APLICACIÓN MÓVIL PARA LA GEOLOCALIZACION DE MASCOTAS HACIENDO USO DE LA TECNOLOGÍA DE CÓDIGO QR.

INTEGRANTES:

JIMENA ASTRID ROJAS DUARTE
JUAN SEBASTIAN BUITRAGO ROMERO



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLÓGICA
SISTEMATIZACIÓN DE DATOS
BOGOTA D.C
2018

APLICACIÓN MÓVIL PARA LA GEOLOCALIZACION DE MASCOTAS HACIENDO USO DE LA TECNOLOGÍA DE CÓDIGO QR.

INTEGRANTES:

JIMENA ASTRID ROJAS DUARTE
CODIGO: 20151578089

JUAN SEBASTIAN BUITRAGO ROMERO
CODIGO: 20142578141

TUTOR:
INGENIERA: ROCIO RODRIGUEZ GUERRERO

PROYECTO DE GRADO PARA OPTAR POR EL TÍTULO DE TECNÓLOGO EN SISTEMATIZACIÓN DE DATOS

TIPO DE PROYECTO

MONOGRAFIA

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLÓGICA
SISTEMATIZACIÓN DE DATOS
BOGOTA D.C
2018

Nota de Aceptación

_____
_____
_____
_____
_____
_____

_____
Firma del Tutor

_____
Firma del Jurado

_____
Firma del Jurado

Bogotá 2 de abril de 2018

AGRADECIMIENTOS

Agradecemos a Dios por permitirnos llegar hasta aquí, a nuestra tutora Roció Rodríguez y los demás docentes que a lo largo de nuestra carrera profesional nos han brindado sus conocimientos, ideas y dirección.

A nuestra familia, compañeros y amigos que hicieron parte de nuestro proceso educativo, brindándonos su apoyo y su compañía.

Muchas Gracias, por permanecer a nuestro lado.

TABLA DE CONTENIDO

INTRODUCCIÓN 11
RESUMEN 12
1. FASE 1 - PLANEACIÓN, DEFINICIÓN Y ORGANIZACIÓN 14
1.1. TITULO 14
1.2. TEMA 14
1.3. PLANTEAMIENTO DEL PROBLEMA 14
1.3.1. Descripción del problema 14
1.3.2. Formulación 14
1.4. ALCANCES Y LIMITACIONES 15
1.4.1. Alcances 15
1.4.2. Delimitaciones 15
1.5. OBJETIVOS 15
1.5.1. Objetivo General 15
1.5.2. Objetivos Específicos 15
1.6. JUSTIFICACION 16
1.7. MARCO REFERENCIAL 17
1.7.1. Marco Teórico 17
1.7.2. Marco Conceptual 19
1.8. FACTIBILIDAD DE DESARROLLO 26
1.8.1. Factibilidad Económica 26
1.8.2. Factibilidad Legal 27
1.8.3. Factibilidad Técnica 27
1.8.4. Factibilidad Operativa 28
1.9. CRONOGRAMA DE ACTIVIDADES 28
2. FASE 2 - ANALISIS DEL PROYECTO 29
2.1. Identificación de los roles 29
2.2. Lista de actividades por rol 29
2.3. Backlog 31
2.4. Sprints 33
3. FASE 3 – DISEÑO DE LOS SPRINT 34
3.1. Sprint 1 Registro de Usuario 34
3.1.1. Definición de Sprint 34

3.1.2. Tareas del Sprint 1 34
3.1.3. Diseño del sprint 1 Registro de Usuario 35
3.1.4. Proceso de desarrollo del software Sprint 1 Registro de Usuarios ... 36
3.1.5. Diagrama de contexto Sprint 1 37
3.1.6. Pruebas del Sprint 1 Registro de Usuarios 37
3.2. Sprint 2 Login 38
3.2.1. Definición de Sprint 2 38
3.2.2. Tareas Sprint 2 Login 38
3.2.3. Diseño de Sprint 2 Login 39
3.2.4. Proceso de desarrollo de software Sprint 2 Login 40
3.2.5. Diagrama de Contexto Sprint 2 40
3.2.6. Prueba de Sprint 2 41
3.3. Sprint 3 Registrar Mascota 42
3.3.1. Definicion de Sprint 3 Registrar mascota 42
3.3.2. Tareas de Sprint 3 42
3.3.3. Diseño de Sprint 3 Registrar Mascota 43
3.3.4. Proceso de desarrollo de software Sprint 3 Registrar mascota 45
3.3.5. Diagrama de contexto Sprint 3 51
3.3.6. Prueba de Sprint 3 51
3.4. Sprint 4 Seleccionar mascota 52
3.4.1. Definición de Sprint 4 Seleccionar Mascota 52
3.4.2. Tareas de Sprint 4 52
3.4.3. Diseño de Sprint 4 Seleccionar mascota 53
3.4.4. Proceso de desarrollo de software Sprint 4 Seleccionar Mascota 53
3.4.5. Diagrama de contexto Sprint 4 54
3.4.6. Prueba de Sprint 4 54
3.5. Definición de Sprint 5 Agregar Eventos 55
3.5.1. Tareas de Sprint 5 55
3.5.2. Diseño de Sprint 5 Agregar Evento 56
3.5.3. Proceso de desarrollo de software Sprint 5 Agregar Evento 57
3.5.4. Diagrama de contexto Sprint 5 59
3.5.5. Prueba de Sprint 5 60
3.6. Definición de Sprint 6 Generar Código QR 60
3.6.1. Tareas de Sprint 6 61

3.6.2. Diseño de Sprint 6 Generar Codigo QR 61
3.6.3. Proceso de desarrollo de software Sprint 6 Generar Código QR 62
3.6.4. Diagrama de contexto Sprint 6 64
3.6.5. Prueba de Sprint 6 64
3.7. Definición de Sprint 7 Buscar mascota 65
3.7.1. Tareas de Sprint 7 66
3.7.2. Diseño de Sprint 7 Buscar mascota 66
3.7.3. Proceso de desarrollo de software Sprint 7 Buscar Mascota 69
3.7.4. Diagrama de contexto Sprint 7 73
3.7.5. Prueba de Sprint 7 73
3.8. Definición de Sprint 8 Base de datos 74
3.8.1. Tareas de Sprint 8 74
3.8.2. Diseño de Sprint 8 Base de datos 76
3.8.3. Prueba de Sprint 8 77
3.9. Definición de Sprint 9 Información 77
3.9.1. Tareas de Sprint 9 78
3.9.2. Diseño de Sprint 9 Información 78
3.9.3. Proceso de desarrollo de software Sprint 9 Información 81
3.9.4. Diagrama de contexto Sprint 9 82
3.9.5. Prueba de Sprint 9 83
3.10. Definición de Sprint 10 Menú Principal 83
3.10.1. Tareas de Sprint 10 84
3.10.2. Diseño de Sprint 10 Menú principal 84
3.10.3. Proceso de desarrollo de software Sprint 10 Menú Principal 85
3.10.4. Diagrama de contexto Sprint 10 87
3.10.5. Prueba de Sprint 10 87
4. FASE 4 – IMPLEMENTACION 88
4.1. Diagrama de paquetes 88
4.2. Diagrama de componentes 88
4.3. Pruebas en dispositivos 89
CONCLUSIONES 91
RECOMENDACIONES 92
BIBLOGRAFIA 93
Anexos 94

LISTA DE TABLAS

Tabla 1 factibilidad Económica – software y hardware 26
Tabla 2 factibilidad Económica – Recursos Humanos 26
Tabla 3 factibilidad Económica – Total Costos 27
Tabla 4 factibilidad Técnica 27
Tabla 5 factibilidad Operativa 28
Tabla 6Roles 29
Tabla 7 Lista de actividades 29
Tabla 8 Backlog 31
Tabla 9 Sprints 33
Tabla 10 Definición Sprint 1 34
Tabla 11Tareas Backlock Sprint 1 34
Tabla 12 Pruebas Backlock Sprint 1 37
Tabla 13 Definición Sprint 2 38
Tabla 14 Tareas Backlock Sprint 2 38
Tabla 15 Pruebas Backlock Sprint 2 41
Tabla 16 Definición Sprint 3 42
Tabla 17 Tareas Backlock Sprint 3 42
Tabla 18 Pruebas Backlock Sprint 3 51
Tabla 19 Definición Sprint 4 52
Tabla 20 Tareas Backlock Sprint 4 52
Tabla 21 Pruebas Backlock Sprint 4 54
Tabla 22 Definición Sprint 5 55
Tabla 23 Tareas Backlock Sprint 5 55
Tabla 24 Prueba Backlock Sprint 5 60
Tabla 25 Definicion Sprint 6 60
Tabla 26 Tareas Backlock Sprint 6 61
Tabla 27 Prueba Backlock Sprint 6 65
Tabla 28 Definición Sprint 7 65
Tabla 29 Tareas Backlock Sprint 7 66
Tabla 30 Pruebas Backlock Sprint 7 73
Tabla 31 Definición Sprint 8 74
Tabla 32 Tareas Backlock Sprint 8 74
Tabla 33 Pruebas Backlock Sprint 8 77
Tabla 34 Definición Sprint 8 77
Tabla 35 Tareas Backlock Sprint 9 78
Tabla 36 Pruebas Backlock Sprint 9 83
Tabla 37 Definición Sprint 10 83
Tabla 38 Tareas Backlock Sprint 10 84
Tabla 39 Pruebas Backlock Sprint 10 87
Tabla 40 Dispositivo usuario 1 89
Tabla 41Dispositivo usuario 2 89
Tabla 42 Dispositivo usuario 3 90

LISTA DE ILUSTRACIONES

Ilustración 1 Flujo de SCRUM 25
Ilustración 2 Cronograma de actividades 28
Ilustración 3 Sprint Registro de Usuario 35
Ilustración 4 Librería Del Motor de bases de datos 36
Ilustración 5 Almacenamiento de Datos 36
Ilustración 6 Diagrama de contexto Sprint 1 37
Ilustración 7 Sprint Login 39
Ilustración 8 Consulta y verificación de datos de Usuario 40
Ilustración 9 Diagrama de contexto Sprint 2 40
Ilustración 10 Dashboard Back4App, tabla de usuarios registrados 41
Ilustración 11 Sprint Registro mascotas 44
Ilustración 12 Sprint Crear Perfil 44
Ilustración 13 Validación de campos de texto 45
Ilustración 14 Captura de datos por medio de los campos de texto 46
Ilustración 15 Validación de que la mascota no exista en la base de datos 46
Ilustración 16 Generar la dirección de la mascota 47
Ilustración 17 Arreglos de imágenes 48
Ilustración 18 Método que recorre el arreglo de imágenes para Caninos 49
Ilustración 19 Método que recorre el arreglo imágenes para Felinos 49
Ilustración 20 Permisos para uso de cámara del dispositivo 50
Ilustración 21 Opciones para seleccionar imagen de perfil 50
Ilustración 22 Diagrama de contexto Sprint 3 51
Ilustración 23 Sprint Seleccionar mascota 53
Ilustración 24 Opciones para seleccionar imagen de perfil 54
Ilustración 25 Diagrama de Contexto Sprint 4 54
Ilustración 26 Sprint Agregar Evento 1 56
Ilustración 27 Sprint Agregar Evento 2 57
Ilustración 28 Registro de eventos 58
Ilustración 29 Consulta de los próximos eventos 59
Ilustración 30 Diagrama de contexto Sprint 5 59
Ilustración 31 Sprint Generar código QR 62
Ilustración 32 Consulta de datos de la mascota y el usuario 63
Ilustración 33 Construcción de código QR por medio de los datos de la mascota y el usuario 63
Ilustración 34 Método que permite descargar el código QR generado 64
Ilustración 35 Diagrama de contexto Sprint 6 64
Ilustración 36 Sprint Buscar Mascota 1 67
Ilustración 37 Sprint Buscar Mascota 2 67
Ilustración 38 Sprint Buscar Mascota 3 68
Ilustración 39 Sprint Buscar Mascota 4 69
Ilustración 40 Lectura del código de barras 69

Ilustración 41 Método que genera la ubicación de la mascota y trae los datos de contacto 70
Ilustración 42 Método que permite decidir si desea contactarse con el dueño de una mascota perdida 70
Ilustración 43 Método que permite contactarse con el dueño de una mascota perdida 71
Ilustración 44 Código que genera los puntos en el mapa 72
Ilustración 45 Diagrama de contexto Sprint 7 73
Ilustración 46 Diagrama Relacional base de datos 76
Ilustración 47 Sprint Información 1 79
Ilustración 48 Sprint Información 2 79
Ilustración 49 Sprint Información 3 80
Ilustración 50 Sprint Información 4 80
Ilustración 51 Sprint Información 5 81
Ilustración 52 Orden de llamado de cada Fragmento 82
Ilustración 53 Diagrama de contexto Sprint 9 82
Ilustración 54 Sprint menú Principal 85
Ilustración 55 Menú inferior circular 86
Ilustración 56 Menú superior 86
Ilustración 57 Diagrama de contexto Sprint 10 87
Ilustración 58 Diagrama de paquetes de la aplicación 88
Ilustración 59 Diagrama de componentes de la aplicación 88

INTRODUCCIÓN

La tecnología hoy en día avanza de manera exponencial, tenemos el privilegio de percibir un incremento por parte de los usuarios al hacer indispensable y asimismo hacer uso de las aplicaciones móviles; considerando que estas deben estar al alcance de todos y con el auge de las redes sociales en los últimos tiempos, se piensa en una solución tecnológica para incluir mascotas como usuarios en las famosas apps.

El proyecto tiene el fin de culminar con una aplicación móvil para geo localizar una mascota haciendo el uso de un código QR que tendrá la información de una persona y de su mascota, teniendo, en cuenta lo común que es extraviarlas y la necesidad que con lleva encontrarla, además de tener una agenda de mascota digital para recordar al dueño los controles básicos para la mascota ya que en la mayoría de las ocasiones no se tienen un buen seguimiento médico, todo esto haciendo uso de una conexión a internet.

El presente documento da a conocer el planteamiento de la problemática que se desea abordar; igualmente la solución que se propone para mostrar la temática en lo que al problema concierne, se realizó una encuesta para explicar mejor la problemática al final se anexan los resultados, la información fue recolectada en base a la primera fase del proyecto que es la planeación en donde se plantearon los objetivos, los alcances y las limitaciones del proyecto; la segunda la fase del proyecto, análisis de proyecto donde se identificaron, los roles de trabajo, los requerimientos y los ciclos de trabajo que se realizaron; la tercera fase del proyecto, se desarrollaron los ciclos anteriormente identificados; y por último la cuarta fase del proyecto la implementación y unificación de cada uno de los módulos del proyecto.

RESUMEN

El proyecto “Aplicación móvil para la geolocalización de mascotas haciendo uso de la tecnología de código QR.” tiene como fin desarrollar un aplicativo móvil para la geolocalización de mascotas y el recordatorio de eventos por medio del registro de una mascota con la finalidad de recordar a los dueños de las fechas más importantes para las vacunas u otros eventos.

Esta aplicación estará disponible para todo tipo usuario, su diseño es dinámico y de fácil mantenimiento, permitirá al usuario manejarlo sin la necesidad de tener un gran conocimiento sobre el dispositivo móvil, con funciones sencillas como agregar, eliminar, editar y consultar información, enfatizado en la geolocalización de mascotas.

El aplicativo contara con módulos: módulo de perfil de la mascota, un módulo de acontecimientos, módulo de agenda de mascota y módulo de geolocalización desarrollados bajo la tecnología JAVA, con ayuda del entorno de desarrollo ANDROID STUDIO y APIS del sistema operativo Android y un motor de bases de datos como lo es Back4app.

ABSTRACT

The project "Mobile application for the geolocation of pets making use of QR code technology." Aims to develop a mobile application for the geolocation of pets and the reminder of events through the registration of a pet in order to remind owners of the most important dates for vaccines or other events.

This application will be available for all types of users, its design is dynamic and easy to maintain, it will allow the user to manage it without the need to have a great knowledge about the mobile device, with simple functions such as adding, deleting, editing and consulting information, emphasized in the geolocation of pets.

The application will have modules: pet profile module, an events module, pet calendar module and geolocation module developed under JAVA technology, with the help of the ANDROID STUDIO and APIS development environment of the Android operating system and an engine of databases such as Back4app.

1. FASE 1 - PLANEACIÓN, DEFINICIÓN Y ORGANIZACIÓN

1.1. TITULO

APLICACIÓN MOVIL PARA LA GEOLOCALIZACION DE MASCOTAS HACIENDO USO DE LA TECNOLOGIA DE CODIGO QR

1.2. TEMA

Aplicación móvil para la geolocalización de mascotas, e integración de una agenda de mascota que implementa tecnología de código QR y redes sociales.

1.3. PLANTEAMIENTO DEL PROBLEMA

1.3.1. Descripción del problema

Es habitual notar que cuando una mascota se extravía lo primero que va a hacer su dueño será publicar a través de varios medios de comunicación su perdida, para esta premisa se realiza una encuesta para entender mejor la problemática que se quiere abordar, se encuestaron 164 personas sobre la temática que gira alrededor de la perdida de mascotas los resultados arrojaron que un 51.2% afirma que nunca han perdido una mascota, mientras que el 48.8% restante afirma haber extraviado la mascota.

Con respecto a los medios de comunicación, la encuesta señalo que el 63% de las personas prefieren buscar a su mascota a través de medios como redes sociales; otro 49.1% prefiere publicar carteles y un 13.9% informar a la autoridad.

Otro ámbito considerable es que, con los afanes de la vida cotidiana de un usuario común, es habitual que este olvide aspectos significativos para una vida sana de su mascota, ya sea el historial de las vacunas aplicadas, la desparasitación o bien sea algo tan sencillo como la comida favorita, o algo tan importante como la ficha veterinaria.

No ajeno a esta hipótesis en algunos casos es necesario para el dueño encontrar a última hora un veterinario, paseadores o guarderías, suplir las necesidades básicas que se le deben suministrar a una mascota para una vida larga y saludable se convierte en un reto con el diario vivir.

1.3.2. Formulación

¿El desarrollo de una aplicación móvil para la geolocalización de mascotas extraviadas a partir de tecnología QR, facilitara su búsqueda y posterior recuperación?

1.4. ALCANCES Y LIMITACIONES

1.4.1. Alcances

- El presente proyecto empleara lógica de programación y tanto creación como administración de objetos SQL, la cual se ha adquirido durante toda nuestra carrera en tecnología en sistematización de datos.
- Se estima un tiempo aproximado de 6 meses para la culminación de todo respecto al código para desarrollar la aplicación.
- En un principio se evalúa hacer pruebas piloto para mascotas de tipo canino dentro de la ciudad de Bogotá.

1.4.2. Delimitaciones

- Encontrar un prestador de servicios web de código QR amoldado a nuestras necesidades.
- El periodo de tiempo necesario para recolección de información para el desarrollo de la aplicación se requiere aproximadamente 6 meses.
- El proyecto se espera abarque la ciudad de Bogotá en un comienzo.
- La aplicación inicialmente será desarrollada para plataformas Android.

1.5. OBJETIVOS

1.5.1. Objetivo General

- Desarrollar una aplicación móvil que permita la geolocalización de mascotas implementando la tecnología código QR y usando el acceso de internet.

1.5.2. Objetivos Específicos

- Desarrollar un módulo de registro de mascotas para los usuarios a partir de un código QR único.
- Implementar la tecnología de geolocalización de google mediante el uso de una API para el sistema operativo de ANDROID
- Desarrollar una base de datos de almacenamiento y consulta que permita guardar la posición instantánea de la mascota, los datos de su dueño y el registro de eventos en la nube.
- Desarrollar una agenda personal para el dueño de la mascota que permita recordarle un evento determinado.

1.6. JUSTIFICACION

El aplicativo se realiza con el propósito de suplir una necesidad del mercado aplicando los conocimientos aprendidos durante el tiempo de estudio transcurrido, es por ello que se desea evidenciar estos mismos, pues, debido a su importancia son necesarios para la culminación y desarrollo de una aplicación móvil Android que sea sólida y completa, es decir, que cumpla con los objetivos anteriormente propuestos, con el fin de alimentar la demanda de apps gratuitas y útiles para el mercado; la idea es hacer uso de la tecnología para Smartphone con el fin de ayudar a los hogares y personas que tienen o gustan de mascotas y que se encuentran en la situación de necesitar localizarlos y/o de llevar una agenda con recordatorios de vacunas, desparasitadas, etc.; o bien sea una persona corriente con proyección a adopción de una mascota a futuro.

La necesidad de tener una aplicación móvil que incluya a mascotas se hace cada vez más evidente teniendo en cuenta que tener una mascota implica cuidado y atención por parte del dueño, es decir, mantener su bienestar se vuelve una responsabilidad; es habitual toparse con personas que olvidan bien sea las vacunas o los paseos diarios; sintetizando, suplir las necesidades básicas que se le deben suministrar a una mascota para una vida larga y saludable, esto, se convierte en un reto con el diario vivir; una app que pueda integrar estas necesidades y solucionarlas de manera tecnológica y novedosa podría ser de gran auxilio a los dueños de mascotas que desean llevar una buena agenda que les ayude a recordar las exigencias para una vida plena de la mascota; con el avance tecnológico exponencial cotidiano se tiene la necesidad de instituir aplicaciones móviles con una perspectiva hacia usuarios nuevos en este caso en específico se quiere incluir animales teniendo como cimientos la perdida de los mismos.

Es cierto que en el mercado podemos encontrar aplicación como redes sociales, agendas e incluso localizadores, con lo nombrado anteriormente se desea integrar todas estas funcionalidades, la aplicación estará al alcance de todo tipo de público, esto manifestado desde un plano económico ya que las aplicaciones localizadoras de mascotas que encontramos en el mercado hoy en día como UPet o Back2Gether funcionan como un chip que es implantado en la mascota, estas nos ofrecen servicios que si bien son altamente novedosos y modernos tan bien lo son en sus precios; la aplicación propuesta en el presente trabajo va a tomar estas funcionalidades similares del chip y las va a poner a disposición de todo tipo de público mediante la generación de códigos QR reduciendo costos pero manteniendo por supuesto el concepto tecnológico y moderno; por otro lado las aplicaciones de tipo agenda como lo son cuidapets, se podrían mejorar en la parte de interfaz además desarrollándola integrando elementos importantes como el posicionamiento de la mascota y el módulo de adopción harían una aplicación más completa, a propósito del módulo de adopción Animales sin Hogar es una aplicación

completa web lo que se pretende es traerla a dispositivos Android teniendo en cuenta lo importante que son las apps móviles hoy en día.

1.7. MARCO REFERENCIAL

1.7.1. Marco Teórico

1.7.1.1. Fuentes de información

Fuentes de información primaria

- LAGUNA, Karen. Tecnologa en Sistematizacion de Datos. Aplicativo móvil que permita almacenar datos de mascotas y un asistente medico al instante. Bogota DC. Universidad Distrital Francisco Jose de Caldas. 2016. 115 p.
- LEQUERICA, Joan. Desarrollo de apps para Android. Ed 2016. España: Anaya multimedia. 2015. 432 p. ISBN 978-84-41-5368-07.
- PONSODA, Daniel. Introduccion a SQLite. Ed 2008. España. Creative Commons. 2008. 13 p.

Fuentes de información Secundaria

- FLOREZ, Héctor. Programación Orientada s Objetos. Usando Java. Ed 2012. Bogotá DC: Ecoe Ediciones. 2012. 400 p. ISBN 978-95-86-4879-62.
- FALCON, Marlon. Comunicación en tiempo real. España. 2010. 25 p.
- CHO, James. Interfaz En: Desarrollo De Juegos Para Android. 2015. España.Anaya.2016. 464 p.
- RIBAS, Joan. Desarrollo de aplicaciones para Android. Ed 2017. España: Anaya. 2017. 463 p. ISBN 97-88-44-15-3809-2.

1.7.1.2. Proyectos relacionados

- **Título:** Aplicación Móvil Animales sin Hogar – Colombia
  Autor: Juan Echavarría
  Año: 2010

  Descripción: Animales Sin Hogar nace como un proyecto web que buscaba crear una red para que las personas pudieran encontrar a sus mascotas perdidas y registrar casos de animales abandonados para buscarles un hogar. Hoy, con una novedosa aplicación móvil, esta red opera en varios países y miles de mascotas han encontrado una familia.

La App permite publicar los datos de contacto de la persona que encontró al animal y crear un pequeño perfil con su nombre, teléfono, correo electrónico. Quien recibe el aviso puede comunicarse con esta persona directamente a través de un mensaje de texto, llamada o mail. La aplicación permite además subir fotografías del lugar donde se encontró al animal y georreferenciarlo a través de un mapa.

- **Título:** Aplicación Móvil UPet
  Autor: Carlos Granada Colombia – Medellín
  Año: 2013

  Descripción: UPet es una aplicación en la cual es posible crear un perfil de una mascota perro o gato por el momento en el cual se incluyen datos del animal, datos del propietario, vacunas y desparasitaciones, además de recordatorios de este tipo de eventos los cuales los dueños suelen olvidar muy fácil además de centros veterinarios, tiendas de mascotas y parques. Lo mejor es que todo eso es gratis.

  Cuando el usuario tiene el perfil completo de la mascota, ya hace parte de esta comunidad, lo que le permite al dueño de la mascota reportarla como perdida en cualquier momento. El hecho genera una alerta con foto y detalles del lugar de la pérdida, la cual es enviada a todos los usuarios más cercanos que tienen la aplicación instalada, lo que aumenta las posibilidades de hallar a la mascota.

- **Título:** Aplicación Móvil Back2Gether
  Autor: Empresa Wunderman – Argentina
  Año: 2013

  Descripción: Se llama Back2Gether y permite marcar en un mapa interactivo el lugar exacto donde el perro se perdió o donde se encontró y brinda la información necesaria de la mascota reportada para que pueda ser hallada por su dueño, quien luego se puede contactar por correo electrónico o SMS.
  La aplicación es gratuita y de fácil utilidad. Está dirigida tanto a quienes perdieron su mascota como a quienes encontraron el animal perdido. Ideada por la agencia Wunderman especializada en Marketing digital ya está disponible en español e inglés.

- **Título:** Aplicación Móvil Wizapet
  Autor: Roberto Carlos Pérez Álvarez – España
  Año: 2016

  Descripción: Es la app colaborativa para emitir alertas y avisos de mascotas perdidos o encontrados en cualquier lugar. Todas las personas en la zona en la que se geo localice un animal perdido o

encontrado recibirán una alerta inmediata en su app WIZAPET con la fotografía y características tu mascota.

Todas las mascotas perdidas se geo localizan en el mapa. Todos los animales encontrados se geo localizan en el mapa.

La persona que haya encontrado tu mascota se pondrá en contacto contigo a través del chat interno de WIZAPET. Con WIZAPET puedes ofrecer una recompensa a la persona que encuentre animal de compañía.

- **Título:** Cuidapets
  Autor: Karen Liseth Laguna Barrero – Universidad Distrital Francisco José de Caldas
  Año: 2016

  Descripción: El proyecto tiene como objetivo crear un aplicativo móvil para llevar un control de la mascota y una guía de medicina veterinaria, en este caso para perro o gato, ya que son los animales con mayor demanda en los hogares Colombianos, y que en muchos casos no tienen un buen control médico lo que conlleva a que el bienestar de dicha mascota se deteriore y a su vez afecte el bienestar del ser humano con el que convive, ya que sin tener el debido control, el animal podrá contraer enfermedades zoonoticas las cuales afectan la salud de los dos individuos.

1.7.2. Marco Conceptual

- **Adopción:** es el proceso de tomar la responsabilidad de un animal que un dueño previamente ha abandonado o dejado en un refugio de animales.

- **Agenda digital (PDA):** Estos dispositivos fueron sustituidos por los teléfonos inteligentes que pueden realizar muchas de las funciones que hace una computadora de escritorio (ver películas, crear documentos, juegos casuales, correo electrónico, navegar por Internet, reproducir archivos de audio, etc.) con la ventaja de ser un objeto del que se dispone constantemente.

- **Android studio:** Es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA

- **Base de datos no relacional** es una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico de SGBDR (Sistema de Gestión de Bases de Datos Relacionales) en aspectos

importantes, siendo el más destacado que no usan SQL como lenguaje principal de consultas.

- **Base de datos relacional:** es una colección de elementos de datos organizados en un conjunto de tablas formalmente descritas desde la que se puede acceder a los datos o volver a montarlos de muchas maneras diferentes sin tener que reorganizar las tablas de la base.

- **Codigo Qr:** Es un módulo para almacenar información en una matriz de puntos o en un código de barras bidimensional. La matriz se lee en el dispositivo móvil por un lector específico (lector de QR) y de forma inmediata nos lleva a una aplicación en internet y puede ser un mapa de localización, un correo electrónico, una página web o un perfil en una red social.

- **Ficha veterinaria:** Es un formato que tiene los datos básicos de una mascota, diseñada para el manejo de la información clínica de un animal.

- **Inkscape:** Es un editor de gráficos vectoriales gratuito y de código libre. Inkscape puede crear y editar diagramas, líneas, gráficos, logotipos, e ilustraciones complejas. El formato principal que utiliza el programa es Scalable Vector Graphics (SVG) versión 1.1.

- **JDK:** Es un software que provee herramientas de desarrollo para la creación de programas en Java. Puede instalarse en una computadora local o en una unidad de red.

- **Sistema de posicionamiento global:** es un sistema que permite determinar en toda la Tierra la posición de un objeto (una persona, un vehículo) con una precisión de hasta centímetros (si se utiliza GPS diferencial), aunque lo habitual son unos pocos metros de precisión

- **SVG:** SVG (acrónimo de "Scalable Vector Graphics") es un formato de archivo libre y abierto, estandarizado para gráficos vectoriales. Es mantenido y desarrollado por el W3C (World Wide Web Consortium), la principal organización internacional de estándares para la web.

- **Vacuna:** Una vacuna contiene típicamente un agente que se asemeja a un microorganismo causante de la enfermedad y a menudo se hace a partir de formas debilitadas o muertas del microbio, sus toxinas o una de sus proteínas de superficie.

1.7.2.1. Sistema de posicionamiento global

El Sistema de Posicionamiento Global (GPS) es un sistema de radionavegación de los Estados Unidos de América, basado en el espacio, que proporciona servicios fiables de posicionamiento, navegación, y cronometría gratuita e ininterrumpidamente a usuarios civiles en todo el mundo. A todo el que cuente con un receptor del GPS, el sistema le proporcionará su localización y la hora exacta en cualesquiera condiciones atmosféricas, de día o de noche, en cualquier lugar del mundo y sin límite al número de usuarios simultáneos. El GPS se compone de tres elementos: los satélites en órbita alrededor de la Tierra, las estaciones terrestres de seguimiento y control, y los receptores del GPS propiedad de los usuarios. Desde el espacio, los satélites del GPS transmiten señales que reciben e identifican los receptores del GPS; ellos, a su vez, proporcionan por separado sus coordenadas tridimensionales de latitud, longitud y altitud, así como la hora local precisa.

Hoy están al alcance de todos en el mercado los pequeños receptores del GPS portátiles. Con esos receptores, el usuario puede determinar con exactitud su ubicación y desplazarse fácilmente al lugar a donde desea trasladarse, ya sea andando, conduciendo, volando o navegando. El GPS es indispensable en todos los sistemas de transporte del mundo ya que sirve de apoyo a la navegación aérea, terrestre y marítima. Los servicios de emergencia y socorro en casos de desastre dependen del GPS para la localización y coordinación horaria de misiones para salvar vidas. Actividades cotidianas como operaciones bancarias, de telefonía móvil e incluso de las redes de distribución eléctrica, ganan en eficiencia gracias a de la exactitud cronométrica que proporciona el GPS. Agricultores, topógrafos, geólogos e innumerables usuarios trabajan de forma más eficiente, segura, económica y precisa gracias a las señales accesibles y gratuitas del GPS.

Ejemplos de Aplicaciones que hacen uso del sistema posicionamiento global
Waze Social GPS Maps & Traffic: Es una herramienta gratuita disponible en la Android Market. En su última versión presenta grandes mejoras en su navegabilidad con una sola mano. Los iconos que te van mostrando a tiempo real la velocidad y dirección actual son ahora más grandes.

Navigator: Permite la navegación offline sin consumir tarifa de internet ni una cantidad elevada de batería, descargando los mapas previamente a la tarjeta microSD del smartphone. A esto agrega que su manejo resulta muy fácil.

BlackBerry Maps: Herramienta de navegación y geolocalización, que brinda apoyo para ubicar con rapidez los sitios de interés que ellos buscan. Cuenta con mapas interactivos.

1.7.2.2. Código QR

Un código QR es un código de barras bidimensional cuadrada que puede almacenar los datos codificados. La mayoría del tiempo los datos es un enlace

a un sitio web (URL). Al escanear un código QR utilizando el teléfono inteligente, se obtiene un acceso inmediato a su contenido. El lector de código QR a continuación, puede realizar una acción, como abrir el navegador web para una URL específica. Otras acciones pueden ser provocadas, como el almacenamiento de una tarjeta de visita en la lista de contactos de su teléfono inteligente o conectarse a una red inalámbrica.

Ejemplos de Aplicaciones que hacen uso del codigo QR

McDonald's, pone códigos QR en las tapas de las bebidas, de modo que los consumidores pueden ver el valor nutricional de sus comidas.

Los Museos colocan sus códigos QR al lado de sus exhibiciones, para dirigir a los visitantes a tours online o auditivos vía sus teléfonos, o para proveer información mas detallada.

En Brancott Estate, utilizan los QR para dar al consumidor mas información acerca del producto, incluyendo sugerencias de comidas con las que combinar y promociones.

La campaña "World Park" del Central Park de Nueva York, con la finalidad de conectar con la audiencia joven para el Arbor Day, convirtió al parque en un juego interactivo utilizando códigos QR, posicionados alrededor del parque, que linkeaban a una gran variedad de información.

1.7.2.3. Herramientas de desarrollo

- **Android Studio**

  Es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan tu productividad durante la compilación de apps para Android, como las siguientes:

  - Un sistema de compilación basado en Gradle flexible.
  - Un emulador rápido con varias funciones.
  - Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.
  - Instant Run para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.
  - Integración de plantillas de código y GitHub para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código.

- Gran cantidad de herramientas y frameworks de prueba.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK
- Soporte incorporado para Google Cloud Platform, lo que facilita la integración de Google Cloud Messaging y App Engine

- **Back4app**

Parse fue una plataforma popular de BaaS, pero lamentablemente Facebook ha decidido cerrarlo. Facebook retiró el servicio de alojamiento de parse.com en enero de 2017, pero el SDK de Parse ha pasado a ser de código abierto. El SDK de Parse tiene un montón de grandes desarrolladores que trabajan en él, y hay un número de nuevo sitios web que ofrecen hosting de back-end como servicio.

- **JDK**

Es un software que provee herramientas de desarrollo para la creación de programas en Java. Puede instalarse en una computadora local o en una unidad de red. En la unidad de red se pueden tener las herramientas distribuidas en varias computadoras y trabajar como una sola aplicación.

- **Inkscape**

Es un editor de gráficos vectoriales gratuito y de código libre. Inkscape puede crear y editar diagramas, líneas, gráficos, logotipos, e ilustraciones complejas. El formato principal que utiliza el programa es Scalable Vector Graphics (SVG) versión 1.1. Inkscape tiene como objetivo proporcionar a los usuarios una herramienta libre de código abierto de elaboración de gráficos en formato vectorial escalable (SVG) que cumpla completamente con los estándares XML, SVG y CSS2.Inkscape se encuentra desarrollado principalmente para el sistema operativo GNU/Linux, pero es una herramienta multiplataforma y funciona en Windows, Mac OS X, y otros sistemas derivados de Unix.

1.7.3. Metodología de desarrollo

Para el desarrollo de esta aplicación se implementará la metodología SCRUM ya que nos asegura que al final de cada sprint (iteración) se pueda entregar un producto que sea funcional, esto nos brinda la posibilidad de ajustar las funcionalidades en base a la necesidad de negocio cliente en nuestro caso serán representadas por las pruebas piloto; SCRUM hace una clara diferencia entre dos grupos para garantizar que quienes tienen la responsabilidad tienen

también la autoridad necesaria para poder lograr el éxito del proceso, y que quienes no la tienen no puedan producen interferencias innecesarias

En SCRUM se trabajan Sprint que es el tiempo en el cual se desarrolla un incremento en la funcionalidad; duración máxima: 30 días; durante el Sprint no puede modificarse el contenido del Sprint Backlog

Sprint Backlog: Lista de tareas extraídas del Product Backlog que serán convertidas en un incremento de funcionalidad; es recomendable que las tareas tengan una duración entre 4 y 16 hr e 4 y 16 hrs, en caso de tareas mayores deben intentar descomponerse en sub-tareas de ese rango de tiempo.

Para empezar en SCRUM existen tres roles:

Product Owner: Representa a todos los interesados en el producto final; marca las prioridades del producto, lleva el control de las estimaciones, retorno de Inversión (ROI)

Scrum Team: Debe transformar las tareas del Sprint Backlog en un incremento de funcionalidad en el software; desarrollar el producto con calidad, Auto-gestionado, Auto-organizado, Multi-funcional.

Scrum Master: Responsable del proceso de Scrum; Incorporación de Scrum en la cultura de la organización, Asegura el cumplimiento de los roles y responsabilidades, Formación y entrenamiento en el proceso.

En SCRUM hay implicados:

- Usuarios finales
- Marketing
- Áreas comerciales
- Áreas contables, etc.

Y hay comprometidos:

- Product Owne
- Scrum Team
- Scrum Master.

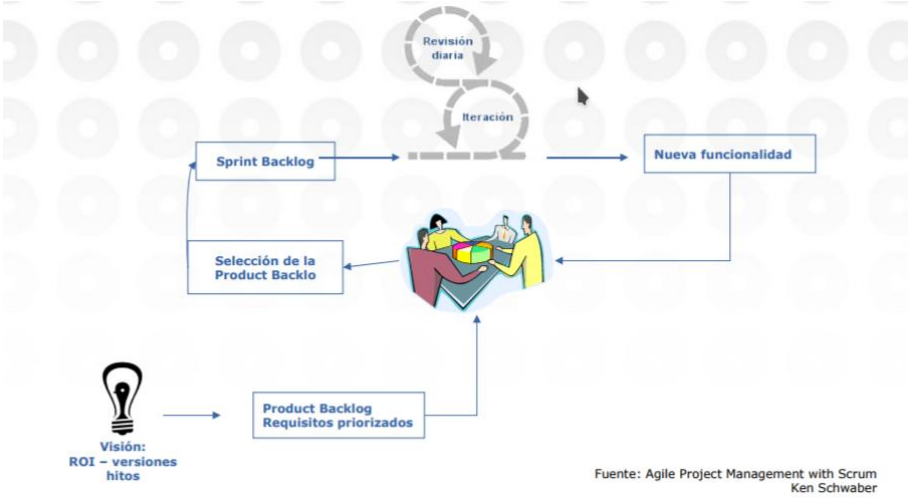


Ilustración 1 Flujo de SCRUM

Transparencia

Los aspectos significativos del proceso deben ser visibles para aquellos que son responsables del resultado. La transparencia requiere que dichos aspectos sean definidos por un estándar común, de tal modo que los observadores compartan un entendimiento común de lo que se está viendo.

Inspección

Los usuarios de Scrum deben inspeccionar frecuentemente los artefactos de Scrum y el progreso hacia un objetivo, para detectar variaciones. Su inspección no debe ser tan frecuente como para que interfiera en el trabajo. Las inspecciones son más beneficiosas cuando se realizan de forma diligente por inspectores expertos, en el mismo lugar de trabajo.

Adaptación

Si un inspector determina que uno o más aspectos de un proceso se desvían de límites aceptables, y que el producto resultante no será aceptable, el proceso o el material que está siendo procesado deben ser ajustados. Dicho ajuste debe realizarse cuanto antes para minimizar desviaciones mayores.

1.8. FACTIBILIDAD DE DESARROLLO

1.8.1. Factibilidad Económica

1.8.1.1. Factibilidad Económica - Software y hardware

Tabla 1 factibilidad Económica – software y hardware

Software	**Herramientas**	**Nombre**		**Costo**
		Sistema Operativo – Linux Mint		Software Libre
		Android Studio		Software Libre
		Inkscape		Software Libre
		Back4app		Software Libre
		Sub-Total		**0**
Hardware	**Elemento**	**Nombre**	**Detalle**	**Costo**
		Portátil	Dell Inspiron 14	$ 1500000
		Procesador	Intel Core i5	
		RAM	8GB	
		Disco Duro	1TB	
		Sub-Total		$ 1500000
Físico		**Valor Mensual**		**Valor Total**
	Transporte	$ 88000		**$ 528000**
	Fotocopias	$ 10000		**$ 40000**
	Impresiones	$ 10000		**$ 40000**
	Otros	$ 20000		**$ 80000**
	Sub-Total	**$ 688000**		

1.8.1.2. Factibilidad Económica - Recursos Humanos

Tabla 2 factibilidad Económica – Recursos Humanos

Nombre	**Función**	**Valor Hora**	**Cantidad/Horas**	**Subtotal**
Rocio Rodríguez Guerrero	Tutora	$ 40000	100	$ 4’000000
Juan Sebastián Buitrago	Desarrollador	$ 20000	420	$ 8’400000
Jimena Astrid Rojas Duarte	Desarrollador	$ 20000	420	$ 8’400000
Sub-Total				**$ 20’800000**

El proyecto es viable ya que las personas que conforman el grupo de recursos humanos tienen conocimiento del tema y tienen énfasis en el mismo

1.8.1.3. Factibilidad Económica – Total Costos

Tabla 3 factibilidad Económica – Total Costos

Recurso	Valor
Total Recursos Humanos	$ 20’800000
Total Recursos Técnicos	$ 1’000000
Total Otros Recursos	$ 80000
Costos Imprevistos 10%	$ 21880000
Total	**$ 24’068000**

1.8.2. Factibilidad Legal

1.8.2.1. Licencia de software

El desarrollo de la aplicación se completará aplicando una licencia sobre el software como es debido, la cual será la licencia Freeware el cual se usa comúnmente para programas que permiten la redistribución, pero no la modificación es decir su código fuente no está disponible.

1.8.3. Factibilidad Técnica

El proyecto que se va a desarrollar necesita como recursos mínimos de hardware los siguientes:

Tabla 4 factibilidad Técnica

Hardware	Software
Disco Duro 1000 GB	Sistema Operativo
Procesador Intel i5	Android Studio
Memoria RAM 6GB	Inkscape

El planteamiento anterior nos indica que el proyecto es viable técnicamente no solo porque se poseen las herramientas y recursos, sino que además el acceso a estos recursos es de cómodo acceso.

1.8.4. Factibilidad Operativa

Tabla 5 factibilidad Operativa

Integrantes del Equipo	Función	Perfil
Roció Rodríguez Guerrero	Tutora y Asesora del proyecto	
Juan Sebastián Buitrago Romero	Desarrollador	Estudiante en Tecnologia en Sistematizacion de Datos
Jimena Astrid Rojas Duarte	Desarrollador	Estudiante en Tecnologia en Sistematizacion de Datos

El proyecto es factible operativamente, porque se cuenta con el equipo de trabajo necesario para su desarrollo.

1.9. CRONOGRAMA DE ACTIVIDADES

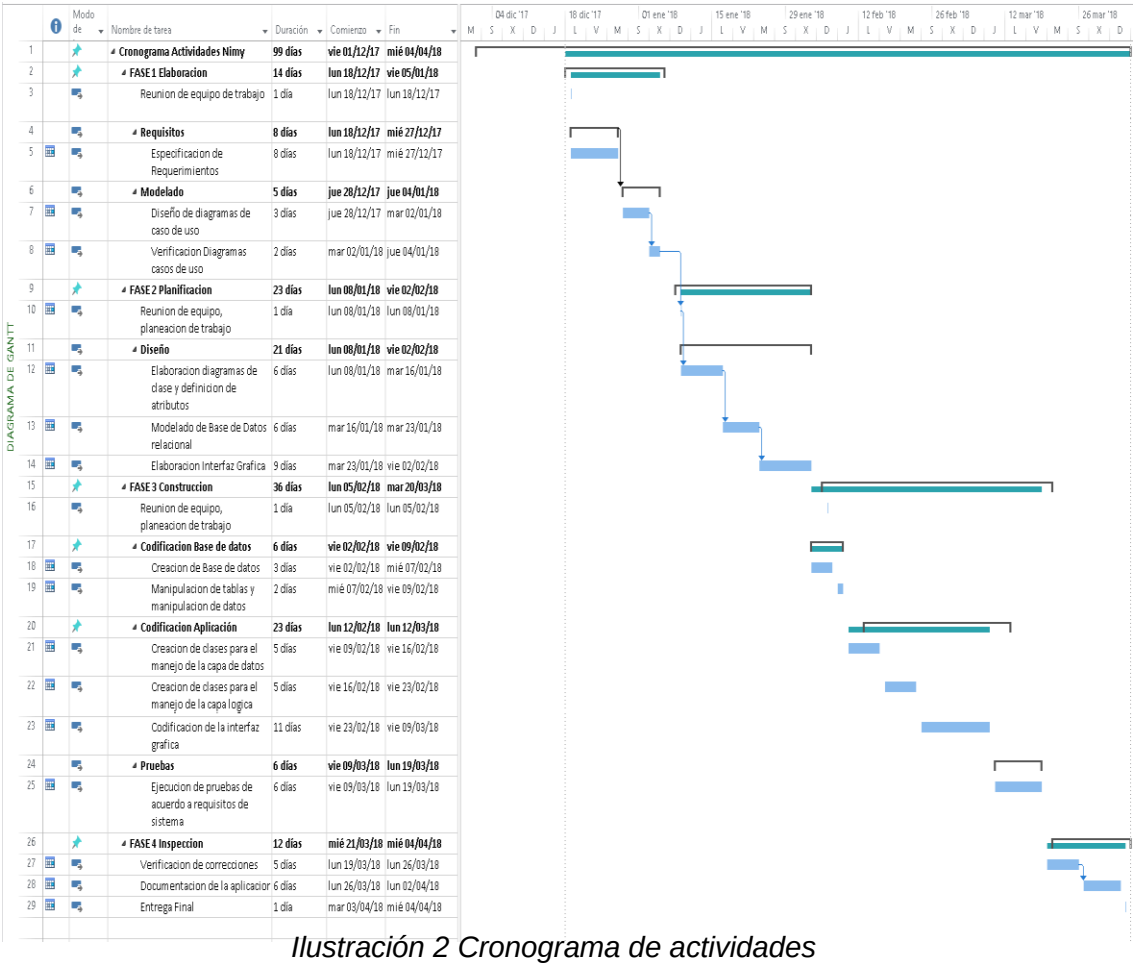


Ilustración 2 Cronograma de actividades

2. FASE 2 - ANALISIS DEL PROYECTO

2.1. Identificación de los roles

Se identificaron los respectivos roles, con el nombre de la persona responsable y una debida descripción de sus funciones

Tabla 6Roles

ROL	RESPONSABLE	DESCRIPCION
Scrum master:	**Jimena Astrid Rojas**	Es la persona conocedora del proceso que se encarga de orientar al equipo
Product owner	**Juan Sebastian Buitrago**	Es la persona que hace de intermediario entre el cliente y el equipo y tiene capacidad de decisión sobre los requisitos a implementar.
Team	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	Es el personal técnico que se encargará de la realización del proyecto.

2.2. Lista de actividades por rol

Se listan cada una de las actividades que los integrantes del team deben realizar según su rol.

Tabla 7 Lista de actividades

ROL	RESPONSABLE	ACTIVIDAD
Scrum master:	**Jimena Astrid Rojas**	• Velar por que todos los participantes del proyecto sigan los valores y principios ágiles, las reglas y proceso de Scrum • Facilitar las reuniones de Scrum • Quitar los impedimentos que el equipo tiene en su camino para

		conseguir el objetivo de cada iteración
Product owner	**Juan Sebastián Buitrago**	• Manejar reportes, cuentas de resultados y responsabilidad sobre el resultado final. • Se encargan de transferir la estrategia de producto a partir de una fragmentación adecuada del Product Backlog. • Definir los objetivos del producto o proyecto.
Team	**Juan Sebastián Buitrago** **Jimena Astrid Rojas**	• Seleccionar los requisitos que se compromete a completar en una iteración. • Estimar la complejidad de cada requisito en la lista de requisitos priorizada del producto o proyecto. • Trabajar de manera conjunta para conseguir los objetivos de la iteración.

2.3. Backlog

Tabla 8 Backlog

ID	DESCRIPCION	PRIORIZACION	ESTIMACION ESFUERZO	RESPONSABLE	SPRINT
1	Seleccionar los datos del usuario	**ALTA**	**6**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**1,2,8**
2	Agregar Mascota	**ALTA**	**6**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**3,8**
3	Eliminar mascota	**ALTA**	**6**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**8,10**
4	Agregar evento	**ALTA**	**6**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**5,8**
5	Seleccionar mascota	**ALTA**	**4**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**4,8**
6	Agregar ubicación mascota	**ALTA**	**5**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**3,8**
7	Crear una base de datos	**MEDIA**	**2**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**1,8**
8	Generar un Codigo QR	**ALTA**	**3**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**6,8**
9	Escanear Codigo QR	**ALTA**	**6**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**9,10**

10	Actualizar ubicación de una mascota	**ALTA**	**6**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**7,8**
11	Consultar agenda	**MEDIA**	**4**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**5,8**
12	Guardar datos usuario	**MEDIA**	**4**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**1,8**
13	Guardar Datos mascota	**MEDIA**	**4**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**3,8**
14	Loggear Usuarios	**ALTA**	**6**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**2,8**
15	Validar Campos de Datos	**ALTA**	**6**	**Juan Sebastian Buitrago** **Jimena Astrid Rojas**	**1,2,3**
16	Solicitar Permisos	**ALTA**	**4**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**3,7**
17	Creación de una interfaz	**ALTA**	**4**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**1,2,3,4,5,6,7,9,10**
18	Crear un perfil	**ALTA**	**6**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**3**
19	Seleccionar una imagen	**ALTA**	**4**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**3**
20	Informar al usuario	**MEDIA**	**4**	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**9**

2.4. Sprints

Lista de ciclos de trabajo de tipo sprints para la realización del proyecto

Tabla 9 Sprints

Sprint	Nombre	Descripción
Sprint 1	**Registro de Usuario**	Ingresa los datos del usuario
Sprint 2	**Login**	Ingresa los datos del usuario para que pueda ingresar a la aplicación
Sprint 3	**Registrar Mascota**	Permite crear un perfil con una imagen y registra los datos de una mascota
Sprint 4	**Seleccionar Mascota**	Selecciona una mascota
Sprint 5	**Agregar Eventos**	Ingresa datos para generar el recordatorio de un evento particular
Sprint 6	**Generar Código QR**	Genera un código de tipo QR a partir de los datos de una mascota
Sprint 7	**Buscar Mascota**	Lee el código QR para luego mostrar la información y la ubicación de la mascota
Sprint 8	**Base de datos**	Almacena datos de toda la aplicación
Sprint 9	**Información**	Permite informar al usuario sobre el funcionamiento de la aplicación
Sprint 10	**Menú Principal**	Unifica todos las funciones en una sola

3. FASE 3 – DISEÑO DE LOS SPRINT

3.1. Sprint 1 Registro de Usuario

En esta fase de sprint, se desarrolló una interfaz gráfica con el fin de registrar los datos de las personas que deseen usar la aplicación.

3.1.1. Definición de Sprint

Tabla 10 Definición Sprint 1

Sprint N. 1	**Nombre**	Registro de Usuario
Objetivo	Registrar los datos del usuario para que pueda usar la aplicación	
Detalle		
Se desarrolla una interfaz en donde el usuario puede ingresar datos tales como Nombre del Usuario, Numero celular, Correo electrónico y Contraseña. Además campos para confirmar Correo electrónico y Contraseña.		
Criterios de aceptación		
• Cada campo de texto debe tener su respectiva información • El tamaño del texto sea visible en todo tipo de usuario		

3.1.2. Tareas del Sprint 1

Tabla 11Tareas Backlock Sprint 1

Product Backlog	**Descripción de la tarea**	**Responsable**	**Estado**	**Duración (Hora)**
1	Se seleccionaron datos que sean acordes con el registro de un usuario para que pueda acceder a la aplicación	**Jimena Astrid Rojas**	**Terminado**	**1**
7	Se desarrolló e implemento una base de datos	**Jimena Astrid Rojas**	**Terminado**	**3**
12	La base de datos permitió guardar los datos del usuario	**Jimena Astrid Rojas**	**Terminado**	**2**

15	Se validaron los datos por cada campo	**Juan Sebastián Buitrago** **Jimena Astrid Rojas**	**Terminado**	**1**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas**	**Terminado**	**4**

3.1.3. Diseño del sprint 1 Registro de Usuario

La información es contenida en campos de tipo texto, para que sean digitados los datos de la persona que se desea registrarse, estos campos son, Nombre del Usuario, Teléfono, Correo Electrónico, Contraseña y campos de confirmación de Correo Electrónico y Contraseña. El registro se dará cuando se oprima el botón de Registrarse.



Ilustración 3 Sprint Registro de Usuario

3.1.4. Proceso de desarrollo del software Sprint 1 Registro de Usuarios

Mediante el gestor de bases de datos no Relacionales Back4app, se guarda la información directamente en la nube, para ello se deben cargar los repositorios y librerías correspondientes al motor para su uso, además este gestor permite facilidad en el manejo de datos de los usuarios.

```
compile 'com.parse:parse-android:1.16.7'
```

Ilustración 4 Librería Del Motor de bases de datos

Los datos se guardan y se insertan en una tabla de forma local, de esta manera de que cuando haya conexión a la internet las tablas se sincronicen en la nube.

```
Parse.initialize(new Parse.Configuration.Builder( context: this) .applicationId("jBJh6jKCJipl2TdDTcTYt51Ftyg9jOBhfcfhubjj")
        .clientKey("chJvmPMPKY0QDIuklkvqmAfrWgxRkBNUiy8dEmoe")
        .server("https://parseapi.back4app.com")
        .enableLocalDataStore()
        .build() );
```

```
ParseUser user=new ParseUser();
user.setUsername(txtcorreo.getText().toString());
user.setEmail(txtcorreo.getText().toString());
user.setPassword(txtcontraseña.getText().toString());
user.put("telephone", txttelefono.getText().toString());
user.put("nombre_usuario", txtnombre.getText().toString());
user.signUpInBackground((e) → {
        if (e == null) {
            Toast.makeText( context: RegistrarUsuario.this, text: "Registro Completado, Por Favor Confirme Correo Electronico", Toast.LENGTH_SHO
                    .show();
            ParseUser.logOut();
            Intent intent = new Intent( packageContext: RegistrarUsuario.this, Login.class);
            startActivity(intent);
        } else {
            Toast.makeText( context: RegistrarUsuario.this, e.getMessage().toString(), Toast.LENGTH_SHORT).show();
            ParseUser.logOut();
        }
});
```

Ilustración 5 Almacenamiento de Datos

3.1.5. Diagrama de contexto Sprint 1

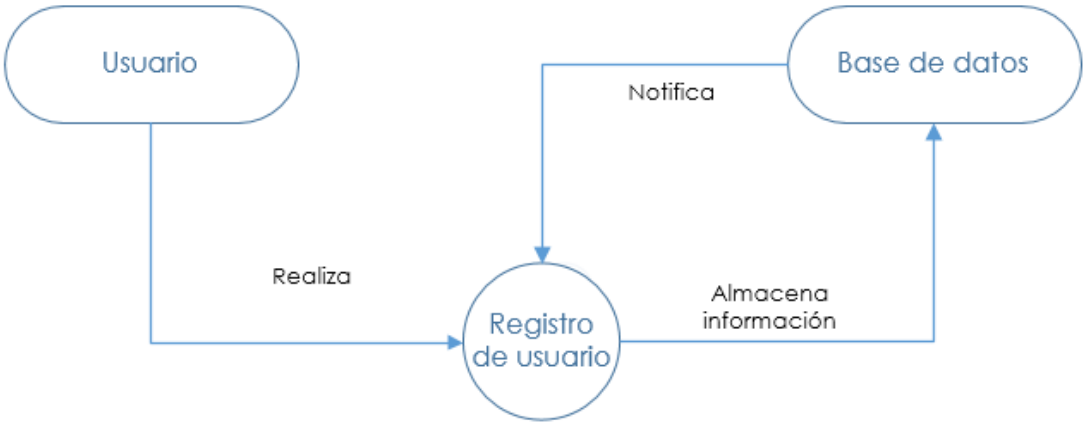


Ilustración 6 Diagrama de contexto Sprint 1

3.1.6. Pruebas del Sprint 1 Registro de Usuarios

Se hicieron pruebas en el Sprint 1 para probar que se cumplieran los criterios de los usuarios

Tabla 12 Pruebas Backlock Sprint 1

Sprint 1	Registro de Usuarios	
Descripción de la prueba	Se usó la herramienta del entorno de desarrollo Android Studio, Android Device Monitor, que permitió verificar que la información se almacenara en la base de datos.	
Herramientas	Android Device Monitor	
Prueba		**¿Cumple?**
1	Se seleccionaron datos acordes con el registro de un usuario	Si
2	Existe una base de datos donde se almacena información del usuario	Si
3	Se almacena con éxito datos del usuario	Si
4	Los campos de textos están acorde con la información correspondiente	Si

5	La interfaz gráfica es intuitiva y amigable con el usuario	Si

3.2. Sprint 2 Login

Este sprint tiene como función, permitir que un usuario registrado pueda ingresar en la aplicación

3.2.1. Definición de Sprint 2

Tabla 13 Definición Sprint 2

Sprint N. 2	**Nombre**	Login
Objetivo	Desarrollar un módulo que permita ingresar usuarios	
Detalle		
Se desarrolla una interfaz en donde el usuario puede ingresar con sus datos previamente registrado a la aplicación		
Criterios de aceptación		
• Cada campo de texto debe tener su respectiva información • El tamaño del texto sea visible en todo tipo de usuario • Notificar cuando una contraseña sea errónea o si el correo también lo es • Notificar que el usuario debe registrarse en caso de que no tenga acceso • Notificar cambio de contraseña en caso de que olvide contraseña		

3.2.2. Tareas Sprint 2 Login

Tabla 14 Tareas Backlock Sprint 2

Product Backlog	**Descripción de la tarea**	**Responsable**	**Estado**	**Duración (Hora)**
1	Se seleccionaron datos que sean acordes con el registro de un usuario para que pueda acceder a la aplicación	**Jimena Astrid Rojas**	**Terminado**	**1**
	Consultar los datos del			

14	usuario, con respecto a su correo electrónico y contraseña, con el fin de permitir el acceso a la aplicación	**Jimena Astrid Rojas**	**Terminado**	**6**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas**	**Terminado**	**4**

3.2.3. Diseño de Sprint 2 Login

Se diseñó una interfaz para el login con campos de texto de acuerdo a Correo Electrónico y contraseña, con los cuales la persona podrá ingresar en la aplicación con el botón iniciar sesión. En caso de que no esté registrado la persona deberá oprimir el botón registrarse para poder registrar los datos de su usuar

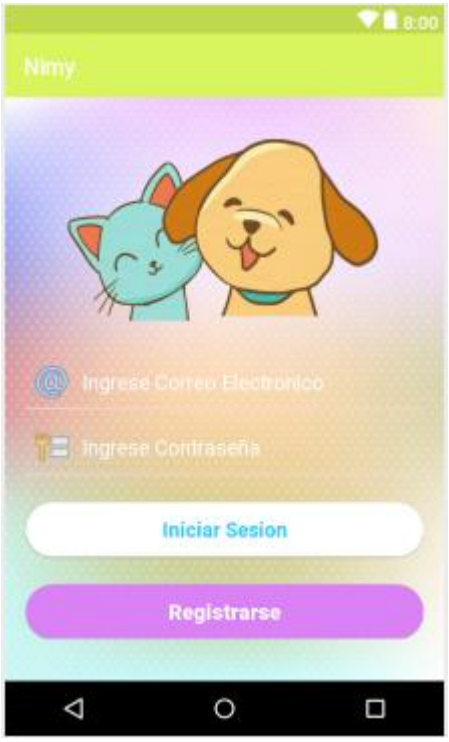


Ilustración 7 Sprint Login

3.2.4. Proceso de desarrollo de software Sprint 2 Login

Con la interfaz de Logueo diseñada, se procede a confirmar los datos del usuario en la base de datos, para ello, la persona deberá digitar su correo electrónico y contraseña así verificar que tiene un usuario registrado y se dará acceso a la aplicación. Esta confirmación está dada por una consulta de datos de correo electrónico y contraseña.

```
if(v==btnLogin){
    if(txtContrasena.getText().toString().isEmpty()){
        Toast.makeText( context: Login.this, text: "Por Favor Ingrese una Contraseña", Toast.LENGTH_SHORT)
                .show();
    }

    if(!txtUsuario.getText().toString().isEmpty() && !txtContrasena.getText().toString().isEmpty()){
        ParseUser.logInInBackground(txtUsuario.getText().toString(),txtContrasena.getText().toString(), (user, e) → {
                if (user != null) {
                    if(user.getBoolean( key: "emailVerified")) {
                        Toast.makeText( context: Login.this, text: "Inicio de Sesion", Toast.LENGTH_SHORT)
                                .show();
                        Intent intent = new Intent( packageContext: Login.this, MainActivity.class);
                        startActivity(intent);
                    }else{
                        ParseUser.logOut();
                        Toast.makeText( context: Login.this, text: "Por Favor Verfique Correo", Toast.LENGTH_SHORT)
                                .show();
                    }
                } else {
                    ParseUser.logOut();
                    Toast.makeText( context: Login.this, e.getMessage(), Toast.LENGTH_LONG).show();
                }

        });
    }
```

Ilustración 8 Consulta y verificación de datos de Usuario

3.2.5. Diagrama de Contexto Sprint 2

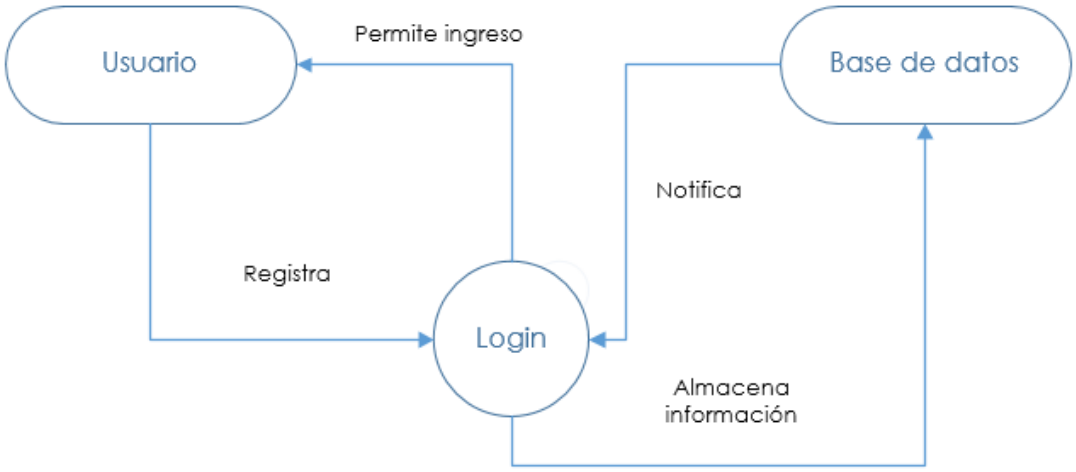


Ilustración 9 Diagrama de contexto Sprint 2

3.2.6. Prueba de Sprint 2

Se hicieron pruebas en el Sprint 2 para probar que se cumplieran los criterios de los usuarios

Tabla 15 Pruebas Backlock Sprint 2

Sprint 2	Login	
Descripción de la prueba	Se Comprueba que el usuario queda registrado, usando el Dashboard.Back4app de la aplicación, se demostró que agrega los usuarios por medio de una consulta directa en el panel de control de motor de bases de datos	
Herramientas	Parse-Dashboard.Back4app	
	Prueba	**¿Cumple?**
1	Los campos de textos son correspondientes con la información	Si
2	El texto es visible para el usuario	Si
3	Se notifica que alguno de los campos son erróneos	Si
4	Se notifica que el usuario debe registrarse	Si
5	Se notifica cambio de contraseña en caso de que sea olvidada por el usuario	Si

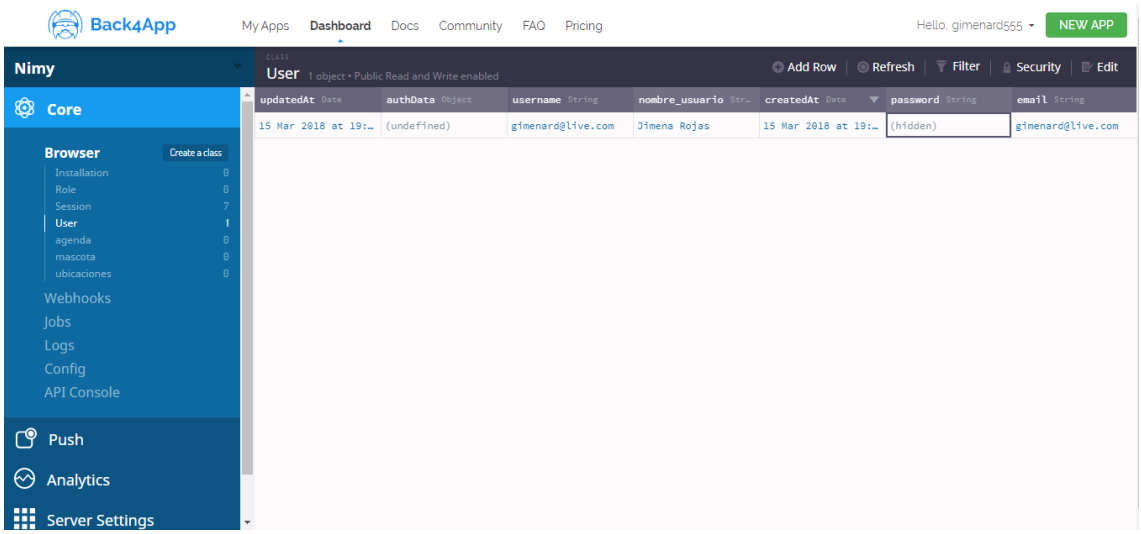


Ilustración 10 Dashboard Back4App, tabla de usuarios registrados

3.3. Sprint 3 Registrar Mascota

En esta fase sprint se desarrollaron y diseñaron interfaces gráficas que permitan al usuario crear un perfil con avatar que le permita agregar datos de una mascota, tales como, nombre, fecha de nacimiento, raza, tipo de mascota, color de mascota y sexo.

3.3.1. Definicion de Sprint 3 Registrar mascota

Tabla 16 Definición Sprint 3

Sprint N. 3	**Nombre**	Registrar mascota
Objetivo	Desarrollar un módulo que permita registrar mascotas	
Detalle		
Se desarrolla una interfaz en donde el usuario puede registrar sus mascotas		
Criterios de aceptación		
• Cada campo de texto debe tener su respectiva información • El tamaño del texto sea visible en todo tipo de usuario • Notificar que la mascota se haya registrado • La fecha de nacimiento no debe ser superior a la fecha actual • Se deben pedir permisos para acceder al GPS • Crear un perfil con avatar donde se encuentren los datos de la mascota		

3.3.2. Tareas de Sprint 3

Tabla 17 Tareas Backlock Sprint 3

Product Backlog	**Descripción de la tarea**	**Responsable**	**Estado**	**Duración (Hora)**
2	Registra los datos de la mascota, tales como Nombre, raza, fecha de nacimiento, tipo de mascota, color y genero	**Jimena Astrid Rojas Juan Sebastian Buitrago**	**Terminado**	**6**
6	Agrega la ubicación mascota por	**Jimena Astrid Rojas Juan Sebastian**	**Terminado**	**5**

	medio del GPS del dispositivo	**Buitrago**		
13	Almacena los datos de la mascota en una tabla dentro de un base de datos	**Juan Sebastián Buitrago**	**Terminado**	**4**
16	Solicitar permisos para acceder al GPS del dispositivo	**Jimena Astrid Rojas**	**Terminado**	**4**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas**	**Terminado**	**4**
18	Crea un perfil con los datos de una mascota	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
19	Selecciona un avatar con los datos de un perfil	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

3.3.3. Diseño de Sprint 3 Registrar Mascota

Se diseñó una interfaz para el registro de mascota en donde el usuario digita datos tales como, nombre, fecha de nacimiento, raza, tipo de mascota (Canino o Felino), color y genero (masculino o femenino); la fecha de nacimiento se validó de tal forma que solo acepte fechas iguales o menores a la fecha actual; y de un botón seleccionar que al presionarse guardara los datos de la mascota en el dispositivo y llevara a seleccionar una imagen ya sea, de la galería de selección de avatar propia de la aplicación, tomar una foto con el dispositivo, o bien de la galería de imágenes del dispositivo, para crear el perfil.



Ilustración 11 Sprint Registro mascotas



Ilustración 12 Sprint Crear Perfil

3.3.4. Proceso de desarrollo de software Sprint 3 Registrar mascota

Con la interfaz de registro de mascota diseñada, se procedió a validar los datos correspondientes, la fecha de nacimiento, además al presionar el botón de siguiente, se tomará la posición del usuario como la ubicación actual de la mascota, por medio del GPS del dispositivo, luego se guardarán los datos de la mascota y la ubicación en el dispositivo en una tabla de bases de datos local, para luego subirse a la base de datos diseñada en la nube.

```
public void onClick(View v) {
    if (v == fecha) {
        final Calendar c = Calendar.getInstance();
        dia = c.get(Calendar.DAY_OF_MONTH);
        mes = c.get(Calendar.MONTH);
        año = c.get(Calendar.YEAR);
        DatePickerDialog datePickerDialog = new DatePickerDialog( context: RegistrarMascota.this,
                R.style.Theme_AppCompat, oyentefecha, año, mes, dia);
        datePickerDialog.getDatePicker().setMaxDate(System.currentTimeMillis());
        datePickerDialog.show();
    }
    if (v == btnSiguiente) {
        if (nombre.getText().toString().isEmpty()) {
            Toast.makeText( context: RegistrarMascota.this,  text: "Por Favor Ingrese Nombre de Mascota", Toast.LENGTH_SHORT)
                    .show();
        } else if (fecha.getText().toString().isEmpty()) {
            Toast.makeText( context: RegistrarMascota.this,  text: "Por Favor Ingrese Fecha de Nacimiento", Toast.LENGTH_SHORT)
                    .show();
        } else if (raza.getText().toString().isEmpty()) {
            Toast.makeText( context: RegistrarMascota.this,  text: "Por Favor Ingrese una Raza", Toast.LENGTH_SHORT)
                    .show();
        } else if (!macho.isChecked() && !hembra.isChecked()) {
            Toast.makeText( context: RegistrarMascota.this,  text: "Por Favor Seleccione un Sexo", Toast.LENGTH_SHORT)
                    .show();
        }
        if (!nombre.getText().toString().isEmpty() && !fecha.getText().toString().isEmpty() && !raza.getText().toString().isEmpty
                && (macho.isChecked() || hembra.isChecked())) {
```

Ilustración 13 Validación de campos de texto

Además, se piden permisos para que la aplicación tenga acceso al GPS del dispositivo, cuando se están guardando los datos en la base de datos.

```
LocationManager locationManager = (LocationManager)
        getSystemService(Context.LOCATION_SERVICE);
Criteria criteria = new Criteria();
if (ActivityCompat.checkSelfPermission( context: this, android.Manifest.permission.ACCESS_FINE_LOCATION)
        != PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission
        ( context: this, android.Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
    return;
}
Location location = locationManager.getLastKnownLocation(locationManager
        .getBestProvider(criteria,  enabledOnly: false));
this.latitud = location.getLatitude();
this.longitud = location.getLongitude();
ParseUser currentUser = ParseUser.getCurrentUser();
this.animal.setEmail_dueño(currentUser.getEmail());
this.animal.setDueño(currentUser.getString( key: "nombre_usuario"));
this.animal.setNumero_tel(currentUser.getString( key: "telephone"));
this.animal.setNombre(nombre.getText().toString());
this.animal.setFecha_nacimiento(fecha.getText().toString());
this.animal.setTipo(tipo.getSelectedItem().toString());
this.animal.setRaza(raza.getText().toString());
this.animal.setColor(color.getSelectedItem().toString());
this.animal.setDireccion(obtenerDireccion(latitud, longitud));
if (hembra.isChecked()) {
    animal.setSexo(hembra.getText().toString());
} else if (macho.isChecked()) {
    animal.setSexo(macho.getText().toString());
}
validar();
```

Ilustración 14 Captura de datos por medio de los campos de texto

Se realizó una validación para la mascota, así logrando evitar que no haya mascotas repetidas.

```
public void validar() {
    ParseQuery<ParseObject> mascotaQuery = ParseQuery.getQuery("mascota");
    mascotaQuery.whereEqualTo("dueno_mascota", ParseUser.getCurrentUser());
    mascotaQuery.findInBackground((objects, e) → {
            if (e == null) {
                for (int i = 0; i < objects.size(); i++) {
                    if (objects.get(i).get("nombre").equals(animal.getNombre())) {
                        Toast.makeText( context: RegistrarMascota.this,  text: "La mascota " + animal.getNombre() + " ya existe",
                                Toast.LENGTH_SHORT).show();
                    } else {
                        Intent intent = new Intent( packageContext: RegistrarMascota.this, CrearPerfil.class);
                        intent.putExtra( name: "mascota", animal);
                        startActivity(intent);
                        Toast.makeText( context: RegistrarMascota.this,  text: "La mascota " + animal.getNombre() + " se ha registrado"
                                Toast.LENGTH_SHORT).show();
                    }
                }
            }
    });
}
```

Ilustración 15 Validación de que la mascota no exista en la base de datos

La dirección se obtiene por medio de dos variables de tipo Latitud y Longitud, fundamentales para generar una coordenada cartesiana, para no mostrar al usuario las coordenadas, se tuvo que diseñar un método que tiene como fin mostrar la dirección en donde se puede encontrar la mascota

```
private String obtenerDireccion(double latitud, double longitud) {
    String strAdd = "";
    Geocoder geocoder = new Geocoder( context: this, Locale.getDefault());
    try {
        List<Address> addresses = geocoder.getFromLocation(latitud, longitud,  maxResults: 1);
        if (addresses != null) {
            Address returnedAddress = addresses.get(0);
            StringBuilder strReturnedAddress = new StringBuilder("");

            for (int i = 0; i <= returnedAddress.getMaxAddressLineIndex(); i++) {
                strReturnedAddress.append(returnedAddress.getAddressLine(i)).append("\n");
            }
            strAdd = strReturnedAddress.toString();
            Log.w( tag: "Mi dirección de ubicación actual", strReturnedAddress.toString());
        } else {
            Log.w( tag: "Mi dirección de ubicación actual",  msg: "Sin dirección devuelta!");
        }
    } catch (Exception e) {
        e.printStackTrace();
        Log.w( tag: "Mi dirección de ubicación actual",  msg: "No se puede obtener la dirección!");
    }
    return strAdd;
}
```

Ilustración 16 Generar la dirección de la mascota

Con los datos de la mascota y la ubicación de esta generada, se procede a crear el perfil de la mascota, en donde se selecciona una imagen:

- Una galería de selección de avatar de la aplicación, esta galería dependerá del tipo de mascota que se haya selecciona.
- Tomando una foto con el dispositivo
- Seleccionando una imagen de la galería de imágenes del dispositivo

La galería de selección de avatar, está dada por una serie de imágenes, almacenadas en arreglos de tipo entero. Las imágenes se encuentran dentro del diseño de la aplicación.

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_seleccionar_avatar);
    Parse.initialize( context: this);
    Intent intent = getIntent();
    animal = new Animal();
    actionBar = getSupportActionBar();
    actionBar.setDisplayHomeAsUpEnabled(true);
    actionBar.setDisplayUseLogoEnabled(true);
    animal = (Animal) intent.getSerializableExtra( name: "mascota");
    linearLayout = (GridLayout) findViewById(R.id.layout_avatar);
    linearLayout.setColumnCount(3);
    imageView = (ImageView) findViewById(R.id.elegirAvatar);
    circleImageViewPerro = new ArrayList<CircleImageView>();
    circleImageViewGato = new ArrayList<CircleImageView>();
    if (animal.getTipo().equals("Canino")) {
        iconos_perro[0] = (R.drawable.perro1);
        iconos_perro[1] = (R.drawable.perro2);
        iconos_perro[2] = (R.drawable.perro3);
        iconos_perro[3] = (R.drawable.perro4);
        iconos_perro[4] = (R.drawable.perro5);
        iconos_perro[5] = (R.drawable.perro6);
        iconos_perro[6] = (R.drawable.perro7);
        iconos_perro[7] = (R.drawable.perro8);
        iconos_perro[8] = (R.drawable.perro9);
        iconos_perro[9] = (R.drawable.perro10);
        iconos_perro[10] = (R.drawable.perro11);
        iconos_perro[11] = (R.drawable.perro12);
        interfaz_perro(iconos_perro);
    } else if (animal.getTipo().equals("Felino")) {
        iconos_gato[0] = (R.drawable.gato1);
        iconos_gato[1] = (R.drawable.gato2);
        iconos_gato[2] = (R.drawable.gato3);
        iconos_gato[3] = (R.drawable.gato4);
        iconos_gato[4] = (R.drawable.gato5);
        iconos_gato[5] = (R.drawable.gato6);
        iconos_gato[6] = (R.drawable.gato7);
        iconos_gato[7] = (R.drawable.gato8);
        iconos_gato[8] = (R.drawable.gato9);
        iconos_gato[9] = (R.drawable.gato10);
        iconos_gato[10] = (R.drawable.gato11);
        iconos_gato[11] = (R.drawable.gato12);
        interfaz_gato(iconos_gato);
    }
}
```

Ilustración 17 Arreglos de imágenes

```
public void interfaz_perro(Integer[] vector) {
    circleImageViewPerro = new ArrayList<CircleImageView>();
    imageView.setImageDrawable(getResources().getDrawable(R.drawable.i_gatitos));
    for (int i = 0; i < iconos_perro.length; i++) {
        LinearLayout.LayoutParams parms = new LinearLayout.LayoutParams( width: 200,  height: 200);
        CircleImageView view = new CircleImageView(getApplicationContext());
        view.setImageDrawable(getResources().getDrawable(vector[i]));
        view.setBorderOverlay(true);
        view.setBorderColor(Color.parseColor( colorString: "#C0DA4E"));
        view.setOnClickListener(this);
        view.setLayoutParams(parms);
        view.setForegroundGravity(Gravity.CENTER_HORIZONTAL);
        view.setPadding( left: 10,  top: 10,  right: 10,  bottom: 10);
        circleImageViewPerro.add(view);
        linearLayout.addView(view);
    }
}
```

Ilustración 18 Método que recorre el arreglo de imágenes para Caninos

```
public void interfaz_gato(Integer[] vector) {
    circleImageViewGato = new ArrayList<CircleImageView>();
    imageView.setImageDrawable(getResources().getDrawable(R.drawable.i_perritos));
    for (int i = 0; i < iconos_gato.length; i++) {
        LinearLayout.LayoutParams parms = new LinearLayout.LayoutParams( width: 200,  height: 200);
        CircleImageView view = new CircleImageView(getApplicationContext());
        view.setImageDrawable(getResources().getDrawable(iconos_gato[i]));
        view.setBorderOverlay(true);
        view.setBorderColor(Color.parseColor( colorString: "#C0DA4E"));
        view.setOnClickListener(this);
        view.setLayoutParams(parms);
        view.setForegroundGravity(Gravity.CENTER_HORIZONTAL);
        view.setPadding( left: 10,  top: 10,  right: 10,  bottom: 10);
        circleImageViewGato.add(view);
        linearLayout.addView(view);
    }
}
```

Ilustración 19 Método que recorre el arreglo imágenes para Felinos

En caso de que el usuario desee tomar directamente la foto, la aplicación deberá usar la cámara del dispositivo.

```
private void abrirCamara() {
    File file = new File(Environment.getExternalStorageDirectory(), MEDIA_DIRECTORY);
    boolean isDirectoryCreated = file.exists();
    if (!isDirectoryCreated)
        isDirectoryCreated = file.mkdirs();
    if (isDirectoryCreated) {
        Long timestamp = System.currentTimeMillis() / 1000;
        String imageName = timestamp.toString() + ".tiff";
        mPath = Environment.getExternalStorageDirectory() + File.separator + MEDIA_DIRECTORY
                + File.separator + imageName;
        File newFile = new File(mPath);
        Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(newFile));
        startActivityForResult(intent, PHOTO_CODE);
    }
}
```

Ilustración 20 Permisos para uso de cámara del dispositivo

```
private void mostrarOpciones() {
    final CharSequence[] option = {"Tomar foto", "Elegir de galeria", "Elegir Avatar", "Cancelar"};
    final AlertDialog.Builder builder = new AlertDialog.Builder( context: CrearPerfil.this);
    builder.setTitle("Elige una opción");
    builder.setItems(option, (dialog, which) → {
            if (option[which] == "Tomar foto") {
                abrirCamara();
            } else if (option[which] == "Elegir de galeria") {
                Intent intent = new Intent(Intent.ACTION_PICK, android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
                intent.setType("image/*");
                startActivityForResult(intent.createChooser(intent, title: "Selecciona imagen para la app"), SELECT_PICTURE);
            } else if (option[which] == "Elegir Avatar") {
                Intent intent = new Intent( packageContext: CrearPerfil.this, SeleccionarAvatar.class);
                intent.putExtra( name: "mascota", animal);
                startActivityForResult(intent, SELECT_AVATAR);

            } else {
                dialog.dismiss();
            }
    });

    builder.show();
}
```

Ilustración 21 Opciones para seleccionar imagen de perfil

3.3.5. Diagrama de contexto Sprint 3

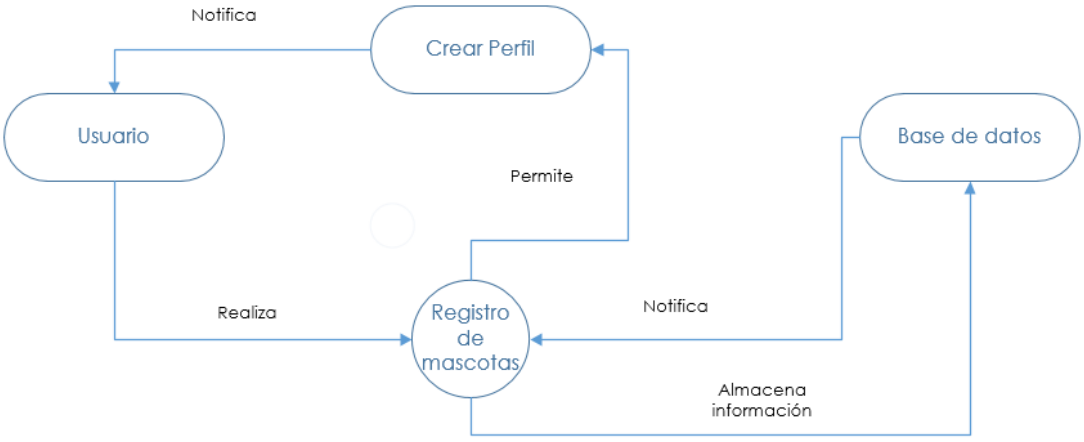


Ilustración 22 Diagrama de contexto Sprint 3

3.3.6. Prueba de Sprint 3

Se hicieron pruebas en el Sprint 3 para probar que se cumplieran los criterios de los usuarios

Tabla 18 Pruebas Backlock Sprint 3

Sprint 3	Registro de mascotas	
Descripción de la prueba	Se Comprueba que estén registrando mascotas haciendo consultas sobre la base de datos.	
Herramientas	Parse-Dashboard.Back4app	
Prueba		**¿Cumple?**
1	Los campos de textos son correspondientes con la información	Si
2	El texto es visible para el usuario	Si
3	Notificar que la mascota se haya registrado	Si
4	La fecha de nacimiento no debe ser superior a la fecha actual	Si
5	Se deben pedir permisos para acceder al GPS	Si
6	Los datos de la mascota y se guardan en un perfil con una imagen	Si

3.4. Sprint 4 Seleccionar mascota

En esta fase Sprint se diseñó y desarrollo una interfaz gráfica que permita al usuario seleccionar una mascota.

3.4.1. Definición de Sprint 4 Seleccionar Mascota

Tabla 19 Definición Sprint 4

Sprint N. 3	**Nombre**	**S**eleccionar mascota
Objetivo	Desarrollar un módulo que permita seleccionar mascotas	
Detalle		
Se desarrolla una interfaz en donde el usuario puede seleccionar una mascota		
Criterios de aceptación		
• Se debe mostrar el perfil de las mascotas registradas • Permitir agregar otro perfil		

3.4.2. Tareas de Sprint 4

Tabla 20 Tareas Backlock Sprint 4

Product Backlog	**Descripción de la tarea**	**Responsable**	**Estado**	**Duración (Hora)**
5	Se diseñó una interfaz que permite seleccionar mascotas	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

3.4.3. Diseño de Sprint 4 Seleccionar mascota

Se diseñó una interfaz permite mostrar los perfiles de mascotas que se han registrado y así seleccionar uno; además cuando el tenga otra mascota que desee registrar en la aplicación, esta permite agregar un nuevo perfil



Ilustración 23 Sprint Seleccionar mascota

3.4.4. Proceso de desarrollo de software Sprint 4 Seleccionar Mascota

Con la interfaz de selección de mascota diseñada, se procedió a realizar una consulta en la base de datos, con el objeto de mostrar los perfiles de mascotas registrados por el usuario.

El usuario se relaciona con la mascota por medio del nombre con el que la persona se registró.

```
public void consulta() {
    ParseQuery<ParseObject> mascotaQuery = ParseQuery.getQuery("mascota");
    mascotaQuery.whereEqualTo("dueno_mascota", ParseUser.getCurrentUser());
    mascotaQuery.findInBackground((mascotas, e) → {
            if (e == null) {
                Log.d( tag: "mascotas",  msg: "Retrieved " + mascotas.size() + " mascotas");
                for (int i = 0; i < mascotas.size(); i++) {
                    Animal animal = new Animal();
                    animal.setNombre((String) mascotas.get(i).get("nombre"));
                    ParseFile foto = (ParseFile) mascotas.get(i).getParseFile( key: "foto");
                    try {
                        Bitmap bitmap = BitmapFactory.decodeByteArray(foto.getData(),  offset: 0, foto.getData().length);
                        animal.setFoto(bitmap);
                    } catch (ParseException e1) {
                        e1.printStackTrace();
                    }
                    animals.add(animal);
                }
                interfaz(animals);
            } else {
                Log.d( tag: "mascota",  msg: "Error: " + e.getMessage());
            }
    });

}
```

Ilustración 24 Opciones para seleccionar imagen de perfil

3.4.5. Diagrama de contexto Sprint 4

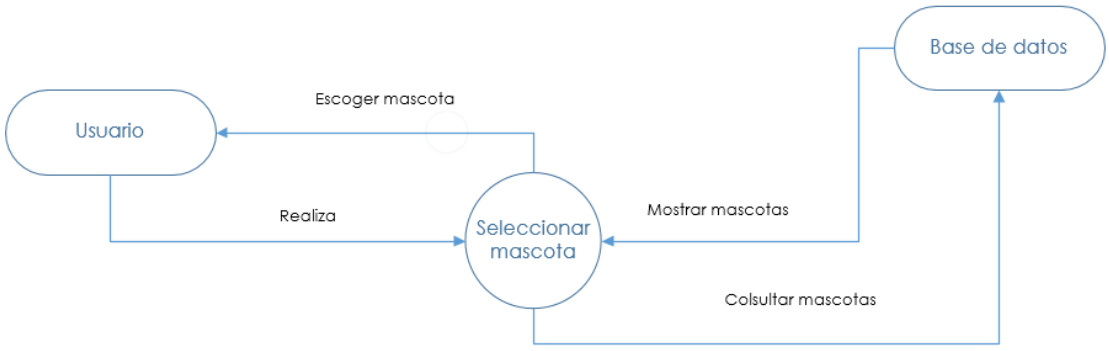


Ilustración 25 Diagrama de Contexto Sprint 4

3.4.6. Prueba de Sprint 4

Se hicieron pruebas en el Sprint 4 para probar que se cumplieran los criterios de los usuarios

Tabla 21 Pruebas Backlock Sprint 4

Sprint 4	Seleccionar Mascota
Descripción de la prueba	Se usó la herramienta del entorno de desarrollo Android Studio, Android Device Monitor, que permitió verificar que la información se almacenara en la base de datos.
Herramientas	Android Device Monitor

Prueba		¿Cumple?
1	Se muestran los perfiles de mascotas	Si
2	Permite agregar otro perfil	Si

3.5. Definición de Sprint 5 Agregar Eventos

Tabla 22 Definición Sprint 5

Sprint N. 5	**Nombre**	**Agregar Eventos**
Objetivo	Desarrollar un módulo que permita registrar eventos	
Detalle		
Se desarrolla una interfaz en donde el usuario pueda registrar eventos de su mascota		
Criterios de aceptación		
• Cada campo de texto debe tener su respectiva información • El tamaño del texto sea visible en todo tipo de usuario • Se debe registrar un evento para una mascota • Generar un recordatorio al usuario del evento más próximo		

3.5.1. Tareas de Sprint 5

Tabla 23 Tareas Backlock Sprint 5

Product Backlog	Descripción de la tarea	Responsable	Estado	Duración (Hora)
4	Se diseñó una interfaz que permita registrar eventos relacionados con una mascota	**Jimena Astrid Rojas Juan Sebastian Buitrago**	**Terminado**	**4**
11	Se diseñó una interfaz que muestre los próximos eventos	**Jimena Astrid Rojas Juan Sebastian Buitrago**	**Terminado**	**4**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas Juan Sebastian Buitrago**	**Terminado**	**4**

3.5.2. Diseño de Sprint 5 Agregar Evento

Se diseñó una interfaz permite mostrar los eventos que se hayan registrado previamente, caso contrario, con el botón agregar evento, se carga otra interfaz en donde se permite que el usuario pueda registrar un evento, el evento tiene datos tales como, descripción, tipo de evento (Vacuna, baño u otro) y fecha del evento.



Ilustración 26 Sprint Agregar Evento 1

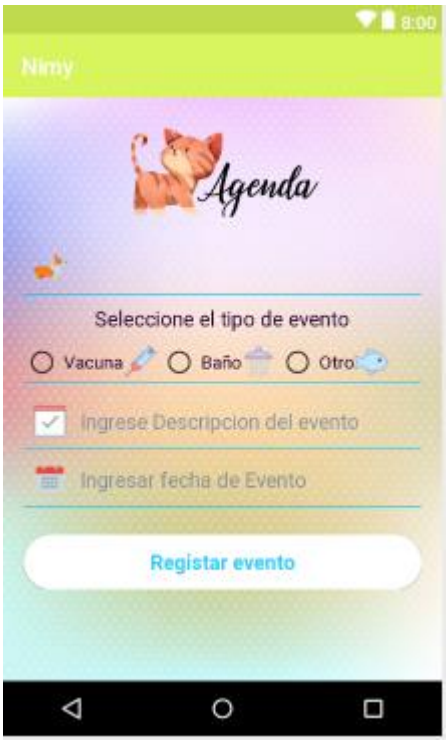


Ilustración 27 Sprint Agregar Evento 2

3.5.3. Proceso de desarrollo de software Sprint 5 Agregar Evento

Con las interfaces para agregar eventos diseñadas, se procedió a realizar una consulta en la base de datos, con el objeto de mostrar los eventos próximos y agregar los datos en una tabla de forma local.

```
public void onClick(View view) {
    if (view == fecha) {
        final Calendar c = Calendar.getInstance();
        dia = c.get(Calendar.DAY_OF_MONTH);
        mes = c.get(Calendar.MONTH);
        anio = c.get(Calendar.YEAR);
        DatePickerDialog datePickerDialog = new DatePickerDialog(
                context: RegistrarEvento.this, R.style.Theme_AppCompat, oyentefecha, anio, mes, dia);
        datePickerDialog.show();
    }
    if (view == btnAgregarEvento) {
        if (!vacuna.isChecked() && !Baño.isChecked() && !Otro.isChecked()) {
            Toast.makeText( context: RegistrarEvento.this,
                    text: "Por Favor Seleccione el tipo de Evento", Toast.LENGTH_SHORT)
                    .show();
        } else if (descripcion.getText().toString().isEmpty()) {
            Toast.makeText( context: RegistrarEvento.this,
                    text: "Por Favor Ingrese la descripcion para el Evento", Toast.LENGTH_SHORT)
                    .show();
        } else if (fecha.getText().toString().isEmpty()) {
            Toast.makeText( context: RegistrarEvento.this, text: "Por Favor Ingrese Fecha para el Evento", Toast.LENGTH_SHORT)
                    .show();
        }
    if (!descripcion.getText().toString().isEmpty() && !fecha.getText().toString().isEmpty() &&
            (vacuna.isChecked() || Baño.isChecked() || Otro.isChecked())) {
        evento.setDescripcion(descripcion.getText().toString());
        evento.setFecha(fecha.getText().toString());
        if (vacuna.isChecked()) {
            evento.setTipo(vacuna.getText().toString());
        }
        if (Baño.isChecked()) {
            evento.setTipo(Baño.getText().toString());
        }
        if (Otro.isChecked()) {
            evento.setTipo(Otro.getText().toString());
        }
        consulta();
        Toast.makeText( context: RegistrarEvento.this, text: "Evento Guardado", Toast.LENGTH_SHORT).show();
        Intent intent = new Intent( packageContext: RegistrarEvento.this, MainActivity.class);
        intent.putExtra( name: "nombre_mascota", animal.getNombre());
        startActivity(intent);
    }
```

Ilustración 28 Registro de eventos

El método Consulta permite agregar los datos que se desean registrar a la base de datos, este método además permite traer los datos del usuario relacionados con la mascota y los registra en una nueva tabla en donde se encontraran los eventos.

```
ParseQuery<ParseObject> mascotaQuery = ParseQuery.getQuery("mascota");
mascotaQuery.whereEqualTo("dueno_mascota", ParseUser.getCurrentUser());
mascotaQuery.findInBackground((mascotas, e) → {
        if (e == null) {
            for (int i = 0; i < mascotas.size(); i++) {
                if (mascotas.get(i).get("nombre").equals(nombre_mascota)) {
                    final ParseQuery<ParseObject> eventoQuery = ParseQuery.getQuery("agenda");
                    eventoQuery.whereEqualTo("mascota", mascotas.get(i));
                    eventoQuery.findInBackground((eventos, e) → {
                        if (e == null) {
                            for (int i = 0; i < eventos.size(); i++) {
                                Evento evento = new Evento();
                                evento.setFecha(eventos.get(i).getString( key: "fecha"));
                                evento.convertirFecha();
                                Event event = new Event(Color.GREEN, evento.getFechaTiempo(),
                                        data: eventos.get(i).getString( key: "tipo") + "-" +
                                                eventos.get(i).getString( key: "descripcion"));
                                calendario.addEvent(event);
                            }
                        }
                    });
                }
            }
        } else {
```

Ilustración 29 Consulta de los próximos eventos

Para saber cuál es el evento que se acerca y así realizar una notificación, se realizó una consulta sobre la tabla eventos, en donde se mostrara en un calendario el evento más próximo.

3.5.4. Diagrama de contexto Sprint 5



Ilustración 30 Diagrama de contexto Sprint 5

3.5.5. Prueba de Sprint 5

Se hicieron pruebas en el Sprint 5 para probar que se cumplieran los criterios de los usuarios

Tabla 24 Prueba Backlock Sprint 5

Sprint 5	Agregar Eventos	
Descripción de la prueba	Se usó la herramienta del entorno de desarrollo Android Studio, Android Device Monitor, que permitió verificar que la información se almacenara en la base de datos.	
Herramientas	Android Device Monitor	
Prueba		**¿Cumple?**
1	Los campos de textos son correspondientes con la información	Si
2	El texto es visible para el usuario	Si
3	El registro de eventos es exitoso	Si
4	Se genera el recordatorio de eventos	Si

3.6. Definición de Sprint 6 Generar Código QR

Tabla 25 Definicion Sprint 6

Sprint N. 6	**Nombre**	**Generar Código QR**
Objetivo	Desarrollar un módulo que permita generar un código QR	
Detalle		
Se desarrolla una interfaz en donde el usuario pueda descargar un código QR con los datos del usuario y su mascota		
Criterios de aceptación		
• Cada campo de texto debe tener su respectiva información • El tamaño del texto sea visible en todo tipo de usuario • Generar un código QR con los datos de un usuario y el perfil de una mascota seleccionada		

3.6.1. Tareas de Sprint 6

Tabla 26 Tareas Backlock Sprint 6

Product Backlog	Descripción de la tarea	Responsable	Estado	Duración (Hora)
8	Se diseñó una interfaz que cargue los datos de un usuario y su mascota y generar un código QR	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

3.6.2. Diseño de Sprint 6 Generar Codigo QR

Se diseñó una interfaz permite mostrar los datos del usuario y una mascota previamente seleccionada, se mostrarán datos tales como, nombre del dueño, teléfono, dirección, color de la mascota y nombre de la mascota



Ilustración 31 Sprint Generar código QR

3.6.3. Proceso de desarrollo de software Sprint 6 Generar Código QR

Con las interfaces para generar el código QR diseñadas, se procedió a realizar una consulta en la base de datos, con el objeto de mostrar los datos de la mascota seleccionada, junto con su dueño, para luego generar el código QR con esos datos.

Por medio de la relación entre dueño y mascota, se traen los datos que necesitan de ambos, el nombre del usuario, teléfono, nombre de la mascota, el color de la mascota y la dirección que se obtuvo en el registro de la mascota.

```
public void consulta() {
    ParseQuery<ParseObject> mascotaQuery = ParseQuery.getQuery("mascota");
    mascotaQuery.whereEqualTo("dueno_mascota", ParseUser.getCurrentUser());
    mascotaQuery.findInBackground((objects, e) → {
            if (e == null) {
                Log.d( tag: "mascotas",  msg: "Retrieved " + objects.size() + " mascotas");
                for (int i = 0; i < objects.size(); i++) {
                    if (objects.get(i).get("nombre").equals(nombreMascota)) {
                        animal.setDueño(ParseUser.getCurrentUser().getString( key: "nombre_usuario"));
                        animal.setNumero_tel(ParseUser.getCurrentUser().getCurrentUser().getString( key: "telephone"));
                        animal.setDireccion(objects.get(i).getString( key: "direccion"));
                        animal.setColor(objects.get(i).getString( key: "color"));
                        animal.setNombre(objects.get(i).getString( key: "nombre"));
                        ParseFile codigoQR = (ParseFile) objects.get(i).getParseFile( key: "codigoQR");
                        try {
                            Bitmap bitmap = BitmapFactory.decodeByteArray(codigoQR.getData(),  offset: 0,
                                    codigoQR.getData().length);
                            animal.setCodigoQR(bitmap);
                        } catch (ParseException e1) {
                            e1.printStackTrace();
                        }
                        pintarInterfaz();
                    }
                }

            } else {
                Log.d( tag: "mascota",  msg: "Error: " + e.getMessage());
            }
```

Ilustración 32 Consulta de datos de la mascota y el usuario

Para diseñar el código QR con los datos de la mascota y el usuario, se desarrolló un método que permite encapsular los datos en formato Bitmap.

```
public void pintarInterfaz() {
    nombre_dueño.setText(this.animal.getDueño());
    nombre_dueño.setTypeface(lovely);
    telefono.setText(this.animal.getNumero_tel());
    telefono.setTypeface(lovely);
    direccion.setText(this.animal.getDireccion());
    direccion.setTypeface(lovely);
    color_mascota.setText(this.animal.getColor());
    color_mascota.setTypeface(lovely);
    nombre_mascota.setText(this.animal.getNombre());
    nombre_mascota.setTypeface(lovely);
    this.codigoQR.setImageBitmap(this.animal.getCodigoQR());
}
```

Ilustración 33 Construcción de código QR por medio de los datos de la mascota y el usuario

Con el código generado, se procede a guardar el código en la base de datos de forma local. Para descargar el código y que se pueda leer el código se desarrolló el método descargarQR, método que guarda el código en el dispositivo.

```
private void descargarQR() {
    File file = new File(Environment.getExternalStorageDirectory(), MEDIA_DIRECTORY);
    boolean isDirectoryCreated = file.exists();
    if (!isDirectoryCreated)
        isDirectoryCreated = file.mkdirs();
    if (isDirectoryCreated) {
        Long timestamp = System.currentTimeMillis() / 1000;
        String imageName = timestamp.toString() + ".jpg";
        mPath = Environment.getExternalStorageDirectory() + File.separator + MEDIA_DIRECTORY
                + File.separator + imageName;
        File newFile = new File(mPath);
        try {
            FileOutputStream fOut = new FileOutputStream(file);
            animal.getCodigoQR().compress(Bitmap.CompressFormat.JPEG,  quality: 85, fOut);
            fOut.flush();
            fOut.close();
            MakeSureFileWasCreatedThenMakeAvabile(file);
            AbleToSave();
        } catch (FileNotFoundException e) {
            UnableToSave();
            Toast.makeText( context: DescargarQR.this, e.getMessage().toString(), Toast.LENGTH_SHORT).show();
        } catch (IOException e) {
            UnableToSave();
            Toast.makeText( context: DescargarQR.this, e.getMessage().toString(), Toast.LENGTH_SHORT).show();
        }
    }
}
```

Ilustración 34 Método que permite descargar el código QR generado

3.6.4. Diagrama de contexto Sprint 6

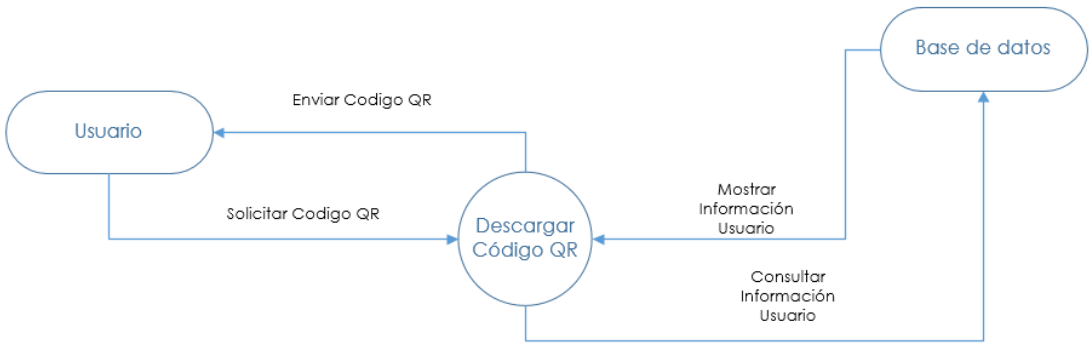


Ilustración 35 Diagrama de contexto Sprint 6

3.6.5. Prueba de Sprint 6

Se hicieron pruebas en el Sprint 6 para probar que se cumplieran los criterios de los usuarios

Tabla 27 Prueba Backlock Sprint 6

Sprint 6	Generar Codigo QR	
Descripción de la prueba	Se usó la herramienta del entorno de desarrollo Android Studio, Android Device Monitor, que permitió verificar que se generara un código QR con los datos de un usuario y su mascota.	
Herramientas	Android Device Monitor	
Prueba		**¿Cumple?**
1	Los campos de textos son correspondientes con la información	Si
2	El texto es visible para el usuario	Si
3	El código QR con los datos del usuario y la mascota seleccionada, se genera con exito	Si

3.7. Definición de Sprint 7 Buscar mascota

Tabla 28 Definición Sprint 7

Sprint N. 7	**Nombre**	**Buscar mascota**
Objetivo	Desarrollar un módulo que permita encontrar una mascota	
Detalle		
Se diseña una interfaz en donde el usuario que encuentra una mascota perdida actualiza la posición y automáticamente se notifica una nueva posición		
Criterios de aceptación		
• Cada campo de texto debe tener su respectiva información • El tamaño del texto sea visible en todo tipo de usuario • Que se actualice la posición de la mascota en el mapa		

3.7.1. Tareas de Sprint 7

Tabla 29 Tareas Backlock Sprint 7

Product Backlog	Descripción de la tarea	Responsable	Estado	Duración (Hora)
10	Se debe actualizar la posición de la mascota encontrada	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
16	La aplicación debe solicitar permiso para usar el gps del dispositivo	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

3.7.2. Diseño de Sprint 7 Buscar mascota

Se diseñó una interfaz permite mostrar los datos del usuario de la mascota que se perdió, esto es posible debido a que se lee el código QR que asignada la respectiva mascota.

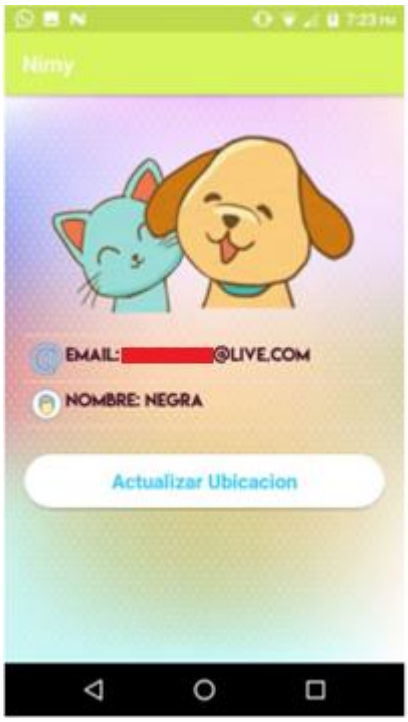


Ilustración 36 Sprint Buscar Mascota 1

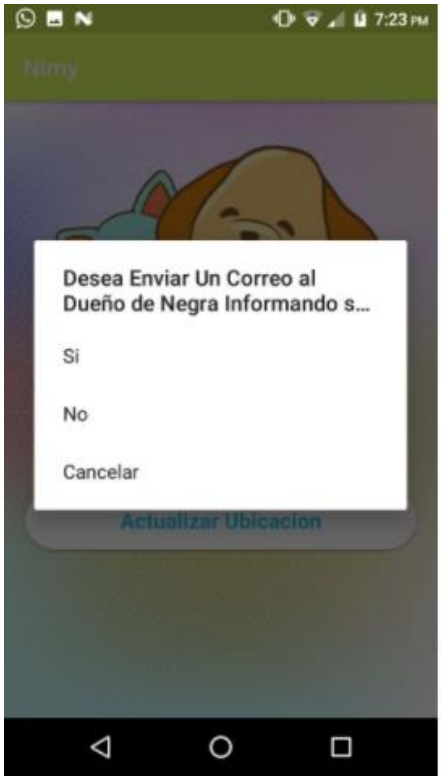


Ilustración 37 Sprint Buscar Mascota 2



Ilustración 38 Sprint Buscar Mascota 3

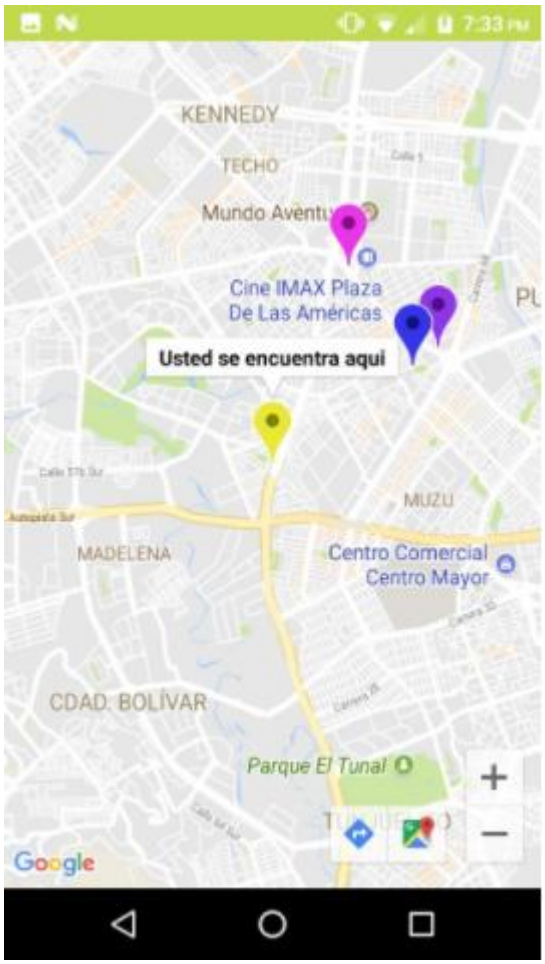


Ilustración 39 Sprint Buscar Mascota 4

3.7.3. Proceso de desarrollo de software Sprint 7 Buscar Mascota

Con la interfaz para Buscar mascotas diseñada, se procedió a dar permisos para que la aplicación pueda usar el GPS, con este, al leer el código QR se carga los datos del usuario dueño de la mascota perdida y en seguida se deberá presionar el botón actualizar ubicación, en donde, enviara vía correo electrónico la notificación de que han encontrado la mascota. Posteriormente se actualizará los puntos donde la mascota ha estado.

```
IntentIntegrator integrator = new IntentIntegrator( activity: MainActivity.this);
integrator.setDesiredBarcodeFormats(IntentIntegrator.QR_CODE_TYPES);
integrator.setPrompt("Escanear Codigo QR");
integrator.setOrientationLocked(true);
integrator.setCameraId(0);
integrator.initiateScan();
```

Ilustración 40 Lectura del código de barras

Los datos del usuario dueño de la mascota perdida son traídos por medio de una consulta en las tablas de usuario y mascota de la base datos, para mostrar el correo electrónico y la dirección de la mascota con la que se registró.

```
public void generar() {
    String[] datos_correo = datos[0].split( regex: " ");
    final String[] datos_nombre = datos[1].split( regex: " ");
    ParseQuery<ParseUser> userQuery = ParseUser.getQuery();
    userQuery.whereEqualTo("email", datos_correo[1]);
    userQuery.findInBackground((objects, e) → {
            if (e == null) {
                ParseQuery<ParseObject> mascotaQuery = ParseQuery.getQuery("mascota");
                mascotaQuery.whereEqualTo("dueno_mascota", objects.get(0));
                mascotaQuery.findInBackground((mascotas, e) → {
                        if (e == null) {
                            for (int i = 0; i < mascotas.size(); i++) {
                                if (mascotas.get(i).get("nombre").equals(datos_nombre[1])) {
                                    ParseObject puntoInsertar = new ParseObject( theClassName: "ubicaciones");
                                    ParseGeoPoint parseGeoPoint = new ParseGeoPoint(ubicacion.getLatitud(),
                                            ubicacion.getLongitud());
                                    puntoInsertar.put("ubicacion_punto", parseGeoPoint);
                                    puntoInsertar.put("fecha_lectura", ubicacion.getFecha_lectura());
                                    puntoInsertar.put("mascota", mascotas.get(i));
                                    puntoInsertar.saveInBackground();
                                } else {

                                }
                            }
                        }
```

Ilustración 41 Método que genera la ubicación de la mascota y trae los datos de contacto

El botón actualizar posición, permitirá generar la nueva ubicación y además le dará la posibilidad de que la persona que llegue a encontrar la mascota se ponga en contacto con el dueño de la mascota, se diseñó un método que permite contactarse más rápido por vía correo electrónico.

```
private void PreguntarCorreo() {
    final String[] datos_nombre = datos[1].split( regex: " ");
    final CharSequence[] option = {"Si", "No", "Cancelar"};
    final AlertDialog.Builder builder = new AlertDialog.Builder( context: ActualizarUbicacion.this);
    builder.setTitle("Desea Enviar Un Correo al Dueño de "+datos_nombre[1]+" Informando su Poscion Actual?");
    builder.setItems(option, (dialog, which) → {
            if (option[which] == "Si") {
                enviarCorreo();
            } else if (option[which] == "No") {
                Intent intent = new Intent( packageContext: ActualizarUbicacion.this, MainActivity.class);
                intent.putExtra( name: "nombre_mascota", nombre_interfaz);
                startActivity(intent);
            } else {
                dialog.dismiss();
            }
    });
    builder.show();
}
```

Ilustración 42 Método que permite decidir si desea contactarse con el dueño de una mascota perdida

```
protected void enviarCorreo() {
    final String[] datos_correo = datos[0].split( regex: " ");
    String[] TO = {datos_correo[1]};
    String[] CC = {""};
    Intent emailIntent = new Intent(Intent.ACTION_SEND);
    emailIntent.setData(Uri.parse("mailto:"));
    emailIntent.setType("text/plain");
    emailIntent.putExtra(Intent.EXTRA_EMAIL, TO);
    emailIntent.putExtra(Intent.EXTRA_SUBJECT,  value: "Nimy Actualizacion de su Mascota");
    emailIntent.putExtra(Intent.EXTRA_TEXT,  value: "He encontrado su mascota y ya he actualizado su posicion, actualmente su " +
            "mascota se encuentra en " + obtenerDireccion(ubicacion.getLatitud(), ubicacion.getLongitud()));
    try {
        startActivity(Intent.createChooser(emailIntent,  title: "Enviar email..."));
        finish();
    } catch (android.content.ActivityNotFoundException ex) {
        Toast.makeText( context: ActualizarUbicacion.this,
                text: "No tienes clientes de email instalados.", Toast.LENGTH_SHORT).show();
    }
}
```

Ilustración 43 Método que permite contactarse con el dueño de una mascota perdida

Luego de actualizarse la posición de la mascota el usuario podrá verificar los puntos en los que esta ha estado, se usó el api de google maps para mostrar en el mapa cada uno de los puntos, estos son dados por las variables de tipo latitud y longitud; se obtienen las coordenadas.

```
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    UiSettings uiSettings = mMap.getUiSettings();
    uiSettings.setZoomControlsEnabled(true);
    ParseQuery<ParseObject> mascotaQuery = ParseQuery.getQuery("mascota");
    mascotaQuery.whereEqualTo("dueno_mascota", ParseUser.getCurrentUser());
    mascotaQuery.findInBackground((objects, e) → {
            if (e == null) {
                Log.d( tag: "mascotas",  msg: "Retrieved " + objects.size() + " mascotas");
                for (int i = 0; i < objects.size(); i++) {
                    if (objects.get(i).get("nombre").equals(nombre_mascota)) {
                        ParseQuery<ParseObject> ubicacionesQuery = ParseQuery.getQuery("ubicaciones");
                        ubicacionesQuery.whereEqualTo("mascota", objects.get(i));
                        ubicacionesQuery.findInBackground((ubicaciones, e) → {
                                if (e == null) {
                                    if (ubicaciones.size() > 0) {
                                        for (int i = 0; i < ubicaciones.size(); i++) {
                                            Ubicacion ubicacion = new Ubicacion();
                                            ParseGeoPoint parseGeoPoint = ubicaciones.get(i).getParseGeoPoint
                                                    ( key: "ubicacion_punto");
                                            ubicacion.setFecha_lectura(ubicaciones.get(i).getString( key: "fecha_lectura"));
                                            ubicacion.setLatitud(parseGeoPoint.getLatitude());
                                            ubicacion.setLongitud(parseGeoPoint.getLongitude());
                                            LatLng marcador=new LatLng(parseGeoPoint.getLatitude(),
                                                    parseGeoPoint.getLongitude());
                                            mMap.addMarker(new MarkerOptions().position(marcador)
                                            .title(nombre_mascota+" estuvo el dia "+
                                                    ubicaciones.get(i).getString( key: "fecha_lectura")+" en este lugar")
                                            .icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_MAGENTA)));
                                        }
                                    }
LocationManager locationManager = (LocationManager)
        getSystemService(Context.LOCATION_SERVICE);
Criteria criteria = new Criteria();
if (ActivityCompat.checkSelfPermission( context: this, android.Manifest.permission.ACCESS_FINE_LOCATION) !=
        PackageManager.PERMISSION_GRANTED &&
        ActivityCompat.checkSelfPermission( context: this, android.Manifest.permission.ACCESS_COARSE_LOCATION) !=
                PackageManager.PERMISSION_GRANTED) {
    return;
}
Location location = locationManager.getLastKnownLocation(locationManager
        .getBestProvider(criteria,  enabledOnly: false));
LatLng marcador=new LatLng(location.getLatitude(), location.getLongitude());
mMap.addMarker(new MarkerOptions().position(marcador)
        .title("Usted se encuentra aqui")
        .icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_YELLOW)));
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(marcador,  v: 10));
```

Ilustración 44 Código que genera los puntos en el mapa

3.7.4. Diagrama de contexto Sprint 7

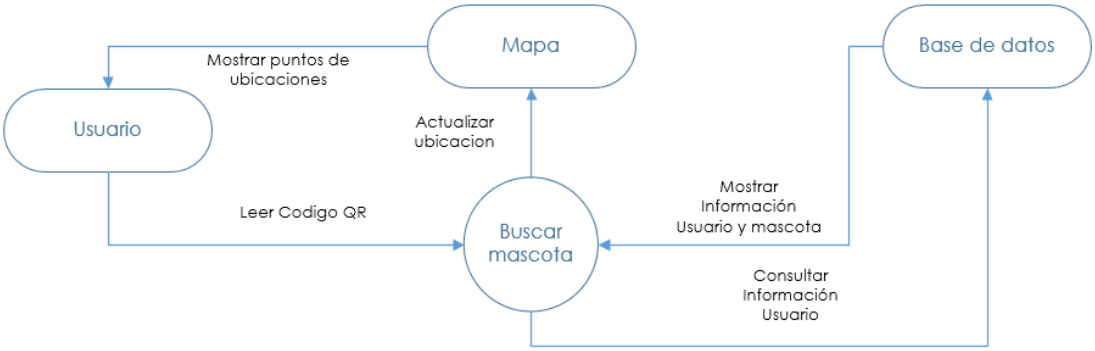


Ilustración 45 Diagrama de contexto Sprint 7

3.7.5. Prueba de Sprint 7

Se hicieron pruebas en el Sprint 7 para probar que se cumplieran los criterios de los usuarios

Tabla 30 Pruebas Backlock Sprint 7

Sprint 6	Buscar mascota	
Descripción de la prueba	Se usó la herramienta del entorno de desarrollo Android Studio, Android Device Monitor, que permitió verificar que por medio del lector QR se puede localizar una mascota	
Herramientas	Android Device Monitor	
Prueba		**¿Cumple?**
1	Los campos de textos son correspondientes con la información	Si
2	El texto es visible para el usuario	Si
3	La posición se actualiza en el mapa por medio del GPS	Si

3.8. Definición de Sprint 8 Base de datos

Tabla 31 Definición Sprint 8

Sprint N. 8	**Nombre**	**Base de datos**
Objetivo	Desarrollar un módulo que permita diseñar una base de datos	
Detalle		
Se diseña una base de datos con el objeto de que almacene la información relacionada con el ingreso a la aplicación, registro de usuarios, creación de perfiles de mascotas, ubicación de la mascota y el código QR generado; tal forma de que se pueda consultar dicha información por las interfaces de la aplicación.		
Criterios de aceptación		
• Persistencia de datos. • Con la información almacenada se pueda realizar consultas desde la aplicación.		

3.8.1. Tareas de Sprint 8

Tabla 32 Tareas Backlock Sprint 8

Product Backlog	**Descripción de la tarea**	**Responsable**	**Estado**	**Duración (Hora)**
1,	Consultar los datos del usuario	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
2	Almacenar datos de una mascota	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
3	Eliminar una mascota de la base de datos	**Jimena Astrid Rojas**	**Terminado**	**4**

		Juan Sebastian Buitrago		
4	Almacenar datos del evento en la base de datos	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
5	Consultar los datos de la mascota seleccionada	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
6	Almacenar la ubicación de la mascota	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
7	Diseñar una base de datos	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
8	Almacenar el código QR generado por la aplicación	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
10	Actualizar la ubicación de la mascota	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
11	Consultar eventos de la mascota	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

12	Almacenar los datos del usuario	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
13	Almacenar datos de la mascota	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
14	Permitir el ingreso a la aplicación	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

3.8.2. Diseño de Sprint 8 Base de datos

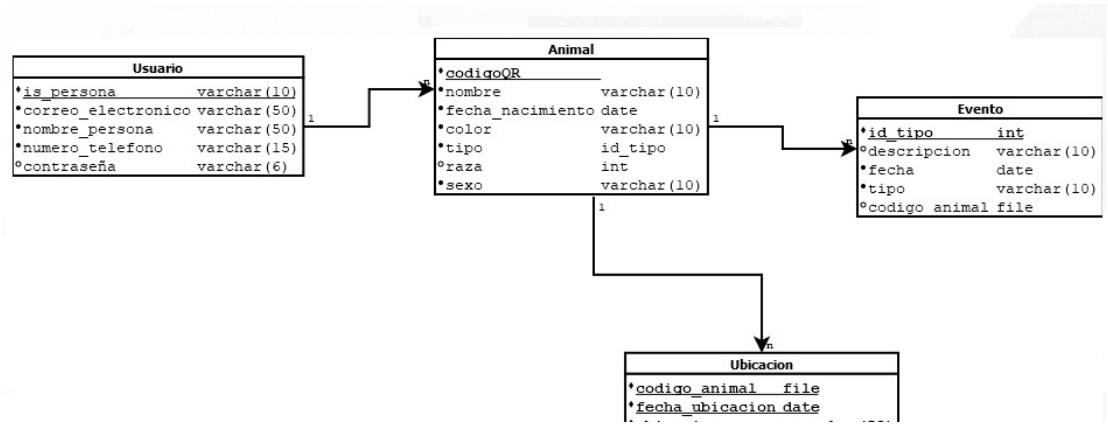


Ilustración 46 Diagrama Relacional base de datos

3.8.3. Prueba de Sprint 8

Tabla 33 Pruebas Backlock Sprint 8

Sprint 8	Base de datos	
Descripción de la prueba	Se comprueba que usando el Dashboard.Back4app, el panel de control de motor de bases de datos se puede verificar que los datos están siendo guardados	
Herramientas	Parse-Dashboard.Back4app	
Prueba		**¿Cumple?**
1	Hay Persistencia de datos con la aplicación	Si
2	Las consultas de la aplicación en la base de datos son exitosas	Si

3.9. Definición de Sprint 9 Información

Tabla 34 Definición Sprint 8

Sprint N. 9	**Nombre**	**Información**
Objetivo	Desarrollar un módulo que permita mostrar las funciones de la aplicación al usuario	
Detalle		
Diseñar un módulo que permita informar al usuario del funcionamiento de la aplicación, explicando cada botón y su respectiva función		
Criterios de aceptación		
• Que sea entendible para el usuario el funcionamiento de la aplicación		

3.9.1. Tareas de Sprint 9

Tabla 35 Tareas Backlock Sprint 9

Product Backlog	Descripción de la tarea	Responsable	Estado	Duración (Hora)
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
20	Se indicó al usuario el funcionamiento de las principales opciones de la aplicación	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

3.9.2. Diseño de Sprint 9 Información

Se diseñó una interfaz permite mostrar las principales funciones de la aplicación, para ello se usó el swipe view, herramienta para mostrar múltiples fragmentos dentro de una sola interfaz, cambiando entre ellas de forma horizontal.



Ilustración 47 Sprint Información 1



Ilustración 48 Sprint Información 2



Ilustración 49 Sprint Información 3



Ilustración 50 Sprint Información 4



Ilustración 51 Sprint Información 5

3.9.3. Proceso de desarrollo de software Sprint 9 Información

Después de diseñar cada fragmento de interfaz, se realizó el llamado de cada uno de ellos, con el objeto de mostrar en orden cada fragmento.

```
public static Fragment newInstance(int sectionNumber) {

    Fragment fragment = null;
    switch (sectionNumber){
        case 1:
            fragment = new pantalla_informacion_2 ();
            break;
        case 2:
            fragment = new pantalla_informacion_3 ();
            break;
        case 3:
            fragment = new pantalla_informacion_6 ();
            break;
        case 4:
            fragment = new pantalla_informacion_4 ();
            break;
        case 5:
            fragment = new pantalla_informacion_5 ();
            break;
    }
    Bundle args = new Bundle ();
    args.putInt ( ARG_SECTION_NUMBER, sectionNumber );

    return fragment;
}
```

Ilustración 52 Orden de llamado de cada Fragmento

3.9.4. Diagrama de contexto Sprint 9

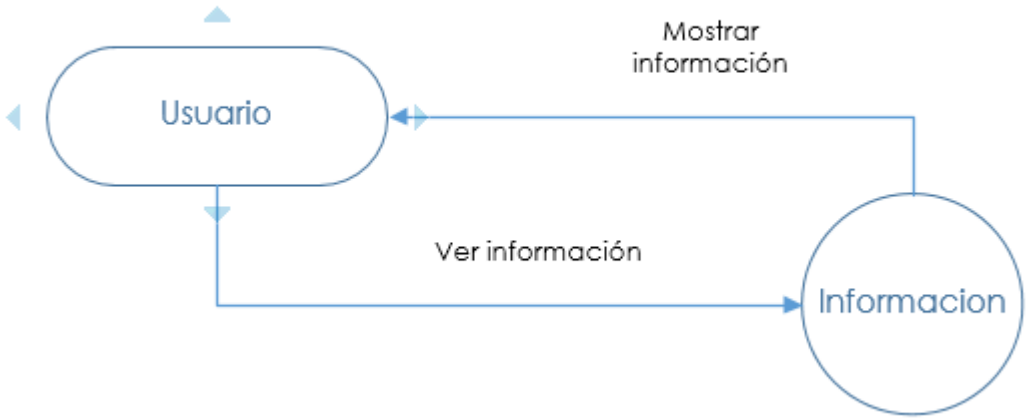


Ilustración 53 Diagrama de contexto Sprint 9

3.9.5. Prueba de Sprint 9

Tabla 36 Pruebas Backlock Sprint 9

Sprint 9	Información	
Descripción de la prueba	Se usó la herramienta del entorno de desarrollo Android Studio, Android Device Monitor, que permitió verificar que por medio del lector QR se puede localizar una mascota	
Herramientas	Android Device Monitor	
Prueba		**¿Cumple?**
1	La información es entendible para los usuarios	Si

3.10. Definición de Sprint 10 Menú Principal

Tabla 37 Definición Sprint 10

Sprint N. 10	**Nombre**	**Menú Principal**
Objetivo	Desarrollar un módulo que permita unificar las demás interfaces	
Detalle		
Diseñar un módulo que permita unificar todas la interfaces, por medio de botones, que al presionarlos se acceda a un función determinada		
Criterios de aceptación		
• Que se puedan acceder a cada una de las funciones que posee la aplicación		

3.10.1. Tareas de Sprint 10

Tabla 38 Tareas Backlock Sprint 10

Product Backlog	Descripción de la tarea	Responsable	Estado	Duración (Hora)
3	Permitir eliminar mascotas	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
9	Permitir el acceso de la cámara del dispostivo para leer un código QR	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**
17	Se diseñó una interfaz gráfica que es amigable para el usuario	**Jimena Astrid Rojas** **Juan Sebastian Buitrago**	**Terminado**	**4**

3.10.2. Diseño de Sprint 10 Menú principal

Se diseñó una interfaz permite mostrar los datos de una mascota que se haya seleccionado, además, tiene todas las interfaces unificadas a través de botones; en la parte superior se encuentran cinco botones donde se podrá, eliminar una mascota seleccionada, seleccionar mascota, leer el código QR, cambiar datos del usuario y la pantalla de información. En la parte inferior de la interfaz se encontrará un menú circular, que contara con tres botones que llevaran a opciones de, buscar mascota, agregar eventos y descarga el código QR



Ilustración 54 Sprint menú Principal

3.10.3. Proceso de desarrollo de software Sprint 10 Menú Principal

Después de diseñar la interfaz, se procedió a implementar en cada botón la función de llevar a cada una de las interfaces.

```
public void iniciarMenu() {
     /*Inicia Menu Principal*/
    circleMenu.addSubMenu(Color.parseColor( colorString: "#C0DA4E"), R.drawable.i_buscar);
    circleMenu.addSubMenu(Color.parseColor( colorString: "#C0DA4E"), R.drawable.i_agenda);
    circleMenu.addSubMenu(Color.parseColor( colorString: "#C0DA4E"), R.drawable.i_descargar);
    circleMenu.setOnMenuSelectedListener(new OnMenuSelectedListener() {
        @Override
        public void onMenuSelected(int i) {
            if (i == 0) {
                Intent intent = new Intent( packageContext: MainActivity.this, BuscarMascota.class);
                intent.putExtra( name: "mascota", animal.getNombre());
                startActivity(intent);
            }
            if (i == 1) {
                Intent intent = new Intent( packageContext: MainActivity.this, AgendaMascota.class);
                intent.putExtra( name: "mascota", animal.getNombre());
                startActivity(intent);
            }
            if (i == 2) {
                Intent intent = new Intent( packageContext: MainActivity.this, DescargarQR.class);
                intent.putExtra( name: "mascota", animal.getNombre());
                startActivity(intent);
            }
        }
    });/*Finaliza Menu Principal*/
}
```

Ilustración 55 Menú inferior circular

```
public void onClick(View v) {
    if (v == boton_informacion) {
        Intent intent = new Intent( packageContext: MainActivity.this, pantalla_informacion.class);
        startActivity(intent);
    }
    if (v == boton_mascota) {
        Intent intent = new Intent( packageContext: MainActivity.this, SeleccionarMascota.class);
        startActivity(intent);
    }
    if (v == boton_leerQR) {
        IntentIntegrator integrator = new IntentIntegrator( activity: MainActivity.this);
        integrator.setDesiredBarcodeFormats(IntentIntegrator.QR_CODE_TYPES);
        integrator.setPrompt("Escanear Codigo QR");
        integrator.setOrientationLocked(true);
        integrator.setCameraId(0);
        integrator.initiateScan();
    }
    if (v == boton_eliminar) {
        eliminar(animal.getNombre());
    }
}
```

Ilustración 56 Menú superior

3.10.4. Diagrama de contexto Sprint 10

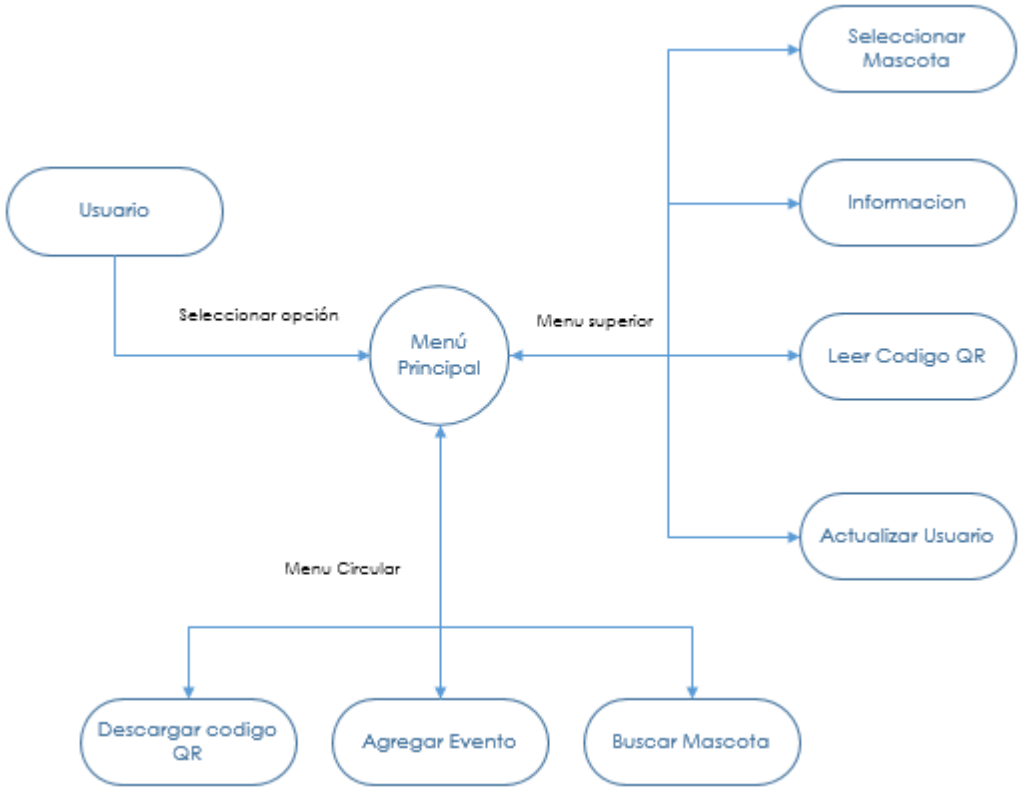


Ilustración 57 Diagrama de contexto Sprint 10

3.10.5. Prueba de Sprint 10

Tabla 39 Pruebas Backlock Sprint 10

Sprint 10	Menú principal	
Descripción de la prueba	Se usó la herramienta del entorno de desarrollo Android Studio, Android Device Monitor, que permitió verificar que cada botón llevara a una función determinada	
Herramientas	Android Device Monitor	
Prueba		**¿Cumple?**
1	Cada botón tiene la función de llevar a una interfaz determinada	Si

4. FASE 4 – IMPLEMENTACION

4.1. Diagrama de paquetes

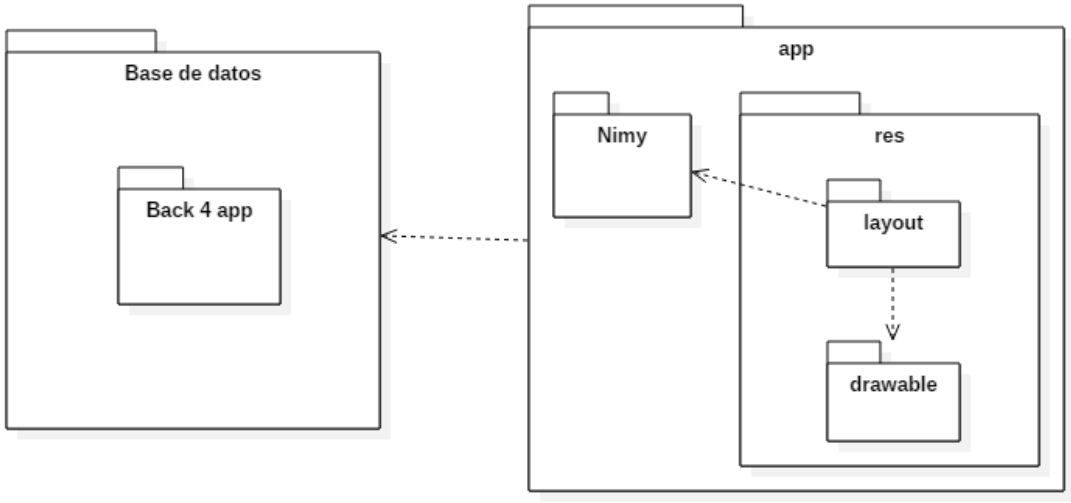


Ilustración 58 Diagrama de paquetes de la aplicación

4.2. Diagrama de componentes

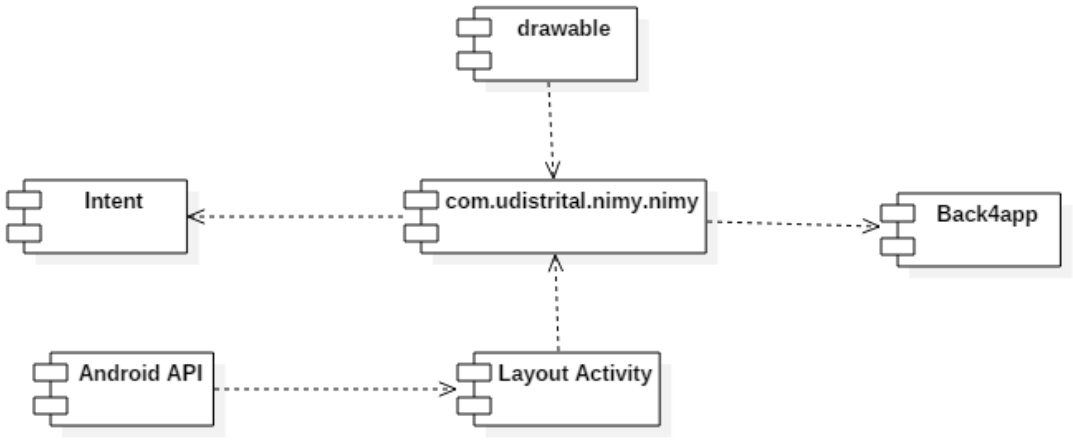


Ilustración 59 Diagrama de componentes de la aplicación

4.3. Pruebas en dispositivos

Se realizaron prueban en los celulares de los usuarios con mascotas que se ofrecieron a probar la aplicación con una pequeña introducción a ella, se hicieron preguntas para verificar el fácil uso de la aplicación, obteniendo los siguientes resultados.

Tabla 40 Dispositivo usuario 1

DISPOSITIVO	**CARACTERÍSTICA**
MOTOROLA E4 PLUS	• Procesador quad-core y velocidad 4G2 • 16GB de almacenamiento y 2GB RAM • Nougat 7.1.1 Android
PRUEBA	**RESULTADO**
REGISTRO DE LOGIN	Correcto
REGISTRO DE USUARIO	Correcto
REGISTRO DE MASCOTA	Correcto
SELECCIONAR MASCOTA	Correcto
DESCARGAR CODIGO QR	Correcto
LEER CODIGO QR	Correcto
BUSCAR MASCOTA	Correcto

Tabla 41 Dispositivo usuario 2

DISPOSITIVO	**CARACTERÍSTICA**
ALCATEL ONETOUCH 5020A	• Procesador MediaTek y velocidad 1Ghz • 4GB de almacenamiento y 512 MB RAM • Jelly-bean 4.1 Android
PRUEBA	**RESULTADO**
REGISTRO DE LOGIN	Correcto
REGISTRO DE USUARIO	Correcto
REGISTRO DE MASCOTA	Correcto
SELECCIONAR MASCOTA	Correcto
DESCARGAR CODIGO QR	Correcto
LEER CODIGO QR	Correcto
BUSCAR MASCOTA	Correcto

Tabla 42 Dispositivo usuario 3

DISPOSITIVO	**CARACTERÍSTICA**
SAMSUNG GALAXY J5	• Procesador Quad-Core y velocidad 1.2GHz • 8GB de almacenamiento y 1.5 GB RAM • Lollipop 5.1 Android
PRUEBA	**RESULTADO**
REGISTRO DE LOGIN	Correcto
REGISTRO DE USUARIO	Correcto
REGISTRO DE MASCOTA	Correcto
SELECCIONAR MASCOTA	Correcto
DESCARGAR CODIGO QR	Correcto
LEER CODIGO QR	Correcto
BUSCAR MASCOTA	Correcto

CONCLUSIONES

- El proyecto ha logrado cumplir todos los objetivos propuestos desde el inicio del proyecto.

- El Registro de los usuarios junto con sus mascotas es satisfactorio, asi permitiendo que la aplicación genere un código QR único con la información del usuario, la herramienta de backend back4app, permite que la información se almacene.

- El módulo de mapas del api de google services es una herramienta la cual permitió que el usuario pudiera actualizar su posición y mostrarla en un mapa permitiendo una interacción gráfica y sencilla de entender

- La implementación del backend back4app para dispositivos móviles fue una buena herramienta, brindó una forma rápida de desarrollo y mayor acaparamiento de información del usuario junto con el de su mascota con un bajo costo de memoria del dispositivo móvil.

- La agenda de eventos es funcional, genera recordatorios al usuario sobre los próximos eventos que su mascota tiene, esto gracias a la herramienta backend back4app, que permitió almacenar los eventos que realiza un usuario de su mascota.

RECOMENDACIONES

- Para poder utilizar los servicios de la aplicación es necesario el acceso a internet.

- Se debe colocar el código QR generado en la mascota en un lugar visible tal como el collar de identificación.

- Se recomienda imprimir el código QR generado por la aplicación de un tamaño mayor a 3cm X 3 cm

- Es importante implementar un usuario y contraseña que pueda recordar fácilmente debido la aplicación no cuenta con la opción de restablecimiento de contraseña.

- Para el uso del aplicativo se recomienda tener con un celular móvil o Tablet, mayor a 4.0” para su adecuada visualización y tener una API mayor al 22.

- Se pueden llegar a realizar muchas mejoras al aplicativo, modulo para notificaciones en tiempo real, crear una base de datos local que se encuentre sincronizada con la base de datos en la nube, y la opción del restablecimiento de contraseña.

BIBLOGRAFIA

- GIRONES, Tomas.El gran libro de Android. 3era Edicion. Marcombo. España. 2013. 594 pag

- HEBUTERNE, Sylvain. Android: guia de desarrollo de aplicaciones Java para smartphones. 3era Edicion.Ediciones ENI. España. 2016. 556 pag

- KHALDI, Jilani. SQLite. 2da Edicion.Italia. 2015. 121 pag

- MENZINSKY, Alexander. Scrum Manager. 2.6. Safe Creative. 2016. 94 pag

- LEQUERICA, Joan. Desarrollo de apps para Android. Ed 2016. España: Anaya multimedia. 2015. 432 p. ISBN 978-84-41-5368-07.

- CHO, James. Interfaz En: Desarrollo De Juegos Para Android. 2015. España.Anaya.2016. 464 p.

- STONEHEM, Bill. Google Android:Firebase. Estados Unidos. 2016. 40 pag

- RIBAS, Joan. Desarrollo de aplicaciones para Android. Ed 2017. España: Anaya. 2017. 463 p. ISBN 97-88-44-15-3809-2.

Anexos

Manual de uso

Nimy es una aplicación que te permite geo localizar a tu mascota por medio del uso del GPS y la cámara de tu dispositivo móvil, que será usada como un lector de códigos QR.

Te aparecerá esta pantalla de inicio la cual te permitirá ingresar a la aplicación



Sino estas registrado, la aplicación te permitirá registrar tus datos tale como un nombre para tu usuario, un número telefónico preferiblemente celular, un correo electrónico con el cual te podremos contactar y una contraseña.



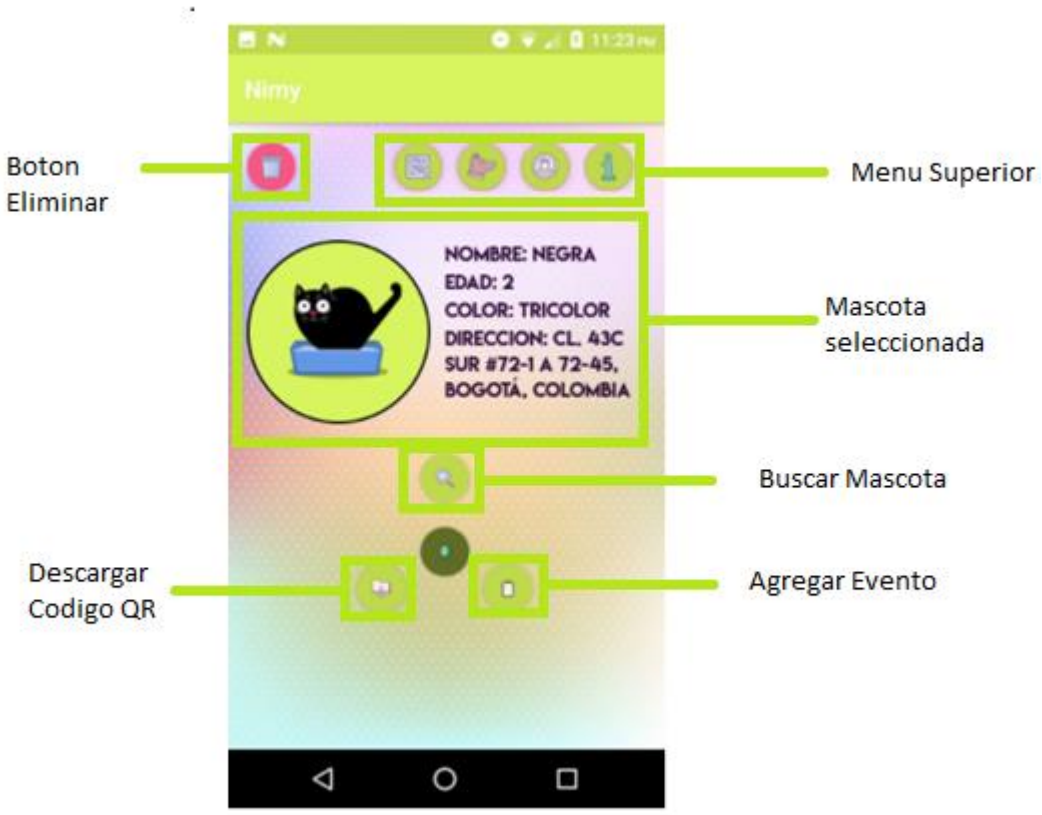
Cuando ingreses te aparecerá la pantalla principal que tendrá los principales botones con los cuales tu podrás acceder a todas las funcionalidades que te ofrece la aplicación, si tienes una mascota registrada te aparecerá aquí.



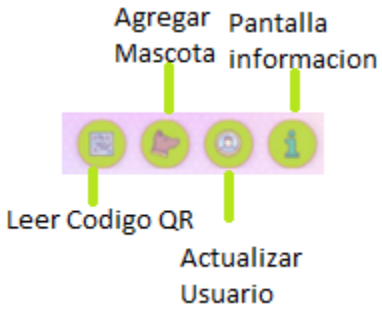
Caso contrario te aparecerá el mensaje de que no tienes mascotas registradas aun



El menú está conformado de la siguiente manera



Dentro del menú superior encuentras los botones de:



Agregar Mascota

El botón de agregar mascota te permite registra los datos de tu mascota, tales como el nombre, la fecha de nacimiento, la raza, el tipo de mascota, es decir si tu mascota es un gato o un perro y por último el sexo de tu mascota



Para continuar con el registro de tu mascota lo que debes hacer ahora es crear un perfil en donde te mostrara la información de tu mascota, puedes seleccionar una imagen de tu galería, tomarle una foto a tu mascota o bien seleccionar de nuestra galería la imagen que más te guste



Presionas el botón de crear perfil y listo, tu mascota ha quedado registrada.

Si deseas agregar otra mascota solo basta con presionar el botón agregar

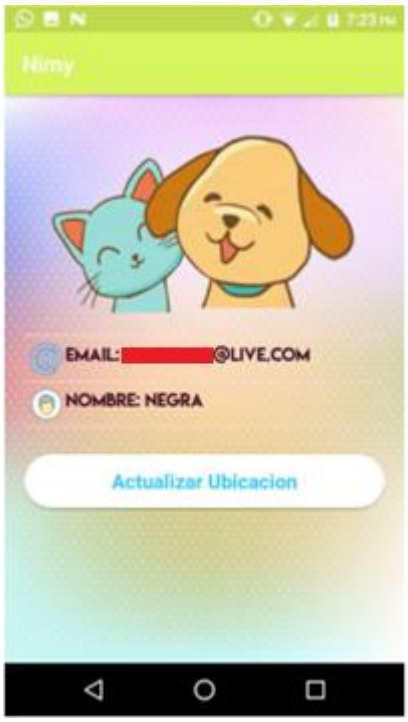


Si presionas un perfil de mascota, quedará este seleccionado y por ende aparecerá en el menú principal y podrás agregar eventos o buscar tu mascota si la llegaras a perder.

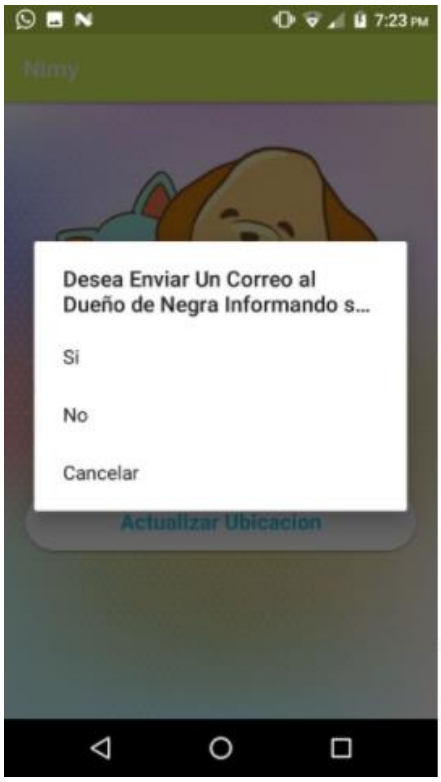
NOTA: El código QR se genera automáticamente con la información de tu mascota y del usuario que la registro.

Leer código QR

Este botón úsalo cuando encuentres una mascota, cuando lo presiones tu cámara será usada para escanear el código, cuando lo hagas te aparecerá algo como esto.



La información que se muestra está relacionada con la de la persona que registro a la mascota que encontraste. Cuando presiones el botón de **actualizar ubicación**, Nimy te permite comunicarte con el dueño por medio de un correo electrónico solamente tú tienes que autorizar si deseas enviar el correo o no.


7:23 PM
Nimy
Desea Enviar Un Correo al Dueño de Negra Informando s...
Si
No
Cancelar
Actualizar Ubicacion


7:23 PM
Redactar
De
gimenard555@gmail.com
Para
[redacted]@live.com
Nimy Actualizacion de su Mascota
He encontrado su mascota y ya he actualizado su posicion, actualmente su mascota se encuentra en Cl. 43c Sur #72-1 a 72-45, Bogotá, Colombia

Actualizar Usuario

Esta opción te permite actualizar los datos del usuario, te carga de nuevo la pantalla de registro de usuario.

NOTA: Si deseas eliminar el usuario, solo basta que presiones el icono de Eliminar.

Botón Eliminar

Úsalo cuando quieras eliminar un usuario o un perfil de mascota seleccionado.

Pantalla información

Este botón te permite ver y consultar las funciones de cada botón, úsalo cuando no recuerdes la función de algún botón

Buscar mascota

Este botón te muestra por medio del GPS de tu dispositivo móvil los lugares donde ha estado tu mascota, te indicamos el lugar donde se encuentra tu mascota cuando la hayan encontrado, actualizamos su posición en el mapa



Descargar Código QR

Este botón te permite descargar tu código QR, este se ha generado cuando tu registras los datos de tu mascota y creas el perfil, además tiene tus datos de usuario para que en caso de que la pierdas, se puedan comunicar contigo, para ello debes imprimir el código y colocarlo preferiblemente en el collar de tu mascota.



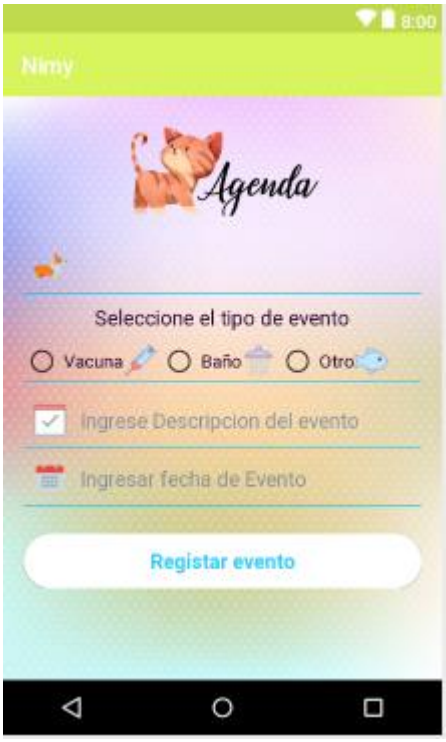
NOTA: Cuando vayas a imprimir tu código QR, te recomendamos que uses el tamaño de 3 cm x 3 cm.

Agregar Evento

Con este botón puedes realizar recordatorios sobre eventos en donde esté relacionado tú y tu mascota, el día del evento Nimy te genera una notificación para que veas que tienes un evento ese día. Te mostrara primero un calendario donde estarán las fechas que tengas.



Cuando quieras agregar un evento, presiona el botón agregar evento, este te mostrara otra pantalla en donde agregaras el evento, solamente debes llenar los datos tales como, el tipo de evento, la descripción del evento y la fecha.



Encuesta Realizada

- ¿Qué tipo de mascota tiene? *

Perro_ Gato_ Hámster_ Pez_ Otro_

- ¿En algún momento de su vida a perdido a una mascota? *

Sí_ No_

- ¿Qué tanta importancia le da a las vacunas de su mascota? *

Poca_ Medio_ Lo necesario_ Mucho_

- ¿Tiene su mascota un control médico? *

Sí_ No_ Tal vez_

- ¿Ha pensado en algún momento adoptar mascota? *

Sí_ No_ Tal vez_

- ¿Con que frecuencia recuerda las vacunas? *

Siempre_ Usualmente_ A veces_ Ocasionalmente_ Nunca_

- ¿Considera usted que es una buena idea tener una aplicación móvil que además de geolocalizar su mascota, le ayude a tener una agenda de su mascota con el control médico de vacunas? *

Sí_ No_ Tal vez_

- ¿Qué medios utiliza cuando pierde a su mascota? *

Redes Sociales_ Autoridades_ Publicar Carteles_ Nunca ha perdido a su mascota_

- ¿Tiene conocimiento de los centros más cercanos para la atención de su mascota? *

Sí_ No_ Tal vez_

- ¿Ha encontrado usted alguna mascota ajena en la calle? *

Sí_ No_ Tal vez_

- ¿Qué hace usted cuando encuentra una mascota con collar en la calle perdida? *

Informar a las autoridades_ Publicar en la Redes Sociales_ Publicar en carteles_ No hace nada_ No me ha pasado_

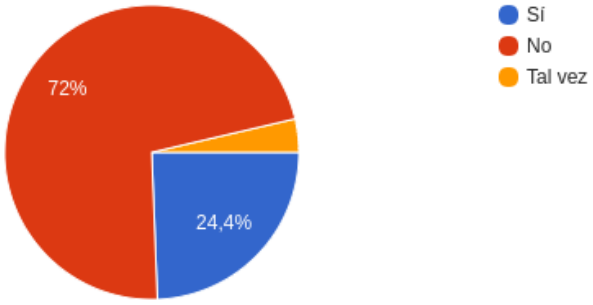
- ¿Que esperaría encontrar en una aplicación para mascotas?

_____
_____
_____
___

Resultados de la encuesta realizada

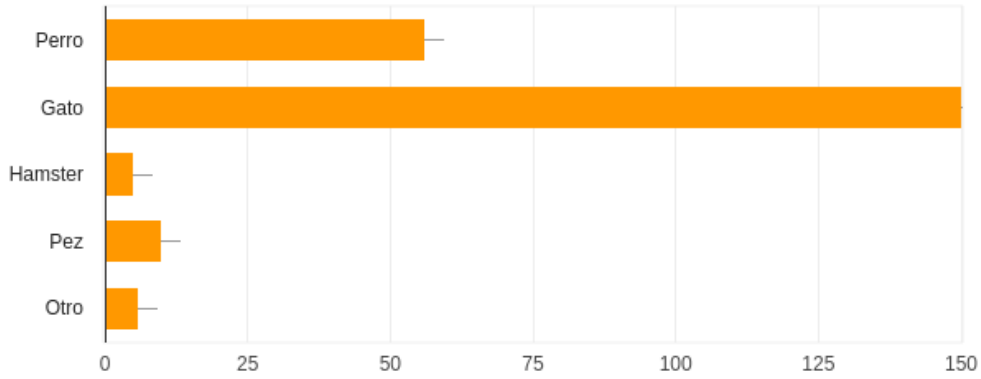
¿Conoce usted aplicaciones para mascotas?

164 respuestas



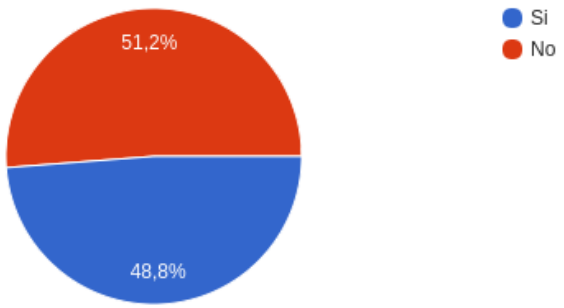
¿Que tipo de mascota tiene?

164 respuestas



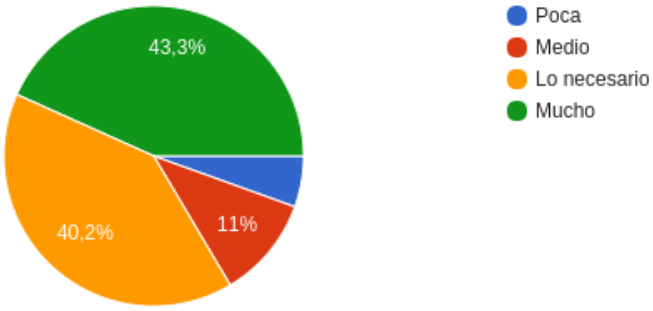
¿En algún momento de su vida a perdido a una mascota?

164 respuestas



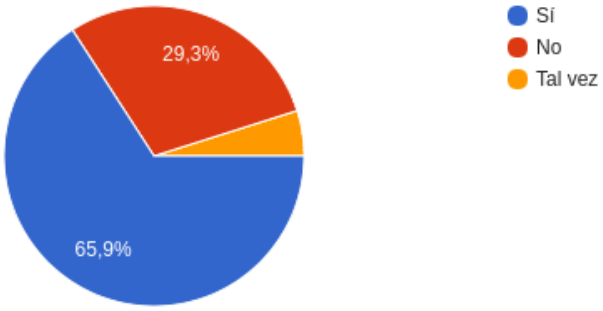
¿Que tanta importancia le da a las vacunas de su mascota?

164 respuestas



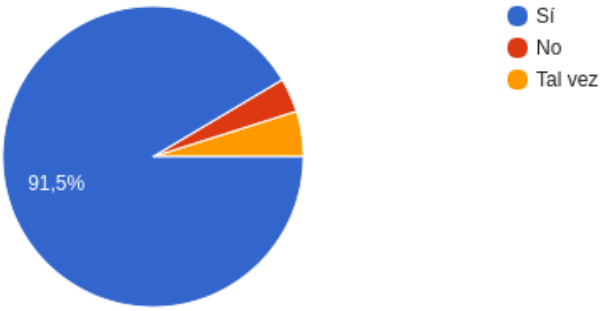
¿Tiene su mascota un control medico?

164 respuestas



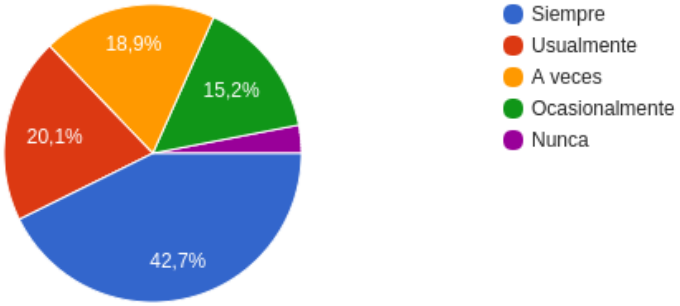
¿Ha pensado en algún momento adoptar mascota?

164 respuestas



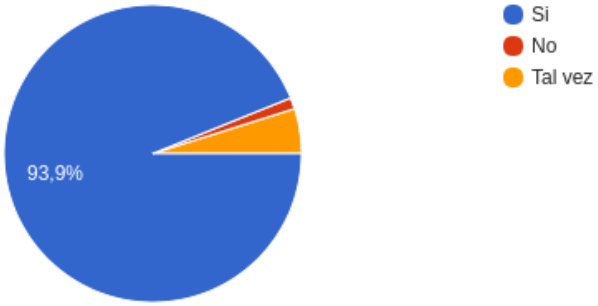
¿Con que frecuencia recuerda las vacunas?

164 respuestas



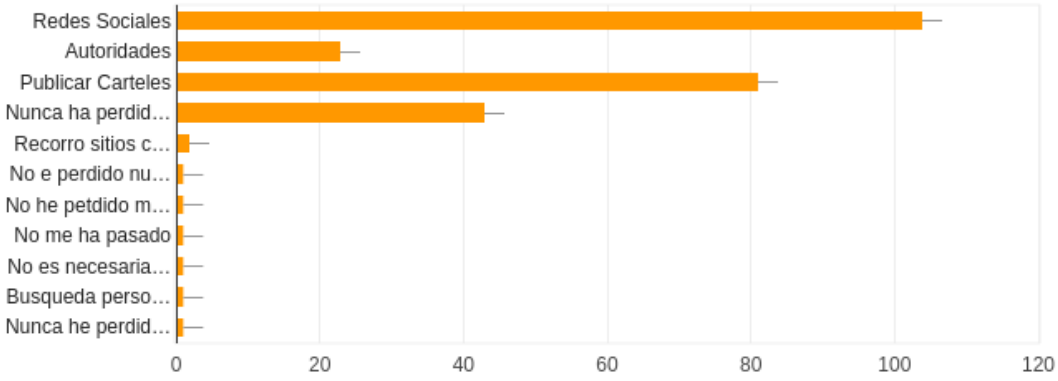
¿Considera usted que es una buena idea tener una aplicación móvil que ademas de geolocalizar su mascota, le...ta con el control medico de vacunas?

164 respuestas



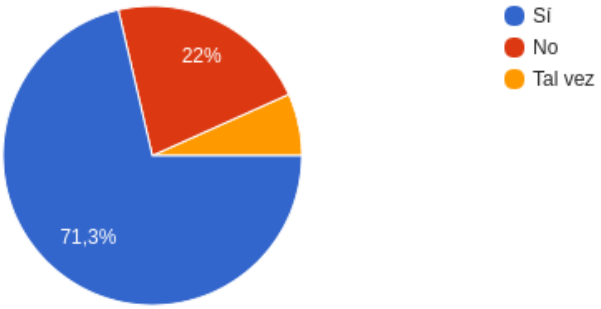
¿Que medios utiliza cuando pierde a su mascota?

164 respuestas



¿Tiene conocimiento de los centros mas cercanos para la atención de su mascota?

164 respuestas



¿Que hace usted cuando encuentra una mascota con collar en la calle perdida?

164 respuestas

