UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE SOFTWARE

Tema:

APLICACIÓN MÓVIL UTILIZANDO REALIDAD AUMENTADA PARA AGILIZAR EL TIEMPO DEL PROCESO DE ADOPCIÓN DE MASCOTAS EN LA "FUNDACIÓN PAQUITO"

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Software

ÁREA: Software

LÍNEA DE INVESTIGACION: Desarrollo de Software

AUTOR: Kevin Jesús Suárez González

TUTOR: Ing. Santiago David Jara Moya, Mg.

Ambato - Ecuador

febrero - 2025

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: APLICACIÓN MÓVIL UTILIZANDO REALIDAD AUMENTADA PARA AGILIZAR EL TIEMPO DEL PROCESO DE ADOPCIÓN DE MASCOTAS EN LA "FUNDACIÓN PAQUITO", desarrollado bajo la modalidad Proyecto de Investigación, por el señor Kevin Jesús Suárez González, estudiante de la Carrera de Software, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, febrero 2025.

SANTIAGO DAVID JARA MOYA

Firmado digitalmente por SANTIAGO DAVID JARA MOYA
Fecha: 2025.02.05 16:06:08 -05'00'

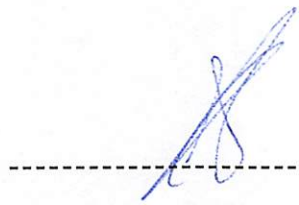
Ing. Santiago David Jara Moya, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: APLICACIÓN MÓVIL UTILIZANDO REALIDAD AUMENTADA PARA AGILIZAR EL TIEMPO DEL PROCESO DE ADOPCIÓN DE MASCOTAS EN LA "FUNDACIÓN PAQUITO" es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, febrero 2025.



Kevin Jesús Suárez González

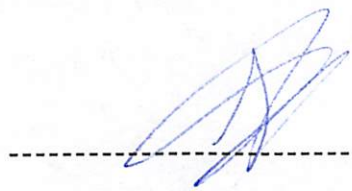
C.C. 2000114724

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, febrero 2025.



Kevin Jesús Suárez González

C.C. 2000114724

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Kevin Jesús Suárez González, estudiante de la Carrera de Software, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado APLICACIÓN MÓVIL UTILIZANDO REALIDAD AUMENTADA PARA AGILIZAR EL TIEMPO DEL PROCESO DE ADOPCIÓN DE MASCOTAS EN LA "FUNDACIÓN PAQUITO" nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con el señor Presidente del Tribunal.

Ambato, febrero 2025.

Ing. Franklin Oswaldo Mayorga Mayorga, Mg.
PRESIDENTE DEL TRIBUNAL

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PROFESOR CALIFICADOR

Ing. Carlos Israel Nuñez Miranda, Mg.
PROFESOR CALIFICADOR

DEDICATORIA

A mi padre y a mi madre, por ser mi pilar fundamental y por su sacrificio, esfuerzo y amor.
A mis hermanos, por el apoyo y la inspiración para alcanzar mis metas.
A mis amigos, por la compañía y apoyo emocional.

Kevin Jesús Suárez González

AGRADECIMIENTOS

En primer lugar, agradezco a Dios, por darme la fortaleza y la sabiduría necesaria para culminar esta etapa de mi vida.

A mi padre y a mi madre, por su apoyo incondicional y sus consejos.

A mis hermanos, por su motivación para enfrentar los desafíos.

A mis amigos, quienes siempre estuvieron para apoyarme y me ayudaron en las dificultades.

A mi tutor, por su paciencia y valiosas aportaciones durante el desarrollo de este trabajo.

A todos los docentes, quienes compartieron sus conocimientos y experiencias.

Finalmente, a la Fundación Paquito, por darme la oportunidad de contribuir a una causa tan signifivativa.

Kevin Jesús Suárez González

ÍNDICE GENERAL DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN EJECUTIVO

El refugio de animales ”Fundación Paquito” es una organización sin fines de lucro que tiene como propósito brindar atención y cuidado a animales de compañía en situación de abandono. Sin embargo, enfrenta diversos desafíos relacionados con la gestión de la información de los animales y del proceso de adopción.

El presente proyecto propone la implementación de una aplicación móvil de realidad aumentada con el objetivo de agilizar el proceso de adopción. La app permite a los usuarios visualizar, a través de la cámara de su dispositivo móvil, un modelo 3D representativo del animal, evitando así la necesidad de desplazarse físicamente hasta la Fundación. Además, la aplicación proporciona un perfil detallado de cada animal, lo que permite a los usuarios conocer mejor sus características y reforzar su decisión de adoptar. Esto representa un beneficio significativo para la Fundación, ya que contribuye a disminuir las devoluciones de animales adoptados.

La aplicación móvil no solo facilita la adopción, sino que también gestiona el ingreso de animales al refugio, así como el registro de desparasitantes y vacunas administrados. Para su desarrollo se utilizaron diversas tecnologías: la interfaz de programación de aplicaciones (API) fue implementada con Spring Boot y PostgreSQL como gestor de base de datos, mientras que la interfaz de usuario fue desarrollada con Flutter. Por su parte, la funcionalidad de realidad aumentada fue construida utilizando el editor de Unity.

Como resultado, se logró desarrollar una aplicación de realidad aumentada que reduce significativamente el tiempo requerido para el proceso de adopción, ya que elimina la necesidad de que los adoptantes se desplacen hasta la Fundación para conocer a los animales.

Palabras clave: Aplicación móvil, Flutter, Spring Boot, Unity, ARCore, realidad aumentada.

ABSTRACT

The animal shelter ”Fundación Paquito” is a non-profit organization that aims to provide care and attention to abandoned companion animals. However, it faces various challenges related to the management of animal information and the adoption process.

This project proposes the implementation of an augmented reality mobile application with the aim of streamlining the adoption process. The app allows users to view, through the camera of their mobile device, a representative 3D model of the animal, thus avoiding the need to physically travel to the Foundation. In addition, the application provides a detailed profile of each animal, allowing users to better understand its characteristics and reinforce their decision to adopt. This represents a significant benefit for the Foundation, as it contributes to reducing the returns of adopted animals.

The mobile application not only facilitates adoption, but also manages the entry of animals to the shelter, as well as the record of dewormers and vaccines administered. Various technologies were used for its development: the application programming interface (API) was implemented with Spring Boot and PostgreSQL as the database manager, while the user interface was developed with Flutter. The augmented reality functionality was built using the Unity editor.

As a result, an augmented reality application was developed that significantly reduces the time required for the adoption process, since it eliminates the need for adopters to travel to the Foundation to meet the animals.

Keywords: Mobile App, Flutter, Spring Boot, Unity, ARCore, augmented reality.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

APLICACIÓN MÓVIL UTILIZANDO REALIDAD AUMENTADA PARA AGILIZAR EL TIEMPO DEL PROCESO DE ADOPCIÓN DE MASCOTAS EN LA "FUNDACIÓN PAQUITO"

1.1.1 Planteamiento del problema

Un estudio realizado en Malasisa revela como los ayuntamientos contratan cazadores privados de perros [8]. Los cazadores tienen la potestad de capturar a los animales callejeros, y también de sacrificarlos. Como resultado, los cazadores se destacan en su labor debido a que el sueldo que reciben depende de la cantidad de animales capturados y sacrificados. Por tal motivo, muchos refugios o albergues fueron construidos por organizaciones privadas, porque la ley de Malasia no es estricta para proteger el bienestar de los animales callejeros. En los refugios los animales reciben los servicios necesarios para vivir, tal como comida y agua. Los animales que llegan a estos lugares es de aproximademente 7.3 millones al año, de los cuales 3.9 millones son perros y 3.4 millones son gatos. Como consecuencia, cada año se sacrifican 2.7 millones. De manera similar, el estudio [9] realizado en España estimaron que 138.307 perros y gatos fueron abandonados en el 2018, 17 de cada 1000 perros y 10 de cada 1000 gatos terminan en un refugio. De igual manera, estudios realizados en [10] por la Universidad de La Salle en Colombia revelaron que pueden existir más de un millón de perros callejeros y solo en Bogotá se estimán mas de 150 mil.

Por tanto, mediante el uso de herramientas tecnológicas se busca controlar y disminuir la sobrepoblación de mascotas abandonadas, para mitigar problemas a futuro. El efecto de la sobrepoblación de mascotas se evidencia en un estudio realizado en Filipinas [11], donde revela acerca de los problemas comunes que se presentan en los albergues o refugios, tales como el hacinamiento, los recursos limitados, la dotación de personal inadecuado, la formación de voluntarios y la alta rotación de personal. Entre otros casos, debido al hacinamiento de los refugios varios animales son abandonados

al no recibir un cuidado adecuado. Además, el proceso de adopción llevado por los centros de refugios son ineficientes, por cuanto no tienen una organización adecuada de los perfiles de mascotas. Por lo tanto, la información primordial sobre los animales es muy poca o nula.

En ciertos paises de latinoamérica como Perú y Colombia mostraron preocupación por el maltrato y abandono de mascotas. Los estudiantes de la Universidad César Vallejo desarrollaron un Sistema Web y Móvil llamado "ADOPTAPE" para el Albergue AVA, en Perú [12]. A partir del funcionamiento y operatividad de los sistemas se redujo el tiempo de proceso de adopción, así mismo, aumentó el número de adopciones en el albergue y el nivel de satisfacción de los voluntarios. Como resultado, mediante el uso del sistema propuesto se redujo a 4 días el tiempo de adopción, por cuanto hubo un decremento de 7 días.

En Ecuador no existen cifras oficiales de animales callejeros al igual que en el cantón Ambato. Sin embargo, solo en la ciudad de Quito, mediante un censo poblacional [13] realizado por la Universidad San Francisco de Quito se identifican las siguientes cifras: en 2013 por cada 59 personas 1 perro abandonado, en el 2014 por cada 49 personas 1 perro abandonado y en 2018 por cada 22 personas 1 perro abandonado. Por tanto, identificarón que en 246.3 km es posible descubrir 4.070 perros abandonados. De igual manera, se identifican las 3 principales parroquias con mayor densidad de perros en la calle que son: El Inca, La Ecuatoriana y Puengasí [14]. Además, solo en la parroquia de Guamaní, se reportaron 169 casos de personas atendidas por mordeduras de perros callejeros en el año 2021.

1.2 Antecedentes investigativos

Una revisión bibliográfica de repositorios digitales y sitios web académicos reveló una variedad de fuentes que podrían usarse como antecedente para este proyecto.

En [10] se desarrolló una aplicación móvil para adopción de mascotas abandonadas llamada "Peluditos.com" en Colombia. Mediante el uso de la app buscaron reducir la cantidad de mascotas abandonadas. Así mimso, fomentaron una cultura responsable en

la tenencia de mascotas. Utilizaron una metodología cuantitativa y como instrumento de recolección de información un cuestionario. El cuestionario se aplicó a una muestra de 384 personas mayores de edad. La aplicación se desarrolló solo para el sistema operativo de Android con el lenguaje de Java, y para el gestor de base de datos se empleó MySQL. La metodología utilizada fue una metodología ágil llamada SCRUM. Entre las funcionalidades del sistema están el registro de Fundaciones o Albergues, ya sean estas públicas o privadas, hoja de vida de la mascota, información del adoptante, el proceso de adopción, apadrinamiento a una mascota, donaciones, denuncias e información de mascotas extraviadas.

En [15] se implementó una aplicación móvil para la gestión de una asociacón de búsqueda y adopción de mascotas llamada “Wallamascotas” en España. La asociación “Wallamascotas” trató de reunir en un solo sitio todos los animales de diferentes asociaciones para tener un mayor alcance hacia el público. Además, ya contaba con un portal web donde permitía acceder a toda la información para la adopción de una mascota. Por tanto, desarrollaron la aplicación utilizando un Framework que permitía la creación de apps multiplataformas para sistemas operativos iOS (del inglés, iPhone Operating System) y Android como IONIC. También, utilizaron el Framework Corodova para acceder a los elementos físicos de un dispositos y generar las comunicaciones con las interfaces. En el manejo de los datos hicieron uso de la base de datos de “Wallamascotas”, el cual empleaba SQL Server. El desarrollo se llevó a acabo bajo la metodología SCRUM. Entre las funcionalidades, permite consultar todas las mascotas pertenencientes a diferentes asociaciones, el perfil completo de la mascota y los medios de contacto. También es posible filtrar por asociación y suscribirse para obtener notificaciones por cada mascota adoptada.

En [16] se elaboró una propuesta de implementación de un sistema web y móvil para una asociación de protección animal llamada “Salvando Vidas” en Cajamarca, Perú. Se trabajó con una muestra de 73 participantes. En la recolección de información se empleó el intrumento del cuestionario. Por otro lado, en el desarrollo del sistema web utilizaron PHP (del inglés, Hypertext Preprocessor) con el Framework Laravel, y para la app, Java con la plataforma Android Studio. En cuanto al gestor de base de datos se empleó MySQL, y como metodología de desarrollo se usó OpenUp. En la arquitectura optaron por el modelo MVC (Model-Vista-Controlador), tanto para

el sistema web y móvil. Entre las funcionalidades, es posible registrar mascotas, adoptar mascotas, ver el perfil de la mascota, denunciar casos de maltrato animal y un apartado sobre consejos para la tenencia responsable. Por otra parte, como resultado del estudio determinaron la existencia de una influencia positiva en el uso de una app y la adopción de mascotas, debido al incremento en el número de registros de adopciones exitosas con un 59%.

En [17] se planteó el desarrollo de un sistema web y una aplicación móvil para ubicar y rescatar animales domésticos extraviados o abandonados para la "Fundación Rescate Animal Ecuador". Se trabajó con una muestra de 70 participantes, y como instrumento de recolección de información se usó el cuestionario. En cuanto al desarrollo del sitema web usaron PHP con el Framework Laravel, y para la app, Java con la plataforma de Android Studio. Acerca del gestor de la base de datos se manejó con MySQL. Referente a la metodología de desarrollo utilizada fue SCRUM. Entre las funcionalidades, el sistema tiene la capacidad de registrar usuarios, reportar mascotas abandonadas o extraviadas enviando la ubicación en tiempo real, generar reportes o informes sobre las mascotas rescatadas y el proceso de adopción de mascotas.

En [9] se llevó a cabo el desarrollo de una plataforma web para la adopción y gestión de animales procedentes de protectoras en España. Acerca de las tecnologías usadas para el desarrollo del sistema web se empleó React y Redux en el lado del frontend, y para generar la API (del inglés, Application Programming Interface) se ocupó NodeJs con ExpressJs. En cuanto al gestor de base de datos se usó MongoDB. Respecto a la metodología de desarrollo para el sistema se usó SCRUM. Entre las funcionalidades, se puede registrar usuarios, registrar mascotas, visualizar el perfil de las mascotas, buscar todos los animales por una protectora y el proceso de adopción.

En [18] se desarrolló una aplicación móvil para la tenencia responsable de mascotas llamada "SMILEYPET" para la asociación "MI PEQUEÑO HERMANO" en Perú. Se trabajó con una muestra de 67 socios y como instrumento de recolección de datos se usó el cuestionario. En el desarrollo de la aplicación utilizaron el lenguaje Java junto con la plataforma de Android Studio. Por el lado del gestor de la base de datos se empleó Cloud Firestore, y referente a la metodología de desarrollo de software de usó OpenUp. La aplicación consistía en una herramienta de soporte para la tenencia

responsable de mascotas, mediante el cumplimiento de ciertos parámetros de cuidado, como un plan de vacunación, un plan de desparasitación, la frecuencia de baño, la frecuencia de limpieza de dientes, la frecuencia de limpieza de orejas, la frecuencia de cepillado de pelo, la frecuencia de corte de uñas y la frecuencia de paseos. Cada actividad cumplida se registraba en la aplicación y a través de notificaciones al usuario, le recordaba que tarea debía cumplir. Como resultado, lograron determinar la fiabilidad de la aplicación usando las métricas seleccionadas de la norma ISO (del inglés, International Organization for Standardization) 9126, donde se obtuvo un valor de 91%.

En [19] se implementó el desarrollo de un sistema web de gestión sectorizado de adopción y rescate de mascotas para la "Fundación Camino a Casa" en Quito. No se trabajó con una muestra, por lo tanto, no se aplicó un instrumento de recolección de datos. Respecto a las tecnologías utilizadas se encuentra el lenguaje Ruby con el framework Ruby on Rails, además de HTML (del inglés, Hypertext Markup Language), CSS (del inglés, Cascading Style Sheets) y JavaScript. En cuanto a la metodología de desarrollo de software se ocupó SCRUM, y como gestor de base de datos PostgreSQL. Entre las funcionalidades, permite registrar usuarios, registrar mascotas, publicar y visualizar los perfiles de mascota, realizar búsquedas por sector y compartir los anuncios de adopción en las redes sociales.

En [20] se implementó una aplicación web para fomentar la adopción de mascotas llamada "YOUCAN" en España. No se tuvo una muestra con la cual trabajar, por lo cual, no se utilizó un instrumento de recolección de datos. Por otra parte, las herramientas que utilizaron para el desarrollo fueron HTML, CSS JavaScript y PHP. Además, se apoyó en Bootstrap como librería de componentes. En cuanto al gestor de base de datos se empleó MySQL. Entre las funcionalidades, permite registrar usuarios, registrar mascotas, realizar búsquedas mediante filtros como el sexo, raza, ciudad o tamaño y publicar los perfiles de mascotas. También cuenta con un módulo de mensajería para la comunicación con los albergues.

En [12] se llevó a cabo el desarrollo de un sistema web y móvil para facilitar la adopción de mascotas en el Albergue "Asociación de Voluntarios Animalistas" en Perú. Se trabajó bajo un enfoquede investigación cuantitativo y como población muestral se tuvo

a 15 personas integrantes del albergue. Respecto a las tecnologías para el desarrollo y el gestor de base de datos no son mencionadas. En cuanto a la metodología de desarrollo de software utilizada fue XP (del inglés, Extreme Programming). Entre las funcionalidades, permite el registro de usuarios, el registro de mascotas, publicar los perfiles de las mascotas y el proceso a seguir para la adopción. Por consecuente, se obtuvieron resultados positivos, al concluir un decremento de 7 días para concretar la adopción, pues de los 13 días que se tardaba, ahora solo se requiere de 4 días.

1.3 Fundamentación teórica

1.3.1 Dart

Es un lenguaje de programación desarrollado por Google que implementa la integración con Web Components, por lo cual permite escribir diferentes aplicaciones. En [21] el autor indica que es uno de los lenguajes de programación con mayor crecimiento en los últimos años y con una amplia comunidad de desarrolladores. Por otro lado, al ser un lenguaje de programación orientado a objetos posibilita manejar procesos de desarrollo más cómodos y rápidos. Entre las ventajas se encontró que es un lenguaje optimizado para aplicaciones en múltiples plataformas, por cuanto puede ser usado para distintos propósitos como desarrollo móvil, desktop, backend y aplicaciones web. Otra ventaja destacada es que puede ser compilado a JavaScript. Por todo lo anterior, es un lenguaje bastante rápido y su rendimiento es muy alto.

1.3.2 Flutter

Es un framework de código abierto desarrollado por Google para crear aplicaciones nativas tanto para Android, iOS y Web. En [21] el autor señala el incremento de su popularidad de forma exponencial, apesar del corto tiempo de lanzamiento que tiene Flutter. A su vez, el tipo de compilación utilizado es AOT (del inglés, Ahead Of Time) en lugar de JIT (del inglés, Just In Time), siendo el primero, más rápido al momento de compilar. Además, permite elaborar de manera sencilla interfaces atractivas a la vista de los ususarios gracias al manejo de widgets. Por otro lado, el director senior de tecnología en Google (Pavan Podilla), afirmó que Flutter ha traído varios cambios para

dar mejores soluciones, debido al uso de un solo lenguaje de código base, sin perder muchas funciones entre plataformas.

1.3.3 Java

Es un lenguaje de programación orientado a objetos creado por Sun Microsystems en 1991. En [22] el autor menciona la capacidad que posee este lenguaje, ya que puede ejecutarse en cualquier contexto y ambiente, siendo la portabilidad uno de sus principales logros.

1.3.4 Java JDK

JDK representa las siglas en inglés de ”Java Development Kit”, es un Kit de desarrollo de Java. En [17] los autores señalan las características de dicha herramienta, como el hecho de incluir un JRE (del inglés, Java Runtime Environment) completo, además, herramientas para desarrollar, depurar y monitorear aplicaciones Java. También, posee una colección de herramientas, utilidades, documentación y ejemplos que facilitan el uso para los desarrolladores al crear aplicaciones java. Por otro lado, es posible crear applets y componentes.

1.3.5 Android Studio

Es el IDE (del inglés, Integrated Development Environment) oficial para la creación de apps del sistema oprativo Android. En [16, 17, 23] los autores denotan que cuenta con una interfaz muy intuitiva y práctica, por lo cual facilita el trabajo del desarrollo del código XML (del inglés, Extensible Markup Language). Entre otras características se tiene, que posee una estructura simple y ordenada; es intuitivo, sencillo y fácil de usar; permite visualizar el aspecto del diseño de la aplicación durante el desarrollo; es posible utilizar emuladores para verificar el funcionamiento a detalle de las aplicaciones durante el desarrollo, y la compatibilidad con sistemas operativos como Linux, Mac y Windows. Además, cuenta con un emulador de funciones, una integración de marcos de desarrollo con bajos costos y la facilidad de ejecutar pruebas.

1.3.6 IONIC Framework

Es una plataforma SDK (Software Development Kit) de frontend de código abierto, basado en tecnologías web (HTML, CSS y JavaScript) para el desarrollo de aplicaciones móviles híbridas. En [15] el autor señala ciertas características de la herramienta, como la posibilidad de trabajar con cualquier marco de desarrollo enfocado en la vista al igual que lo hace React, Angular o Vue. Además, cuenta con la flexibilidad de usar los componentes de Ionic sin necesidad de instalar algún marco de interfaz de usuario. Por consecuente, Ionic ofrece componentes web propios con los que el desarrollador puede interactuar por medio de sus métodos, tales como el componente Ion Virtual Scroll el cual dibuja un scroll sobre una lista de miles de elementos sin presentar una pérdida de rendimiento. Por otro lado, utiliza la aceleración de hardware nativa disponible en el navegador, y optimiza el rendimiento de las vistas al eludir la manipulación excesiva del DOM (del inglés, Document Object Model). Además, minimiza el tiempo de proceso de las transiciones y transformaciones CSS, de tal manera que aligera el GPU (del inglés, Graphics Processing Unit).

1.3.7 Apache Cordova

Es un framework de código abierto para el desarrollo de aplicaciones móviles. En [15, 24] los autores señalan acerca de las funcionalidades que tiene Cordova como marco de desarrollo, al emplear tecnologías web estándar como HTML, CSS y JavaScript. En consecuencia, se tienen aplicaciones multiplataformas, evitando la codificación para cada plataforma móvil. Además, proporciona varios plugins que sirven como interfaces de comunicación con los diferentes componentes nativos. Entre los más usados están: la cámara, contacts, device, battery, device-motion, device-orientation y file.

1.3.8 Base de Datos Relacional

Es un tipo de base de datos que organiza los datos en filas y columnas, que en conjunto forman una tabla. Los datos se estructuran en varias tablas, que se pueden unir a través de una clave principal o una clave externa. Estos identificadores únicos demuestran las diferentes relaciones que existen entre las tablas.

- **MySQL:** En [16, 17, 20, 22, 25–27] los autores señalan la importancia de poder ser ejecutado en más de un solo sistema operativo como Windows, Linux y Unix. Utiliza (del inglés, Relational Database Management System) para crear y administrar bases de datos. Además, destaca en el mercado gracias a su rendimiento probado, a su fiabilidad y facilidad de uso.

- **PostgreSQL:** En [19, 28] los autores denotan que se trata de una herramienta de libre acceso y flexible configuración. Además, se ejecuta en los principales sistemas opertivos como Linux, UNIX, Mac, Solaris y Windosw. Entre sus características está que es un base de datos 100% ACID (del inglés, Atomicity, Consistency, Isolation, Durability), posee integridad referencial, transacciones anidadas, copias de seguridad en caliente, admite caracteres internacionales, acceso SSL (del inglés, Secure Sockets Layer) con encriptado y una amplia documentación.

1.3.9 Base de Datos No Relacional

Es un sistema de almacenamiento de información que se caracteriza por no usar el lenguaje SQL (del inglés, Structured Query Language) para las consultas. Esto no significa que no puedan usar el lenguaje SQL, pero no lo hacen como herramienta de consulta, sino como apoyo. Otra de sus principales características es que no trabajan con estructuras definidas. Es decir, los datos no se almacenan en tablas, y la información tampoco se organiza en registros o campos.

- **MongoDB:** En [9] el autor señala que se trata de una base de datos multiplataforma, de código abierto, donde permite tener un esquema dinámico al almacenar la información como documentos. Entre su principal ventaja está su velocidad, al ofrecer un balance entre rendimiento y funcionalidad.

- **Firebase:** En [12, 21, 23, 29, 30] los autores indican que Firebase es una plataforma de desarrollo web y móvil que proporciona una amplia gama de servicios en la nube, que facilitan el desarrollo de aplicaciones. Entre sus características mencionadas están: incluyen una base de datos en tiempo real que permite la sincronización instantánea de datos entre clientes, autenticación segura de usuarios, alojamiento web, mensajería en la nube, funcionalidad sin

servidor para ejecutar código en respuesta a eventos y herramientas integrales de análisis y prueba. Firebase simplifica el desarrollo al proporcionar una infraestructura administrada y escalable para que los desarrolladores puedan concentrarse en crear experiencias de usuario ricas y funcionales.

1.3.10 PHP y Laravel

PHP es un lenguaje de código abierto adecuado para el desarrollo web y que puede ser incrustado en HTML. En [16, 17, 20, 22, 25] los autores denotan que se trata de un lenguaje de programación que se procesa en un servidor y es especialmente utilizado para crear páginas web dinámicas, por cuanto da facilidad de acceso a diferentes bases de datos. Entre las características a destacar está la velocidad al momento de la ejecución de los programas sin crear retraso en la máquina; la estabilidad por contar con su propio sistema de administración de recursos; la seguridad al permitir al usuario configurar sus propias restricciones; y la simplicidad que le da al usuario para poder de realizar múltiples tareas al momento de crear algún proyecto.

Laravel es un framework de código abierto para el desarrollo de aplicaciones web con PHP. En [16, 17] los autores indican que se trata de un framework funcional creado para el desarrollo de servicios web y aplicativos móviles que facilitan a los desarrolladores la forma de integrar varias funcionalidades, con una sintaxis sencilla y refinada reduciendo el tiempo de la elaboración de los proyectos. Además, se basa en el patrón MVC y tiene una sintaxis elegante y simple.

1.3.11 TypeScript y Angular

TypeScript es un lenguaje de programación de código abierto, es decir, de acceso y distribución libre, pensado para ejecutar grandes proyectos y que permite trabajar de manera estructurada. En [24] el autor indica la faciliidad de definir la lógica, efectos de animación y comunicación con servicios y otros componentes del sistema web y móvil.

Angular es una plataforma y un framework para crear aplicaciones de una sola página en el lado del cliente usando HTML y TypeScript. En [24, 27, 29] los autores señalan

que se caracteriza por seguir una arquitectura basada en componentes. Además, emplea una adaptación de MVC y un modelo SPA (del inglés, Single Page Application), lo que permite a las páginas enviadas al navegador ser recargadas durante su uso. Por consecuente, existe un tiempo de respuesta mucho más rápido en comparación con las aplicaciones web tradicionales.

1.3.12 HTML, CSS y JavaScript

HTML es el lenguaje con el que se define el contenido de las páginas web. En [19,22, 24] los autores denotan la utilidad al momento de establecer la estructura y el contenido de una vista, ya sea de texto, objetos e imágenes. Por lo tanto, permitió definir la estructura para cada una de las vistas y organizar la información y el contenido mediante el manejo de etiquetas.

CSS corresponde a la expresión inglesa Cascading StyleSheets, que puede traducirse como "Hojas de estilo en cascada" el cual es un lenguaje empleado en el diseño gráfico. En [19, 22, 24] los autores expresan el provecho de como pudieron describir la presentación de los documentos estructurados mediante reglas de estilos. Por tanto, definieron el estilo visual para cada una de las vistas, con el objetivo de crear diseños adaptables a cualquier dispositivo y tamaño de pantalla.

JavaScript es un lenguaje de programación o de secuencias de comandos que permite implementar funciones complejas en páginas web. En [19, 20, 24, 25] los autores declaran acerca de como establecieron las acciones que se van a ejecutar dentro de la aplicación web. Además, la utilidad para que el contenido sea dinámico e intuitivo para el usuario, debido a su flexibilidad, sencillez y potencia.

1.3.13 Bootstrap

Es un framework de desarrollo web gratuito y de código abierto, diseñado para facilitar el proceso de desarrollo de los sitios web responsivos y orientados a los dispositivos móviles, al proporcionar una colección de sintaxis para diseños de plantillas. En [19, 20, 24, 25] los autores señalan la capacidad que tiene el framework, pues al estar basado en librerías CSS y jQuery permite desarrollar interfaces HTML atractivas y

responsive. Por consecuente, representa una experiencia de usuario considerable, debido al desarrollo de sitios web que se ajustan al tamaño de la pantalla de cualquier dispositivo.

1.3.14 Python

Es un lenguaje de programación orientado a objetos de alto nivel y fácil de interpretar con sintaxis fácil de leer, tiene un amplio uso en computación científica, desarrollo web y automatización. En [25] el autor indica el fascinante uso del reconocimiento de imágenes para la búsqueda de mascotas extraviadas mediante la visión artificial, ayudado por el uso de histogramas para la clasificación de pixeles mediante la aplicación del algoritmo de K-Medias. Esta funcionalidad se apoyó en la librería llamada OpenCV (del inglés, Open Computer Vision) para la clasificación de imágenes.

1.3.15 C#

Es un lenguaje de programación orientado a componentes y orientado a objetos, desarrollado por Microsoft y diseñado para compilar diversas aplicaciones que se ejecutan en .NET Framework. En [27] el autor expresa el manejo fácil por su sintaxis básica derivada de C/C++ y hace uso del modelo de objetos de la plataforma .NET. Además, que incluye mejoras de otros lenguajes.

1.3.16 React

Es una librería JavaScript de código abierto que ayuda a crear interfaces de usuario de manera sencilla. En [25] el autor indica como la creación de interfaces es muy flexible debido a la arquitectura basada en componentes. Por consecuente, esta arquitectura es fácil de mantener y muy reutilizable al tener como resultado un proceso de desarrollo acelerado. Por otro lado, posee un Virtual Data Object Model (DOM) que le permite crear más de 200,000 nodos por segundo, lo cual agilita cada cambio. Además, incorpora JSX (del inglés, JavaScript XML), el cual es una extensión de JavaScript para escribir código HTML como si fuera código JavaScript.

1.3.17 SCRUM

Es un marco de trabajo o metodología ágil para la gestión de proyectos, especialmente de desarrollo de software. En [10, 15, 17, 22, 24, 28] el autor indica los motivos por los cuales usó ésta metodología, entre los cuales destaca por el tipo de proyecto escalable, las variaciones y cambios de los requisitos. Además, por el acoplamiento para el equipo de trabajo multidisciplinarios y multinivel. Entre otros beneficios, se encuentra la obtención de un producto funcional desde las primeras fases, el feedback del cliente de manera rápida, la gestión de los cambios de manera flexible, la reducción de riesgos a largo plazo, y conocimiento del progreso real del proyecto desde el inicio. Por otro lado, la organización del proyecto a través del uso de Sprints ayuda a gestionar los errores encontrados y rectificar las prioridades del proyecto en cualquier momento.

1.3.18 Kanban

Es una manera de ayudar a los equipos a ubicar un equilibrio entre el trabajo que necesitan hacer y la disponibilidad de cada miembro del equipo. Además, se basa en una filosofía centrada en la mejora continua, donde las tareas se extraen de una lista de actividades pendientes en un flujo de trabajo constante.

1.3.19 Metodología XP

Es una metodología ágil de gestión de proyectos centrada en la velocidad y la simplicidad con ciclos de desarrollo cortos. En [26] el autor menciona como la metodología cumple con una entrega de software de manera más rápida, lo cual permite alcanzar las necesidades del negocio. Además, entre las fases que la componen para ejecutar la metodología está: la planeación, el diseño, la codificación y la creación de pruebas a aplicar al sistema.

1.3.20 Metodología en Cascada

Es un enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la culminación de la etapa anterior. En [21,29] los autores señalan como permite estipular los requerimientos, plantear el diseño del software, la implementación, la verificación y el mantenimiento del sistema. Por otro lado, como desventaja, se tiene la obligación de que los requisitos se encuentren bien definidos desde el principio.

Al finalizar la revisión de varios artículos y tesis, se resalta el uso de PHP y Laravel para la creación de aplicaciones web acompañado de Bootstrap como librería de componentes. Además, el uso de JavaScript y CSS no se queda de lado debido a la interactividad que aporta a PHP y Laravel. Por otro lado, el uso del framework Angular surge como una tecnología atractiva y moderna a diferencia de Laravel, debido al enfoque SPA que presenta como resultado una respuesta mucho más rápido en comparación con las aplicaciones web tradicionales. Por el lado de las aplicaciones móviles, tienen mayor preferencia por el desarrollo para sistemas operativos Android con el lenguaje de Java. En cuanto para el manejo de la base de datos se utiliza en mayor medida MySQL. Para finalizar, la metodología de desarrollo más utilizada es SCRUM debido a la flexibilidad y adaptación con las variaciones que se presentan en medio del desarrollo.

1.4 Objetivos

1.4.1 Objetivo general

Implementar una aplicación móvil utilizando Realidad Aumentada para agilizar el tiempo de proceso de adopción de mascotas en la “Fundación Paquito”.

1.4.2 Objetivos específicos

- Investigar herramientas y procesos de implementación para el uso de Realidad Aumentada.

- Analizar los procesos para agilizar el tiempo de adopción de mascotas de la Fundación Paquito.
- Desarrollar una aplicación móvil que permita gestionar el proceso de adopción de mascotas utilizando Realidad Aumentada para la Fundación Paquito en Ambato.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para el presente proyecto, con el propósito de conocer los procesos de la Fundación Paquito y el proceder de los potenciales adoptantes, se aplicaron dos entrevistas y una encuesta. La primera entrevista fue dirigida al fundador y propietario de la Fundación, junto con el gerente. La segunda entrevista fue para el personal de la Fundación, con el enfoque de conocer y analizar la perspectiva que tienen acerca de los procesos para el ingreso de nuevos animales, así como el proceso de solicitud de adopción.

Además, se llevó a cabo una encuesta dirigida a personas con el interes de adoptar animales de compañía, con el fin de analizar la perspectiva acerca del proceso de adopción.

2.2 Métodos

2.2.1 Modalidad de Investigación

a. Investigación de campo

La investigación es de campo debido a que la recopilación de datos referente al problema necesitan ser extraidos directamente del entorno de los hechos, es decir, de la Fundación Paquito.

b. Investigación bibliográfica

La investigación es bibliográfica o documental por la necesidad de realizar una revisión de material documental respecto al tema a estudiar, como artículos académicos, tesis y libros. Los cuales se encargarán de alimentar la propuesta a desarrollar.

c. Investigación aplicada

La investigación es aplicada porque busca la aplicación o utilización de conocimientos desde un área especializada, para satisfacer una necesidad en concreto, mediante el desarrollo de la propuesta.

d. Enfoque de la investigación

El presente estudio está diseñado para seguir un enfoque metodológico cuantitativo, debido a la adaptación que presenta referente a las características y necesidades del estudio. Además, por cuanto consiste en la recolección y análisis de datos numéricos, el cual se lleva a cabo mediante el uso de herramientas estadísticas y matemáticas con el propósito de cuatificar el problema de estudio.

2.2.2 Población y muestra

La población de estudio se conformó por las personas de la Fundación Paquito y por las personas que se consideraron potenciales adoptantes. En el caso de los potenciales adoptantes, se consideró una población desconocida. Por tanto se extrajo la muestra mediante la siguiente ecuación:

$$n = \frac{Z^2 \cdot p \cdot q}{e^2} \tag{1}$$

Donde:

- n = tamaño de muestra buscado
- Z = nivel de confianza, para el cual se consideró trabajar con un 90%, cuya puntuación estándar es de 1.64.
- p = probabilidad de éxito o proporción esperada
- pq = varianza de la población, cada una equivale a 0.50, (p=q=0.50)
- q = probabilidad de fracaso
- e = error de estimación máximo aceptable, para el cual se considera un error de 10%.

$$n = \frac{(1.64)^2 \cdot 0,50 \cdot 0,50}{(0,1)^2}$$

$$n = 67.24 \approx 67$$

Por tanto, al emplear la ecuación 1 con un nivel de confianza del 90% y un error de estimación máximo aceptable del 10% se percibe una muestra aproximada de 67 personas.

En la siguiente tabla 1 se visualiza un resumen acerca de la población de estudio.

Tabla 1. Población de estudio

Población	Número	Porcentaje
Directores Fundación Paquito	2	2.74%
Personal de la Fundación Paquito	4	5.48%
Potenciales adoptantes	67	91.78%
Total	73	100%

2.2.3 Recolección de información

La recolección de información se realizó por medio de entrevistas y encuestas. Las entrevistas fueron aplicadas al grupo de la Fundación Paquito, y las encuestas a los potenciales adoptantes.

Las guías de entrevista para el gerente y personal de la Fundación Paquito se encuentran en el Anexo A y Anexo B respectivamente. De igual forma, la encuesta aplicada a potenciales adoptantes se encuentra en el Anexo C.

A continuación se exponen las respuestas, las cuales no son transcripciones específicas de las entrevistas. Las respuestas fueron modificadas para acondicionarlas al contexto de la investigación.

Tabla 2. Entrevista dirigida al coordinador de la Fundación

N.	Respuesta	Observación
1.	Entre las funciones que realizan están el rescatar animales perdidos, cuidarlos, rehabilitarlos y darlos en adopción a personas previamente calificadas. Además de coordinar cada una de las funciones antes mencionadas.	Las funciones que llevan a cabo van desde el ingreso del animal, incluido el adecuado cuidado del mismo hasta la entrega en adopción.
2.	Los animales de compañía que atiende la fundación son perros y gatos.	Están comprometidos específicamente con los perros y gatos.
3.	Mediante un censo reciente se tiene una población de 103 animales de compañía, entre perros y gatos.	Es una población medianamente grande para 5 personas encargadas de atenderlos.
4.	Al ser rescatados son evaluados por un veterinario. Además, les administramos todas las vacunas necesarias, los desparacitamos y esterilizamos. Se maneja una ficha de ingreso.	La ficha de ingreso es fundamental tanto para la Fundación como para los potenciales adoptantes.
5.	Las personas interesadas en adoptar deben ser evaluadas mediante un bloque de preguntas (entrevista) y aceptar ciertas condiciones. Además, se solicita una copia de su cédula, fotos del domicilio y ubicación. Incluso, se realiza una visita al domicilio de la persona para verificar que no está mintiendo.	Muestran un proceso riguroso para poder efectuar la adopción del animal de compañía.
6.	Información relevante como fotos y videos. También la raza, edad, sexo, condición de salud, vacunas y la personalidad de la mascota.	Interés en dar a conocer a los animales de compañía no solo mediante fotos sino mediante videos.
7.	Mediante una visita al domicilio de la persona se verifica la información que proporcionó en la entrevista.	La persona interesada debe aceptar ser visitado por el personal de la Fundación para verificar la información.
8.	La persona responsable es Sandra Ojeda, miembro fundadora de la Fundación y rescatista por más de 20 años. Además, consideramos a una persona apta (responsable) en base a la entrevista que se le aplicó.	La entrevista es clave para dictaminar si una persona es apta para adoptar un animal de compañía.

9.	Como inconvenientes se tiene que las personas buscan de preferencia animales pequeños. También, hay casos en que devuelven al animal por motivos de cambio de domicilio. Entre los problemas, se tiene la falta de seriedad y compromiso por parte de los adoptantes que devuelven al animal.	Existe una preferencia de adopción hacia los animales pequeños. Por otro lado, hay personas que pierden el interés en la mascota al punto de devolverlo a la Fundación.
10.	Se gestiona un aproximado de 4 a 5 adopciones por mes.	Es un número bajo de adopciones.
11.	No he tenido la necesidad u oportunidad de utilizar una app con realidad aumentada.	No tienen experiencia con apps de realidad aumentada.
Conclusión: Los procesos que maneja la Fundación cuando rescatan a un animal, así como el de dar en adopción, son procesos consecutivos. El hecho de presentar un perfil del animal de compañía al público mediante un registro adecuado es algo fundamental, que necesita ser preciso. De igual manera, es necesario enfocarse en la eficiencia al momento de llenar el bloque de preguntas (entrevista) por parte de las personas interesadas en adoptar. Por lo anterior mencionado se sugiere buscar un enfoque eficiente a través de la automatización de los procesos que garantice el manejo y precisión de los datos involucrados.		

Tabla 3. Síntesis de las entrevistas realizadas al personal de la Fundación

N.	Respuesta	Observación
1.	Desempeñan sus tareas como voluntarios.	Son personas que no perciben un sueldo por las labores que realizan para la Fundación.
2.	Entre los datos o información que se solicitan para iniciar con la adopción están las fotos de la casa o predio, la ubicación exacta del domicilio y la cédula del adoptante.	Destaca la importancia de la ubicación del domicilio para conocer dónde va a estar el animal de compañía y si el ambiente es el adecuado, para así evitar futuras devoluciones.
3.	Los adoptantes al no tener casa propia y al momento de cambiarse de domicilio se ven en la necesidad de devolver al animal de compañía, debido a esto es preferible que se tenga casa propia.	Se indica que es preferible dar en adopción a las personas que tienen casa propia.
4.	Es complicado documentar las adopciones que se van realizando y de igual manera el hecho de almacenar la información procesada.	El inadecuado manejo de la documentación de las adopciones efectuadas como el almacenamiento es un problema para la consistencia y precisión de la información.
5.	Las personas adoptantes necesitan asesorarse bien antes de adoptar un animal de compañía, tienen que ser sumamente conscientes de la responsabilidad de tener un animal de compañía, de cuestionarse cómo se verán en un futuro con su animal de compañía. Por otro lado, se debe prestar atención a la esterilización de los animales.	Destaca la importancia de concientizar a las personas adoptantes sobre la responsabilidad que representa tener un animal de compañía.
6.	Se tiene fichas de adopción los cuales se almacenan en formato excel y pdf.	Los datos de las personas adoptantes pueden llegar a extraviarse o corromperse.
7.	Se considera conocer acerca de las vacunas que se le han administrado, la condición de salud y fotos.	Destacan la importancia de la condición de salud del animal de compañía. La condición de salud del animal llama la atención de un adoptante.
8.	Si se conoce la experiencia que ofrece la realidad aumentada mediante un dispositivo móvil.	Han experimentado con apps de realidad aumentada.

Conclusión: Los procesos de ingresar o registrar un animal de compañía a la Fundación necesita ser adaptado a un sistema que agilize el almacenamiento y a su vez garantice la información. Además, la información acerca de los adoptantes requiere ser procesada de tal manera que permita una facilidad al momento de tomar una decisión acerca de, si una persona es apta o no para adoptar un animal de compañía. Los inconvenientes como el almacenamiento de la información de las adopciones realizadas y el registro manual del ingreso de animales a la Fundación evidencian la urgencia de implementar mejoras en los procesos.

a. Resultados de la encuesta aplicada a personas que se identifican como potenciales adoptantes

Pregunta 1: ¿Qué información acerca de un animal de compañía considera usted que es importante conocer antes de adoptarla?

Tabla 4. Resultados de la encuesta para la pregunta uno

Indicador	Frecuencia	Porcentaje
Fotos e información	12	17.91%
Historial médico y vacunas	10	14.93%
Personalidad	6	8.96%
Todas las anteriores	39	58.20%
Suma total	67	100%

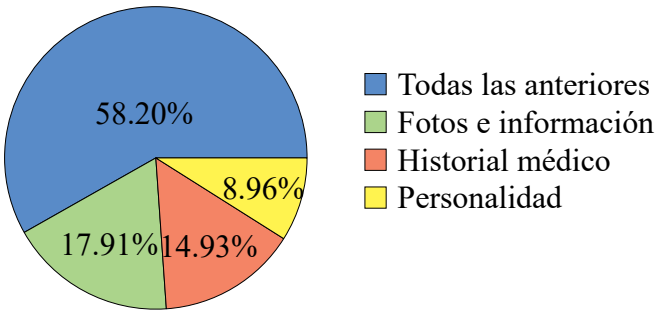


Figura 1. Resultados de la encuesta para la pregunta uno

Análisis e interpretación:

Como se exponen los resultados en la Figura 1 el 17.91% considera como información importante las fotos e información (raza, edad, sexo, etc.), el 14.93% cree que es importante conocer acerca de los antecedentes médicos (como vacunas) y el 8.96% considera importante saber sobre la personalidad que tiene el animal de compañía,

mientras que el 58.20% de los potenciales adoptantes consideran importante conocer de manera detallada los datos o información, fotos, historial médico y personalidad del animal de compañía. Por tanto, se evidencia un mayor interés en conocer de manera amplia al animal de compañía antes de adoptarla.

Pregunta 2: ¿Considera usted un inconveniente la movilización hacia la Fundación Paquito para iniciar el proceso de adopción?

Tabla 5. Resultados de la encuesta para la pregunta dos

Indicador	**Frecuencia**	**Porcentaje**
Sí	52	77.61%
No	15	22.39%
Suma total	67	100%

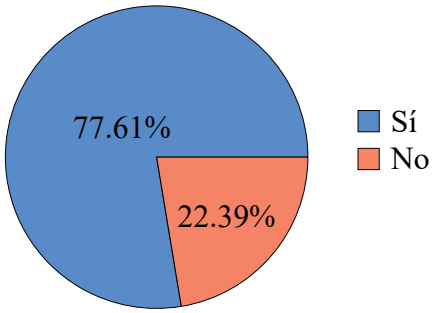


Figura 2. Resultados de la encuesta para la pregunta dos

Análisis e interpretación:

Como se muestran los resultados en la Figura 2 el 77.61% indican que es un inconveniente tener que movilizarse a la Fundación para tener que iniciar con el proceso de adopción, mientras que el 22,39% no lo identifica como un inconveniente el terner que acercarse a la Fundación para iniciar con el proceso de adopción. Por tanto, existen personas que desean una experencia más fluida al momento de adoptar una animal de compañía.

Pregunta 3: ¿Le gustaría realizar una búsqueda de animales de compañía en una app mediante el nombre, color, raza y género?

Tabla 6. Resultados de la encuesta para la pregunta tres

Indicador	**Frecuencia**	**Porcentaje**
Totalmente de acuerdo	36	53.73%
De acuerdo	19	28.36%
Ni de acuerdo ni en desacuerdo	8	11.94%
En desacuerdo	4	5.97%
Totalmente en desacuerdo	0	0.0%
Suma total	67	100%

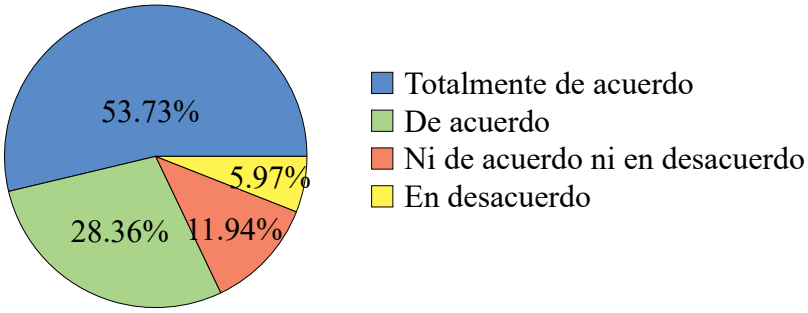


Figura 3. Resultados de la encuesta para la pregunta tres

Análisis e interpretación:

Como se contemplan los resultados en la Figura 3 el 53.73% de los potenciales adoptantes está totalmente de acuerdo en que le gustaría buscar animales de compañía en una app mediante parámetros como el color, raza, edad y sexo del animal, el 28.36% señala estar de acuerdo, el 11.94% no está ni de acuerdo ni en desacuerdo y el 5.97% está en desacuerdo. Por lo tanto, se manifiesta el interés del adoptante de tener la facilidad de conocer animales de compañía en base a características específicas.

Pregunta 4: ¿Le gustaría recibir notificaciones sobre nuevos animales de compañía disponibles para adopción mediante la app?

Tabla 7. Resultados de la encuesta para la pregunta cuatro

Indicador	**Frecuencia**	**Porcentaje**
Totalmente de acuerdo	38	56.72%
De acuerdo	18	26.87%
Ni de acuerdo ni en desacuerdo	9	13.43%
En desacuerdo	2	2.98%
Totalmente en desacuerdo	0	0.0%
Suma total	67	100%

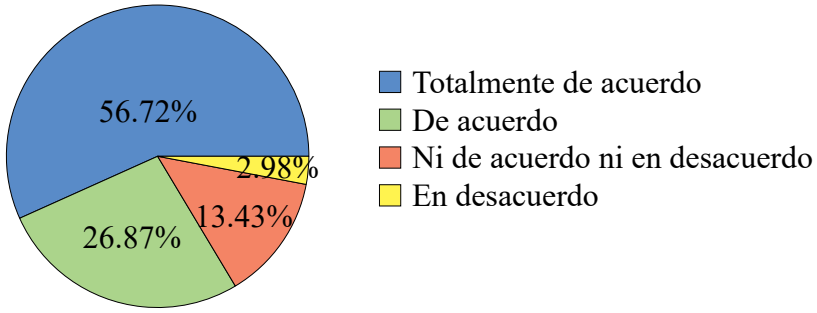


Figura 4. Resultados de la encuesta para la pregunta cuatro

Análisis e interpretación:

Como se presentan los resultados en la Figura 4 el 56.72% de los adoptantes les gustaría recibir notificaciones sobre nuevos animales de compañía que estén disponibles para adopción mediante la app, el 26.87% está de acuerdo, el 13.43% no está ni de acuerdo ni en desacuerdo y el 2.98% está en desacuerdo. En consecuencia, los potenciales adoptantes les interesa mantenerse informados acerca de nuevos candidatos para poder adoptar.

Pregunta 5: ¿Conoce o tiene nociones acerca de lo que es la realidad aumentada?

Tabla 8. Resultados de la encuesta para la pregunta cinco

Indicador	**Frecuencia**	**Porcentaje**
Sí	38	56.72%
No	29	43.28%
Suma total	67	100%

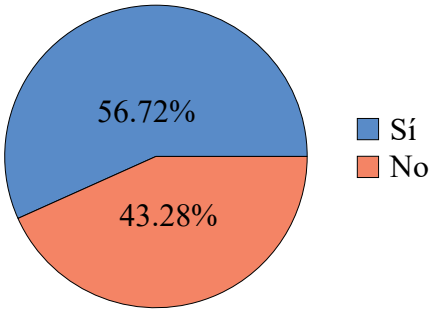


Figura 5. Resultados de la encuesta para la pregunta cinco

Análisis e interpretación:

Como se evidencian los resultados en la Figura 5 el 56.72% de potenciales adoptantes conoce o tiene nocion acerca de lo que es la realidad aumentada, mientras que el 43.28% no tiene conocimiento respecto al tema de realidad aumentada. Por tanto, a pesar de haber personas que conocen acerca de realidad aumentada, se deja ver, que hay personas que desconocen del tema.

Pregunta 6: ¿Le gustaría visualizar en la app al animal de compañía mediante realidad aumentada?

Tabla 9. Resultados de la encuesta para la pregunta seis

Indicador	**Frecuencia**	**Porcentaje**
Totalmente de acuerdo	31	46.27%
De acuerdo	21	31.34%
Ni de acuerdo ni en desacuerdo	13	19.40%
En desacuerdo	0	0.0%
Totalmente en desacuerdo	2	2.99%
Suma total	67	100%

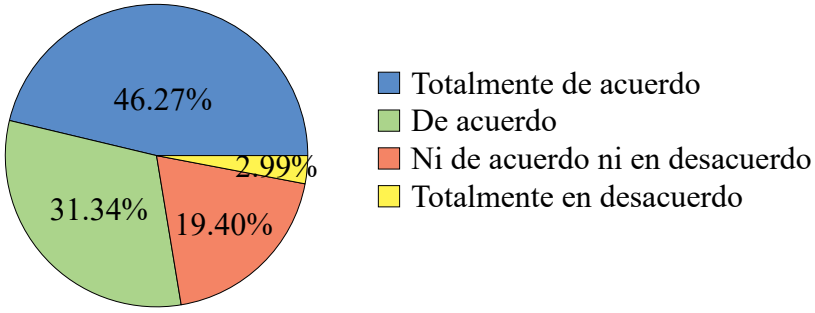


Figura 6. Resultados de la encuesta para la pregunta seis

Análisis e interpretación:

Como se puede apreciar en la Figura 6 el 46.27% de potenciales adoptantes está totalmente de acuerdo en que le gustaría visualizar con su celular al animal de compañía mediante realidad aumentada, el 31.34% está de acuerdo, el 19.40% no está ni de acuerdo ni en desacuerdo y el 2.99% está totalmente en desacuerdo. Por lo tanto, es evidente que los potenciales adoptantes están interesados en conocer al animal de compañía mediante realidad aumentada.

2.2.4 Procesamiento y análisis de datos

Mediante la información recolectada con las entrevistas y encuestas se pudo concluir que:

- La Fundación necesita enfocarse en gestionar de una manera eficiente y adecuada el proceso de ingreso de un animal de compañía, ya que la información acerca de los animales de compañía representa la carta de presentación ante los potenciales adoptantes. En consecuencia, es necesario prestarle el debido cuidado a la información y datos de los animales, con el fin de poder garantizar la información al público, y así mismo, se incrementan las posibilidades de que los animales sean adoptados.

- El proceso de adopción tiene su curva de complejidad, pues el haber completado la entrevista para la adopción no garantiza que dicha persona sea considerada apta para el cuidado responsable del animal de compañía. Además, si una persona quiere adoptar un animal de compañía en mas de una ocasión, se ve obligada a llenar la entrevista una y otra vez, cuando hoy en día es posible mediante un sistema hacerlo una sola vez y cuando la persona considere actualizar su información, entonces sera ahí cuando realmente seaa necesario tocar la entrevista.

- Existe una brecha entre los potenciales adoptantes y los animales de compañía que cuida la Fundación, pues es difícil que la persona pueda conocer al animal de compañía de su interes entre los mas de 100 animales que se encuentran en la Fundación. En consecuencia, al no conocer algo en específico del animal, se da el caso que la persona devuelve el animal a la Fundación. Por tanto, es necesario que la persona pueda decidir de manera segura cual es el animal que desea, pues cada persona puede tener un gusto más hacia los animales de tamaño pequeño o quizas mas grandes, o les agrada más un color que otro. Por lo antes mencionado, se requiere enfocar en la estrategia de como se presenta al animal de compañía ante un potencial adoptante.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Análisis y selección de las tecnologías para el desarrollo de la app

3.1.1 Análisis y selección de herramientas para implementar realidad aumentada

a. ARCore

Google ARCore es una herramienta desarrollada por Google, para crear experiencias de realidad aumentada. En [31] detalla como por medio del uso de varias API, ARCore permite que el dispositivo móvile detecten su entorno, comprenda el mundo e interactúe con la información. ARCore usa tres funciones clave para integrar contenido virtual en el mundo real visto por medio de la cámara del móvil.

- **Detección de movimiento.** Permite al teléfono comprender y rastrear su posición con respecto al mundo real.
- **Comprensión del entorno.** Otorga al teléfono la capacidad de poder rastrear el tamaño y la posición de todo tipo de superficies (superficies horizontales, verticales, inclinadas como pisos, paredes, etc.).
- **Estimación de la luz.** El teléfono tiene la capacidad de calcular las condiciones de iluminación actuales del entorno.

b. Vuforia

En [32] describe a Vuforia como un software que propoorciona las herramientas para trabajar con realidad aumentada. Es una herramienta compatible con diferentes dispositivos, como smartphones, tabletas y gafas. Con su uso, es posible desarrollar aplicaciones, visualizar gráficos y animaciones en 3D creadas por Unity 3D en tiempo real. También, usa marcadores para combinar el mundo real con el virtual. Por otra parte, ofrece seguimiento a una variedad de objetos y espacios que se categorizan como objetos, imágenes o ambientes, lo cual permite al desdarrollador posicionar y orientar objetos virtuales.

c. ARToolKit

En [33] describe a ARToolKit como un sistema fiducial visual que utiliza marcadores cuadrados de color blanco y negro con un patrón de imagen en el centro. Funciona con la cámara del dispositivo móvil, identificando los marcadores y proyectando los modelos virtuales. El reconocimiento de cada patrón de imagen lo realiza mediante el método tracking, el cual está basado en algoritmos de visión por computadora, determinando la posición y orientación de la cámara y los marcadores para establecer sistemas de coordenadas locales y globales.

Una vez presentadas las herramientas se elaboró una tabla comparativa de las herramientas para proyección de objetos 3D en el entorno real.

Tabla 10. Comparación SDK de Realidad Aumentada [1]

Criterio	**ARCore**	**Vuforia**	**ARToolKit**
Licencia	Libre	Libre y Comercial	Libre
Plataformas Compatibles	Java, Unity, Unreal, Android y iOS	Android, iOS y Unity	Linux, Android, iOS, macOS y Unity
Caracteríticas principales	▪ Uso de anclas para situar objetos ▪ Experiencias multiusuario ▪ Detección de imágenes ▪ Detección de planos y el entorno	▪ Detección de imágenes ▪ Detección de Vumark ▪ Detección de objetos ▪ Rastreo de planos	▪ Reconocimiento de marcadores (se debe usar un recuadro negro al rededor) ▪ Tracking de superficies planas ▪ Detección de marcas personalizadas (No simétricas)
Comunidad y Soporte	Comunidad masiva de Unity; soporte frecuente y activo.	Comunidad grande; actualizaciones regulares.	Comunidad pequeña

3.1.2 Técnicas para proyectar modelos 3D en el entorno real

a. Basada en marcadores

Para implementar Realidad Aumentada basada en marcadores se hace uso de patrones

visuales predefinidos, conocido como marcadores. Los marcadores también pueden ser imágenes las cuales son reconocidas por la cámara del dispositivo móvil para superponer los objetos 3D en tiempo real. Además, tiene una alta precisión en colocar objetos virtuales y es relativamente sencilla de implementar. A continuación se presentan los pasos necesarios de manera breve para implementar esta técnica. [34]

- Seleccionar o crear los marcadores visuales: Para implementar este tipo de proyección se debe definir uno o varios marcadores, dependiendo de la funcionalidad de la app.
- Preparar herramientas: Se debe configurar las herramientas con las que se va a trabajar, como Vuforia o ARCore.
- Entrenar el reconocimiento de los marcadores: Luego de la configuración de las herramientas, es necesario que el sistema pueda reconocer los patrones de los marcadores o imágenes para poder colocar el objeto 3D. Por ejemplo, Vuforia tiene un sistema que permite rastrear una imagen como marcador, cargando la imagen en una base de datos alojada en la nube.
- Integrar el contenido 3D: Para mostrar el modelo 3D deseado, debe ser importado dentro del proyecto. Una vez importado en el proyecto, se debe configurar las coordenadas para que esté alineado con el marcador.
- Probar la app: Con el uso de cámara del dispositivo móvil se debe enfoncar el marcador y verificar que el contenido 3D se muestre sobre el marcador tal como se configuró en el proyecto.

b. Sin marcadores

La implementación de Realidad Aumentada sin marcadores se basa en usar las características del entorno físico, como superficies planas o paredes. Todas las características del entorno son rastreadas por la cámara del dispositivo móvil para ser procesadas, y así superponer el objeto 3D en tiempo real. El uso es bastante flexible ya que no depende de marcadores para posicionar los objetos 3D, pero se requiere de hardware con mayores capacidades para una buena experiencia de Realidad Aumentada. A continuación se muestran los pasos necesarios de manera breve para esta técnica. [35]

- Seleccionar herramientas: Es necesario definir la herramienta adecuada para desarrollar. Esto depende del ssitema operativo del dispositivo en donde se va a probar la app. Por ejemplo, si es para Android se usará ARCore y si es iOS se usará ARKit.

- Configurar el proyecto: Se debe integrar el SDK en el proyecto ya sea que se use Unity o Unreal Engine. Luego, se debe configurar los componentes como la cámara del dispositivo para que detecte el entorno. Por ejemplo, en Unity se hace uso de los componentes AR Session y XR Origin.

- Implementar detección de planos y superficies: Es necesario hacer uso de las funcionalidades que ofrecen las herramientas de Realidad Aumentada. Es momento de habilitar las funcionalidades que permiten el rastreo de superficies y detección de planos.

- Integrar objetos 3D: Es momento de carrgar los modelos 3D que se desean proyectar en el entorno real. Se debe configurar la posición y orientación del modelo 3D respecto a los puntos rastreados por la cámara del dispositivo. Además, es posible agregar interacciones para que el usuario pueda mover, escalar o rotar el modelo 3D.

- Probar la app: Verificar que los modelos se posicionan correctamente en las superficies detectadas.

c. Basada en ubicación

Esta clase de implementación de Realidad Aumentada se basa en los datos de GPS y sensores del dispositivo para colocar objetos 3D en ubicaciones geográficas específicas. Esta técnica es útil para aplicaciones al aire libre donde depende de la ubicación del usuario, ya que incentiva al usuario a explorar el entorno real. A continuación se presenta de manera breve los pasos para esta técnica. [36]

- Configurar sensores de ubicación: Se debe asegurar que el dispositivo tiene habilitado el GPS, la brújula y el giroscopio.

- Configurar el proyecto: Se debe integrar el SDK de mapas en el proyecto. Además, se debe configurar los permisos necesarios para acceder a la ubicación

del dispositivo. Por ejemplo, si se trata para un dispositivo Android, se debe configurar los permisos en el archivo *AndroidManifest*.

- Configurar coordenadas a los objetos 3D: Es necesario definir las ubicaciones geográficas donde se espera que se muestre el objeto 3D. Se usan las coordenadas para posicionar el objeto 3D en el mundo real.
- Vincular el objeto con la ubicación: Se debe configurar el objeto para que se active y sea visible cuando el usuario está cerca de la ubicación.
- Probar la app: Verificar que los modelos se aparecen en las ubicaciones correctas. Si es necesario se ajustan las coordenadas y el radio de activación.

d. Basada en reconocimiento de objetos

Esta clase de implementación de Realidad Aumentada consiste en que el sistema reconoce objetos tridimensionales específicos en el entorno real y puede superponer modelos 3D o información relacionada directamente sobre ellos. Esta implementación permite a los usuarios tener interacciones más naturales y contextuales con objetos del mundo real. A continuación se muestra de manera breve los pasos para esta técnica. [34]

- Seleccionar herramientas: Se deben seleccionar herramientas para reconocimieto de objetos. Por ejemplo Vuforia o OpenCV.
- Preparar objetos físicos: Los objetos reales que se desean reconocer deben ser escaneados. Luego de haber escaneado, los datos deben ser importados al proyecto. Para escanear, se pueden usar herramientas como Blender.
- Entrenar el sistema de reconocimiento: Se usa el SDK para registrar los objetos físicos en la base de datos del proyecto. Además, se definen puntos clave del objeto físico para facilitar el reconocimiento.
- Integrar el contenido virtual: Se debe cargar el modelo 3D en el entorno de desarrollo. Configurar el modelo virtual para que se alinee con el obejo físico cuando sea detectado.
- Probar la app: Usar la cámara para detectar los objetos físicos y verificar que los modelos asociados se muestren.

Una vez finalizado el análisis de las herramientas para el desarrollo de aplicaciones de Realidad Aumentada, se decidio usar ARCore, debido a la facilidad de encontrar documentación, esto gracias la gran comunidad que tiene y las constantes actualizaciones de soporte que recibe. Para la técnica de proyección de modelos 3D en el entorno real, se decidio usar la Realidad Aumentada sin marcadores, debido a la flexibilidad que le da al usuario de poder mostrar un modelo 3D en cualquier superficie o plano que pueda ser detectado. Además, ofrece una mayor interacción con el modelo a diferencia de las otras técnias presentadas previamente.

3.1.3 Aplicaciones web para generar modelos 3D

a. Tripo AI

Es una herramienta de modelado 3D impulsada por inteligencia artificial que permite generar modelos 3D de alta calidad a partir de texto o imágenes en cuestión de segundos. Sus características incluyen la generación instantánea de modelos 3D en solo 8 segundos, precisión impulsada por IA gracias a su entrenamiento con millones de conjuntos de datos 3D, y modelos listos para su integración inmediata en tuberías 3D profesionales. Tripo AI está diseñado para ser utilizado por una amplia gama de usuarios, desde desarrolladores de juegos hasta entusiastas de la impresión 3D. [37]

b. Meshy

Es una herramienta de generación de modelos 3D que permite a los usuarios crear modelos y animaciones 3D a partir de texto o imágenes. Sus características incluyen la conversión de texto a 3D, la conversión de imágenes a 3D, la generación de texturas a partir de texto, y la creación de animaciones. Meshy también ofrece soporte para mapas PBR, integración API, y exportación en múltiples formatos 3D como OBJ, FBX, USDZ, GLB, STL y BLEND. Es conocida por su velocidad y facilidad de uso. [38]

c. CSM AI

Es una plataforma avanzada de inteligencia artificial que permite generar mundos 3D controlables y listos para motores de juego a partir de fotos, textos o bocetos. Sus características incluyen la generación de modelos 3D animados, la capacidad de transformar bocetos en 3D, y la integración de entradas multimodales para optimizar

los flujos de trabajo en 3D. CSM AI está diseñada para desarrolladores de videojuegos, diseñadores de realidad virtual y aumentada, artistas 3D e ingenieros de simulación. [39]

Acontinuación se presenta una tabla comparativa de las aplicaciones web para generar modelos 3D. Además, se muestra un ejemplo del modelo 3D generado por cada aplicación web a partir de una sola imagen, como se muestra en la Figura 7.

Tabla 11. Comparación de aplicaciones web para generar modelos 3D

Criterio	**Tripo AI**	**Meshy**	**CSM AI**
Generación de modelos 3D	Texto e imágenes	Texto e imágenes	Fotos, textos y bocetos
Precisión	Alta (entrenamiento con IA)	Alta (detalles finos)	Alta (modelos animados)
Integración	API, múltiples formatos 3D	API, múltiples formatos 3D	Motores de juego
Funciones adicionales	Texturización, animación	Texturización, animación	Animación
Créditos plan gratuito	600	200	10
Usuarios objetivo	Desarrolladores, entusiastas 3D	Artistas 3D, desarrolladores	Desarrolladores de videojuegos, diseñadores VR/AR, artistas 3D, ingenieros de simulación

Figura 7. Foto original del animal de compañía.

Figura 8. Ejemplo de modelo 3D generado por la aplicación web Tripo AI.

Figura 9. Ejemplo de modelo 3D generado por la aplicación web Meshy.

Figura 10. Ejemplo de modelo 3D generado por la aplicación web CSM AI.

Tras finalizar el análisis de las aplicaciones web para generar modelos 3D, se decidió usar Tripo AI, debido a la facilidad de uso y la rapidez con la que genera modelos 3D de alta calidad. Además, ofrece la posibilidad de integrar los modelos en múltiples formatos 3D. Por otro lado, se considero el plan gratuito que cada uno ofrece con

el número de modelos 3D que se pueden generar mensualmente, y la calidad de los modelos generados.

3.1.4 Análisis y selección de la metodología de desarrollo

Una metodología de desarrollo de software [40] consta de una serie de fases o procesos que llevan a cabo sistemáticamente el diseño, desarrollo, implementación y mantenimiento del software, desde las etapas iniciales de identificación de las necesidades del producto hasta el logro de los objetivos de desarrollo. También son usadas para lograr el trabajo en equipo de manera organizada. Otras ventajas relevantes del uso de las metodologías es que permiten un mejor control sobre el avance del trabajo, reducen la incidencia de errores y predicen situaciones adversas. Por otro lado ayuda a gestionar los recursos disponibles y ahorrar tiempo.

a. Comparación entre las metodologías tradicionales y las metodologías ágiles

En la Tabla 12 se presenta un estudio comparativo entre las metodologías tradicionales y las metodologías ágiles.

Tabla 12. Comparación entre las metodologías tradicionales y las metodologías ágiles [2]

Criterio	**Metodologías Tradicionales**	**Metodologías Ágiles**
Hipótesis fundamental	Los sistemas son totalmente especificables, predecibles y se desarrollan mediante una planificación detallada y extendida	El proyecto es adaptativo y de alta calidad, es desarrollado por pequeños equipos que utilizan el principiode mejora continua del diseño y las pruebas basadas en una respuesta y cambio rápido
Estilo de gestión	Comando y control	Liderazgo y colaboración
Conocimiento Administrativo	Explícito	Tácito
Modelo de desarrollo	Modelo de ciclo de vida (cascada, espiral o modelos modificados)	Modelo evolutivo de entrega
Estructura organizacional	Mecánico, orientado a grandes organizaciones	Organización flexible y participativo, fomenta la cooperación social. Orientado a pequeñas y medianas organizaciones

Control de calidad	Planificación difícil y control estricto. Pruebas difíciles y tardías	Control permanente de requisitos y pruebas permanentes
Requisitos de usuario	Detallado y definido antes de la codificación e implementación	Entrada interactiva
Costo de reinicio	Alto	Bajo
Dirección de desarrollo	Fijo	Fácilmente cambiante
Pruebas	Después de completar la codificación	Cada iteración
Participación del cliente	Bajo	Alto
Habilidades adicionales requeridos en los desarrolladores	Habilidades interpersonales y conocimientos básicos del negocio	Habilidades interpersonales y conocimientos básicos del negocio
Escala apropiada del proyecto	Gran escala	Baja y mediana escala
Clientes	Con acceso al conocimiento, cooperativo, representativo y empoderado	Dedicado, informado, cooperativo, representativo y capacitado
Tamaño	Grandes equipos y proyectos	Pequeños/Medianos equipos y proyectos
Requisitos	Bien definidos antes de usar	Dinámicos
Modelo de desarrollo	Ciclo de vida	Entrega evolutiva
Gestión de escalado	El problema se escala a los gerentes del proyecto	Cuando surgen problemas todo el equipo trabaja junto para resolverlo
Preferencias del modelo	Favorece la anticipación	Favorece la adaptación
Producto o proceso	Más enfocados sobre los procesos que sobre el producto	Menos enfoque en los procesos formales
Planificación	Todo con gran detalle	De sprint en sprint
Estimación del esfuerzo	El gestor del proyecto estimay obtiene la aprobación del propietario del proyecto	Se tiene un líder que facilita las tareas y el equipo hace la estimación
Revisiones y aprobaciones	Constantes revisiones y aprobaciones por parte de los líderes del proyecto	Se realizan después de cada iteración
Duración	De media / larga duración	Corta duración

Adaptabilidad al cambio	Proyecto cerrado	Proyecto abierto a cambios
Documentación	Documentación rigurosa	Poca documentación
Costos	Se acerca a lo estimado	Puede dispararse
Metodología de ejecución	Basados en estándares de desarrollo	Basados en heurísticas
Roles	Específicos y no intercambiables	Genéricos y flexibles
Gestión del cambio	Resistencia en la mayoría de los casos. No se esperan cambios considerables	Se esperan cambios constantes
Gestión contractual	Contrato prefijado	Flexibilidad en el contrato

Tras finalizar el análisis comparativo se decidió usar una metodología ágil. Esto debido a las características que posee tales como la adaptabilidad durante el desarrollo, la flexibilidad a cambios, el enfoque a proyectos de tamaño de pequeño/mediano, una planificación a corto plazo, el enfoque a equipos pequeños/medianos y una alta participación por parte del cliente.

Por tanto se procedió a realizar un estudio comparativo entre las metodologías ágiles, la cual se muestra en la Tabla 13 con la finalidad de identificar la metodología más adecuada para el desarrollo del proyecto.

Tabla 13. Comparación de las metodologías ágiles [3–5]

Criterio	XP	Lean	RAD
Enfoque	Iterativo e incremental	Iterativo e incremental	Prototipado
Principios	Integración continua, programación en pares, desarrollo basado en pruebas, comentarios de los clientes	Centrarse en el valor, eliminar desperdicios, flujo, mejora continua, respeto por las personas	Desarrollo rápido, parrticipación del usuario, desarrollo iterativo, creación de prototipos
Tamaño del equipo	3 - 5	2 - 3	2 - 3
Tamaño del proyecto	Pequeños y medianos	Grandes	Pequeños y medianos con requisitos cambiantes
Ventajas	Calidad y comunicación	Velocidad y flexibilidad	Costo y tiempo
Simplicidad	Simplicidad en el código, diseño y solución de problemas	Se busca eliminar complejidad innecesaria con el fin de mejorara la eficiencia	Enfocado en desarrollar soluciones simples y rápidas, evitando excesos de diseño
Entrega de software	Frecuente y regular	Incremental	Entrega rápida
Planificación	Continua	Planificación y modelado	Planificación rápida y flexible

Tras finilizar el análisis en la Tabla 13 se decidio usar RAD como método de desarrollo debido a su enfoque de desarrollar soluciones simples y rápidas. Además, por la planificación rápida y flexible. También, al contar con la participación del usuario, es posible tener una mayor daptabilidad a los cambios que se presenten durante el desarrollo, impulsando una mejora continua.

3.1.5 Análisis y selección de framework para frontend

Para el desarrollo de la interfaz de usuario se contemplaron tres frameworks,como son: React Native, Flutter y Ionic. A continuación, se presenta una comparación con el objetivo de elegir el framework más adecuado para el proyecto.

Tabla 14. Análisis entre React Native, Flutter y Ionic [3–5]

Criterio	React Native	Flutter	Ionic
Popularidad	Muy Alta	Alta	Moderada
Rendimiento	Moderado	Alto	Moderado
Soporte por la comunidad	Grande	Mediana	Pequeña
Curva de aprendizaje	Baja	Mediana	Mediana
Conocimientos necesarios	JavaScript	Dart	JavaScript
Migraciones	Moderada	Fácil	Fácil
Flexibilidad	Alta	Alta	Moderada

Tabla 15. Ventajas y desvetajas de React Native, Flutter y Ionic [3–5]

Framework	Ventajas	Desventajas
React Native	▪ Posee una única base de código para múltiples aplicaciones. ▪ Incorpora JavaScript, usado por muchos desarrolladores. ▪ Requiere menos tiempo de depuración. ▪ Posee una curva de aprendizaje baja.	▪ El rendimiento y la experencia de la interfaz de usuario son similares a los de las aplicaciones nativas, pero no son los mismos. ▪ Existen bibliotecas que han sido abandonadas, mientras que otras son de baja calidad. ▪ Rendimiento bajo en el uso de memoria (CPU).
Flutter	▪ Permite crear aplicaciones para iOS y Android con una base de código. ▪ Tiene un rendimiento relativamente alto en el uso de la memoria (CPU). ▪ Carga de trabajo más bajo comparado con otros frameworks. ▪ Excelente documentación y comunidad de soporte. ▪ Posibilidad de realizar interfaces personalizadas con facilidad.	▪ Un mayor tamaño de las aplicaciones. ▪ Menor cantidad de bibliotecas comparado con otros frameworks. ▪ El lenguaje Dart no es tan utilizado. ▪ Depende del soporte continuo a largo plazo de Google.

Ionic	▪ Fácil de aprender si ya se conoce Angular. ▪ Gran cantidad de componentes preconstruidos. ▪ Permite tener una experencia de interfaz de apariencia nativa. ▪ Reduce el tiempo, el esfuerzo y los recursos necesarios para crear aplicaciones multiplataforma.	▪ La aplicación completa se actualiza cada vez que un desarrollador realiza un cambio, lo que afecta la velocidad de desarrollo. ▪ Necesita plugins para acceder a características nativas. ▪ Depende en gran medida de WebView.

Después de analizar acerca de las características, ventajas y desventajas de cada framework, se ha concluido que Flutter es la opción más adecuada para este proyecto. Su capacidad para crear aplicaciones nativas con una sola base de código, alto rendimiento en el uso de memoria, excelente documentación y una comunidad de soporte activa, destacan como sus principales ventajas. Además, la flexibilidad para diseñar interfaces personalizadas y poseer una menor carga de trabajo comparada con otros frameworks, refuerzan su posición como la mejor elección frente a React Native e Ionic.

3.1.6 Análisis y selección de framework para backend

Para poder identificar el framework adecuado para el desarrollo del backend se realizo un estudio comparativo, el cual se muestra a continuación en la Tabla 16.

Tabla 16. Comparación de Frameworks para backend [6]

Framework	Ventajas	Desventajas
Spring Boot	▪ Integración con el ecosistema Spring ▪ Configuración automática ▪ Alta escalabilidad ▪ Comunidad y soporte	▪ Curva de aprendizaje pronunciada ▪ Consumo de memoria

Express	▪ Simplicidad ▪ Flexibilidad ▪ Ligero y rápido ▪ Amplia adopción	▪ No impone una estructura rígida ▪ Menos herramientas integradas
Django	▪ ORM robusto ▪ Seguridad ▪ Desarrollo rápido	▪ Rendimiento ▪ Monolítico ▪ Curva de aprendizaje

Después de finalizar el análisis en la Tabla 16 se decidio trabajar con Spring Boot antes que con Express y Django debido a la robustes que tiene al estar integrado con el ecosistema Spring, el cual es esencial para aplicaciones a gran escala. Aunque tiene una curva de aprendizaje pronunciada y un alto consumo de memoria, sus funciones de configuración automática, su alta extensibilidad y su sólido soporte comunitario lo convierten en una opción ideal para un desarrollo complejo.

3.1.7 Análisis y selección de gestor de base de datos

Los sistemas gestores de bases de datos (SGBD) [7] son esenciales para gestionar y administrar datos en las organizaciones modernas. Estos sistemas permiten almacenar, organizar y acceder a grandes cantidades de información de manera eficiente y segura. La importancia de un SGBD radica en su capacidad para respaldar operaciones críticas, facilitar la toma de decisiones basada en datos, garantizar la integridad y seguridad de la información y mejorar la eficiencia operativa. Con el crecimiento exponencial de los datos en la era digital, SGBD se ha convertido en una herramienta esencial para gestionar y aprovechar eficazmente el valor de los datos.
Acontinuación se presenta mediante una Tabla 17 un estudio comparativo entre los dos gestores de base de datos más usados.

Tabla 17. Comparación de Gestores de bases de datos [7]

Criterio	PostgreSQL	MySQL
Velocidad de respuesta	Alta, especialmente en operaciones complejas	Muy alta en operaciones de lectura
Modelo conceptual	Objeto-relacional	Relacional
Tipos de datos soportados	Avanzados (JSON, XML, Arrays, Hstore)	Comunes (INT, VARCHAR, DATE)

Fiabilidad	Muy alta, soporte ACID completo	Alta, pero depende del motor de almacenamiento (InnoDB es ACID, MyISAM no)
Extensibilidad	Altamente extensible con funciones definidas por el usuario, tipos de datos, operadores	Limitada en comparación
Arquitectura de implementación	Cliente-servidor con diseño modular	Cliente-servidor con múltiples motores de almacenamiento
Manejo de transacción	Transacciones ACID completas	ACID (principalmente con InnoDB)
Concurrencia	Concurrencia robusta con MVCC (Multiversion Concurrency Control)	Buena, pero menos eficiente que PostgreSQL en altas cargas
Tamaño de registros	Soporta registros muy grandes (hasta 1.6TB)	Limite de tamaño de fila es menor (hasta 64KB)
Respaldo de base de datos	Herramientas avanzadas como pg_dump, WAL archiving	Herramientas como mysqldump, pero menos sofisticadas
Permisos de usuarios	Granularidad fina con roles y privilegios	Menos granular, pero suficiente para la mayoría de aplicaciones
Disponibilidad de drivers para JDBC	Disponibles y ampliamente soportados	Disponibles y ampliamente soportados
Interfaz y plataforma soportada	Soporta múltiples interfaces (CLI, GUI) y plataformas (Windows, Linux, macOS)	Amplio soporte de interfaces y plataformas

Tras el análisis en la Tabla 17 se escogio usar PostgreSQL a MySQL debido a su cumplimiento de los estándares SQL, soporte ampliado de tipos de datos, mayor confiabilidad y escalabilidad y mejor concurrencia. Aunque MySQL tiene ventajas en velocidad de lectura y facilidad de configuración, PostgreSQL es más adecuado para aplicaciones complejas que requieren un sistema de base de datos potente con funciones avanzadas.

3.2 Análisis del proceso actual

Teniendo en cuenta los resultados obtenidos de las entrevistas realizadas de las cuales se detalla en el capítulo anterior, se procede a describir los procesos fundamentales que lleva a cabo la Fundación para la respectiva adopción de los animales de compañía.

Procesos importantes como completar un formulario y firmar un acta de adopción que los cuales se pueden conocer en el Anexo D. En la Figura 11 se detalla el proceso, donde:

1. El potencial adoptante se acerca de manera presencial a la Fundación Paquito y expone su deseo al personal en adoptar un animal de compañía.
2. El personal de la Fundación Paquito da la apertura al potencial adoptante de explorar entre los animales de compañía.
3. El potencial adoptante da a conocer al personal de la Fundación cual es el animal de compañía de su interés y con esto solicita dar inicio con el respectivo proceso de adopción.
4. El personal de la Fundación acepta, y a su vez, explica que debe acpetar las condiciones de adopción de la Fundación y solicita al potencial adoptante que debe participar de una entrevista formulada por la Fundación.
5. El potencial adoptante decide si participa o no en la entrevista, en donde:
    a) Si el potencial adoptante acepta participar en la entrevista, entonces se procede a realizar la entrevista.
    b) Si el potencial adoptante no desea participar en la entrevista, se da por finalizado el proceso.
6. El personal de la Fundación solicita una copia de la cédula, fotos de la vivienda y una visita al domicilio para cerciorarse en relación a las respuestas obtenidas en la entrevista, en donde:
    a) Si el personal de la Fundación descubre que el potencial adoptante mintio, decide dar por finalizado el proceso.
    b) Si el personal de la Fundación valida la información como veraz, donde:
        i. Si el personal de la Fundación califica como adoptante apto para el cuidado responsable del animal de compañía, el proceso continua.
        ii. Si el personal de la Fundación califica como adoptante no apto para el cuidado responsable del animal de compañía, el proceso finaliza.

7. El representate de la Fundación entrega al potencial adoptante el animal de compañía y un certificado de adopción, en donde:

    a) El potencial adoptante lo acepta y el proceso de adopción finaliza.

    b) El potencial adoptante cancela la adopción del animal de compañía, por tanto el proceso finaliza.

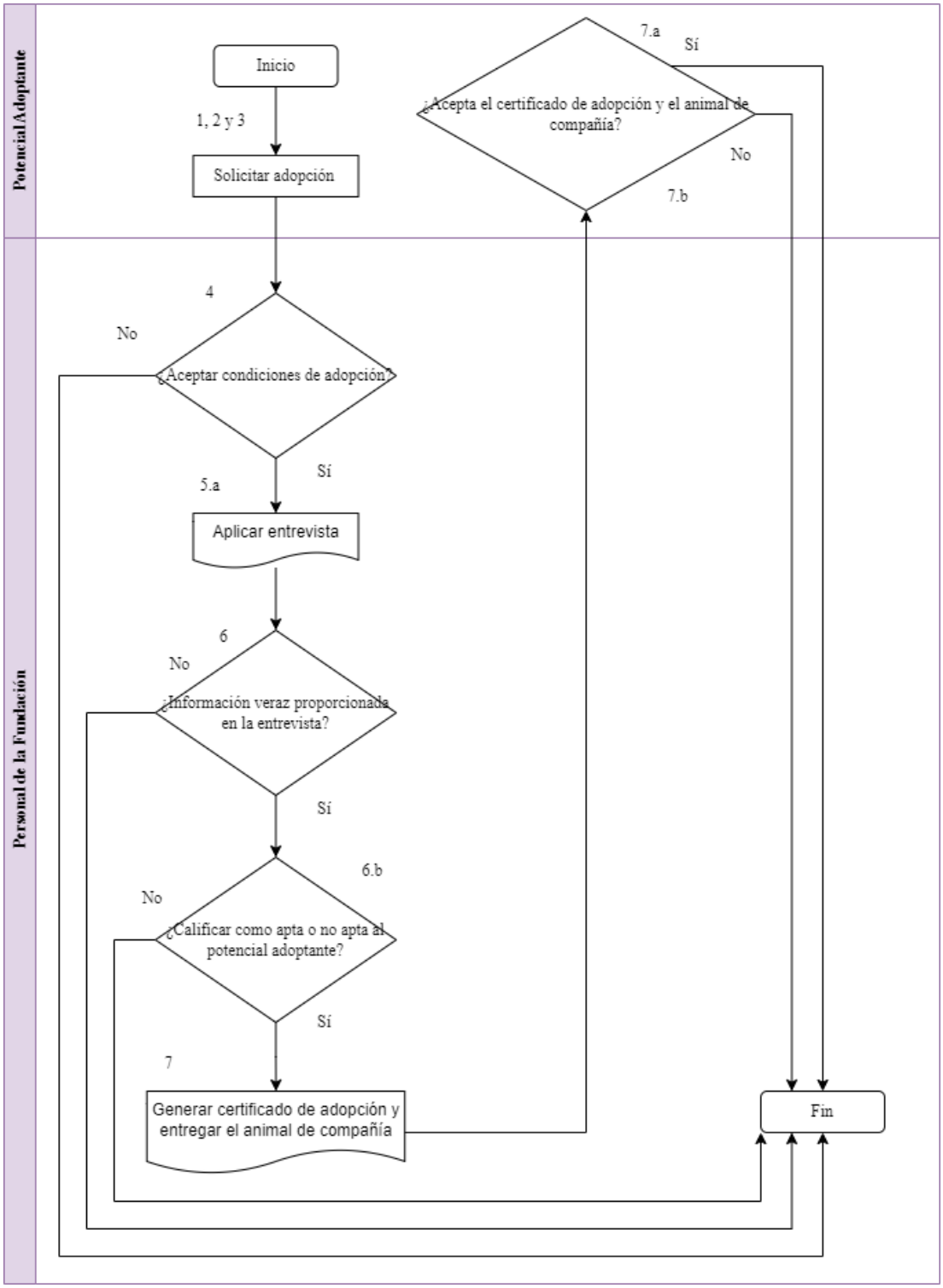


Figura 11. Diagrama de flujo del proceso actual para realizar la adopción de un animal de compañía

3.3 Desarrollo de la propuesta

Para el desarrollo de la propuesta se empleó la metodología RAD. La metodología RAD [41] creada por James Martin en la década de 1970 pero presentada oficialmente mediante la publicación de un libro en 1991 consiste en las siguientes cuatro fases.

- Planificación de requerimientos
- Diseño de usuario
- Construcción
- Transición

En la fase de planififcación de requerimientos se establecen los objetivos del proyecto y se identifican los requisitos del sistema. En la segunda fase se crean diagramas UML y el diseño de la interfaz de usuario mediante prototipos, con el fin obtener retroalimentación. La fase de construcción se centra en el desarrollo del sistema mediante la codificación y, finalmente, la cuarta fase se basa en poner a prueba la aplicación.

3.3.1 Planificación de requerimientos

En esta primera fase para el desarrollo del proyecto se extraen varios requisitos mediante la información recolectada en las entrevistas y encuestas realizadas.
En la Tabla 18 se exponen los usuarios que tendrán interacción con la app móvil. De igual forma, en la Tabla 19 se muestran los requisitos establecidos para el desarrollo del presente proyecto.

Tabla 18. Detalle de usuarios

Usuario	Descripción
Coordinador de la Fundación	Encargado de gestionar el ingreso de los animales de compañía a la Fundación y evaluar las solicitudes de adopción.
Adoptante	Encargado de generar solicitudes de adopción.

Tabla 19. Definición de requerimientos

ID	Requerimiento	Descripción	Prioridad	Riesgo
Todos los usuarios				
R1	Iniciar sesión	El usuario deberá autenticarse con sus credenciales: correo electrónico y contraseña.	Alta	Alto
R2	Cerrar sesión	El usuario tiene permitido cerrar la sesión en cualquier momento.	Alta	Alto
R3	Reestablecer contraseña	El usuario puede reestablecer su contraseña en cualquier momento.	Alta	Alto
Coordinador de la Fundación				
R4	Gestionar tipos de animales de compañía	El Coordinador de la Fundación podrá visualizar, crear, editar y remover tipos de animales.	Alta	Alto
R5	Gestionar tipos de vacunas	El Coordinador de la Fundación podrá visualizar, crear, editar y remover vacunas.	Alta	Alto
R6	Gestionar tipos de desparasitantes	El Coordinador de la Fundación podrá visualizar, crear, editar y remover desparasitantes.	Alta	Alto
R7	Gestionar usuarios	El Coordinador de la Fundación podrá visualizar, crear, editar (solo el rol del personal de la Fundación) e inhabilitar usuarios de la Fundación. Además podrá visualizar e inhabilitar a los usuarios adoptantes.	Alta	Alto
R8	Gestionar animales de compañía	El Coordinador de la Fundación podrá visualizar, crear, editar y remover a los animales de compañía.	Alta	Alto
R9	Gestionar modelos 3D de los animales de compañía	El Coordinador de la Fundación podrá visualizar, cargar, editar y remover los modelos 3D de los animales de compañía.	Alta	Alto
R10	Gestionar solicitudes de adopción	El Coordinador de la Fundación podrá revisar la información de las solicitudes de adopción de los solicitantes. Además podrá aprobar o denegar la solicitud.	Alta	Alto

R11	Gestionar actas de adopción	El Coordinador de la Fundación podrá generar e invalidar la respectiva Acta de Adopción para los adoptantes.	Alta	Alto
Adoptante				
R12	Registro de usuario	El potencial adoptante debe registrarse completando un formulario con sus datos personales. Además no está permitido registrarse varias veces con el mismo correo electrónico.	Alta	Alto
R13	Editar datos personales	El potencial adoptante podrá editar sus datos personales.	Alta	Alto
R14	Cambiar contraseña	El potencial adoptante podrá actualizar su contraseña.	Alta	Alto
R15	Visualizar animales de compañía	El potencial adoptante podrá visualizar una lista en columna vertical de animales de compañía.	Alta	Alto
R16	Buscar animales de compañía	El potencial adoptante podrá buscar animales de compañía en base a los siguientes parámetros: el nombre, la raza, el género, tamaño y el color del animal.	Alta	Alto
R17	Visualizar perfil del animal de compañía	El potencial adoptante podrá visualizar a detalle el perfil del animal de compañía.	Alta	Alto
R18	Visualizar animal de compañía mediante Realidad Aumentada	El potencial adoptante podrá visualizar al animal de compañía mediante realidad aumentada solo si su dispositivo móvil es compatible.	Alta	Alto
R19	Generar solicitud de adopción	El potencial adoptante podrá generar una solicitud de adopción tras completar un formulario.	Alta	Alto
R20	Visualizar solictudes de adopción	El potencial adoptante podrá visualizar las solicitudes de adopción que ha generado.	Alta	Alto
R21	Visualizar actas de adopción	El potencial adoptante podrá visualizar las actas de adopción generadas por el sistema.	Alta	Alto

R22	Responder al acta de adopción	El potencial adoptante podrá aceptar o cancelar el acta de adopción generada por el sistema.	Alta	Alto

Después de haber definido los requerimientos del sistema se establece el plan de trabajo para cada iteración que permite tener un control en el desarrollo del proyecto.

Tabla 20. Identificación de los requerimientos del sistema

N° Iteración	**N°**	**ID**	**Requerimiento**	**Tiempo estimado**	
				Horas	**Días**
Iteración 1	1	R1	Iniciar sesión	7	1
	2	R2	Cerrar sesión	1	1
	3	R3	Reestablecer contraseña	3	1
	4	R4	Gestionar tipos de animales de compañía	5	1
	5	R5	Gestionar tipos de vacunas	5	1
	6	R6	Gestionar tipos de desparasitantes	5	1
	7	R7	Gestionar usuarios	12	2
	8	R8	Gestionar animales de compañía	20	4
Iteración 2	9	R9	Gestionar modelos 3D de animales de compañía	18	3
	10	R10	Gestionar solicitudes de adopción	20	4
	11	R11	Gestionar actas de adopción	15	3
	12	R12	Registro de usuario	6	1
	13	R13	Editar datos personales	6	1
	14	R14	Cambiar contraseña	4	1
	15	R15	Visualizar animales de compañía	16	3
	16	R16	Buscar animales de compañía	8	1
Iteración 3	17	R17	Visualizar perfil del animal de compañía	14	2
	18	R18	Visualizar animal de compañía mediante Realidad Aumentada	20	4
	19	R19	Generar solicitud de adopción	15	3
	20	R20	Visualizar solicitudes de adopción	8	1
	21	R21	Visualizar actas de adopción	8	1
	22	R22	Responder al acta de adopción	8	1

3.3.2 Diseño de usuario

a. Análisis del proceso propuesto

En la Figura 12 se muestra el proceso de solicitud de adopción propuesto por medio del uso de la app móvil, donde:

1. Si el usuario adoptante no está registrado en la app deberá crearse una cuenta completando un formulario con sus datos personales.
2. El usuario adoptante selecciona el animal de compañía de su interés del cual puede revisar el perfil de información e incluso ver al animal de compañía mediante realidad aumentada.
3. El usuario adoptante debe generar la solicitud de adopción para el animal de compañía complentando y enviando la solicitud de adopción a la Fundación.
4. El personal de la Fundación revisa y evalúa la información proporcionada por el usuario adoptante a través de la solicitud de adopción generada, donde.
    a) Si la solicitud de adopción es aprobada por el personal de la Fundación el proceso continúa.
    b) Si la solicitud de adopción es denegada por el personal de la Fundación el proceso finaliza.
5. El personal de la Fundación después de haber aprobado la solicitud de adopción, el sistema generará un acta de adopción, en donde:
    a) Si el usuario adoptante revisa el acta de adopción y la acepta, el proceso finaliza.
    b) Si el usuario adoptante revisa el acta de adopción y no acepta, el proceso finaliza.

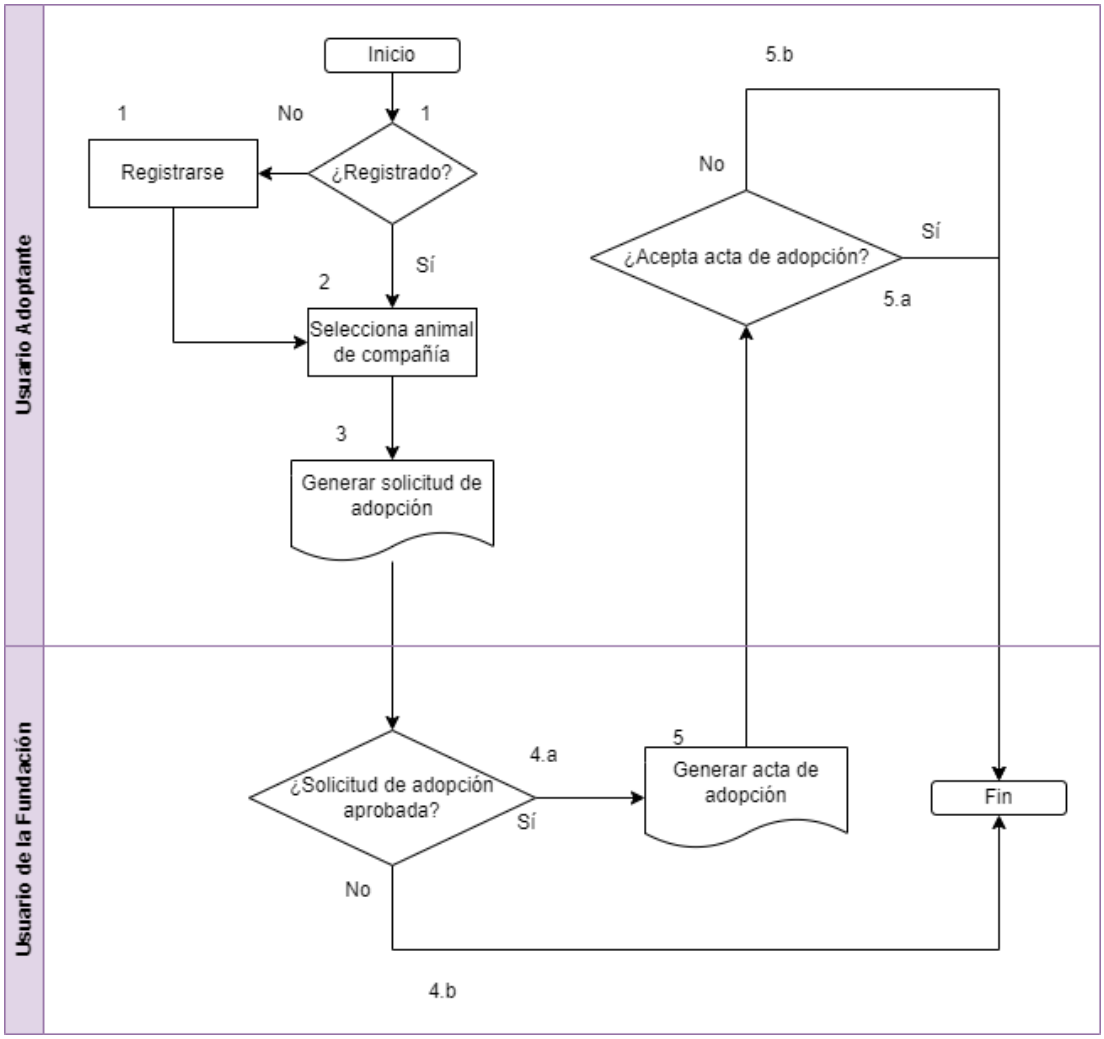


Figura 12. Diagrama de flujo del proceso propuesto para la app

b. Arquitectura

Para el desarrollo del sistema se utilizó la arquitectura cliente-servidor 13. La arquitectura cliente-servidor consiste en la participación de dos componentes, uno es el cliente y otro es el servidor. Por el lado del cliente se encuentra la interfaz de usuario que proporciona la app móvil para permitir la interacción con el usuario (usuario adoptante y usuario de la Fundación). Por tanto, el usuario interesado en adoptar hará uso de los recursos necesarios (disponibles por el servidor) para generar su solicitud de adopción, y de igual forma el usuario de la Fundación podrá gestionar las solicitudes y actas de adopción. Por el lado del servidor se tiene la API, encargada de procesar las peticiones de los usuarios de la app y proveer de los recursos necesarios que satisfacen los requerimientos. Además, la Api es la responsable de manejar la lógica de negocio y

ofrecer la conexión con la base de datos para que los datos puedan persistir. Por último la API tendrá conexión con los servicios de almacenamiento de archivos de AWS S3 para el almacenamiento en la nube.

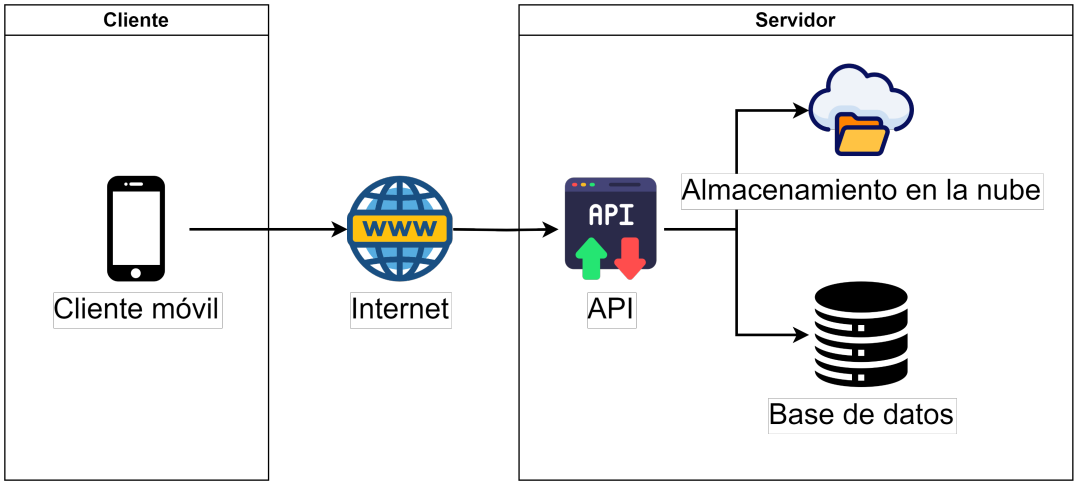


Figura 13. Arquitectura del sistema

c. Prototipado

Como parte de la fase del Diseño de Usuario de la metodologìa RAD, se elaboraron prototipos con el propósito de mostrarlos a los usuarios y poder adquirir una retroalimentación, además, de presentarles una vista previa de la app final. Por otra parte, la retroalimentación ofrece la oportunidad de realizar modificaciones que cubran las necesidades del usuario.

En la Figura 14 se presenta la interfaz donde los usuarios adoptantes deben registrarse en la app completando el formulario. Tienen que ingresar sus datos como un correo y una contraseña que usarán para autenticarse en la app.

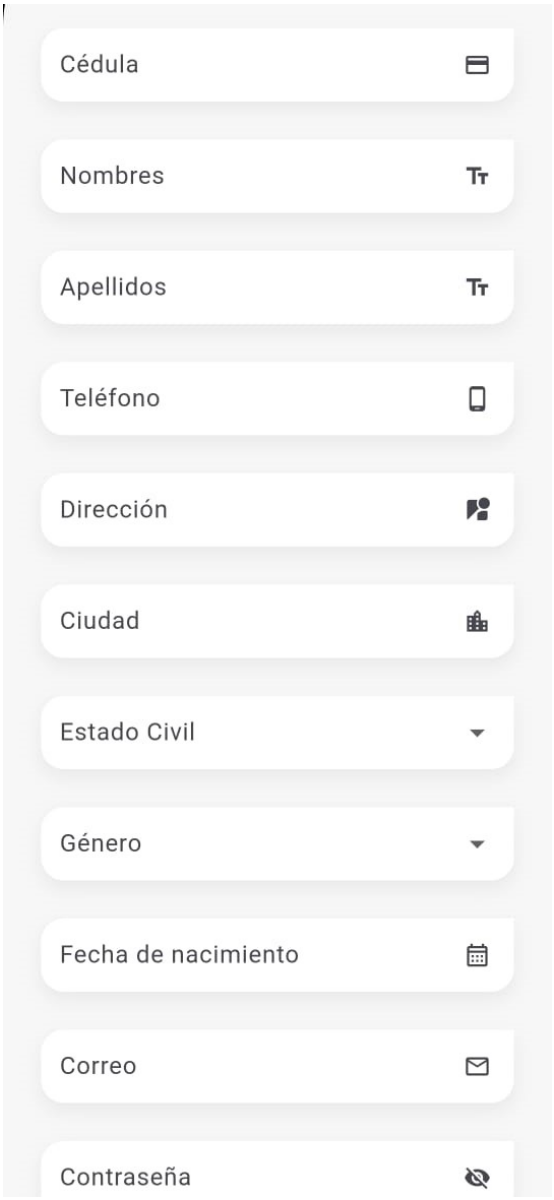


Figura 14. Prototipo de registro para usuario adoptante

En la Figura 15 se presenta la interfaz donde los usuarios adoptantes y el personal de la Fundación deben ingresar sus credenciales como el correo electrónico y la contraseña para ingresar a la app.

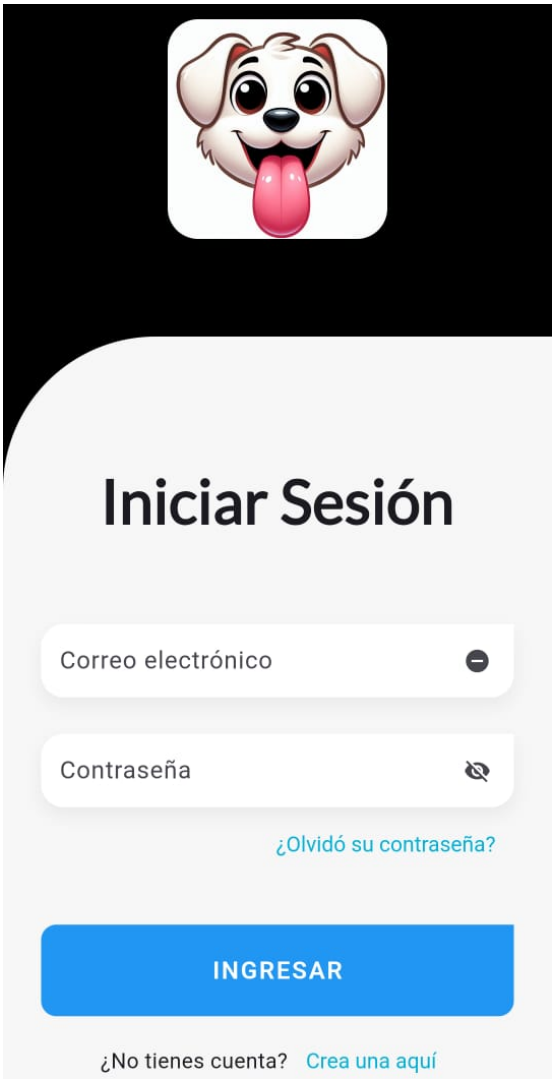


Figura 15. Prototipo de inicio de sesión

En la Figura 16 se presenta la interfaz donde tanto el usuario adoptante como el usuario de la Fundación pueden recuperar la contraseña de su cuenta enviando su correo electrónico.

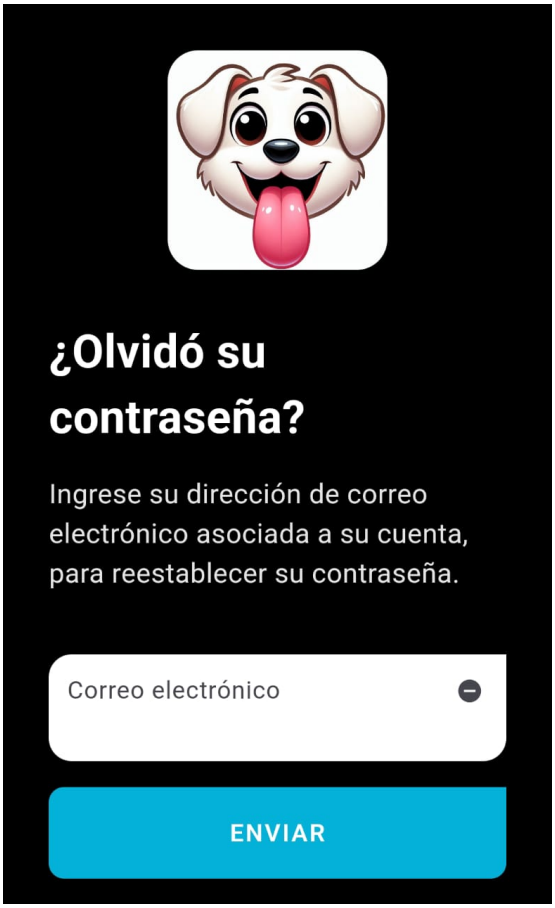


Figura 16. Prototipo de recuperar contraseña

En la Figura 17 se presenta la interfaz principal para el usuario adoptante donde puede visualizar 2 pestañas (Perros y Gatos). Cada pestaña muestra una lista de imagenes de perros y de gatos que cuida la Fundación. Además, el usuario con el ícono de la lupa podrá realizar una búsqueda mediante los valores como nombre, color, raza, género y tamaño del animal de compañía. Por otro lado, cada imagen viene acompañado de un ícono el cual permitirá al usuario ver al animal de compañía mediante Realidad Aumentada.

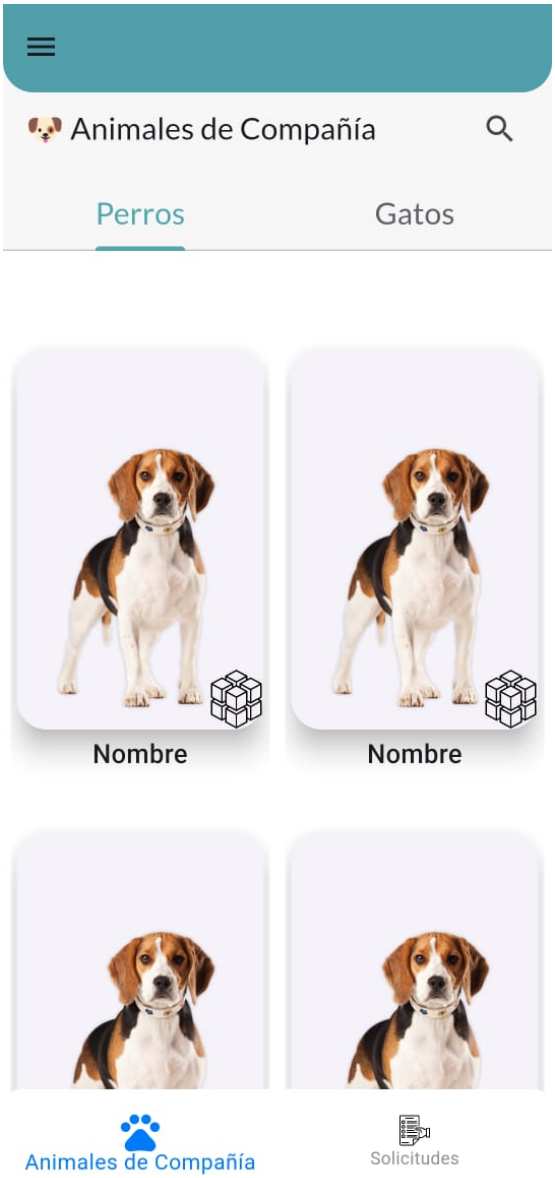


Figura 17. Prototipo de interfaz principal del usuario adoptante

En la Figura 18 se presenta la interfaz para visualizar el perfil del animal de compañía. El usuario adoptante podrá revisar la información del animal como la edad, la raza, el género, el tamaño, el color y si está esterilizado. Además, el historial de control de vacunas y desparasitantes que ha tenido por parte de la Fundación. Por último, tiene un botón el cual representaría el inicio del proceso de adopción.

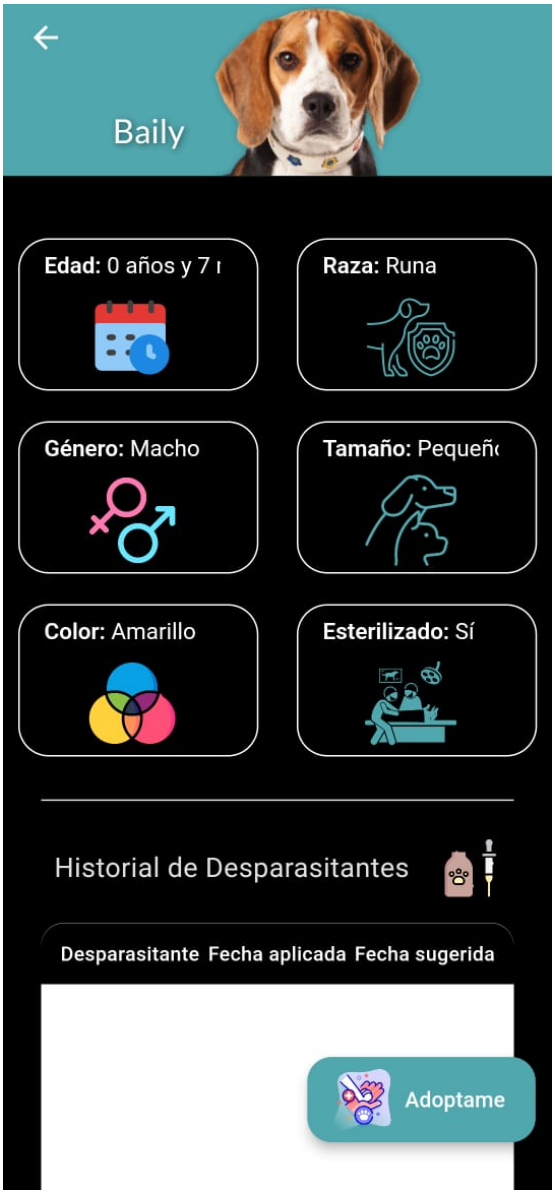


Figura 18. Prototipo de interfaz para el perfil del animal de compañía

En la Figura 19 se presenta la primera página que compone el formulario para la solicitud de adopción. En esta vista, se mostrarán valores tales como la fecha de elaboración, datos del animal de compañía que se desea adoptar y una tabla para conocer el resultado de la solicitud.

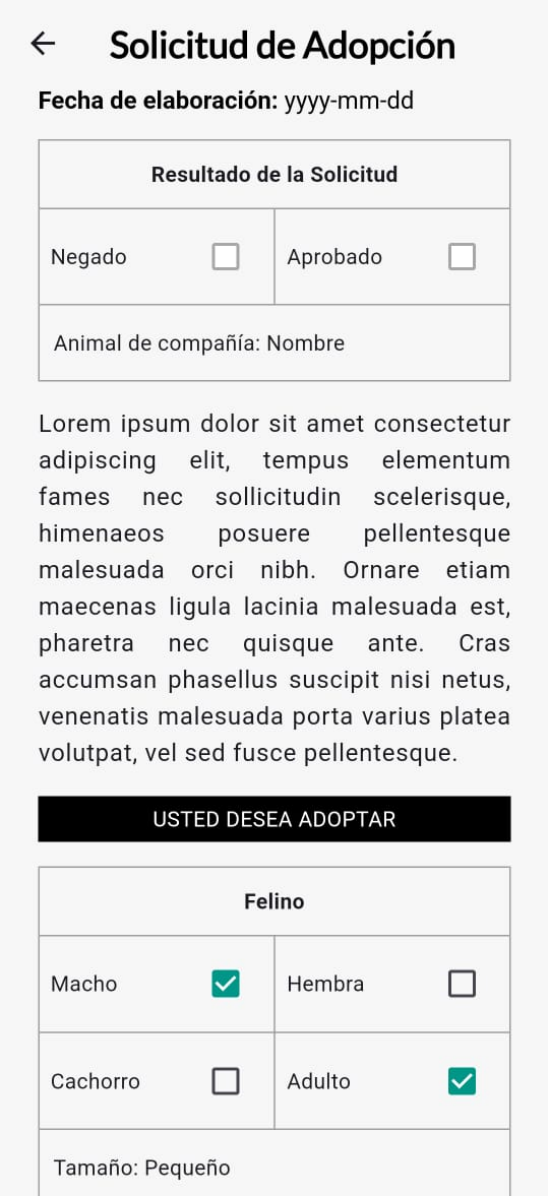


Figura 19. Prototipo formulario de solicitud de adopción - cabecera

En la Figura 20 se presenta la segunda página que compone el formulario para la solicitud de adopción. En esta interfaz, se mostrará información personal del usuario adoptante, sin embargo, el usuario debe completar varios campos faltantes y posteriormente guardar la información presionando el botón “Guardar”.

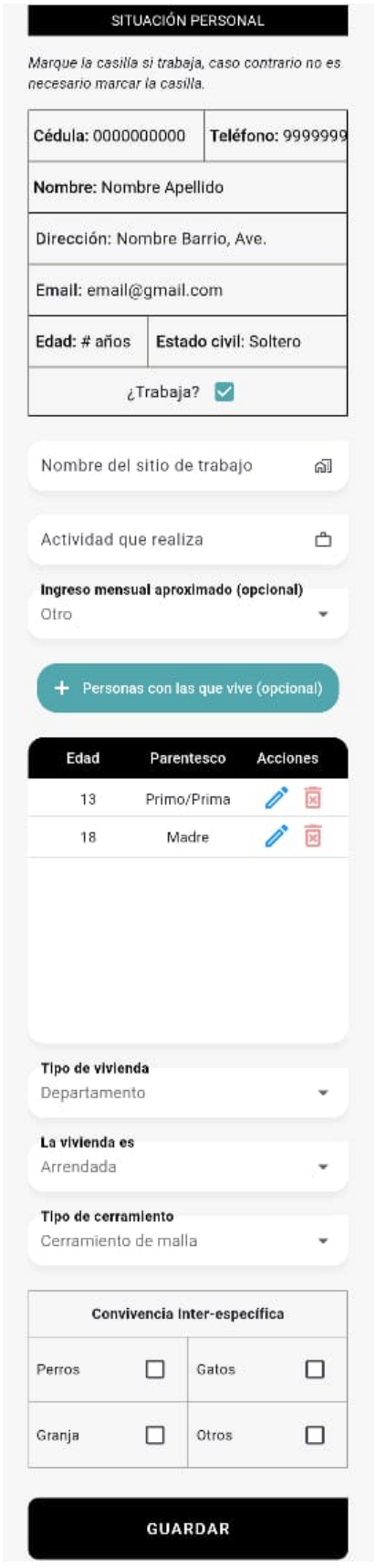


Figura 20. Prototipo formulario de solicitud de adopción - situación personal

En la Figura 21 se presenta la tercera página que compone el formulario para la solicitud de adopción. En esta interfaz, se mostrarán varias tablas donde el usuario adoptante debe seleccionar una o varias opciones para cada tabla (habrá excepciones que solo necesita seleccionar una opción). Además, debe cargar una foto legible de su cédula y finalmente enviar la solicitud presionando el botón “Enviar Solicitud”.

Figura 21. Prototipo formulario de solicitud de adopción - relación con los animales

En la Figura 22 se presenta la interfaz para ver las solicitudes generadas por el usuario adoptante. Esta interfaz estará disponible tanto para el usuario adoptante, como para el usuario de la Fundación. Cada solicitud llevará la fecha en que se envió, junto con el nombre del animal de compañía. Además, el botón indicará si la solicitud fue aprobada o denegada, y en caso de que no haya sido aún calificada, mostrará la palabra "En revisión". Por otro lado, el botón permite ver el contenido de la solicitud de adopción, para que el usuario de la Fundación pueda revisar y calificar.

Figura 22. Prototipo de interfaz para las solicitudes generadas

En la Figura 23 se presenta la interfaz para ver las actas de adopciones generadas por el sistema. Esta interfaz estará disponible tanto para el usuario adoptante, como para el usuario de la Fundación. De la misma manera que las solicitudes de adopción estas mostrarán la fecha en que se envió y el nombre del animal de compañía. Además, tendrá un botón para ver el contenido de la acta de adopción, para que el usuario adoptante

pueda revisar su contenido y aceptarla como se muestra en la Figura 24.

Figura 23. Prototipo de interfaz para las actas generadas

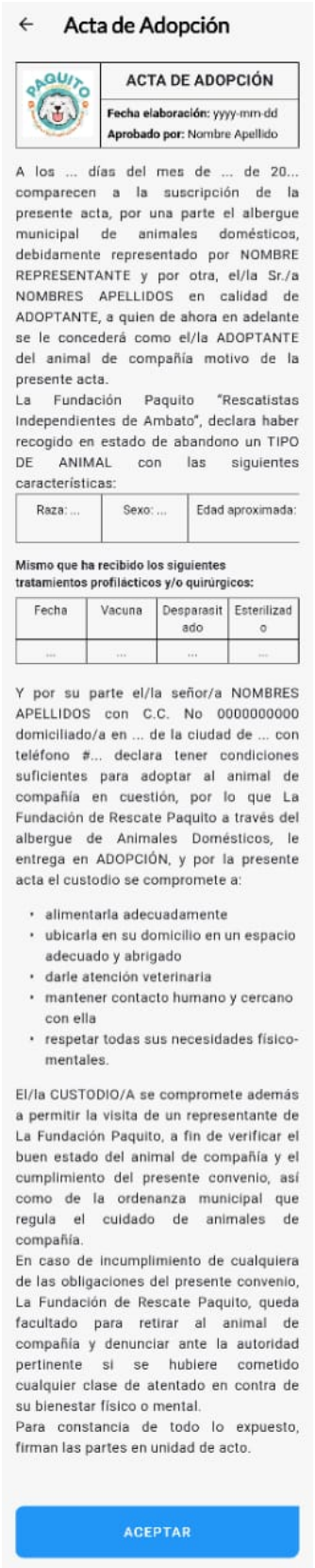


Figura 24. Prototipo de interfaz de acta de adopción

En la Figura 25 se presenta la interfaz principal para el usuario de la Fundación

(administrador). La interfaz principal tendrá una barra de navegación en la parte inferior con las opciones de menú. También, tendrá un botón “Nuevo” para registrar un animal de compañía como se muestra en la Figura 26 en la app. Los animales de compañía registrados se podrán ver en la interfaz acompañados de una imagen, un botón para desactivarlos y otro botón para gestionar el archivo del modelo 3D mediante una interfaz como se muestra en la Figura 27. Además, será posible editar los datos del animal de compañía presionando sobre la imagen.

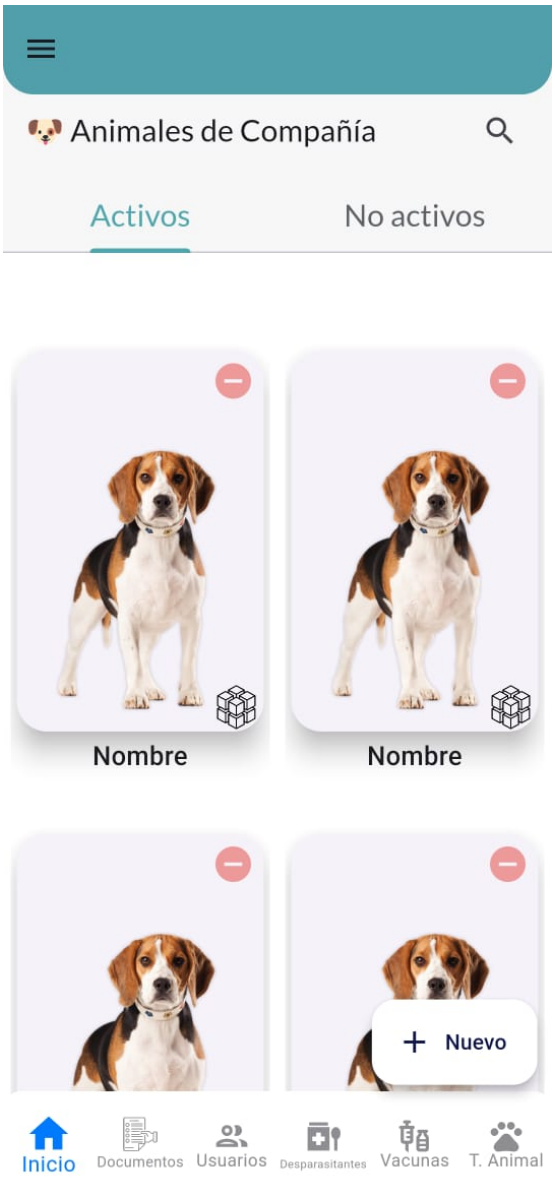


Figura 25. Prototipo de interfaz principal para el administrador

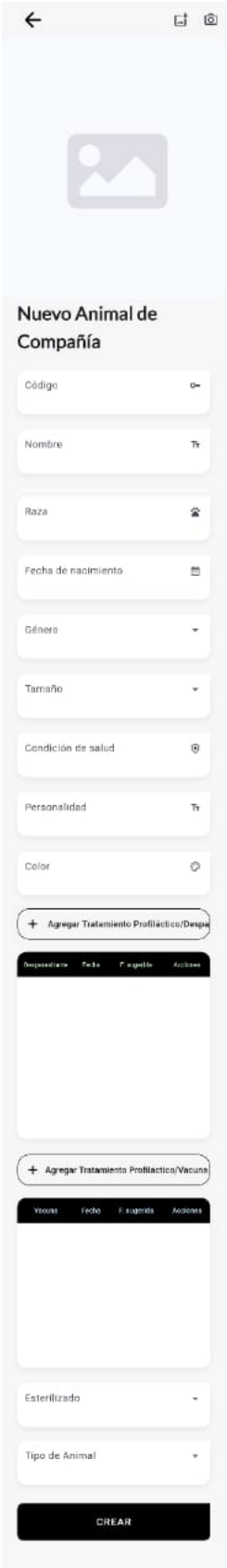


Figura 26. Prototipo de interfaz para registrar un animal de compañía

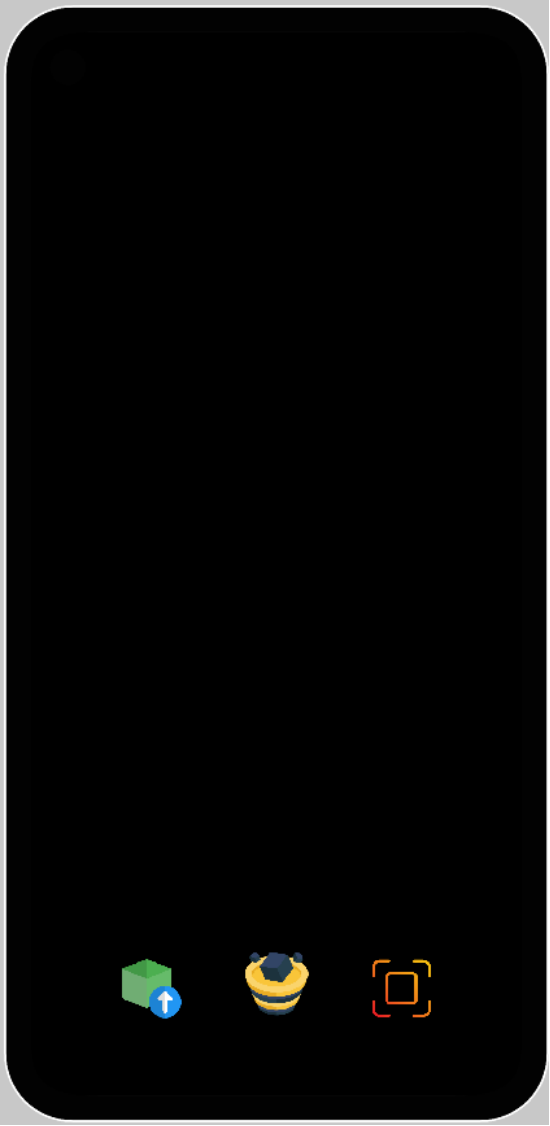


Figura 27. Prototipo de interfaz para gestionar el archivo del modelo 3D

En la Figura 28 se presenta la interfaz para gestionar los usuarios registrados en la app. La interfaz muestra una columna de cartas que contiene los datos del usuario como cédula, nombres, apellidos y el rol que tiene dentro de la app. También, el administrador podrá realizar las siguientes acciones como desactivar o activar al usuario; ver la información personal y registrar a un nuevo usuario como se muestra en la Figura 29.

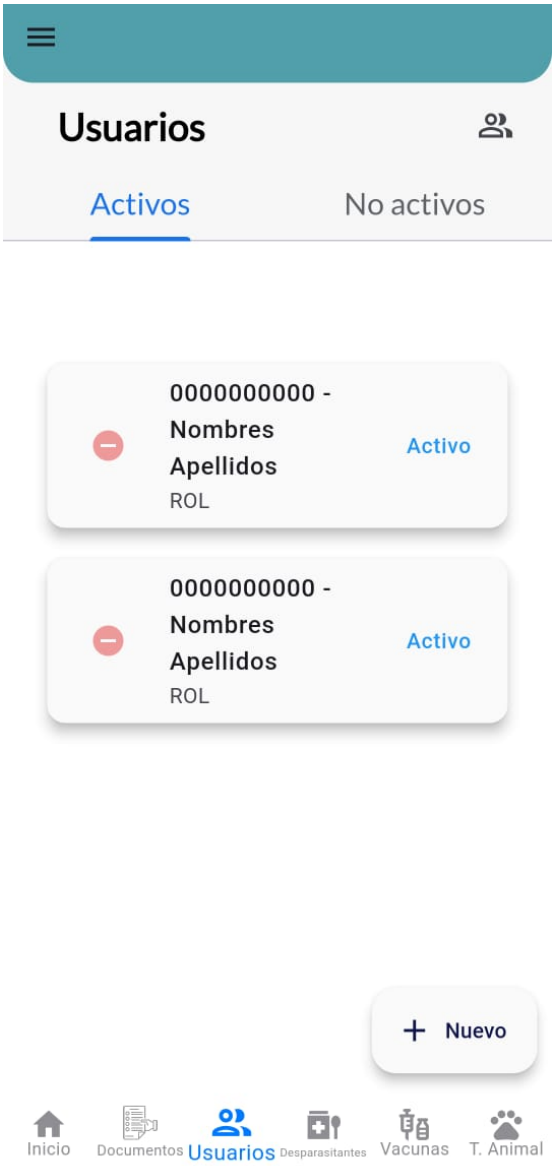


Figura 28. Prototipo de interfaz usuarios para el administrador

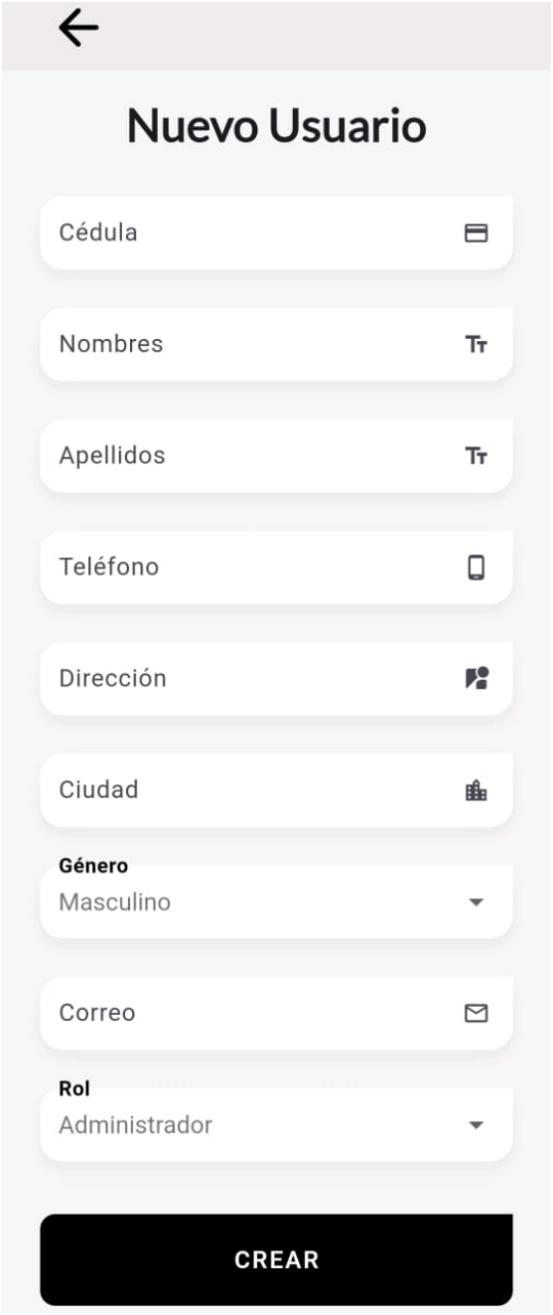


Figura 29. Prototipo de interfaz para registrar usuarios

En la Figura 30 se presenta la interfaz para gestionar los desparasitantes que serán administradas a los animales de compañía. La interfaz muestra una columna de cartas que contiene el nombre del desparasitante y un botón para desactivar el desparasitante (cuando este desactivado puede volver a activarlo). También, el administrador podrá registrar nuevos desparasitantes y presionando sobre uno ya registrado podrá editar la información.

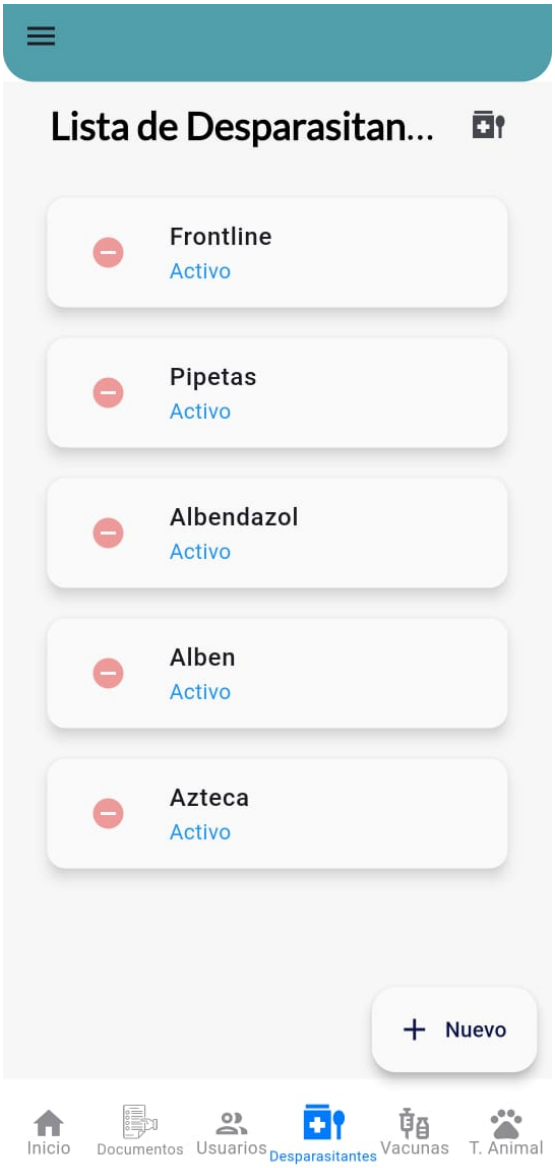


Figura 30. Prototipo de interfaz desparasitantes para el administrador

En la Figura 31 se presenta la interfaz para gestionar las vacunas que serán aplicadas a los animales de compañía. La interfaz muestra una columna de cartas que contiene el nombre de la vacuna y un botón para desactivar la vacuna (cuando este desactivado puede volver a activarlo). También, el administrador podrá registrar nuevas vacunas y presionando sobre uno ya registrado podrá editar la información.

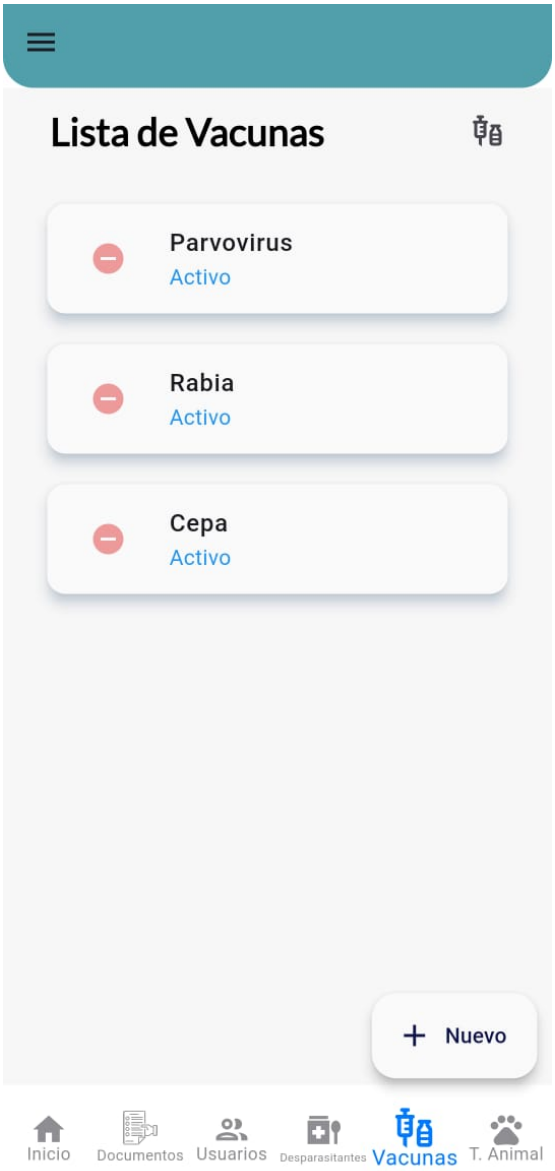


Figura 31. Prototipo de interfaz vacunas para el administrador

En la Figura 32 se presenta la interfaz para gestionar los tipos de animales que habitan en la Fundación. La interfaz muestra una columna de cartas que contiene el nombre del tipo de animal y un botón para desactivar el tipo (cuando este desactivado puede volver a activarlo). También, el administrador podrá registrar nuevos tipos de animales y presionando sobre uno ya registrado podrá editar la información.

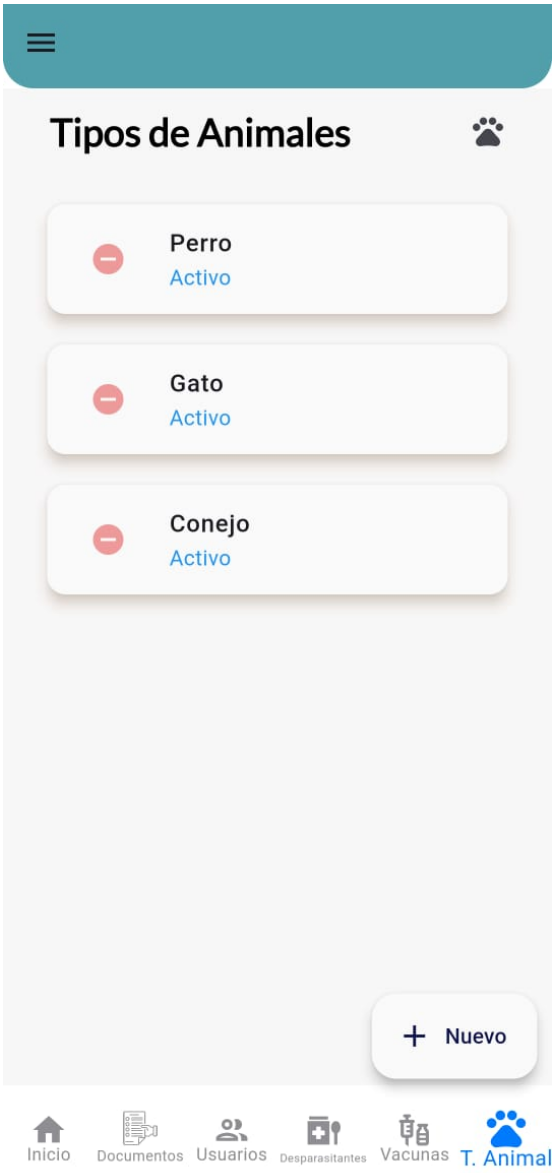


Figura 32. Prototipo de interfaz tipos de animales para el administrador

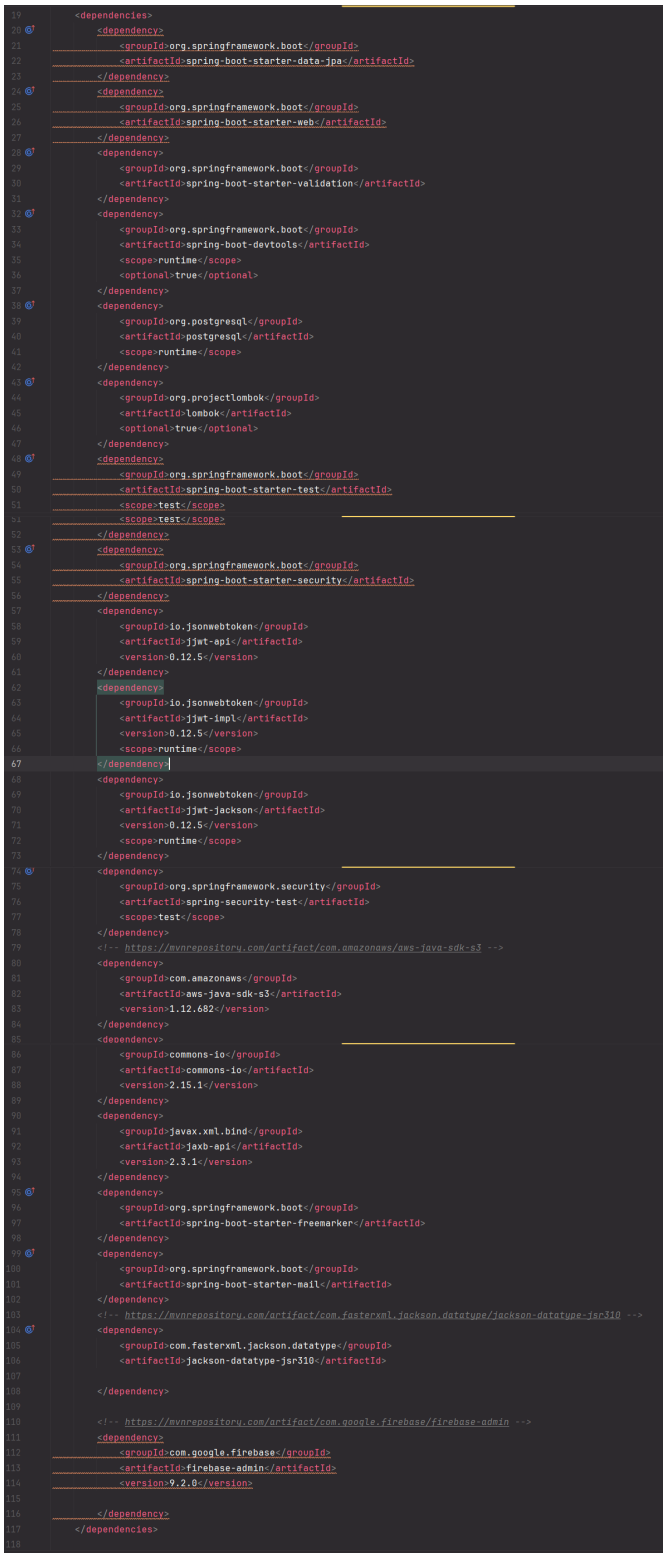
3.3.3 Construcción

En la siguiente sección se describe la construcción de la propuesta del sistema de adopción de animales de compañía.

a. Backend

- ***Dependencias de la API.*** Como herramienta para gestionar las dependencias de la API se utilizó Maven. En la Figura 33 se presenta el archivo pom.xml en donde se

definieron las dependencias necesarias para el funcionamiento de la API.



```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-validation</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-devtools</artifactId>
        <scope>runtime</scope>
        <optional>true</optional>
    </dependency>
    <dependency>
        <groupId>org.postgresql</groupId>
        <artifactId>postgresql</artifactId>
        <scope>runtime</scope>
    </dependency>
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <optional>true</optional>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt-api</artifactId>
        <version>0.12.5</version>
    </dependency>
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt-impl</artifactId>
        <version>0.12.5</version>
        <scope>runtime</scope>
    </dependency>
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt-jackson</artifactId>
        <version>0.12.5</version>
        <scope>runtime</scope>
    </dependency>
    <dependency>
        <groupId>org.springframework.security</groupId>
        <artifactId>spring-security-test</artifactId>
        <scope>test</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/com.amazonaws/aws-java-sdk-s3 -->
    <dependency>
        <groupId>com.amazonaws</groupId>
        <artifactId>aws-java-sdk-s3</artifactId>
        <version>1.12.682</version>
    </dependency>
    <dependency>
        <groupId>commons-io</groupId>
        <artifactId>commons-io</artifactId>
        <version>2.15.1</version>
    </dependency>
    <dependency>
        <groupId>javax.xml.bind</groupId>
        <artifactId>jaxb-api</artifactId>
        <version>2.3.1</version>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-freemarker</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-mail</artifactId>
    </dependency>
    <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.datatype/jackson-datatype-jsr310 -->
    <dependency>
        <groupId>com.fasterxml.jackson.datatype</groupId>
        <artifactId>jackson-datatype-jsr310</artifactId>

    </dependency>

    <!-- https://mvnrepository.com/artifact/com.google.firebase/firebase-admin -->
    <dependency>
        <groupId>com.google.firebase</groupId>
        <artifactId>firebase-admin</artifactId>
        <version>9.2.0</version>

    </dependency>
</dependencies>
```

Figura 33. Definición de las dependencias Maven en el archivo *pom.xml*

- ***Configuración de las propiedades de la API.*** En el archivo *application.properties* se definieron las propiedades como la conexión a la base de datos; el puerto en el que se

ejecutará la API; la configuración para el envío de mensajes via Simple Mail Transfer Protocol (SMTP); la configuración de variables para los servicios de Amazon Web Services y Amazon Simple Storage Service (S3) como se presenta en la Figura 34.

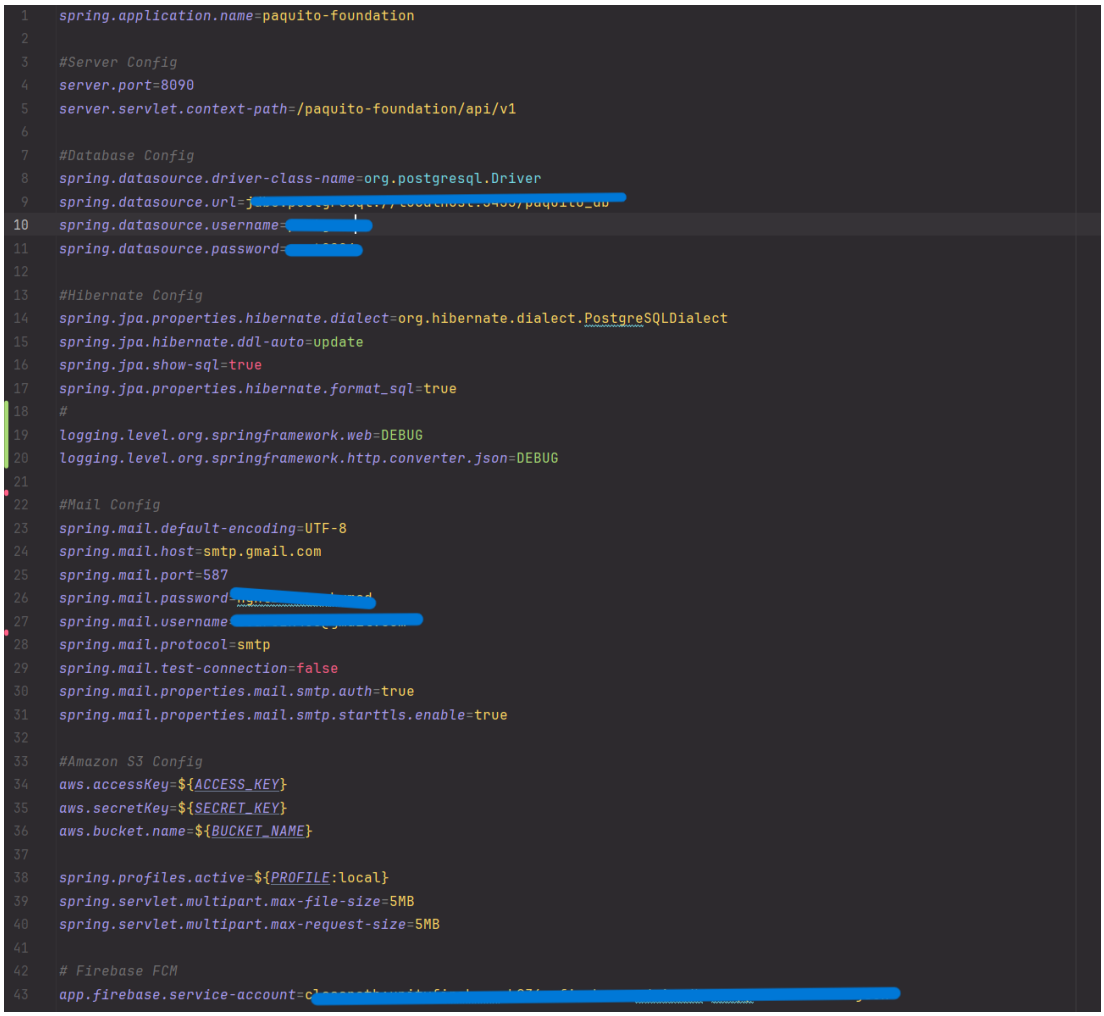


Figura 34. Definición de las propiedades de la API

- ***Configuración de seguridad de la API.*** Para configurar la seguridad de la API se utilizó Spring Security, donde se definieron los parámetros de seguridad para el acceso a los puntos de entrada expuestos en la API con el uso de Json Web Token (JWT) para la autenticación y autorización. También, se configuró los Cross-Origin Resource Sharing (CORS) para el acceso a la API desde la app como se muestra en la Figura 35.

```java
@Configuration
@EnableWebSecurity
@EnableMethodSecurity
public class SecurityAdapter {

    @Autowired
    private JwtAuthFilter jwtAuthFilter;

    suarezKevin
    @Bean
    public UserDetailsService userDetailsService() { return new UserDetailsServiceImpl(); }

    suarezKevin
    @Bean
    public PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }

    suarezKevin
    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        return http
                .csrf(AbstractHttpConfigurer::disable)
                .cors(
                        corsConfig -> corsConfig.configurationSource(corsConfigurationSource())
                )
                .authorizeHttpRequests(
                        authorize -> authorize
                                .requestMatchers("/public/**").permitAll()
                                .requestMatchers("/public/auth/login").permitAll()
                                .requestMatchers("/public/auth/recover-password/**").permitAll()
                                .requestMatchers("/public/auth/validate-token").authenticated()
                                .requestMatchers("/protected/**").authenticated()
                                .anyRequest().denyAll()
                )
                .exceptionHandling(
                )
                .exceptionHandling(
                        exceptionHandling -> exceptionHandling
                                .authenticationEntryPoint((request, response, authException) ->
                                        response.sendError(
                                                HttpServletResponse.SC_UNAUTHORIZED,
                                                authException.getMessage()))
                                .accessDeniedHandler((request, response, accessDeniedException) ->
                                        response.sendError(
                                                HttpServletResponse.SC_FORBIDDEN,
                                                accessDeniedException.getMessage()))
                )
                .sessionManagement(
                        sessionManagement -> sessionManagement.sessionCreationPolicy(SessionCreationPolicy.STATELESS)
                )
                .authenticationProvider(authenticationProvider())
                .addFilterBefore(jwtAuthFilter, UsernamePasswordAuthenticationFilter.class)
                .build();
    }

    suarezKevin
    @Bean
    public AuthenticationProvider authenticationProvider(){
        DaoAuthenticationProvider provider = new DaoAuthenticationProvider();
        provider.setUserDetailsService(userDetailsService());
        provider.setPasswordEncoder(passwordEncoder());
        return provider;
    }

    suarezKevin
    @Bean
    public AuthenticationManager authenticationManager(AuthenticationConfiguration configuration) throws Exception {
        return configuration.getAuthenticationManager();
    }

                )
                .authenticationProvider(authenticationProvider())
                .addFilterBefore(jwtAuthFilter, UsernamePasswordAuthenticationFilter.class)
                .build();
    }

    suarezKevin
    @Bean
    public AuthenticationProvider authenticationProvider(){
        DaoAuthenticationProvider provider = new DaoAuthenticationProvider();
        provider.setUserDetailsService(userDetailsService());
        provider.setPasswordEncoder(passwordEncoder());
        return provider;
    }

    suarezKevin
    @Bean
    public AuthenticationManager authenticationManager(AuthenticationConfiguration configuration) throws Exception {
        return configuration.getAuthenticationManager();
    }

    suarezKevin
    @Bean
    public CorsConfigurationSource corsConfigurationSource() {
        return request -> {
            var cors = new CorsConfiguration();
            cors.setAllowedOrigins(List.of( e1: "http://localhost:4200"));
            cors.setAllowedMethods(List.of("GET", "POST", "PUT", "DELETE","PATCH" ,"OPTIONS"));
            cors.setAllowedHeaders(List.of( e1: "*"));
            return cors;
        };
    }
}
```

Figura 35. Configuración de seguridad de la API

- ***Persistencia y acceso de datos.*** Para manejar la persistencia y el acceso de datos se utilizó Spring Data JPA. Spring Data JPA se encarga de crear de manera fácil las tablas de la base de datos y ofrece una implementación rápida de operaciones CRUD (Create, Read, Update y Delete). Spring Data JPA logra esto mediante la implementación del mapeo objeto-relacional (ORM) de Hibernate.

  Para que una clase pueda ser mapeada a una tabla de la base de datos se utiliza la anotación @Entity. En la Figura 36 se presenta la configuración de una clase abstracta. AbstractEntity se configuró como la clase padre de todas las clases al definirle los atributos comunes que comparten las clases. En la Figura 37 se presenta una clase con la anotación @Entity, estbleciendole en la propiedad name el nombre que se desea que tenga en la base de datos. Además, se establecieron los atributos y las relaciones que esta clase (Tabla) tiene con otras clases para que sean mapeadas en la base de datos.

```
19 inheritors  suarezKevin
@MappedSuperclass
@Getter
@Getter
public abstract class AbstractEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    Long id;
    @CreationTimestamp
    private Instant createdAt;
    @UpdateTimestamp
    private Instant updatedAt;
    @Column(columnDefinition = "boolean default true")
    private Boolean isActive;

    suarezKevin
    @PrePersist
    public void defaultActive(){
        if (isActive == null){
            isActive = Boolean.TRUE;
        }
    }
}
```

Figura 36. Clase Abstracta para las demas entidades

Figura 37. Clase para la entidad VaccineApplied

Para manejar los datos de la base de datos se utilizó la interfaz JpaRepository como se muestra en la Figura 38. JpaRepository contiene las operaciones CRUD básicas. Además, es posible crear métodos personalizados cuando es necesario elaborar alguna consulta compleja.

```java
import com.suarez.paquitofoundation.domain.entities.VaccineApplied;
import org.springframework.data.jpa.repository.JpaRepository;

import java.util.List;

5 usages  suarezKevin
public interface VaccineAppliedRepository extends JpaRepository<VaccineApplied, Long> {

    1 usage  suarezKevin
    List<VaccineApplied> findAllByAnimalcompanion_Id(Long animalCompanionId);

    1 usage  suarezKevin
    void deleteAllByAnimalcompanion_Id(Long animalId);

}
```

Figura 38. Interface que extiende de JpaRepository para la entidad VaccineApplied

Una vez terminado con la construcción de todas las entidades, se generó un diagrama entidad-relación como se presenta en la Figura 39.

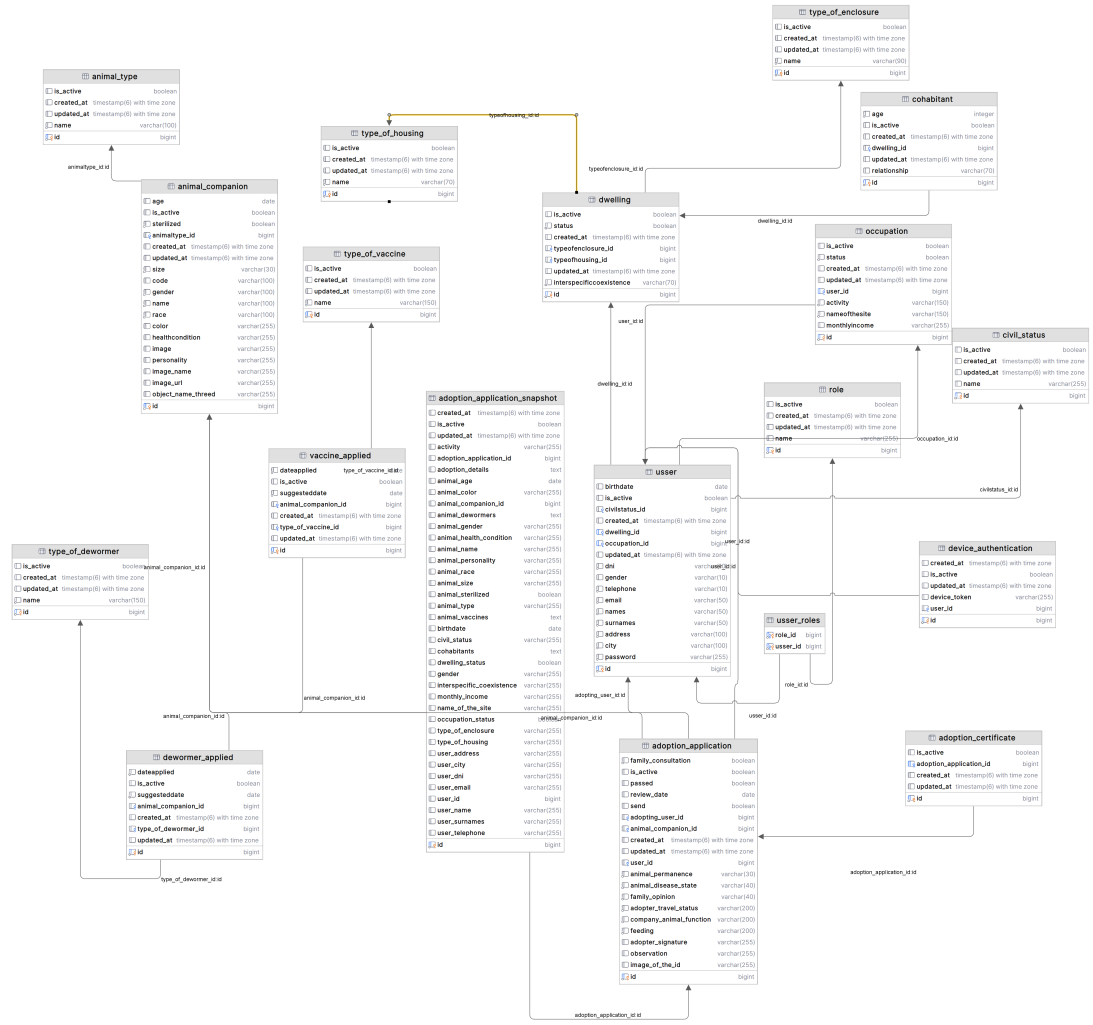


Figura 39. Diagrama entidad-relación de la base de datos

- ***Configuración del almacenamiento de archivos en S3.*** Como se presenta en la Figura 40, para el uso de los servicios de almacenamiento de AWS S3 se creó un bean de configuración, donde se requieren las credenciales de acceso del usuario a AWS y la región en donde se encuentra alojado el bucket de almacenamiento.

```java
@Configuration
public class S3Config {

    @Value("${ACCESS_KEY}")
    private String accessKey;

    @Value("${SECRET_KEY}")
    private String secretKey;

    suarezKevin
    @Bean
    public AmazonS3 initS3Client() {
        AWSCredentials credentials = new BasicAWSCredentials(accessKey, secretKey);
        return AmazonS3ClientBuilder.standard()
                .withRegion(Regions.US_EAST_2)
                .withCredentials(new AWSStaticCredentialsProvider(credentials))
                .build();
    }

}
```

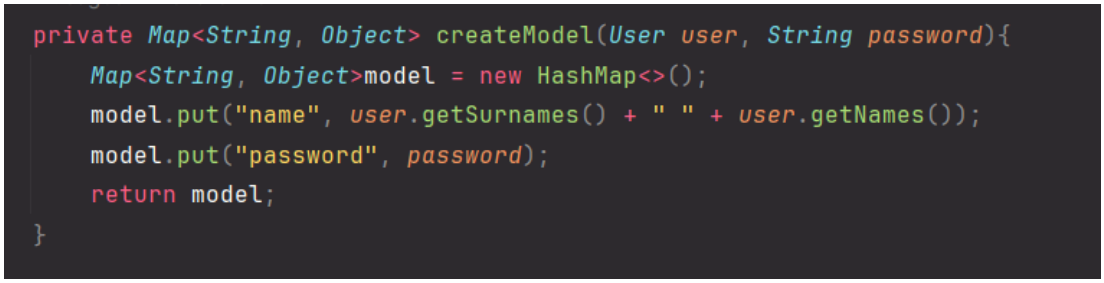
Figura 40. Configuración de los servicios AWS S3

Para el manejo de la carga y eliminación de archivos en el bucket de almacenamiento de AWS S3, se creó un servicio y se definió un método que recibe como parámetro el archivo a cargar. El método uploadFile como se presenta en la Figura 41 recibe el archivo a cargar para asignarle un nombre único y proceder a realizar una petición de carga al bucket mediante el método putObject de la instancia del bean inyectado AmazonS3. Por el lado del método de eliminación, recibe como parámetro el nombre del archivo a eliminar y mediante el método deleteObject se realiza la petición de eliminación.

```java
@Service
@RequiredArgsConstructor
public class FileServiceImpl implements IFileService {

    @Value("${BUCKET_NAME}")
    private String bucketName;

    private final AmazonS3 amazonS3;

    2 usages  suarezKevin *
    @Override
    public String uploadFile(MultipartFile multipartFile) throws IOException {
        //Convertir MultipartFile a File
        File file = new File(Objects.requireNonNull(multipartFile.getOriginalFilename()));
        try(FileOutputStream fos = new FileOutputStream(file)){
            fos.write(multipartFile.getBytes());
        }
        //Generate the file name
        String fileName = generateFileName(multipartFile);
        //Cargar el archivo a S3 bucket
        PutObjectRequest request = new PutObjectRequest(bucketName, fileName, file);
        ObjectMetadata metadata = new ObjectMetadata();
        metadata.setContentType(FilenameUtils.getExtension(multipartFile.getOriginalFilename()));
        metadata.addUserMetadata("Title", "File Upload-" + fileName);
        metadata.setContentLength(file.length());
        request.setMetadata(metadata);
        amazonS3.putObject(request);
        file.delete();
        return fileName;
    }

    suarezKevin
    @Override
    public void delete(String fileName) {
        if (fileName == null || fileName.isEmpty()) return;
        DeleteObjectRequest deleteObjectRequest = new DeleteObjectRequest(bucketName, fileName);
        amazonS3.deleteObject(deleteObjectRequest);
    }

    1 usage  suarezKevin
    private String generateFileName(MultipartFile multipartFile) {
        // Generar un UUID único
        String uuid = UUID.randomUUID().toString();
        // Obtener el nombre original del archivo y reemplazar los espacios en blanco con guiones bajos
        String originalFilename = multipartFile.getOriginalFilename().replaceAll( regex: " ",  replacement: "_");
        return uuid + "_" + Objects.requireNonNull(originalFilename);
    }

}
```

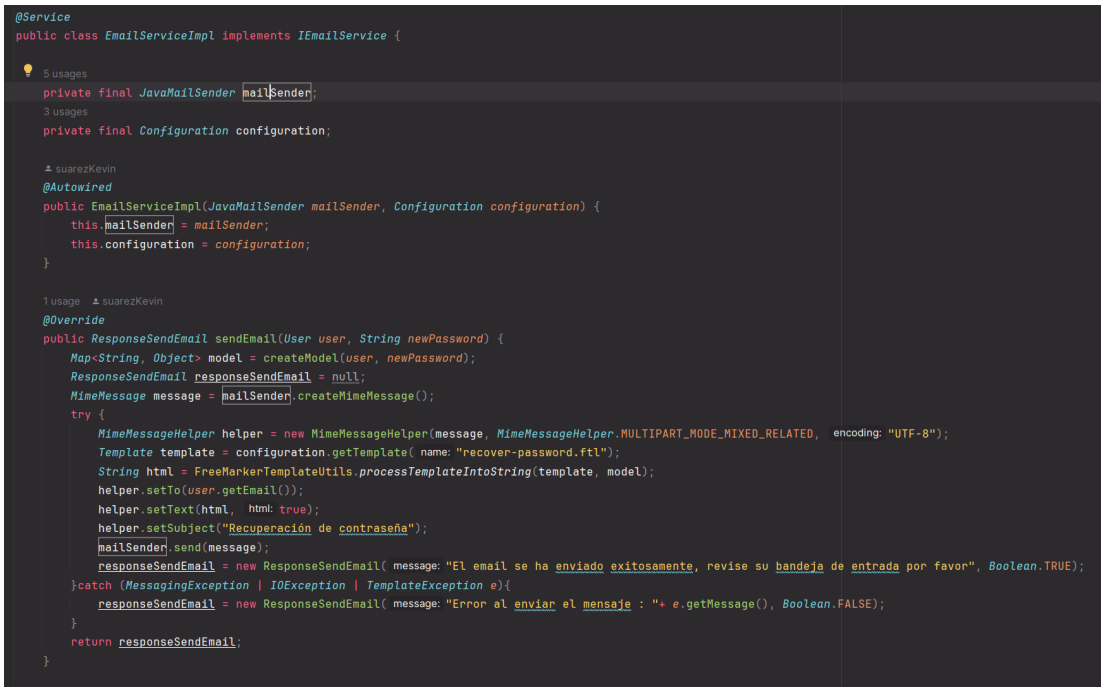
Figura 41. Servicio de carga y eliminación de archivo en S3

- ***Configuración del servicio de envío de correos electrónicos.*** Para enviar correos electrónicos a los usuarios en caso de recuperación de contraseña, se creó un servicio. Como se muestra en la Figura 42, se necesita crear un modelo para ser mapeado en la platilla HTML FreeMaker. Luego como se muestra en la Figura 43, se envía el correo medainte la instancia de la clase JavaMailSender con el método send.



```java
private Map<String, Object> createModel(User user, String password){
    Map<String, Object>model = new HashMap<>();
    model.put("name", user.getSurnames() + " " + user.getNames());
    model.put("password", password);
    return model;
}
```

Figura 42. Método para crear el módelo del correo electrónico



```java
@Service
public class EmailServiceImpl implements IEmailService {

    5 usages
    private final JavaMailSender mailSender;
    3 usages
    private final Configuration configuration;

    suarezKevin
    @Autowired
    public EmailServiceImpl(JavaMailSender mailSender, Configuration configuration) {
        this.mailSender = mailSender;
        this.configuration = configuration;
    }

    1 usage  suarezKevin
    @Override
    public ResponseSendEmail sendEmail(User user, String newPassword) {
        Map<String, Object> model = createModel(user, newPassword);
        ResponseSendEmail responseSendEmail = null;
        MimeMessage message = mailSender.createMimeMessage();
        try {
            MimeMessageHelper helper = new MimeMessageHelper(message, MimeMessageHelper.MULTIPART_MODE_MIXED_RELATED,  encoding: "UTF-8");
            Template template = configuration.getTemplate( name: "recover-password.ftl");
            String html = FreeMarkerTemplateUtils.processTemplateIntoString(template, model);
            helper.setTo(user.getEmail());
            helper.setText(html,  html: true);
            helper.setSubject("Recuperación de contraseña");
            mailSender.send(message);
            responseSendEmail = new ResponseSendEmail( message: "El email se ha enviado exitosamente, revise su bandeja de entrada por favor", Boolean.TRUE);
        }catch (MessagingException | IOException | TemplateException e){
            responseSendEmail = new ResponseSendEmail( message: "Error al enviar el mensaje : "+ e.getMessage(), Boolean.FALSE);
        }
        return responseSendEmail;
    }
```

Figura 43. Método de envío de correo electrónico para recuperar la contraseña

- ***Configuración de Firebase para el servicio de Firebase Cloud Messaging (FCM) y el servicio de envío de notificaciones.*** Para el envío de notificaciones push a los dispositivos móviles se utilizó el servicio de Firebase Cloud Messaging. Como se presenta en la Figura 44 se creó un bean de configuración para establecer las credenciales de acceso a Firebase. Las credenciales se obtuvieron de un archivo Json que se generó desde la consola de Firebase y luego se configuró en el archivo *application.properties* como se presentó anteriormente en la Figura 34.

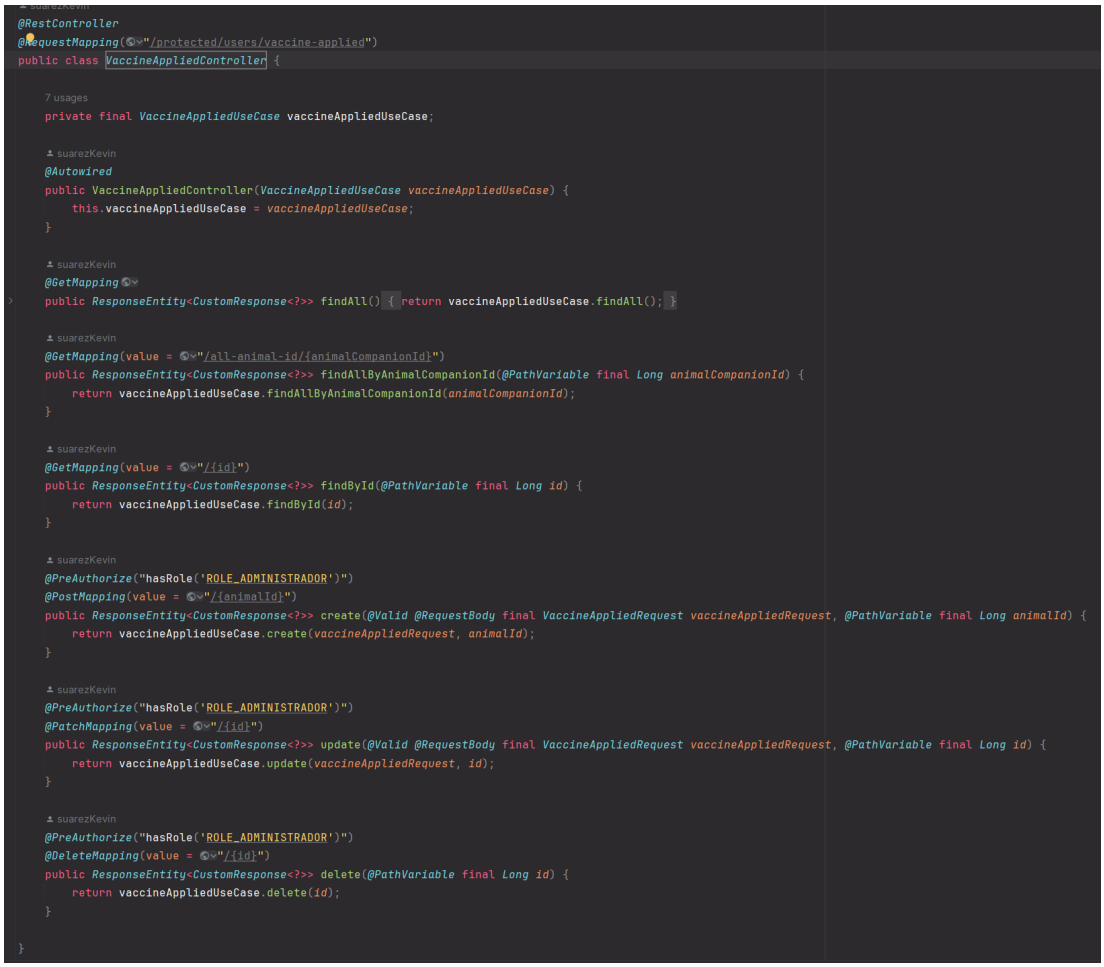
```java
@Configuration
@EnableConfigurationProperties(FirebaseProperties.class)
@RequiredArgsConstructor
public class FirebaseConfig {

    private final FirebaseProperties firebaseProperties;

    suarezKevin
    @Bean
    public GoogleCredentials googleCredentials(){
        try {
            if (firebaseProperties.getServiceAccount() != null) {
                try(InputStream is = firebaseProperties.getServiceAccount().getInputStream()){
                    return GoogleCredentials.fromStream(is);
                }
            }else{
                return GoogleCredentials.getApplicationDefault();
            }
        } catch (Exception ex){
            throw new RuntimeException("Error al cargar las credenciales de Firebase", ex);
        }
    }

    suarezKevin
    @Bean
    public FirebaseApp firebaseApp(GoogleCredentials googleCredentials){

        FirebaseOptions options = FirebaseOptions.builder()
                .setCredentials(googleCredentials)
                .build();
        if (FirebaseApp.getApps().isEmpty()) {
            return FirebaseApp.initializeApp(options);
        }
        return FirebaseApp.getInstance();

    }

    suarezKevin
    @Bean
    public FirebaseMessaging firebaseMessaging(FirebaseApp firebaseApp){
        return FirebaseMessaging.getInstance(firebaseApp);
    }
}
```

Figura 44. Configuración del servicio FCM de Firebase para las notificaciones push

Posterior a la creación del bean, se creó un servicio para definir los métodos de envío de notificaciones push, como el método para enviar una notificación a una persona en específico, para enviar al administrador, para enviar a varios dispositivos y para actualizar el token del dispositivo móvil como se presenta en la Figura 45.

Figura 45. Configuración del servicio FCM de Firebase para las notificaciones push

- ***Controladores.*** Los controladores, como se presenta en la Figura 46, son los encargados de exponer los endpoints de la aplicación para que los servicios sean consumidos por parte de los usuarios. Aquí se definieron las rutas y los métodos que se ejecutarán dependiendo del tipo de petición que se realice. Además, existen métodos que solo pueden ser ejecutados si cumplen con un rol en particular.



```
@RestController
@RequestMapping("/protected/users/vaccine-applied")
public class VaccineAppliedController {

    7 usages
    private final VaccineAppliedUseCase vaccineAppliedUseCase;

    suarezKevin
    @Autowired
    public VaccineAppliedController(VaccineAppliedUseCase vaccineAppliedUseCase) {
        this.vaccineAppliedUseCase = vaccineAppliedUseCase;
    }

    suarezKevin
    @GetMapping
    public ResponseEntity<CustomResponse<?>> findAll() { return vaccineAppliedUseCase.findAll(); }

    suarezKevin
    @GetMapping(value = "/all-animal-id/{animalCompanionId}")
    public ResponseEntity<CustomResponse<?>> findAllByAnimalCompanionId(@PathVariable final Long animalCompanionId) {
        return vaccineAppliedUseCase.findAllByAnimalCompanionId(animalCompanionId);
    }

    suarezKevin
    @GetMapping(value = "/{id}")
    public ResponseEntity<CustomResponse<?>> findById(@PathVariable final Long id) {
        return vaccineAppliedUseCase.findById(id);
    }

    suarezKevin
    @PreAuthorize("hasRole('ROLE_ADMINISTRADOR')")
    @PostMapping(value = "/{animalId}")
    public ResponseEntity<CustomResponse<?>> create(@Valid @RequestBody final VaccineAppliedRequest vaccineAppliedRequest, @PathVariable final Long animalId) {
        return vaccineAppliedUseCase.create(vaccineAppliedRequest, animalId);
    }

    suarezKevin
    @PreAuthorize("hasRole('ROLE_ADMINISTRADOR')")
    @PatchMapping(value = "/{id}")
    public ResponseEntity<CustomResponse<?>> update(@Valid @RequestBody final VaccineAppliedRequest vaccineAppliedRequest, @PathVariable final Long id) {
        return vaccineAppliedUseCase.update(vaccineAppliedRequest, id);
    }

    suarezKevin
    @PreAuthorize("hasRole('ROLE_ADMINISTRADOR')")
    @DeleteMapping(value = "/{id}")
    public ResponseEntity<CustomResponse<?>> delete(@PathVariable final Long id) {
        return vaccineAppliedUseCase.delete(id);
    }
}
```

Figura 46. Controlador de Vacunas aplicadas

b. Frontend

Sistema Administrativo

- ***Dependencias del sistema.*** Para el desarrollo de la app, donde se usó el Framework Flutter se instalaron varias dependencias. Las dependencias se encuentran definidas en el archivo de configuración *pubspec.yaml* como se presenta en la Figura 47

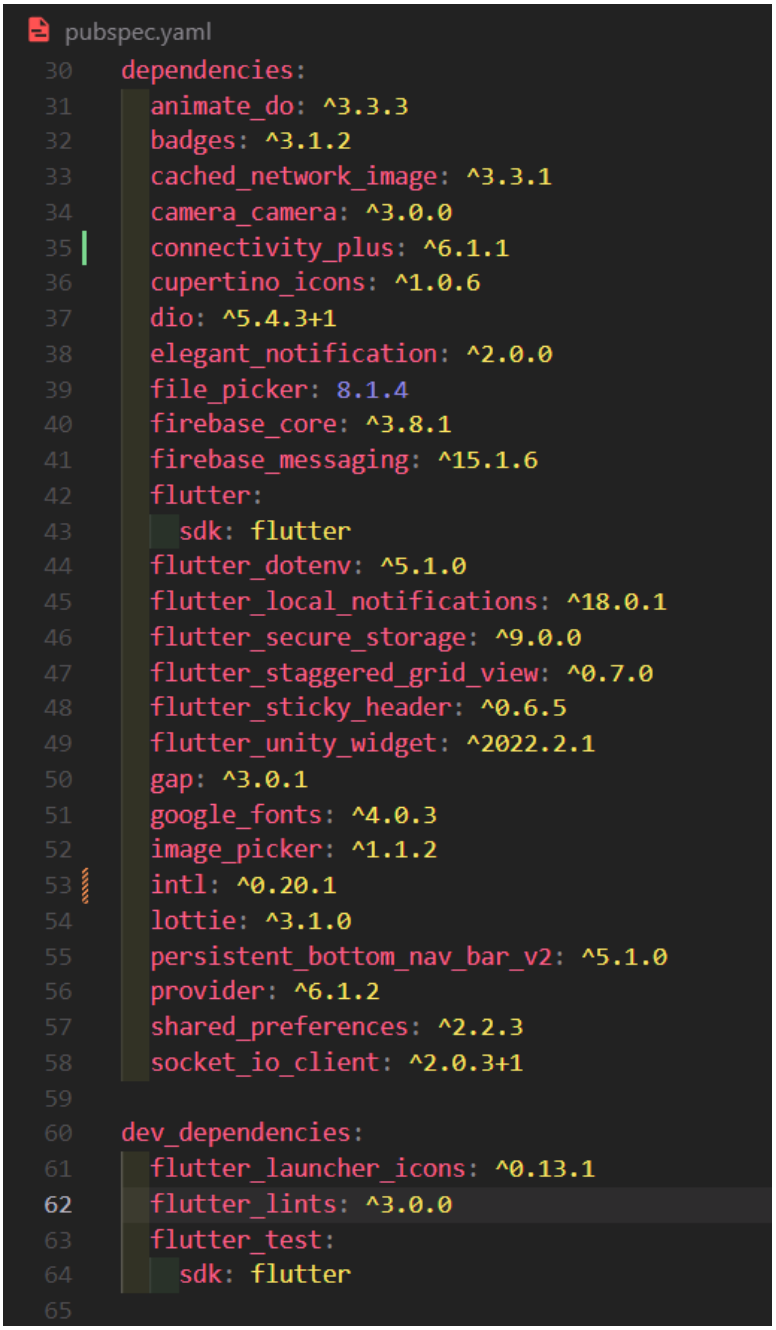


Figura 47. Archivo *pubspec.yaml* con las dependencias del sistema

- ***Configuración de la app.*** La app se desarrolló bajo la arquitectura MVVM (Model-View-ViewModel) como se presenta en la Figura 48. MVVM se encarga de separar la lógica de negocio y la interfaz de usuario, ofreciendo un código más limpio, estructurado y fácil de mantener. MVVM se estructura en tres módulos (carpetas), donde Model (Repository) representa los datos y la lógica de negocio de la aplicación como se presenta en la Figura 49, además, de que interactua con las fuentes de datos como APIs. View (Screens) representa la interfaz de usuario y su responsabilidad

es mostrar los datos que recibe del ViewModel y responder a las interacciones del usuario, enviándolas al ViewModel como se presenta en la Figura 50. Por úlitmo, está el ViewModel que conecta el Model con la View como se presenta en la Figura 51, responsable de transformar los datos del Model para que sean fáciles de usar en la vista, además, de proveer un mecanismo reactivo (ChangeNotifier) para actualizar la View cuando los datos cambian.

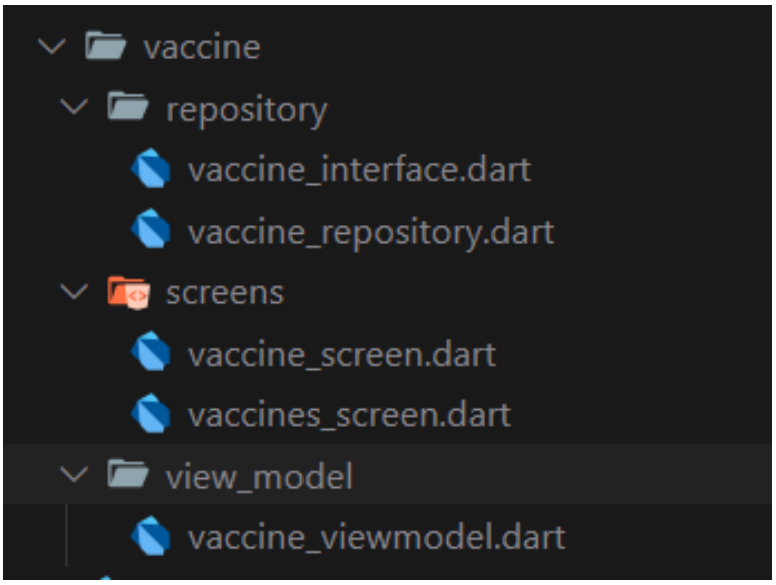


Figura 48. Arquitectura MVVM para la entidad Vacuna

```dart
class VaccineRepository extends VaccineInterface {
  @override
  Future create(Vaccine vaccine) async {
    final response = await Api.post(EndPoints.vaccine, vaccine.toJson());
    return response;
  }

  @override
  Future findAll() async {
    final response = await Api.httpGet(EndPoints.vaccine);
    return response;
  }

  @override
  Future findById(String vaccineId) async {
    final response = await Api.getWithParameter(EndPoints.vaccine, vaccineId);
    return response;
  }

  @override
  Future update(String vaccineId, Vaccine vaccine) async {
    final response = await Api.patchWithParameter(
        EndPoints.vaccine, vaccineId, vaccine.toJson());
    return response;
  }

  @override
  Future reactivate(String vaccineId) async {
    final response =
        await Api.patchWithParameter(EndPoints.reactivateVaccine, vaccineId);
    return response;
  }

  @override
  Future remove(String vaccineId) async {
    final response = await Api.deleteById(EndPoints.vaccine, vaccineId);
    return response;
  }
```

Figura 49. Repository para Vacuna

```
  Widget build(BuildContext context) {
                                  style: textStyles.titleMedium), // Text
                              ), // Padding
                              const SizedBox(height: 25),
                              SizedBox(
                                height: 70,
                                child: TextInputField(
                                  textInputAction: TextInputAction.done,
                                  textIsEmpty: 'El campo es requerido',
                                  lenLimitTextInpFmt: 100,
                                  suffixIcon: Icons.vaccines_outlined,
                                  label: 'Nombre',
                                  keyboardType: TextInputType.text,
                                  inputController: nameController,
                                ), // TextInputField
                              ), // SizedBox
                              const SizedBox(height: 25),
                              PrimaryButton(
                                isLoading:
                                    context.watch<VaccineViewModel>().vaccineIn,
                                text: (vaccine.id != null) ? 'Editar' : 'Crear',
                                onTap: () async {
                                  if (vaccine.id != null) {
                                    handleVaccineToEdit(
                                        (vaccine.id).toString());
                                  } else {
                                    handleNewVaccine();
                                  }
                                },
                                outlined: false,
                                color: AppColors.black,
                              ), // PrimaryButton
                              // const Spacer(flex: 1),
                              // const Spacer(flex: 1),
                            ],
                          ), // Column
                        ), // Form
                      ), // Padding
                elevation: 0,
                leading: IconButton(
                  padding: const EdgeInsets.only(left: 30),
                  iconSize: 40,
  Widget build(BuildContext context) {
                                  lenLimitTextInpFmt: 100,
                                  suffixIcon: Icons.vaccines_outlined,
                                  label: 'Nombre',
                                  keyboardType: TextInputType.text,
                                  inputController: nameController,
                                ), // TextInputField
                              ), // SizedBox
                              const SizedBox(height: 25),
                              PrimaryButton(
                                isLoading:
                                    context.watch<VaccineViewModel>().vaccineIn,
                                text: (vaccine.id != null) ? 'Editar' : 'Crear',
                                onTap: () async {
                                  if (vaccine.id != null) {
                                    handleVaccineToEdit(
                                        (vaccine.id).toString());
                                  } else {
                                    handleNewVaccine();
                                  }
                                },
                                outlined: false,
                                color: AppColors.black,
                              ), // PrimaryButton
                              // const Spacer(flex: 1),
                              // const Spacer(flex: 1),
                            ],
                          ), // Column
                        ), // Form
                      ), // Padding
                    ), // SizedBox
                  ],
                ), // Column
              ), // SingleChildScrollView
            )), // SafeArea // Scaffold
      ); // GestureDetector
    }
  }
```

Figura 50. View para Vacuna

Figura 51. ViewModel para Vacuna

Por otro lado está la configuración del ChangeNotifierProvider, el cual se encargará de manejar y compartir el estado de la app. Los ChangeNotifierProvider se establecieron en el archivo *main.dart* que es la raiz de la app como se presenta en la Figura 52.

Figura 52. Configuración de los ChangeNotifierProvider

También se estblecierón los permisos en el archivo *AndroidManifest.xml* que se requieren del dispositivo móvil como el permiso de usar la cámara, permiso de lectura y escritura del almacenamiento externo como se presenta en la Figura 53.

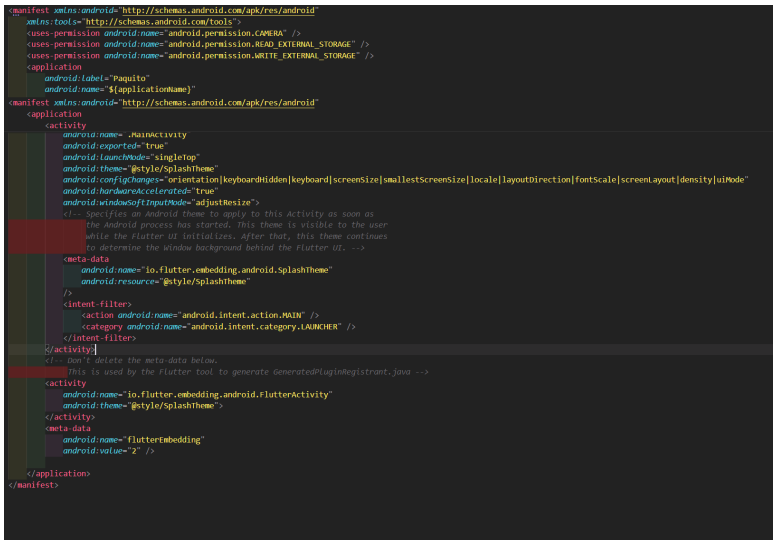


Figura 53. Configuración de permisos en el archivo *AndroidManifest.xml*

Para el enrutamiento de las rutas se centralizó en un solo archivo, usando un mapa de rutas para asociar nombres de rutas con las respectivas pantallas como se muestra en la Figura 54.

```dart
class RouteGenerator {
  static Route<dynamic> generateRoute(RouteSettings settings) {
    final Map<String, dynamic Function(BuildContext)> routes = {
      LoginScreen.id: (_) => const LoginScreen(),
      RegisterScreen.id: (_) => const RegisterScreen(),
      RecoverPasswordScreen.id: (_) => const RecoverPasswordScreen(),
      AnimalCompanionsScreen.id: (_) => const AnimalCompanionsScreen(),
      RegisterAnimalScreen.id: (_) => const RegisterAnimalScreen(),
      DewormersScreen.id: (_) => const DewormersScreen(),
      DewormerScreen.id: (_) => const DewormerScreen(),
      VaccinesScreen.id: (_) => const VaccinesScreen(),
      VaccineScreen.id: (_) => const VaccineScreen(),
      AnimalTypesScreen.id: (_) => const AnimalTypesScreen(),
      AnimalTypeScreen.id: (_) => const AnimalTypeScreen(),
      UsersScreen.id: (_) => const UsersScreen(),
      UserScreen.id: (_) => const UserScreen(),
      PetsScreen.id: (_) => const PetsScreen(),
      PetProfileScreen.id: (_) => const PetProfileScreen(),
      Unity3DModelScreen.id: (_) {
        final args = settings.arguments as Map<String, dynamic>;
        final threeDObject = args['threeDObject'] as ThreeDObject;
        final onUpdateCallback = args['onUpdateCallback'] as VoidCallback?;
        return Unity3DModelScreen(
          threeDObject: threeDObject,
          onUpdateCallback: onUpdateCallback,
        );
      },
      AdoptionFormScreen.id: (_) {
        if (settings.arguments != null) {
          final args = settings.arguments as Map<String, dynamic>;
          final application = args['application'] as AdoptionApplication;
          return AdoptionFormScreen(
            application: application,
          );
        }
        return const AdoptionFormScreen();
      },
      DocumentsScreen.id: (_) => const DocumentsScreen(),
      ApplicationScreen.id: (_) => const ApplicationScreen(),
      AdoptionRecordScreen.id: (_) => const AdoptionRecordScreen(),
      CertificateScreen.id: (_) {
        final args = settings.arguments as Map<String, dynamic>;
        final application = args['application'] as AdoptionApplication;
        return CertificateScreen(
          application: application,
        );
      },
      BottomNavyBar.id: (_) => const BottomNavyBar()
    };

    final routeBuilder = routes[settings.name];
    if (routeBuilder != null) {
      return MaterialPageRoute(
        builder: (context) => routeBuilder(context),
        settings: settings,
      );
    }

    return MaterialPageRoute(builder: (context) => const LoginScreen());
  }
}
```

Figura 54. Configuración de rutas para el sistema

Para las solicitudes a la API, con el uso del paquete Dio se configuró y centralizó en un solo archivo. Se definió varios métodos para las peticiones de tipo GET, POST, UPDATE, PATCH y DELETE, considerando los parámetros y cuerpos JSON que son requeridos para que las peticiones se ejecuten sin problemas Figura 55.

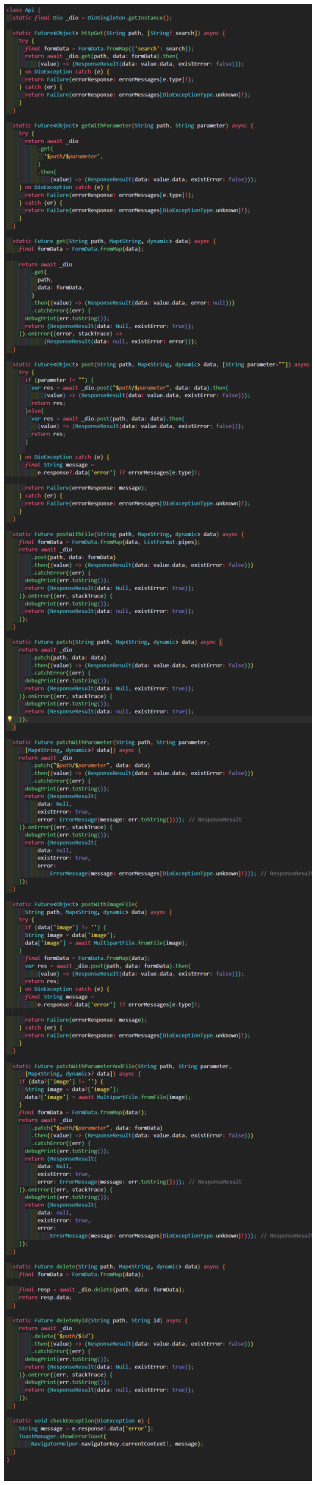


Figura 55. Configuración de los métodos para las peticiones a la API

- ***Inicio de sesión.*** Para que el usuario pueda iniciar sesión requiere ingresar sus credenciales. Las credenciales son tomadas y enviadas al ViewModel, donde se realiza el llamado al método del repositorio para que se encargue de la petición a la API y espera a por una respuesta. En caso de ser válidas la credenciales, se genera un

token JWT, el cual se guarda en el dispositivo para ser utilizado en todas las peticiones que lo requieran.

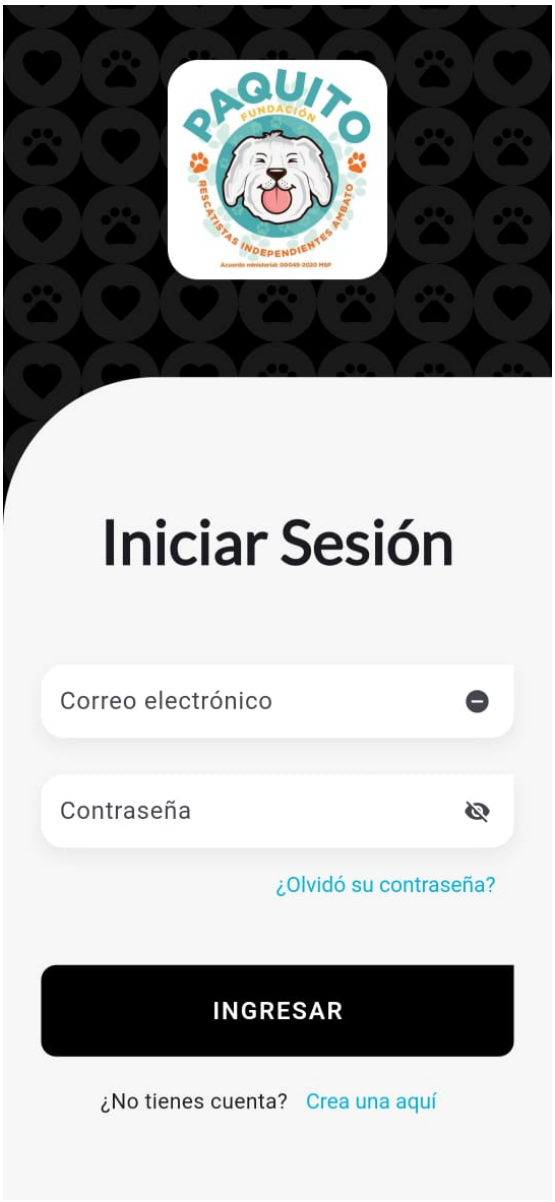


Figura 56. Interfaz de inicio de sesión

- ***Registro de usuario adoptante.*** El usuario adoptante debe registrarse en la app completando un formulario. Los datos del usuario son tomados y enviados al ViewModel, donde se realiza el llamado al método del repositorio para que se encargue de la petición a la API y espera a por una respuesta. Una vez registrado el usuario es llevado a la pantalla principal de la app como si hubiera iniciado sesión.

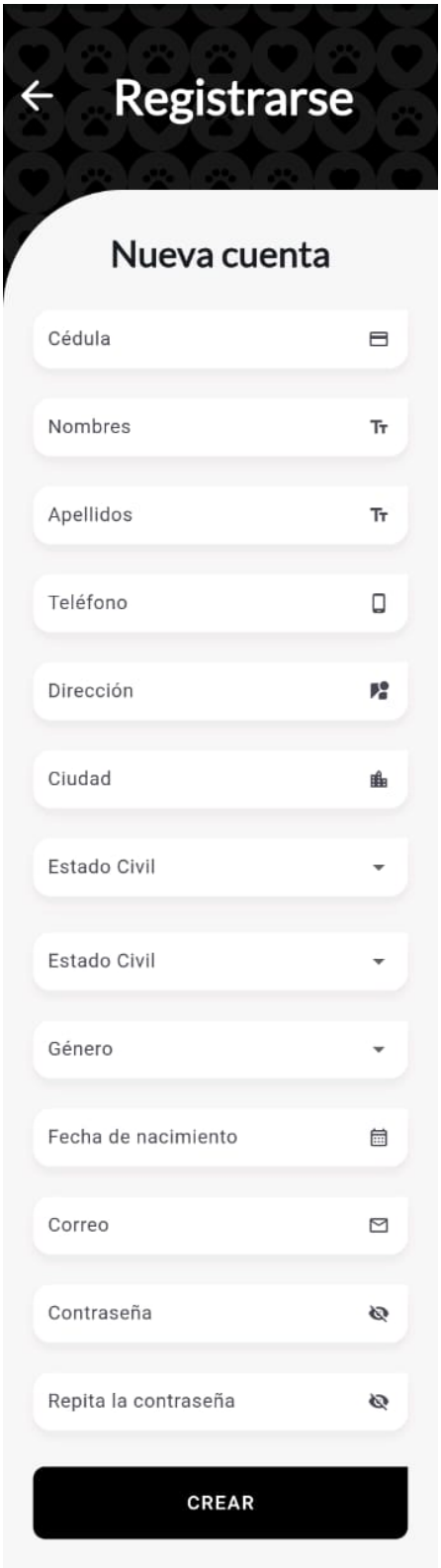


Figura 57. Interfaz de registro de usuario adoptante

- ***Recuperación de contraseña.*** El usuario debe ingresar su correo electrónico con el que se registro en la app. El correo será tomado y enviado al ViewModel, donde se

realiza el llamado al método del repositorio para que se encargue de la petición a la API y espera a por una respuesta. El usuario podrá revisar la bandeja de entrada de su correo y ver la nueva contraseña.

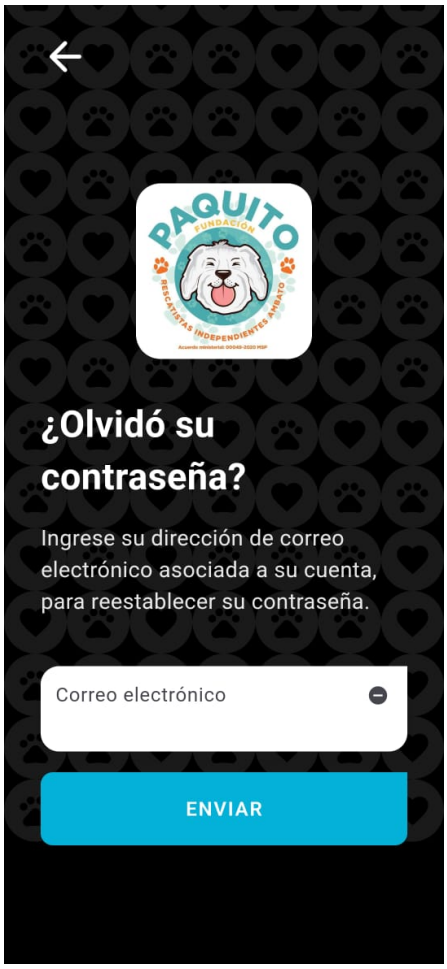


Figura 58. Interfaz para recuperar contraseña

- ***Gestión de usuarios.*** El usuario administrador puede ver la información de los usuarios registrados. Además, puede registrar un nuevo usuario y asignarle un rol. Los usuarios son cargados desde dos métodos diferentes, el primero trae una lista de usuarios activos y el segundo trae una lista de usuarios no activos.

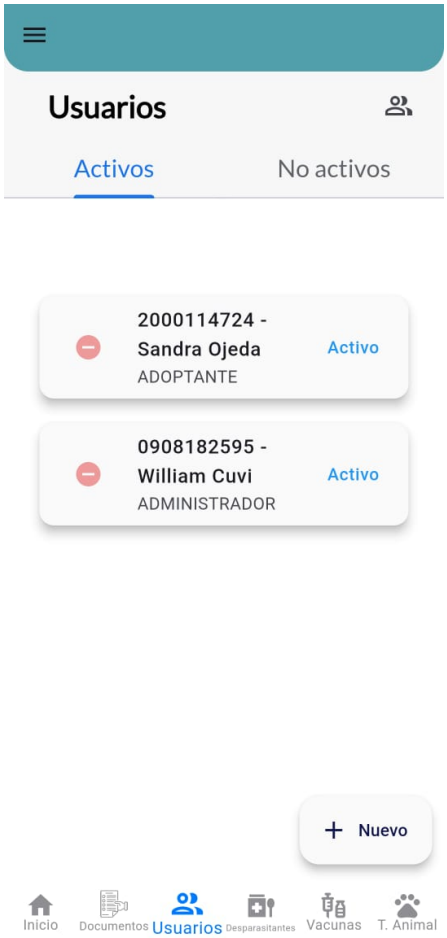


Figura 59. Interfaz para gestionar los usuarios

- ***Gestión de desparasitantes.*** El usuario administrador puede ver la información de los desparasitantes registrados. Además, puede registrar, editar, desactivar y activar un desparasitante. Los desparasitantes son cargados desde un método y se almacenan en una lista con los datos procesados para mostrar en pantalla.

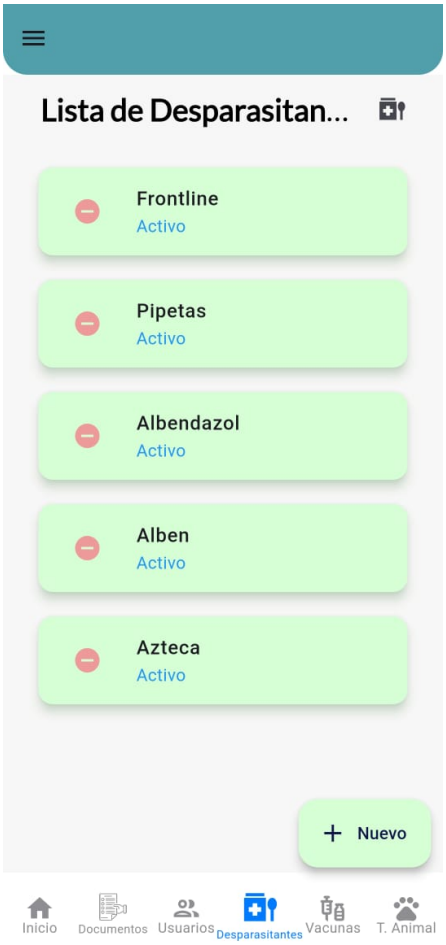


Figura 60. Interfaz para gestionar los usuarios

- ***Gestión de vacunas.*** El usuario administrador puede ver la información de las vacunas registradas. Además, puede registrar, editar, desactivar y activar una vacuna. Las vacunas son cargadas desde un método y se almacenan en una lista con los datos procesados para mostrar en pantalla.

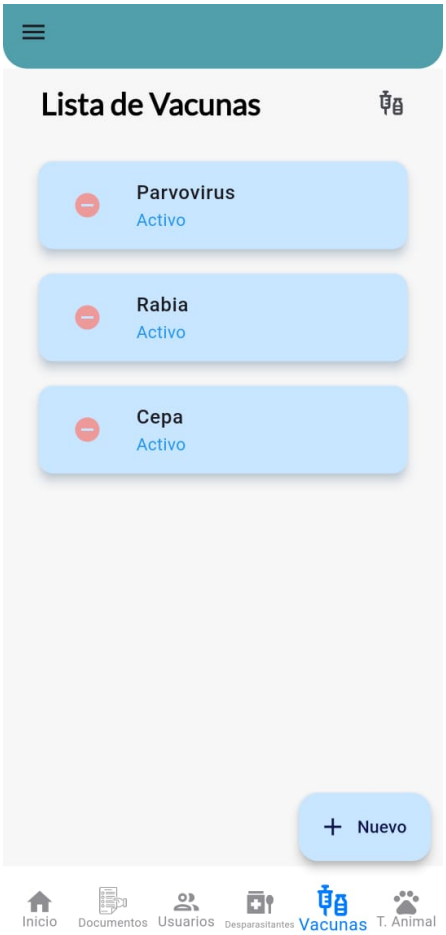


Figura 61. Interfaz para gestionar las vacunas

- ***Gestión de tipos de animales.*** El usuario administrador puede ver la información de los tipos de animales. Además, puede registrar, editar, desactivar y activar un tipo de animal. Los tipos de animales son cargadas desde un método y se almacenan en una lista con los datos procesados para mostrar en pantalla.

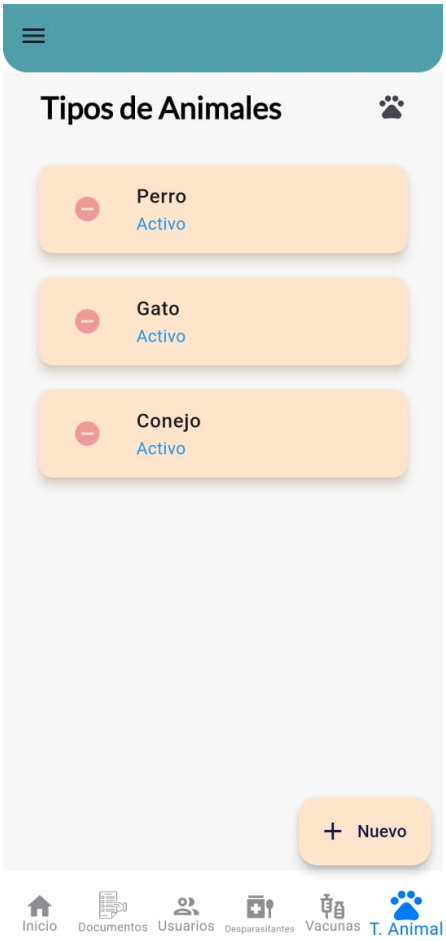


Figura 62. Interfaz para gestionar los tipos de animales

- ***Gestión de animales de compañía.*** El usuario administrador puede ver la información de los animales de compañía. Además, puede registrar, editar, desactivar y activar un animal de compañía. Los animales de compañía son cargados por dos métodos diferentes, el primero carga una lista de los animales activos y el segundo carga una lista con los animales no activos. Ambas listas contienen los datos procesados a mostrar en pantalla.

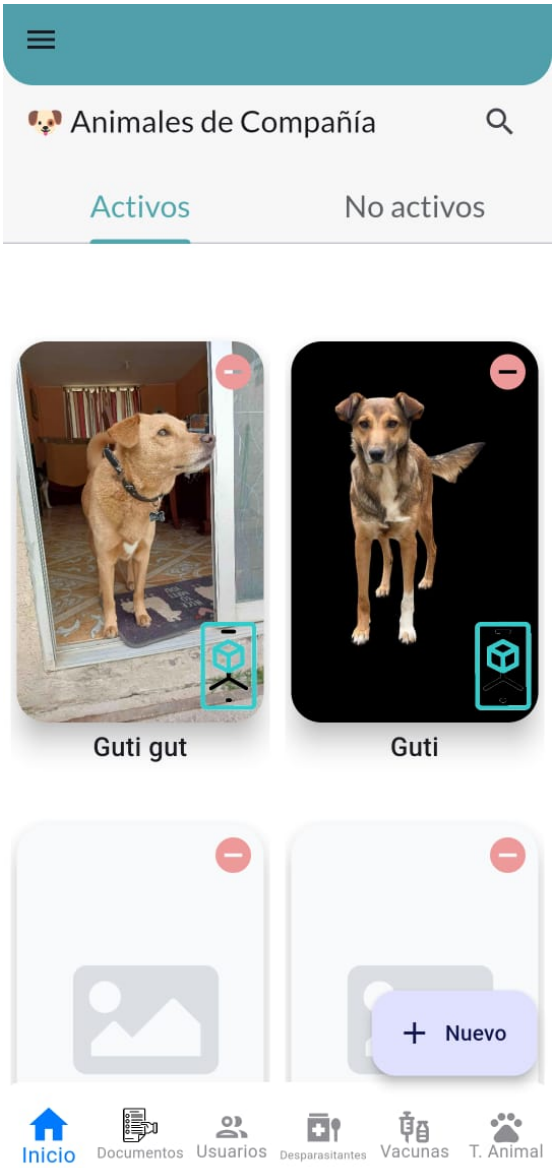


Figura 63. Interfaz de inicio para gestionar el animal de compañía

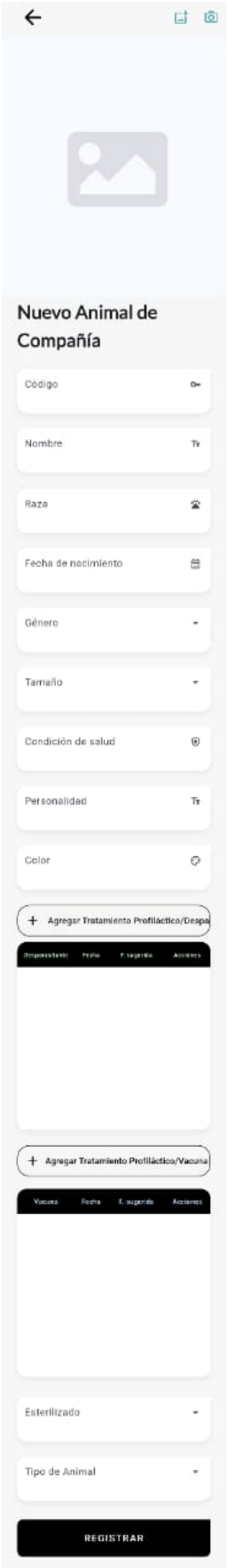


Figura 64. Interfaz para registrar un animal de compañía

- ***Búsqueda de animales de compañía.*** El usuario administrador como el usuario adoptante pueden buscar animales de compañía utilizando el ícono de la lupa. Los parámetros de búsqueda incluyen nombre, género, tamaño, color y raza. Los resultados de la búsqueda se muestran en una lista filtrada que cumple con los criterios especificados.

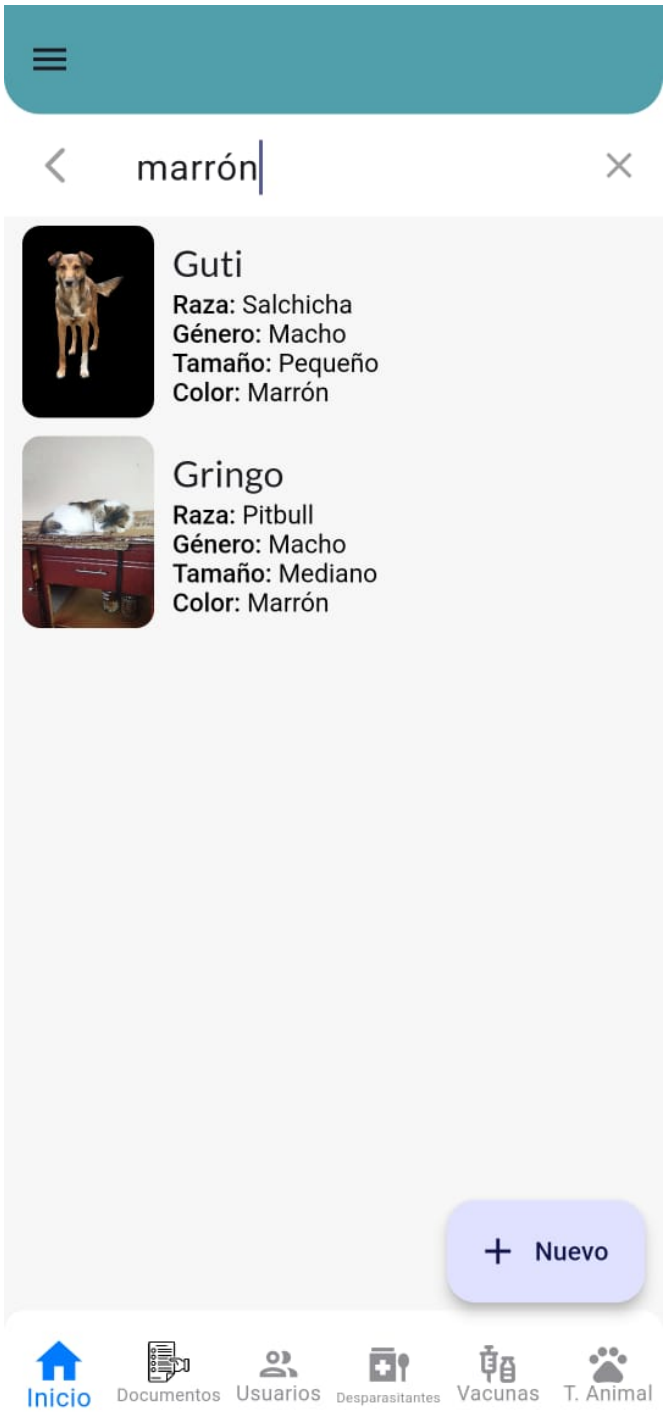


Figura 65. Interfaz para registrar un animal de compañía

En la gestión de animales de compañía también se maneja la proyección de modelos 3D mediante Realidad Aumentada, lo cual se explica a continuación.

- ***Configuración y desarrollo de la app para Realidad Aumentada.*** Para construir la app de Realidad Aumentada se usó el editor de desarrollo de juegos y aplicaciones Unity, específicamente la versión 2022.3.40f1. Primero se creó el proyecto y se instalaron los paquetes necesarios para el desarrollo de la app como se muestra en la Figura 66.

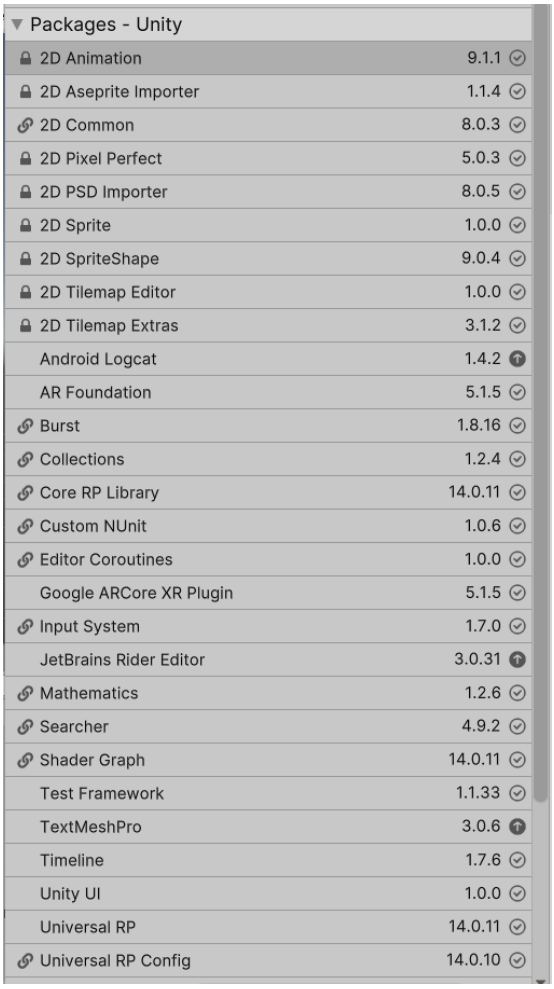


Figura 66. Paquetes instalados en el proyecto Unity

En la interfaz de “Project Settings -> Player” en la pestaña de Android se configuró el proyecto con los siguientes parámetros como se muestra en la Figura 67.

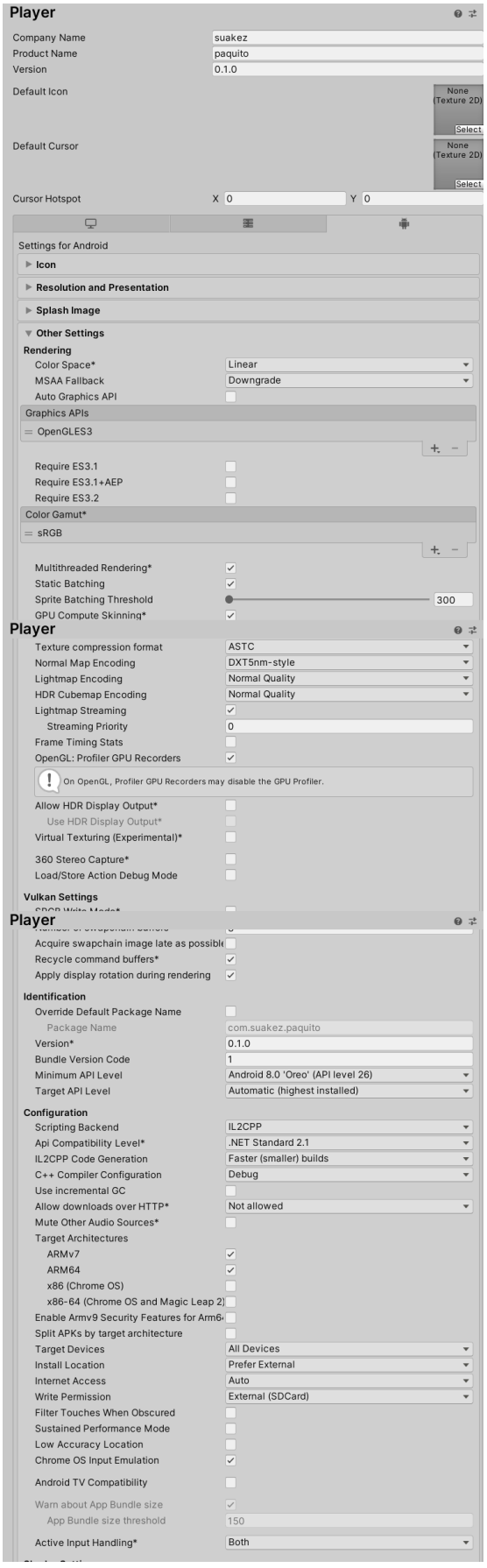


Figura 67. Configuración del proyecto Unity en el Project Settings

También en el “Project Settings -> Player” en la pestaña Publishing Settings se marcó las siguientes casillas para generar los archivos de configuración de Android como el AndroidManifest.xml, entre otros Figura 68.

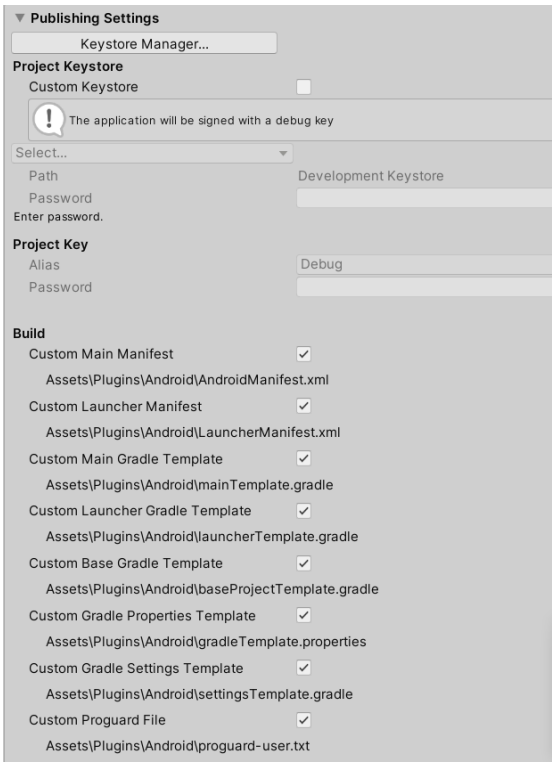


Figura 68. Ajustes para generar los archivos de configuración de Android

Arquitectura. La arquitectura de la app Unity está basada en el patrón evento. El patrón evento consiste en la interacción entre los publicadores y suscriptores. En el lado de los publicadores se emiten eventos y del lado de los suscriptores hay métodos que están suscritos a uno o más eventos. Entonces, cuando se dispara un evento, este notifica y llama a ejecutar el código que está suscrito a dicho evento como se representa en la Figura 69.

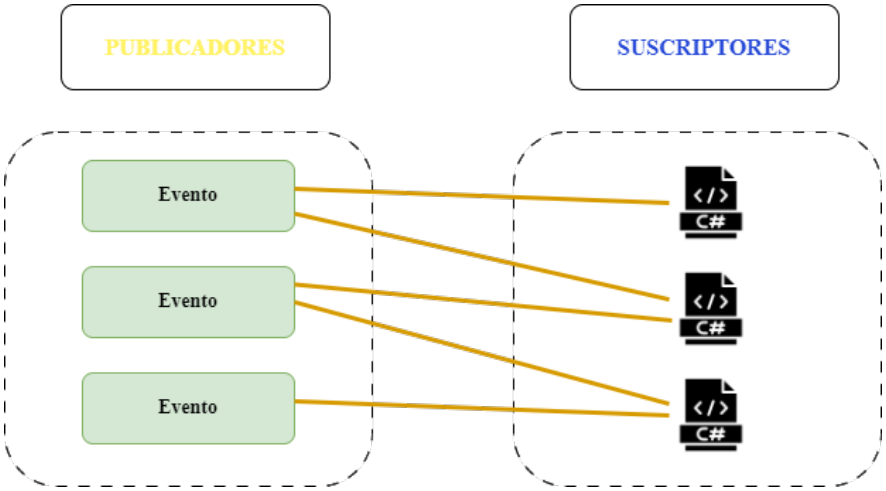


Figura 69. Representación del patrón evento

Por tanto, la app se dividio en tres eventos como se muestra en la Figura 70 que son el OnMainMenu, el OnItemMenu y el OnARPosition. El OnMainMenu es donde inicia la app y se pueden ver los modelos 3D fijados en Realidad Aumentada. El OnItemMenu es donde se observa el modelo 3D disponible para ser seleccionado. Por último, el OnARPosition es para posicionar el modelo 3D en el entorno real.

```
public class GameEventManager : MonoBehaviour
{
    // Crear los 3 eventos que representaran los estados de nuestra app

    3 references
    public event Action OnMainMenu;
    5 references
    public event Action OnItemMenu;
    2 references
    public event Action OnARPosition;

    //Patron Singleton para accerder a los eventos y poder suscribirse
    14 references
    public static GameEventManager instance;
    0 references
    private void Awake()
    {
        if (instance != null && instance != this)
        {
            Destroy(gameObject);
        }
        else
        {
            instance = this;
        }
    }

    0 references
    void Start()
    {
        MainMenu();
    }

    // Signo de pregunta constata que existe algo suscrito a este evento
    3 references
    public void MainMenu()
    {
        OnMainMenu?.Invoke();
        Debug.Log("Main Menu Activated");
    }

    0 references
    public void ItemMenu()
    {
        OnItemMenu?.Invoke();
        Debug.Log("Item Menu Activated");
    }

    2 references
    public void ARPosition()
    {
        OnARPosition?.Invoke();
        Debug.Log("AR Position Activated");
    }
}
```

Figura 70. Script con la arquitectura de eventos

Se creó una interfaz para cada evento usando los Canvas propios del editor de Unity como se presentan en las siguientes Figuras 72, 73 y 74. Sin embargo, para las animaciones de las interfaces se usó el paquete DOTween como se presenta en la Figura 71.

```
public class UIManager : MonoBehaviour
{
    // Crear los campos de los GameObjects que contiene nuestra interfaz
    12 references
    [SerializeField] private GameObject mainMenuCanvas;
    6 references
    [SerializeField] private GameObject itemMenuCanvas;
    4 references
    [SerializeField] private GameObject ARPositionCanvas;

    //Crear las funciones que van a estar suscritas a los eventos de nuestro GameManager
    0 references
    void Start()
    {
        GameEventManager.instance.OnMainMenu += ActivateMainMenu;
        GameEventManager.instance.OnItemMenu += ActivateItemMenu;
    }

    // En estas funciones es donde se comienzan a animar nuestra interfaz
    1 reference
    private void ActivateMainMenu()
    {
        mainMenuCanvas.transform.GetChild(1).transform.DOScale(new Vector3(1, 1, 1), 0.3f);
        mainMenuCanvas.transform.GetChild(2).transform.DOScale(new Vector3(1, 1, 1), 0.3f);
        mainMenuCanvas.transform.GetChild(3).transform.DOScale(new Vector3(1, 1, 1), 0.3f);
        // Ocultar los botones de ItemsMenu
        itemMenuCanvas.transform.GetChild(0).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        itemMenuCanvas.transform.GetChild(1).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        // Ocultar los botones de ARPosition
        ARPositionCanvas.transform.GetChild(0).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        ARPositionCanvas.transform.GetChild(1).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
    }
    1 reference
    private void ActivateItemMenu()
    {
        mainMenuCanvas.transform.GetChild(0).transform.DOScale(new Vector3(0, 0, 0), 0.3f);//3 segundos
        mainMenuCanvas.transform.GetChild(1).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        mainMenuCanvas.transform.GetChild(2).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        mainMenuCanvas.transform.GetChild(3).transform.DOScale(new Vector3(0, 0, 0), 0.3f);

        itemMenuCanvas.transform.GetChild(0).transform.DOScale(new Vector3(1, 1, 1), 0.3f);
        itemMenuCanvas.transform.GetChild(1).transform.DOScale(new Vector3(1, 1, 1), 0.3f);
    }

    1 reference
    private void ActivateARPosition()
    {
        mainMenuCanvas.transform.GetChild(0).transform.DOScale(new Vector3(0, 0, 0), 0.3f);//3 segundos
        mainMenuCanvas.transform.GetChild(1).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        mainMenuCanvas.transform.GetChild(2).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        mainMenuCanvas.transform.GetChild(3).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        // Ocultar los botones de ItemMenu
        itemMenuCanvas.transform.GetChild(0).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        itemMenuCanvas.transform.GetChild(1).transform.DOScale(new Vector3(0, 0, 0), 0.3f);
        // Ocultar los botones de ARPosition
        ARPositionCanvas.transform.GetChild(0).transform.DOScale(new Vector3(1, 1, 1), 0.3f);
        ARPositionCanvas.transform.GetChild(1).transform.DOScale(new Vector3(1, 1, 1), 0.3f);
    }
```

Figura 71. Script para controlar el comportamiento de la interfaz de usuario

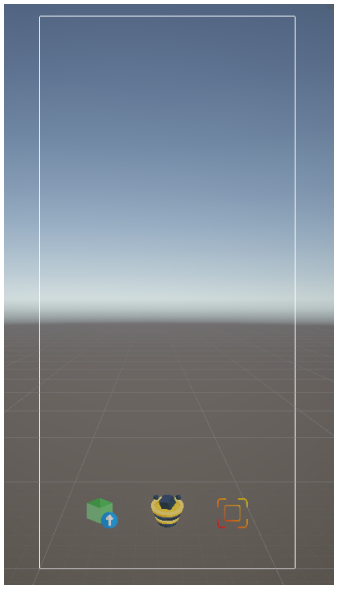


Figura 72. Canvas para la interfaz de OnMainMenu

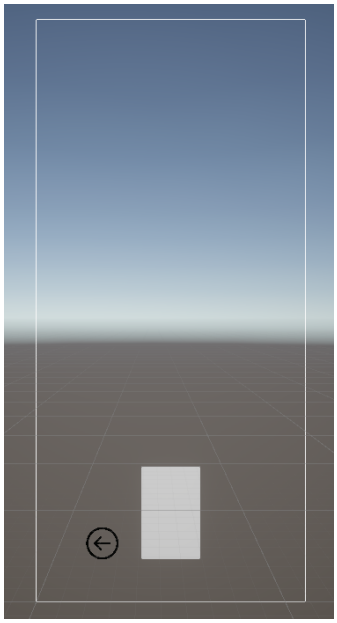


Figura 73. Canvas para la interfaz de OnItemMenu

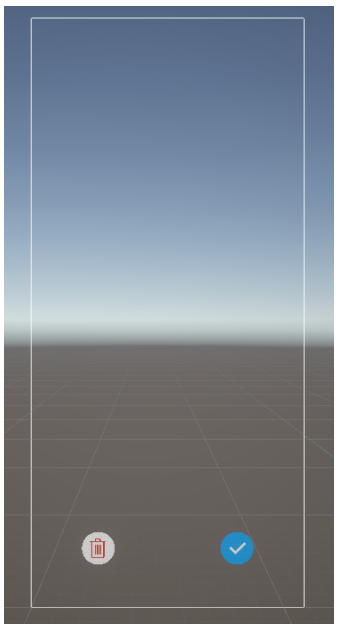


Figura 74. Canvas para la interfaz de OnARPosition

Configuración de detección de superficies y rastreo de planos. Para detectar superficies y rastrear planos, se usaron los paquetes AR Foundation y Gloogle ARCore, los cuales fueron instalados en el inicio de la configuración como se mencionó en la Figura 66. Para implementar tales características, se usaron los componentes AR Session y XR Origin del paquete AR Foundation poniendolos en la escena del editor de Unity como se muestra en la Figura 75. ARSession es el componente que se encarga de controlar el ciclo de vida de una experencia de

Realidad Aumentada (AR) en una plataforma, es decir, habilita o deshabilita la AR en la plataforma. Por otro lado, XR Origin, se encarga de actualizar constantemente la posición y rotación del usuario en el mundo virtual, además, permite colocar elementos virtuales en el espacio físico, asegurándose que estén correctamente alineados con el entorno mediante la detección de superficies.

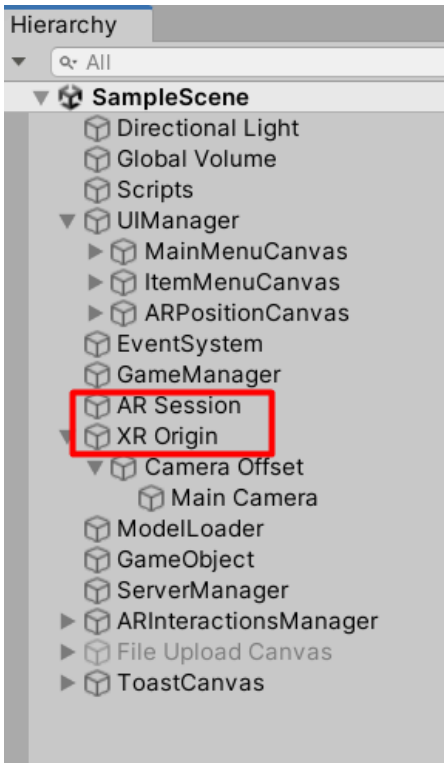


Figura 75. Uso de los componentes ARSession y XR Origin en la escena

También se implemento un sistema de nube de puntos en el componente XR Origin, el cual sirve para que el usuario pueda visualizar a través de la cámara de su dispositivo móvil las características rastreables del entorno, como se muestra en la Figura 76. Además, se configuró solo la detección de planos horizontales.

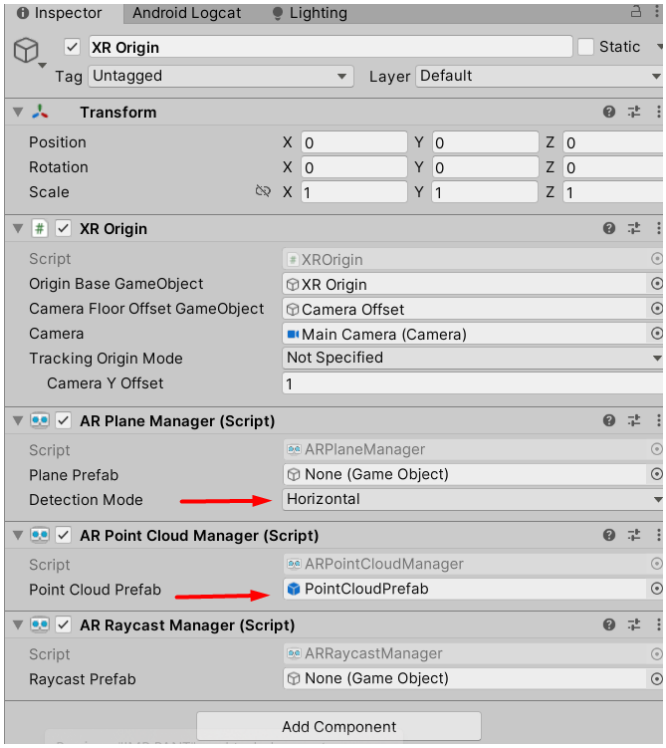


Figura 76. Configuración de nube de puntos y detección de planos horizontales

Configuración de la carga del archivo del modelo 3D. Para cargar el archivo del modelo 3D se usó el servicio de Firebase Cloud Storage. Se descargó el SDK de Firebase para Unity y se instaló el paquete como se muestra en la Figura 77.

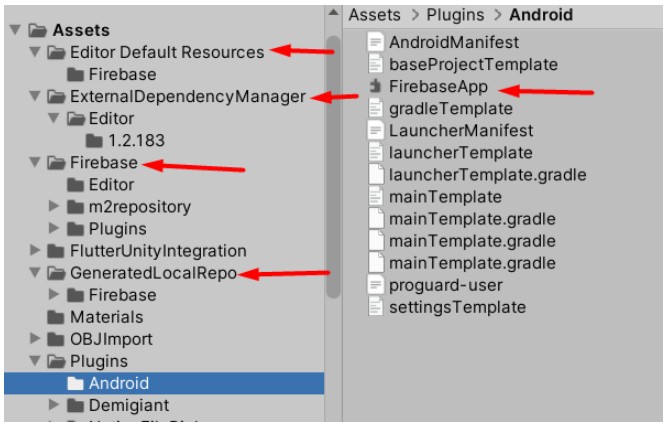


Figura 77. Instalación del SDK de Firebase para Unity

Para empezar a usar el servicio de Firebase, se inicializó la instancia de FirebaseStorage y posteriormente se configuró el botón que se encargará de abrir el administrador de archivos del dispositivo móvil para que se pueda seleccionar el archivo (tiene que ser un archivo .zip) como se muestra en la Figura 78.

```
private void Start()
{
    if (gifUploadImage != null)
    {
        animator = gifUploadImage.GetComponent<Animator>();
        animator.SetBool("isAnimating", false);
    }
    storage = FirebaseStorage.DefaultInstance;
    storageRef = storage.GetReferenceFromUrl("[redacted]");

    Button uploadButton = loadButton.GetComponent<Button>();
    uploadButton.onClick.AddListener(CheckAndRequestPermission);
}
```

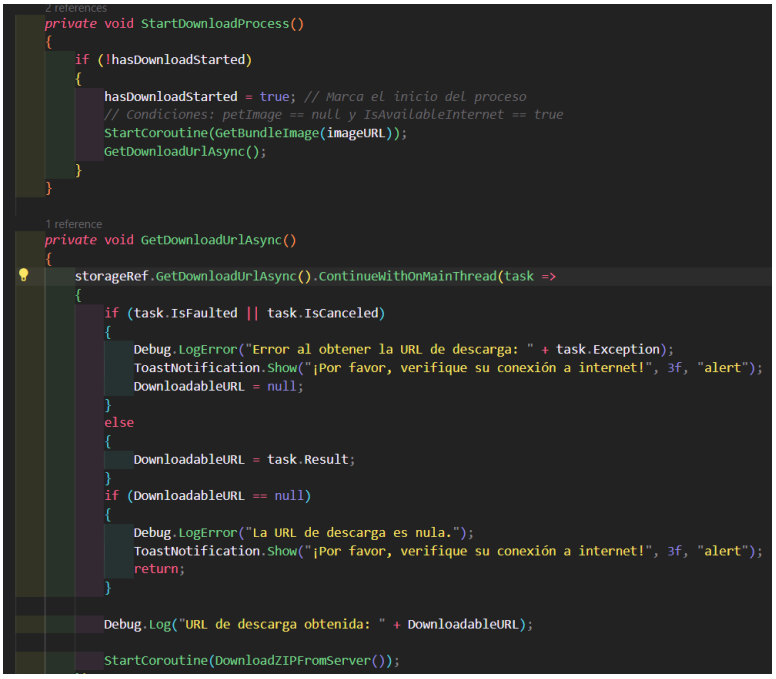
Figura 78. Código de inicializar el FirebaseStorage y el botón de carga

El archivo seleccionado es validado y convertido a Bytes para ser cargado como se muestra en la Figura 79.

```
2 references
private void UploadFileToFirebase(String fileName, byte[] data)
{
    ToastNotification.Show("¡Subiendo archivo...!", 3f, "info");

    // Activar la pantalla de carga al iniciar la subida
    var task = storageRef.Child("compressedFiles/" + fileName)
        .PutBytesAsync(data, null,
        new StorageProgress<UploadState>(state =>
        {
            // called periodically during the upload
            Debug.Log(String.Format("Progress: {0} of {1} bytes transferred.",
                state.BytesTransferred, state.TotalByteCount));
            // TextTMP
            float progress = (float)state.BytesTransferred / state.TotalByteCount;
            progressText.text = $"Cargando: {Mathf.RoundToInt(progress * 100)}%";
            // Actualizar el progreso en el Slider
            Debug.Log(String.Format("Progress: {0}%", progress * 100));
        }), CancellationToken.None, null);

    task.ContinueWith(resultTask =>
    {
        if (!resultTask.IsFaulted && !resultTask.IsCanceled)
        {
            Debug.Log("Upload finished.");
            fileUploadCanvas.SetActive(false);
        }
        else
        {
            Debug.Log(resultTask.Exception.ToString());
            fileUploadCanvas.SetActive(false);
        }
    });
```

Figura 79. Código de cargar el archivo del modelo 3D a Firebase Cloud Storage

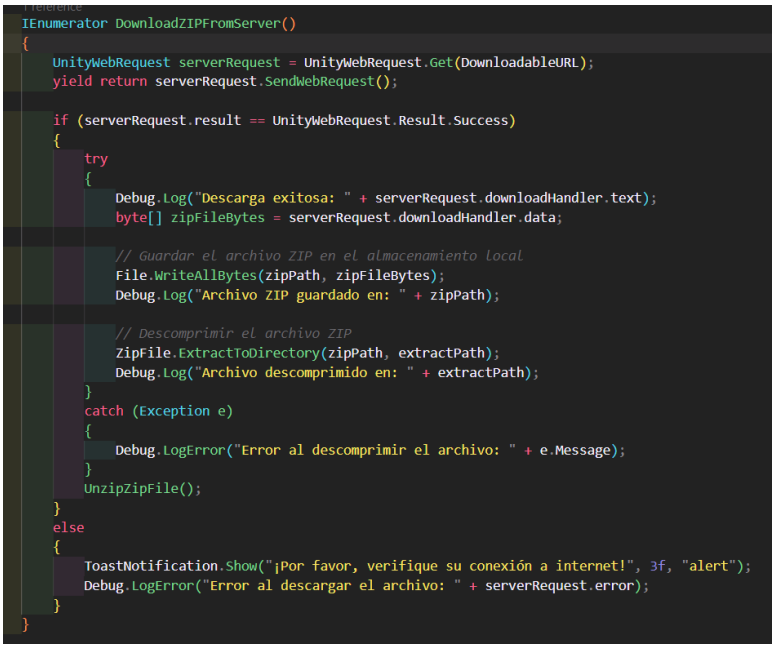
Configuración de la descarga del archivo del modelo 3D. Para descargar el archivo del modelo 3D se configuró la instancia de FirebaseStorage y se realizó la petición para obtener la URL de descarga como se muestra en la Figura 80.



```
2 references
private void StartDownloadProcess()
{
    if (!hasDownloadStarted)
    {
        hasDownloadStarted = true; // Marca el inicio del proceso
        // Condiciones: petImage == null y IsAvailableInternet == true
        StartCoroutine(GetBundleImage(imageURL));
        GetDownloadUrlAsync();
    }
}

1 reference
private void GetDownloadUrlAsync()
{
    storageRef.GetDownloadUrlAsync().ContinueWithOnMainThread(task =>
    {
        if (task.IsFaulted || task.IsCanceled)
        {
            Debug.LogError("Error al obtener la URL de descarga: " + task.Exception);
            ToastNotification.Show("¡Por favor, verifique su conexión a internet!", 3f, "alert");
            DownloadableURL = null;
        }
        else
        {
            DownloadableURL = task.Result;
        }
        if (DownloadableURL == null)
        {
            Debug.LogError("La URL de descarga es nula.");
            ToastNotification.Show("¡Por favor, verifique su conexión a internet!", 3f, "alert");
            return;
        }

        Debug.Log("URL de descarga obtenida: " + DownloadableURL);

        StartCoroutine(DownloadZIPFromServer());
```

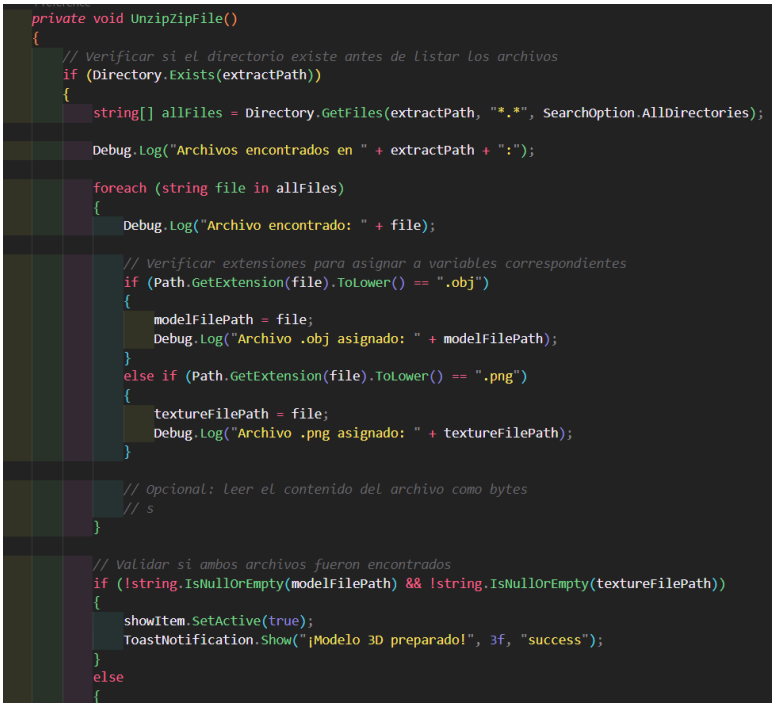
Figura 80. Código para obtener la URL descargable del archivo del modelo 3D

Después se definió el metodo para iniciar la descarga Figura 81. Una vez finalizada la descarga, se descomprimió el archivo para que sean procesados y asignados a un espacio de memoria dentro de la app, para que posteriormente sean proyectados en Realidad Aumentada, como se muestra en la Figura 82.



```
1 reference
IEnumerator DownloadZIPFromServer()
{
    UnityWebRequest serverRequest = UnityWebRequest.Get(DownloadableURL);
    yield return serverRequest.SendWebRequest();

    if (serverRequest.result == UnityWebRequest.Result.Success)
    {
        try
        {
            Debug.Log("Descarga exitosa: " + serverRequest.downloadHandler.text);
            byte[] zipFileBytes = serverRequest.downloadHandler.data;

            // Guardar el archivo ZIP en el almacenamiento local
            File.WriteAllBytes(zipPath, zipFileBytes);
            Debug.Log("Archivo ZIP guardado en: " + zipPath);

            // Descomprimir el archivo ZIP
            ZipFile.ExtractToDirectory(zipPath, extractPath);
            Debug.Log("Archivo descomprimido en: " + extractPath);
        }
        catch (Exception e)
        {
            Debug.LogError("Error al descomprimir el archivo: " + e.Message);
        }
        UnzipZipFile();
    }
    else
    {
        ToastNotification.Show("¡Por favor, verifique su conexión a internet!", 3f, "alert");
        Debug.LogError("Error al descargar el archivo: " + serverRequest.error);
    }
}
```

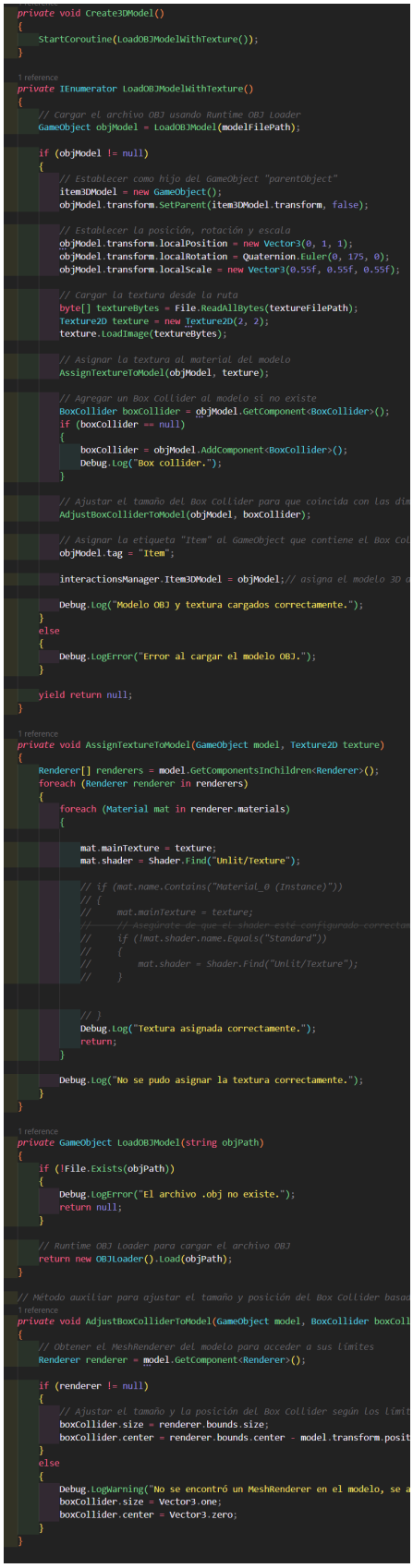
Figura 81. Código para descargar el modelo 3D



```
private void UnzipZipFile()
{
    // Verificar si el directorio existe antes de listar los archivos
    if (Directory.Exists(extractPath))
    {
        string[] allFiles = Directory.GetFiles(extractPath, "*.*", SearchOption.AllDirectories);

        Debug.Log("Archivos encontrados en " + extractPath + ":");

        foreach (string file in allFiles)
        {
            Debug.Log("Archivo encontrado: " + file);

            // Verificar extensiones para asignar a variables correspondientes
            if (Path.GetExtension(file).ToLower() == ".obj")
            {
                modelFilePath = file;
                Debug.Log("Archivo .obj asignado: " + modelFilePath);
            }
            else if (Path.GetExtension(file).ToLower() == ".png")
            {
                textureFilePath = file;
                Debug.Log("Archivo .png asignado: " + textureFilePath);
            }

            // Opcional: leer el contenido del archivo como bytes
            // s
        }

        // Validar si ambos archivos fueron encontrados
        if (!string.IsNullOrEmpty(modelFilePath) && !string.IsNullOrEmpty(textureFilePath))
        {
            showItem.SetActive(true);
            ToastNotification.Show("¡Modelo 3D preparado!", 3f, "success");
        }
        else
        {
```

Figura 82. Código para descomprimir el archivo zip

Configuración para posicionar el modelo 3D en el entorno real. Para cargar el modelo 3D en tiempo de ejecución se usó el paquete Runtime OBJ Loader. Una vez cargado, se asigna la textura y se guarda en un GameObject, quedando listo para ser proyectado en el entorno real, como se muestra en la Figura 83.



```
private void Create3DModel()
{
    StartCoroutine(LoadOBJModelWithTexture());
}

1 reference
private IEnumerator LoadOBJModelWithTexture()
{
    // Cargar el archivo OBJ usando Runtime OBJ Loader
    GameObject objModel = LoadOBJModel(modelFilePath);

    if (objModel != null)
    {
        // Establecer como hijo del GameObject "parentObject"
        item3DModel = new GameObject();
        objModel.transform.SetParent(item3DModel.transform, false);

        // Establecer la posición, rotación y escala
        objModel.transform.localPosition = new Vector3(0, 1, 1);
        objModel.transform.localRotation = Quaternion.Euler(0, 175, 0);
        objModel.transform.localScale = new Vector3(0.55f, 0.55f, 0.55f);

        // Cargar la textura desde la ruta
        byte[] textureBytes = File.ReadAllBytes(textureFilePath);
        Texture2D texture = new Texture2D(2, 2);
        texture.LoadImage(textureBytes);

        // Asignar la textura al material del modelo
        AssignTextureToModel(objModel, texture);

        // Agregar un Box Collider al modelo si no existe
        BoxCollider boxCollider = objModel.GetComponent<BoxCollider>();
        if (boxCollider == null)
        {
            boxCollider = objModel.AddComponent<BoxCollider>();
            Debug.Log("Box collider.");
        }

        // Ajustar el tamaño del Box Collider para que coincida con las dim
        AdjustBoxColliderToModel(objModel, boxCollider);

        // Asignar la etiqueta "Item" al GameObject que contiene el Box Col
        objModel.tag = "Item";

        interactionsManager.Item3DModel = objModel;// asigna el modelo 3D a

        Debug.Log("Modelo OBJ y textura cargados correctamente.");
    }
    else
    {
        Debug.LogError("Error al cargar el modelo OBJ.");
    }

    yield return null;
}

1 reference
private void AssignTextureToModel(GameObject model, Texture2D texture)
{
    Renderer[] renderers = model.GetComponentsInChildren<Renderer>();
    foreach (Renderer renderer in renderers)
    {
        foreach (Material mat in renderer.materials)
        {

            mat.mainTexture = texture;
            mat.shader = Shader.Find("Unlit/Texture");

            // if (mat.name.Contains("Material_0 (Instance)"))
            // {
            //     mat.mainTexture = texture;
            //     // Asegúrate de que el shader esté configurado correcta
            //     if (!mat.shader.name.Equals("Standard"))
            //     {
            //         mat.shader = Shader.Find("Unlit/Texture");
            //     }


            // }
            Debug.Log("Textura asignada correctamente.");
            return;
        }

        Debug.Log("No se pudo asignar la textura correctamente.");
    }
}

1 reference
private GameObject LoadOBJModel(string objPath)
{
    if (!File.Exists(objPath))
    {
        Debug.LogError("El archivo .obj no existe.");
        return null;
    }

    // Runtime OBJ Loader para cargar el archivo OBJ
    return new OBJLoader().Load(objPath);
}

// Método auxiliar para ajustar el tamaño y posición del Box Collider basad
1 reference
private void AdjustBoxColliderToModel(GameObject model, BoxCollider boxColl
{
    // Obtener el MeshRenderer del modelo para acceder a sus límites
    Renderer renderer = model.GetComponent<Renderer>();

    if (renderer != null)
    {
        // Ajustar el tamaño y la posición del Box Collider según los límit
        boxCollider.size = renderer.bounds.size;
        boxCollider.center = renderer.bounds.center - model.transform.posit
    }
    else
    {
        Debug.LogWarning("No se encontró un MeshRenderer en el modelo, se a
        boxCollider.size = Vector3.one;
        boxCollider.center = Vector3.zero;
    }
}
```

Figura 83. Código para cargar el modelo 3D en tiempo de ejecución

Con el modelo 3D cargado y listo para ser proyectado, solo toca presionar el botón para mostrar el modelo 3D en el entorno real como se muestra en la Figura 84. El modelo se mostrará como se presenta en la Figura 85, y el usuario podrá seleccionar el modelo y arrastrarlo a la posición que desee siempre y cuando sea una superficie horizontal. Además, podrá rotar el modelo 3D, seleccionando el modelo y desplazando con un segundo dedo sobre la pantalla.

Figura 84. Interfaz con botón para proyectar el modelo 3D

Figura 85. Interfaz para proyectar el modelo 3D

Configuración para tomar una captura del modelo 3D y compartirlo en redes sociales. Para poder compartir contenido que genera la app se usó el paquete NativeShare. Se configuró el método para deshabilitar por un momento el sistema de nube de puntos para poder tomar la captura y posteriormente se llama el método que captura la imagen, la procesa y le asigna al método de NativeShare para ser

compartido, como se muestra en la Figura 86.

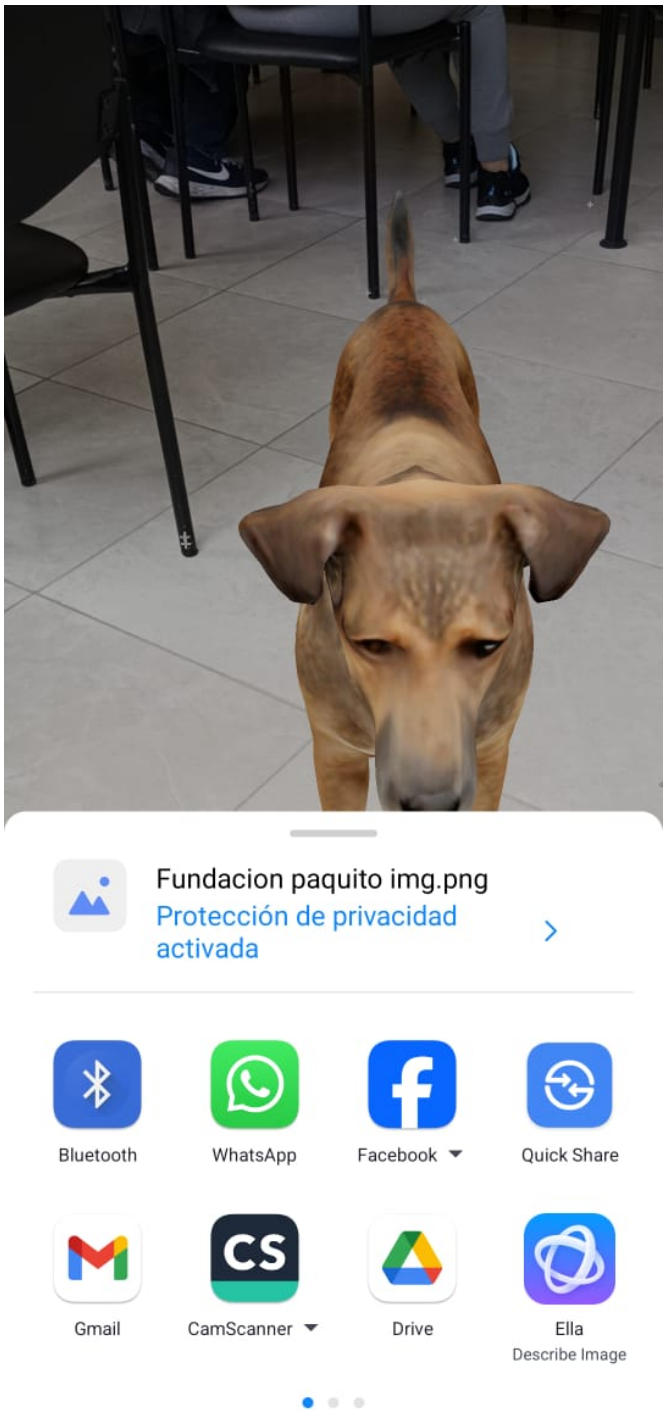


Figura 86. Código para compartir una captura del modelo 3D en redes sociales

- ***Configuración para integrar la app Unity en el entorno de Flutter.*** Para integrar la app Unity en Flutter se usaron dos paquetes. El paquete `fuw-2022.2.0.unitypackage` se instaló en el editor de Unity para poder exportar el proyecto a la plataforma de Android como se presenta en la Figura 87. Mientras

el paquete `flutter_unity_widget` con la versión 2022.2.1 se instaló en el archivo *pubspec.yaml* como se muestra en la Figura 88. Para completar la integración se configuraron varios archivos de la carpeta *android* ubicado en la raíz del proyecto de Flutter, tal como indica la documentación del paquete `flutter_unity_widget`.

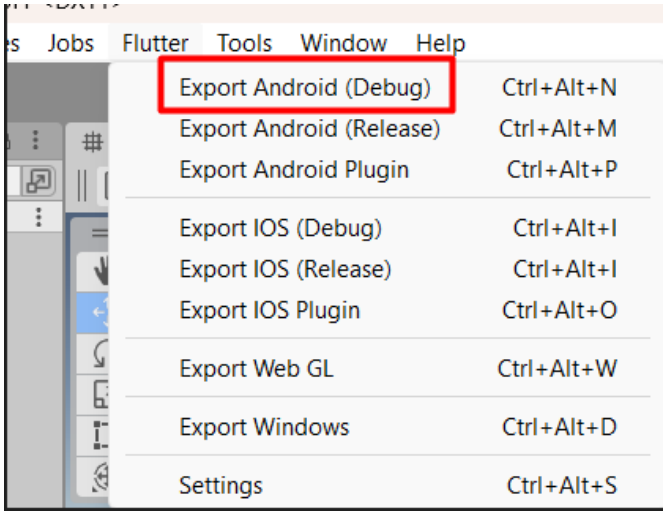


Figura 87. Configuración del paquete `fuw-2022.2.0.unitypackage` en el editor de Unity

```
dependencies:
  animate_do: ^3.3.3
  badges: ^3.1.2
  cached_network_image: ^3.3.1
  camera_camera: ^3.0.0
  connectivity_plus: ^6.1.1
  cupertino_icons: ^1.0.6
  dio: ^5.4.3+1
  elegant_notification: ^2.0.0
  file_picker: 8.1.4
  firebase_core: ^3.8.1
  firebase_messaging: ^15.1.6
  flutter:
    sdk: flutter
  flutter_dotenv: ^5.1.0
  flutter_local_notifications: ^18.0.1
  flutter_secure_storage: ^9.0.0
  flutter_staggered_grid_view: ^0.7.0
  flutter_sticky_header: ^0.6.5
  flutter_unity_widget: ^2022.2.1
```

Figura 88. Configuración del paquete `flutter_unity_widget` en Flutter

Una vez completada la configuración de los archivos en la carpeta *android*, se creó una interfaz para usar el `UnityWidget`. `UnityWidget` es el widget encargado de instanciar y mostrar la app Unity dentro del entorno de Flutter, tal como se muestra en la Figura 89. Además, se definieron métodos para enviar datos desde Flutter a Unity y viceversa.

```dart
class Unity3DModelScreen extends StatefulWidget {
  static const String id = "unity_3dmodel_screen";
  final ThreeDObject threeDObject;
  final VoidCallback? onUpdateCallback;

  const Unity3DModelScreen({
    super.key,
    required this.threeDObject,
    this.onUpdateCallback,
  });

  @override
  State<Unity3DModelScreen> createState() => _Unity3DModelScreenState();
}

class _Unity3DModelScreenState extends State<Unity3DModelScreen> {
  late UnityWidgetController controller;
  bool isUnityVisible = true;
  bool hasAttemptedReload = false;
  bool isNew3DObject = false;

  @override
  void initState() {
    super.initState();
    isNew3DObject = false;
    _attemptReloadUnity();
  }

  void onUnityCreated(UnityWidgetController controller) {
    this.controller = controller;
    send3DObjectToUnity();
  }

  void send3DObjectToUnity({String? objectName}) {
    if (objectName != null) {
      widget.threeDObject.setObjectName(objectName);
    }
    String jsonObject = jsonEncode(widget.threeDObject.toJsonForUnity());
    controller.postMessage(
        "ModelLoader", "ReceiveMessageFromFlutter", jsonObject);
  }

  void delete3DObject() {
    controller.postMessage("ARInteractionsManager",
        "ReceiveMessageToDeleteItem", "Eliminar Modelo 3D");
  }

  // Método para alternar la visibilidad de UnityWidget y forzar su recarga
  void _attemptReloadUnity() {
    Future.delayed(const Duration(seconds: 1), () {
      if (!hasAttemptedReload) {
        setState(() {
          isUnityVisible = false; // Ocultar UnityWidget temporalmente
        });
        Future.delayed(const Duration(milliseconds: 500), () {
          setState(() {
            isUnityVisible = true; // Volver a mostrar UnityWidget
            hasAttemptedReload =
                true; // Marcar como recargado para evitar bucles
          });
        }); // Future.delayed
      }
    }); // Future.delayed
  }

  @override
  void dispose() {
    // Notificar que se registró un modelo 3D antes de cerrar la pantalla
    if (isNew3DObject) {
      widget.onUpdateCallback?.call();
    }
    // Aquí destruyes el controlador Unity y cualquier recurso.
    delete3DObject();
    controller.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: isUnityVisible
          ? UnityWidget(
              onUnityCreated: onUnityCreated,
              onUnityMessage: (objectName) {
                if (widget.threeDObject.id != null) {
                  print("Excelente");    Don't invoke 'print' in production code
                  context.read<AnimalViewModel>().save3dObjectName(
                      widget.threeDObject.id.toString(),
                      ThreeDObject(name: objectName));
                  send3DObjectToUnity(objectName: objectName);
                  isNew3DObject = true;
                }
              },
            ) // UnityWidget
          : const Center(
              child: CircularProgressIndicator(),
            ), // Center
    ); // Scaffold
  }
}
```

Figura 89. Configuración para ejecutar la app Unity en Flutter

- ***Generación de solicitud de adopción.*** El proceso para generar una solicitud de

adopción es sencillo. El usuario adoptante, después de haber seleccionado en la interfaz principal un animal de compañía, podrá ver el perfil del animal como se presenta en la Figura 90. El perfil tiene toda la información que la Fundación tiene acerca de dicho animal. En este perfil, debe presionar el botón “Adoptame” para llenar el formulario de solicitud de adopción. Una vez completado el formulario, podrá enviar la solicitud como se muestra en las Figuras 91, 92 y 93.

Figura 90. Interfaz de perfil del animal de compañía

Figura 91. Interfaz de formulario de solicitud de adopción - cabecera

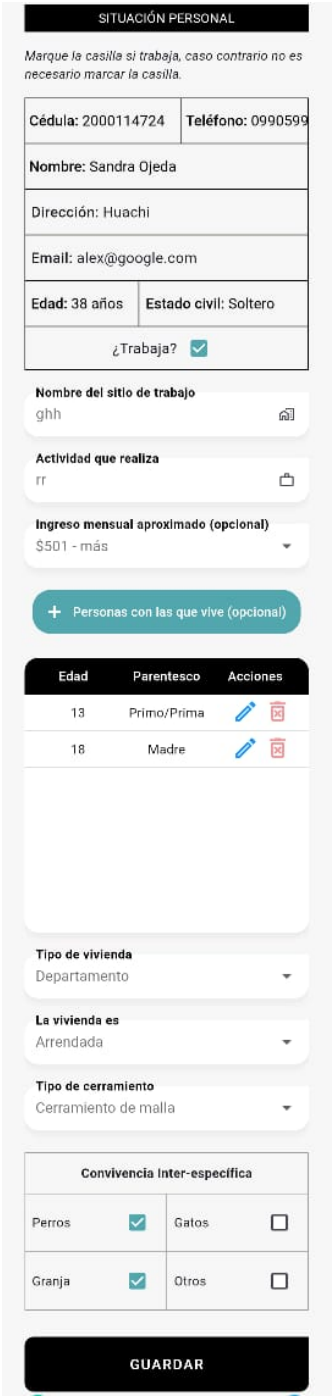


Figura 92. Interfaz de formulario de solicitud de adopción - situación personal

Figura 93. Interfaz de formulario de solicitud de adopción - relación con los animales

- ***Gestión de solicitudes de adopción.*** El usuario administrador puede ver una lista de las solicitudes de adopción como se muestra en la Figura 94. Además, podrá revisar la información enviada por el usuario adoptante para aprobar o denegar la solicitud de adopción como se muestra en la Figura 95.

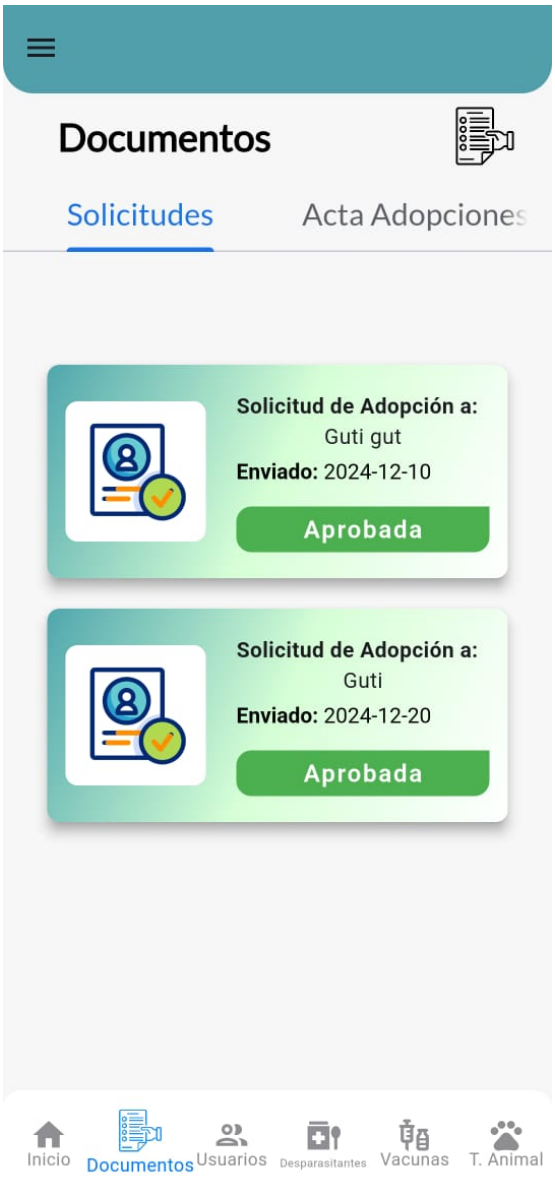


Figura 94. Interfaz lista de solicitudes de adopción

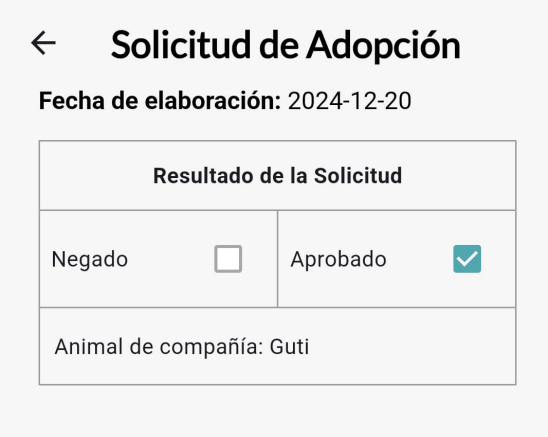


Figura 95. Interfaz para aprobar o denegar la solicitud de adopción

- ***Gestión de actas de adopciones.*** El usuario adoptante puede ver una lista de actas de adopción como se muestra en la Figura 96. Además, podrá revisar la información del apta para aceptar o denegar el acta de adopción como se muestra en la Figura 97.

Figura 96. Interfaz lista de actas de adopción

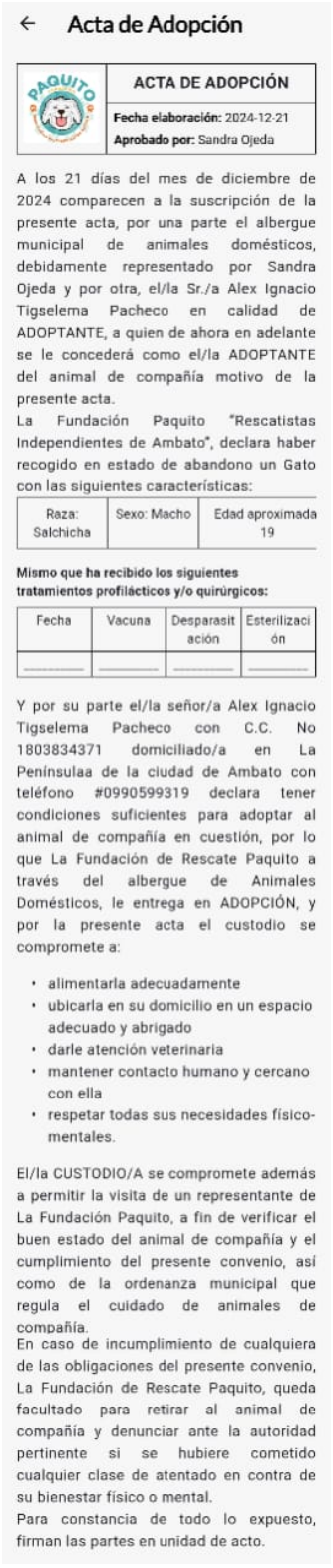


Figura 97. Interfaz para aceptar o denegar el acta de adopción

3.3.4 Transición

a. Pruebas de aceptación

Las pruebas de aceptación permitieron que el usuario interactuara con la app, con el propósito de validar las características funcionales, y así, asegurar el cumplimiento de los requisitos planteados en la primera fase. Las pruebas se organizaron de manera iterativa, por tanto, se definieron criterios específicos a evaluar en cada iteración de desarrollo.

Tabla 21. Pruebas de aceptación - primera iteración

N°	Criterio	Estado	Usuario
1	Al iniciar sesión se redirecciona a la pantalla principal	Aceptado	Administrador
2	Al iniciar sesión se redirecciona a la pantalla principal	Aceptado	Adoptante
3	Al cerrar sesión se redirecciona a la pantalla de iniciar sesión	Aceptado	Administrador
4	Al cerrar sesión se redirecciona a la pantalla de iniciar sesión	Aceptado	Adoptante
5	Después de registrarse en la app se redirecciona a la pantalla principal	Aceptado	Adoptante
6	Para recuperar la contraseña, el sistema envía una nueva contraseña al correo registrado	Aceptado	Administrador
7	Para recuperar la contraseña, el sistema envía una nueva contraseña al correo registrado	Aceptado	Adoptante
8	Probar las acciones de crear, habilitar, inhabilitar y editar rol de usuarios	Aceptado	Administrador
9	Visualizar los usuarios registrados en la app	Aceptado	Administrador
10	Actualizar información personal y cambiar contraseña	Aceptado	Administrador
11	Actualizar información personal y cambiar contraseña	Aceptado	Adoptante

Tabla 22. Pruebas de aceptación - segunda iteración

N°	Criterio	Estado	Usuario
1	Probar acciones de crear, editar, habilitar e inhabilitar los desparasitantes	Aceptado	Administrador
2	Visualizar los registros de desparasitantes	Aceptado	Administrador
3	Probar acciones de crear, editar, habilitar e inhabilitar las vacunas	Aceptado	Administrador
4	Visualizar los registros de vacunas	Aceptado	Administrador

5	Probar acciones de crear, editar, habilitar e inhabilitar los tipos de animales	Aceptado	Administrador
6	Visualizar los registros de tipos de animales	Aceptado	Administrador
7	Probar acciones de crear, editar, habilitar e inhabilitar los animales de compañía	Aceptado	Administrador
8	Visualizar los animales de compañía registrados	Aceptado	Administrador
9	Visualizar los animales de compañía registrados	Aceptado	Adoptante
10	Probar la funcionalidad de búsqueda de animal de compañía por nombre, raza, género, color y tamaño	Aceptado	Administrador
11	Probar la funcionalidad de búsqueda de animal de compañía por nombre, raza, género, color y tamaño	Aceptado	Adoptante
12	Visualizar el perfil del animal de compañía	Aceptado	Adoptante

Tabla 23. Pruebas de aceptación - tercera iteración

N°	**Criterio**	**Estado**	**Usuario**
1	Probar la funcionalidad del rastreo de superficies y detección de planos mediante la nube de puntos	Aceptado	Adoptante
2	Probar la funcionalidad del rastreo de superficies y detección de planos mediante la nube de puntos	Aceptado	Administrador
3	Probar la funcionalidad de cargar el archivo del modelo 3D del animal de compañía	Aceptado	Administrador
4	Probar la funcionalidad de proyectar el modelo 3D del animal de compañía en el entorno real	Aceptado	Administrador
5	Probar la funcionalidad de proyectar el modelo 3D del animal de compañía en el entorno real	Aceptado	Adoptante
6	Probar la interacción con el modelo 3D como seleccionar, rotar y arrastrar el modelo	Aceptado	Administrador
7	Probar la interacción con el modelo 3D como seleccionar, rotar y arrastrar el modelo	Aceptado	Adoptante
8	Probar la funcionalidad de tomar una captura al animal de compañía en el entorno real y compartirlo en redes sociales	Aceptado	Administrador
9	Probar la funcionalidad de tomar una captura al animal de compañía en el entorno real y compartirlo en redes sociales	Aceptado	Adoptante
10	Probar la funcionalidad de generar la solicitud de adopción	Aceptado	Adoptante
11	Visualizar las solicitudes de adopción generadas por el usuario adoptante	Aceptado	Administrador

12	Visualizar las solicitudes de adopción generadas por el usuario adoptante	Aceptado	Adoptante
13	Probar la funcionalidad de aprobar o denegar la solicitud de adopción	Aceptado	Administrador
14	Visualizar la respuesta de aprobado o denegado de la solicitud de adopción enviada	Aceptado	Adoptante
15	Visualizar las actas de adopción generadas por el sistema	Aceptado	Administrador
16	Visualizar las actas de adopción generadas por el sistema	Aceptado	Adoptante
17	Probar la funcionalidad de aceptar o cancelar el acta de adopción	Aceptado	Adoptante

a. Pruebas de adopción

Para evaluar el impacto que tiene la app en el tiempo de proceso de adopción, se comparó el tiempo promedio que se necesita para concretar una adopción antes y después de implementar la app. Antes del uso de la app y conforme a la información que manifestó el coordinador de la Fundación durante las pruebas, el tiempo promedio que se demora el proceso de adopción es de tres días, como se muestra en la Tabla 24. Sin embargo, el coordinador agregó que han habido casos en donde el proceso de adopción se extiende hasta más de 5 días, debido a que se demoran en envíar el formulario de solicitud de adopción o en responder al acta de adopción. Mientras que con el uso de la app y la participación de cinco personas para simular el proceso de adopción, el tiempo promedio fue de dos días, como se muestra en la Tabla 25. Por tanto, se puede afirmar que el tiempo de proceso de adopción con el uso de la app se redujo en un día. Sin embargo, al ser una prueba de simulación no se puede considerar que siempre durará dos días, debido a que depende de la disponibilidad tanto de la persona adoptante como del personal de la Fundación para realizar la inspección en el domicilio. En cuanto a la experiencia de los participantes, manifestaron que les fue de mucha ayuda la visualización en 3D del animal y el perfil al momento de decidir cual adoptar. Por tal razón, el tiempo que la persona necesita para decidir cual animal desea adoptar es menor. Además, basado en la comparativa de las tablas se puede ver una diferencia de 8 minutos. Es necesario aclarar, que no se consideró el tiempo que le toma a una persona adoptante movilizarse hasta la Fundación para buscar un animal de su interés. Para la parte del formulario, también se redujo tiempo, debido a que hay campos que solo requieren ser llenados una sola vez, y en caso de que la persona desee adoptar otro animal de compañía, no será necesario volver a llenar esos

campos, a no ser que desee actualizarlos.Para la inspección del domicilio por parte de la Fundación Paquito, no se redujo tiempo. Para la respuesta de la Fundación a la solicitud de adopción, ellos se reservan el tiempo de responder en un lapso de veinticuatro horas (puede variar hasta 48 horas), por lo que no se reduce tiempo en este paso. Para la confirmación del adoptante a la Fundación, si hay una reducción de tiempo ya que el tiempo promedio sin la app es de un día mientras que con la app se redujo a solo minutos. Para finalizar con la entrega del animal por parte de la Fundación al adoptante, no hubo cambios. Sin embargo, se puede considerar que el tiempo de entrega puede variar dependiendo de la disponibilidad de la persona adoptante y de la Fundación. Por tanto, se puede concluir que la app ayuda a reducir el tiempo de proceso de adopción, pero no garantiza que siempre será de dos días, ya que depende de la disponibilidad de las personas involucradas en el proceso de adopción.

Tabla 24. Tiempo promedio del proceso de adopción sin la app

Participante	**Selección del animal (min)**	**Formulario (min)**	**Inspección domicilio (min)**	**Respuesta Fundación (días)**	**Confirmación adoptante (min)**	**Entrega del animal (días)**	**Tiempo total (días)**
Persona 1	15	10	26	1	1	1	3.04
Persona 2	17	14	27	1	1	1	3.04
Persona 3	16	13	27	1	1	1	3.04
Persona 4	19	15	26	1	1	1	3.04
Persona 5	30	14	25	1	1	1	3.05
Promedio	19.4	13.2	26.2	1	1	1	3.04

Tabla 25. Tiempo promedio del proceso de adopción con la app

Participante	Selección del animal (min)	Formulario (min)	Inspección domicilio (min)	Respuesta Fundación (días)	Confirmación adoptante (min)	Entrega del animal (días)	Tiempo total (días)
Persona 1	8	5	26	1	12	1	2.04
Persona 2	14	5	27	1	19	1	2.05
Persona 3	13	6	27	1	31	1	2.05
Persona 4	10	6	26	1	14	1	2.04
Persona 5	11	6	25	1	8	1	2.03
Promedio	11.2	5.6	26.2	1	16.8	1	2.04

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

El presente proyecto se desarrolló con la finalidad de implementar una aplicación móvil enfocada en agilizar el proceso de adopción de animales de compañía que maneja la Fundación Paquito.

- El análisis sobre las herramientas tecnológicas que se utilizaron para el desarrollo de la app, permitió seleccionar las más adecuadas para cumplir con los objetivos planteados. La app se desarrolló en Flutter, un framework de código abierto que permite crear aplicaciones nativas para dispositivos móviles. Además, se utilizó el editor de Unity junto con ARCore para la proyección de modelos 3D en Realidad Aumentada. La integración de Unity en Flutter se logró gracias al paquete `flutter_unity_widget`. También, fue de gran apoyo la documentación que tienen los paquetes de Flutter y Unity, ya que permitieron comprender de manera detallada el uso de las herramientas.

- Con el apoyo en la información adquirida por las entrevistas aplicadas al coordinador y el personal de la Fundación Paquito, fue posible identificar los aspectos que requerían una mayor atención debido a la delicadesa del asunto de adopción de animales de compañía. La entrevista al coordiandor de la Fundación fue sustancial para determinar el proceso de adopción de los animales de compañía, así mismo, el proceso que cumplen al ingresar un nuevo animal de compañía a la Fundación. Además, que manifestó la falta de seriedad y compromiso por parte de los adoptantes, siendo este tema, un aspecto complicado de gestionar. Por otro lado, las encuestas dirigidas a potenciales adoptantes manifestaron lo valioso que es conocer a detalle sobre el animal de compañía de su interés, como por ejemplo, el historial de vacunas, el historial de desparasitantes, el color, el género, etc.

- Un aspecto relevante de la app es que permite gestionar de manera sencilla el ingreso de un animal de compañía a la Fundación. El registro del animal de compañía mediante la app agiliza el proceso de guardar todos los datos relevantes. Estos mismos datos son los que se muestran para los posibles

adoptantes. Además, la app permite que la Fundación pueda gestionar aspectos importantes como las vacunas, los desparasitantes y los tipos de animales de compañía. Esto es vital para la Fundación debido a la calidad de servicio que ofrecen no solo a los animales de compañía, sino también a los posibles adoptantes. En cuanto a los tipos de animales, se debe a que la Fundación no solo se enfoca en el cuidado de perros y gatos, sino en otras especies consideradas como animales de la Fauna urbana.

- La implementación de Realidad Aumentada sin marcadores para proyectar modelos 3D en el entorno real, ofrece flexibilidad para el usuario adoptante, al no depender de imágenes como marcadores o estar en una ubicación específica para ver al animal de compañía. Además, ofrece una interacción con el modelo 3D, al poder seleccionarlo para arrastrar y rotar el modelo. Por otro lado, la proyección de modelos 3D demostró ser de gran ayuda para decidir que animal de compañía se desea adoptar, esto debido al tiempo que se ahorra en movilizarse hasta la Fundación. De igual forma, la presentación de un perfil detallado del animal de compañía, permite tener más seguridad al decidir adoptar.

4.2 Recomendaciones

- Se sugiere al coordinador de la Fundación Paquito que reciba una capacitación sobre la utilidad de la app para esclarecer toda duda que tenga, para así, pueda usar la app de una forma adecuada.

- Se recomienda que la Fundación Paquito realice una capacitación sobre el uso de la app para que la población esté actualizada sobre los nuevos servicios que ponen a disposición del público. Además, para concientizar sobre el uso responsable que deben hacer con la app.

- Debido a que la app es compatible únicamente con el sistema operativo Android, se recomienda expandir su compatiblidad a dispositivos con sistema operativo iOS. Esto es muy importante para la Fundación ya que al ampliar el alcance de usabilidad, se incrementan las posibilidades de que un animal de compañía encuentre un hogar.

REFERENCIAS BIBLIOGRÁFICAS

[1] N. A. Jaramillo Aguiar y R. E. Macas Narváez, “Desarrollo de una aplicación móvil con realidad aumentada que apoye el proceso de enseñanza-aprendizaje del uso de los equipos del laboratorio de máquinas cnc (control numérico computarizado) de la carrera de mecánica de la universidad politécnica salesiana,” B.S. thesis, 2020.

[2] N. D. Riano Nossa, “Estudio comparativo de metodologías tradicionales y ágiles aplicadas en la gestión de proyectos,” 2021.

[3] D. You y M. Hu, “A comparative study of cross-platform mobile application development,” 07 2021.

[4] A. E. Fentaw, “Cross platform mobile application development: a comparison study of react native vs flutter,” Tesis de Maestria, 2020.

[5] InApp, “React native vs. flutter vs. ionic vs. xamarin vs. nativescript (a detailed comparison),” 2022. [En línea]. Disponible en: https://inapp.com/blog/react-native-vs-flutter-vs-ionic-vs-xamarin-vs-nativescript-a-detailed-comparison/

[6] D. Choma, K. Chwaleba, y M. Dzieńkowski, “The efficiency and reliability of backend technologies: Express, django, and spring boot,” *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska*, vol. 13, no. 4, p. 73–78, Dec. 2023. [En línea]. Disponible en: https://ph.pollub.pl/index.php/iapgos/article/view/4279

[7] P. López Herrera, “Comparación del desempeño de los sistemas gestores de bases de datos mysql y postgresql,” 2016.

[8] R. Kolandaisamy, K. Subaramaniam, I. Kolandaisamy, y L. S. Li, “Stray animal mobile app,” *2016*, 2016.

[9] T. Muñoz Testón, “Plataforma web para la adopción y gestión de animales procedentes de protectoras,” Ph.D. dissertation, ETSI_Sistemas_Infor, 2019.

[10] A. C. MENDEZ MELO, J. MARTINEZ ESPAÑA, A. VILLAFAÑE CASTAÑO, J. A. CRIOLLO *et al.*, “Aplicación móvil para adopción de mascotas abandonadas “peluditos. com”.”

[11] B. S. Campanilla, J. O. Etcuban, A. P. Maghanoy, P. A. P. Nacua, y N. S. Galamiton, “Pet adoption app to free animal shelters,” *Journal of Positive School Psychology*, vol. 6, no. 8, pp. 5993–6006, 2022.

[12] D. F. Flores Campos y C. E. Zavaleta León, ““adoptape” sistema web y móvil para facilitar la adopción de mascotas en el albergue asociación de voluntarios animalistas en trujillo,” 2020.

[13] C. R. Del Castillo Garzón, “Animales de la calle en quito: propuestas y soluciones,” B.S. thesis, Quito: Universidad de las Américas, 2020, 2020.

[14] D. A. Paz Valladares, “Caracterización de riesgos para la salud humana por la presencia de perros y gatos en la ciudad de quito, 2020-2021,” Tesis de Maestria, PUCE-Quito, 2023.

[15] X. Sotomayor, “Aplicación móvil para la gestión de una asociación de búsqueda y adopción de mascotas,” Ph.D. dissertation, ETSI_Sistemas_Infor, 2019.

[16] I. K. Cabrera Pérez y M. A. Gonzales Espinoza, “Influencia del uso de una aplicación móvil con estándares de usabilidad en la tenencia responsable de mascotas domésticas en una asociación de protección animal en cajamarca,” 2019.

[17] C. J. Barcenes Llanos y M. F. Cedeño Avelino, “Desarrollo de un servicio web y aplicación móvil en android que permita ubicar y rescatar animales domésticos extraviados o abandonados para la fundación rescate animal ecuador.” B.S. thesis, Universidad de Guayaquil Facultad de Ciencias Matemáticas y Físicas Carrera …, 2019.

[18] K. K. Into Arroyo, “Fiabilidad de la aplicación móvil smileypet para la tenencia responsable de las mascotas en los socios de la asociación mi pequeño hermano, cajamarca 2019,” 2021.

[19] M. I. TOAPANTA GUALPA, “Desarrollo de un sistema de gestión sectorizado de adopción de mascotas para la fundación camino a casa,” B.S. thesis, Quito, 2019.

[20] V. M. d. Pozo Gómez *et al.*, “Youcan: aplicación web para fomentar la adopción de mascotas,” 2019.

[21] C. C. Cruz Bustos *et al.*, “Benepet, aplicación móvil multiplataforma para la prevención del abandono de mascotas en la ciudad de bogotá dc,” 2021.

[22] C. Y. Ortiz Diaz y B. G. Diaz Moreno, “Desarrollo de una aplicación web basado en un enfoque de procesos de adopción, donación, esterilización para mejorar la calidad de vida de los animales domésticos en la zona alto magdalena,” Ph.D. dissertation, Corporación Universitaria Minuto de Dios, 2021.

[23] E. A. B. Ulloa, M. A. C. Molina, L. M. H. Guamán, y F. A. M. Chiluiza, “Prototipo móvil para la geolocalización de mascotas callejeras,” *Revista Odigos*, vol. 1, no. 3, pp. 77–96, 2020.

[24] E. D. Hernández Almagro y M. F. Tibanta Legña, “Desarrollo de un sistema web y aplicación móvil para la gestión de fundaciones de animales de compañía en la ciudad de quito,” B.S. thesis, Quito, 2020., 2020.

[25] C. M. B. DANIELA, “Aplicación web y móvil para el seguimiento de adopción y búsqueda de mascotas en la fundación refugio pana,” Ph.D. dissertation, UNIVERSIDAD AGRARIA DEL ECUADOR, 2022.

[26] J. P. Monar Anrango, “Aplicación móvil de búsqueda de mascotas perdidas y abandonadas para la fundación protección animal ecuador-pae,” B.S. thesis, 2021.

[27] M. F. Silvente *et al.*, “Plataforma de adopción de animales y gestión de organizaciones proteccionistas,” B.S. thesis, 2022.

[28] M. G. Morales Mateo, “Desarrollo de una aplicación web, como aporte a la gestión y seguimiento del proceso de adopción de mascotas en la fundación huellas felices.” Ph.D. dissertation, Universidad de Guayaquil. Facultad de Ingeniería Industrial. Carrera de …, 2021.

[29] B. Machín Morera, “Desarrollo de una aplicación web para adoptar perros y gatos,” 2022.

[30] M. d. Rincón García, “Desarrollo de sistema cliente-servidor para la adopción de mascotas,” 2022.

[31] M. J. Mariño Toaza, “Implementación de una plataforma de realidad aumentada basada en google arcore enlazada a moodle como apoyo a la docencia en la

unidad educativa mons. edmundo carmody de la ciudad de guayaquil en el cantón de durán 2021.” B.S. thesis, Instituto Superior Universitario Bolivariano de Tecnología., 2024.

[32] M. L. F. BRAVO y L. L. L. NARVAEZ, “Estrategia didáctica con el uso de unity y vuforia para la construccion del aprendizaje de las ciencias naturales en estudiantes del grado septimo.”

[33] I. M. Céspedes, Y. S. Pellicer, y P. G. Ávila, “Docente del departamento de informática. universidad de granma (cuba). e-mail: jaleyva@ udg. co. cu orcid: https://orcid. org/0000-0001-8420-7102.”

[34] E. Arias-Méndez, E. González-Damazio, y D. Porras-Alfaro, “Amón_ra: procesamiento de marcadores de realidad aumentada para el reconocimiento de grandes objetos en el mundo real,” *Investiga. TEC*, no. 36, pp. 6–10, 2019.

[35] Y. Morillo, “Realidad aumentada | qué es, ventajas, tipos y aplicaciones,” 2020. [En línea]. Disponible en: https://futuroelectrico.com/realidad-aumentada/

[36] C. A. Rey Bermúdez y M. S. Quijano Esparza, “Implementación de un sistema de realidad aumentada (ar) con unity que permita obtener información sobre artefactos históricos del museo guane de la unab, utilizando reconocimiento de imágenes,” 2020.

[37] Aimojo, “Tripo ai,” 2024. [En línea]. Disponible en: https://aimojo.io/es/tools/tripo-ai/

[38] M. H. Center, “What is meshy?” 2024. [En línea]. Disponible en: https://help.meshy.ai/en/articles/9991736-what-is-meshy

[39] creati, “Csm ai: Generar mundos 3d a partir de fotos, textos y bocetos,” 2024. [En línea]. Disponible en: https://creati.ai/es/ai-tools/csm/

[40] S. X. Zhuño Quizhpe, “Desarrollo de un software educativo aplicando la metodología kanban, para el aprendizaje de vocabulario básico de inglés,” B.S. thesis, Riobamba, Universidad Nacional de Chimborazo, 2023.

[41] F. Sabaté y M. del Milagro, “” desarrollo de un sistema para gestión de becas, aplicando metodología rad,” 2023.

ANEXOS

Anexo A. Guía de entrevista aplicada al gerente de la Fundación Paquito

Tabla A1. Entrevista dirigida al gerente de la Fundación

N.	Pregunta	Respuesta
1.	¿Cuáles son las funciones del personal de la Fundación?	
2.	¿Qué tipo de animales de compañía atiende la Fundación? (perros, gatos, etc.)	
3.	¿Cuál es el tamaño de la población de animales de compañía que atiende la Fundación?	
4.	¿Cuál es el proceso actual para ingresar o registrar un animal de compañía a la Fundación?	
5.	¿Cuál es el proceso actual para adoptar un animal de compañía de la Fundación?	
6.	¿Qué información acerca de los animales de compañía le gustaría compartir con el personal de la Fundación y los adoptantes a través de la aplicación móvil? (por ejemplo: Fotos, videos, información sobre la mascota como raza, edad, sexo, condición de salud, vacunas, personalidad de la mascota, etc.)	
7.	¿De qué manera validan y verifican la información proporcionada por la persona adoptante?	
8.	¿Qué persona es la responsable en aprobar la solicitud de adopción y en base a qué parámetros califican si una persona es apta para adoptar? (detalle los parámetros)	
9.	¿Qué problemas o inconvenientes han tenido con el actual proceso de adopción de animales de compañía?	
10.	¿Cuántas adopciones gestionan aproximadamente al mes?	

11.	¿Conoce o tiene nociones acerca de lo que es la realidad aumentada?	

Elaborado por: el investigador

Anexo B. Guía de entrevista aplicada al personal de la Fundación Paquito

Tabla B1. Entrevista dirigida al personal de la Fundación

N.	Pregunta	Respuesta
1.	¿Cuál es el rol que ejerce para la Fundación Paquito?	▪ a) Voluntario ▪ b) Empleado a tiempo completo ▪ c) Empleado a tiempo parcial ▪ d) Otro (especificar)
2.	¿Cuáles son los requisitos de información o documentos necesarios que se debe presentar por parte de la persona adoptante para iniciar el proceso de adopción?	
3.	¿Cuáles han sido las razones por las que un adoptante a devuelto al animal de compañía a la Fundación?	
4.	¿Qué problemas o inconvenientes han tenido con el proceso de adopción actual?	
5.	¿Qué acciones sugiere usted que se puedan tomar para ayudar a mitigar los problemas o inconvenientes?	
6.	¿De que forma se almacenan los datos de las personas adoptantes? (por ejemplo: guardan en excel, pdf, etc)	
7.	¿Cuáles son los datos que considera esencial para incluir en el perfil de un animal de compañía? (por ejemplo: Fotos, videos, información sobre la mascota como raza, edad, sexo, condición de salud, vacunas, personalidad de la mascota, etc.)	
8.	¿Conoce o tiene nociones acerca de lo que es la realidad aumentada?	

Elaborado por: el investigador

Anexo C. Guía de encuesta dirigida a potenciales adoptantes

Tabla C1. Encuesta dirigida a potenciales adoptantes

N.	Pregunta	Respuesta
1.	¿Qué información acerca de un animal de compañía considera usted que es importante conocer antes de adoptarla?	▪ a) Fotos e información (raza, edad, sexo, etc.) ▪ b) Antecedentes (historial médico y vacunas) ▪ c) Personalidad ▪ d) Videos ▪ e) Todas las anteriores
2.	¿Considera usted un inconveniente la movilización hacia la Fundación Paquito para iniciar el proceso de adopción?	▪ a) Sí ▪ b) No
3.	¿Le gustaría realizar una búsqueda de animales de compañía en una app mediante el color, raza, edad, sexo?	▪ a) Totalmente de acuerdo ▪ b) De acuerdo ▪ c) Ni de acuerdo ni en desacuerdo ▪ d) En desacuerdo ▪ e) Totalmente en desacuerdo
4.	¿Le gustaría recibir notificaciones sobre nuevos animales de compañía disponibles para adopción mediante la app?	▪ a) Totalmente de acuerdo ▪ b) De acuerdo ▪ c) Ni de acuerdo ni en desacuerdo ▪ d) En desacuerdo ▪ e) Totalmente en desacuerdo
5.	¿Conoce o tiene nociones acerca de lo que es la realidad aumentada?	▪ a) Sí ▪ b) No

6.	¿Le gustaría visualizar en la app al animal de compañía mediante realidad aumentada?	▪ a) Totalmente de acuerdo ▪ b) De acuerdo ▪ c) Ni de acuerdo ni en desacuerdo ▪ d) En desacuerdo ▪ e) Totalmente en desacuerdo

Elaborado por: el investigador

Anexo D. Documentos que intervienen para el proceso de adopción del animal de compañía en la Fundación Paquito

PAQUITO	SOLICITUD DE ADOPCIÓN	**Fecha elaboración:**
		Ultima aprobación:
		Revisión:
	Elaborado por: / **Revisado por:**	**Aprobado por:**

Fecha:	**Mascota escogida:**	**Código/chip:**	**Negado**
		Características:	**Aprobado**

Nuestro principal compromiso es asegurar que la mascota a ser adoptada llegue a un hogar que presente las mejores condiciones en cuanto a su bienestar.
Esta solicitud será aprobada o rechazada de manera inmediata y en casos necesarios en un plazo máximo de 48 horas, luego de ello nosotros nos contactaremos con usted.
La Fundación Paquito tiene el derecho absoluto sobre los animales y se reserva el derecho de proporcionar información sobre la aprobación o rechazo de la solicitud.
Todas las mascotas entregadas en adopción La Fundación Paquito lo harán previamente ESTERILIZADAS respetando la normativa legal vigente, política interna y como parte de un manejo responsable de mascotas para evitar su reproducción.

USTED DESEA ADOPTAR

CANINO				FELINO			
Macho		Cachorro		Macho		Cachorro	
Hembra		Adulto		Hembra		Adulto.	
Tamaño:				Tamaño:			

SITUACIÓN PERSONAL

Nombre completo.						CC.	
Dirección:					Teléfono:		
Email:			Edad:			Estado civil:	
Trabaja? Si. ☐ No. ☐		Nombre de sitio de trabajo.				Actividad.	
Ingreso mensual aproximado (opcional)	0 – 200		**Personas con las que vive. (opcional)**	Edad	parentesco:	**Convivencia Inter-especifica**	Perros
	201 -500						Gatos
	501 - mas						Granja
	Otro.						Otros

Tipo de vivienda:						**Permanencia de la mascota**			
Casa		Arrienda.		Cerramiento de malla		Encadenado		Patio/jardín	
Departamento		Propio.		Cerramiento pared		Terraza		Interior	

RELACION CON LOS ANIMALES.

En caso de enfermedad de su mascota Ud.				**La mascota adoptada será para**			
Medica usted.		Espera a que se sane.		Compañía		Juguete niños	
Lleva al veterinario		Lleva al centro de salud.		Cuidar casa		otro	
Dieta que proporcionara a su mascota				**En caso de Viajes largos su mascota.**			
Balanceado		Restos		Viaja con Ud.		Familiar u amigo	
Dieta casera		Otros		Hospedaje		Otro.	

La Adopción fue discutida con su familia.				**La Familia está?**			
Si		**No**		Totalmente de acuerdo		Desacuerdo	
				La aceptan por Ud.		Indiferente	

Firma del Solicitante.	
Atendido por:	
Observaciones:	

Figura D1. Formulario de solicitud de adopción.

	ACTA DE ADOPCIÓN		Fecha elaboración:
			Ultima aprobación
			Revisión:
	Elaborado por:	Revisado por:	Aprobado por:

A los.....días del mes de.....de 20..... Comparecen a la suscripción de la presente acta, por una parte el albergue municipal de animales domésticos, debidamente representado por ..... y por otra, el/la señor/a.....en calidad de ADOPTANTE, a quien de ahora en adelante se le conocerá como el/la ADOPTANTE de la mascota motivo de la presente acta.

La Fundación Paquito "Rescatistas Independientes de Ambato", declara haber recogido en estado abandono un.....con las siguientes características:

Raza:	Sexo:	Edad aproximada:	Tamaño:
ID:	Rasgos característicos:		

Mismo que ha recibido los siguientes tratamientos profilácticos y/o quirúrgicos:

Fecha:	Vacuna:	Desparasitación:	Fecha de revacunación sugerida:	Esterilización:

Y por su parte el/la señor/a.....con C.C. No.....Domiciliado/a en..... de la ciudad de .....teléfono #.....declara tener condiciones suficientes para adoptar a la mascota en cuestión, por lo que La Funfación de Rescate Paquito a través del albergue de Animales Domésticos, le entrega en ADOPCIÓN, y por la presente acta el custodio se compromete a:

- alimentarla adecuadamente
- ubicarla en su domicilio en un espacio adecuado y abrigado
- darle atención veterinaria
- mantener contacto humano y cercano con ella
- respetar todas sus necesidades físico-mentales.

El/la CUSTODIO/A se compromete además a permitir la visita de un representante del La Fundación Paquito, a fin de verificar el buen estado de la mascota y el cumplimiento del presente convenio, así como de la ordenanza municipal que regula el cuidado de mascotas.

En caso de incumplimiento de cualquiera de las obligaciones del presente convenio, La Fundación de Rescate Paquito, queda facultado para retirar a la mascota y denunciar ante la autoridad pertinente si se hubiere cometido cualquier clase de atentado en contra de su bienestar físico o mental.

Para constancia de todo lo expuesto, firman las partes en unidad de acto.

Albergue Municipal de Animales Domésticos. ADOPTANTE.

Figura D2. Acta de adopción.