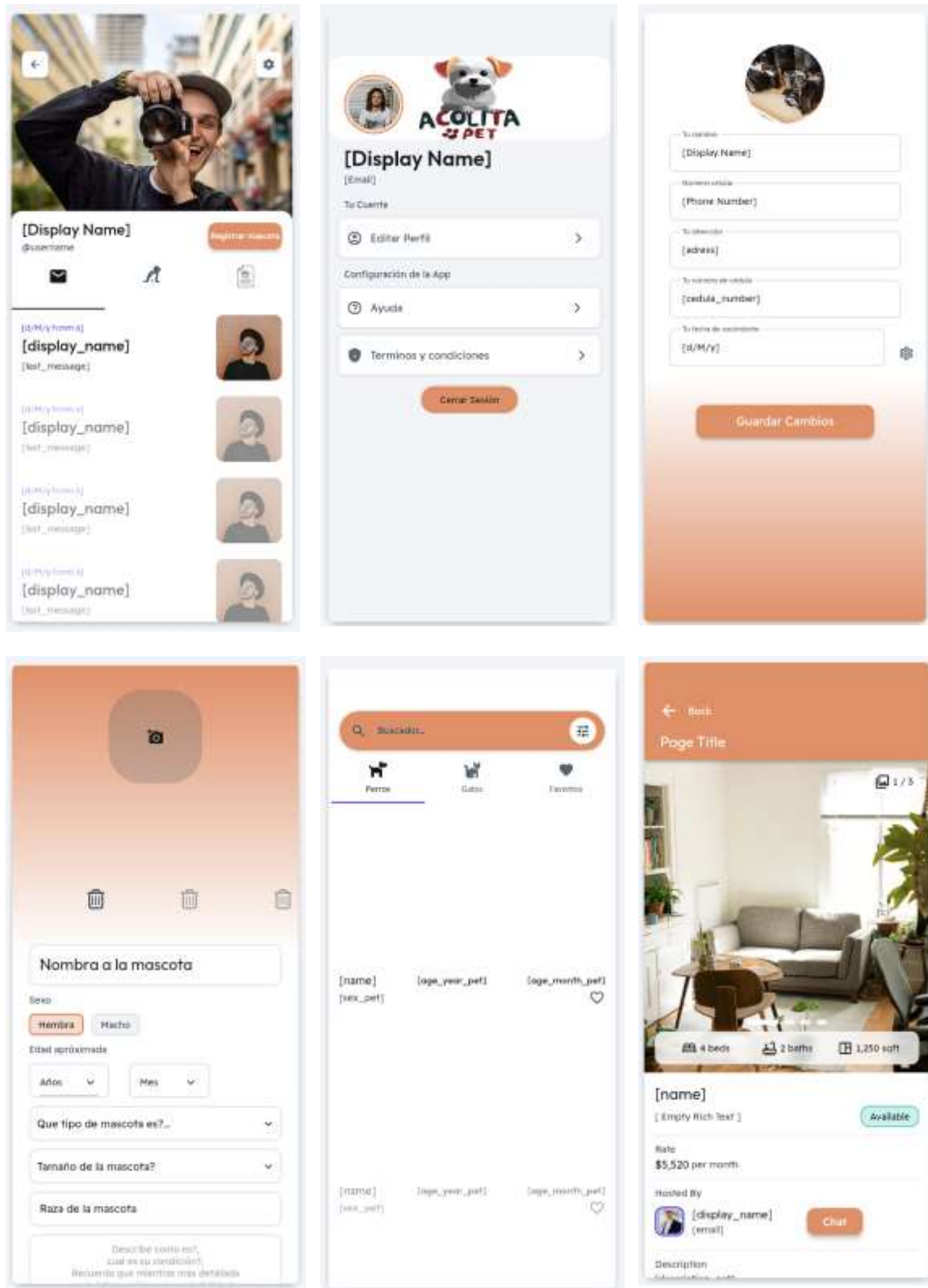
Para constancia de lo estipulado en el presente instrumento las partes firman en unanimidad de acto y en dos ejemplares.

Silvia Ramos Samaniego
COORDINADORA PAE CAPITULO RIOBAMBA

Firma del Adoptante
C.I.:

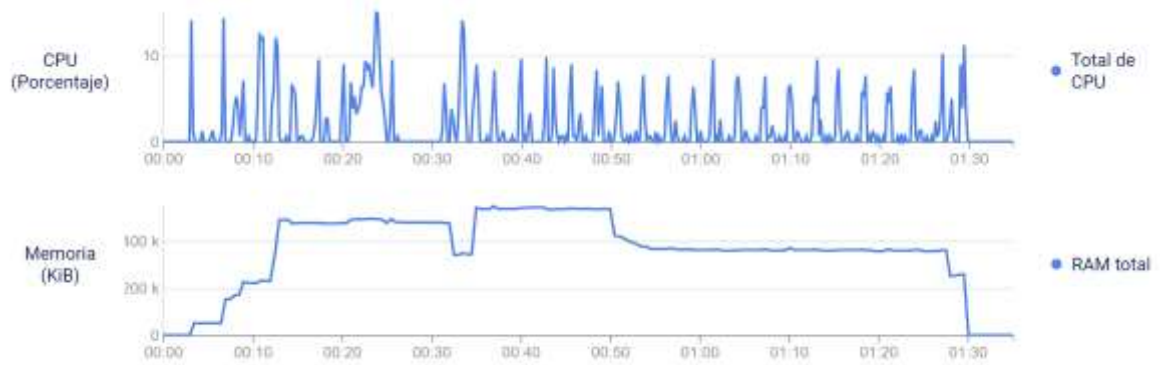
ANEXO 5

Interfaces graficas de la aplicación móvil

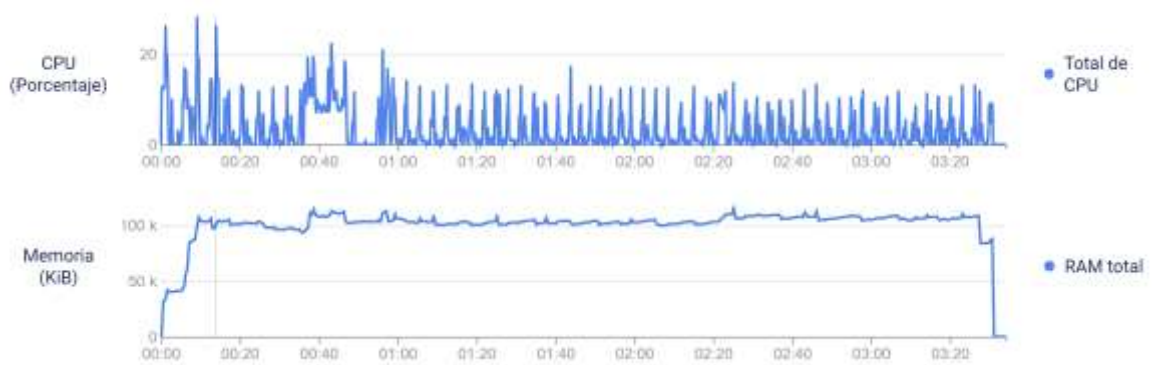


ANEXO 6

Prueba Robo, Galaxy Z Fold5, nivel de API 34 (Gama Alta)



Prueba Robo, Galaxy A10, nivel de API 29 (Gama media)



ANEXO 7

Presentación de la aplicación móvil a la Dra. Silvia Ramos, organizadora de PAE-Riobamba

