



UNIVERSIDAD DE GRANADA

Universidad de Granada

INGENIERÍA INFORMÁTICA

Trabajo fin de grado

DESARROLLO DE APLICACIÓN MÓVIL PARA ADOPCIÓN CANINA

Autora

Laura Vega Palacios

Tutor

Juan José Escobar Pérez

Junio de 2024

Resumen

A día de hoy, comenzar un proceso de adopción puede ser algo tedioso debido a lo descentralizada que está toda la información relacionada con las protectoras y refugios. Una persona que quiera adoptar algún animal normalmente va a necesitar consultar diversas plataformas o redes para encontrar una protectora cercana o un animal que se adapte a su estilo de vida.

El desarrollo de este *Trabajo de Fin de Grado* tiene como finalidad la planificación y desarrollo de una aplicación móvil que permita la gestión de adopciones, la información de las protectoras y facilite la comunicación entre los adoptantes y las protectoras. En esta aplicación, se podrán dar de alta protectoras o refugios, con sus correspondientes datos y ubicación. Las protectoras o refugios que estén verificadas por los administradores tendrán la capacidad de subir a la plataforma todos los caninos, que aparecerán en las listas de adopción o acogida dependiendo de lo que se haya marcado para cada uno. También se podrán dar de alta usuarios que busquen adoptar o acoger algún can y se incluirán formularios para dar de alta o editar la información de ellos, con distintos apartados para cumplimentar todos los datos requeridos como la descripción, la raza, la edad, el peso, etc. Los usuarios también dispondrán de formularios para darse de alta y editar su información. La aplicación utilizará un sistema de geolocalización para mostrar a los usuarios resultados dentro de la aplicación y su distancia. Estos resultados se podrán ver en un mapa insertado en la misma gracias a la inclusión de barras de búsqueda y filtros rápidos para acotar resultados según los requisitos del usuario. Por otro lado, la aplicación incluirá un sistema de mensajería para facilitar la comunicación tanto entre usuarios y protectoras como con los administradores. Este permitirá el envío de mensajes de texto e imágenes. Se añadirá, además, un sistema de favoritos y la posibilidad de compartir caninos de la aplicación de forma externa a través de un enlace. Finalmente la aplicación contendrá una página de información legal y de contacto a disposición del usuario.

Abstract

Nowadays, starting an adoption process can be tedious due to the decentralized that is the information related to shelters and rescue organizations. A person looking to adopt an animal usually needs to check different platforms to find a nearby shelter or an animal that fits their lifestyle.

The purpose of this *Final Degree Project* is to plan and develop a mobile application that facilitates adoption management, provides information about shelters, and eases communication between adopters and shelters. In this application, shelters or rescue organizations can register with their corresponding data and location. Shelters or rescues verified by administrators will have the ability to upload all the dogs to the platform, which will appear in adoption or foster lists depending on the status marked for each dog. Users looking to adopt or foster a dog can also register, and forms will be included to add or edit their information, with different sections to fill in all required data such as description, breed, age, weight, etc. Users will also have forms to register and edit their information. The application will use a geolocation system to show users results within the application and their distance. These results can be viewed on an embedded map thanks to the inclusion of search bars and quick filters to narrow down results according to the user's requirements. Additionally, the application will include a messaging system to facilitate communication between users and shelters as well as with administrators. This will allow the sending of text messages and images. Furthermore, a favorites system will be added, along with the ability to share dogs from the application externally through a link. Finally, the application will contain a legal information and contact page available to the user.

Agradecimientos

A mis tíos, que me dieron la oportunidad de estudiar y seguir adelante.

Índice

1	Introducción	13
1.1	Motivación	13
1.2	Objetivos del proyecto	17
2	Gestión del proyecto	21
2.1	Planificación temporal	21
2.2	Estimación de costes	23
3	Estado del arte	25
4	Análisis de requisitos	28
4.1	Definición y especificación de requisitos	28
4.1.1	Requisitos funcionales	28
4.1.2	Requisitos no funcionales	35
4.2	Diagramas de casos de uso	38
5	Herramientas y software utilizado	42
6	Diseño de la Aplicación	45
6.1	La base de datos	45
6.1.1	Guardado de usuarios	45
6.1.2	Guardado de caninos	46
6.1.3	Guardado de direcciones	48
6.1.4	Guardado de chats	49
6.1.5	Guardado de mensajes	50
6.1.6	Guardado de notificaciones	51
6.1.7	Almacenamiento de imágenes	52
6.2	Prototipado de la interfaz de usuario	52
6.2.1	Tema	53
6.2.2	Nombre	54
6.2.3	Logo e iconos	55
6.2.4	Diseños	58
6.3	Arquitectura interna	64
7	Aspecto final de la aplicación	79
8	Conclusiones	109

Índice de tablas

15	Costes del proyecto	24
62	Tabla de usuarios en la base de datos	45
63	Tabla de caninos en la base de datos	47
64	Tabla de direcciones en la base de datos	48
65	Tabla de chats en la base de datos	49
66	Tabla de mensajes en la base de datos	50
67	Tabla de notificaciones en la base de datos	51
68	Tabla de peticiones en la base de datos	51

Índice de figuras

1	Diagrama de Gantt.	21
2	Diagrama de casos de uso: usuario.	39
3	Diagrama de casos de uso: protectora.	40
4	Diagrama de casos de uso: administrador.	41
5	Paleta de colores utilizada en la aplicación.	53
6	Fuente utilizada en la aplicación (<i>Nunito Sans</i>).	54
7	Logos principales de la aplicación.	55
8	Logo por defecto.	55
9	Icono para listas de adopción.	55
10	Icono para listas de acogida.	55
11	Icono de hueso.	56
12	Icono de casa.	56
13	Iconos usuarios.	56
14	Iconos de patas.	57
15	Pantalla de carga.	58
16	Pantalla de inicio de sesión.	58
17	Pantalla de registro/edición de usuario.	59
18	Pantalla de registro/edición de protectora.	59
19	Pantalla de inicio de usuario.	59
20	Pantalla de inicio para protectora.	59
21	Pantalla de lista de caninos.	60
22	Pantalla de lista de usuarios.	60
23	Pantalla del perfil de usuario/administrador.	60
24	Pantalla del perfil de protectora.	60
25	Pantalla del perfil de canino.	61

26	Pantalla de compartir canino.	61
27	Pantalla de mapa y lista.	61
28	Pantalla de redirección a perfil.	61
29	Pantalla de registro/edición de canino.	62
30	Pantalla del menú.	62
31	Pantalla del inicio del administrador.	62
32	Pantalla del pop-up de verificación.	62
33	Pantalla de notificaciones.	63
34	Pantalla de chats.	63
35	Pantalla de recuperación de contraseña.	63
36	Módulos y sus dependencias.	64
37	Módulo <i>appbar</i>	65
38	Módulo <i>chats</i>	66
39	Módulo <i>common</i>	68
40	Módulo <i>condition</i>	69
41	Módulo <i>contact</i>	69
42	Módulo <i>db</i>	70
43	Módulo <i>dog</i>	71
44	Módulo <i>home</i>	72
45	Módulo <i>init</i>	72
46	Módulo <i>legal</i>	73
47	Módulo <i>login</i>	73
48	Módulo <i>message</i>	74
49	Módulo <i>orderby</i>	74
50	Módulo <i>register</i>	75
51	Módulo <i>section</i>	76
52	Módulo <i>services</i>	76
53	Módulo <i>update</i>	77
54	Módulo <i>user</i>	78
55	Pantalla de carga.	79
56	Pantalla de inicio de sesión.	80
57	Pantalla de recuperación de contraseña.	81
58	Pantalla de opciones de registro.	82
59	Pantallas de registro.	83
60	Pantallas de registro con errores.	84
61	Barras de búsqueda de direcciones.	85
62	Pantallas de inicio según el rol.	86
63	Menús laterales según el rol.	87
64	Pantalla de protectora no verificada.	88
65	Pantalla de lista de usuarios.	89

66	Pantalla de mapa.	90
67	Pantalla de lista de caninos.	92
68	Pop-up de filtros para caninos.	93
69	Pantalla de perfil personal.	94
70	Pantallas de actualización de datos personales.	95
71	Pantalla <i>Mis perros</i>	96
72	Pantalla de registro de canino.	97
73	Pantallas de perfil de canino.	98
74	Pantalla que muestra pop-up para marcar/desmarcar como adoptado un canino.	99
75	Pantalla para borrar un canino.	100
76	Pantalla para compartir un canino.	100
77	Pantalla de actualización de caninos.	101
78	Pantalla con notificación de actualización de canino correcta.	102
79	Pantallas del servicio de mensajería.	103
80	Pantalla de información legal.	104
81	Pantalla de información de contacto.	105
82	Pantalla de notificación.	106
83	Vista de administrador.	108

1 Introducción

1.1 Motivación

Todos los años, numerosas familias se animan a incluir una mascota en su círculo. Sin embargo, miles de mascotas son, a su vez, abandonadas por muchas de estas familias en todo el mundo. En 2021, se publicó en *National Geographic* un artículo [1] en el que se critican duramente las cifras de abandono que se mantienen en España año tras año y que rondan los 700 abandonos por día. Anualmente, la Fundación Affinity realiza estudios de abandonos y adopción, donde se pueden consultar estos datos [2] que indican, no sólo que las cifras de abandono no parecen disminuir, sino que además las adopciones disminuyen a medida que pasan los años. Todos estos datos hacen saltar las alarmas de muchas de las asociaciones que luchan por el bienestar y los derechos de los animales. De todos los animales que son abandonados, muchos permanecen en las calles durante el resto de su vida. Otros son recogidos por las autoridades pertinentes y terminan en protectoras o refugios a la espera de encontrar otra familia. Existen organizaciones sin ánimo de lucro que se encargan de ayudar a muchas de las mascotas que están en las calles. Algunas de estas organizaciones son públicas y subvencionadas por el estado. Durante muchos años, en nuestro país no ha existido una buena regulación de los derechos de los animales. Por ejemplo, en algunas comunidades autónomas no existía ningún tipo de ley para regular la cría y compraventa de animales, lo que beneficiaba directamente al tráfico ilegal de mascotas [3]. Para garantizar el bienestar y la protección de los animales, en España recientemente se publicó una ley [4] en la que se tratan principalmente los siguientes puntos:

- **Principios generales**
 - Reconocimiento de los animales como seres dotados de sensibilidad.
 - Fomento de la adopción en lugar de la compra de animales.
 - Prohibición de prácticas que causen sufrimiento o estrés innecesario a los animales.
 - Concienciar acerca del bienestar y respeto animal.
- **Responsabilidad y tenencia responsable de animales de compañía**
 - Establecimiento de requisitos mínimos de bienestar, cuidado y alojamiento.
 - Obligación de identificación y registro de los animales de compañía.
 - Establecimiento de las obligaciones de los propietarios en cuanto a la tenencia responsable, incluyendo la alimentación, hogar y atención veterinaria.
 - Prohibición de mantener animales en condiciones inadecuadas o de privarles de cuidados esenciales.

- **Cría y comercio de animales**
 - Regulación estricta de la cría y comercio de animales de compañía para evitar la explotación y el maltrato.
 - Obligación de los criadores y comerciantes de cumplir con requisitos específicos de bienestar animal.
 - Prohibición de la cría indiscriminada y la venta de animales en tiendas físicas, salvo excepciones debidamente justificadas.
- **Concienciación y medidas del estado contra abandonos y abusos**
 - Promoción de la educación, el respeto y protección de los animales.
 - Campañas de concienciamiento pública sobre el bienestar animal y la tenencia responsable.
 - Tipificación de conductas de maltrato y abandono como infracciones administrativas o penales, además del establecimiento de sanciones proporcionales a la gravedad de las infracciones.
 - Obligación de las administraciones públicas de establecer y mantener refugios y centros de acogida para animales abandonados.
 - Creación de un registro nacional de animales de compañía y establecimientos relacionados con ellos.

Esta ley ha sido un avance muy significativo en la legislación española en materia de protección animal, ayudando a crear un entorno más respetuoso y justo para los animales.

A pesar de que la compra de animales en España es legal, muchas organizaciones de bienestar animal y de los derechos de los animales recomiendan optar por la adopción en lugar de la compra. Esta recomendación tiene su origen en lo habitual que es encontrar criaderos donde los animales no disponen de las condiciones necesarias para vivir adecuadamente. Algunos de estos criaderos carecen de cuidados veterinarios, de una alimentación adecuada y de higiene. En algunos artículos [5] hay disponible más información sobre esta problemática. Si se opta finalmente por la compra, se recomienda visitar los criaderos a los que se va a comprar el animal para poder garantizar que no se están fomentando criaderos ilegales y que cumplen con las ley de bienestar animal.

Cuando se opta por la adopción, es necesario cumplir un proceso que puede variar según la organización a la que se acuda. En internet se pueden encontrar algunas listas con protectoras legítimas en España [6] para evitar encontrarse con posibles fraudes. Hay ciertos pasos que son comunes después de escoger una organización:

- **Elección del animal:** en este paso, se deberán tener en cuenta las necesidades de la mascota para determinar cuál se adapta mejor al estilo de vida

del hogar donde va a residir. Lo habitual en esta etapa es interactuar con diferentes mascotas candidatas a ser adoptadas.

- **Solicitud de adopción:** es común que se tenga que cumplimentar un formulario y realizar una entrevista posterior para asegurar que el animal va a ir a un hogar adecuado.
- **Evaluación del hogar:** se organiza una visita al hogar donde va a ir el animal para garantizar que el entorno es seguro y adecuado. También se verifica que se disponga de todo lo necesario para recibir a la mascota, como comederos, juguetes, cama, etc.
- **Contrato de adopción:** es indispensable para garantizar la protección legal de las mascotas adoptadas. El nuevo dueño debe comprometerse legalmente a proporcionar atención veterinaria y a no abandonar al animal, entre otros puntos. En la mayoría de las organizaciones, es habitual vacunar y colocar un chip al animal antes de que vaya al hogar del nuevo dueño.
- **Recogida del animal:** el día de la recogida, se entrega todo el historial médico del animal (si se dispone de él) y la organización suele proporcionar consejos para los nuevos dueños.
- **Seguimiento:** se requiere un seguimiento posterior a la adopción para garantizar el bienestar del animal y brindar apoyo o asesoramiento al dueño.

Todo este proceso es guiado por la organización correspondiente y es importante completarlo, aunque pueda resultar un poco tedioso. Para iniciar este proceso de adopción, la persona deberá ponerse en contacto previamente con las distintas protectoras o refugios en su área. Esta fase inicial puede presentar diferentes problemas a los adoptantes. En la mayoría de las protectoras, trabajan voluntarios y se sostienen de donaciones, por lo que es común que no cuenten con fondos para generar visibilidad. Muchas protectoras utilizan diversas redes sociales o sus propios sitios web para promocionarse, lo que implica que una persona interesada en adoptar podría tener que utilizar varias plataformas para ponerse en contacto con alguna. A veces, es difícil mantener una comunicación adecuada con diferentes protectoras incluso después de la adopción. Otro problema es que los futuros adoptantes pueden encontrar dificultades al elegir una mascota, ya que la información sobre las diferentes mascotas está dispersa y puede ser una tarea complicada buscar una que cumpla con todos los requisitos del adoptante.

Después de revisar los pasos y los posibles inconvenientes para solicitar un proceso de adopción, se propone una solución en forma de aplicación móvil. Con esta aplicación, se pretende solventar la problemática de la fase inicial de búsqueda de protectoras, proporcionando a los usuarios listas de protectoras cercanas y caninos disponibles, incluyendo filtros para acotar la búsqueda. También se propone como una forma de facilitar a las protectoras la gestión de sus caninos y posibles

adoptantes. En definitiva, el objetivo principal de esta aplicación es centralizar la información de las organizaciones y sus mascotas, además de concienciar e incentivar a los usuarios a adoptar.

1.2 Objetivos del proyecto

Para satisfacer las necesidades del proyecto propuesto, se han definido una serie de objetivos divididos en dos categorías: los obligatorios, necesarios para garantizar el funcionamiento esperado de la aplicación, y los opcionales, que tienen como propósito añadir funcionalidades adicionales o facilitar su uso.

Objetivo 1

Título	<i>Registro de usuarios con diferentes roles</i>
Tipo	Obligatorio
Descripción	La aplicación debe ofrecer la opción de registrarse como usuario o como protectora, los formularios correspondientes y almacenando los datos en la base de datos.

Objetivo 2

Título	<i>Registro de caninos</i>
Tipo	Obligatorio
Descripción	La aplicación debe ofrecer la posibilidad de registrar a varios caninos con diferentes características por de las protectoras y almacenar los datos en la base de datos.

Objetivo 3

Título	<i>Listado de caninos con filtros y búsqueda</i>
Tipo	Obligatorio
Descripción	La aplicación debe mostrar diferentes listas de caninos, permitiendo aplicar filtros rápidos y realizar búsquedas.

Objetivo 4

Título	<i>Listado de usuarios con búsqueda</i>
Tipo	Obligatorio
Descripción	La aplicación debe mostrar diversas listas de usuarios que permitan realizar búsquedas en ellas.

Objetivo 5

Título	<i>Inclusión de sección de caninos favoritos</i>
Tipo	Obligatorio
Descripción	Se debe incluir un botón de favoritos en los perfiles de los caninos que permita a los usuarios añadir a su sección de favoritos a aquellos que les gusten.

Objetivo 6

Título	<i>Inclusión de sistema de mensajería entre usuarios</i>
Tipo	Obligatorio
Descripción	Se debe permitir a los usuarios abrir un chat con otros usuarios dentro de la aplicación. Además, los usuarios deberán recibir una notificación cuando reciban un mensaje.

Objetivo 7

Título	<i>Geolocalización y mapas con resultados</i>
Tipo	Obligatorio
Descripción	Se debe incluir la ubicación para ordenar los resultados de la aplicación según la distancia del usuario. Además, se debe agregar una página con un mapa que muestre la ubicación de los resultados junto con una lista de los mismos.

Objetivo 8

Título	<i>Actualización de datos de caninos</i>
Tipo	Obligatorio
Descripción	Se debe permitir actualizar los atributos de los caninos, como el peso, edad, descripción, etc. Además, se debe proporcionar la opción de marcar si está disponible para adoptar/acoger o si ya ha sido adoptado.

Objetivo 9

Título	<i>Actualización de datos de usuarios</i>
Tipo	Obligatorio
Descripción	Se debe permitir a un usuario actualizar sus datos personales y sus credenciales.

Objetivo 10

Título	<i>Inclusión de barra de búsqueda de direcciones</i>
Tipo	Opcional
Descripción	Posibilidad de que un usuario pueda buscar su dirección y autocompletar automáticamente los diferentes campos del formulario.

Objetivo 11

Título	<i>Diseño de tema oscuro</i>
Tipo	Opcional
Descripción	Posibilidad de que un usuario pueda cambiar al tema oscuro dentro de la aplicación.

Objetivo 12

Título	<i>Inclusión de un blog</i>
Tipo	Opcional
Descripción	Posibilidad de que un usuario pueda añadir publicaciones con imágenes y textos, además de añadir comentarios en las publicaciones para compartir ideas con otros usuarios.

Objetivo 13

Título	<i>Compartición de caninos a través de un enlace</i>
Tipo	Opcional
Descripción	Posibilidad de incluir un botón de compartir que permita al usuario generar un enlace al can correspondiente para poder compartirlo a aplicaciones externas.

Objetivo 14

Título	<i>Envío de imágenes dentro del chat</i>
Tipo	Opcional
Descripción	Posibilidad de enviar imágenes dentro del chat.

2 Gestión del proyecto

2.1 Planificación temporal

En esta sección se definen todas las etapas en las que se dividirá el proyecto, los cuales se resumen en un diagrama de Gantt (ver Figura 1). Este diagrama abarca desde enero de 2024 hasta junio del mismo año, estableciendo un marco temporal para cada fase del proyecto y asignando recursos de manera eficiente para alcanzar los objetivos establecidos.

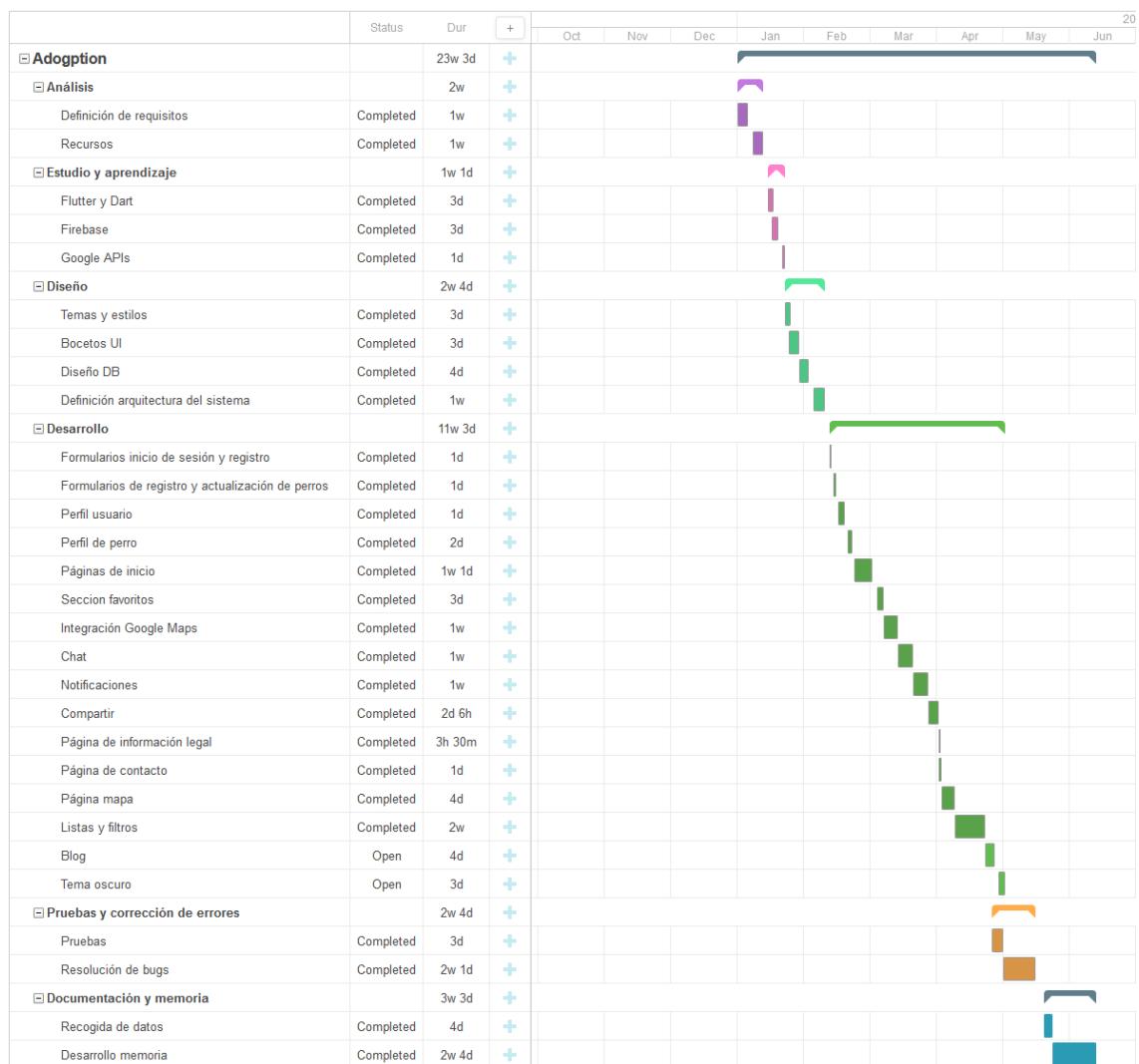


Figura 1: Diagrama de Gantt.

Dentro del diagrama se pueden observar las diferentes etapas del proyecto:

- **Análisis:** esta etapa tiene una duración aproximada de dos semanas, durante la cual se definen todos los requisitos del sistema y se identifican los recursos humanos, de hardware y de software necesarios.
- **Estudio y aprendizaje:** se asigna un breve período previo para aprender a utilizar los lenguajes y servicios que se emplearán en el desarrollo de la aplicación.
- **Diseño:** se destinan casi tres semanas para definir temas, bocetos, la estructura de la base de datos y la arquitectura del sistema.
- **Desarrollo:** esta es la etapa más extensa, dividida en secciones para cada funcionalidad a desarrollar, a las cuales se les asigna un tiempo adecuado según los servicios y componentes involucrados que definirán su dificultad.
- **Pruebas y corrección de errores:** una vez completado el desarrollo, se realizan pruebas exhaustivas para identificar errores y faltas de funcionalidades. Se reserva tiempo suficiente para corregir errores o implementar mejoras de rendimiento si es necesario.
- **Documentación y memoria:** se dedica un período para recopilar todos los datos y elaborar la memoria en LaTeX, documentando el proceso y los resultados obtenidos en las etapas anteriores.

2.2 Estimación de costes

Para el desarrollo del producto, se puede definir un presupuesto que abarque todos los recursos requeridos.

En este proyecto participará únicamente una persona: la autora de la memoria. Se trata de una ingeniera de software, responsable de todos los aspectos del desarrollo. El horario laboral será de lunes a viernes, a media jornada. Para calcular el costo relacionado con el personal, se ha tomado como referencia el sueldo medio de un programador junior, que ronda los *24.000€* anuales, lo que equivale a *12.000€* a media jornada. Haciendo cálculos, al mes corresponden unos *1.000€*. Dado que la duración del proyecto es de 5 meses y medio, el costo total en salarios para el empleado es de *5.500€*.

Para este proyecto, el hardware disponible incluye un ordenador de sobremesa personal y una tablet Android.

- Ordenador de sobremesa, para el desarrollo y las pruebas, con los siguientes componentes:
 - **Procesador:** AMD Ryzen 7 3700X 8-Core Processor - 3.60 GHz
 - **Memoria RAM:** DDR4 3000 2x16GB
 - **Tarjeta gráfica:** NVIDIA GeForce GTX 1660 SUPER
 - **Placa base:** ASUS TUF GAMING B550-PLUS WIFI II
 - **Disipador:** Noctua NH-D15
- Tablet Android Hi9plus, para realizar pruebas en un dispositivo Android físico.

Debido a que el equipo utilizado no es nuevo, se consideran los gastos del equipo teniendo en cuenta que un equipo informático se puede amortizar hasta 8 años. El precio aproximado del ordenador es de unos *1.000€*. Un año tiene 12 meses, lo que resulta en 96 meses en total de amortización, lo que implica que cuesta unos *11€* al mes. Dado que la duración del proyecto es de 5 meses y medio, el costo del ordenador sería de *60,50€* en total. El precio de la tablet es de unos *100€*, así que utilizando los mismos cálculos, el costo de la tablet es de *5,72€* en total.

Con respecto al software, para el alcance de este proyecto y los objetivos que se pretenden alcanzar, al menos durante la primera fase, se ha optado por utilizar las versiones gratuitas del software utilizado, lo que resulta en un coste total de *0€*.

- Firebase - Plan Spark - Guía de planes [7]
- Google APIs - Ofrecen créditos gratuitos de hasta *200€* para cubrir los gastos que se generen por el uso de la API. - Precios [8]

- IDE + Framework - Gratuitos

Tipo de recurso	Recurso	Coste (€)
Personal	Sueldo empleada	5.500
Hardware	Ordenador	60,50
Hardware	Tablet	5,72
Software	Firebase	0
Software	Google APIs	0
Software	IDE + Framework	0
-	-	Total: 5.566,22€

Tabla 15: Costes del proyecto

3 Estado del arte

El principal desafío en el ámbito de la adopción de caninos sigue siendo la sobrepoblación y el abandono masivo que sufren. A pesar de los esfuerzos de las organizaciones y plataformas de adopción, muchas veces los refugios se ven desbordados y no pueden satisfacer la demanda, lo que resulta en una difícil situación para los animales sin hogar. El abandono de perros no solo afecta el bienestar animal, sino que también tiene repercusiones en la salud pública y el medio ambiente. Promover la adopción responsable no solo ayuda a reducir el número de animales sin hogar y mejorar sus vidas, sino que también fomenta una cultura de cuidado y respeto hacia los animales ya la sociedad.

Hoy en día, la tecnología desempeña un papel fundamental para ayudar a las personas a encontrar protectoras o mascotas para adoptar. Existen diferentes aplicaciones que se centran en las adopciones de caninos u otras mascotas que se utilizan en España, algunas han servido como inspiración durante el desarrollo de *Adogption*. A continuación se incluyen algunas aplicaciones gratuitas que recomienda *La Vanguardia* [9]:

Miwuki Pet Shelter

Miwuki Pet Shelter es una plataforma dedicada a la adopción de una amplia variedad de mascotas, incluyendo perros, gatos, roedores, aves, etc. Su principal objetivo es facilitar la conexión entre refugios de animales y posibles adoptantes. La plataforma permite a los refugios crear perfiles detallados de los animales disponibles, proporcionando información detallada sobre su salud, comportamiento y necesidades específicas. Los usuarios pueden explorar la plataforma utilizando filtros avanzados para encontrar mascotas que se adapten a sus preferencias y condiciones de vida. Además de facilitar la adopción, *Miwuki* promueve activamente la responsabilidad animal mediante recursos educativos sobre el cuidado de mascotas, campañas de esterilización y vacunación, así como ofreciendo seguros veterinarios opcionales para cubrir los gastos asociados al cuidado de las mascotas adoptadas [10].

Ventajas

Ésta destaca por su interfaz intuitiva y sus funciones de búsqueda avanzadas. Además, *Miwuki* incluye un blog actualizado regularmente donde se comparten noticias y consejos relacionados con el mundo animal. También cuenta con una sección de informes que presenta cifras y estadísticas sobre el abandono de animales en España, proporcionando así una visión más amplia del problema y fomentando la concienciación sobre la adopción [11].

Inconvenientes

La limitación más importante es que su alcance está principalmente centrado en España.

KLYGO

KLYGO es una aplicación móvil especializada en la adopción y acogida de gatos y perros. Facilita la conexión entre usuarios interesados en adoptar o acoger mascotas y diversas protectoras de animales. Una de las características destacadas de ésta es su sistema de geolocalización integrado, que permite a los usuarios visualizar en un mapa las protectoras cercanas. Además, la aplicación ofrece un chat en tiempo real que facilita la comunicación directa entre los adoptantes potenciales y las protectoras, agilizando así el proceso de adopción [12].

Ventajas

La ventaja más destacable de *KLYGO* es el sistema de geolocalización antes mencionado y la amplia red de refugios y asociaciones colaboradoras de las que disponen. Además, el chat en tiempo real es de gran utilidad para facilitar el contacto instantáneo entre usuarios y protectoras.

Inconvenientes

Uno de los aspectos que algunos usuarios han reportado es lo poco atractiva e intuitiva que es. Además, la información que contienen las mascotas en sus perfiles no es muy clara o específica.

Amazdog

Una de las aplicaciones más llamativas del mercado es *Amazdog* [13]. Esta aplicación se centra en facilitar la adopción de perros y gatos como otras aplicaciones que se han mencionado previamente, sin embargo, esta aplicación ofrece muchas otras funcionalidades que la convierten en un “todo en uno” dentro del mercado de aplicaciones para mascotas. Ésta ofrece fácil acceso a información relacionada con alojamientos o establecimientos que aceptan caninos, incluyendo playas. Por otra parte, contiene una sección de animales perdidos donde una persona puede fácilmente subir un post anunciando la pérdida de su mascota y que esa información se extienda rápidamente.

Ventajas

Es una aplicación muy completa y visualmente atractiva, ofrece funcionalidades muy llamativas y útiles para los dueños de mascotas. A día de hoy, es una de las aplicaciones más utilizadas y recibe actualizaciones y mantenimiento de forma constante.

Inconvenientes

Algunos reportes de usuarios indican que tienen problemas a la hora de recibir notificaciones para mantenerse actualizados acerca del estado de sus adopciones. Además, a la hora de registrarse como usuario, pide muchos datos y puede hacer que una persona se eche para atrás a la hora de escoger esta aplicación.

Tras revisar algunas de las plataformas de adopción más utilizadas en España, se observa que en general ofrecen funcionalidades muy similares y los inconvenientes no son significativos. Con *Adogption*, el objetivo es proporcionar una aplicación que integre las funcionalidades más simples y atractivas, diseñada para motivar a los usuarios a adoptar mascotas de manera fácil y sin complicaciones. El enfoque principal es promover la adopción responsable, facilitando el proceso para los adoptantes y protectoras.

4 Análisis de requisitos

4.1 Definición y especificación de requisitos

En esta sección se detallan los requisitos identificados para la aplicación que se desarrollará. Es fundamental definir y comprender estos requisitos para asegurar resultados óptimos en el desarrollo y diseño del sistema.

4.1.1 Requisitos funcionales

Los Requisitos Funcionales (RF) describen las acciones específicas que el sistema debe realizar, los servicios que debe proporcionar y cómo debe responder a diversas entradas. Estos requisitos se centran en el “qué” del sistema, delineando las funcionalidades clave que los usuarios esperan encontrar en la aplicación.

RF1: Registro de usuarios y protectoras.

Descripción	Ofrece un formulario de registro para usuarios y protectoras. El formulario debe indicar claramente cuales son los datos obligatorios.
Datos de Entrada	Datos ingresados por el usuario en el formulario de registro, incluyendo datos de contacto, correo electrónico y dirección.
Datos de Salida	Datos del usuario almacenados en la base de datos.

RF2: Registro de caninos.

Descripción	Ofrece un formulario de registro para añadir caninos con atributos específicos. El formulario debe indicar claramente cuales son los datos obligatorios.
Datos de Entrada	Datos ingresados por la protectora, tales como raza, peso, edad, color y descripción.
Datos de Salida	Datos del canino almacenados en la base de datos, junto al ID de su protectora.

RF3: Listado de usuarios.

Descripción	Muestra diferentes listas de usuarios en la aplicación, con filtros predeterminados.
Datos de Entrada	Ninguno.
Datos de Salida	Listas de usuarios de la aplicación que cumplen con los filtros.

RF4: Listado de caninos.

Descripción	Muestra diferentes listas de caninos en la aplicación, con filtros predeterminados
Datos de Entrada	Ninguno.
Datos de Salida	Listas de caninos de la aplicación que cumplen con los filtros.

RF5: Filtrado de caninos según sus atributos.

Descripción	Filtrá los resultados de las listas según los valores seleccionados en los filtros que ofrece relacionados con los atributos de los caninos.
Datos de Entrada	Valores de los filtros proporcionados por el usuario.
Datos de Salida	Lista de caninos de la aplicación que cumplen con los filtros.

RF6: Búsqueda en listas de caninos.

Descripción	Filtrá los resultados de las listas según el texto ingresado en una barra de búsqueda, la búsqueda se realiza en diferentes atributos de los caninos.
Datos de Entrada	Texto de búsqueda ingresado por el usuario.
Datos de Salida	Lista de caninos de la aplicación que cumplen con el criterio de búsqueda en alguno de los campos.

RF7: Búsqueda en listas de usuarios.

Descripción	Filtre los resultados de las listas según el texto ingresado en una barra de búsqueda, la búsqueda se realiza en diferentes campos de los usuarios.
Datos de Entrada	Texto de búsqueda ingresado por el usuario.
Datos de Salida	Lista de usuarios de la aplicación que cumplen con el criterio de búsqueda en alguno de los campos.

RF8: Marcar/desmarcar canino como favorito.

Descripción	Permitir a un usuario o protectora marcar como favorito a un canino.
Datos de Entrada	Clic del usuario sobre el botón de favoritos.
Datos de Salida	Datos del canino con la actualización de IDs de usuarios que lo han marcado/desmarcado como favorito.

RF9: Listado de caninos favoritos.

Descripción	Muestra una sección de caninos marcados como favoritos en ese momento por el usuario en su página de inicio.
Datos de Entrada	Ninguno.
Datos de Salida	Lista horizontal de caninos favoritos en la parte inferior de la página de inicio de los usuarios.

RF10: Listado de adopciones recientes.

Descripción	Muestra un <i>swiper</i> con imágenes de caninos marcados como adoptados recientemente.
Datos de Entrada	Ninguno.
Datos de Salida	<i>Swiper</i> con imágenes y nombres de caninos adoptados en la aplicación.

RF11: Cerrar sesión.

Descripción	Se incluye un botón para que un usuario pueda cerrar sesión.
Datos de Entrada	Clic en el botón de cerrar sesión.
Datos de Salida	Mostrar página de inicio de sesión, después de haber eliminado todos los datos relacionados con el usuario iniciado.

RF12: Edición de datos personales y de contacto.

Descripción	Proporciona formularios de edición para los datos del usuario y credenciales.
Datos de Entrada	Datos ingresados por el usuario en el formulario. Puede haber datos sin actualizar.
Datos de Salida	Datos del usuario actualizados en la base de datos.

RF13: Edición de datos de los caninos.

Descripción	Proporciona formularios de edición para los datos de los caninos.
Datos de Entrada	Datos ingresados por el usuario en el formulario. Puede haber datos sin actualizar.
Datos de Salida	Datos del canino actualizados en la base de datos.

RF14: Recuperación de contraseña.

Descripción	Proporciona formulario para recuperar la contraseña.
Datos de Entrada	Correo electrónico ingresado por el usuario.
Datos de Salida	Correo electrónico con las instrucciones de recuperación de contraseña.

RF15: Acceder a página de perfil de usuario/protectora.

Descripción	Proporciona una interfaz que condense toda la información de un usuario en una sola página.
Datos de Entrada	Ninguno.
Datos de Salida	Datos del usuario correspondiente, incluyendo foto de perfil, correo y los caninos (si los tiene).

RF16: Acceder a página de perfil de canino.

Descripción	Proporciona una interfaz que condense toda la información relacionada con un canino en una sola página.
Datos de Entrada	Ninguno.
Datos de Salida	Datos del canino correspondiente, incluyendo foto de perfil, características del canino (edad, raza, peso, descripción, etc.). Además se indicará su ubicación en un mapa, y se añadirán los botones para marcar como favorito, compartir y abrir chat, además de los botones de edición si el canino pertenece al usuario que está visualizando el perfil.

RF17: Compartir caninos a través de enlace.

Descripción	Incluye un botón para compartir los perfiles de los caninos a aplicaciones externas, que se encarga de generar un enlace para el perfil correspondiente.
Datos de Entrada	Ninguno.
Datos de Salida	Url a la aplicación con los parámetros correspondientes para redirigir al perfil de canino.

RF18: Listado de resultados de protectoras insertados en un mapa.

Descripción	Reúne todos los datos en una interfaz que contiene un mapa con diferentes marcadores para cada una de las protectoras y un listado de las mismas debajo de éste.
Datos de Entrada	Ninguno.
Datos de Salida	Mapa con marcadores y listado de protectoras, los elementos de listado tienen la capacidad de mover la cámara a los diferentes marcadores del mapa.

RF18: Listado de chats disponibles.

Descripción	Muestra un listado de chats abiertos dentro de la aplicación.
Datos de Entrada	Ninguno.
Datos de Salida	Listado de chats disponibles en la aplicación, incluyen una preview del último mensaje y datos del usuario correspondiente.

RF19: Envío de mensajes de texto al chat.

Descripción	Permite escribir y mandar mensajes de texto a otro usuario dentro de la aplicación y manda una notificación al receptor.
Datos de Entrada	Texto ingresado por el usuario.
Datos de Salida	Mensaje guardado en la base de datos Se actualiza la tabla de chats de la base de datos con el último mensaje enviado y generar el id del chat si es el primer mensaje. Notificación al usuario receptor.

RF20: Envío de imágenes al chat.

Descripción	Permite mandar imágenes a otro usuario dentro de la aplicación y manda una notificación al receptor.
Datos de Entrada	Imagen adjuntada por el usuario.
Datos de Salida	Imagen guardada además de su referencia en la tabla de mensajes. La tabla de los chats se actualiza con el ID del último mensaje enviado.

RF21: Abrir un chat nuevo.

Descripción	Permite a un usuario abrir un chat con otro usuario a través de un botón de contacto.
Datos de Entrada	Ninguno.
Datos de Salida	Interfaz del chat, con la cabecera que incluye los datos del usuario receptor.

RF24: Eliminar un chat.

Descripción	Permite a un usuario deslizar sobre un chat para eliminarlo de la lista de chats disponibles.
Datos de Entrada	Ninguno.
Datos de Salida	Lista actualizada de los chats.

RF25: Marcar/desmarcar canino como adoptado.

Descripción	Incluye un botón dentro del perfil del canino que permite a una protectora marcar/desmarcar a un canino de la aplicación como adoptado.
Datos de Entrada	Estado nuevo según si se quiere marcar o desmarcar.
Datos de Salida	Perfil del canino actualizado.

RF26: Mostrar la información de contacto.

Descripción	Proporciona una interfaz que reúne datos de contacto de administradores.
Datos de Entrada	Ninguno.
Datos de Salida	Interfaz que incluye correo electrónico y un botón de contacto para abrir un chat con un administrador.

RF37: Mostrar la información legal.

Descripción	Proporciona una interfaz que reúne toda la información legal y advertencias.
Datos de Entrada	Ninguno.
Datos de Salida	Interfaz que incluye toda la información legal de la aplicación.

RF28: Búsqueda de direcciones.

Descripción	Incluye una barra de búsqueda de direcciones que proporciona sugerencias de direcciones y autocompleta el formulario con los datos al seleccionar una.
Datos de Entrada	Una cadena de caracteres que puede contener el nombre de la calle, la ciudad, el código postal, etc.
Datos de Salida	Un listado de direcciones que coinciden con la búsqueda, y que al seleccionar una de ellas, autocomplete los datos del formulario.

RF29: Cambiar a tema oscuro.

Descripción	Ofrece al usuario la posibilidad de cambiar el tema de la aplicación a través de un botón.
Datos de Entrada	Ninguno.
Datos de Salida	Interfaz con el tema oscuro.

RF30: Publicar post en el blog.

Descripción	Incluye una sección donde un usuario subir un post al blog.
Datos de Entrada	Contenido que puede ser texto o imágenes.
Datos de Salida	Post publicado en el blog de la aplicación.

RF31: Comentar post del blog.

Descripción	Ofrece la posibilidad de publicar un comentario en un post en el blog.
Datos de Entrada	Texto que puede incluir emoticonos.
Datos de Salida	Comentario publicado en el post.

4.1.2 Requisitos no funcionales

Los Requisitos No Funcionales (RNF) son responsables de proporcionar al usuario una experiencia robusta y óptima a lo largo de la aplicación. Para el desarrollo de la aplicación se han identificado los siguientes requisitos no funcionales:

RNF1: Limitación en el tiempo de carga de una pantalla.

Tipo	Rendimiento
Descripción	Una pantalla de la aplicación no debe tardar más de tres segundos en cargar completamente.

RNF2: Limitación en el tiempo de respuesta del servidor.

Tipo	Rendimiento
Descripción	El tiempo de respuesta del servidor debe ser menor de dos segundos.

RNF3: Manejo de consultas sin degradación.

Tipo	Rendimiento
Descripción	La base de datos debe realizar consultas simples y complejas en menos de un segundo. Además, debe soportar aproximadamente 10.000 usuarios y consultas concurrentes sin sufrir degradación en el rendimiento.

RNF4: Manejo de búsquedas y filtrado.

Tipo	Rendimiento
Descripción	El tiempo de procesamiento tras realizar una búsqueda y/o filtrado debe ser menor de seis segundos.

RNF5: Proporcionar componentes reutilizables.

Tipo	Escalabilidad
Descripción	Un componente base debe recoger todas las propiedades comunes, como por ejemplo, un componente base para los botones debe incluir el tamaño de la fuente, la forma del botón, colores, etc.

RNF6: Deprecar funcionalidades de forma sencilla.

Tipo	Escalabilidad
Descripción	Las funcionalidades desarrolladas deben poder ser depredadas fácilmente si es necesario, sin dependencias innecesarias entre ellas.

RNF7: Ofrecer una interfaz agradable e intuitiva.

Tipo	Usabilidad
Descripción	La interfaz debe ser limpia, organizada e intuitiva, con colores agradables a la vista. Si es necesario, proporcionar una guía de uso.

RNF8: Garantizar una navegación eficaz.

Tipo	Usabilidad
Descripción	La navegación debe cumplir con la norma de los tres clics: cualquier usuario debe poder alcanzar la información crítica en un máximo de tres clics.

RNF9: Testear con usuarios reales.

Tipo	Usabilidad
Descripción	Se debe testear la aplicación con usuarios reales antes del primer despliegue para realizar posibles correcciones.

RNF10: Actualización de componentes existentes.

Tipo	Confiabilidad
Descripción	La actualización de un componente base debe reflejarse en todos los componentes derivados de ese tipo.

RNF11: Informar a los usuarios de los errores.

Tipo	Confiabilidad
Descripción	Los usuarios deben ser informados con mensajes apropiados si ocurre un error crítico.

RNF12: Mitigar errores.

Tipo	Confiabilidad
Descripción	Implementar mecanismos para mitigar errores conocidos y herramientas para prevenir errores en todas las operaciones críticas, como pueden ser la lectura y escritura en la base de datos.

RNF13: Asegurar la integridad y persistencia de los datos.

Tipo	Confiabilidad
Descripción	Crear copias de seguridad para garantizar que no se pierdan datos en caso de un error crítico.

RNF14: Protección de datos sensibles.

Tipo	Seguridad
Descripción	Cifrar contraseñas y datos sensibles e incluir mecanismos de prevención contra ataques comunes como inyecciones SQL, XSS y CSRF.

RNF15: Garantizar la veracidad de la información.

Tipo	Seguridad
Descripción	Limitar a protectoras verificadas la posibilidad de dar de alta a caninos dentro de la aplicación. Además, proporcionar a los administradores permisos para poder verificar, modificar y eliminar información si es necesario.

RNF16: Documentación.

Tipo	Documentación
Descripción	Se debe almacenar toda la información relevante a lo largo de todas las etapas, ya sea en el código, en documentos aparte o en la memoria del proyecto.

4.2 Diagramas de casos de uso

Un caso de uso es una descripción detallada de cómo los usuarios interactúan con un sistema para lograr un objetivo específico. Estos pueden ser representados gráficamente mediante diagramas que muestran la interacción entre el usuario y el sistema para alcanzar dicho objetivo. A continuación se incluyen los diagramas de casos de uso definidos para cada uno de los roles: usuario, protectora y administrador.

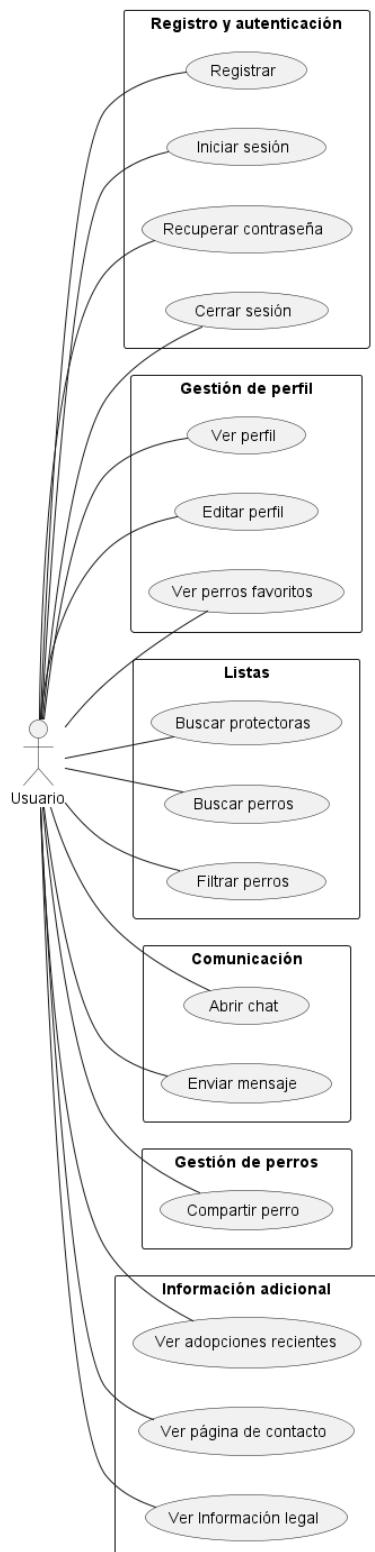


Figura 2: Diagrama de casos de uso: usuario.

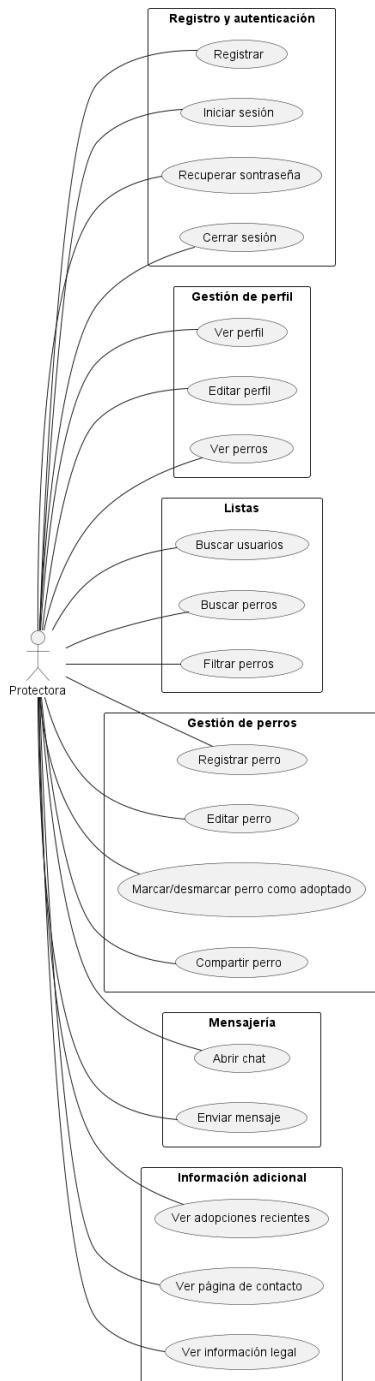


Figura 3: Diagrama de casos de uso: protectora.

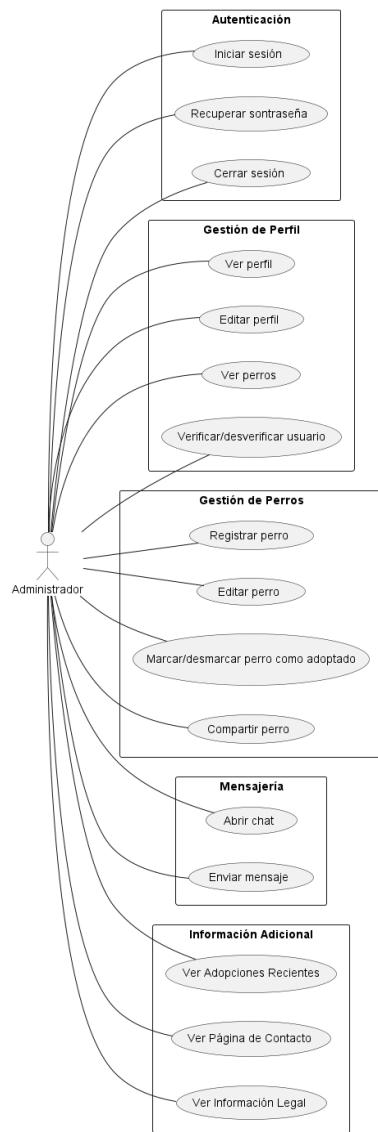


Figura 4: Diagrama de casos de uso: administrador.

5 Herramientas y software utilizado

En esta sección se definen las herramientas y librerías necesarias para el desarrollo de la aplicación.

El sistema operativo utilizado es el que estaba instalado previamente en el ordenador personal, concretamente *Windows 10*.

En la etapa de diseño y documentación han sido necesarias algunas herramientas para generar bocetos y diagramas. Para definir los bocetos se ha trabajado con *Miro* [14], que se trata de una plataforma que proporciona una gran variedad de herramientas como pizarras virtuales, diagramas, mapas mentales, etc. para trabajar de forma colaborativa. Además, esta plataforma ofrece integración con aplicaciones externas o bibliotecas para facilitar tareas como el diseño. Por otra parte, para realizar los diagramas se ha trabajado con *PlantUML* [15]. Esta herramienta permite generar diagramas a partir de texto descriptivo y se puede integrar en diferentes IDE como *Android Studio Hedgehog*.

Para realizar el desarrollo, se ha utilizado *Android Studio Hedgehog*. Este IDE provee herramientas tales como emuladores y *debuggers* que facilitan el desarrollo durante la creación de una aplicación. A día de hoy sigue recibiendo actualizaciones importantes, mantenimiento y está integrado con distintos servicios de Google. Todo esto asegura que las aplicaciones que se desarrollen con éste sean compatibles en una amplia gama de dispositivos.

A la hora de seleccionar el *framework* para utilizar durante el desarrollo, se ha investigado sobre varios *frameworks* disponibles para desarrollar aplicaciones Android. Existen varios disponibles, como son: *React Native* [16], *Flutter* [17], *Kotlin Multiplatform* [18], etc., que ofrecen integración con Android para facilitar el desarrollo. En este proyecto, se ha optado por utilizar *Flutter*, dado que se ha trabajado previamente con él. Éste, además, permite realizar desarrollo multiplataforma de forma simple y ofrece un alto rendimiento.

El lenguaje utilizado ha sido *Dart*. Este lenguaje es muy similar a Java y está orientado a objetos. Se utiliza principalmente para el desarrollo de aplicaciones del lado del cliente. Es un lenguaje bastante popularizado. Uno de los motivos para optar por este lenguaje ha sido din duda el mantenimiento que recibe a día de hoy y su simplicidad para programar. Para añadir funcionalidades extra, se han escogido varias bibliotecas que ofrecen microservicios o widgets.

Por último, para la base de datos, se ha optado por utilizar *Firebase*. Se trata de una plataforma de desarrollo de aplicaciones web muy popularizada, también desarrollada por Google. Ofrece diversos servicios para la autenticación, el *hosting* de la aplicación y bases de datos en tiempo real. Esto lo convierte en el candidato

adecuado para manejar funciones como el chat, la funcionalidad de compartir y otras características que la aplicación requiere.

- **Sistema Operativo:** Windows 11 x64
- **IDE:** Android Studio Hedgehog — 2023.1.1 Patch 2 [19]
- **Framework:** Flutter 3.19.0 - Flutter DEV [17]
- **Lenguaje:** Dart 3.3.0 - Dart DEV [20]
- **Paquetes:**
 - cupertino_icons: ^1.0.2
 - firebase_core: ^2.24.2
 - firebase_database: ^10.3.8
 - firebase_storage:
 - cloud_firestore: ^4.13.6
 - firebase_auth: ^4.15.3
 - responsive_sizer: ^3.3.0+1
 - flutter_login: ^5.0.0
 - google_fonts: ^4.0.4
 - resize: ^1.0.0
 - awesome_notifications: ^0.9.2
 - csc_picker: ^0.2.7
 - map_address_picker: ^0.3.5
 - geocoding: ^3.0.0
 - search_map_location: 0.0.6
 - image_picker: ^1.0.7
 - image_cropper: ^4.0.1
 - path_provider: ^2.1.2
 - flutter_image_compress: ^2.1.0
 - address_form: ^0.0.2
 - address: ^0.1.0+2
 - google_maps_webservice: ^0.0.20-nullsafety.5
 - meta_validator: ^0.0.2
 - geolocator: ^9.0.2
 - dropdown_search: ^5.0.6
 - card_swiper: ^3.0.1
 - filter_list: ^1.0.2
 - flutter_filter_dialog: ^1.2.0
 - choice: ^2.3.2
 - animate_gradient: ^0.0.2+1
 - firebase_messaging: ^14.9.1
 - flutter_chat_bubble: ^2.0.2
 - chat_bubbles: ^1.6.0

- share_plus: ^9.0.0
- app_links: ^6.0.1
- url_launcher: ^6.2.6
- firebase_dynamic_links: ^5.5.4
- go_router: ^14.0.2
- get: ^4.6.6
- linkwell: ^2.0.6
- toastification: ^1.0.0

- **Base de datos:** Firebase database - Firebase console [21]

6 Diseño de la Aplicación

En esta parte se incluye toda la estructura de la aplicación, es decir, tanto la interfaz de usuario como el backend y la base de datos.

6.1 La base de datos

Se deben almacenar todas las entidades y sus atributos, definiendo claramente el tipo de cada una.

6.1.1 Guardado de usuarios

Los usuarios son otra entidad de gran relevancia dentro de la aplicación. Aunque un usuario tiene bastantes datos dentro de la aplicación, muchos de ellos se almacenan en otras tablas y se accede a ellos referenciando sus IDs. Además, para manejar los registros de usuarios se utiliza el servicio de autenticación que proporciona Firebase, el cual facilita las herramientas para registrar usuarios y permite almacenar tanto el correo electrónico como la contraseña, entre otras cosas. En esta tabla se guarda toda la información de los usuarios, independientemente de su rol, que puede ser: **user**, **company** o **admin**, indicando si el usuario es un usuario estándar, una protectora o un administrador, respectivamente.

Campo	Tipo	Dato	Obligatorio
chats	array	IDs de los chats que pertenecen al usuario	Sí
email	string	ID del chat desde el que se manda la notificación	Sí
profilePic	string	URL de la imagen de foto de perfil	Sí
rol	string	Rol del usuario	Sí
username	string	Nombre de usuario	Sí
verified	string	Indica si ha sido verificado por un administrador	Sí, solo si su rol es protectora

Tabla 62: Tabla de usuarios en la base de datos

A continuación, se muestra un ejemplo de usuario guardado en la base de datos:

```
{  
  "chats": [  
    ...  
  ]  
}
```

```

    "wfmRPtHiGvUI9vi9Fs7k"
],
"email": "protectora@test.com",
"profilePic": "",
"rol": "company",
"username": "Protectora super chula",
"verified": true
}

```

6.1.2 Guardado de caninos

Esta entidad es una de las más importantes de la aplicación, ya que es la principal afectada por la mayoría de funcionalidades que se proponen. Principalmente, esta tabla contendrá todas las propiedades de los caninos que se manejarán dentro de la app, aunque hay algunas consideraciones importantes para algunas de ellas. Por ejemplo, para el género se han propuesto dos valores fijos que luego se transformarán en la cadena correspondiente cuando se carguen en la interfaz de usuario. Los valores son **male** y **female**. Se propone esta solución para cuando se decida a futuro añadir traducciones en la aplicación. Por otro lado, para almacenar la edad, se ha optado por hacerlo junto a las unidades, ya que así se le da al usuario la posibilidad de especificar mejor la edad del can (el caso de uso es muy concreto, pero añade valor). Los valores de las unidades son fijos, pudiendo ser:

- Meses
- Años

Lo mismo ocurre con el peso se opta por almacenarlo junto a las unidades. Independientemente de las unidades que marque el usuario, el peso en la base de datos siempre será tratado como Kg para facilitar el filtrado. Cuando se cargue en la interfaz de usuario, se mostrarán las unidades y el valor original, que pueden ser:

- Gramos
- Kg

Una alternativa sería almacenar los caninos favoritos en la tabla de usuarios. Sin embargo, almacenar los usuarios que han marcado al canino como favorito en esta tabla facilita las actualizaciones en tiempo real de los resultados. Se realizaron algunas pruebas con la primera alternativa que requería que para mostrar el número de veces que un canino estaba marcado como favorito, hubiera que consultar todas las entradas existentes de usuarios. Por otra parte, requería recargar constantemente los resultados de los caninos en las listas tras marcar o desmarcar un favorito, lo que reducía considerablemente el rendimiento de la aplicación. Con

este modelo, se puede trabajar únicamente escuchando los cambios en las diferentes entradas de la tabla para los caninos y actualizar la información en pantalla en tiempo real sin comprometer el rendimiento.

Campo	Tipo	Dato	Obligatorio
adopted	booleano	Indica si el can ha sido adoptado	Sí
age	número	Edad	Sí
ageUds	string	Unidades de la edad	Sí
breed	string	Raza	Sí
castrated	booleano	Indica si está castrado	Sí
color	string	Color	Sí
created	marca de tiempo	Fecha y hora de cuándo se ha dado de alta	Sí
description	string	Descripción	No
favoriteBy	array	IDs de usuarios que lo han marcado como favorito	No
forAdoption	booleano	Indica si está disponible para adopción	No
forFoster	booleano	Indica si está disponible para acogida	No
gender	string	Género	Sí
name	string	Nombre	Sí
ownerId	string	ID del propietario	Sí
profilePic	string	URL de la imagen de perfil de la mascota	No
weight	decimal	Peso	Sí
weightUds	string	Unidades de peso	Sí

Tabla 63: Tabla de caninos en la base de datos

A continuación, se muestra un ejemplo real de un canino almacenado en la base de datos:

```
{
  "adopted": false,
  "age": 1,
  "ageUds": "Años",
  "breed": "Breed 2",
  "castrated": true,
```

```

    "color": "Color 1",
    "created": "23 de abril de 2024, 10:56:48 a.m. UTC+2",
    "description": "",
    "favoriteBy": [],
    "forAdoption": true,
    "forFoster": true,
    "gender": "female",
    "name": "Katrina",
    "ownerId": "S4wXwckahiSOLI0Q3exXDVRh2yl2",
    "profilePic": "",
    "weight": 1,
    "weightUds": "KG"
}

```

6.1.3 Guardado de direcciones

Para todas las protectoras, es necesario almacenar la dirección en la que están ubicadas. Esta información es crucial para mostrar a los usuarios la ubicación de cada protectora y generar marcadores en el mapa correspondiente. Por un lado, se almacenarán todas las partes de la dirección, como la ciudad, la calle, el código postal, etc. Por otro lado, cuando un usuario introduzca su dirección, se generarán automáticamente las coordenadas de esa dirección para almacenarlas también, evitando tener que calcularlas cada vez que se carguen los mapas en la interfaz. Asimismo, cuando un usuario actualice su dirección, se actualizarán también sus coordenadas correspondientes en la base de datos.

Para todas las direcciones se genera una entrada en el que su ID corresponde con el del usuario al que pertenece.

Campo	Tipo	Dato	Obligatorio
city	string	Ciudad	Sí
country	string	País (España siempre)	Sí
province	string	Provincia	Sí
street	string	Calle	Sí
street_number	string	Número de la calle	No
zipcode	string	Código postal	Sí
lat	decimal	Latitud	Sí
lng	decimal	Longitud	Sí

Tabla 64: Tabla de direcciones en la base de datos

A continuación, se muestra un ejemplo real de una dirección almacenada en la base de datos:

```
{
  "city": "Murcia",
  "country": "España",
  "lat": 37.9879153,
  "lng": -1.1315578,
  "province": "Murcia",
  "street": "Calle Maestro Alonso",
  "street_number": "4",
  "zipcode": "30005"
}
```

6.1.4 Guardado de chats

Para la tabla de chats, se ha decidido almacenar los IDs de los usuarios implicados. Además, se guarda un array de IDs que indica para quién está disponible el chat en ese momento. Al igual que en otras aplicaciones de mensajería, cuando se elimina un chat, éste simplemente desaparece para el usuario que lo borra, lo que modifica la propiedad de quién puede acceder al chat en ese momento. Sin embargo, los mensajes no se borran. Por ejemplo, si una persona A borra su chat con la persona B y luego vuelve a abrir un chat con esa misma persona, los mensajes anteriores seguirán estando disponibles. Almacenar aquí los IDs de los usuarios implicados permite que, si se proponen chats de grupo, sea más sencillo escalarlo, ya que la base de datos ya lo está soportando. Lo mismo ocurre con la propiedad que hace referencia a qué usuarios pueden ver el chat. Para todos los chats se genera un ID automático, al que se hace referencia desde la tabla de usuarios.

Campo	Tipo	Dato	Obligatorio
availableTo	array	IDs de usuarios que tienen el chat visible en su página	Sí
lastMessage	string	ID del último mensaje mandado en el chat	Sí
users	array	IDs de usuarios que intervienen en el chat	Sí

Tabla 65: Tabla de chats en la base de datos

A continuación se muestra un ejemplo real de un chat almacenado en la base de datos:

```
{
  "availableTo": [
    "kuaKjc6XoiVot1rMeGqBySA5pE13",
    "bu04Vkazs302oVKetugxrvTfRw12"
  ],
  "lastMessage": "BaKmuuo8debUTdSNsvw0",
  "users": [
    "kuaKjc6XoiVot1rMeGqBySA5pE13",
    "bu04Vkazs302oVKetugxrvTfRw12"
  ]
}
```

6.1.5 Guardado de mensajes

A parte de almacenar toda la información de los chats, es necesario almacenar los mensajes que se envían a través de los mismos. Además de guardar el contenido del mensaje, se propone hacer una diferenciación por tipos. Concretamente, dos: **default** e **image**, que indican si un mensaje es texto o imagen, respectivamente. Esta definición de tipos ayudará a determinar cómo cargar el contenido en la interfaz. Para todos los mensajes, se genera un ID automático que se puede referenciar desde la tabla de chats.

Campo	Tipo	Dato	Obligatorio
chatId	string	ID del chat	Sí
from	string	ID del emisor del mensaje	Sí
messageContent	cadena	Contenido del mensaje	Sí
read	booleano	Indica si el receptor a leído el mensaje	Sí
sent	marca de tiempo	Fecha y hora en la que se mandó el mensaje	Sí
to	string	ID del receptor del mensaje	Sí
type	string	Tipo del mensaje	Sí

Tabla 66: Tabla de mensajes en la base de datos

A continuación, se muestra un ejemplo real de un mensaje almacenado en la base de datos:

```
{
  "chatId": "Sm1Ji0chb9uhofa20VW5",
  "from": "kuaKjc6XoiVot1rMeGqBySA5pE13",
```

```

    "messageContent": "Hola, ¿estás?",
    "read": true,
    "sent": "6 de mayo de 2024, 1:57:30 a.m. UTC+2",
    "to": "bu04Vkazs302oVKetugxrvTfRw12",
    "type": "default"
}

```

6.1.6 Guardado de notificaciones

En *Firebase*, a las tablas se les llama colecciones y dentro de cualquier colección se pueden incluir subcolecciones. Usar subcolecciones evita la necesidad de referenciar los datos a través de IDs de entradas de otras tablas. En el caso de la tabla de notificaciones, se ha definido que para cada usuario se genera una entrada en la tabla, y dentro de cada entrada se crea una colección para las peticiones. Al iniciar sesión, se añade un *listener* que escucha todos los cambios de esta tabla, y cuando llega una nueva solicitud, se envía una notificación a través del canal correspondiente. Cuando el usuario receptor recibe la notificación, esta solicitud se marca como vista. El ID de cada entrada en la tabla de notificaciones corresponde al ID del usuario al que pertenece.

Campo	Tipo	Dato	Obligatorio
requests	collection	Colección de peticiones	Sí

Tabla 67: Tabla de notificaciones en la base de datos

A continuación, se muestra la estructura de la colección de peticiones:

Campo	Tipo	Dato	Obligatorio
body	string	Contenido de la notificación	Sí
chatId	string	ID del chat desde el que se manda la notificación	Sí
new	booleano	Indica si la notificación se ha mostrado o no	Sí
summary	string	Vista previa de la notificación	Sí
title	string	Titulo de la notificacion	Sí
userId	string	Usuario que manda la notificación	Sí

Tabla 68: Tabla de peticiones en la base de datos

A continuación, se muestra una petición real almacenada en la base de datos:

```
{  
  "body": "Hola, ¿cómo estás?",  
  "chatId": "Sm1Ji0chb9uhofa20VW5",  
  "new": false,  
  "summary": "Hola, ¿cómo estás?",  
  "title": "Laura Vega Palacios",  
  "userId": "kuaKjc6XoiVot1rMeGqBySA5pE13"  
}
```

6.1.7 Almacenamiento de imágenes

En cuanto al almacenamiento de imágenes, se ha utilizado el servicio que proporciona Firebase llamado *Storage*. Este servicio permite almacenar imágenes, vídeos y audios y viene integrado de manera nativa con otros de sus servicios, como el de *Authentication* mencionado anteriormente. Dentro del almacenamiento reservado para la aplicación, se ha organizado según la siguiente jerarquía de directorios:

- *images*
 - *chats*
 - *profilePics*

Las imágenes de perfil de usuarios y caninos se almacenan en el directorio *profilePics*. Por otro lado, cuando se manda una imagen por chat, ésta se almacena en el directorio *chats*. Para trabajar con las imágenes dentro de la aplicación, se pueden obtener los enlaces que se generan cuando se van almacenando dentro de *Storage*. Estas URLs obtenidas se almacenan en las tablas definidas anteriormente. Si una de las imágenes es borrada, se tiene que borrar su referencia en la tabla de la base de datos y la imagen almacenada en *Storage*.

6.2 Prototipado de la interfaz de usuario

El diseño de la Interfaz de Usuario (UI) se centra en crear una Experiencia de Usuario (UX) intuitiva y atractiva. Inicialmente, se define un estilo y tema que se debe mantener a lo largo de todos los bocetos o prototipos a generar. Es importante tener en cuenta que el diseño sea *responsive* para que se adapte adecuadamente a todos los dispositivos móviles.

6.2.1 Tema

Para los colores principales de la aplicación, se ha optado por escoger el color morado, el cuál dentro de UX puede evocar diferentes sensaciones en el usuario. Muchas marcas de lujo utilizan este color para sus logos, aunque también puede evocar sentimientos de calma o relajación según las tonalidades que se utilicen. Teniendo el morado como color principal, se ha añadido a la paleta su color análogo: el azul. Para finalizar los colores principales, a la paleta se ha incluido el color complementario del azul, que es el naranja. Estos tres colores conforman los elementos de la aplicación: el morado se utiliza para la mayoría de los elementos, indicando que son importantes. El azul se mezcla con el morado en algunos elementos para hacer más dinámicos los fondos y utilizarlo en elementos no tan relevantes. Por último, el naranja es un color que se ha utilizado como acento o para llamar la atención del usuario. Junto a él se podrán utilizar algunos de sus colores análogos como el amarillo o el rojo. Finalmente, a la paleta de colores se ha añadido el color blanco roto pues es adecuado para los fondos de la aplicación.



Figura 5: Paleta de colores utilizada en la aplicación.

Como fuente, se opta por escoger la *Nunito Sans*, que proporciona Google. Su elección se debe a que es una fuente bastante estándar y legible para cualquier tipo de usuario.

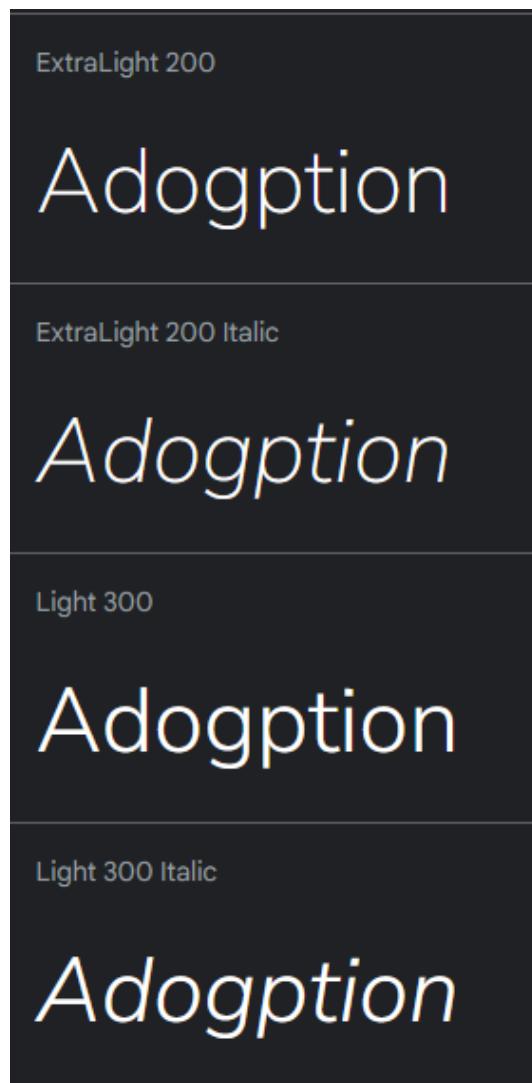


Figura 6: Fuente utilizada en la aplicación (*Nunito Sans*).

6.2.2 Nombre

Como nombre de la aplicación, se ha tenido en cuenta que inicialmente la aplicación está dirigida a la adopción de caninos, así que se propone un juego de palabras. Uniendo las palabras *Adoption* y *Dog*, que son la traducción de las palabras adopción y perro, conformando el nombre: ***Adogption***.

6.2.3 Logo e iconos

La última parte relacionada con el *branding* en este proyecto es la de generación de logos e iconos. En esta sección, se incluye tanto el logo de la aplicación como los iconos que conforman la misma. Algunos de los iconos podrían finalmente no utilizarse. La idea del logo es que se perciba claramente que la aplicación trata sobre ellos. Los *Corgis* son una de las razas más populares en el mundo. Destacan por ser muy amigables y queridos dentro de la cultura *pop*, lo que los convierte en la cara perfecta para llamar la atención. Además del logo principal, se ha diseñado uno animado para incluir en pantallas de carga u otros lugares. Por último, se han hecho algunos iconos por defecto que serán utilizados en perfiles o listas.

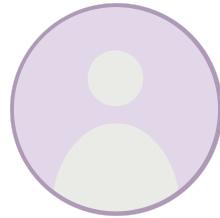


(a) Público.

ADOPTION

(b) Animado.

Figura 7: Logos principales de la aplicación.



(a) Usuario.



(b) Canino.

Figura 8: Logo por defecto.



Figura 9: Icono para listas de adopción.



Figura 10: Icono para listas de acogida.

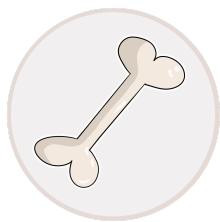


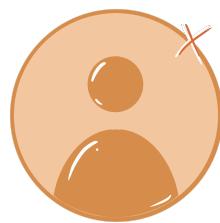
Figura 11: Icono de hueso.



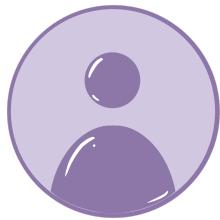
Figura 12: Icono de casa.



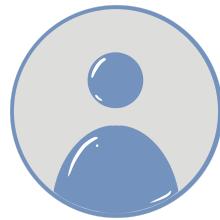
(a) Verificado.



(b) No verificado.



(c) Color morado.



(d) Color azul.

Figura 13: Iconos usuarios.



(a) Color naranja.



(b) Color morado.



(c) Color morado.

Figura 14: Iconos de patas.

6.2.4 Diseños

En esta sección, se incluyen todos los diseños realizados en *Miro* para las pantallas principales de la aplicación y componentes relevantes. En la sección de implementación se especificará la función de cada una de las pantallas.



Figura 15: Pantalla de carga.



Figura 16: Pantalla de inicio de sesión.

Datos personales

- Nombre completo
- Correo electrónico
- Confirma el correo electrónico
- Contraseña
- Confirma contraseña

Confirmar

Figura 17: Pantalla de registro/edición de usuario.

Dirección

- Search
- País: España CP
- Provincia
- Ciudad
- Calle
- Información adicional

Confirmar

Figura 18: Pantalla de registro/edición de protectora.



Figura 19: Pantalla de inicio de usuario.



Figura 20: Pantalla de inicio para protectora.

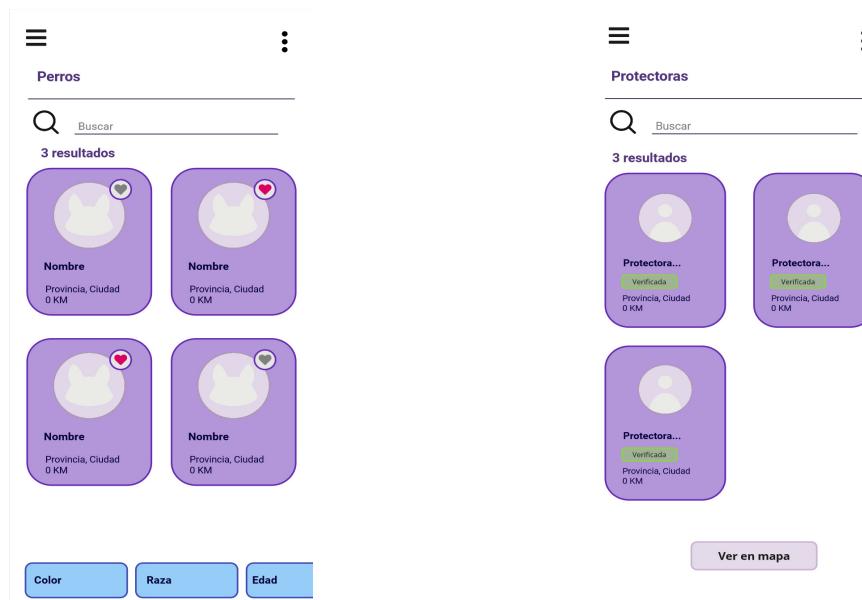


Figura 21: Pantalla de lista de caninos.



Figura 22: Pantalla de lista de usuarios.

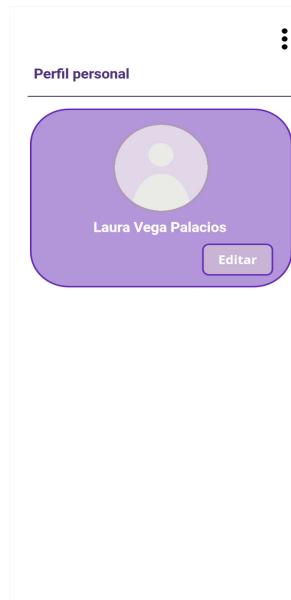


Figura 23: Pantalla del perfil de usuario/administrador.



Figura 24: Pantalla del perfil de protectora.

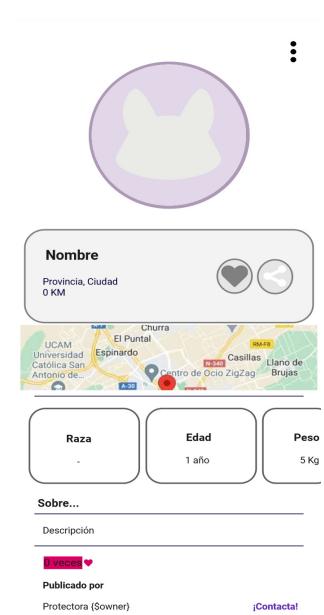


Figura 25: Pantalla del perfil de canino.

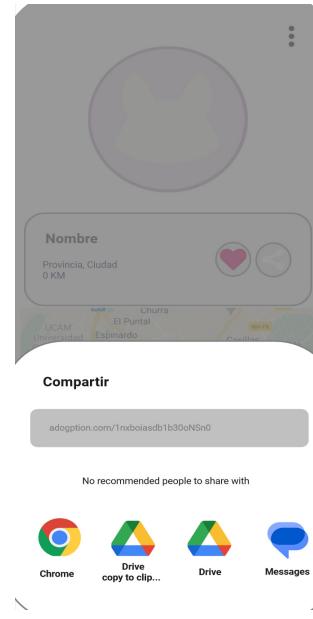


Figura 26: Pantalla de compartir canino.



Figura 27: Pantalla de mapa y lista.

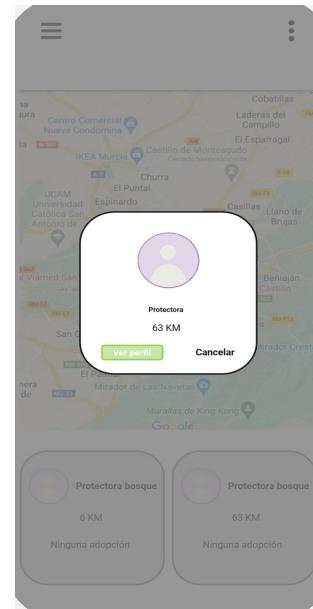


Figura 28: Pantalla de redirección a perfil.



Figura 29: Pantalla de registro/edición de canino.



Figura 30: Pantalla del menú.



Figura 31: Pantalla del inicio del administrador.



Figura 32: Pantalla del pop-up de verificación.

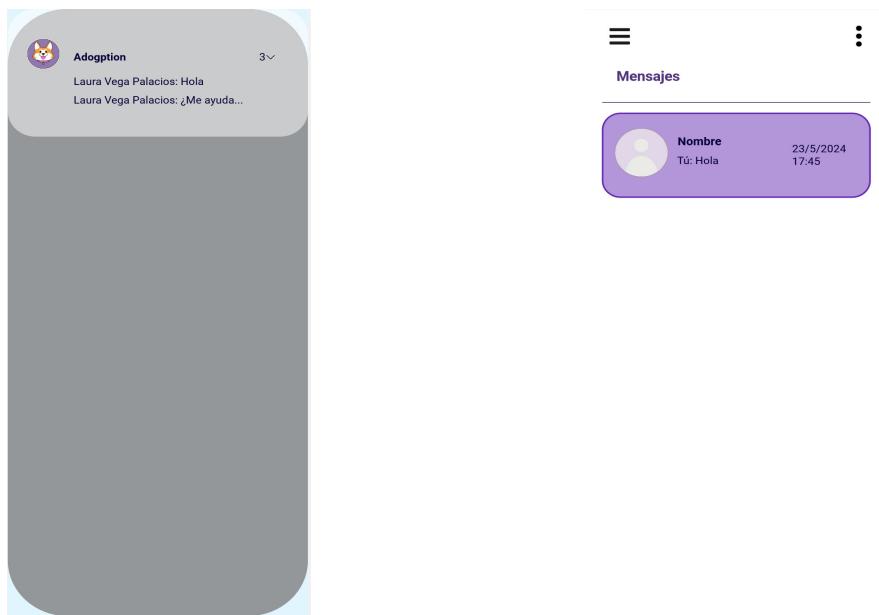


Figura 33: Pantalla de notificaciones.

Figura 34: Pantalla de chats.



Figura 35: Pantalla de recuperación de contraseña.

6.3 Arquitectura interna

Durante el desarrollo se han implementado diferentes módulos, cada uno con una finalidad específica. Mantener los componentes granulados en módulos y submódulos asegura una mejor escalabilidad y mantenibilidad de la aplicación y facilita tareas como refactorizaciones y cambios de estilos, además de evitar muchas repeticiones de código.

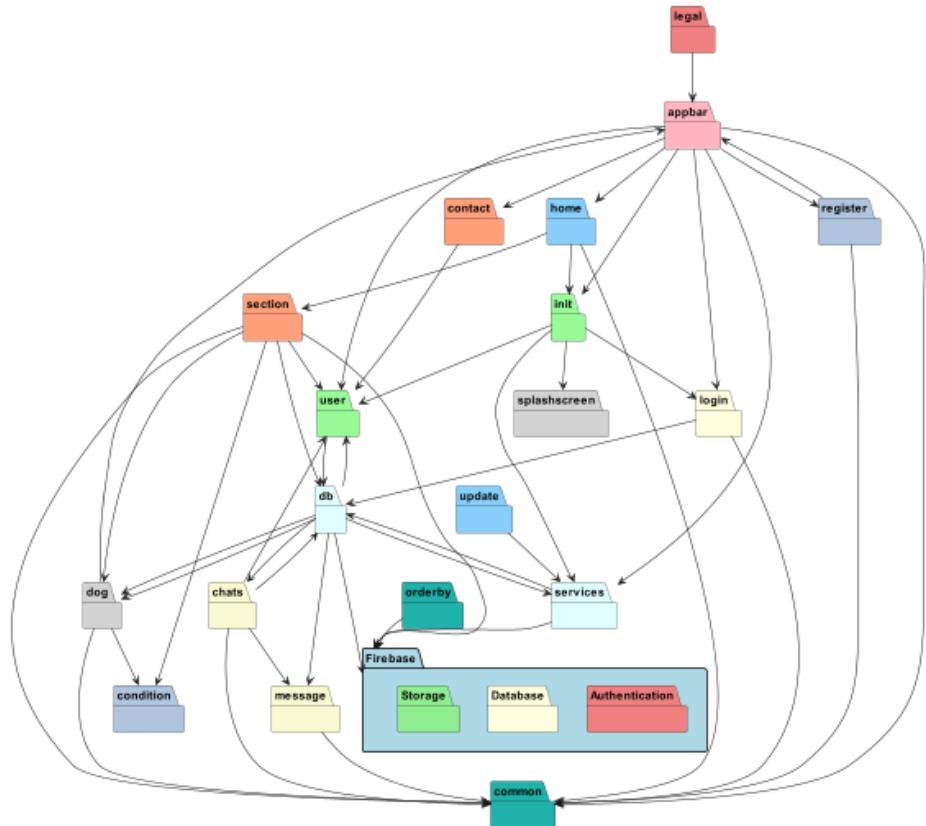


Figura 36: Módulos y sus dependencias.

appbar

Este módulo se utiliza para generar las diferentes *appbar* que se utilizan en las pantallas de la aplicación. Una *appbar* es el componente que se coloca en la parte superior de la pantalla y puede contener botones u otros componentes tales como barras de búsqueda o imágenes. La única clase que contiene el módulo se encarga de instanciar el componente en las diferentes pantallas según el tipo de barra que se necesite.

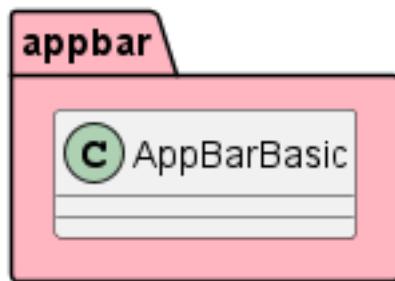


Figura 37: Módulo *appbar*.

chats

En este módulo se incluyen todas las clases que necesita un chat para funcionar correctamente:

- **ChatInput:** componente que permite a los usuarios escribir mensajes y adjuntar imágenes a los chats.
- **ChatRoomPage:** interfaz del chat.
- **ChatsPage:** es la página principal que contiene todos los chats. Contiene todos los chats que un usuario tiene en ese momento, son una lista de *ChatWidget*.
- **ChatWidget:** este componente muestra una previsualización del chat, que incluye la información del usuario, el último mensaje y la fecha y hora en la que se envió. Además, incluye el número de mensajes sin leer del chat.
- **ChatModel:** clase que se encarga de manejar modelos de chat que se generan a partir de toda la información de la base de datos relacionada con ellos.

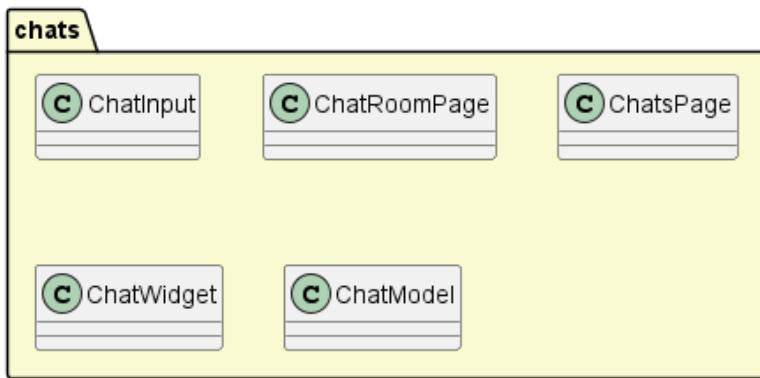


Figura 38: Módulo *chats*.

common

En este módulo se definen todos los componentes básicos que incluyen las diferentes pantallas generadas. Además, se definen diferentes submódulos para cada uno de los componentes creados. Los submódulos contienen una clase base en la que se definen los estilos y funcionalidades básicas, y el resto de las clases extienden de estas clases base. Es importante destacar que, aunque se hayan creado clases propias para los componentes, todas ellas instancian componentes de las bibliotecas por defecto de Flutter, como la biblioteca *material*. Sus clases son las siguientes:

- **button**: en este submódulo se definen todos los botones utilizados en diversos componentes y pantallas de la aplicación. Estos botones tienen diferentes funcionalidades, como la redirección a otras pantallas, la actualización de datos o la visualización de pop-ups.
- **card**: una tarjeta es un componente similar a un contenedor pero con un aspecto más atractivo. Se ha creado un componente base que se reutiliza en diferentes widgets y que contiene este módulo.
- **checkbox**: contiene los diferentes checkboxes desarrollados, junto con su comportamiento asociado.
- **container**: define un contenedor básico con un estilo concreto para reutilizarlo en diferentes lugares.
- **dialog**: agrupa todos los pop-ups desarrollados y su comportamiento correspondiente.
- **drawer**: contiene todos los menús laterales y los componentes asociados. Estos menús incluyen imágenes de perfil y botones que redirigen a otras pantallas.
- **dropdown**: incluye los diferentes dropdowns utilizados en la aplicación.

Algunos de los cuales tienen la funcionalidad de búsqueda de opciones.

- **form**: contiene todos los formularios utilizados en la aplicación.
- **icon**: agrupa iconos básicos con funcionalidad. Por ahora, solo incluye el ícono de favoritos.
- **images**: incluye clases que proporcionan funcionalidades para seleccionar imágenes del dispositivo y componentes para facilitar la inserción de imágenes en la aplicación.
- **input**: contiene componentes que permiten a los usuarios introducir información de diversos tipos, como texto, números o componentes compuestos que contienen dropdowns o checkboxes.
- **maps**: incluye toda la funcionalidad relacionada con los mapas, es decir, tanto la página de mapas como los mapas que se pueden insertar en diferentes partes de la aplicación.
- **padding**: define un *padding* que se reutiliza en toda la aplicación.
- **picker**: incluye componentes relacionados con la selección de direcciones en mapas u opciones similares.
- **searchbar**: define las barras de búsqueda implementadas en la aplicación junto con su funcionalidad.
- **swiper**: contiene el swiper utilizado para mostrar las adopciones recientes.
- **text**: contiene clases que definen textos utilizados en la aplicación, como títulos, subtítulos y avisos.
- **widget**: incluye componentes comunes compuestos por otros componentes comunes.



Figura 39: Módulo *common*.

condition

Este módulo se encarga de gestionar los filtros de las listas de una manera más dinámica. Sus clases son las siguientes:

- **Basic**: define el componente básico para añadir filtros a las listas, encargándose de abrir el pop-up con las opciones del filtro.
- **Gender**: define el filtro para el género.
- **Weight**: define el filtro para el peso.
- **ConditionModel**: clase que se encarga de generar y manejar modelos de condiciones a partir de la información proporcionada por la aplicación o el usuario. Esta clase, además, proporciona un método que se encarga de añadir todos los filtros que haya aplicados en ese momento a la consulta.

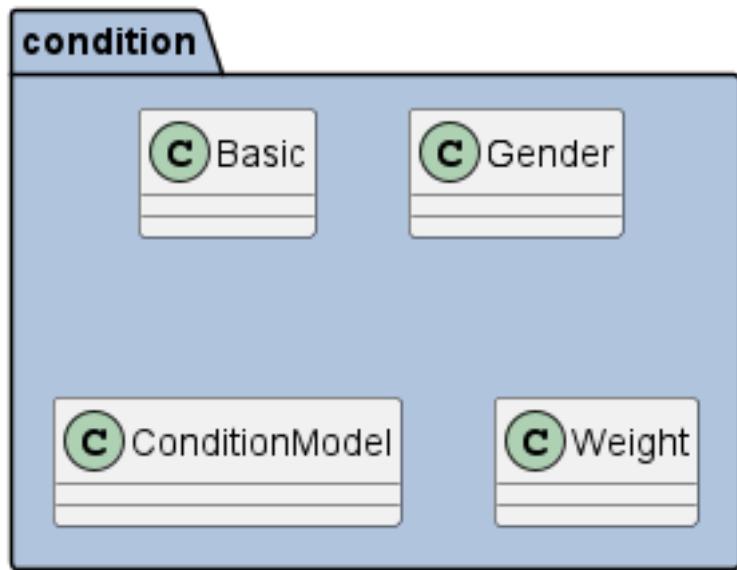


Figura 40: Módulo *condition*.

contact

Aunque este módulo solo contiene una clase que se encarga de mostrar toda la información de contacto con los administradores, se ha definido por separado para poder añadir más clases en un futuro.

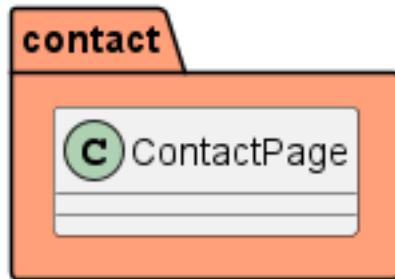


Figura 41: Módulo *contact*.

db

Este módulo es el encargado de comunicarse directamente con la base de datos de Firebase. Contiene diferentes clases para las distintas entidades que existen y para el almacenamiento:

- **DBBase:** contiene los métodos básicos que deben incluir el resto de clases.
- **DBAddresses:** maneja las direcciones y coordenadas.
- **DBChats:** maneja toda la información relacionada con chats.
- **DBDogs:** maneja toda la información relacionada con caninos y su información.
- **DBFavorites:** se encarga exclusivamente de actualizar la información de los favoritos dentro de la tabla de caninos.
- **DBMessages:** maneja toda la información relacionada con mensajes.
- **DBNotifications:** maneja toda la información relacionada con notificaciones.
- **DBStorage:** se encarga de insertar, actualizar y eliminar todas las imágenes relacionadas con alguna de las entidades.
- **DBUsers:** maneja toda la información relacionada con los usuarios.

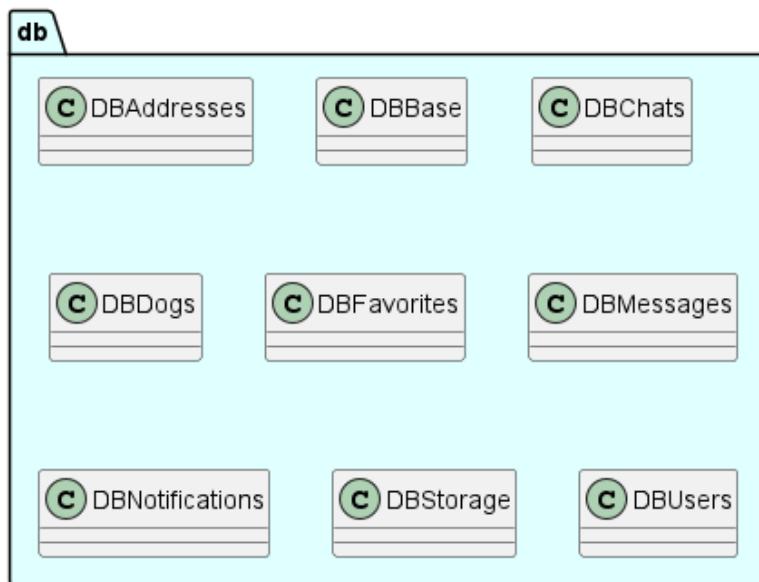


Figura 42: Módulo *db*.

dog

Módulo que contiene las clases principales relacionadas con los caninos. Sus clases son las siguientes:

- **AvailableDogsPage:** pantalla de lista de caninos que contiene los filtros y una barra de búsqueda. Se encarga de manejar las consultas y mostrar los resultados.

- **DogFeature**: widget que se instancia con la información de alguno de los atributos del canino para mostrarlo en el perfil.
- **DogProfile**: pantalla que contiene toda la información de un canino, incluyendo el botón de favoritos y compartir.
- **DogModel**: clase que se encarga de generar y manejar modelos de caninos a partir de la información en la base de datos.

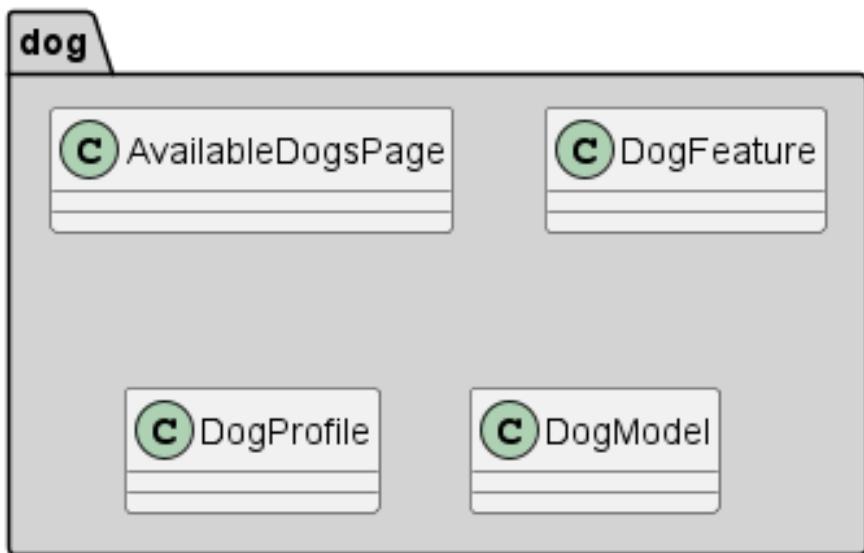


Figura 43: Módulo *dog*.

home

Incluye todas las pantallas de inicio definidas para los diferentes roles. Sus clases son las siguientes:

- **AdminHome**: pantalla de inicio para administradores.
- **MyHome**: pantalla de inicio para usuarios.
- **MyHomePageCompany**: pantalla de inicio para protectoras.

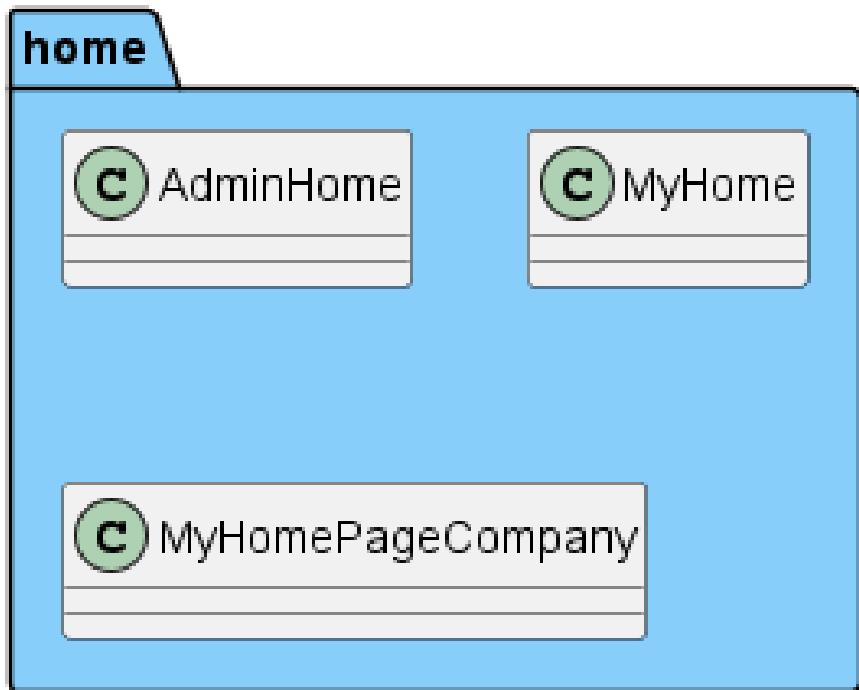


Figura 44: Módulo *home*.

init

La clase contenida en este módulo se encarga de manejar la pantalla que se debe mostrar cuando se abre la aplicación, además de inicializar todos los servicios que lo requieran. La pantalla mostrará la pantalla de carga y, dependiendo de si hay un usuario iniciado, se redirigirá a la pantalla de inicio del rol correspondiente o a la pantalla de inicio de sesión.

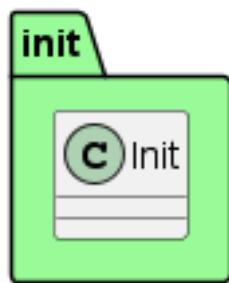


Figura 45: Módulo *init*.

legal

Este módulo solo contiene una pantalla que contiene toda la información legal relacionada con las adopciones y los usos de la aplicación.



Figura 46: Módulo *legal*.

login

Este módulo contiene la página para iniciar sesión y la de recuperación de contraseña.

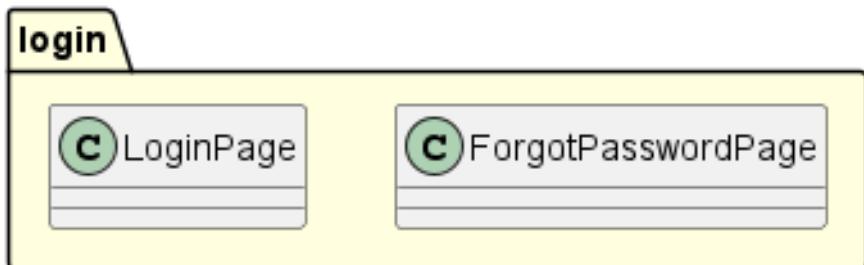


Figura 47: Módulo *login*.

message

En este módulo se definen los widgets relacionados con los mensajes que se utilizan dentro de las salas de chat. Sus clases son las siguientes:

- **DateMessageDivide**: este widget se coloca entre mensajes de días diferentes para indicar la fecha de los mensajes siguientes.
- **MessageWidget**: este widget muestra el contenido del mensaje, que puede ser texto o imagen, y también incluye un ícono que indica si el mensaje ha sido leído o no.

- **MessageModel:** clase que se encarga de generar y manejar modelos de mensajes a partir de la información en la base de datos.

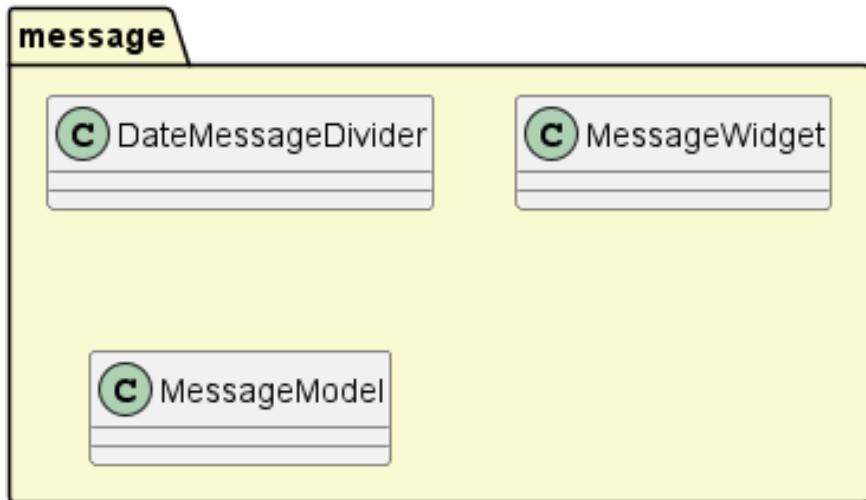


Figura 48: Módulo *message*.

orderby

Este módulo contiene el modelo para añadir la cláusula de orden a las consultas.

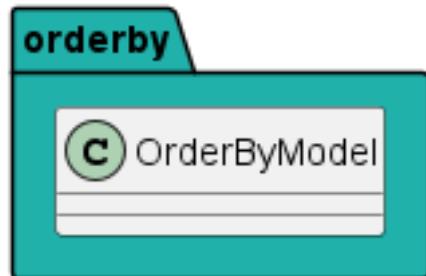


Figura 49: Módulo *orderby*.

register

Este módulo todas las páginas relacionadas con los registros dentro de la app. Cada una de las pantallas incluye un formulario específico que se encarga de todas las validaciones. Sus clases son las siguientes:

- **RegisterAsCompanyPage**: pantalla de registro de protectora.
- **RegisterAsUserPage**: pantalla de registro de usuario.
- **RegisterDog**: pantalla de registro de canino.
- **RegisterPage**: pantalla que ofrece las opciones para registrarse como protectora o usuario.

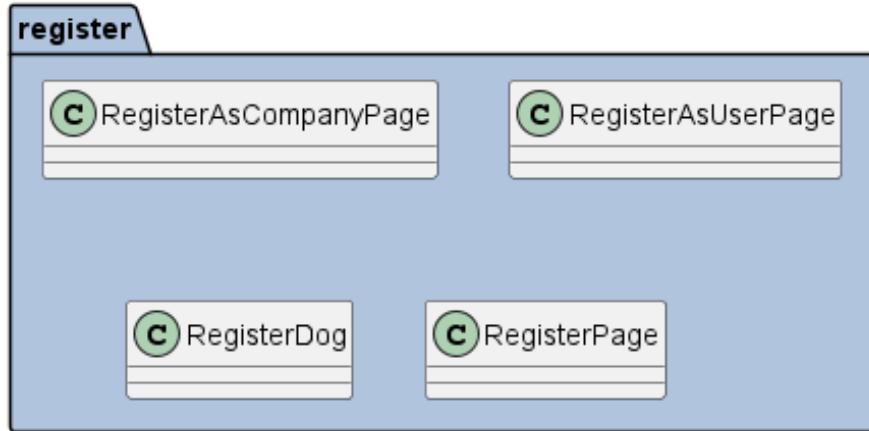


Figura 50: Módulo *register*.

section

Este módulo contiene diferentes widgets que definen secciones que se reutilizan en algunas de las pantallas. Sus clases son las siguientes:

- **AdminCategorySection**: incluye la sección principal de la página de inicio de administrador.
- **CompanyCategorySection**: incluye la sección principal de la página de inicio de protectora.
- **RecentAdoptionsSection**: define la sección de adopciones recientes que se usa en diferentes páginas de inicio.
- **UserCategorySection**: incluye la sección principal de la página de inicio de usuario.
- **UserFavoritesSection**: define la sección de favoritos de un usuario específico.
- **UserOwnedDogsSection**: define la sección de caninos que son de un usuario específico.

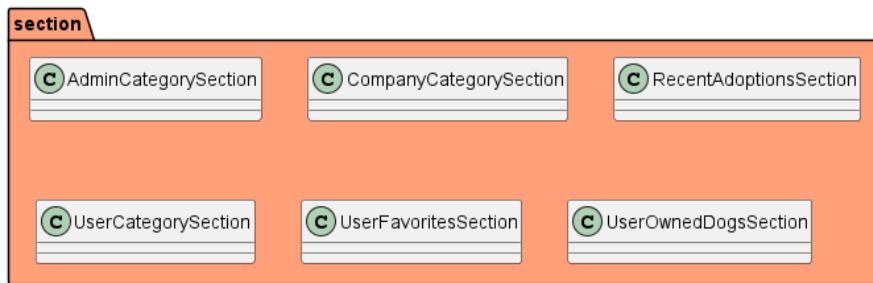


Figura 51: Módulo *section*.

services

En este módulo, se incluyen clases que manejan los servicios que proporciona Flutter o Firebase además de otras clases que son servicios creados específicamente para la aplicación. Sus clases son las siguientes:

- **AddressService**: servicio para manejar los modelos de las direcciones y las coordenadas.
- **AwesomeNotificationService**: clase que maneja el servicio de notificaciones.
- **GeocodingService**: clase que maneja el servicio de *geocoding* que proporciona Google dentro de la app.
- **RoutingService**: servicio para manejar las rutas de la aplicación.
- **Auth - AuthUsers**: clase que maneja el servicio de autenticación que proporciona Firebase.

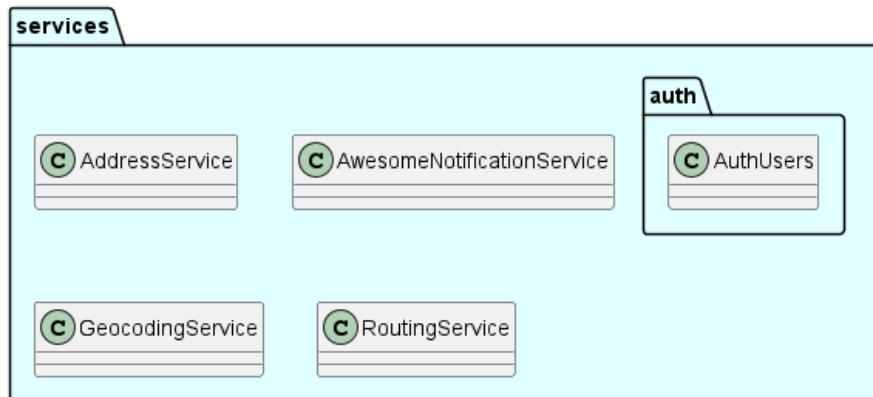


Figura 52: Módulo *services*.

update

Este módulo contiene las páginas que incluyen formularios para la actualización de datos de usuarios y de caninos.

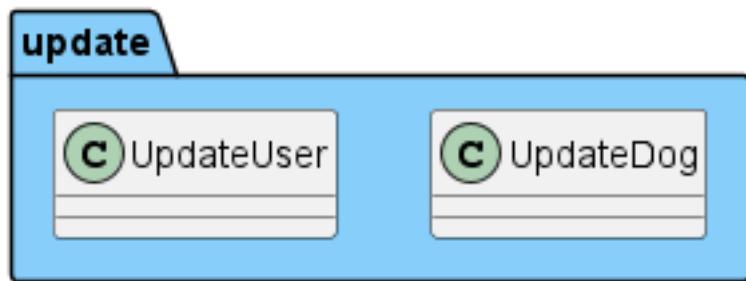


Figura 53: Módulo *update*.

user

Módulo que contiene todas las clases relacionadas con los usuarios y protectoras. Sus clases son las siguientes:

- **CurrentUser**: clase *singleton* que se encarga de manejar todos los datos relacionados con el usuario iniciado a lo largo de la aplicación.
- **Logged UserModel**: clase para manejar el modelo del usuario iniciado.
- **User Model**: clase que se encarga de generar y manejar modelos de usuarios a partir de la información en la base de datos.
- **Profile Page**: pantalla que contiene toda la información personal del usuario. Puede contener un listado de caninos y un botón para abrir una página de mapa dependiendo del rol de usuario.
- **User Map Widget**: elemento que se instancia en las listas usadas en los mapas con toda la información del usuario.
- **Users Page**: pantalla de la lista de usuarios que contiene además la barra de búsqueda. Se encarga de manejar las consultas y mostrar los resultados.

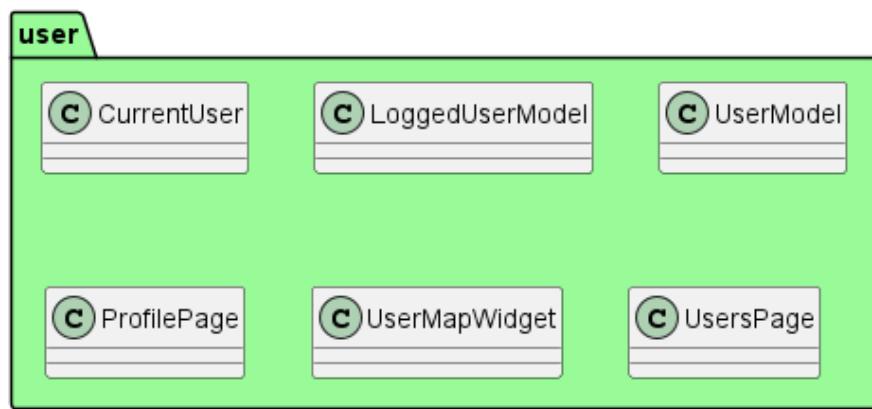


Figura 54: Módulo *user*.

7 Aspecto final de la aplicación

En esta sección se incluyen las pantallas tras finalizar el proceso de desarrollo.

Pantalla de carga

La pantalla de carga es una pantalla de transición que se utiliza justo después de iniciar sesión, mientras se carga toda la información del usuario.



Figura 55: Pantalla de carga.

Inicio de sesión

La pantalla de inicio de sesión muestra un formulario básico para introducir el correo electrónico y la contraseña. Este formulario indicará si hay errores durante el intento de iniciar sesión. El botón de inicio de sesión maneja toda la autenticación mediante el servicio de autenticación de Firebase.

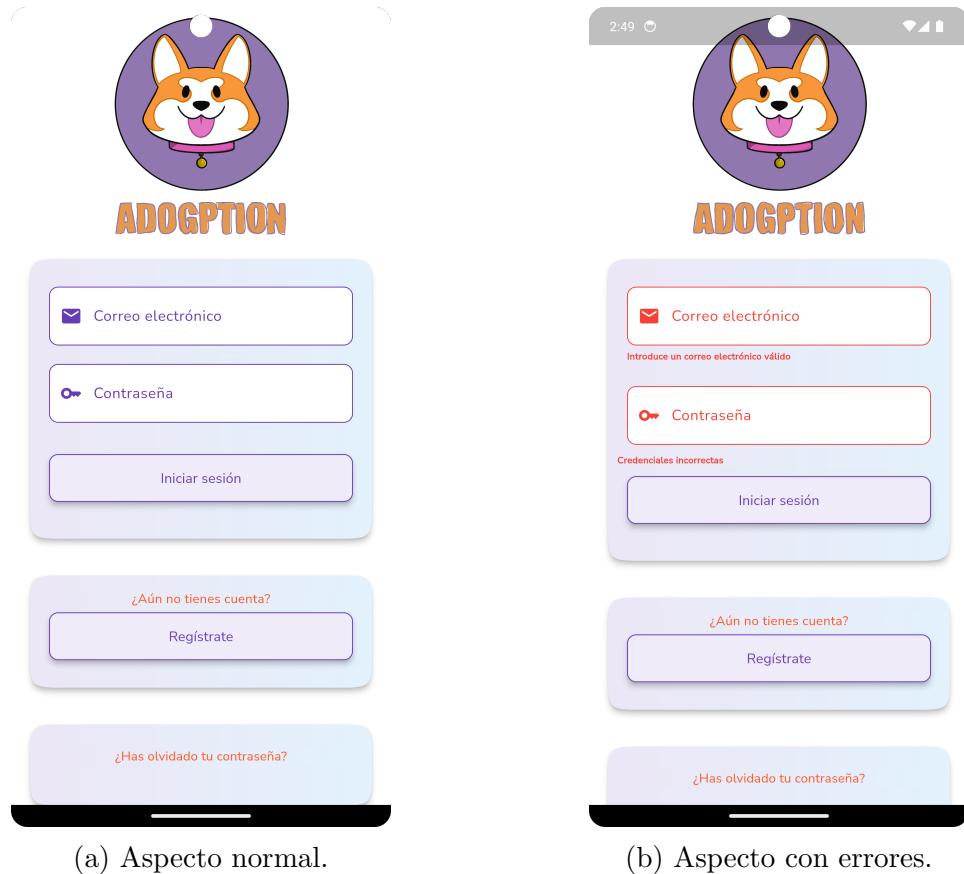


Figura 56: Pantalla de inicio de sesión.

Recuperación de contraseña

Esta pantalla contiene un formulario que solo incluye un campo para introducir el correo electrónico. Si el éste está registrado, el servicio de autenticación de Firebase se encargará de enviar al usuario, protectora o administrador un correo electrónico con las instrucciones para cambiar la contraseña.



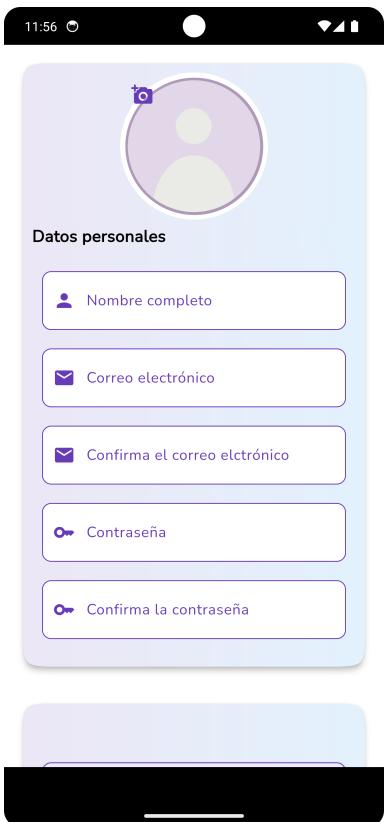
Figura 57: Pantalla de recuperación de contraseña.

Registro

Si se pulsa el botón *Regístrate* que se incluye en la pantalla de inicio de sesión mostrada en la Figura 56, se redirige a una pantalla que muestran dos botones que llevan a los formularios para registrarse como usuario normal o como protectora respectivamente. Ambos formularios de registro piden los mismos datos a excepción de la dirección, que solo se requiere en el caso de las protectoras.



Figura 58: Pantalla de opciones de registro.

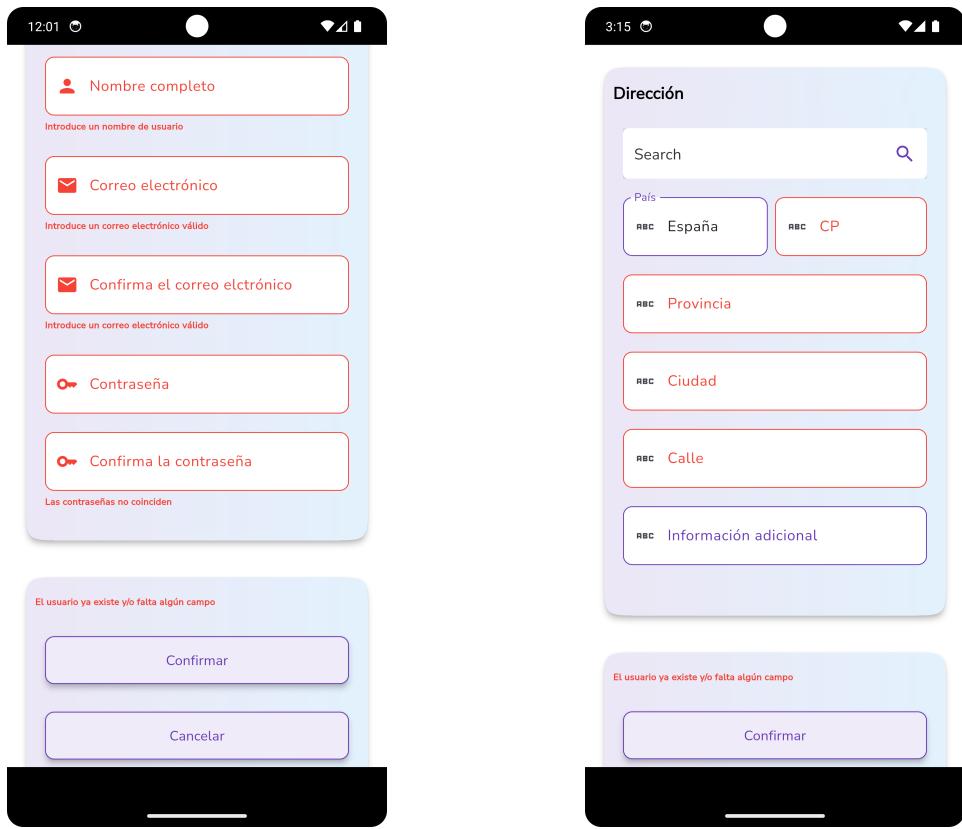


(a) Registro como usuario.



(b) Registro como protectora.

Figura 59: Pantallas de registro.

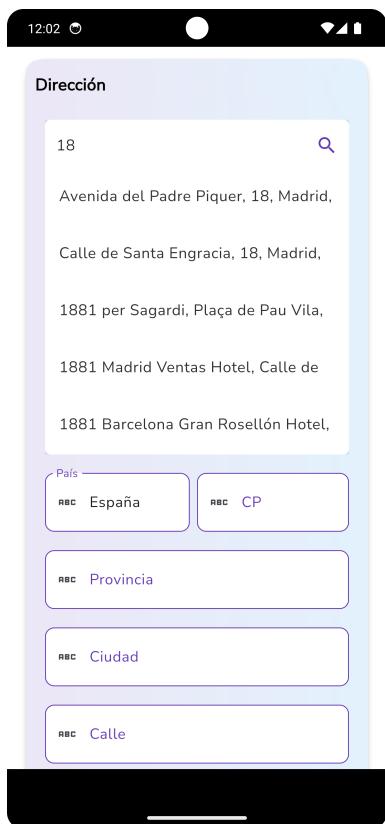


(a) Formulario base.

(b) Formulario de dirección con errores.

Figura 60: Pantallas de registro con errores.

El componente más complejo de estos formularios es la barra de búsqueda de direcciones. Este componente se ha desarrollado en base a otro proporcionado por el paquete *search_map_location* [?], que incluye la funcionalidad básica para buscar direcciones con la API de Google Maps. A este componente se le ha añadido el comportamiento de autocompletar los campos del formulario una vez seleccionada una dirección, además de ajustar todos los estilos según los de la aplicación.



(a) Sugerencias de direcciones.



(b) Autocompletado del formulario.

Figura 61: Barras de búsqueda de direcciones.

Pantallas de inicio

Para cada uno de los roles definidos, se ha creado una pantalla de inicio específica. Ambas pantallas incluyen bastante contenido en común, como la sección de adopciones recientes o la sección de favoritos. Se incluyen listas para mostrar los caninos marcados para adoptar o acoger en ambas pantallas, con la diferencia de que una protectora solo verá sus propios caninos en esas listas.

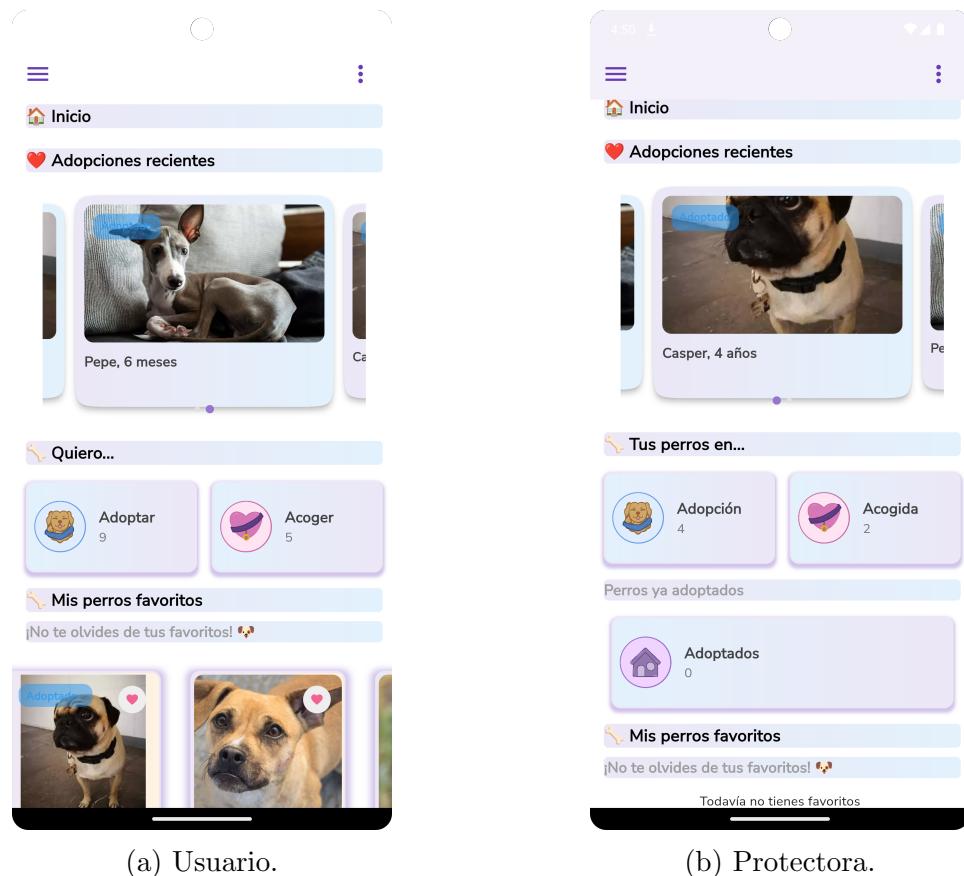
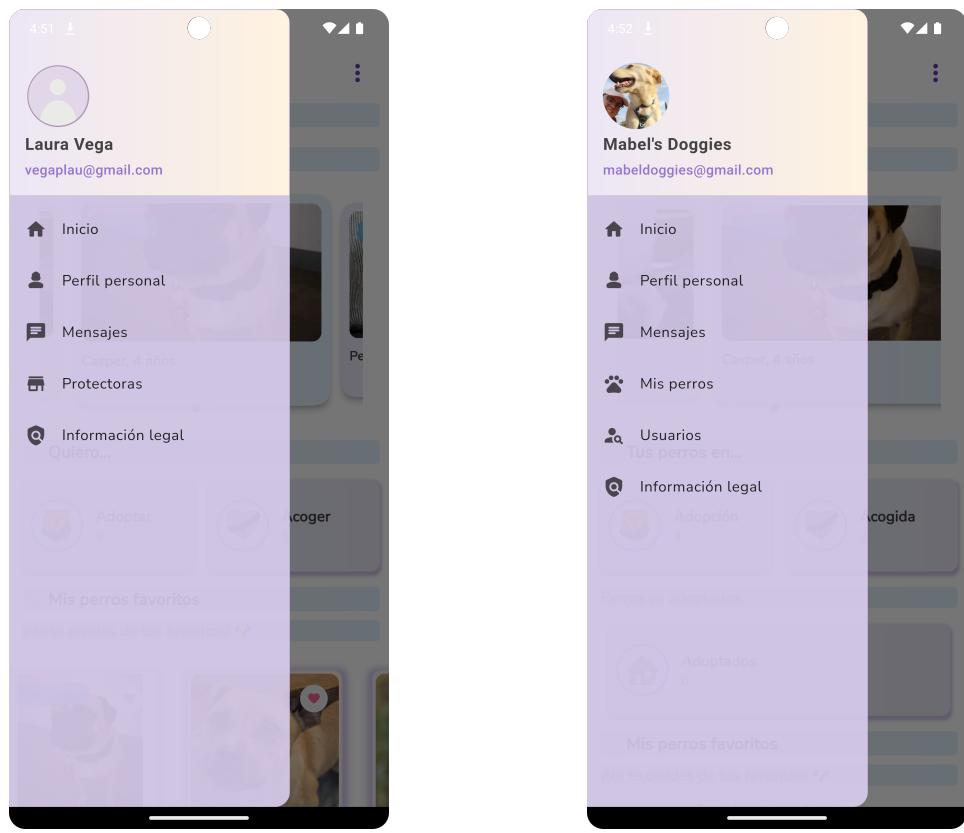


Figura 62: Pantallas de inicio según el rol.

Para cada uno de los roles también se han implementado menús laterales con distintas opciones.



(a) Usuario.

(b) Protectora.

Figura 63: Menús laterales según el rol.

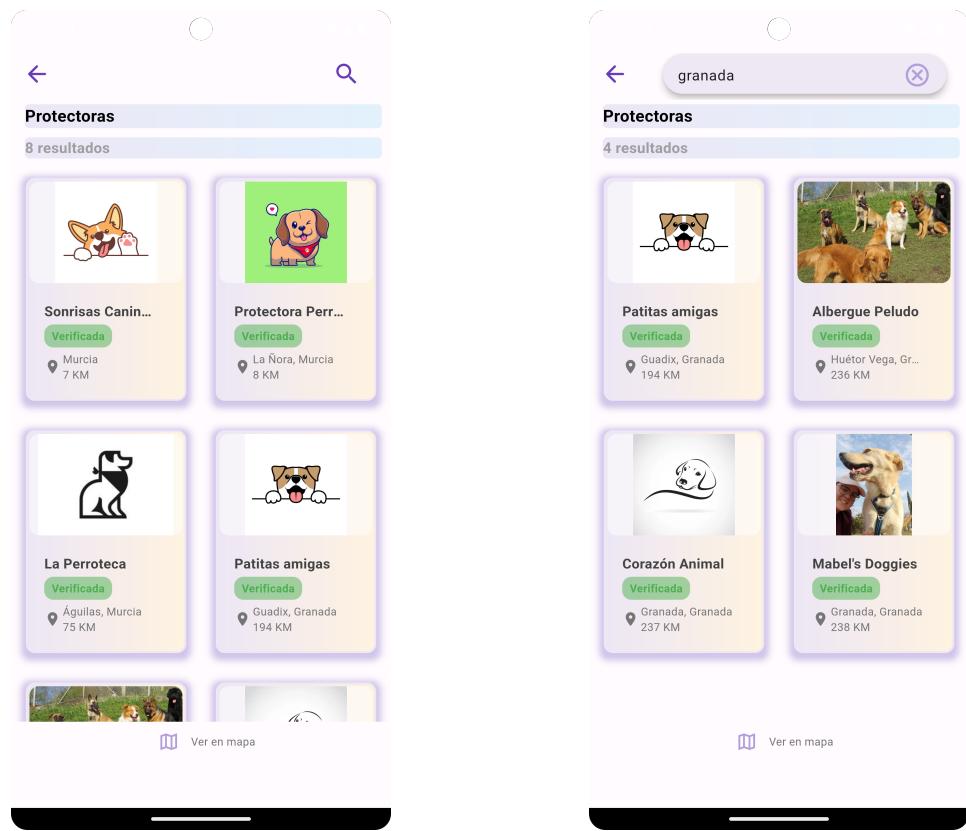
Si una protectora inicia sesión sin estar verificada por un administrador, se muestra una pantalla informándole de que tiene que esperar la verificación para poder seguir usando la aplicación. Aunque la protectora no pueda subir caninos o mensajear, sí podrá abrir un chat con un administrador desde la página de contacto.



Figura 64: Pantalla de protectora no verificada.

Listas de usuarios

Las listas de usuarios pueden incluir tanto usuarios normales como protectoras. Algunas de las listas construidas en la aplicación contienen filtros específicos para mostrar únicamente listas de un rol determinado. Este tipo de listas disponen de una barra de búsqueda que filtra resultados según el texto que se introduzca, el cual busca coincidir con diferentes propiedades del usuario tales como la dirección o el nombre.

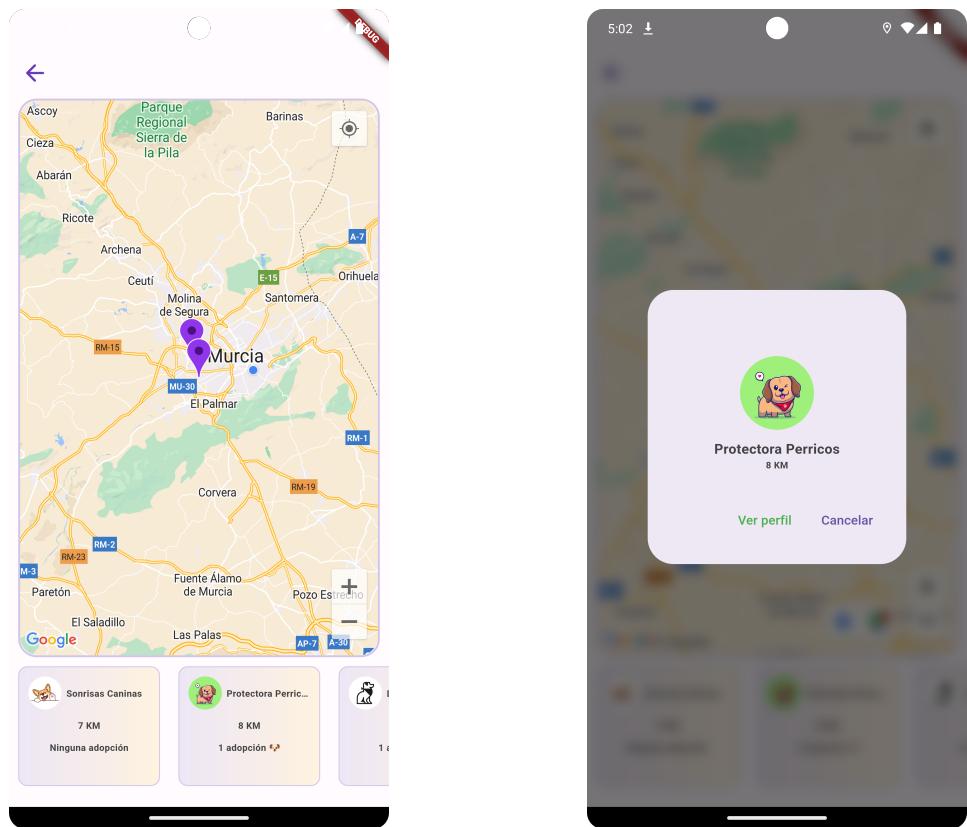


(a) Sin búsqueda.

(b) Con búsqueda.

Figura 65: Pantalla de lista de usuarios.

Estas listas, además, incluyen un botón que redirige a una pantalla que contiene un mapa y la lista actual (incluyendo la búsqueda y filtros aplicados) debajo del mapa, cada elemento de la lista redirige la cámara del mapa encima de su marcador correspondiente. Los marcadores pueden ser pulsados para abrir un pop-up que redirige a la pantalla del perfil de la protectora. Si la lista contiene usuarios sin dirección, no se añadirán al mapa.



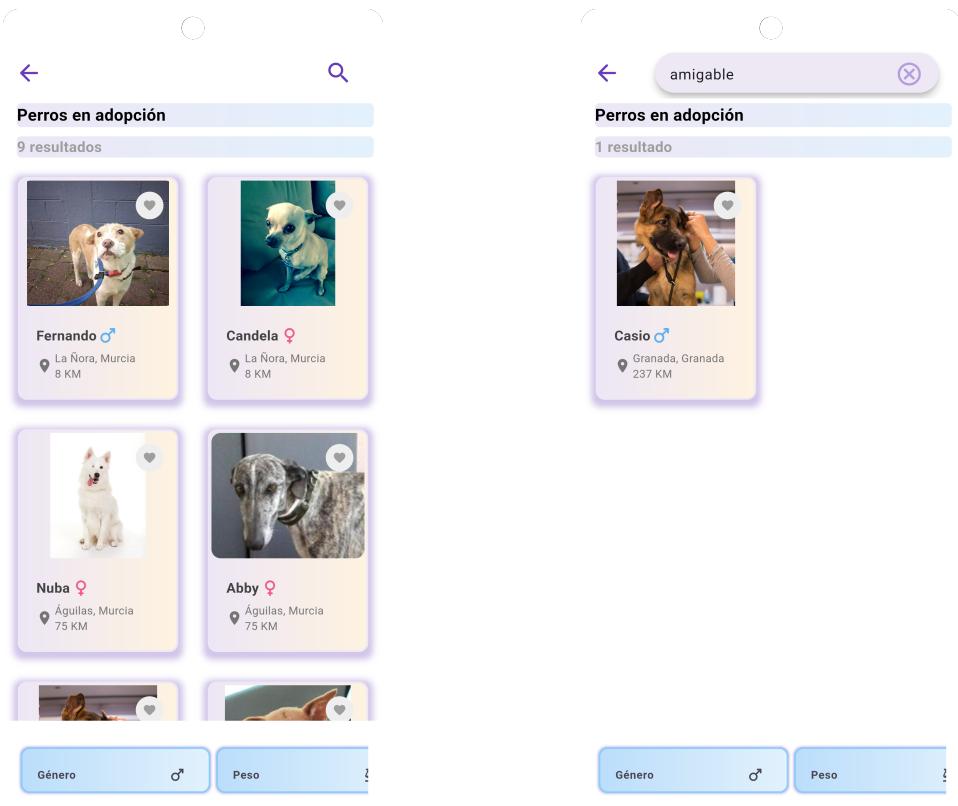
(a) Mapa y lista de usuarios/protectoras.

(b) Acción del marcador.

Figura 66: Pantalla de mapa.

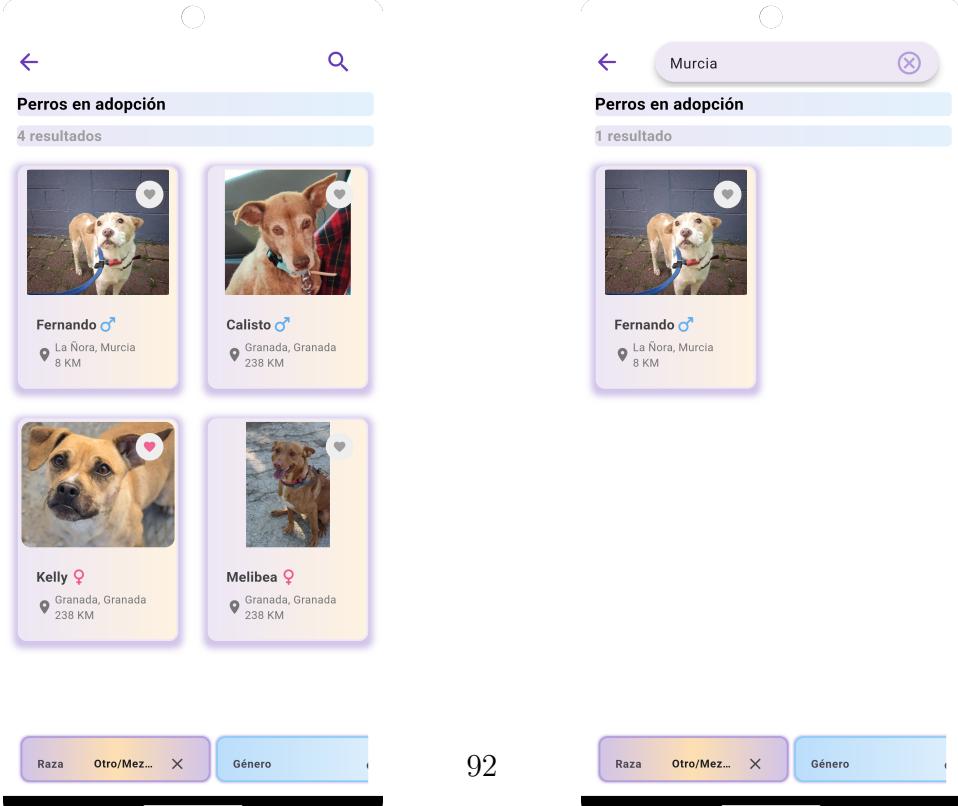
Listas de caninos

Las listas de caninos incluyen cualquier tipo de can que esté dado de alta en la aplicación. Los resultados dependerán de los filtros introducidos por el usuario o los definidos a la hora de construir la lista. Estas listas también incluyen una barra de búsqueda para filtrar resultados además de una barra con varios filtros rápidos que corresponden con algunos de los atributos de los caninos.



(a) Sin búsqueda ni filtros.

(b) Con búsqueda.



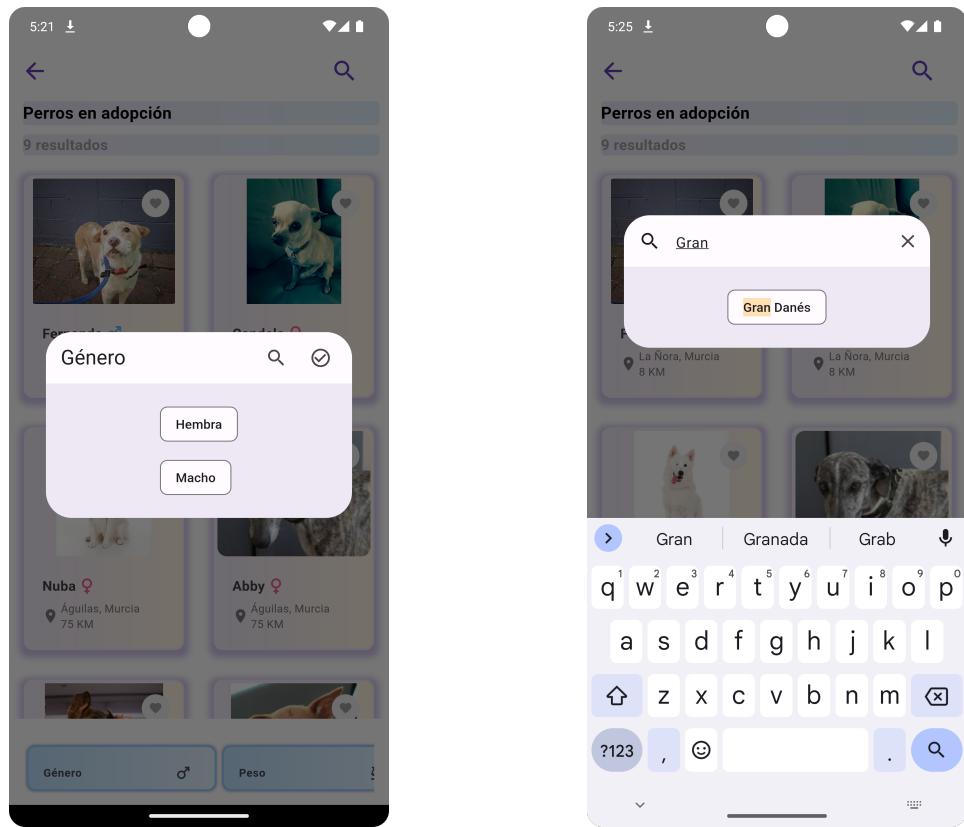
92

(c) Con filtros.

(d) Con búsqueda y filtros.

Figura 67: Pantalla de lista de caninos.

Los widgets de los filtros abren un pop-up con las diferentes opciones disponibles cuando se pulsa en ellos. Además, se incluye una barra de búsqueda para filtrar las opciones disponibles.



(a) Sin búsqueda.

(b) Con búsqueda.

Figura 68: Pop-up de filtros para caninos.

Perfil personal

A continuación, se muestra la pantalla de perfil personal que se genera para cada uno de los usuarios registrados. Dependiendo de algunas condiciones se mostrarán unos botones u otros:

- Si el usuario es una protectora, se incluye un botón para abrir un mapa con dicha protectora ubicada en él.
- Si el perfil que se visita no es el propio, se incluye un botón de contacto para abrir un chat con el usuario.

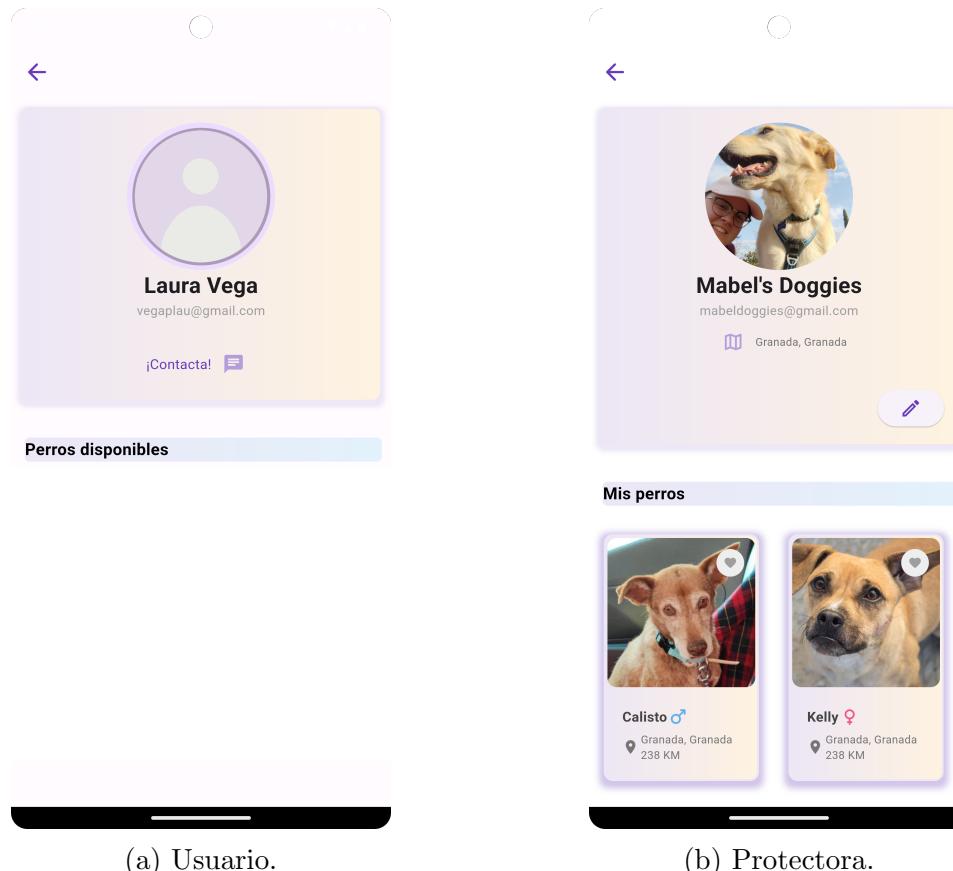
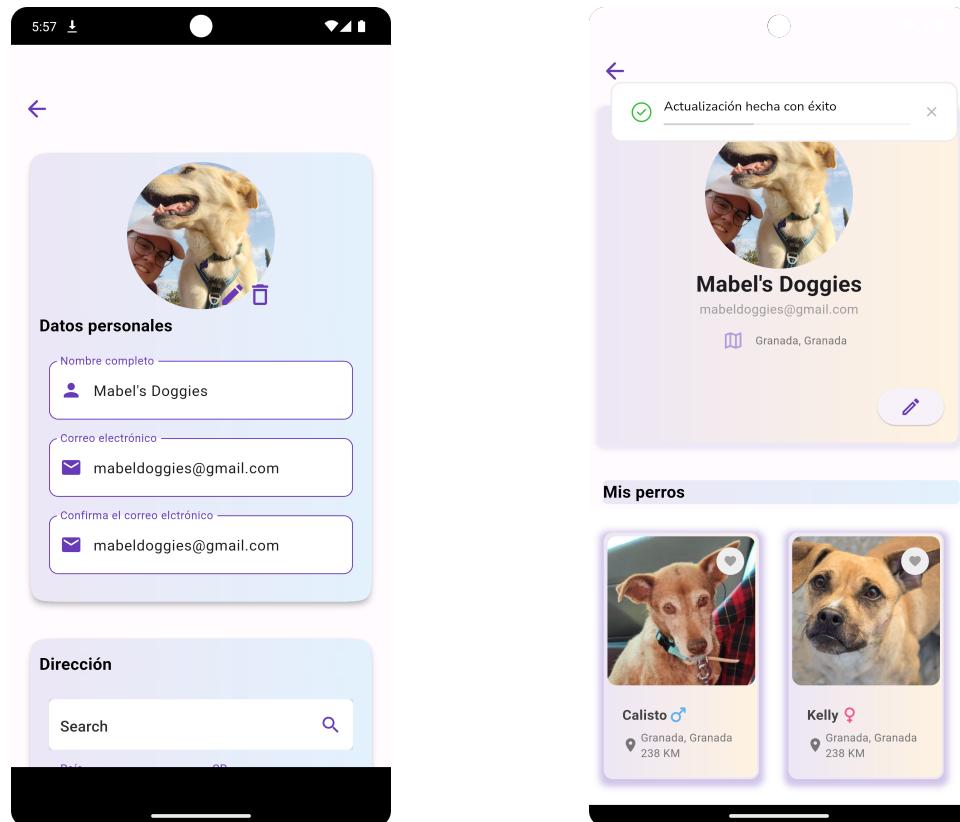


Figura 69: Pantalla de perfil personal.

Actualización de datos de usuario

Los formularios para actualizar los datos de usuario son casi idénticos a los proporcionados para registrarse. En primer lugar, los formularios cargan todos los datos actuales del usuario y permiten que se introduzcan todos los cambios que se requieran. Tras confirmar los cambios, si la actualización es correcta, el usuario es redirigido a su perfil y una notificación es mostrada indicando que la actualización se ha hecho correctamente.



(a) Formulario de actualización de los datos de usuario.

(b) Notificación de correcta actualización de los datos de un usuario.

Figura 70: Pantallas de actualización de datos personales.

Registro de caninos

Las protectoras pueden añadir caninos a la aplicación desde su lista personal llamada *Mis perros*. En esta pantalla, se incluye una barra superior con un botón específico que redirige al formulario de registro para caninos. Éste incluye todos los atributos que requiere un canino para ser dado de alta.

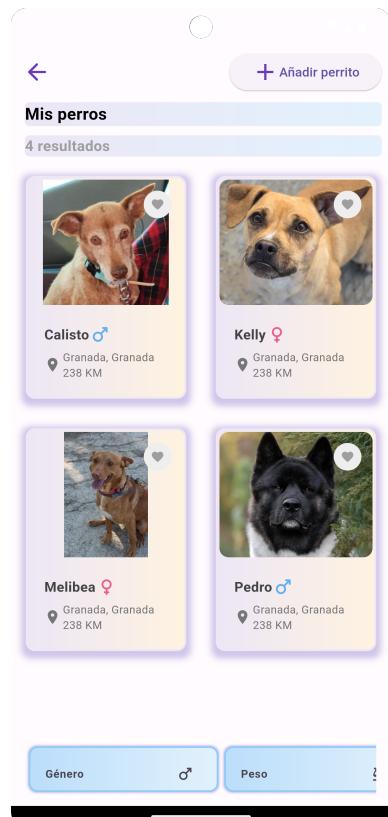


Figura 71: Pantalla *Mis perros*.

Datos del canino

Nombre: Kelly

Raza: Otro/Mezcla

Color: Marrón

Edad: 1 Años

Peso: 2 Kg

Sexo: Hembra

* Para que aparezca en las listas por lo menos una opción de esta sección tiene que estar marcada

Para adoptar:

Para acoger:

Castración hecha:

Descripción: Kelly es una perrita muy agradable!

Confirmar

Cancelar

Figura 72: Pantalla de registro de canino.

Perfil de canino

Para mostrar toda la información de un canino a un usuario, se ha condensado toda la información en una pantalla. En ella, se incluye toda la información relacionada con el canino, tales como nombre, raza, peso, etc. Se incluye también un mapa que muestra la ubicación de la protectora a la que pertenece, además de un botón de contacto para abrir chat con la protectora. Si el usuario que visita el perfil es el dueño del canino, puede visualizar una sección inferior con diferentes botones para abrir la edición del perfil, eliminar el canino de la aplicación o marcarlo como adoptado.

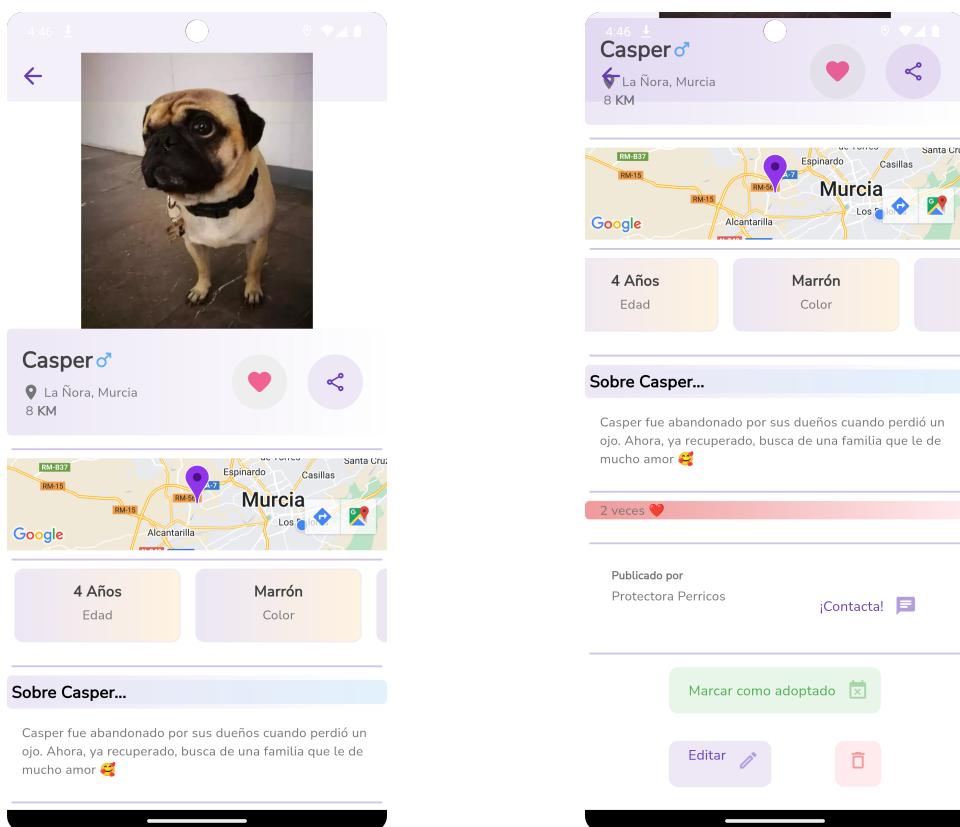


Figura 73: Pantallas de perfil de canino.

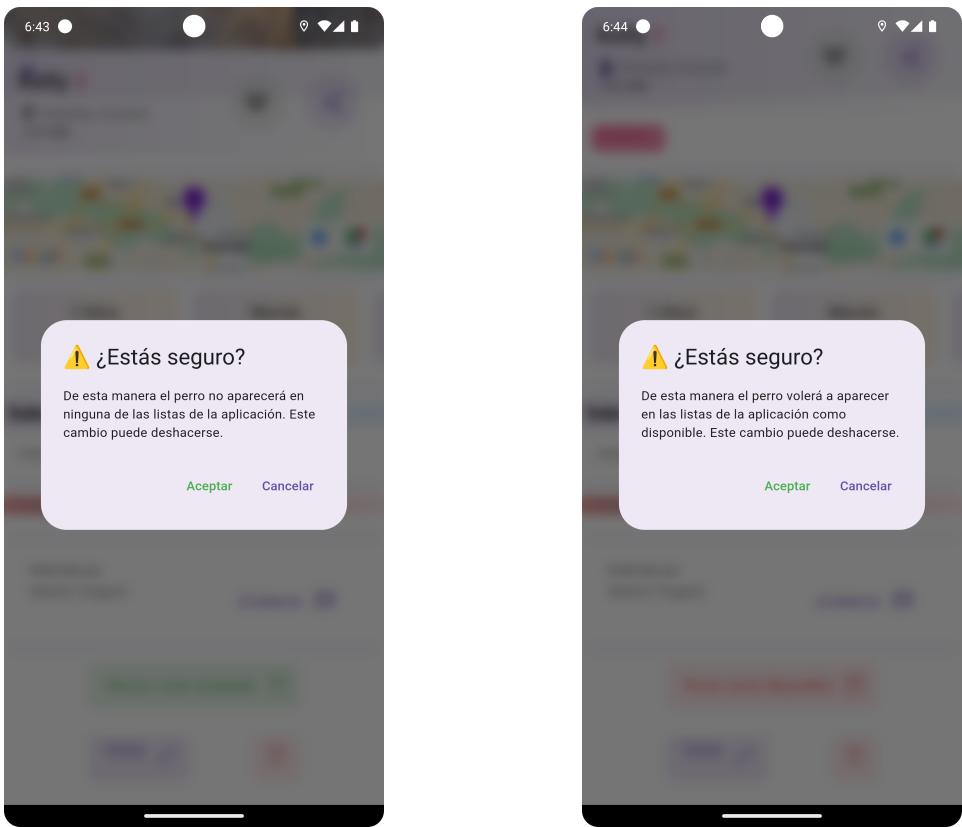


Figura 74: Pantalla que muestra pop-up para marcar/desmarcar como adoptado un canino.

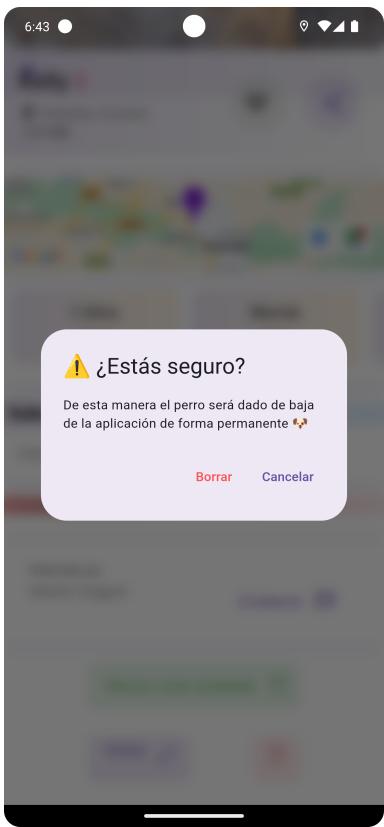


Figura 75: Pantalla para borrar un canino.

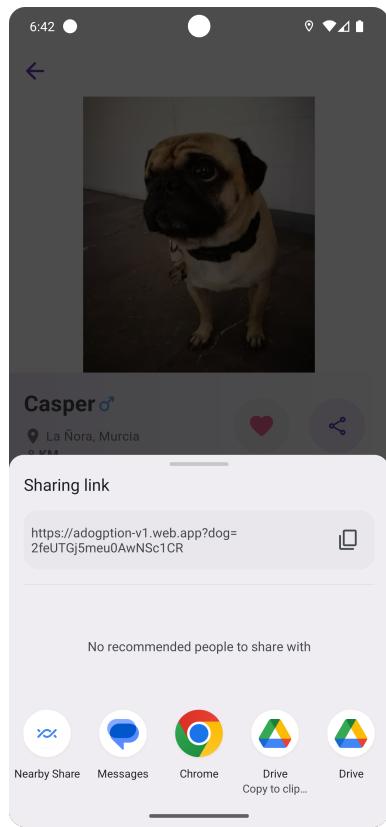


Figura 76: Pantalla para compartir un canino.

Actualización de datos de caninos

Los formularios para actualizar los datos de los caninos son con muy similares a los proporcionados para darlos de alta. Inicialmente, los formularios cargan todos los datos actuales del canino y permiten que se introduzcan todos los cambios que se requieran, además éstos no permiten que un canino se quede sin foto de perfil. Tras confirmar los cambios, si la actualización es correcta, se muestra una notificación indicándolo.

The figure consists of two side-by-side screenshots of a mobile application interface for updating dog data.
Left Screenshot (6:16): Shows the 'Datos del canino' (Dog Data) section. It includes fields for Nombre (Pedro), Raza (Akita), Color (Negro), Edad (4 Años), Peso (13 Kg), and Género (Macho). There is also a circular profile picture of a black dog. A purple edit icon is located at the bottom right of this section.
Right Screenshot (6:17): Shows the 'Confirmación' (Confirmation) section. It includes a dropdown for Género (Macho), a note about listing requirements, checkboxes for 'Para adoptar' (checked) and 'Para acoger' (unchecked), a checkbox for 'Castración hecha' (checked), and a Descripción field containing the text '¡Es un perro muy animado!'. At the bottom is a large blue 'Confirmar' (Confirm) button.

Figura 77: Pantalla de actualización de caninos.

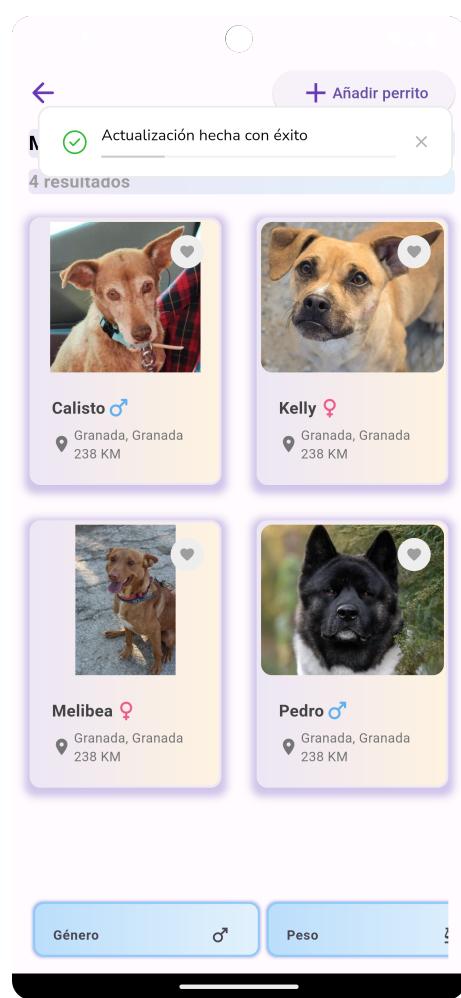


Figura 78: Pantalla con notificación de actualización de canino correcta.

Chats

La estructura de la pantalla de los chats es muy similar a cualquier otra aplicación de mensajería, es decir, se incluye una lista de chats que permiten ser borrados deslizando el widget. Cuando se hace click en alguno de estos widgets, se redirige a la pantalla del chat específico.

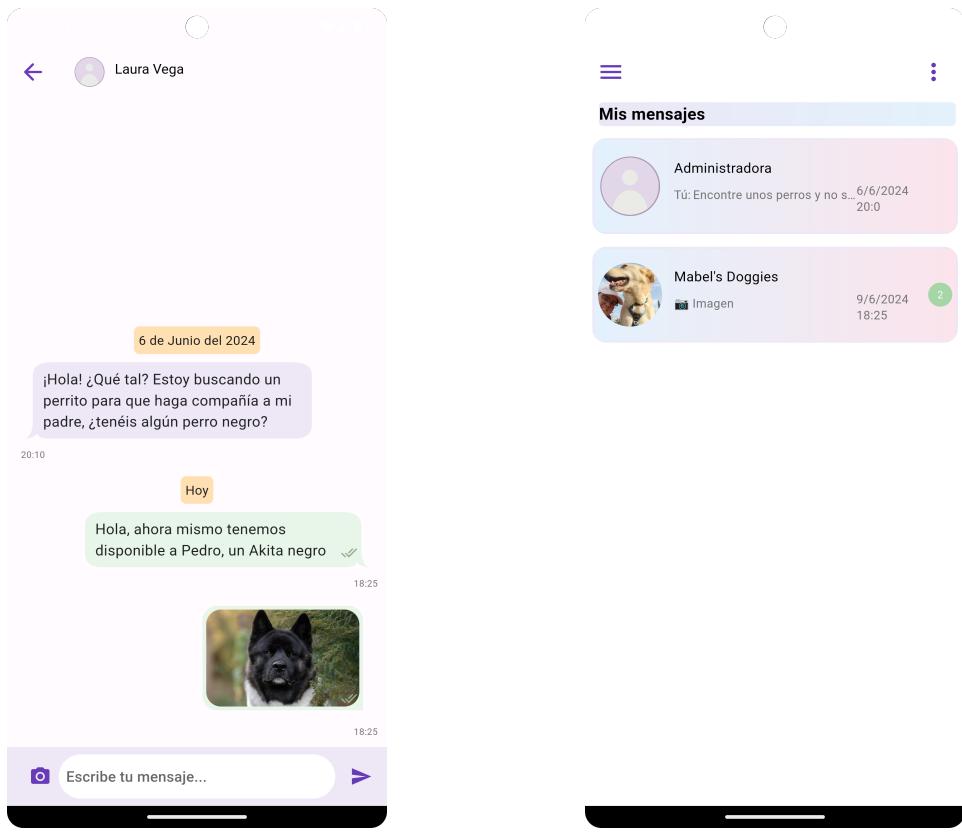


Figura 79: Pantallas del servicio de mensajería.

Información legal

La pantalla de información legal recoge los términos y condiciones de uso además de toda la información de política de privacidad de la aplicación.

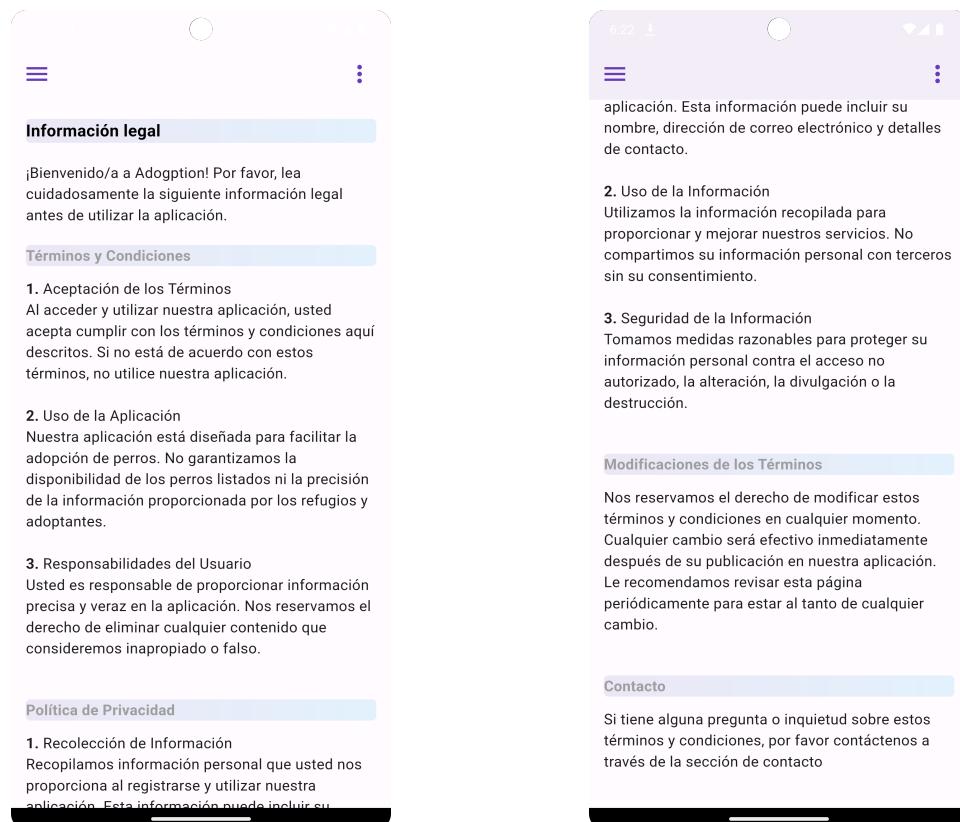


Figura 80: Pantalla de información legal.

Contacto

En la pantalla de contacto se incluye un correo electrónico de contacto y botón que abre un chat con un administrador de la aplicación.

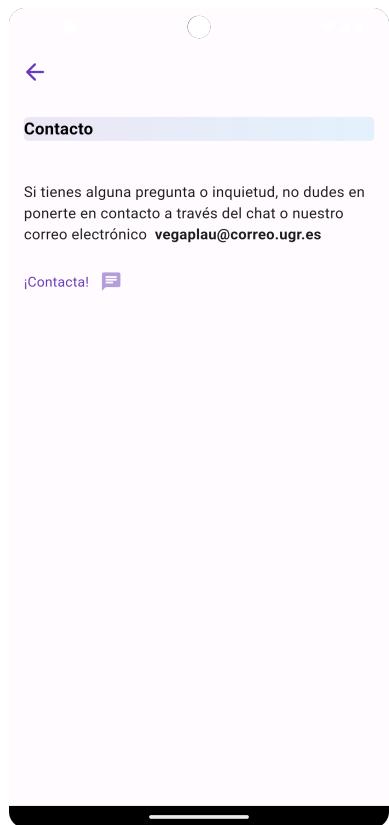


Figura 81: Pantalla de información de contacto.

Notificaciones

Las notificaciones se lanzan cuando un usuario recibe un mensaje de otro usuario, tal y como se muestra en la siguiente figura:

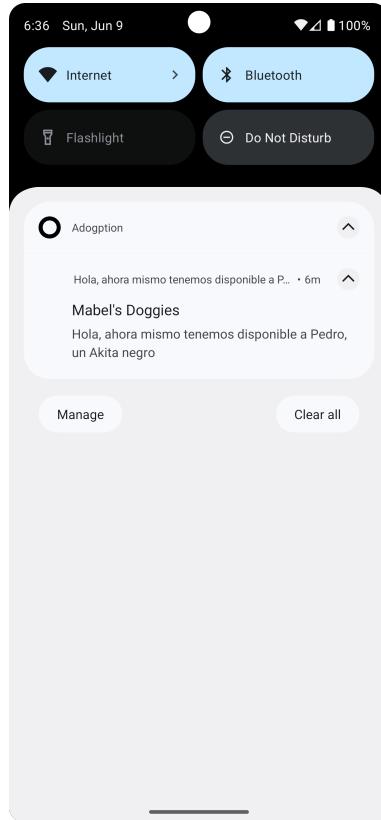


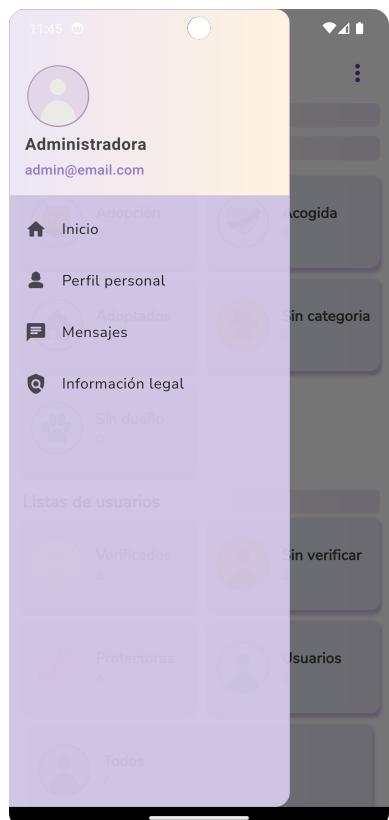
Figura 82: Pantalla de notificación.

Vista de administrador

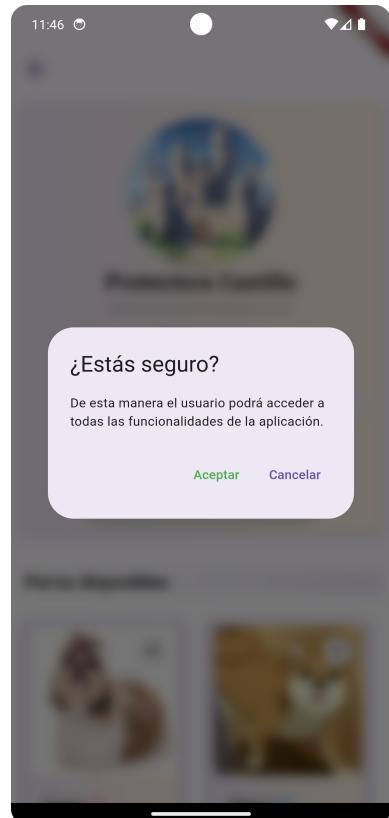
La vista de administrador es la más simple ya que contiene diferentes listas para manejar los usuarios y caninos que están dados de alta en la aplicación. Puede editar datos de cualquier usuario o canino además de verificar y desverificar perfiles de usuario.



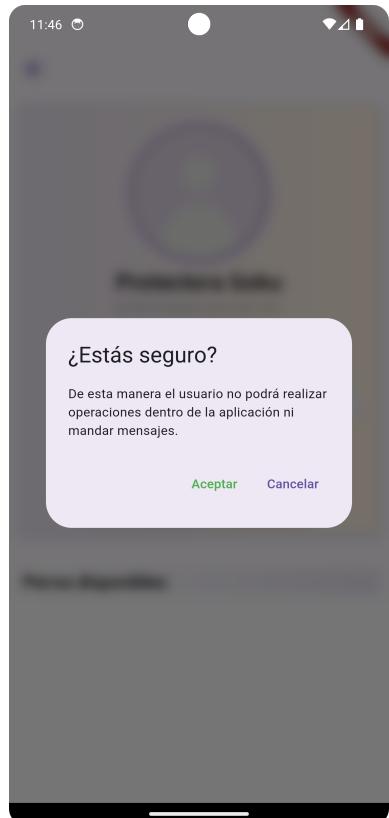
(a) Pantalla de inicio.



(b) Menú lateral.



(c) Pop-up confirmación.



(d) Pop-up desconfirmación.

108

Figura 83: Vista de administrador.

8 Conclusiones

En este TFG se ha desarrollado una aplicación de adopción canina que permite a personas que quieren adoptar o acoger algún canino encontrar y contactar de forma más eficaz con las protectoras disponibles en su zona. Se ha implementado el registro de usuarios por rol, además de proporcionar diferentes vistas para cada uno de los roles, que se han definido como *Usuario*, *Protectora* y *Administrador*. A lo largo de la aplicación se han incluido diferentes listas de usuarios, protectoras y caninos con barras de búsqueda y filtros para acotar resultados. Para proporcionar una mejor visibilidad de las protectoras que hay en un área, se ha desarrollado una pantalla que contiene un mapa con todas las protectoras cercanas. Se ha implementado un chat en tiempo real para proporcionar contacto directo entre usuarios y protectoras, además de permitir a estos contactar con un administrador de la aplicación. Toda la información legal y de contacto se ha recogido en una pantalla a disposición de cualquier rol de la aplicación. Para finalizar, se han implementado las funcionalidades de compartir y marcar como favoritos los perfiles de los caninos disponibles.

Al revisar la lista de objetivos, se concluye que la aplicación cumple con todas las funcionalidades obligatorias. Sin embargo, algunos requisitosopcionales no se han completado en esta versión. Específicamente, el blog y el tema oscuro. El blog resultó ser demasiado ambicioso y no se ha podido desarrollar en el tiempo estipulado, mientras que el tema oscuro se pospuso debido a retrasos en la implementación de otras funcionalidades obligatorias. Estos requisitos pendientes se considerarán para futuras versiones de la aplicación. Por tanto se han planteado algunas propuestas para añadir valor a la aplicación y mejorar la experiencia de uso del usuario:

- Añadir la posibilidad de crear listas con filtros a las protectoras.
- Permitir subir múltiples imágenes para los caninos dados de alta, además de vídeos y poder generar un álbum para cada canino.
- Permitir asignar a un usuario como adoptante/acogedor de un canino, además de indicar en el perfil del canino quién es su adoptante/acogedor.
- Añadir la funcionalidad de mandar mensajes de audios y vídeos dentro del chat.
- Añadir como atributos opcionales de las protectoras sus redes sociales e incluso el número de teléfono.
- Permitir compartir perfiles de protectoras.
- Integrar los servicios de Google para iniciar sesión.
- Incluir más filtros en las listas de caninos.

Todas estas propuestas buscan mejorar la experiencia de uso de la aplicación.

Algunas de ellas surgen a partir del *feedback* proporcionado por los usuarios que han probado la aplicación. Además, aunque la aplicación está diseñada inicialmente para facilitar las adopciones de caninos, se contempla la posibilidad de incluir otros tipos de animales, como gatos, en futuras actualizaciones.

9 Bibliografía

Referencias

- [1] National Geographic. (2021) España, líder europea en abandono de animales: 700 cada día. [Online]. Available: <https://www.nationalgeographic.es/animales/2021/12/espana-lider-europea-en-abandono-de-animales-700-cada-dia>
- [2] F. Affinity. (2023) Las cifras del abandono de perros y gatos. [Online]. Available: <https://www.fundacion-affinity.org/perros-gatos-y-personas/busco-un-animal-de-compania/las-cifras-del-abandono-de-perros-y-gatos-aun>
- [3] National Geographic. (2017) Tráfico ilegal de mascotas en Europa. [Online]. Available: <https://www.nationalgeographic.es/animales/2017/11/el-trafico-ilegal-de-cachorros-en-la-ue-una-triste-historia-perruna>
- [4] BOE. (2023) Ley de protección de los animales. [Online]. Available: <https://www.boe.es/buscar/doc.php?id=BOE-A-2023-7936>
- [5] P. Latino. (2024) Animales que sufren por el comercio de mascotas. [Online]. Available: <https://investigaciones.petlatino.com/animales-sufren-comercio-mascotas/>
- [6] ADDA. Listado de protectoras de animales en España. [Online]. Available: <https://www.addaong.org/es/informacion/listado-de-protectoras-de-animales/>
- [7] Firebase. (2024) Precios de Firebase. [Online]. Available: <https://firebase.google.com/pricing?hl=es-419>
- [8] Google. (2024) Precios de APIs de Google. [Online]. Available: <https://mapsplatform.google.com/intl/es/pricing/>
- [9] La Vanguardia. (2022) Mejores aplicaciones para adoptar gratuitas. [Online]. Available: <https://www.lavanguardia.com/andro4all/aplicaciones/apps-gratuitas-para-adoptar-perros-y-gatos>
- [10] Miwuki. Miwuki Pet Shelter. [Online]. Available: <https://www.miwuki.com/>
- [11] Looker Studio, Miwuki. Abandonos en España por Miwuki. [Online]. Available: <https://lookerstudio.google.com/u/0/reporting/bfdde81-1a48-4370-8616-e7b527240a82/page/axT5>

- [12] KLYGO. KLYGO - Adopta perros y gatos. [Online]. Available: <https://play.google.com/store/apps/details?id=com.klygo.app&hl=ca&pli=1>
- [13] Amazdog Universal, S.L. (2024) Amazdog app. [Online]. Available: <https://play.google.com/store/apps/details?id=com.amazdog.app>
- [14] Miro. Página web de Miro. [Online]. Available: <https://miro.com/es/>
- [15] PlantUML. PlantUML documentación. [Online]. Available: <https://plantuml.com/es/>
- [16] React Native. React Native framework. [Online]. Available: <https://reactnative.dev/>
- [17] Flutter. (2024) Documentación de Flutter. [Online]. Available: <https://flutter.dev>
- [18] Kotlin. Kotlin Native Multiplataform documentación. [Online]. Available: <https://kotlinlang.org/docs/multiplatform.html>
- [19] Google. Android Studio Hedgehog — 2023.1.1 (noviembre de 2023). [Online]. Available: <https://developer.android.com/studio/releases/past-releases/as-hedgehog-release-notes?hl=es-419>
- [20] Dart. (2024) Documentación de Dart. [Online]. Available: <https://flutter.dev>
- [21] Firebase. (2024) Consola de Firebase. [Online]. Available: <https://console.firebaseio.google.com>