UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

INGENIERO EN SISTEMAS INFORMÁTICOS

TEMA:

DESARROLLO DE UN SISTEMA DE GESTIÓN SECTORIZADO DE ADOPCIÓN DE MASCOTAS PARA LA FUNDACIÓN CAMINO A CASA

AUTOR:
MARCELO IVAN TOAPANTA GUALPA

TUTOR:
MSc. Ing. PABLO M RECALDE V

QUITO, ECUADOR

2019

DECLARACIÓN DE AUTORÍA

El documento de tesis con título: "DESARROLLO DE UN SISTEMA DE GESTIÓN SECTORIZADO DE ADOPCIÓN DE MASCOTAS PARA LA FUNDACIÓN CAMINO A CASA", ha sido desarrollado por el señor Marcelo Iván Toapanta Gualpa con C.C. No. 1721050290 persona que posee los derechos de autoría y responsabilidad, restringiéndose la copia o utilización de la información de esta tesis sin previa autorización.

_____

MARCELO IVAN TOAPANTA GUALPA

UNIVERSIDAD TECNOLÓGICA ISRAEL

APROBACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Titulación certifico:

Que el trabajo de titulación **"DESARROLLO DE UN SISTEMA DE GESTIÓN SECTORIZADO DE ADOPCIÓN DE MASCOTAS PARA LA FUNDACIÓN CAMINO A CASA"**, presentado por MARCELO IVAN TOAPANTA GUALPA, estudiante de la Carrera Ingeniería en Sistemas Informáticos, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado, que se designe, para su correspondiente estudio y calificación.

Quito D. M., agosto 2019

TUTOR

_____

MSc. Ing. Pablo M Recalde V.

AGRADECIMIENTOS

Agradezco este trabajo a mi familia, por su apoyo en cada etapa alcanzada satisfactoriamente.

A la Universidad Israel por la apertura a los estudiantes y poder cumplir un objetivo más en nuestra vida profesional.

DEDICATORIA

Dedico este trabajo a mi familia por enseñarme que con constancia es posible alcanzar nuestros objetivos planeados.

A mi esposa Verónica Mena y mi hija Lida Toapanta por todo el apoyo en esta etapa estudiantil.

TABLA DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

El presente trabajo consiste en el "DESARROLLO DE UN SISTEMA DE GESTIÓN SECTORIZADO DE ADOPCIÓN DE MASCOTAS PARA LA FUNDACIÓN CAMINO A CASA" el cual fue elaborado para plataforma web con el fin de optimizar el proceso de adopciones y rescates de mascotas para la ciudad de Quito.

La metodología utilizada para este proyecto fue *Scrum*, la cual busca obtener el mejor resultado posible, utilizando un conjunto de buenas prácticas orientadas al trabajo en equipo el entregó al cliente avances parciales con el objetivo de resolver situaciones que no se cumplan con las necesidades del cliente, reduciendo costos y logrando una calidad aceptable.

El sistema está conformado por los siguientes módulos: Usuarios, Anuncios, Mensajes, Panel sectorizado, Configuraciones. Con el fin de gestionar y controlar el contenido de los anuncios existen tres roles de usuarios: Administrador que podrá gestionar la información de los anuncios si restricciones, Aprobador el cual está encargado de revisar, aprobar y publicar los anuncios del sistema, por último, el Anunciante el cual es un usuario que puede crear anuncios una vez que se haya registrado en el sistema.

Con el desarrollo del sistema se pretende mejorar el control de las adopciones y rescates de la fundación, además de poner a disposición los anuncios para todos los seguidores de la fundación de forma sectorizada del D.M. de Quito.

PALABRAS CLAVES: Desarrollo, Ruby, Rails, *Scrum*, Automatización, Control, Sectorización, Mascotas, Adopción, Rescate.

ABSTRACT

The present work consists in "DEVELOPMENT A SECTORIZED ADVOCACY MANAGEMENT SYSTEM FOR PETS FOR CAMINO A CASA FOUNDATION" which was developed for a web platform in order to optimize the pet adoptions and rescues process for the city of Quito.

Development methodology used was *Scrum*, which seeks to obtain the best possible result, using a set of good practices oriented to teamwork, delivering partial advances to the client in order to solve situations that do not meet the client's needs. Reducing costs and achieving an acceptable quality.

The system is made up of the following modules: Users, Announcements, Messages, Sectored Dashboard, and Configurations. In order to manage and control the content of the ads, there are three user roles: System administrator who can manage all the system information, Approver which is in charge of reviewing, approving and publishing the system links, finally the Advertiser which is a user who can create ads once they have registered in the system.

The development of the system aims to improve the control of adoptions and rescues of the foundation, in addition to making available the announcements for all the followers of the foundation in a sectored way in the city of Quito.

KEY WORDS: Development, Ruby, Rails, *Scrum*, Automation, Control, Pets, Adoption, Rescue.

INTRODUCCIÓN

Antecedentes de la situación objeto de estudio

Fundación Camino a Casa funciona en el D.M. de Quito desde 28 de junio del 2011 la cual se encarga de alimentar y curar a animales domésticos callejeros hasta que se encuentre personas que puedan encargarse de los mismos, fomentando siempre una adopción responsable.

Se realizó una entrevista con la propietaria de la fundación en la misma se puso en conocimiento la situación de ese momento del proceso de rescate y adopción gestionados por la fundación. Un rescate de una mascota consiste en solucionar los problemas como heridas, traumas, desnutrición y abandono, el cual es gestionado por la fundación mediante dos vías: reporte de sus seguidores de Facebook y llamadas directas al centro, la fundación cuenta con recursos limitados por tal razón se restringe la cantidad de casos gestionados y se prioriza los más graves.

Una adopción se realiza cuando la mascota rescatada se ha recuperado de su mal estado, este proceso inicia con la publicación en su cuenta de Facebook, después el interesado establece comunicación con la fundación, se estable los términos y compromisos de la adopción y finalmente se hace entrega de la mascota.

Planteamiento del problema

A nivel nacional la tenencia responsable de mascotas se ha convertido en un gran problema, tampoco el Ministerio de Salud Pública juntamente con el Municipio de Quito ha podido concienciar a la población en general sobre el significado de tenencia responsable.

A pesar de ello, el ver mascotas abandonadas a su suerte ya es el común de todos los días, es por ello que mediante el Desarrollo del Sistema de Gestión para la Fundación Camino a casa, entidad sin fines de lucro se apoya a la tenencia responsable de mascotas (Castellano, 2018) .

Fundación Camino a Casa tiene Facebook, sin embargo el problema que se encuentra cuando se trata de adoptar mascotas, es el desorden y el tiempo que toma para la adopción, ya que, por este medio digital no se lleva la cuenta de adopciones por categorías, como la raza, el lugar de procedencia, la edad, no se puede identificar los anuncios con adopciones exitosas, no se lleva un historial de conversaciones con las personas interesadas, no se puede hacer un filtrado ni búsqueda por nombre y tampoco tienen una forma para mostrar las adopciones exitosas.

Justificación

Este proyecto busca incentivar a la comunidad del D.M. de Quito a realizar más adopciones y rescates de mascotas callejeras y evitar la compra/venta de las mismas. El tiempo de adopción se reducirá con la implementación del sistema, debido a que la información de las mascotas en adopción estará disponible en la web en cualquier momento, de forma ordenada, categorizada y segura, esto representará un impacto positivo hacia la comunidad que desee adoptar a una mascota.

Objetivos

Los objetivos de un proyecto de tesis sirven para exponer como se piensa abordar la problemática de la investigación.

General

Optimizar el proceso de adopción y rescate de mascotas para la fundación camino a casa mediante el desarrollo de un Sistema de Gestión sectorizado de anuncios para adopción y rescate de mascotas, utilizando el lenguaje Ruby y el *framework* Ruby on Rails y como base de datos el SGBD PostgreSQL.

Objetivos específicos

- Desarrollar los módulos del sistema necesarios, dentro de los cuales están: usuarios, anuncios, mensajes, panel sectorizado.
- Implementar un blog dentro del sistema, donde se pueda registrar las historias de adopciones.
- Desplegar un panel de adopciones por sector dentro del D.M. de Quito.

Descripción de los capítulos

En el capítulo 1. Fundamentación teórica; En este capítulo se resume la información más importante y relevante como normas, conceptos, artículos e informes que ayudaron al desarrollo del sistema.

En el capítulo 2. Marco metodológico; Se describe la metodología que se aplicó durante el desarrollo del sistema, se realizó una investigación con un enfoque cuantitativo debido a que el principal problema de la fundación es el tiempo que toma el proceso actual en realizar una adopción exitosa, con un estudio experimental en el cual se recolecto la información mediante una entrevista a la dueña de la fundación y los comentarios de la cuenta de Facebook de la fundación.

En el capítulo 3. Propuesta; Se propone optimizar el proceso actual de rescate y adopción de mascotas para reducir el tiempo de adopción, mediante el ordenamiento sistemático de los procesos y actividades, entradas y salidas de los procesos y la definición de roles, además mediante herramientas no propietarias disponibles en el mercado para el desarrollo de un sistema de plataforma web para la automatización de dicho proceso.

El capítulo 4. Implementación; Refleja el uso de *Scrum* como metodología de desarrollo de software, además se generó el *Sprint Backlog*, cronograma de cada *Sprint* y sus entregables, el diagrama de base de datos, diseño de interfaces, los estándares de programación, la arquitectura y requisitos del sistema y por último las pruebas realizadas por parte del equipo de desarrollo.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

En esta sección se evidencia las normas, conceptos, artículos e informes que ayudaron al desarrollo del sistema.

1.1 Estado del arte

En Ecuador, en 2017, se aprobó el Código Orgánico del Ambiente, que da disposiciones generales para el manejo responsable de la fauna urbana. En el artículo 139 señala que uno de los objetivos es erradicar la violencia contra los animales, fomentar un trato adecuado para evitarles sufrimientos innecesarios y prevenir su maltrato: “La tenencia de animales conlleva la responsabilidad de velar por su bienestar, y su manejo deberá promover una relación armoniosa con los seres humanos” (Monroy, 2018).

El Código Integral Penal (COIP) sanciona en sus Art.249 y 250 el maltrato animal con cincuenta horas de trabajo comunitario y prisión desde siete hasta treinta días, de acuerdo al daño causado. Sin embargo, con la ley en los papeles todavía no existe la debida concientización en la población ecuatoriana, seguimos viendo animales violentados, abandonados a su suerte, falta que las demás entidades Públicas como el Ministerio de Salud Pública, el Ministerio de Educación entre otras, se reúnan y lleguen con acuerdos y soluciones para este problema latente.

Al crear este Sistema se busca sistematizar las adopciones, pero también crear un pensamiento positivo hacia las mismas.

1.2 Lógica del negocio

El programa de adopciones de la fundación permite que decenas de perros y gatos callejeros, encuentren familias que les brinden amor y protección, una vez finalizado su proceso de recuperación.

El proceso de recuperación consiste analizar las denuncias de mascotas pérdidas o en mal estado reportado por los seguidores si la administración considera viable de la fundación, si la administración considera viable el rescate de la mascota la fundación procede a recoger y curar a la mascota, cuando el proceso termina la fundación realiza un anuncio de adopción.

El proceso de curación en algunos casos consiste en desparasitar y limpiar mascotas, sin embargo, existen casos más complejos donde la mascota necesita intervención médica y reposo.

1.3 Herramientas técnicas

En esta sección se detalla las herramientas que fueron utilizadas por el equipo de trabajo para cumplir con los objetivos planteados en este proyecto, todas las herramientas que se describen a continuación son de uso libre y están disponibles en Internet.

Lenguaje de programación

Se denomina como leguaje de programación al conjunto de instrucciones que permiten crear programas de cómputo, para describir procesos.

Entre los principales lenguajes de programación se tiene a Ruby, que es un lenguaje que incorpora tanto programación funcional como imperativa. Este tipo de lenguaje de programación fue liberado en 1995. Ruby on Rails la primera es un framework de aplicaciones web de código abierto caracterizado por su brevedad a la hora de escribir código, por seguir un sólido patrón MVC. (Moreno, 2003)

Framework Ruby on Rails

Ruby on Rails (*RoR*) integra componentes que permiten adicionar funcionalidad, conocidas como gemas "librería en Ruby". Para este proyecto se escogió a Ruby como lenguaje de programación sobre otros que tienen gran acogida en el mercado ecuatoriano como *Java* o *PHP* por las siguientes razones:

- Facilidad de aprendizaje
- Código de escritura simple
- Reutilizable
- Desarrollo rápido (lenguaje interpretado y no compilado)
- Vigencia en el mercado
- Costos menores de inversión
- Experiencia amplia en el mercado

RoR es un entorno flexible y adaptable ya que es capaz de integrar componentes y funcionalidades con la instalación de librerías denominadas como gemas (Gems). (Hartl, 2016)

RoR implementa el paradigma de desarrollo MVC la cual separa al sistema en las siguientes entidades:

Modelo: Mantiene el estado de la aplicación, si es transitorio será almacenado en memoria, si es permanente en una base de datos, el almacenamiento usa las reglas del negocio.

Vista: Responsable de generar las interfaces que interactúan con el usuario basado en los datos de un modelo.

Controlador: Reciben datos desde el medio externo a la aplicación, interactúan con el modelo y presentan los datos en una vista apropiada para el usuario.

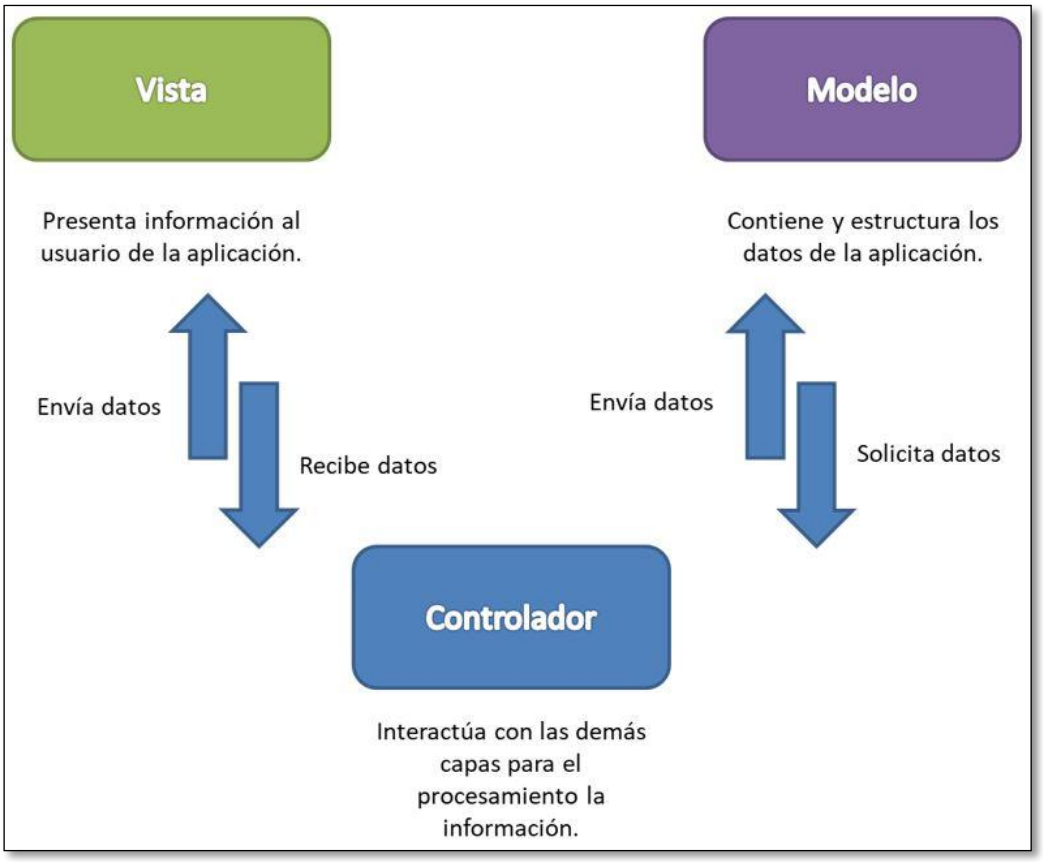


Figura 1.1. Estructura del modelo, vista, controlador (Perez, 2018).

En la Figura 1.1 se observa el flujo de comunicación entre el modelo, la vista y el controlador, de tal forma se puede entender como el sistema es capaz de recibir una petición, procesarla y retornar una vista como resultado visible al cliente.

Sistema de gestión de bases de datos *PostgreSQL*

Para el almacenamiento de los datos fue necesario el uso de un sistema de base de datos, en el mercado existen varias opciones, sin embargo todas cumplen con principios fundamentales con el uso del álgebra relacional para la manipulación de datos.

El presente proyecto hizo uso de *PostgreSQL* para gestionar el almacenamiento de los datos generados, al ser una herramienta de libre acceso y de flexible configuración se integra correctamente con el framework *Ruby on Rails*. (The Group PostgreSQL Global Development, 2019)

Acorde la página web oficial de PostgreSQL se detalla a continuación las principales características:

- Es una base de datos 100% ACID
- Integridad referencial
- Transacciones anidadas
- Copias de seguridad en caliente
- Caracteres internacionales
- Métodos de autentificación
- Acceso SSL con encriptado
- Amplia documentación
- Compatible con Linux, UNIX y Windows 32/64bit

Lenguaje de marcado de hipertexto

HTML (*Hyper Text Markup Language*) es un lenguaje de marcado de hipertexto utilizado en el lado del cliente, los bloques serán interpretados por un navegador web y describe el contenido de una página web.

Figura 1.2 Estructura HTML (Duckett, 2011).

Hojas de estilo en cascada

Para la forma de los elementos HTML se usó CSS (*Cascading Style Sheets*), las cuales son hojas de estilo en cascada, mediante reglas de estilos, entendiendo que casada es el hecho que se visualizará la última regla a las anteriores. (Duckett, 2011)

Librería *JavaScript*

JavaScript y *JQuery* fueron los encargados de permitir que el contenido sea dinámico e intuitivo para el usuario, el uso de esta tecnología ha incrementado potencialmente en los últimos años por su flexibilidad, sencillez y potencia. (Chaffe & Swedberg, 2013)

1.4 Alternativas de solución

Existen fundaciones a beneficio de mascotas en D.M. Quito con problemas y soluciones similares al del cliente, dentro de las soluciones se tienen las siguientes:

Mi Mascota: Sitio web en Perú disponible para al público en general, sin embargo no se ajusta a la necesidad del cliente debido a que las operaciones del mismo no abarca otros países.

Voluntarios: En fundaciones grandes existen programas de voluntarios los cuales gestionan las adopciones y rescates personalmente sin intenciones de lucro.

Patrocinio de empresa privada: Algunas fundaciones cuentan con el patrocinio de empresas privada las cuales gestionan los gatos de recuperación y las adopciones de las mascotas.

Las soluciones listadas pueden solucionar el problema de la fundación, sin embargo, es difícil llegar a acuerdos con personas externas o alcanzar un patrocinio esto debido a que no existe una cultura de cuidado de mascotas en el país, además dichas soluciones utiliza grandes cantidades de dinero.

CAPÍTULO 2. MARCO METODOLÓGICO

El presente proyecto se enmarca en un enfoque cuantitativo ya que se pretende aumentar los rescates y adopciones de la fundación con la optimización del proceso.

2.1 Tipo de investigación

En la etapa de fundamentación teórica el proyecto se enmarcó dentro del método deductivo, ya que se fundamentó en el Código Orgánico del Ambiente y el Código Integral Penal (COIP) que contienen disposiciones generales para el manejo responsable de la fauna urbana, específicamente en el artículo 139 sobre la tenencia responsable de mascotas ver Anexo 1 y el artículo 249 y 250 del COIP sobre las sanciones para el maltrato animal ver Anexo 2.

Mediante una entrevista, la dueña expuso la situación de la fundación, se pudo identificar los problemas que presenta el proceso actual, se identificaron los roles y actividades que intervienen en este proceso.

2.1.1 Metodología seleccionada

Método deductivo

Se parte de aspectos generales para llegar a los específicos, con el cual fue de ayuda para la recopilación de la información general para poder plantear el problema del proyecto utilizando técnicas como la encuesta y la entrevista.

Método inductivo

El método inductivo propone obtener conclusiones generales a partir de premisas particulares, las pruebas realizadas por el equipo de trabajo se puede observar en los Anexo 8 y Anexo 9 con las cuales se comprobó que la aplicación desarrollada cumpla con los requerimientos planteados en la fase de planeación, permitiendo corregir errores en la etapa de pruebas (Gutiérrez, 2006).

2.2 Recopilación de información

Esta sección describe los métodos de utilizados para recolección de información.

2.2.1 Técnicas de recopilación de información

Entrevista

Posibilita la recuperación de información por medio del diálogo inmediato entre el profesional que efectúa la función científico investigativo y los individuos que son fuentes de información. En esta constatación particular el entrevistador tiene la posibilidad de profundizar en los criterios, intereses, estimaciones. La entrevista se puede hacer a una persona o en grupo, en todos los sucesos el científico debe realizar un plan que detalle los aspectos a tratarse. La exigencia del plan puede llegar a un punto de elaborar todas las consultas que serán realizadas en el diálogo (Sabariego, Dorio, & Massot, 2004).

Se realizó una entrevista inicial con la directora de la fundación la Sra. Cristina Calderón, en la cual manifestó los problemas con el proceso de rescate y adopción de mascotas en el D.M. Quito, la cual se detalla en el Anexo 3. La propietaria manifestó que estaba interesada en adquirir un software que le ayude a realizar la gestión de adopciones de mascotas entre otros requerimientos, ya que en la actualidad este proceso se realiza de manera no automatizada sin tener un control eficaz. Además, se logró recopilar información representativa para el desarrollo del sistema informático como cuáles son las mascotas más recurrentes en la fundación y requisitos para poder realizar la adopción.

Encuesta

Se realizó una encuesta a los seguidores de la fundación en Facebook los cuales al momento de redactar el informe son 16.000, con el fin de recopilar información para la propuesta a la problemática presentada, las preguntas incluyen aspectos sociales, económicos y tecnológicos. La encuesta tuvo una participación de 1.624 seguidores, para observar las preguntas realizadas en la encuesta ver Anexo 4.

Tabulación de la encuesta.

La encuesta permite cuantificar la posibilidad de aceptación o rechazo de los involucrados en el proceso. Los datos recopilados de la encuesta deben ser procesados y analizados para ser aplicados en la solución al problema.

La encuesta estuvo formada por nueve preguntas, para este caso se tomó una muestra de 1624 seguidores de la fundación en Facebook y el análisis de cada respuesta se muestra continuación.

Tabla 2.1. ***Encuesta, Pregunta 1***

Pregunta Nro. 1.- ¿Si usted busca adoptar una mascota, prefiere la opción de contactar directamente con la persona que sede la mascota o acudir a la fundación?

Respuesta	**Valor**	**Porcentaje**
Persona	1221	75.18 %
Fundación	403	24.82 %
Total	1624	100 %

Fuente: Elaboración propia

Las repuestas para la primera pregunta muestran que la mayoría de los seguidores prefieren la opción de contactar directamente con la persona que sede la mascota en adopción lo que establece que la solución debe contar con un mecanismo de comunicación directa entre los seguidores de la fundación, ver Tabla 2.1.

Tabla 2.2. ***Encuesta, Pregunta 2***

Pregunta Nro. 2.- ¿Cómo se entera de las publicaciones de la fundación: Facebook, boca a boca, ¿recomendación?

Respuesta	**Valor**	**Porcentaje**
Facebook	856	52.71 %
Boca a boca	461	28.39 %
Recomendación	307	18.90 %
Total	1624	100 %

Fuente: Elaboración propia

Como se observa en la

Tabla **2.2** los seguidores de la fundación en su mayoría se enteran de las publicaciones que hace la fundación por Facebook, y en menor cantidad de boca a boca y por recomendaciones, por tal razón la solución debe integrar un mecanismo para publicar en Facebook.

Tabla 2.3. ***Encuesta, Pregunta 3***

Pregunta Nro. 3.- ¿Cómo prefiere que se ordenen las publicaciones de la fundación: Fecha, tipo de mascota, sector, ¿nombre de la mascota?

Respuesta	**Valor**	**Porcentaje**
Sector	1003	61.76 %
Nombre de mascota	298	18.35 %
Fecha	271	16.69
Tipo de mascota	52	3.20 %
Total	1624	100 %

Fuente: Elaboración propia

La preferencia de los seguidores para ordenar las publicaciones de la fundación es por sector, lo que quiere decir que la solución de integrar la funcionalidad de mostrar las publicaciones por sector, por tal razón se debe restringir el área donde se aplicará la solución, también se puede observar que para los seguidores es importante en menor medida identificar las publicaciones por fecha y nombre de la mascota, ver Tabla 2.3.

Tabla 2.4. ***Encuesta, Pregunta 4***

Pregunta Nro. 4.- ¿Para reportar el rescate de una mascota prefiere hacerlo por redes sociales o contratar directo con la fundación?

Respuesta	**Valor**	**Porcentaje**
Redes sociales	1387	85.41 %
Fundación	237	14.59 %
Total	1624	100 %

Fuente: Elaboración propia

Con respecto a los rescates los seguidores de la fundación en la Tabla **2.4** se puede observar que en su mayoría prefieren hacerlo por redes sociales que contactar con la fundación, lo que establece que las publicaciones de los rescates deben poder ser compartidos en Facebook.

Tabla 2.5. ***Encuesta, Pregunta 5***

Pregunta Nro. 5.- ¿Por qué medio prefiere aportar con donaciones a la fundación: transacción bancaria, efectivo, PayPal, ¿donación de alimentos?

Respuesta	**Valor**	**Porcentaje**
Transacción bancaria	684	42.12 %
PayPal	612	37.68 %
Donación de alimentos	235	14.47 %
Efectivo	93	5.73 %
Total	1624	100 %

Fuente: Elaboración propia

Las donaciones permiten el funcionamiento de la fundación, por tal razón es necesario aumentar la publicidad a cerca de los métodos por los cuales los seguidores pueden aportar a la fundación, la Tabla 2.5 muestra que los métodos más aceptados por los seguidores son las transacciones bancarias y PayPal, dejando como las menos aceptadas a la donación de alimentos y el efectivo. Por tal razón es necesario dar a conocer los métodos más aceptados por los seguidores.

Tabla 2.6. ***Encuesta, Pregunta 6***

Pregunta Nro. 6.- ¿Tiene acceso a Internet en su casa?

Respuesta	Valor	Porcentaje
Si	1482	91.26 %
No	142	8.74 %
Total	1624	100 %

Fuente: Elaboración propia

En la Tabla 2.6 se observa que la mayoría de los seguidores tienen acceso a Internet en su casa, por lo que es viable una solución para plataforma web, lo cual establece que el proceso debe ser automatizado para cumplir con la optimización.

Tabla 2.7. ***Encuesta, Pregunta 7***

Pregunta Nro. 7.- ¿Tiene acceso a Internet desde su teléfono celular?

Respuesta	Valor	Porcentaje
Si	842	51.85 %
No	782	48.15 %
Total	1624	100 %

Fuente: Elaboración propia

En la Tabla 2.7 se observa que un más de la mitad de los seguidores tienen acceso a desde dispositivos móviles.

Tabla 2.8. ***Encuesta, Pregunta 8***

Pregunta Nro. 8.- ¿Cuál es el navegador de internet más usado por usted: Firefox, Chrome, Microsoft Edge, Safari?

Respuesta	Valor	Porcentaje
Chrome	933	57.45 %
Firefox	367	22.60 %
Edge	236	14.53 %
Safari	88	5.42 %
Total	1624	100 %

Fuente: Elaboración propia

Debido a que la fundación tiene gran acogida en la red social Facebook, la solución deber ser orientada a la web y optimizada para el o los navegadores que los seguidores usen con mayor frecuencia. Chrome y Firefox son los navegadores más usados como se muestra en la Tabla 2.8, por tal razón las soluciones web deben ser optimizadas para los mismos.

Tabla 2.9. ***Encuesta, Pregunta 9***

Pregunta Nro. 9.- ¿En una publicación de la fundación en cuál de los siguientes elementos presta más su atención, nombre, foto, descripción, ubicación?

Respuesta	**Valor**	**Porcentaje**
Foto	1020	62.81 %
Ubicación	402	24.75 %
Descripción	153	9.42 %
Nombre	49	3.02 %
Total	1624	100 %

Fuente: Elaboración propia

En la Tabla 2.9 se observa que los seguidores dedican más su atención en las fotos y la ubicación de las publicaciones por lo tanto la solución debe enfocarse en mostrar mayormente estos elementos para tener una mayor aceptación de los seguidores.

CAPÍTULO 3. PROPUESTA

A continuación, se estructura la propuesta que pretende llevar a cabo este trabajo.

3.1 Diagramas de procesos

La Figura 3.1 muestra el diagrama de proceso actual, no automatizado con el cual trabaja la fundación para el proceso de rescate y adopción de las mascotas, dicho proceso se pretende automatizar con la implementación del sistema.

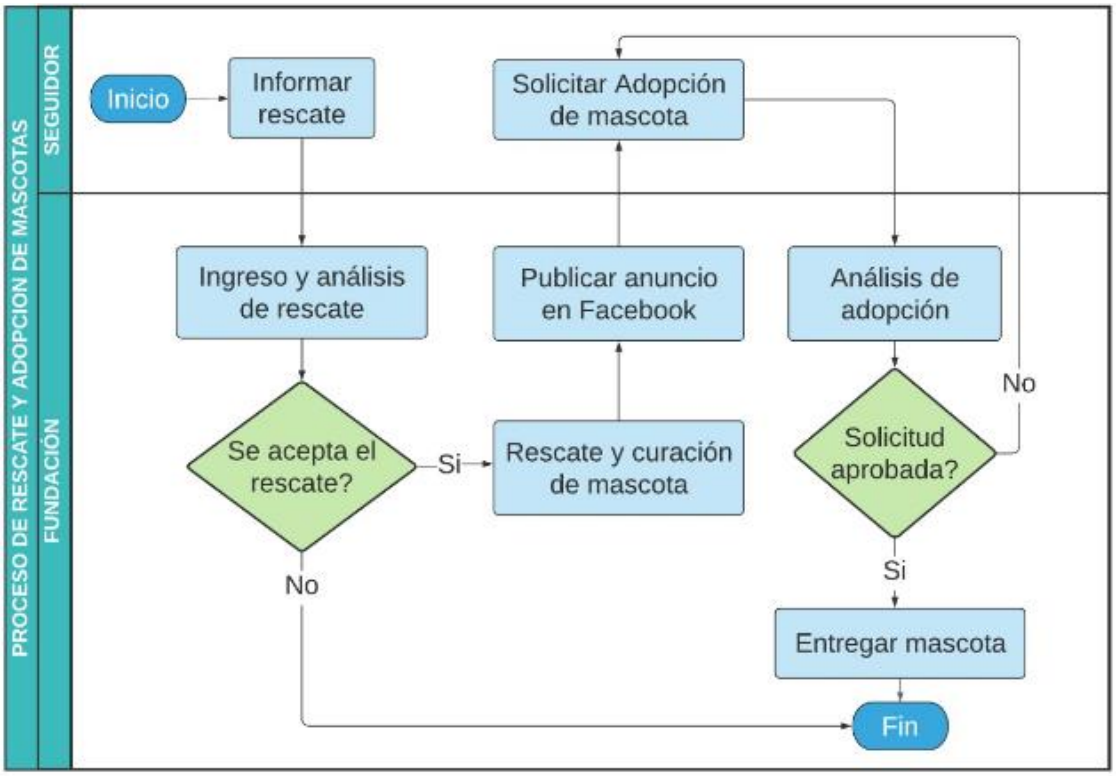


Figura 3.1 Diagrama de proceso manual. **Fuente:** Elaboración propia.

El proceso manual tiene problemas relacionados con el tiempo de respuesta, como se observa en la Figura 3.1 el ingreso y análisis de una mascota a la fundación mediante un correo electrónico, o publicación de Facebook en la cuenta de la fundación, si es aceptada se procede al rescate y curación, cuando una mascota ha sido curada se procede a publicar la donación de la mascota en la cuenta de Facebook. Por último, se realiza un análisis y se entrega la mascota, de lo contrario se repite el proceso de análisis para la entrega.

En la Figura 3.2 se describe observar el flujo de datos para el proceso automatizado, cabe resaltar que se toman en cuenta los departamentos involucrados en cada fase del proceso. El diagrama fue presentado y validado por la propietaria de la Fundación Camino a casa.

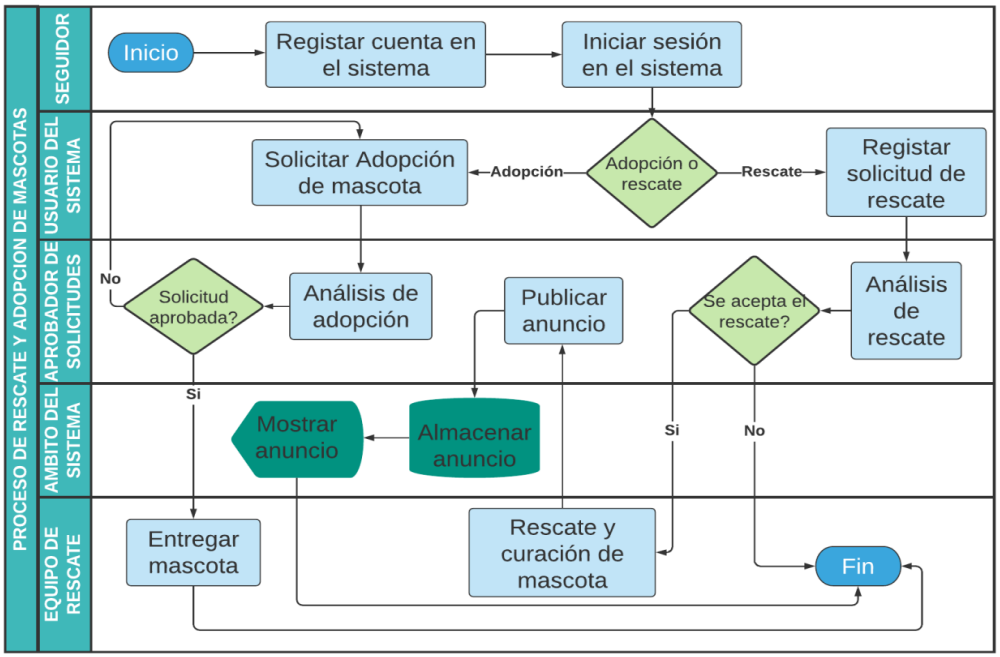


Figura 3.2. Diagrama de proceso automatizado. **Fuente:** Elaboración propia.

El proceso automatizado define los involucrados en el mismo, cualquier usuario puede visualizar los anuncios del sistema, para poder crear o editar anuncios es necesario registrarse en el sistema, por último la aprobación de los anuncios está a cargo del personal administrativo de la fundación, para evitar la publicación de cualquier anuncio indebido. El rescate y curación se realiza en hospitales, clínicas o veterinarias que colaboran con la fundación, de tal forma representa un proceso externo.

3.2 Factibilidad técnica

Para el desarrollo e implementación de este proyecto se utilizó los siguientes recursos técnicos los cuales fueron proporcionados por el estudiante.

Hardware

- MacBook Pro Core I5, 8Gb RAM, disco duro 1 Tb
- IPhone 6S Plus,
- Impresora Canon 8020

Software

- Framework *Ruby on Rails*
- Librerías de *Ruby*: *Devise*, *Bootstrap*, etc.
- Herramientas Web: HTML, CSS, *JQuery*
- Editor de Texto *Sublime Text*
- Terminal de *MacOS*
- Gestor de base de datos PostgreSQL
- Plataforma de alojamiento *Heroku*

Los dispositivos que se usaron para el desarrollo y las pruebas fueron proporcionados por el estudiante autor de este proyecto, y las herramientas de software son de uso libre disponibles en internet.

3.3 Factibilidad operacional

Para el desarrollo e implementación del proyecto bajo la metodología *Scrum* el equipo de trabajo fue formado por el estudiante Marcelo Toapanta como desarrollador y *Scrum master*, y como *Product owner* a la Sra. Cristina Calderón. Una vez desplegado el proyecto se realizó capacitaciones con el cliente sobre el flujo principal del sistema y sus interfaces resultantes.

3.4 Factibilidad económica-financiera

Fundación camino a casa es una organización sin fines de lucro, las misma se sustenta con autogestión de la dirección y donaciones de los seguidores, el uso de las herramientas de desarrollo no generaron costos para la fundación, sin embargo, los costos de desarrollo se detallan en la Tabla 3.1.

Tabla 3.1. ***Costos de desarrollo***

Costos de desarrollo			
Cantidad Meses	**Valor**	**Detalle de recurso**	**Total**
4	$7	Plataforma Heroku	$28
4	$10	Luz eléctrica	$40
4	$600	Programador	$2.400
4	$18	Internet	$72
Total			$2.540

Fuente: Elaboración propia

3.5 Especificación de requerimientos

Según la IEEE_830 la Especificación de Requisitos Software (ERS) debe contemplar toda la información presentada en dicho estándar y, aunque propone una organización de dicha información, no exige estrictamente el formato de dicha información.

3.5.1 Ámbito del software

El nombre del *software* es Mascota Perdida este es un modelo independiente que espera alcanzar los objetivos establecidos para optimizar los procesos de adopción y facilitar el control de animales rescatados, perdidas y encontradas. Los usuarios tendrán acceso a toda la información sobre mascotas perdidas, rescatadas y encontradas, además su registro será de una manera gratuita y confiable.

3.5.2 Funciones del producto

Las historias de usuario son una herramienta recomendada por *Scrum* y recopila la información de las reuniones con el cliente.

Existen varias platillas para las historias de usuario, para este proyecto se utilizó una plantilla con los datos listados a continuación:

- **Número:** Secuencial identificador.
- **Tipo usuario:** Nombre que describe las funciones del usuario.
- **Nombre de historia:** Descripción rápida.
- **Prioridad:** Nivel de importancia con relación a los requerimientos del cliente.
- **Riesgo en desarrollo:** (alta, media, baja) permite identificar posibles complicaciones debido a la complejidad de la tarea.
- **Responsable:** Miembro del equipo de trabajo responsable.
- **Descripción:** Detalle de las funciones en el sistema, se puede hacer uso de conceptos técnicos.

En la Tabla 3.2 se observa un modelo de la historia de usuario con sus respectivos campos que se usó en este proyecto.

Tabla 3.2. ***Historias de Usuario***

HISTORIA DE USUARIO	
Número: #	**Tipo de usuario**: Rol
Nombre historia:	
Prioridad en negocio: Alta, Media, Baja	**Prioridad en desarrollo**: Alta, Media, Baja
Puntos estimados: 10 - 100	**Iteración asignada:** Módulo
Descripción:	
Observación:	

Fuente: Elaboración propia

3.5.3 Características de los usuarios del sistema

La Figura 3.3 muestra las características de los usuarios categorizados por roles.

Tabla 3.3. ***Roles de usuario***

Nombre de Rol	Tipo de Usuario	Área Funcional	Actividad
Administrador	Administrador del Sistema	Administración	Administrar el sistema Administrar cuentas de usuario Crear/Modificar anuncios Visualizar/Buscar anuncios
Aprobador	Aprobador de anuncios	Dirección	Aprobar anuncios Eliminar anuncios Publicar anuncios Crear/Modificar anuncios Visualizar/Buscar anuncios
Anunciante	Anunciantes de anuncios	Atención al cliente	Crear/Modificar anuncios Visualizar/Buscar anuncios
Visitante	Visor	Atención al cliente	Visualizar/Buscar anuncios

Fuente: Elaboración propia

3.5.4 Restricciones de desarrollo

Las restricciones para el desarrollo del sistema se describen a continuación:

- Se utilizará el sistema gestor de base de datos PostgreSQL y la herramienta Ruby on Rails para el desarrollo del sistema. Con el propósito de realizar pruebas y demostraciones para el cliente el sistema fue alojado en la plataforma web Heroku.
- La fundación realiza sus operaciones en el D.M de Quito, por este motivo el sistema web será desarrollado para ser usado en dicha ciudad, sin embargo, tendrá un diseño escalable y puede aumentar su capacidad para otras ciudades.

3.5.5 Requisitos

Los requisitos que el cliente solicita en el desarrollo del sistema están plasmados en las listas de requerimientos funcionales y no funcionales mostrados a continuación.

Funcionales.

Los requisitos funcionales del sistema fueron definidos por las historias de usuario recolectadas, las mismas describen los requerimientos del cliente que serán entregados de forma progresiva por el equipo de trabajo.

Según *Scrum* el *Product Backlog* es el conjunto de todas las historias de usuario definidas por el equipo de trabajo y sus puntos estimados que permiten priorizar las tareas más importantes, también permiten generar un tiempo estimado para completar las tareas y verificar su avance en cada *Sprint*. A continuación, en la Tabla 3.4 se observa la lista de historias de usuario.

Tabla 3.4. ***Lista historias de usuario***

Lista de historias de usuario		
Nro. Referencia	**Historia de usuario**	**Puntos estimados**
RF01	Menú principal	60
RF02	Registrar usuario	15
RF03	Gestión de usuario	30
RF04	Gestión de anuncios	40
RF05	Aprobar/Publicar anuncios	10
RF06	Panel de anuncios por publicar	10
RF07	Buscar anuncios	10
RF08	Contactar anunciante	30
RF09	Panel sectorizado	40
RF10	Compartir en redes sociales	10

Fuente: Elaboración propia

Las historias de usuario fueron agrupadas en *Sprints* y fueron desglosadas en tareas para que el equipo de trabajo las realice, a continuación se muestra cómo se definieron los *Sprint* con las historias de usuario, este paso facilitó a la presentación de cada módulo al cliente ver Tabla 3.5. Lista de historias de usuario ver Anexo 5.

Tabla 3.5. ***Lista de Sprints***

Lista de *Sprints*

Numero	Nombre	Historias de Usuario	Duración días
0	Crear base de datos y datos iniciales	No aplica	5
1	Módulo de usuario	RF02, RF03	10
2	Módulo de anuncios	RF04, RF05, RF06, RF07	15
3	Módulo de mensajes	RF08	5
4	Módulo de panel sectorizado	RF09, RF10	5
5	Diseño ajustable y navegabilidad	RF01	5

Fuente: Elaboración propia

Cada *Sprint* fue cubierto de forma sistemática, ya que las historias de usuario fueron descompuestas en tareas, para gestionar el avance o desfase de cada *Sprint*, en el caso de que una tarea no se pueda completar se debe agregar al siguiente *Sprint* y notificar al cliente del retraso en la planificación original.

Prototipos de pantallas

Los prototipos de pantalla ayudan al equipo de trabajo a mostrar al cliente un resultado cercano al producto final. De tal manera se puede tener una idea más efectiva de lo que se espera al final de *Sprint*.

En el diseño de los prototipos se observa los elementos con los cuales los usuarios podrán interactuar y los enlaces para navegar entre las pantallas del sistema. Un prototipo debe ser funcional, es decir que permita realizar los objetivos de la pantalla fácilmente (Cuello & Vittone, 2013).

El equipo de trabajo diseñó los prototipos de pantallas los cuales que fueron agrupados por módulos.

Módulo de usuarios

Figura 3.3. Prototipo – Perfil de usuario. **Fuente:** Elaboración propia.

El perfil del usuario ayuda a establecer una vía de comunicación entre los usuarios registrados en el sistema, ya que los datos que registren en esta pantalla podrán ser vistos por los otros usuarios.

Figura 3.4. Prototipo – Iniciar sesión. **Fuente:** Elaboración propia.

En la pantalla de inicio de sesión se muestra que es necesario el correo electrónico y la contraseña para iniciar una sesión de forma fácil y segura en el sistema.

Figura 3.5. Prototipo – Registrar usuario. **Fuente:** Elaboración propia.

La pantalla para registrar usuario permite crear usuarios en el sistema, los seguidores de la fundación pueden ingresar a este formulario para llenar los campos correo electrónico, contraseña y confirmar contraseña y posteriormente utilizar las funciones del sistema.

Módulo de anuncios

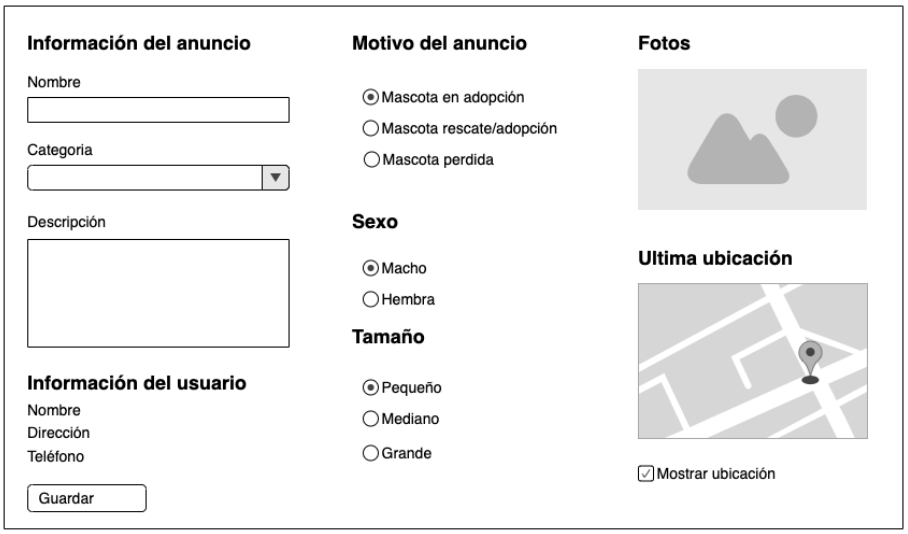


Figura 3.6. Prototipo – Formulario anuncio. **Fuente:** Elaboración propia.

Para crear anuncios el sistema tiene un formulario que solo puede ser accedido por los usuarios registrados. El formulario de anuncios se puede ingresar los siguientes datos:

- Nombre: Nombre de la mascota o título del anuncio
- Categoría: Perros, Gatos, Aves
- Descripción: Detalle del anuncio
- Información del usuario: Datos del usuario conectado
- Motivo del anuncio: Adopción, Rescato/donación, Perdida
- Sexo: Macho, Hembra
- Tamaño: Pequeño, Mediano, Grande
- Fotos: Hasta tres fotos de la mascota
- Ultima ubicación: Ubicación en el mapa

Figura 3.7. Prototipo – Detalle de anuncio. **Fuente:** Elaboración propia.

Los usuarios podrán ver los anuncios al detalle en la pantalla con la información agrupada y dispuesta de la siguiente manera: Foto, Descripción, Información de mascotas, Información de usuario y botones de acción.

Figura 3.8. Prototipo – Lista de anuncios. **Fuente:** Elaboración propia.

La lista de anuncios permite ver todos los anuncios publicados más recientemente en forma de tarjetas, las cuales tienen fotos, nombre del anuncio, fecha de anuncio, descripción. En esta pantalla se puede ver filtros para la lista de anuncios: Adopción, Rescate/Donación, Pérdidas, Todas.

Módulo de mensajes

Figura 3.9. Prototipo – Lista de mensajes. **Fuente:** Elaboración propia.

En el módulo de mensajes existe una página con una lista de conversación a la izquierda, una lista de mensajes por cada conversación a la derecha y por ultimo un cuadro de texto para enviar mensajes.

Módulo de panel sectorizado

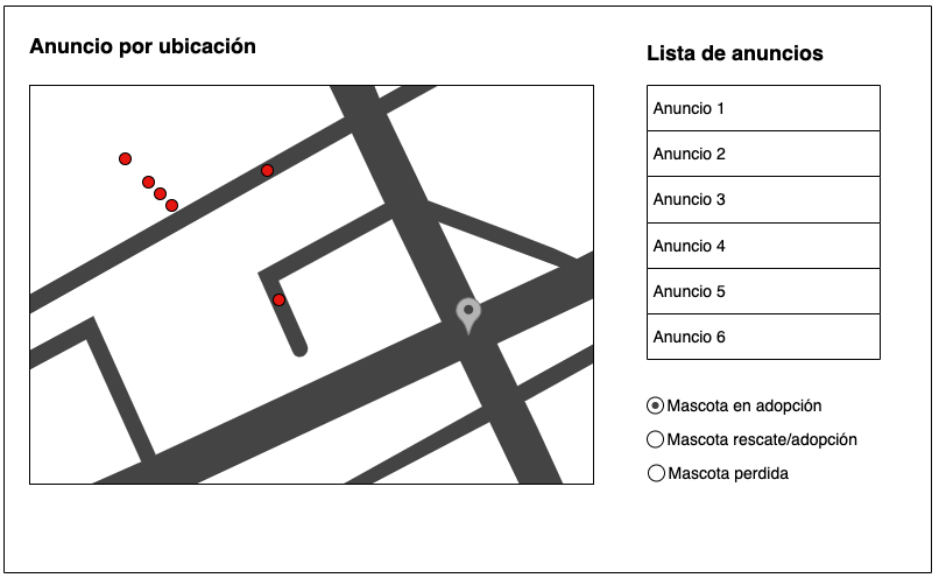


Figura 3.10. Prototipo – Panel sectorizado. **Fuente:** Elaboración propia.

En el módulo de panel sectorizado existe una pantalla con un mapa y una lista de anuncios publicados, los cuales pueden ser filtrados la ubicación. Adicionalmente existe un filtro de similar al de la lista: Adopción, Rescate/Donación, Pérdida.

Diseño ajustable y navegabilidad

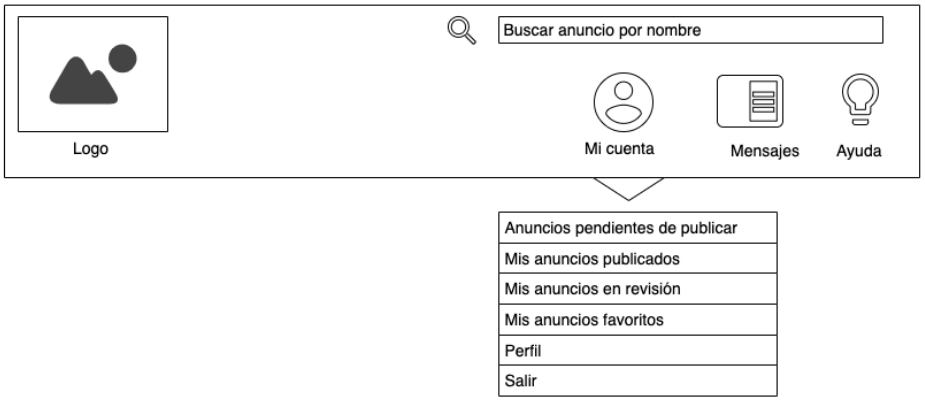


Figura 3.11. Prototipo – Menú de navegación. **Fuente:** Elaboración propia.

Como último prototipo el equipo de trabajo realizó el diseño del menú de navegación el cual permite a los usuarios navegar por los diferentes módulos del sistema, el menú está formado por un logo, enlace mi cuenta con los enlaces a las listas de anuncios, perfil y salir, enlace mensajes para acceder a la lista de mensajes y ayuda.

No funcionales.

En esta sección se listan los requisitos no funcionales separados por categorías.

Eficiencia.

RNF01: Todos los anuncios que sean creados o modificados por los usuarios no deben tomar más de 2 segundos en realizar la operación en la base de datos.

RNF02: Las operaciones de buscar y listar anuncios no debe tomar más de 5 segundos.

Seguridad, consistencia de datos.

RNF03: El administrador del sistema es el encargado de gestionar los permisos de cada rol de usuario.

RNF04: La herramienta *Ruby on Rails* proporciona la seguridad para el sistema con los mejores estándares conocidos hasta la fecha.

RNF05: La seguridad de la información de usuario es gestionada por la gema "devise", la cual permite realizar inicios de sesión, registro de usuarios, recuperación de clave, control de accesos no autorizados.

RNF06: La plataforma Heroku realiza respaldos en periodos configurables de acuerdo a las necesidades del cliente.

Usabilidad.

RNF07: El sistema fue diseñado para ser entendido fácilmente, con interfaces sencillas y enlaces de acceso rápido a los módulos del sistema.

RNF08: El sistema fue desarrollo para plataforma web, por lo cual los usuarios pueden gestionar sus anuncios a través de Internet.

RNF09: Las interfaces para el sistema son ajustables a las pantallas de los dispositivos que pueden acceder al mismo.

CAPÍTULO 4. IMPLEMENTACIÓN

En este capítulo se desarrolla el sistema de forma lógica y programática.

4.1 Ejecución de *Sprints*

Acorde a la metodología *Scrum*, se debe detallar cada *Sprint* y su producto final según la planificación realizada para el proyecto. El *Sprint Backlog* es el conjunto de tareas necesarias para completar un *Sprint.*

Sprint 0 - Crear base de datos y datos iniciales

Tabla 4.1. ***Detalle Sprint 0***

Detalle *Sprint* 0	
Objetivo:	Generar base de datos funcional
Duración:	14/01/2019 – 18/01/2019
Historias por implementar:	Ninguna
Tiempo de ejecución:	5 días laborables
Puntos totales	25

Fuente: Elaboración propia

Para gestionar la conexión con la base de datos *RoR* integra una gema llamada *ActiveRecord* para crear tablas, columnas, restricciones y relaciones a través de migraciones, mediante archivos de texto plano se creó los modelos y posteriormente la base de datos.

A continuación, se lista las tareas que fueron necesarias para completar el ciclo.

Tabla 4.2. ***Tarea 1 – Sprint 0***

Tarea: Crear modelos en RoR		
Número: 1	**Historia**: Ninguna	**Puntos:** 10
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Escribir los modelos: Usuario, Perfil, Anuncio, Anuncio Favorito, Categoría, Conversación, Mensaje.		

Fuente: Elaboración propia

Tabla 4.3. ***Tarea 2 – Sprint 0***

Tarea: Crear base de datos		
Número: 2	**Historia**: Ninguna	**Puntos:** 10
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Generar migración para crear base de datos.		

Fuente: Elaboración propia

Tabla 4.4. ***Tarea 3 – Sprint 0***

Tarea: Crear usuarios administrativos		
Número: 3	**Historia**: Ninguna	**Puntos:** 5
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Crear usuario Administrador del sistema y Aprobador.		

Fuente: **Elaboración propia**

Burndown inicial

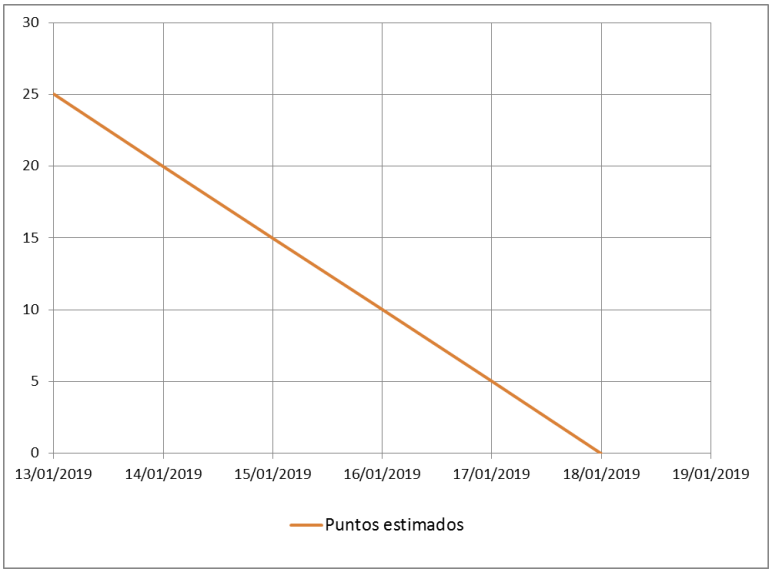


Figura 4.1. *Sprint* 0 – Burndown inicial. **Fuente:** Elaboración propia.

Burndown final

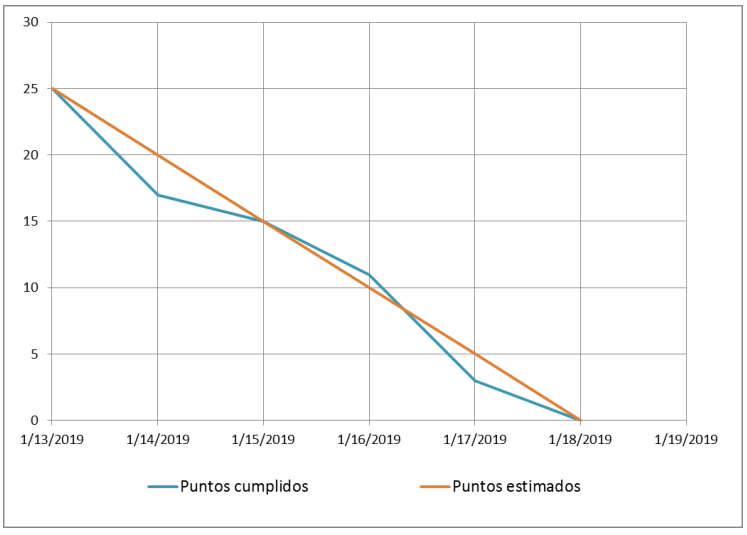


Figura 4.2. *Sprint* 0 – Burndown final. **Fuente:** Elaboración propia.

La Figura 4.1 muestra la planificación ideal del *Sprint* 0, el cual consta de 3 tareas que suman 25 puntos y fueron cubiertos exitosamente en 5 días. La Figura 4.2 refleja el avance en los 5 días, nótese que, al inicio, terminado el día uno hubo un avance mayor al estimado como lo refleja la línea azul, esto se debe a que los modelos, sus columnas y relaciones fueron completados casi en su totalidad.

ActiveRecord ejecutó las sentencias SQL necesarias para que gestor de base de datos pueda crear las tablas, columnas y relaciones basadas en los modelos.

Resultados

Figura 4.3. *Sprint* 0 – Diagrama de base de datos. **Fuente:** Elaboración propia.

intial_data.rake

```
namespace :tasks do
  desc "Datos Iniciales"
  task :datos_iniciales => :environment do
    # CREAR USUARIOS
    # Usuario Administrador
    user = User.new
    user.email = "admin@mascota-perdida.com"
    user.password = "xxxxxx"
    user.password_confirmation = "xxxxxx"
    user.role = :admin
    user.save
    profile = Profile.new
    profile.user_id = user.id
    profile.name = "Administrador"
    profile.direction = "Calle 21 de Agosto E3-200, S27A"
    profile.phone = "0995844878"
    profile.save
    # Usuario Fundación
    user = User.new
    user.email = "camino-a-casa@mascota-perdida.com"
    user.password = "xxxxxx"
    user.password_confirmation = "xxxxxx"
    user.role = :organization
    user.save
    profile = Profile.new
    profile.user_id = user.id
    profile.name = "Administrador de Fundación"
    profile.direction = "Calle 21 de Agosto E3-200, S27A"
    profile.phone = "0995844878"
    profile.save
    # Usuario Fundación 1
    user = User.new
    user.email = "patitas-callejeras@mascota-perdida.com"
    user.password = "xxxxxx"
    user.password_confirmation = "xxxxxx"
    user.role = :organization
    user.save

    profile = Profile.new
    profile.user_id = user.id
    profile.name = "Administrador de Fundación 1"
    profile.direction = "Calle 21 de Agosto E3-200, S27A"
    profile.phone = "0995844878"
    profile.save
  end
end
```

Figura 4.4. *Sprint* 0 – Crear usuarios. **Fuente:** Elaboración propia.

Retroalimentación

La ejecución del *Sprint* 0 permitió al equipo de trabajo crear la base de datos funcional para el sistema y los usuarios administradores, ya que el resultado no es algo visible para el cliente, se infirmo la terminación exitosa del *Sprint* solo por vía telefónica.

El primer ciclo no tuvo incidentes lo cual permitió al equipo de trabaja cumplir con las tareas a tiempo con la planificación inicial.

Sprint 1 - Módulo de usuario

Tabla 4.5. ***Detalle Sprint 1***

Detalle *Sprint* 1	
Objetivo:	Implementar el módulo de usuario, gema "devise"
Duración:	21/01/2019 – 01/02/2019
Historias por implementar:	RF02, RF03
Tiempo de ejecución:	10 días laborables
Puntos totales	45

Fuente: Elaboración propia

La gema "devise" permite gestionar usuarios y sesiones de forma segura acorde a los últimos estándares disponibles en el mercado, esta gema es constantemente actualizada por la comunidad de Ruby Gems (RubyGems.org).

Entre las funciones principales de la gema se encuentran:

- Gestión de inicio de sesión
- Gestión de Registro de usuario
- Gestión de recuperación de contraseña
- Permanecía de sesión mediante cookies
- Envió de notificaciones vía correo electrónico

A continuación, se lista las tareas que fueron necesarias para completar el ciclo.

Tabla 4.6. ***Tarea 1 – Sprint 1***

Tarea: Implementar registro de usuario		
Número: 1	**Historia**: RF01	**Puntos:** 15
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Crear formulario para que los seguidores de la fundación puedan registrarse en el sistema, los campos necesarios para realizar esta operación son un correo electrónico y una contraseña mínimo de seis caracteres.		

Fuente: Elaboración propia

Tabla 4.7. ***Tarea 2 – Sprint 1***

Tarea: Implementar inicio de sesión		
Número: 2	**Historia**: RF03	**Puntos:** 15
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Crear formulario para que los usuarios del sistema inicien una sesión de forma fácil y segura, si el usuario no ingresa correctamente uno de los campos el formulario debe mostrar un mensaje que indique los errores cometidos.		

Fuente: Elaboración propia

Tabla 4.8. ***Tarea 3 – Sprint 1***

Tarea: Implementar perfil de usuario		
Número: 3	**Historia**: RF03	**Puntos:** 15
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Crear formulario para que los usuarios gestionen sus perfiles, estos datos serán compartido entre los usuarios del sistema para que se puedan comunicar entre ellos.		

Fuente: Elaboración propia

Burndown inicial

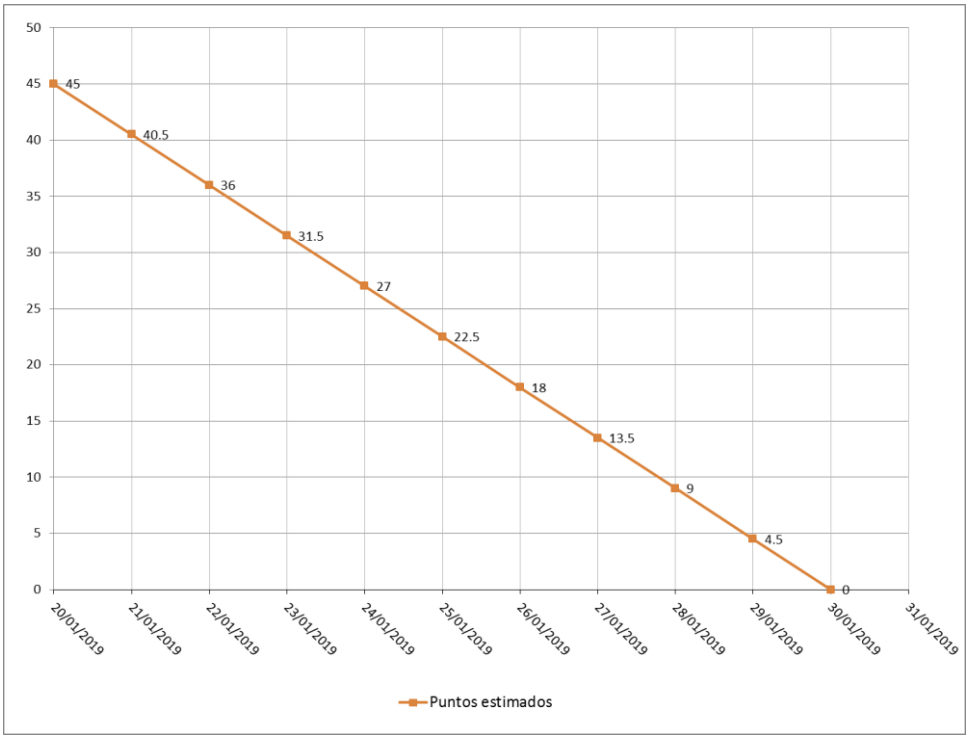


Figura 4.5. *Sprint* 1 – Burndown inicial. **Fuente:** Elaboración propia.

Burndown final

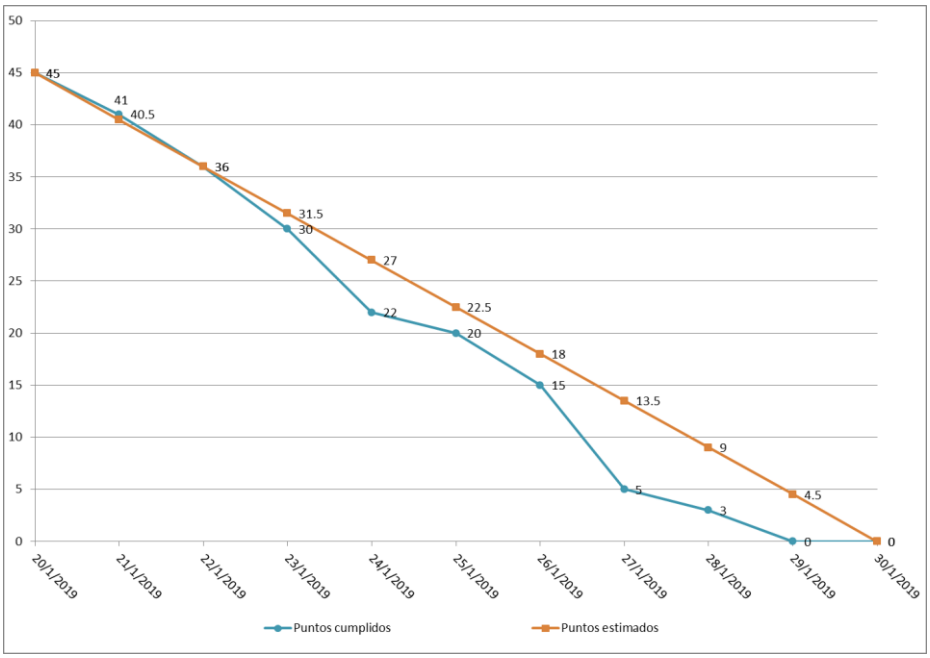


Figura 4.6. *Sprint* 1 – Burndown final. **Fuente:** Elaboración propia.

En este *Sprint* se puede observar que el equipo de trabajo pudo completar las tareas antes del tiempo planificado inicialmente, es decir los puntos de las tareas que fueron cubiertos satisfactoriamente son mayores a los estimados, como se puede ver en la Figura 4.6 se completaron los todos los puntos el 29 de enero del 2019.

Resultados

Figura 4.7. *Sprint* 1 – Perfil de usuario. **Fuente:** Elaboración propia.

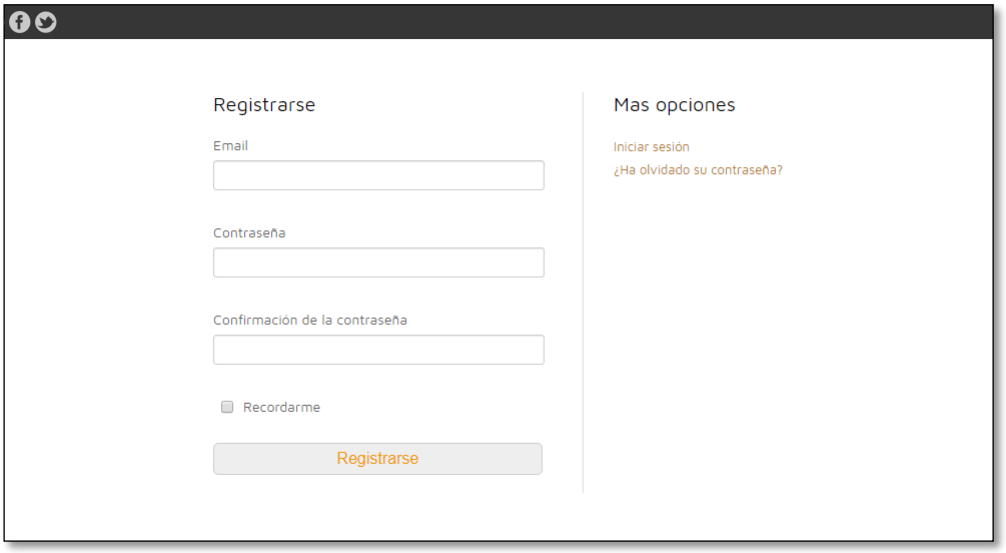


Figura 4.8. *Sprint* 1 – Registro de usuario. **Fuente:** Elaboración propia.

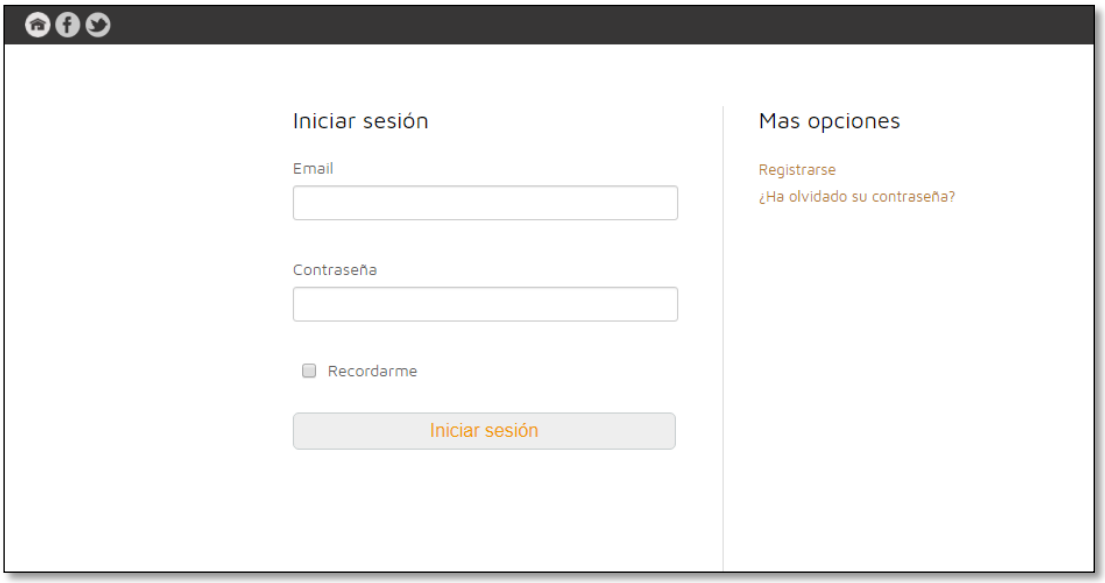


Figura 4.9. *Sprint* 1 – Iniciar sesión. **Fuente:** Elaboración propia.

Retroalimentación

Se realizó una reunión el 30 de enero del 2019 con el cliente para una demostración de inicio de sesión, el cliente creo una sesión con la cuenta de Administrador y después edito su perfil. Después se mostró como los seguidores de la fundación pueden registrarse y acceder al sistema.

Sprint 2 - Módulo de anuncios

Tabla 4.9. ***Detalle Sprint 2***

Detalle *Sprint* 2	
Objetivo:	Implementar el módulo de anuncios
Duración:	04/02/2019 – 22/02/2019
Historias por implementar:	RF04, RF05, RF06, RF07
Tiempo de ejecución:	15 días laborables
Puntos totales	80

Fuente: Elaboración propia

El *Sprint* 2 implementó los requerimientos correspondientes al módulo de anuncios, la cantidad de historias y tareas por cubrir representaron un alto grado de complejidad para el equipo de trabajo.

El equipo de trabajo debe cumplir 80 puntos en 15 días laborables lo que corresponde a 3 semanas, por la complejidad y duración del *Sprint* se procedió a ejecutar un control a la mitad del *Sprint* para mitigar incidentes que pueda presentarse. A continuación, se lista las tareas que fueron necesarias para completar el ciclo.

Tabla 4.10. ***Tarea 1 – Sprint 2***

Tarea: Implementar registro de anuncios		
Número: 1	**Historia**: RF04	**Puntos:** 20
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Crear formulario para registrar y editar anuncios, los campos necesarios para crear un anuncio es el nombre de la mascota y una categoría, como campos opcionales que ayudan al usuario a describir el anuncio tenemos, descripción, destalle de la mascota, ultima ubicación y fotos.		

Fuente: Elaboración propia

Tabla 4.11. ***Tarea 2 – Sprint 2***

Tarea: Implementar lista de anuncios		
Número: 2	**Historia**: RF04	**Puntos:** 10
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Crear pantalla que permita listar los anuncios aprobados en el sistema, esta pantalla debe contener anuncios en forma de bloque con la imagen, nombre, fecha de publicación, sexo y tamaño de la mascota.		

Fuente: Elaboración propia

Tabla 4.12. ***Tarea 3 – Sprint 2***

Tarea: Implementar búsqueda de anuncios		
Número: 3	**Historia**: RF04	**Puntos:** 10
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Crear pantalla que permita buscar los anuncios aprobados en el sistema, los filtros que contendrá la pantalla son: Adopción, Rescate / donación, Perdida, Todas		

Fuente: Elaboración propia

Tabla 4.13. ***Tarea 4 – Sprint 2***

Tarea: Compartir anuncios en redes sociales		
Número: 4	**Historia**: RF10	**Puntos:** 20
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: El sistema debe contener botones que permitan compartir los anuncios en Facebook y Twitter.		

Fuente: Elaboración propia

Tabla 4.14. ***Tarea 5 – Sprint 2***

Tarea: Aprobar y publicar anuncios		
Número: 5	**Historia**: RF05	**Puntos:** 10
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: El usuario aprobador es el único que tendrá la opción de publicar los anuncios.		

Fuente: Elaboración propia

Tabla 4.15. ***Tarea 6 – Sprint 2***

Tarea: Crear lista de anuncios por publicar		
Número: 6	**Historia**: RF05	**Puntos:** 10
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Implementar pantalla para listar los anuncios pendientes de aprobación.		

Fuente: Elaboración propia

Burndown inicial

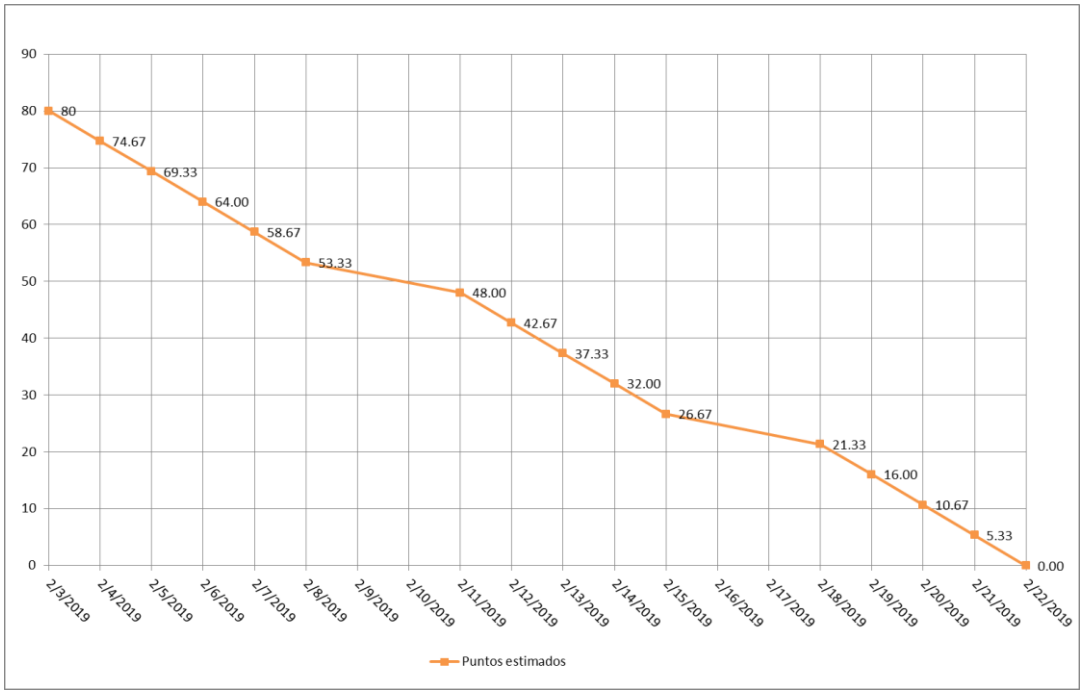


Figura 4.10. *Sprint* 2 – Burndown inicial. **Fuente:** Elaboración propia.

Burndown Control

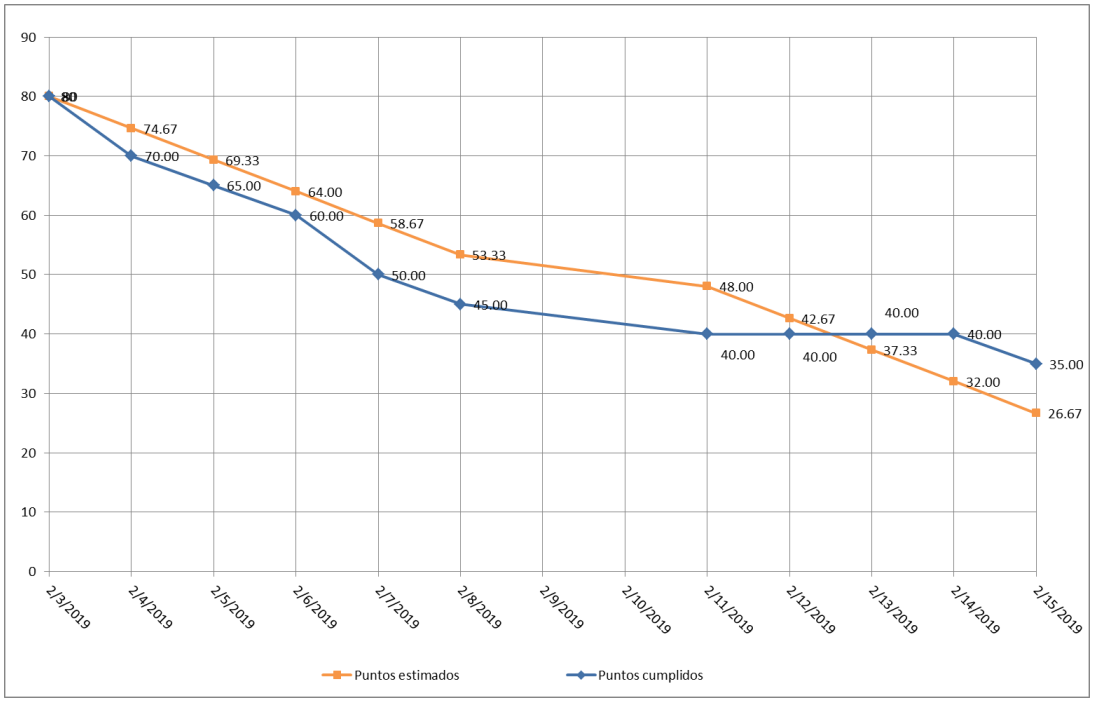


Figura 4.11. *Sprint* 2 – Burndown control. **Fuente:** Elaboración propia.

Burndown final

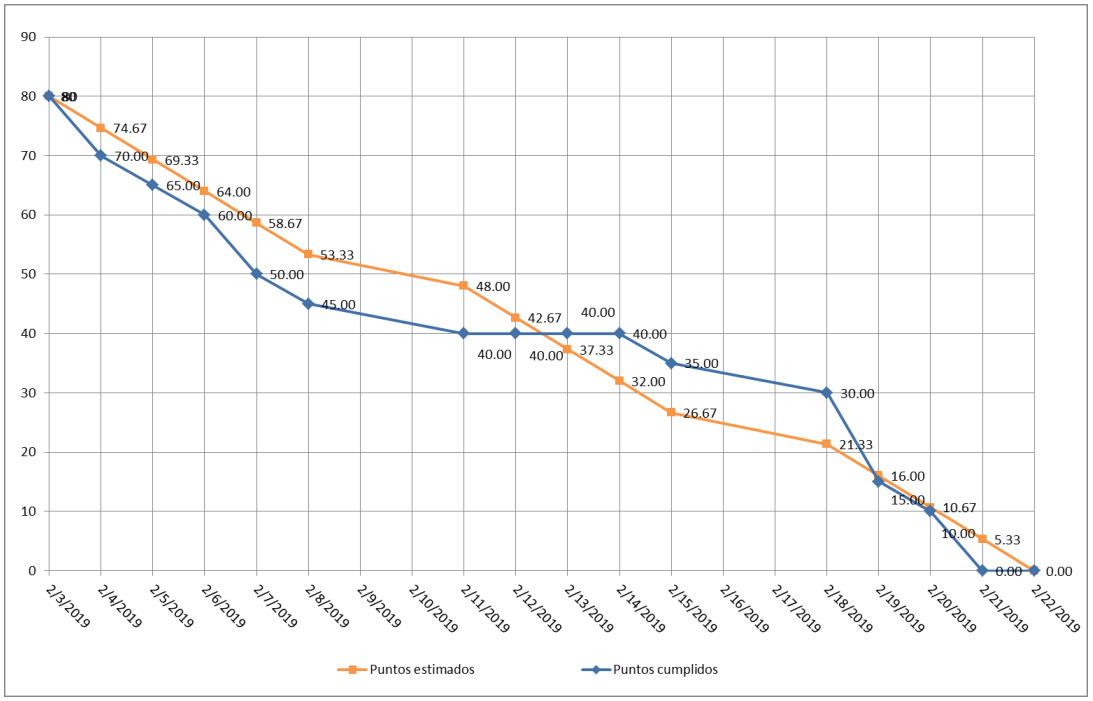


Figura 4.12. *Sprint* 2 - Burndown final. **Fuente:** Elaboración propia.

Resultados

En el *Sprint* 2 se presentó un incidente con la tarea 4 que busca implementar botones para que se pueda compartir el anuncio en Facebook y Twitter, dado a que no es una tarea fácil compartir en redes sociales como se estimó inicialmente.

Como se observa en la Figura 4.11 el Burndown de control la línea azul se coloca arriba de la line naranja los días 13, 14 y 15 de enero, esto representó un retraso en la planificación inicial.

Para mitigar el incidente con la tarea 4 fue necesario realizar una investigación sobre cómo implementar dicha funcionalidad, de acuerdo a la documentación de Facebook es necesario implementar atributos de HTML bajo el protocolo *Open Graph* para completar la funcionalidad y puedan ser entendidos al momento de compartir. Una vez realizado este paso se procedió a instalar la gem “social-share-button” para crear los botones y estilos en la pantalla de anuncios.

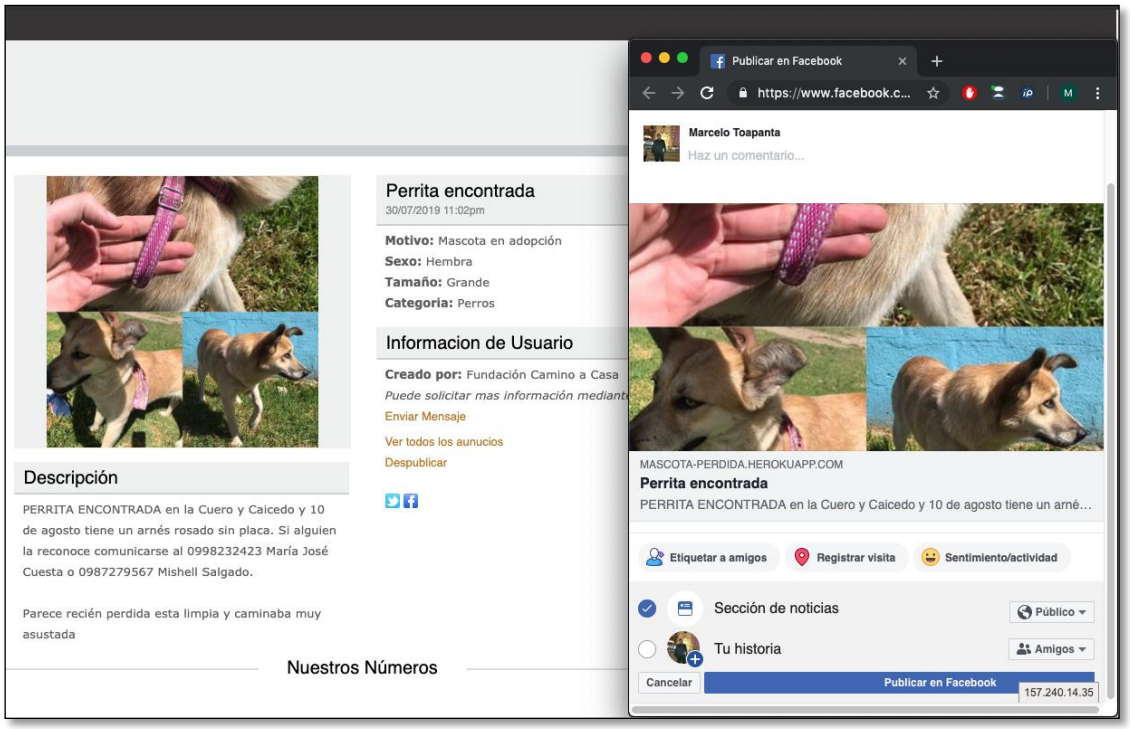


Figura 4.13. *Sprint* 2 – Compartir redes sociales. **Fuente:** Elaboración propia.

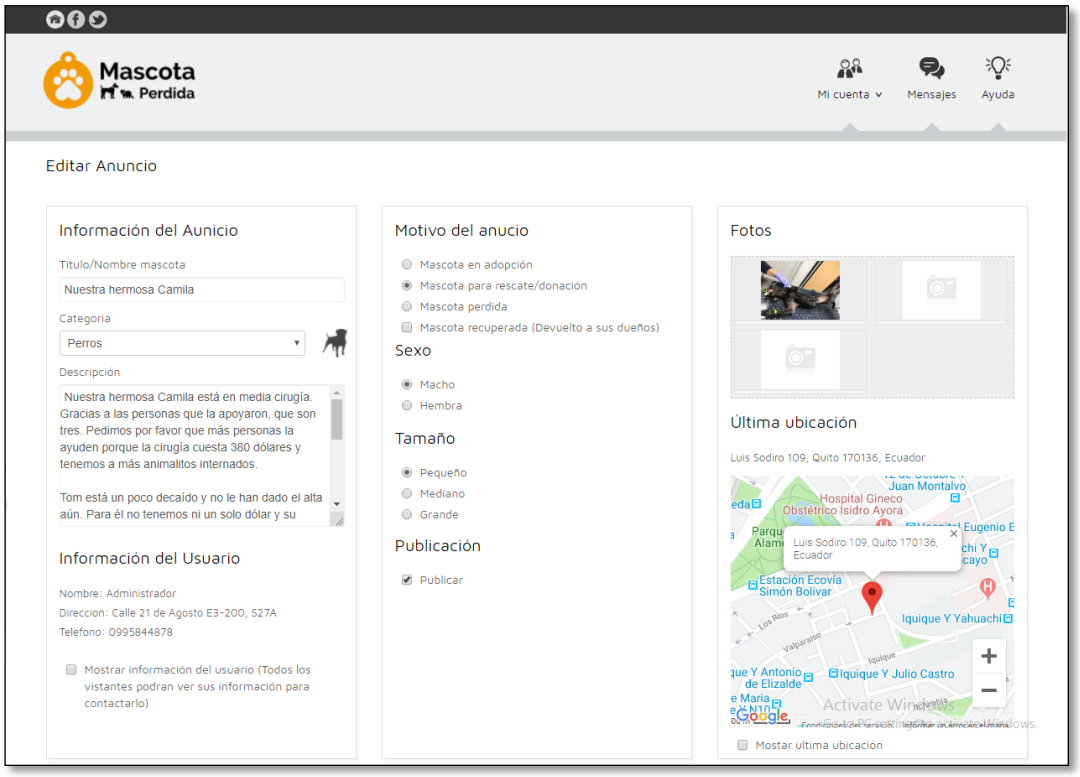


Figura 4.14. *Sprint* 2 – Formulario anuncios. **Fuente:** Elaboración propia.

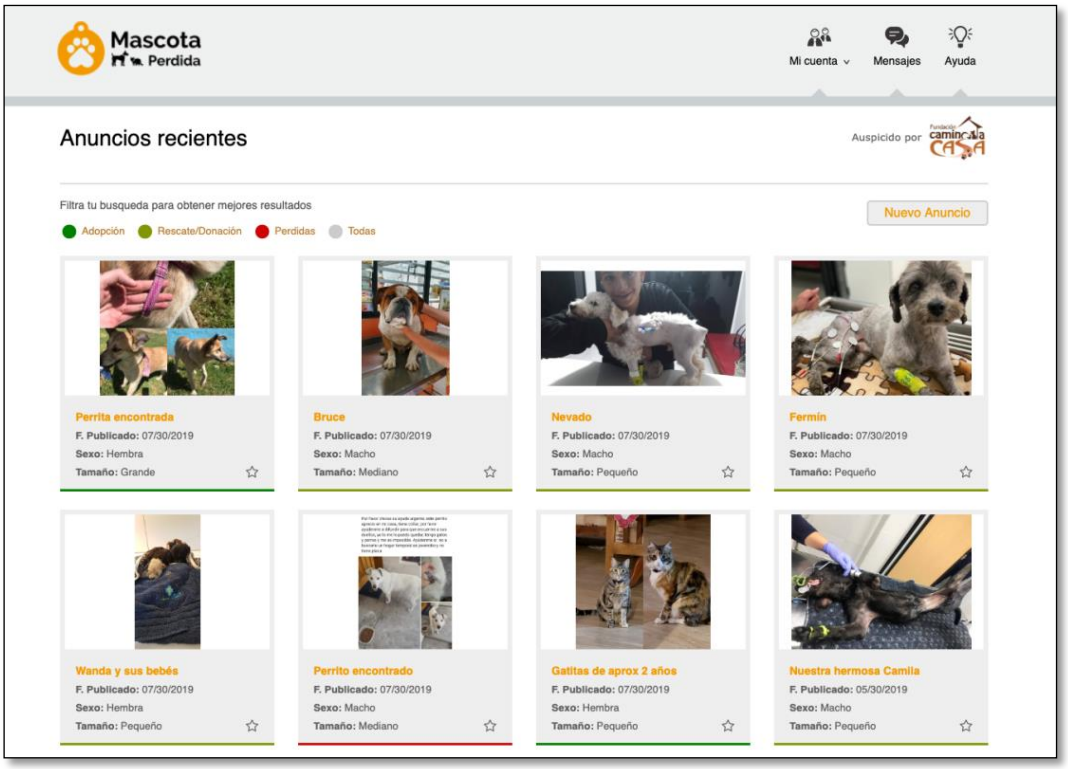


Figura 4.15. *Sprint* 2 – Lista de anuncios. **Fuente:** Elaboración propia.

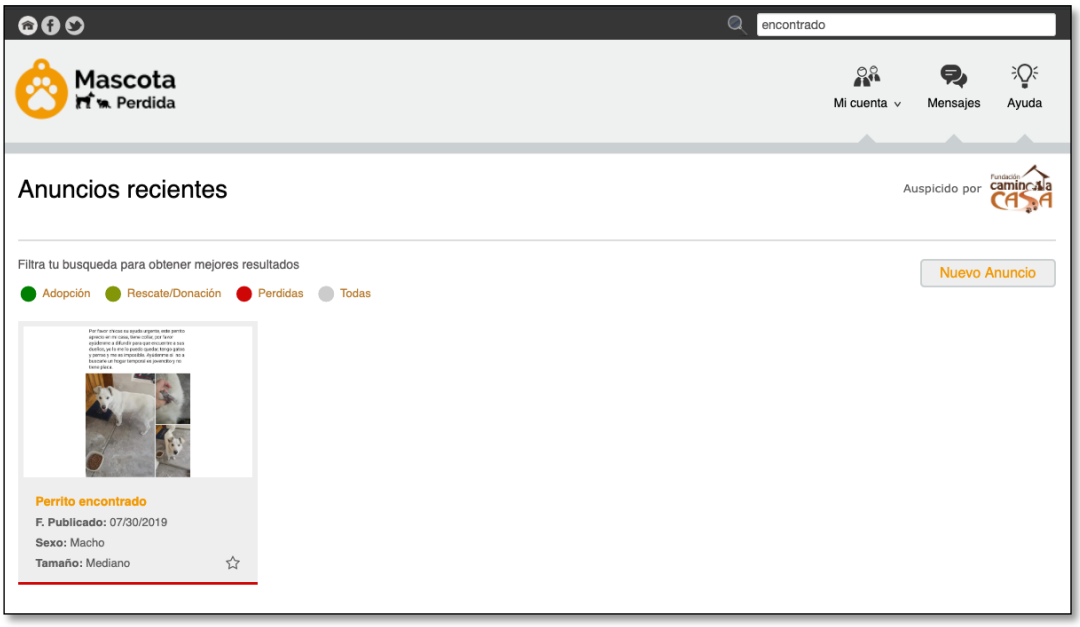


Figura 4.16. *Sprint* 2 – Búsqueda de anuncios. **Fuente:** Elaboración propia.

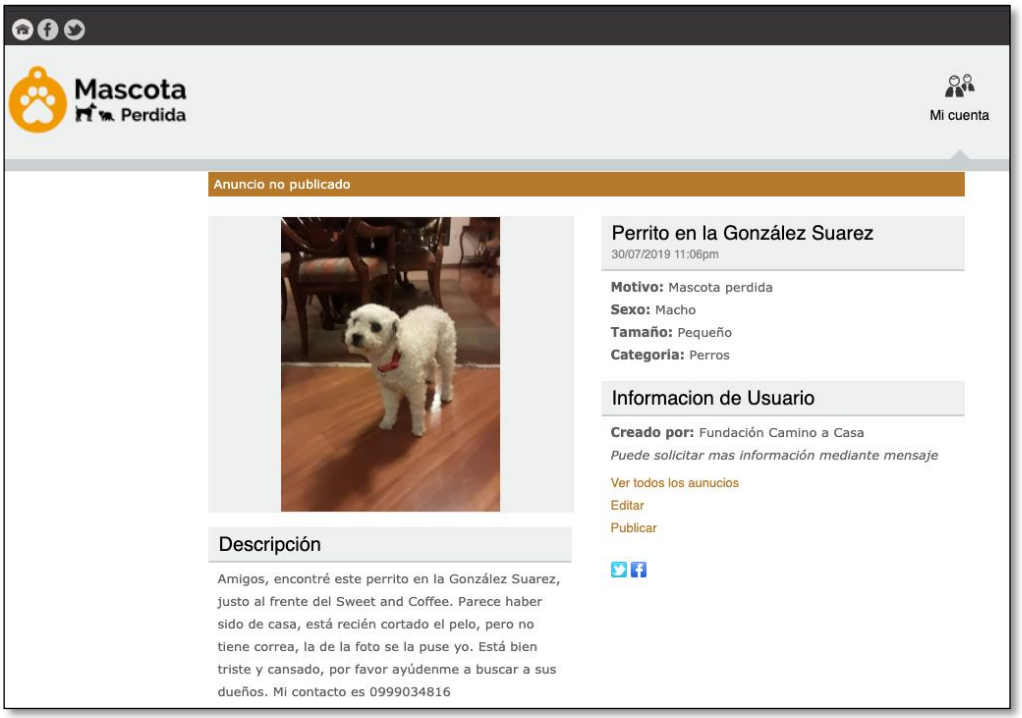


Figura 4.17. *Sprint* 2 – Publicar anuncio. **Fuente:** Elaboración propia.

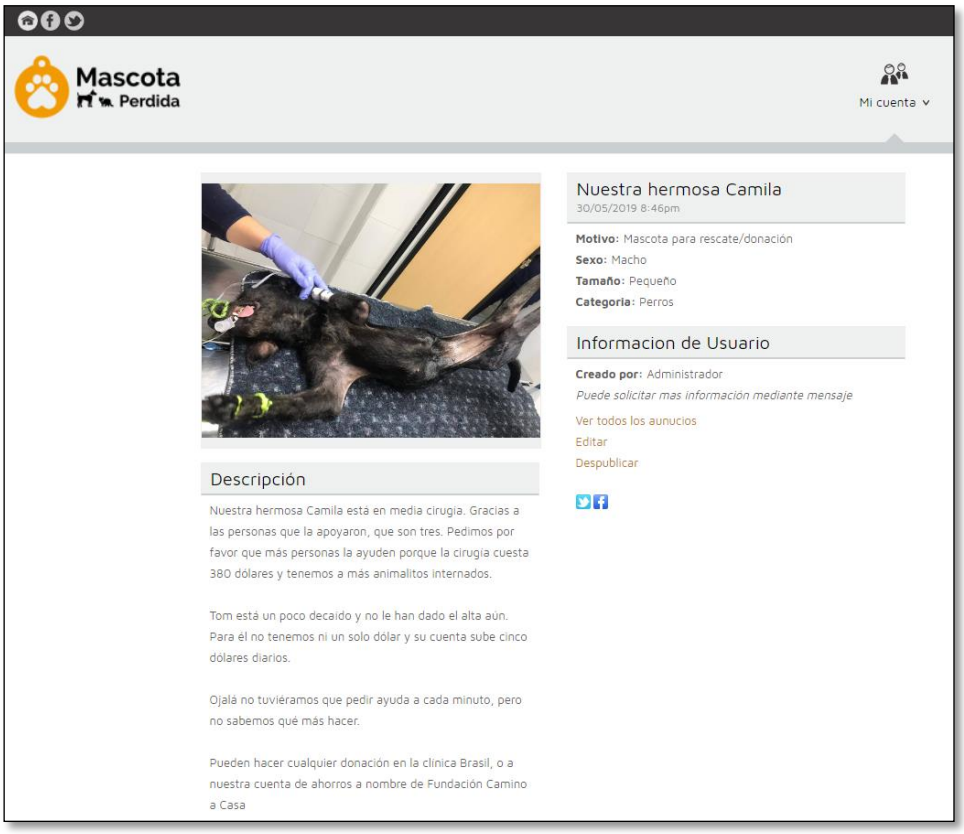


Figura 4.18. *Sprint* 2 – Detalle de anuncio. **Fuente:** Elaboración propia.

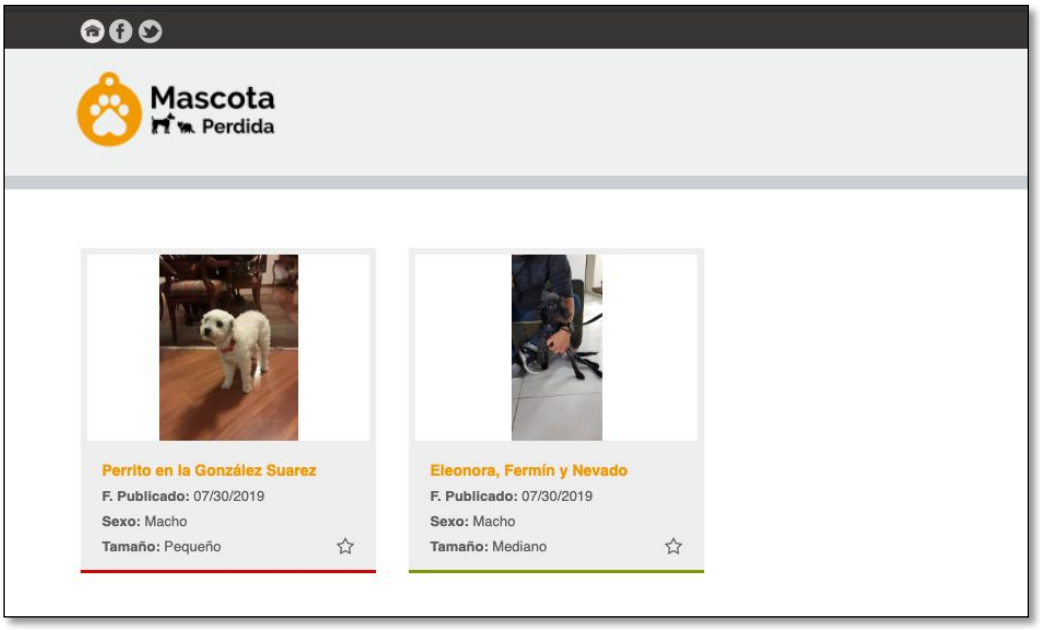


Figura 4.19. *Sprint* 2 – Lista anuncios por publicar. **Fuente:** Elaboración propia.

Retroalimentación

Los requerimientos del cliente referentes al módulo de anuncios fueron cubiertos en el tiempo planificado inicialmente, sin embargo, como se describió en el Burndown de control el incidente retardo el avance del equipo de trabajo. Cuando se superó esta tarea se buscó optimizar el tiempo entre las tareas restantes y cumplir con el plazo establecido inicialmente.

El 23 de enero del 2019 se mantuvo una reunión para mostrar el módulo de anuncios y todas sus funciones. El cliente creó un anuncio y como usuario aprobador procedió a publicar el anuncio, una vez publicado el anuncio se probó los botones para compartir en redes sociales.

Sprint 3 - Módulo de mensajes

Tabla 4.16. ***Detalle Sprint 3***

Detalle *Sprint* 3	
Objetivo:	Implementar el módulo de mensajes
Duración:	25/02/2019 – 01/03/2019
Historias por implementar:	RF08
Tiempo de ejecución:	5 días laborables
Puntos totales	30

Fuente: Elaboración propia

Para establecer una comunicación entre los usuarios el equipo de trabajo implementó un módulo de mensajes.

A continuación, se lista las tareas que fueron necesarias para completar el ciclo.

Tabla 4.17. ***Tarea 1 – Sprint 3***

Tarea: Implementar módulo de mensajes		
Número: 1	**Historia**: RF08	**Puntos:** 30
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: El sistema debe contener una pantalla con las conversaciones y mensajes de los usuarios, este módulo permitirá a los usuarios comunicarse de forma efectiva.		

Fuente: Elaboración propia

Tabla 4.18. ***Tarea 2 – Sprint 3***

Tarea: Crear enlaces a modulo mensajes		
Número: 2	**Historia**: RF08	**Puntos:** 10
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Cada anuncio debe tener un enlace para la conversación entre usuarios.		

Fuente: Elaboración propia

Burndown inicial

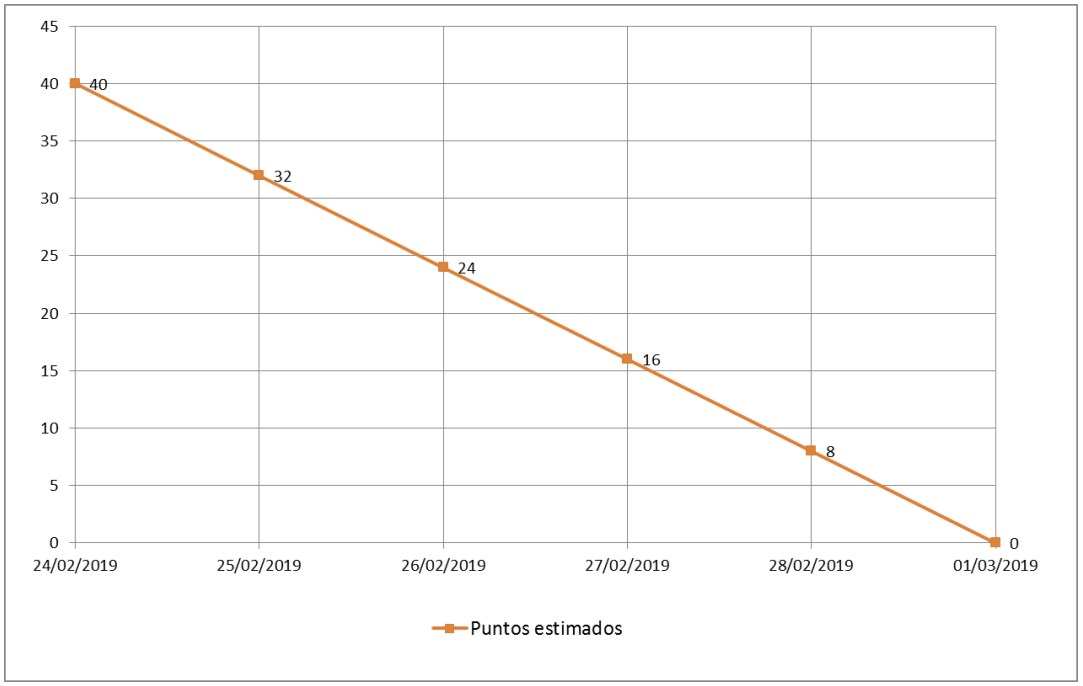


Figura 4.20. *Sprint* 3 – Burndown inicial. **Fuente:** Elaboración propia.

Burndown final

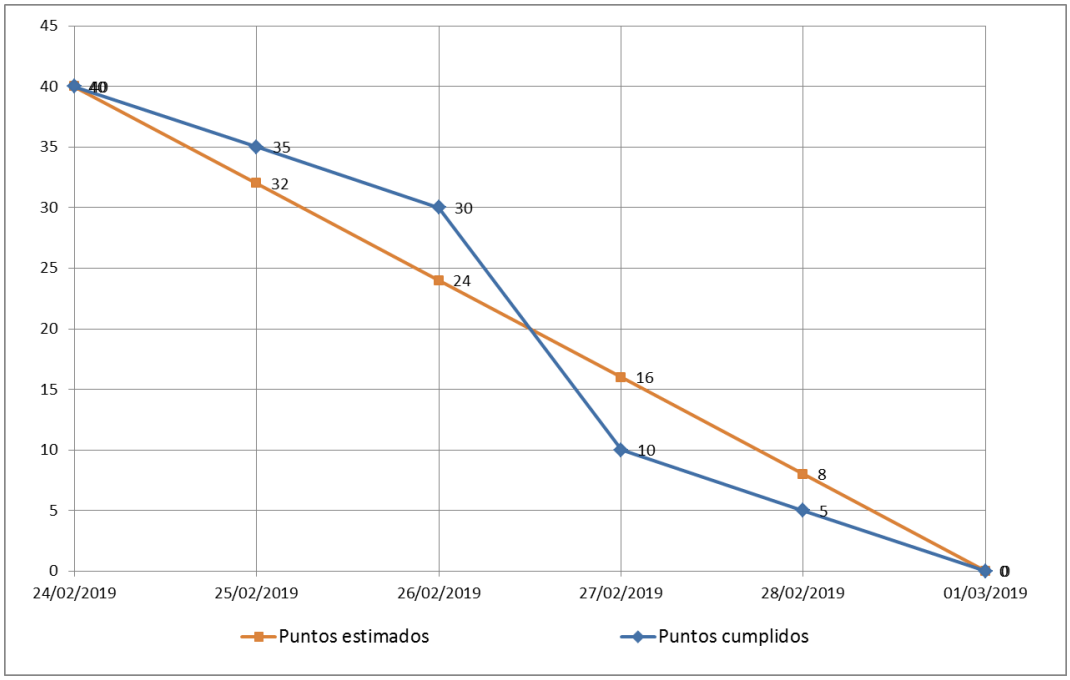


Figura 4.21. *Sprint* 3 – Burndown final. **Fuente:** Elaboración propia.

Resultados

La Figura 4.20 muestra la planificación inicial del *Sprint* 3, la cual consistió en cubrir 40 puntos en 5 días, para implementar el módulo de mensajes.

Se generó un incidente que retrasó el avance del equipo de trabajo como se observa en la Figura 4.21 la línea azul se ubica sobre la línea naranja los días 25 y 26 de febrero de 2019, dicho incidente se presentó porque, fue necesario instalar una librería de *RoR* para que los mensajes se vean inmediatamente si el otro usuario está conectado al sistema. La librería presento un conflicto de versiones con las otras gemas instaladas en *RoR*, sin embargo, se realizado una investigación de las versiones anteriores de esta gema y finalmente se procedió a instalar y cambiar los métodos que no son compatibles con las versiones ya instaladas.

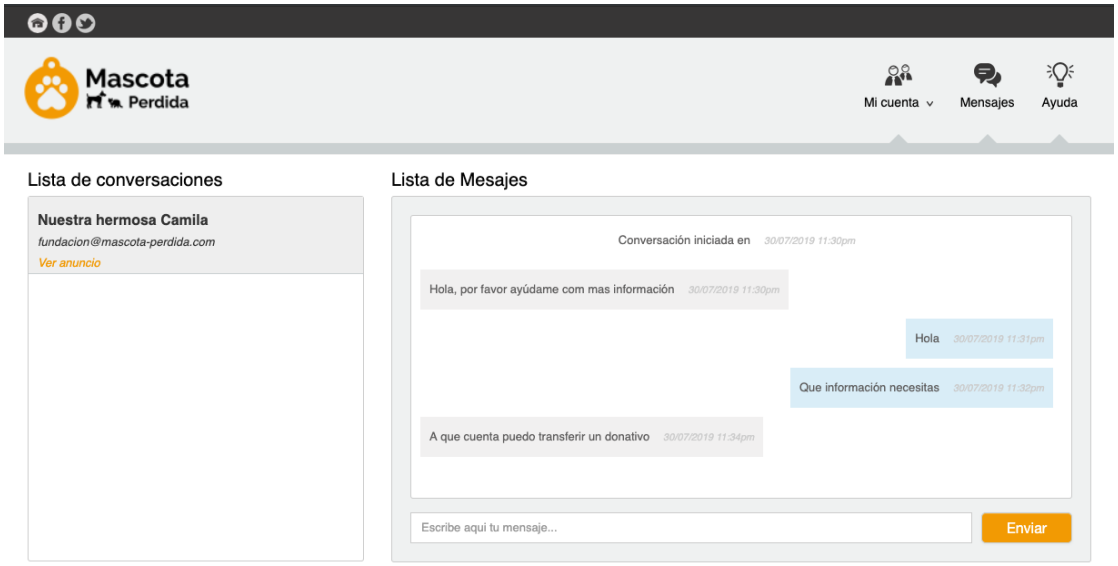


Figura 4.22. Sprint 3 – Lista de conversaciones. **Fuente:** Elaboración propia.

Retroalimentación

El 2 de marzo del 2019 se acordó una reunión para mostrar el resultado final del módulo de mensajes, el cliente se mostró satisfecho con la demostración que consistió en crear una conversación y enviar mensajes entre el usuario administrador y el usuario aprobador.

Sprint 4 - Módulo de panel sectorizado

Tabla 4.19. ***Detalle Sprint 4***

Detalle Sprint 4	
Objetivo:	Implementar el panel sectorizado
Duración:	11/03/2019 – 15/03/2019
Historias por implementar:	RF09, RF10
Tiempo de ejecución:	5 días laborables
Puntos totales	40

Fuente: Elaboración propia

Para completar este ciclo se dispuso a crear solo una tarea debido a que el resultó final fue una sola pantalla con un mapa que permite ver la ubicación de los anuncios por sector en el D.M. de Quito.

A continuación, se lista las tareas que fueron necesarias para completar el ciclo.

Tabla 4.20. ***Tarea 1 – Sprint 4***

Tarea: Implementar panel sectorizado		
Número: 1	**Historia**: RF08	**Puntos:** 40
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: El sistema debe contener una pantalla donde se pueda observar los anuncios por sector, de esta forma los usuarios pueden localizar los anuncios más cercanos. • Instalar librería *JavaScript* de *Google Maps* • Configurar mapa para mostrar anuncios • Configurar marcadores del mapa • Implementar filtros por ubicación		

Fuente: Elaboración propia

Burndown inicial

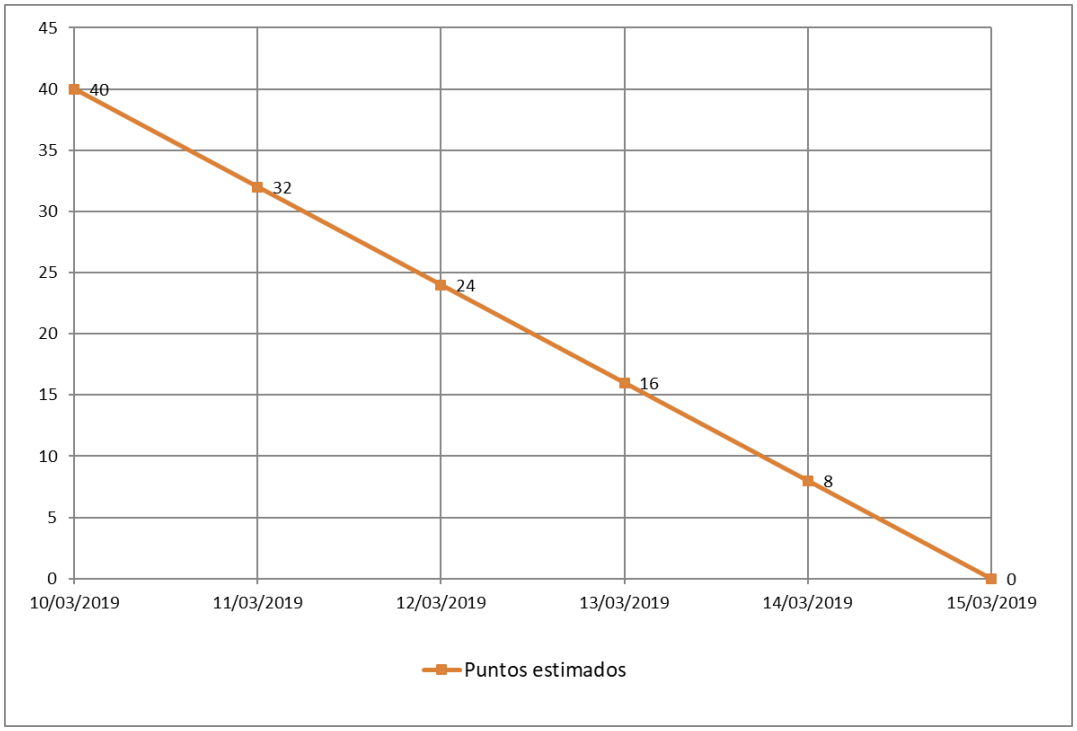


Figura 4.23. *Sprint* 4 – Burndown inicial. **Fuente:** Elaboración propia.

Burndown final

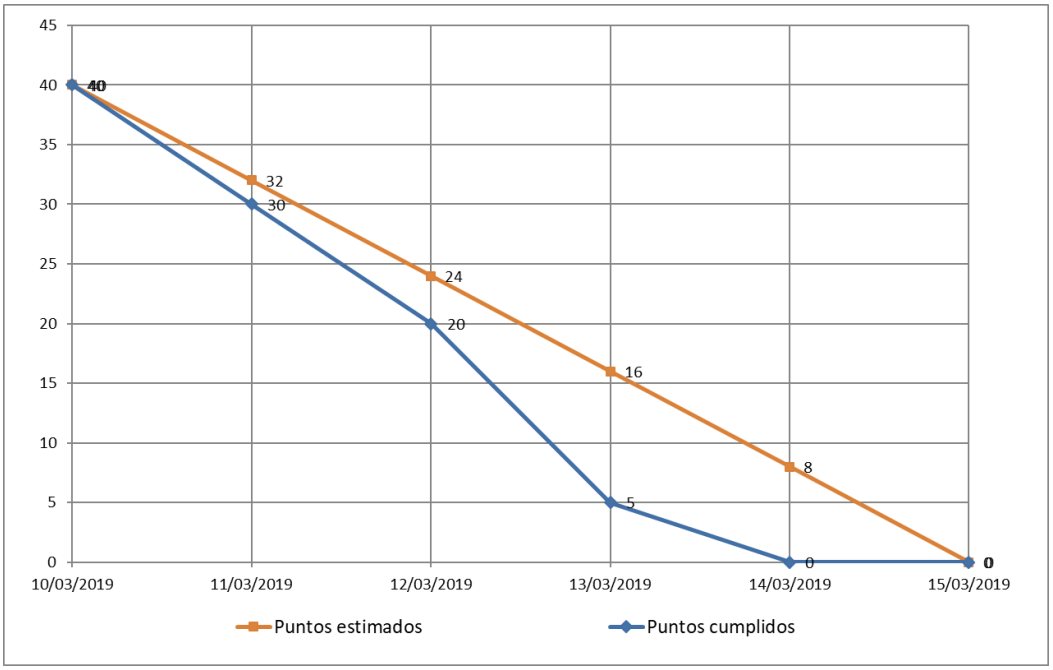


Figura 4.24. *Sprint* 4 – Burndown final. **Fuente:** Elaboración propia.

Resultados

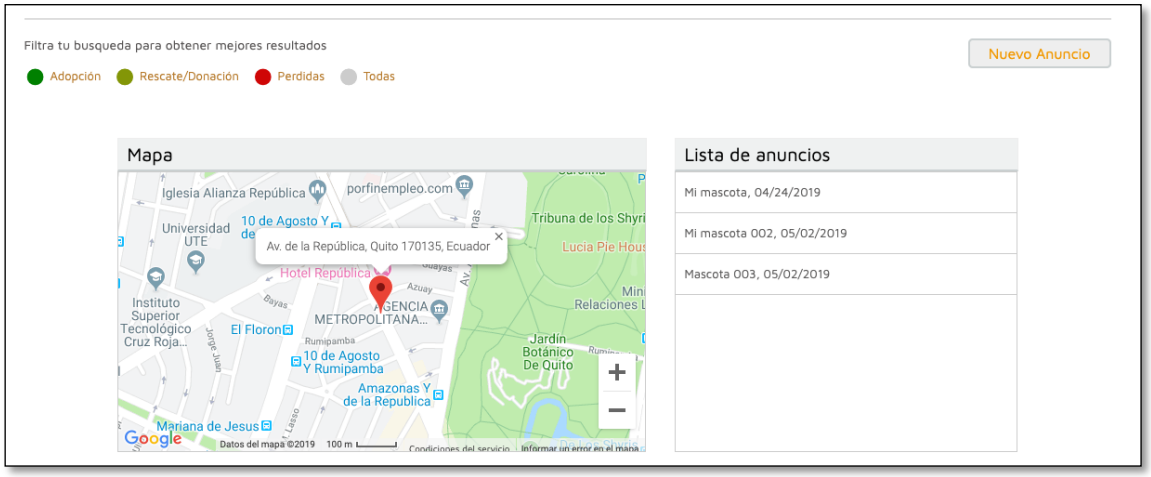


Figura 4.25. *Sprint* 4 – Mapa sectorizado. **Fuente:** Elaboración propia.

Retroalimentación

En el *Sprint* 4 no se presentó incidentes, y el equipo de trabajo pudo completar todos los puntos un día antes del planificado inicialmente, esto fue aprovechado para adelantar un día la reunión con el cliente, la demostración consistió en mostrar los anuncios por sector y moverse entre los marcadores del mapa que representan a los anuncios.

Sprint 5 - Diseño ajustable y navegabilidad

Tabla 4.21. ***Detalle Sprint 5***

Detalle *Sprint* 5	
Objetivo:	Implementar diseño ajustable y navegabilidad
Duración:	18/03/2019 – 22/03/2019
Historias a implementar:	RF01
Tiempo de ejecución:	5 días laborables
Puntos totales	60

Fuente: Elaboración propia

Para acceder a los módulos del sistema fue necesario crear un menú de navegación, y un diseño ajustable para que los usuarios puedan visualizar las pantallas del sistema en sus dispositivos móviles.

Para cumplir con el diseño ajustable se utilizó el marco de trabajo "*Bootstrap*", con ayuda de su sistema de rejillas es posible mostrar una pantalla ajustable al tamaño del dispositivo, además de mostrar, ocultar, mover, cambiar de tamaño, cambiar de posición bloques de HTML. Esta herramienta es de libre acceso, respaldo por el soporte de Twitter quien es el desarrollador. A continuación, se lista las tareas que fueron necesarias para completar el ciclo.

Tabla 4.22. ***Tarea 1 – Sprint 5***

Tarea: Implementar menú de navegación		
Número: 1	**Historia**: RF01	**Puntos:** 30
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Para un fácil acceso las funcionalidades del sistema, se debe implementar un menú con enlaces a los módulos del sistema.		

Fuente: Elaboración propia

Tabla 4.23. ***Tarea 2 – Sprint 5***

Tarea: Implementar funcionalidad responsiva		
Número: 2	**Historia**: RF01	**Puntos:** 30
Responsable Desarrollo:	Marcelo Toapanta	
Responsable Calidad:	Cristina Calderón	
Descripción: Todas las pantallas implementadas para el sistema deben ser ajustables al tamaño de las pantallas de dispositivos móviles como celulares y tablets.		

Fuente: Elaboración propia

Burndown inicial

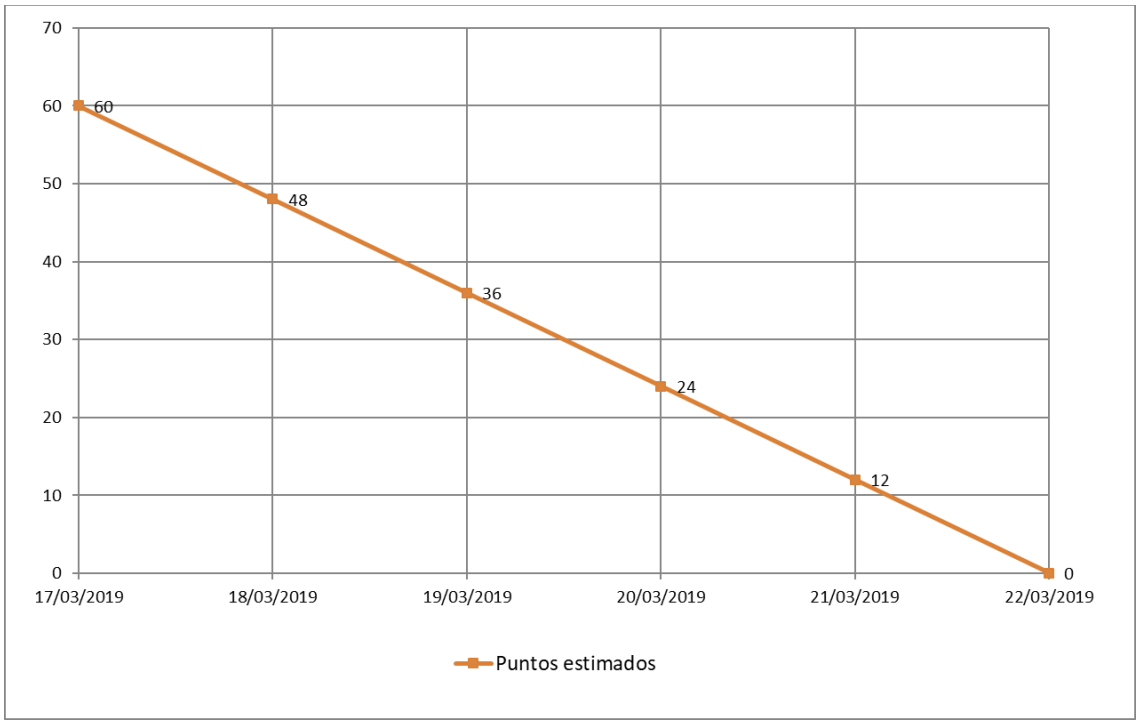


Figura 4.26. *Sprint* 5 – Burndown inicial. **Fuente:** Elaboración propia

Burndown final

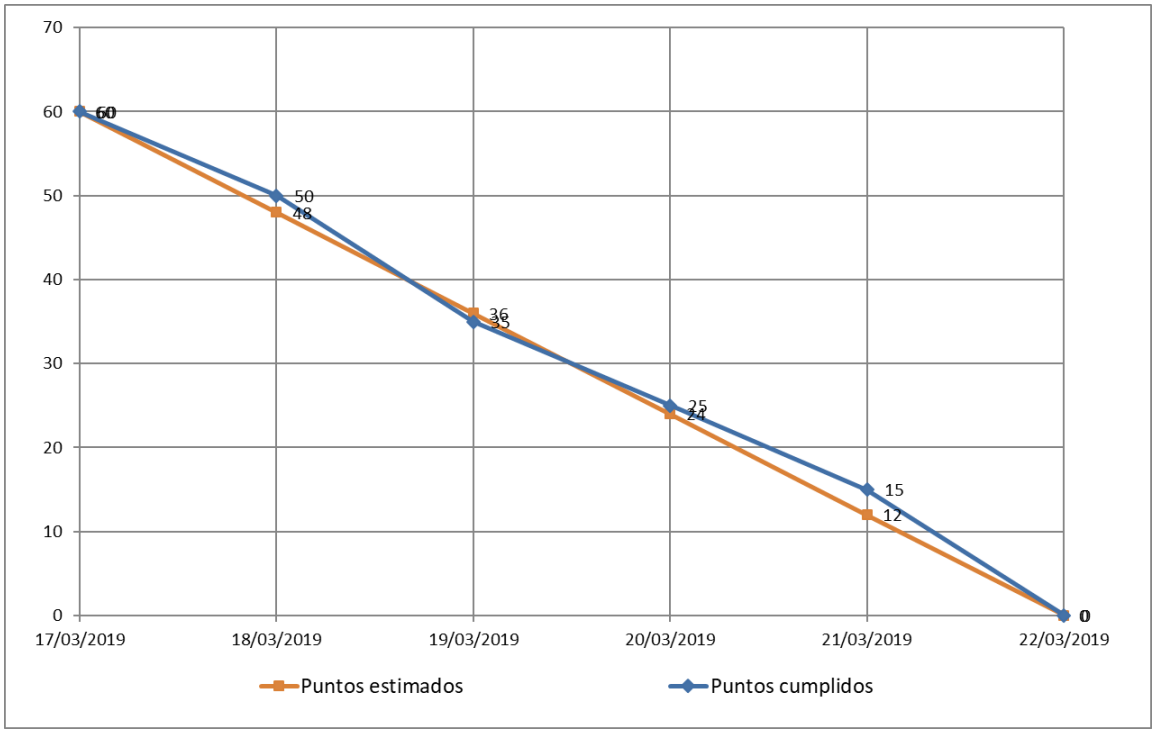


Figura 4.27. *Sprint* 5 – Burndown – final. **Fuente:** Elaboración propia

Resultados

La Figura 4.27 muestra que el equipo de trabajo termino con los puntos en el tiempo planificado inicialmente, no existió incidentes, como se puede ver la línea azul siempre estuvo cerca de la línea naranja, es decir los puntos se completaron cada día acorde la planificación.

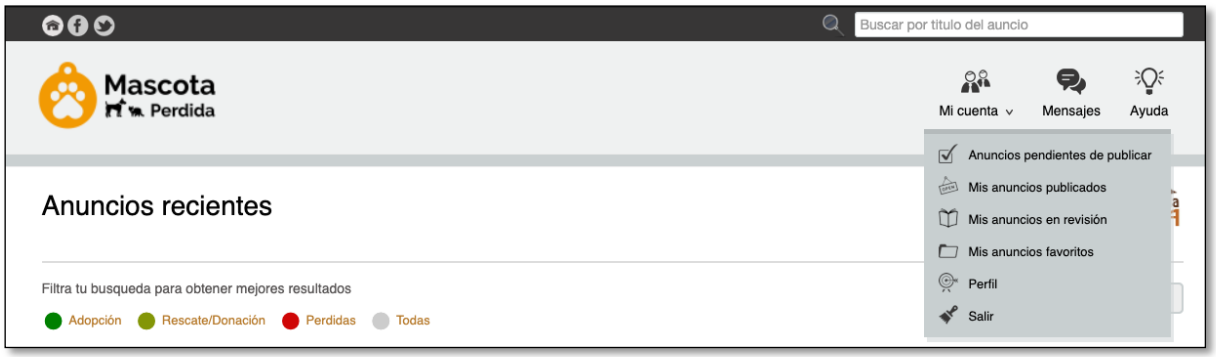


Figura 4.28. *Sprint* 5 – Menú de navegación. **Fuente:** Elaboración propia.

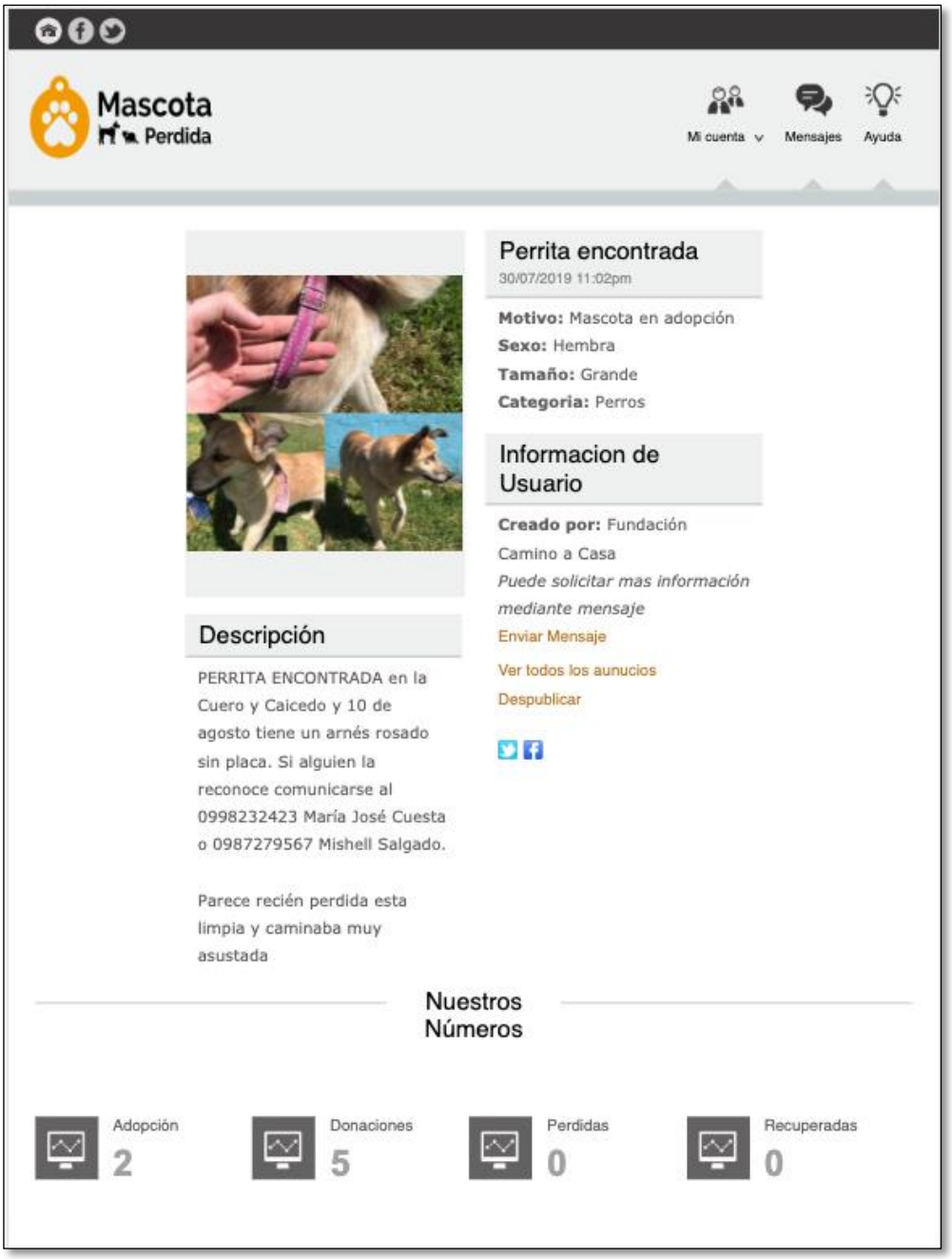


Figura 4.29. *Sprint* 5 – Vista de *iPad*/Tableta. **Fuente:** Elaboración propia.

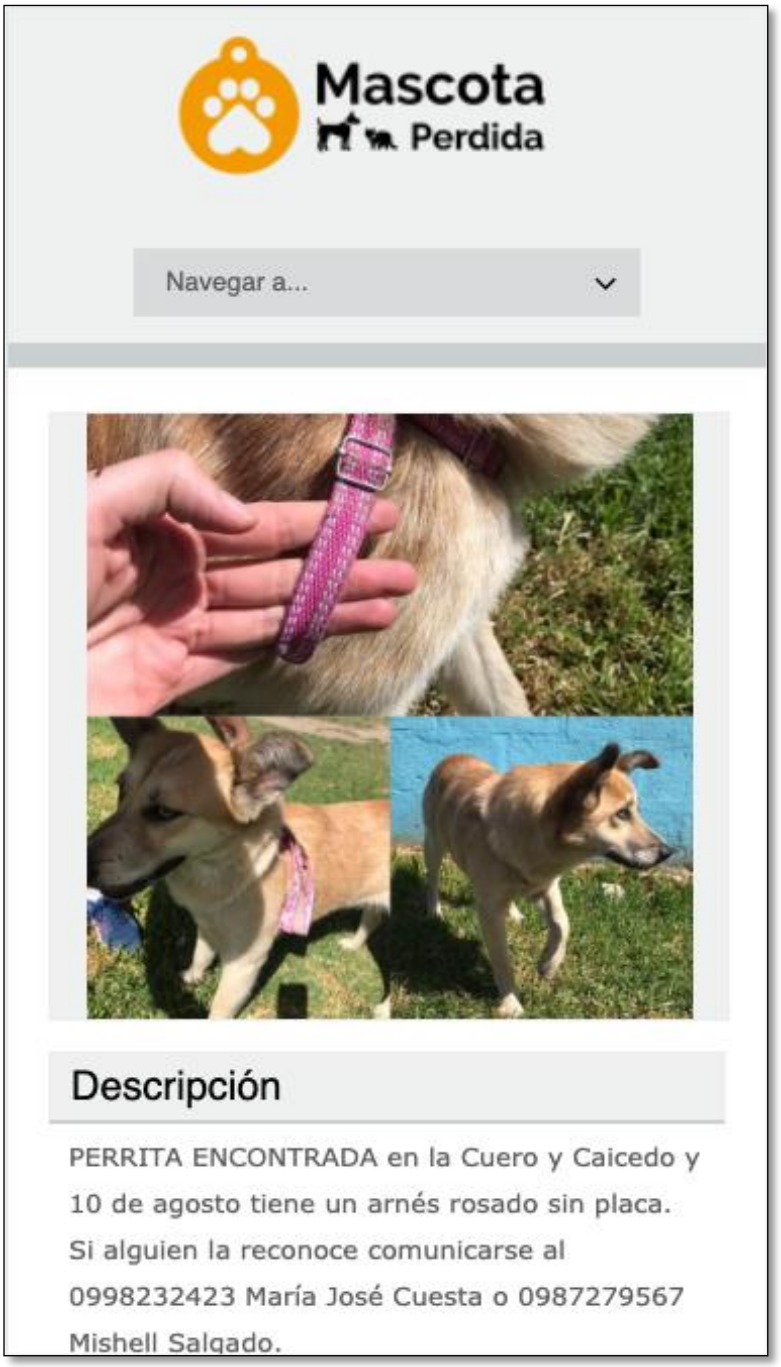


Figura 4.30. *Sprint* 5 – Vista de celular. **Fuente:** Elaboración propia.

Retroalimentación

En la reunión con el cliente, se mostró el funcionamiento del menú de navegación, y como las pantallas se ajustan a diferentes tamaños de dispositivos, como se puede ver en la Figura 4.29 y Figura 4.30 el diseño ajustable funciona correctamente para un celular y una Tablet. La demostración se realizó en el simulador del navegador Chrome.

4.2 Diagrama de la arquitectura del sistema

RoR utiliza una arquitectura MVC para el desarrollo de sistemas web, dicha arquitectura fue detalla en el Capítulo 1, sección herramientas técnicas, ver Figura 1.1.

4.3 Estándares de programación utilizados

Los estándares de programación usados en este proyecto fueron proporcionados por Ruby, los cuales son listados a continuación (Ponce Moreno, 2013):

Estándares de Ruby para nombrado

- **Snake case:** para nombrar métodos, todas las letras en minúscula y las palabras separadas por barra baja, ejemplo: “snake_case”.
- **Camel case:** para nombrar clases y módulos, primera letra de cada palabra en mayúscula y todas las palabras juntas, ejemplo: “CamelCase”.
- **Screaming snake case:** para nombrar constantes, todas las letras en mayúsculas y las palabras separadas por barra baja, ejemplo: “SCREAMING_SNAKE_CASE”.

Estándares de Ruby para formato

- Usar identación o sangría de 2 espacios (No usar Tabs).
- Usar espacios alrededor de los operadores, después de las comas, después de dos puntos, después de punto y coma, alrededor de {y antes de}.
- No usar espacios después de (, [ni antes de],).
- Mantén las líneas con menos de 80 caracteres.

Estándares de Ruby para sintaxis

- Usar **def** con paréntesis cuando tenga argumentos.
- Usar **&&** y || para expresiones booleanas.
- Usar **and** y **or** para flujos de control.
- Evitar **return** donde no se requiera.

Estándares de Ruby para base de datos

Las tablas de la base de datos usan **snake_case**, los nombres de las tablas son plurales, los nombres de columna en la base de datos usan **snake_case**, pero generalmente son singulares.

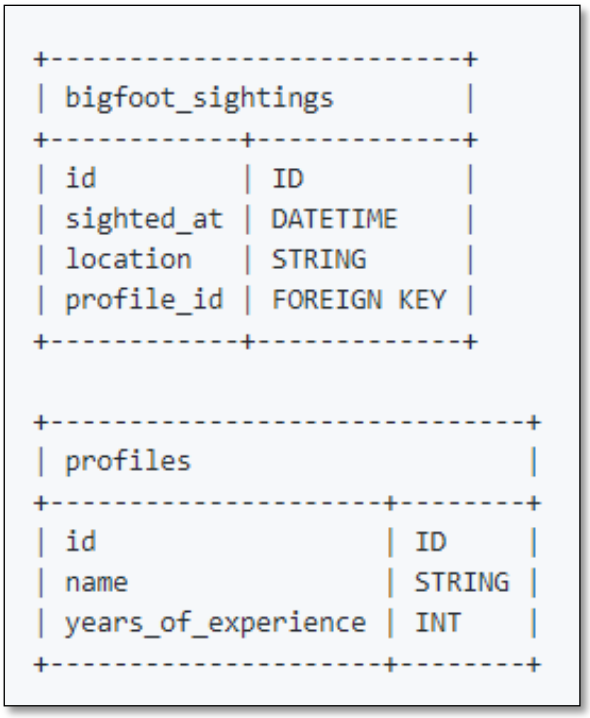


Figura 4.31. *Estándar base de dato.* **Fuente:** Elaboración propia.

Las relaciones entre tablas usan **snake_case**, por lo que **has_one** y **belong_to** son singulares, es decir se usa para relaciones uno a uno, mientras que **has_many** es plural para relaciones uno a muchos. Rails espera que las claves externas en la base de datos tengan un sufijo **_id**, las relaciones con esas claves son gestionadas automáticamente si los nombres se alinean (Young, 2017).

- **belongs_to** :profile, para relación uno a uno requerida
- **has_one** :user, para relación uno a uno opcional
- **has_many** :bigfoot_sightings, para relación uno a muchis

Uso de estándares de programación

Los nombres de clase de controlador usan **CamelCase** y tienen ***Controller*** como sufijo. El sufijo del controlador siempre es singular. El nombre del recurso suele ser plural.

Las acciones del controlador usan **snake_case** y generalmente coinciden con los nombres de ruta estándar (*index, show, new, create, edit, update, delete*).

Tabla 4.24. ***Lista de controladores***

Lista de Controladores	
Nombre	**Acciones**
AdsController	*index, show, new, create, edit, update, delete, search*
UsersController	*create_profile, sign_up, sign_in, sign_out, password*
ConversationsController	*index, show, new, create, edit, update, delete*
ProfilesController	*index, show, edit, update*
MessagesController	*index, show, new, create, edit, update, delete*
PhotosController	*show, new, create, delete*

Fuente: **Elaboración propia**

Los nombres de archivos de vista son asignados de forma predeterminada, coinciden con el nombre de controlador y la acción a la que están vinculados.

Formato de asignación **/views/#{resource_name}/#{action_name}.html.erb**.

- app/views/ads/index.html.erb
- app/views/ads/show.html.erb
- app/views/profiles/show.html.erb

4.4 Pruebas

En esta sección se recapitula las pruebas realizadas al sistema, dentro de las cuales tenemos:

4.4.1 Pruebas de funcionalidad (Aceptación de usuario)

Las pruebas de aceptación del usuario se muestran en la retroalimentación descrita al final de cada *Sprint* por el cliente y sus resultados mostrados en figuras de cada acción realizada.

4.4.2 Pruebas de rendimiento (Aceptación técnica)

El equipo de trabajo realizo las pruebas de rendimiento en un servidor para denostaciones de la plataforma Heroku, dichas pruebas fueron agrupadas en módulos para demostrar su rendimiento dentro de los parámetros descritos y aceptados por el cliente, ver Anexo 8.

4.4.3 Pruebas de carga y estrés (Aceptación técnica)

Las pruebas de carga y estrés muestran la capacidad del sistema para almacenar información y la concurrencia de sesiones activas que soporta.

Estas pruebas fueron realizadas en un servidor de pruebas en la plataforma *Heroku*, ver Anexo 9.

4.5 Implementación

En esta sección se detalla los pasos que se realizó para la implementación del sistema.

4.5.1 Plan de implementación

A continuación, se detalla los recursos utilizados para la implementación.

Tabla 4.25. ***Plan de implementación***

Plan de implementación		
Tarea	**Fecha**	**Recursos**
Crear cuenta en Heroku	25/03/2019	Internet, Macbook Pro
Crear aplicación para RoR	25/03/2019	Internet, Macbook Pro
Sincronizar repositorio del proyecto con Heroku	25/03/2019	Internet, Macbook Pro, Git
Desplegar el código en Heroku	25/03/2019	Internet, Macbook Pro
Ejecutar migración de datos para crear base de datos	25/03/2019	Internet, Macbook Pro
Iniciar sesión y probar el sistema	25/03/2019	Internet, Macbook Pro

Fuente: **Elaboración propia**

Los pasos realizados en la implementación se describen en la página oficial de Heroku, esta plataforma es capaz de alojar sistemas desarrollados en Ruby, PHP, Java, *Node*, *Python*, *Go*, *Scala*. Para ver una demostración de cómo desplegar un proyecto RoR visitar el siguiente enlace: https://www.heroku.com/ruby.

4.5.2 Requerimientos de implementación

En esta sección se describe los requerimientos de Software y Hardware necesarios para un proyecto RoR con una base de datos PostgreSQL.

Software para servidor

- PostgreSQL versión 11.
- Servidor Web Nginx.
- Servidor de aplicaciones Puma.
- Gemas de *RoR*.

Software para usuario.

- Navegador web, Chrome 41 o superior, Firefox 57 o superior, Edge 41 o superior, Safari 6 o superior.

Hardware para servidor

- RAM al menos de 4Gb.
- Procesador Core I7.
- Tarjeta de Red 1Gbps.
- Disco duro de 500Gb

Hardware para usuario

- RAM al menos de 1Gb.
- Tarjeta de Red 100 Mbps.

4.5.3 Manual de usuario

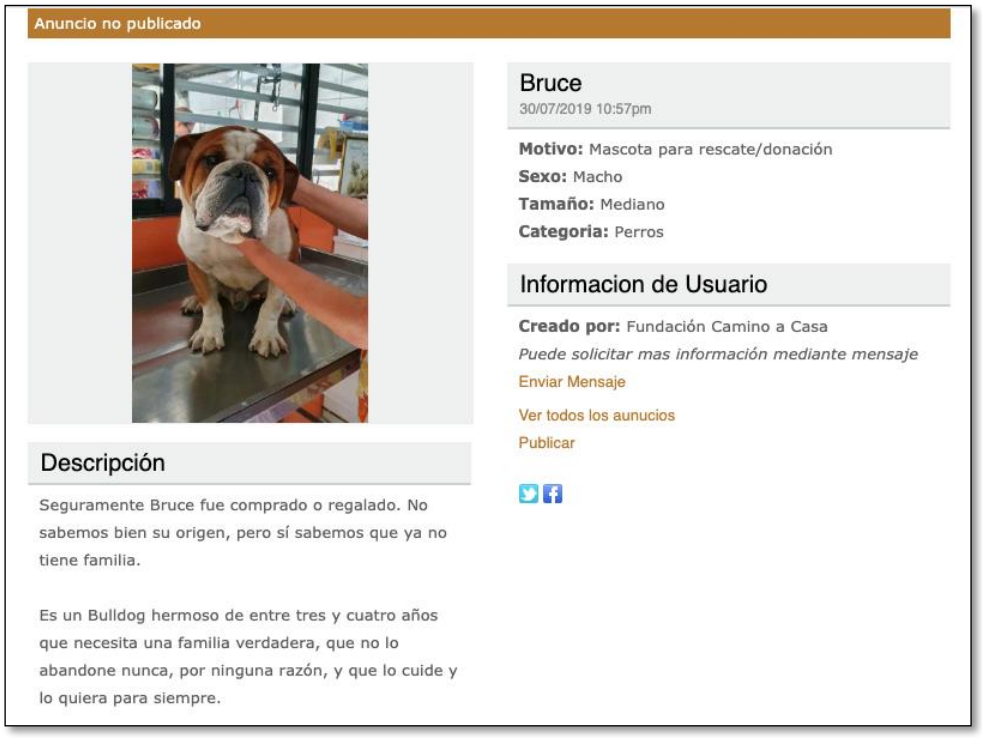
El manual de usuario se describe en el Anexo 6.

4.5.4 Manual técnico

El manual de usuario se describe en el **Ocultar anuncios**

1. El usuario creador del anuncio puede entrar al a la pantalla detalle del anuncio y debe dar clic en el siguiente enlace.

2. El anuncio no se mostrará más en la búsqueda o en la lista de los anuncios, es decir no será visible para los demás usuarios.



Anexo 7.

4.5.5 Plan de capacitación

Acorde a *Scrum* se realizó una demostración del funcionamiento de cada módulo al final de cada *Sprint*, de tal forma el cliente fue capacitado en el flujo de proceso del sistema, Para los usuarios seguidores que se interesen en ser usuarios del sistema y publicar anuncios se creó un enlace en el menú de navegación hacia el manual de usuario, el mismo detalla los pasos para realizar las principales funciones del sistema.

CONCLUSIONES

El sistema web de gestión de adopciones y rescates provee a los seguidores de la fundación la comodidad de examinar los anuncios de la fundación a través de dispositivo con acceso a Internet con seguridad y confianza.

Mediante la metodología *Scrum* el equipo de trabajo realizó controles planificados y sistemáticos para mitigar incidentes en el desarrollo del proyecto, El cliente mantuvo reuniones con el equipo de trabajo para realizar demostraciones con los resultados en cada ciclo y a su vez recibir retroalimentaciones para las futuras iteraciones, con el fin de mejorar la calidad del producto y el tiempo de producción. De tal forma, se concluye la metodología *Scrum* fue una elección acertada debido a que cumplió los objetivos planificados para el proyecto.

Las herramientas RoR y PostgreSQL, de libre acceso en el mercado para plataformas web, permitieron la implementación de los requisitos funcionales y no funcionales de forma exitosa, a través de proceso y métodos simples que permiten a los desarrolladores centrarse en la lógica del negocio, o solución al problema de investigación.

El registro de las adopciones y rescates satisfactorios permite al aprobador dar a conocer e incentivar el pensamiento positivo de adopciones responsables en la comunidad seguidora de la fundación.

El panel sectorizado proporciona a los seguidores de la fundación la comodidad de buscar anuncios en un sector específico, gracias a esto los seguidores pueden encontrar los anuncios de su interés de forma rápida y efectiva.

RECOMENDACIONES

Se sugiere implementar una aplicación móvil para aumentar el alcance de este proyecto y aumentar la cantidad de seguidores de la fundación.

Se recomienda implementar un sistema de recompensas a los usuarios que aporten más con la fundación y el sistema de gestión.

Desarrollar campanas mensuales en lugares públicos de las macotas rescatadas para reducir el tiempo de participación da la fundación, lo que se traduce en reducir gastos.

Implementar un sistema de patrocinio de mascotas, con un pago mensual, lo cual sería de gran ayuda para las mascotas que permaneces a cargo de la fundación por mucho tiempo como mascotas de edad avanzada.

REFERENCIAS BIBLIOGRÁFICAS

Castellano, G. (17 de Agosto de 2018). *El comercio*. Obtenido de https://www.elcomercio.com/narices-frias/infracciones-tenencia-mascotas-quito-agenciadecontrol.html

Chaffe, J., & Swedberg, K. (2013). *Learning jQuery*. Birmingham: Packt Publishing Ltd.

Cuello, J., & Vittone, J. (2013). *Diseñando apps.* Barcelona: Catalina Duque Giraldo.

Duckett, J. (2011). *HTML Y CSS.* Indianapolis: John Wiley & Sons, Inc.

EcuaRed. (20 de Septiembre de 2011). *EcuaRed*. Obtenido de https://www.ecured.cu/Agile_Unified_Process

Gutiérrez, R. (2006). *Introducción al Método científico.* México: Esfinge.

Hartl, M. (2016). *Ruby on Rails Tutorial.* New York: Derek Sivers.

Krall, C. (2006). *JavaScript desde cero.* Barcelona: Aprender a programar.

Monroy, A. (06 de Mayo de 2018). *El telégrafo*. Recuperado el 05 de 01 de 2019, de https://www.eltelegrafo.com.ec/noticias/quito/1/quito-abandono-animales

Moreno, S. P. (2003). *Desarrollo práctico de aplicaciones web.* Madrid: RC Libros.

Perez, J. (2018). *BlockBliss*. Recuperado el 1 de Junio de 2019, de https://blockbliss.com/es/que-es-mvc/

Ponce Moreno, S. (2013). *Desarrollo práctico de aplicaciones web.* Madrid: RC Libros.

RubyGems.org. (s.f.). *RubyGems*. Obtenido de https://rubygems.org/

Sabariego, M., Dorio, I., & Massot, M. (2004). *Características generales de la investigación cualitativa.* Madrid: La Muralla.

The Group PostgreSQL Global Development. (20 de Junio de 2019). *PostgreSQL*. Recuperado el 1 de Junio de 2019, de https://www.postgresql.org/about/

Young, I. (15 de Enero de 2017). *Rails naming conventions*. Recuperado el 20 de Enero de 2019, de https://gist.github.com/iangreenleaf/b206d09c587e8fc6399e

ANEXOS

Anexo 1 – Código Orgánico del Ambiente, Art. 139

SECCION I DISPOSICIONES GENERALES PARA EL MANEJO RESPONSABLE DE LA FAUNA URBANA

Art. 139.- Objeto. El presente capítulo tiene por objeto la promoción y la garantía del bienestar animal, a través de erradicar la violencia contra los animales, fomentar un trato adecuado para evitarles sufrimientos innecesarios y prevenir su maltrato, y de aplicar y respetar los protocolos y estándares derivados de instrumentos internacionales reconocidos por el Estado.

La tenencia de animales conlleva la responsabilidad de velar por su bienestar, y su manejo deberá promover una relación armoniosa con los seres humanos.

Anexo 2 – Código Orgánico Integral Penal, Art. 249 y 250

Artículo 249.- Maltrato o muerte de mascotas o animales de compañía. - La persona que por acción u omisión cause daño, produzca lesiones, deterioro a la integridad física de una mascota o animal de compañía, será sancionada con pena de cincuenta a cien horas de servicio comunitario. Si se causa la muerte del animal será sancionada con pena privativa de libertad de tres a siete días.

Se exceptúan de esta disposición, las acciones tendientes a poner fin a sufrimientos ocasionados por accidentes graves, enfermedades o por motivos de fuerza mayor, bajo la supervisión de un especialista en la materia.

Artículo 250.- Peleas o combates entre perros. - La persona que haga participar perros, los entrene, organice, promocione o programe peleas entre ellos, será sancionada con pena privativa de libertad de siete a diez días. Si se causa mutilación, lesiones o muerte del animal, será sancionada con pena privativa de libertad de quince a treinta días.

Anexo 3 – Entrevista a la propietaria de la Fundación camino a casa

Fecha: 15/01/2018
Entrevistador: Marcelo Toapanta
Entrevista: Cristina Calderón

Nro.	Pregunta	Respuesta
1	¿Qué función realiza en la fundación?	Actualmente se desempeña como gerente administrativa
2	¿Qué tipo de actividades y servicios da la fundación a la comunidad?	Rescate de animales en estado de vulnerabilidad. Recuperación y adopción de animales
3	¿Cuáles son los animales con los que se trabaja frecuentemente?	Perros y gatos
4	¿Tiene la fundación un sistema de gestión informático para realizar los procesos de adopción?	No tiene ningún sistema de gestión informático, solo redes sociales como Facebook
5	¿Cuál es el proceso de adopción de animales?	Se realiza una visita a la casa del solicitante para conocer el lugar donde vivirá la mascota. Una vez que se verifique el lugar se procede a firmar un acta compromiso y posteriormente se entrega el animalito esterilizado.
6	¿Qué requisitos deben tener las personas interesadas en adoptar?	Ser mayor de edad, caso contrario una autorización firmada por los padres. Ubicación del domicilio Datos personales en general
7	¿Cuánto tiempo tarda en gestionarse una adopción?	Generalmente de 3 a 5 días
8	¿Cómo se almacena la información de los solicitantes?	Toda la información es archivada en carpetas organizadas por fechas.
9	¿Existe alguna normativa o ley del país que regule los pasos para una adopción?	No
10	En caso de que una mascota se encuentre extraviada ¿cómo se procede para reportar estos casos?	Se debe enviar una foto actualizada de la mascota y el lugar exacto de dónde se perdió.

Anexo 4 – Encuesta a los seguidores en Facebook de la Fundación

Fecha: 20/01/2018
Entrevistador: Marcelo Toapanta
Entrevistos: Seguidores en Facebook de la fundación (1624)

Nro.	Pregunta	Respuesta
1	¿Si usted busca adoptar una mascota, prefiere la opción de contactar directamente con la persona que sede la mascota o acudir a la fundación?	Persona (1221), Fundación (403)
2	¿Cómo se entera de las publicaciones de la fundación: Facebook, boca a boca, recomendación?	Facebook (856), Boca a boca (461), Recomendación (307)
3	¿Cómo prefiere que se ordenen las publicaciones de la fundación: Fecha, tipo de mascota, sector, nombre de la mascota?	Sector (1003), Fecha (271), Nombre de mascota (298), Tipo de mascota (52)
4	¿Para reportar el rescate de una mascota prefiere hacerlo por redes sociales o contratar directo con la fundación?	Redes sociales (1387), Fundación (237)
5	¿Por qué medio prefiere aportar con donaciones a la fundación: transacción bancaria, efectivo, PayPal, donación de alimentos?	Transacción bancaria (684), PayPal (612), donación de alimentos (235), efectivo (93)
6	¿Tiene acceso a internet en su casa?	Si (1482), No (142)
7	¿Tiene acceso a internet desde su teléfono celular?	Si (842), No (782)
8	¿Cuál es el navegador de internet más usado por usted: Firefox, Chrome, Microsoft Edge, Safari?	Chrome (933), Firefox (367), Edge (236), Safari (88)
9	¿En una publicación de la fundación en cuál de los siguientes elementos presta más su atención, nombre, foto, descripción, ubicación?	Foto (1020), Ubicación (402), Descripción (153), Nombre (49)

Anexo 5 – Historias de usuario

HISTORIA DE USUARIO	
Número: RF01	**Tipo de usuario**: Todos
Nombre historia: Menú principal	
Prioridad en negocio: Baja	**Prioridad en desarrollo**: Media
Puntos estimados: 60	**Iteración asignada:** Navegación y diseño
Descripción: Para facilitar la navegación de los seguidores de la fundación en el sistema debe existir un menú de fácil acceso, debe ser fácil de encontrar y mostrar las opciones según el tipo de usuario del sistema.	
Observación: Los usuarios registrados tendrán la opción de escribir mensajes a otros usuarios, cambiar los datos de su perfil, lista de anuncios por publicar y lista de anuncios favoritos.	

HISTORIA DE USUARIO	
Número: RF02	**Tipo de usuario**: Seguidores
Nombre historia: Registrar usuario	
Prioridad en negocio: Media	**Prioridad en desarrollo**: Media
Puntos estimados: 15	**Iteración asignada:** Gestión de usuario
Descripción: Los seguidores de la fundación deben tener la opción de registrarse en el sistema para poder usar las funciones del mismo.	
Observación: Los datos necesarios para el registro en el sistema son: correo electrónico y una contraseña de mínimo 6 caracteres.	

HISTORIA DE USUARIO

Número: RF03 **Tipo de usuario**: Administrador, Aprobador, Anunciante

Nombre historia: Gestión de usuario

Prioridad en negocio: Media **Prioridad en desarrollo**: Alta

Puntos estimados: 30 **Iteración asignada:** Gestión de usuario

Descripción: El usuario debe poder gestionar su cuenta y sus sesiones seguras. Deben tener una pantalla para ingresar sus datos, adicional pueden administrar su perfil de usuario para poder comunicarse con los otros usuarios.

Observación: La información del usuario debe ser almacenado de forma segura, y debe ser visualizada en los anuncios con autorización del usuario.

HISTORIA DE USUARIO

Número: RF04 **Tipo de usuario**: Administrador, Aprobador, Anunciante

Nombre historia: Gestión de anuncios

Prioridad en negocio: Alta **Prioridad en desarrollo**: Alta

Puntos estimados: 40 **Iteración asignada:** Gestión de anuncios

Descripción: La gestión de anuncios para los usuarios del sistema deber tener las siguientes opciones:

- Crear/Editar anuncios
- Listar anuncios
- Eliminar anuncios
- Ver anuncios

Observación: Todos los anuncios deben contener información acerca de las mascotas y no representar ningún intento de lucro.

HISTORIA DE USUARIO

Número: RF05 **Tipo de usuario**: Administrador, Aprobador

Nombre historia: Aprobar/Publicar anuncios

Prioridad en negocio: Media **Prioridad en desarrollo**: Baja

Puntos estimados: 10 **Iteración asignada:** Gestión de anuncios

Descripción: Los usuarios aprobadores y administradores deben filtrar los anuncios que contengan información que no cumpla con el propósito de la fundación.

Observación: Los anuncios no deben contener información con: fines de lucro, ofensas a otros usuarios, ofensas a la fundación, publicidad, comercializaciones de mascotas.

HISTORIA DE USUARIO

Número: RF06 **Tipo de usuario**: Administrador, Aprobador, Anunciante

Nombre historia: Panel de anuncios por publicar

Prioridad en negocio: Baja **Prioridad en desarrollo**: Baja

Puntos estimados: 10 **Iteración asignada:** Gestión de anuncios

Descripción: Cada usuario debe tener acceso a la lista de sus anuncios que estén pendientes de publicar, el usuario administrador y aprobador tendrán un lista de anuncios general.

Observación: Los usuarios anunciantes deben tener acceso solo a una lista con sus anuncios y no pueden ver los anuncios de otros usuarios hasta que sean publicados.

HISTORIA DE USUARIO

Número: RF07 **Tipo de usuario**: Administrador, Aprobador, Anunciante, Visitante

Nombre historia: Buscar anuncios

Prioridad en negocio: Media **Prioridad en desarrollo**: Media

Puntos estimados: 10 **Iteración asignada:** Gestión de anuncios

Descripción: Los usuarios deben tener acceso a una pantalla para buscar anuncios por su nombre o título.

Observación: Los anuncios listados solo deben ser los que ya han sido publicados, esta pantalla debe tener filtros de tipo de anuncio.

HISTORIA DE USUARIO

Número: RF08 **Tipo de usuario**: Administrador, Aprobador, Anunciante

Nombre historia: Contactar anunciante

Prioridad en negocio: Media **Prioridad en desarrollo**: Alta

Puntos estimados: 30 **Iteración asignada:** Gestión de anuncios

Descripción: Los usuarios deben tener acceso a una pantalla para contactar directamente con el usuario creador del anuncio, es decir puede crear una conversación para poder pedir información por interno.

Observación: Solo los usuarios registrados podrán crear conversaciones.

HISTORIA DE USUARIO

Número: RF09 **Tipo de usuario**: Administrador, Aprobador, Anunciante, Visitante

Nombre historia: Panel sectorizado

Prioridad en negocio: Media **Prioridad en desarrollo**: Alta

Puntos estimados: 30 **Iteración asignada:** Panel sectorizado

Descripción: Los usuarios deben tener acceso a una pantalla con un mapa de los anuncios filtrados por sector, es decir debe mostrar una lista de anuncios según se muestre el mapa.

Observación: Todo los usuarios podrán ver el mapa.

HISTORIA DE USUARIO

Número: RF10 **Tipo de usuario**: Administrador, Aprobador, Anunciante, Visitante

Nombre historia: Compartir en redes sociales

Prioridad en negocio: Media **Prioridad en desarrollo**: Alta

Puntos estimados: 30 **Iteración asignada:** Gestión de anuncios

Descripción: Debe existir la posibilidad de compartir en anuncio en Facebook y Twitter.

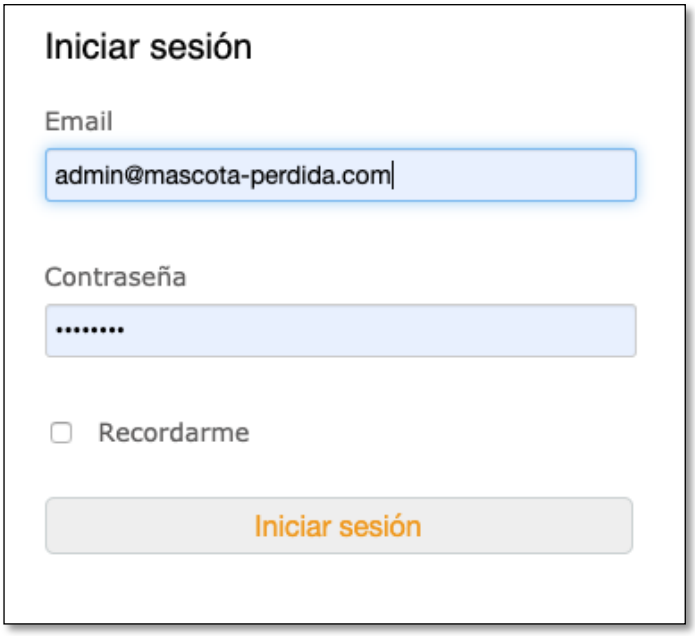
Observación: Todo los usuarios podrán realizar esta acción.

Anexo 6 - Manual de usuario

Crear anuncio

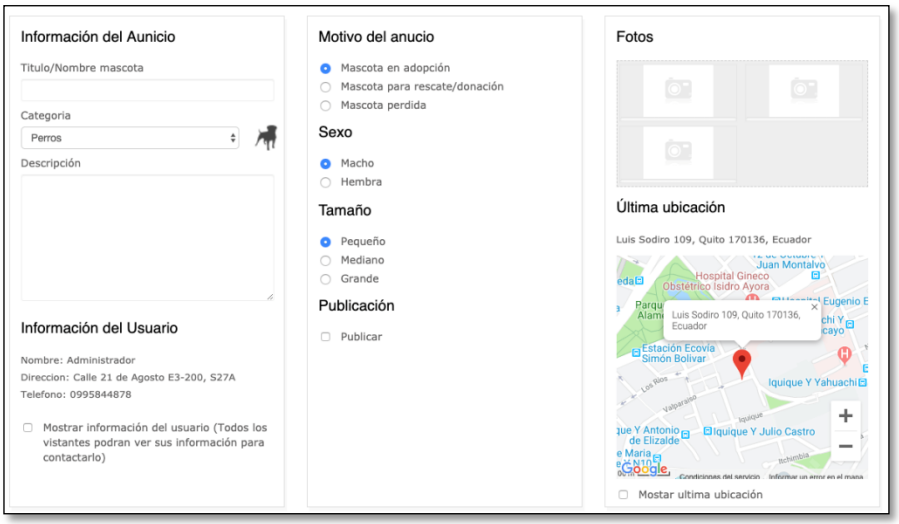
1. Iniciar sesión en el sistema, clic en el botón iniciar.

2. Ingresar datos del usuario.



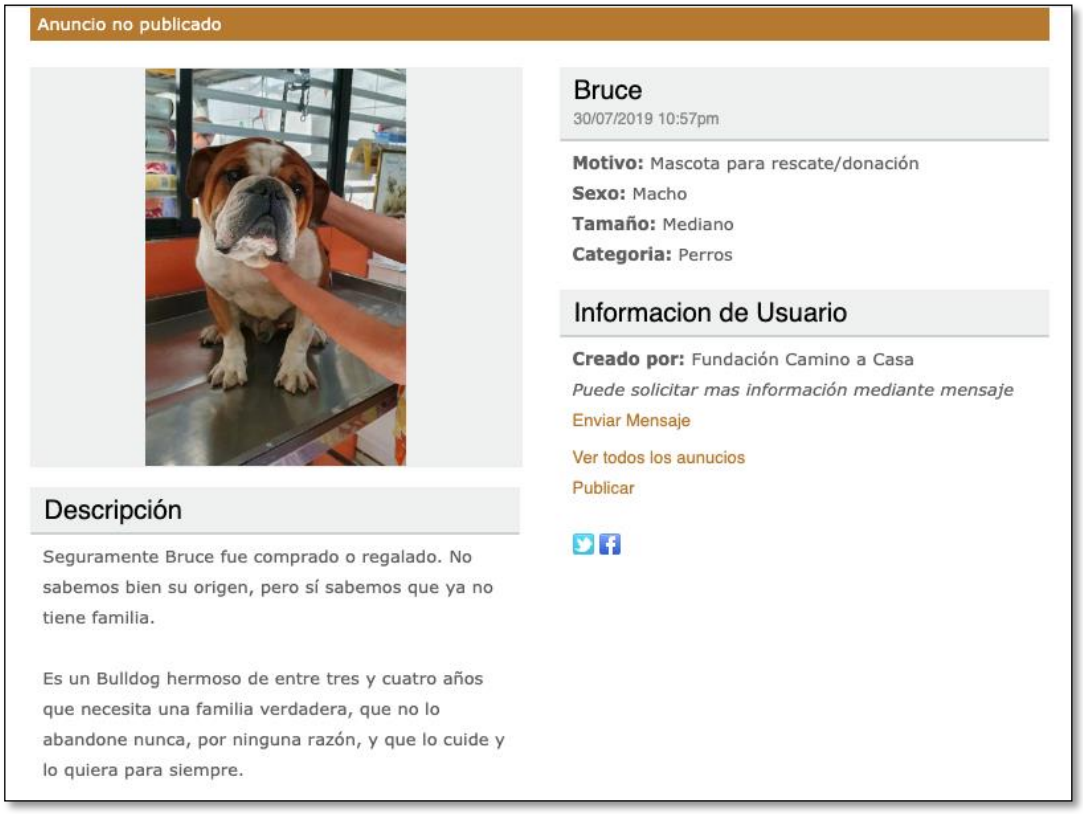
3. En la página principal clic en Nuevo anuncio

4. Se desplegará el siguiente formulario para ingresar la información importante de la mascota y el motivo del anuncio.



5. Clic en Publicar, el anuncio se mostrará en la lista de revisión.

6. Se desplegará la pre-visualización del anuncio.

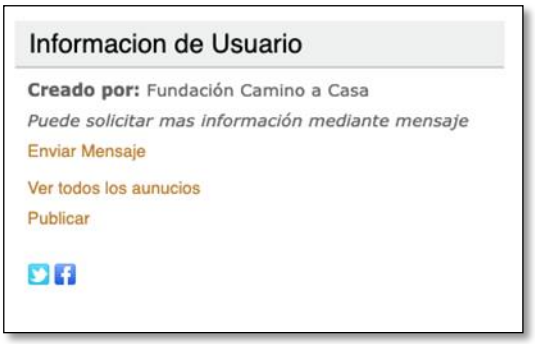


Publicar anuncio (Usuario aprobador)

1. Clic en el icono “Mi cuenta”

2. Clic en el enlace “Anuncios pendientes de publicar”

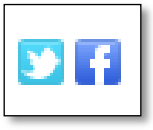
3. Seleccionar el anuncio que desea publicar, se desplegará el anuncio.



4. En la sección “Información de Usuario” Clic en publicar.

Publicar en redes sociales

1. Solucionar el anuncio, clic en icono de Facebook o Twitter.



Enviar mensajes

1. Seleccionar el anuncio, clic en el enlace “Enviar mensaje”.

2. Seleccionar conversación de la lista que se encuentra a la izquierda.

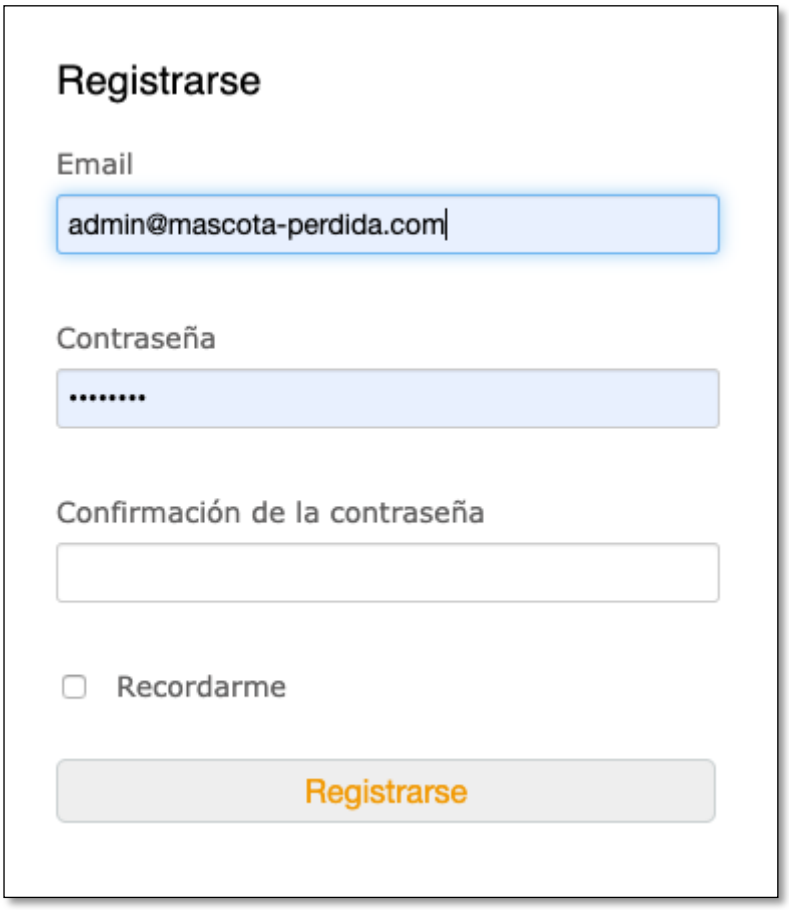
3. Escribir mensaje en el cuadro de texto inferior derecho, clic en “Enviar”.

4. Para ver todas las conversaciones clic en el icono de “Mensajes” en el menú de navegación

Registrarse en el sistema

1. Ingresar al sistema al sitio web y clic en “Registrar”.

2. Ingresar un email válido y una contraseña de mínimo 6 caracteres.



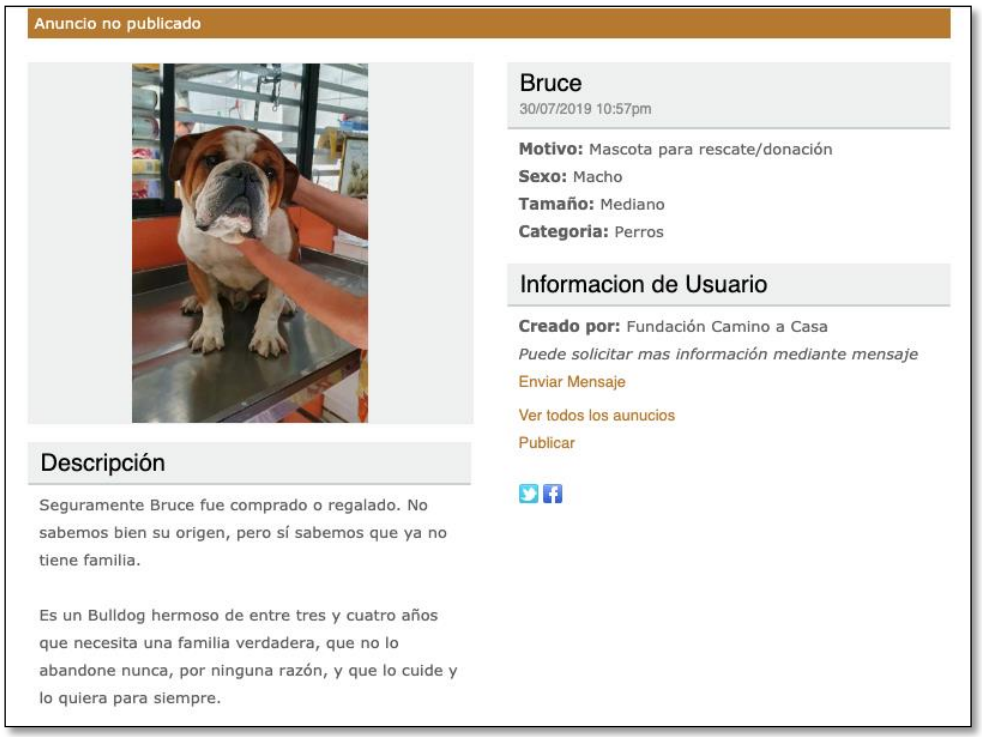
Menú de ayuda

1. Para acceder a este manual clic en el icono de “Ayuda”.

Ocultar anuncios

3. El usuario creador del anuncio puede entrar al a la pantalla detalle del anuncio y debe dar clic en el siguiente enlace.

4. El anuncio no se mostrará más en la búsqueda o en la lista de los anuncios, es decir no será visible para los demás usuarios.



Anexo 7 - Manual técnico

Objetivo del manual

El objetivo del presente manual es orientar en los conocimientos técnicos relacionados con las tecnologías implementadas en este proyecto, también detallar los pasos a seguir para la puesta en funcionamiento del software y sus componentes.

Instalación y despliegue del sistema

Para instalar el sistema que se obtuvo como resultado de este trabajo en un servidor de Heroku se deben seguir los siguientes pasos:

1. En el terminal, inicie sesión con la dirección de correo electrónico y la contraseña que utilizó al crear su cuenta Heroku.
   ```
   $ heroku login
   ```
2. En la terminal, instale las gemas especificadas en el ***Gemfile***
   ```
   $ bundle install
   ```
3. Asegúrese de que **config/database.yml** esté utilizando el **postgresql adaptador**.
   ```
   production:
     <<: *default
     adapter: postgresql
     database: db/production
   ```
4. Subir tus cambios en Git:
   ```
   $ git add .
   $ git commit -m "Heroku config"
   ```
5. En la terminal, crea una aplicación en Heroku:
   ```
   $ heroku create
   ```
6. Empuje su código a Heroku:
   ```
   $ git push heroku master
   ```
7. Ejecutar migraciones de la base de datos ejecutando:
   ```
   $ heroku run rake db:migrate
   ```
8. Obtenga la URL de su aplicación y visítela en el navegado
   ```
   $ heroku apps:info
   ```

Diccionario de datos

A continuación se presentan la definiciones y descripciones de los datos que van a ser utilizados en el aplicativo. Inicialmente se nombra cada entidad con su descripción y tabla con los campos y sus características especiales.

Tabla Usuario

users				
Campo	**Tipo de dato**	**Obligatorio**	**C. primaria**	**C. foránea**
id	int	Si	X	
email	varchar	Si		
encrypted_password	varchar	Si		
role	int	Si		
created_at	timestamp	No		
updated_at	timestamp	No		

Tabla Perfil

profiles				
Campo	**Tipo de dato**	**Obligatorio**	**C. primaria**	**C. foránea**
id	int	Si	X	
name	varchar	Si		
direction	varchar	No		
phone	varchar	No		
enews	boolean	No		
total_ads	int	No		
user_id	int	Si		X
created_at	timestamp	No		
updated_at	timestamp	No		

Tabla Categorías

categories				
Campo	**Tipo de dato**	**Obligatorio**	**C. primaria**	**C. foránea**
id	int	Si	X	
name	varchar	Si		
total_ads	int	No		
order	int	No		
created_at	timestamp	No		
updated_at	timestamp	No		

Tabla Fotos

photos				
Campo	**Tipo de dato**	**Obligatorio**	**C. primaria**	**C. foránea**
id	int	Si	X	
file	varchar	Si		
caption	varchar	No		
credits	varchar	No		X
ad_id	int	No		
created_at	timestamp	No		
updated_at	timestamp	No		

Tabla Anuncios

ads				
Campo	**Tipo de dato**	**Obligatorio**	**C. primaria**	**C. foránea**
id	int	Si	X	
title	varchar	Si		
description	varchar	No		
published	boolean	No		
coordinates	varchar	No		
category_id	int	No		X
lost	boolean	No		
found	boolean	No		
gener	boolean	No		
size	int	No		
views	int	No		
status	varchar	No		
show_user_info	boolean	No		
show_location_info	boolean	No		
created_at	timestamp	No		
updated_at	timestamp	No		

Tabla Mensajes

messages				
Campo	**Tipo de dato**	**Obligatorio**	**C. primaria**	**C. foránea**
id	int	Si	X	
body	varchar	Si		
conversation_id	varchar	No		X
created_at	timestamp	No		
updated_at	timestamp	No		

Tabla Conversaciones

conversations				
Campo	**Tipo de dato**	**Obligatorio**	**C. primaria**	**C. foránea**
id	int	Si	X	
recipient_id	int	Si		X
sender_id	int	Si		X
ad_id	int	Si		X
created_at	timestamp	No		
updated_at	timestamp	No		

Estructura de archivos

Archivo/Carpeta	**Propósito**
app/	Contiene los controladores, modelos y vistas de la aplicación.
config/	Configura las reglas de ejecución de la aplicación, rutas, base de datos y más.
config.ru	Configuración Rack para servidores basados en Rack usados para iniciar la aplicación.
db/	Contiene el esquema actual de tu base de datos, así como las migraciones de la base de datos.
Gemfile Gemfile.lock	Estos archivos te permiten especificar qué dependencias de gemas son necesitadas para tu aplicación Rails. Estos archivos son usados por la gema Bundler.
lib/	Módulos extendidos para tu aplicación.
log/	Archivos de Log de tu aplicación.
public/	La única carpeta vista por el mundo tal como es. Contiene los archivos estáticos y assets compilados.
Rakefile	Este archivo localiza y carga tareas que pueden ser ejecutadas desde la línea de comandos.
script/	Contiene el script de Rails que inicia tu aplicación y contiene otros scripts usados para desplegar o correr tu aplicación.
test/	Pruebas unitarias, fixtures y otras pruebas. Éstos son cubiertos en Testing Rails Applications.
tmp/	Archivos temporales (como archivos de caché, PID y archivos de sesiones).
vendor/	Lugar para código de terceros. En una típica aplicación Rails, ésta incluye librerías y plugins.

Modelos

Los modelos que conforman el sistema son los siguientes

Modelo	Tabla en base de datos
User	users
Profile	profiles
Category	categories
Photos	photos
Ads	ads
Messages	messages
Conversations	conversations

Vistas

Las vistas contienes el código necesario para que los navegadores web interpreten y desplieguen visualmente los formularios para interactuar con el usuario.

Nombre de la vista	Propósito
index.html	Devuelve una lista de objetos.
new.html	Devuelve el formulario para crear un objeto.
create.html	Permite crear un objeto y responde con una redirección.
edit.html	Devuelve el formulario para editar un objeto.
update.html	Permite editar un objeto y responde con una redirección.
detroy.html	Permite eliminar un objeto y responde con una redirección.
show.html	Devuelve un objeto.

Controladores

Los controladores permiten a la vista comunicarse con los modelos.

Nombre del controlador	Métodos	Formatos
UserController	index, new, create, edit, update, destroy, show	html, js
ProfileController	index, new, create, edit, update, destroy, show	html, js
CategoryController	index, new, create, edit, update, destroy, show	html, js
PhotosController	index, new, create, edit, update, destroy, show	html, js
AdsController	index, new, create, edit, update, destroy, show	html, js
MessagesController	index, create	html, js
ConversationsController	index, create, close, add_to_conversations	html, js
ApplicationController	set_basic_data, set_meta_description	html, js

Clases y objetos

ApplicationController

Es la clase principal que permite crear métodos accesibles a todo el proyecto, detro de los cuales encontramos:

- **set_basic_data.-** devuelve todos los anuncios agrupados por motivo y las categorías de los anuncios
- **set_meta_description:** devuelve las etiquetas necesarias para compartir las páginas en redes sociales.
- **record_not_found:** permite redirecciones a una página HTML cuando se genera un error 404, registro no encontrado.

ActiveRecord

ActiveRecord es la capa que nos permite acceder y manipular la información de la base de datos sin necesidad de escribir SQL (Structured Query Language), todos los modelos de la aplicación son heredadas de esta clase.

ActiveRecord::Migration

Las migraciones nos permiten hacer cambios sobre la estructura de la base de datos de forma fácil y consistente, especialmente si son múltiples personas trabajan sobre el mismo proyecto, o el proyecto ya se encuentra en producción. Ejemplo

```
$ rails generate migration AddPriceToAds price:decimal
```

Que generaría la siguiente migración

```
class AddPriceToAds < ActiveRecord::Migration
  def change
    add_column :ads, :price, :decimal
  end
end
```

Aplicación de Consola Rails

Consola que nos permite, entre otras cosas, generar código a través de comandos, podemos crear objetos de las clases que constituyen la aplicación y manipularos desacuerdo a nuestros intereses. Para accede a la consola ingresar el comando:

```
$ rails console
```

Métodos y atributos de clase

Los métodos y atributos están asociados a un objeto.

Métodos de clase.- Un método de clase es muy parecido a un método de instancia, con la diferencia de que el nombre tiene el prefijo self.

```
class User
  def self.mi_metodo_de_clase
    puts "Este es un método de clase"
  end
end
```

Para invocar este método debe hacerlo directamente sobre la clase omitiendo el prefijo self.

```
$ User.mi_metodo_de_clase
```

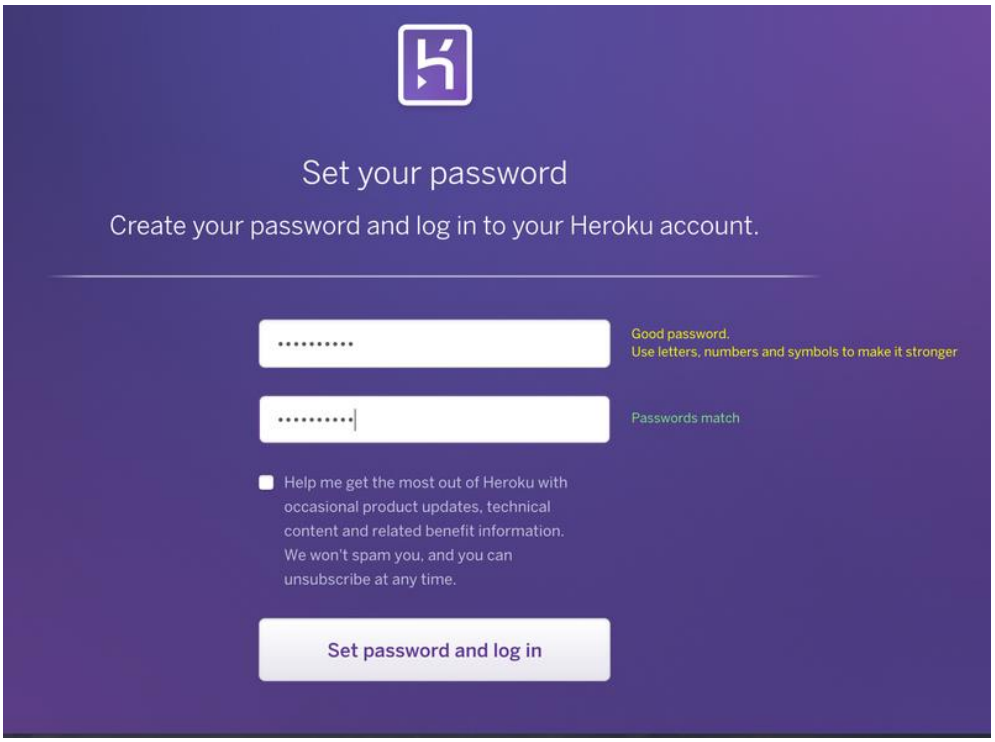
Atributos de clase.- Para contar cuántos objetos se crean a partir de la clase User lo podemos almacenar en un atributo de clase. Los atributos de clase se identifican porque utilizan dos @@ en vez de una.

```
class User
  @@people_count = 0

  def initialize
    @@people_count += 1
  end

  def self.people_count
    @@people_count
  End
end
```

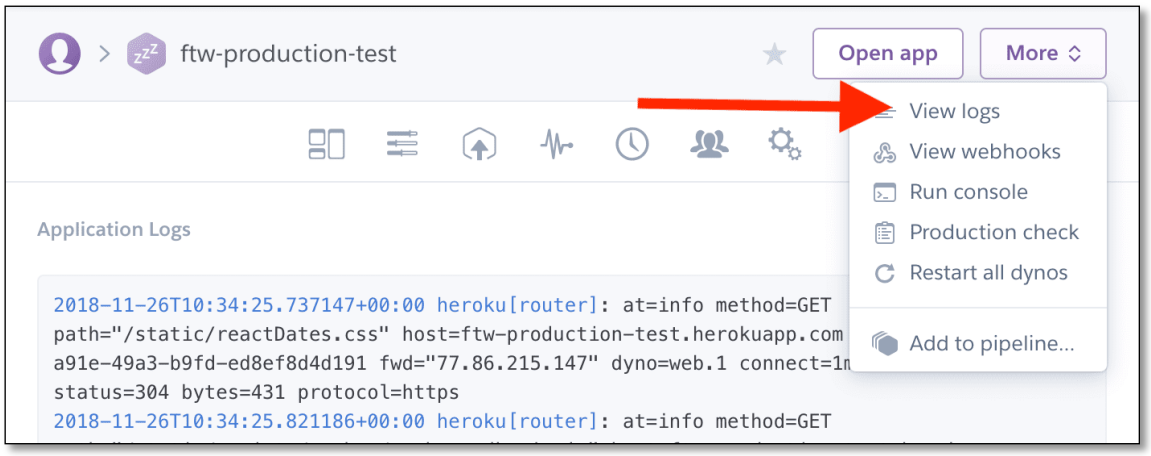
Revisión de bitácora

La plataforma brinda herramientas para monitorear el sistema web.

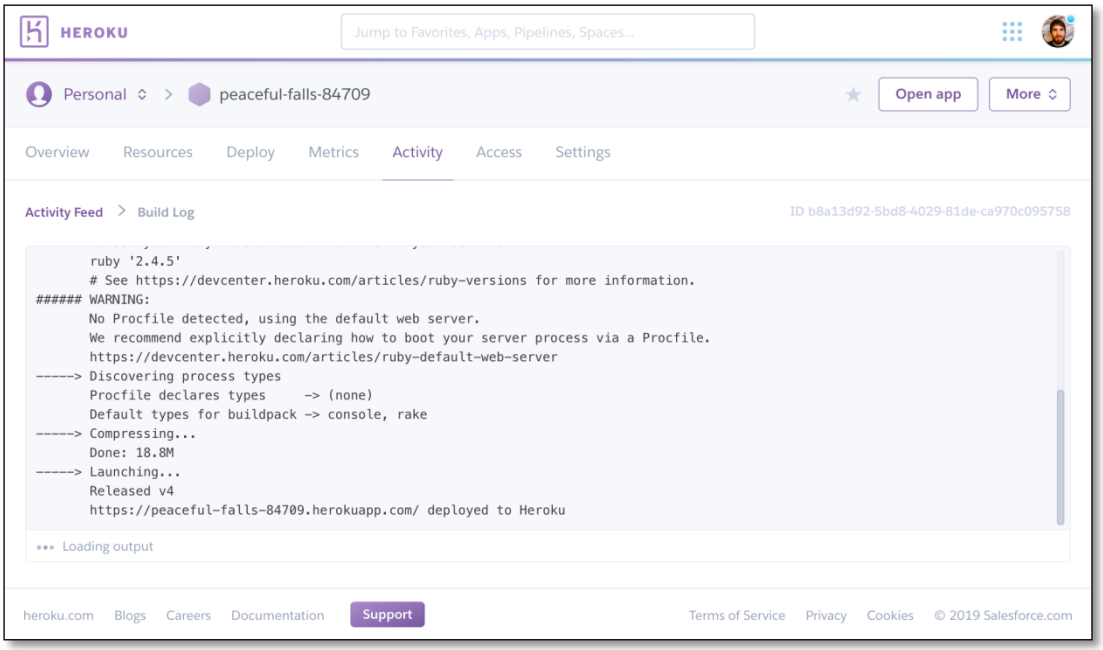
1. Iniciar sesión en heroku.com



2. Seleccionar la aplicación, clic en “View logs”.



3. Se desplegará una pantalla que mostrará las acciones de que se realizan en el sistema web.



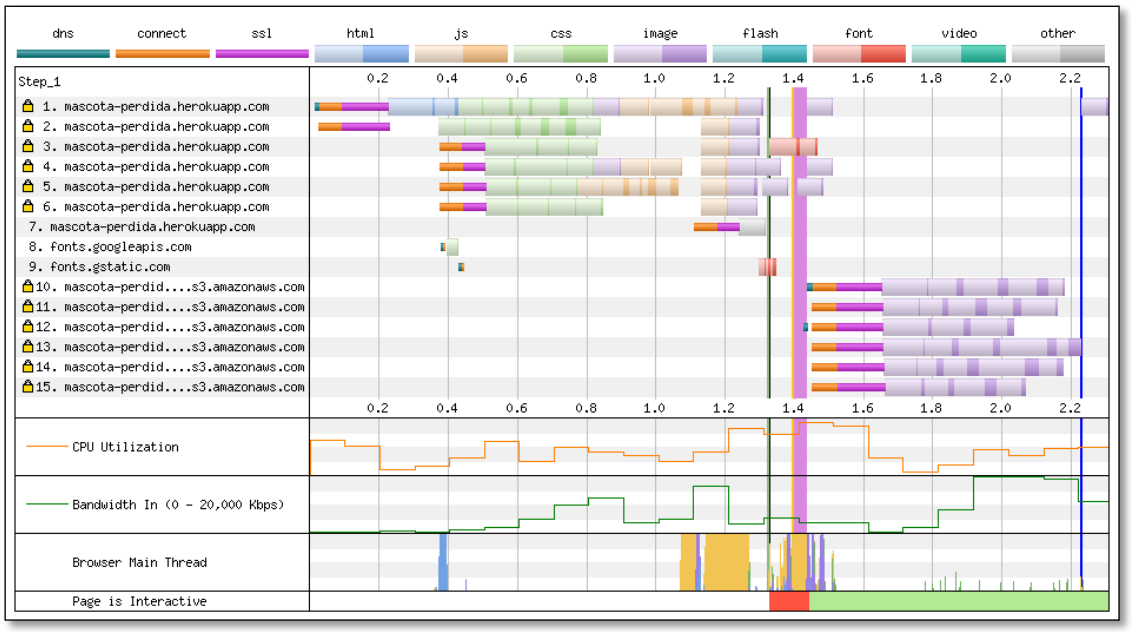
Anexo 8 - Pruebas de rendimiento

En esta sección se puede observar los resultados de las pruebas de rendimiento realizadas, para esto se utilizó las siguientes herramientas en Internet.

- *WebPageTest*, sitio web: https://www.webpagetest.org/
- LoadImpact, sitio web https://loadimpact.com/

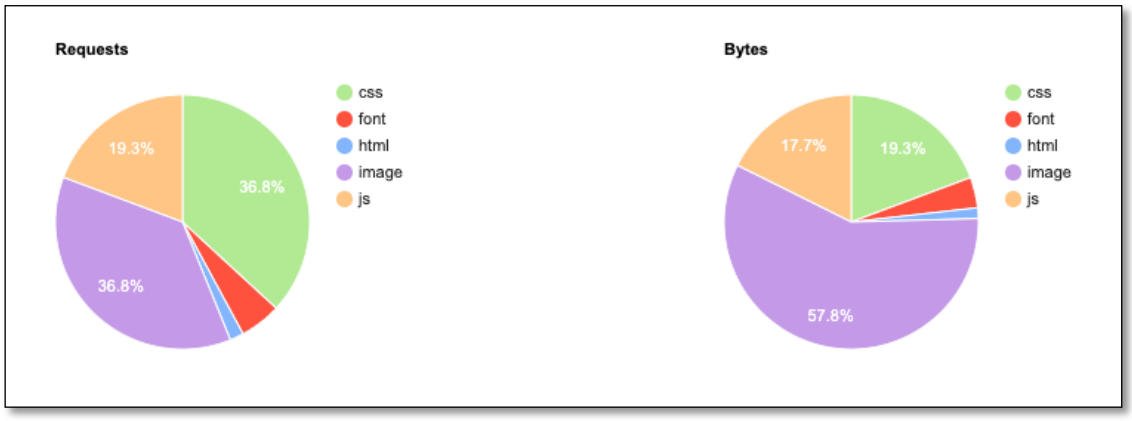
Las pruebas de rendimiento fueron realizadas por el equipo de trabajo en las páginas que se consideran que tendrán mayor concurrencia y se detallan a continuación:

Lista de anuncios



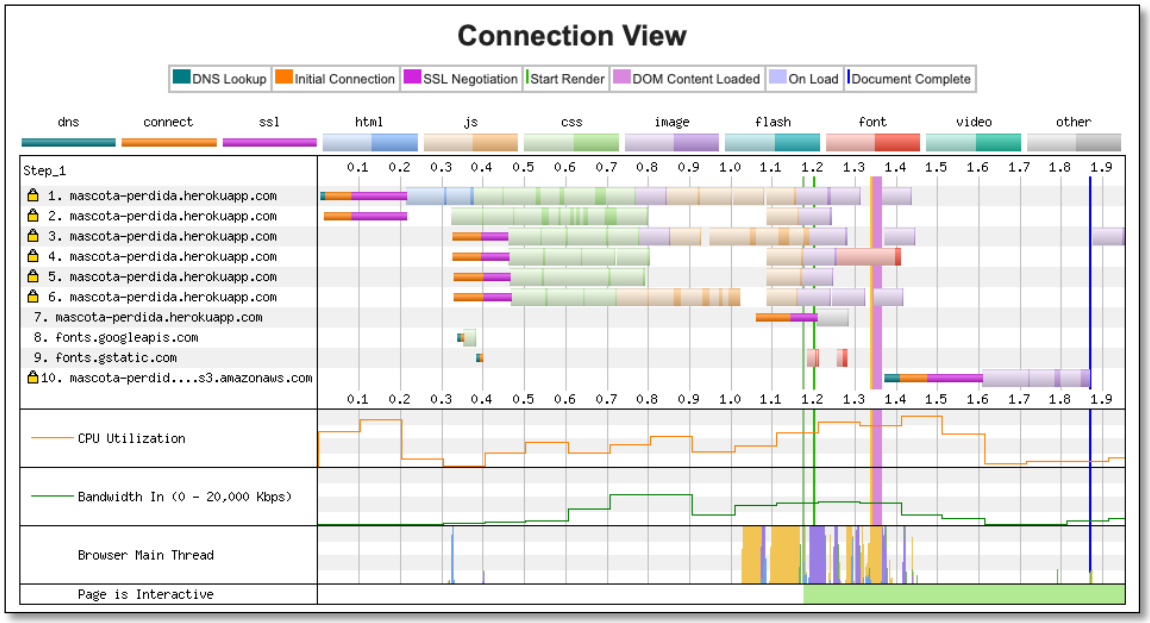
Como se observa en la figura anterior el tiempo para visualizar el documento es 1.4 segundos, tiempo aceptable dentro de los parámetros definidos inicialmente por el equipo de trabajo y el cliente.

Se puede notar que los elementos que consumen mayor cantidad de recursos son las imágenes, seguido de los archivos CSS, sin embargo no representan ningún problema al momento de cargar la página web con todos sus elementos.

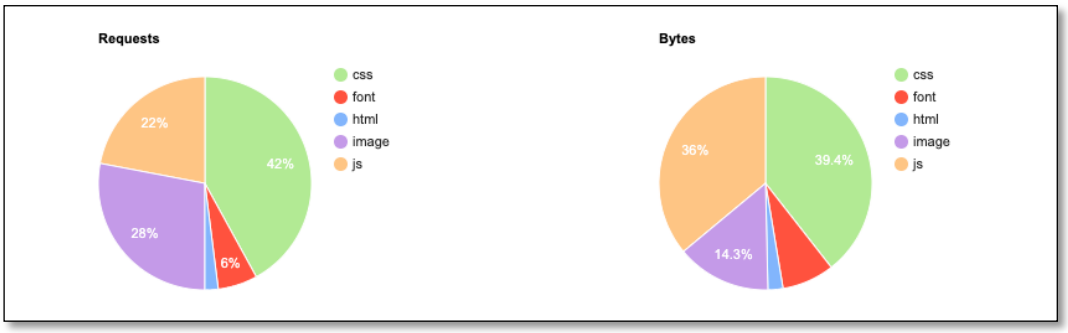


En la figura anterior se muestra los porcentajes de los tipos de elementos que contiene la página, se puede observar que las imágenes representa la mayor cantidad de tráfico.

Búsqueda de anuncios

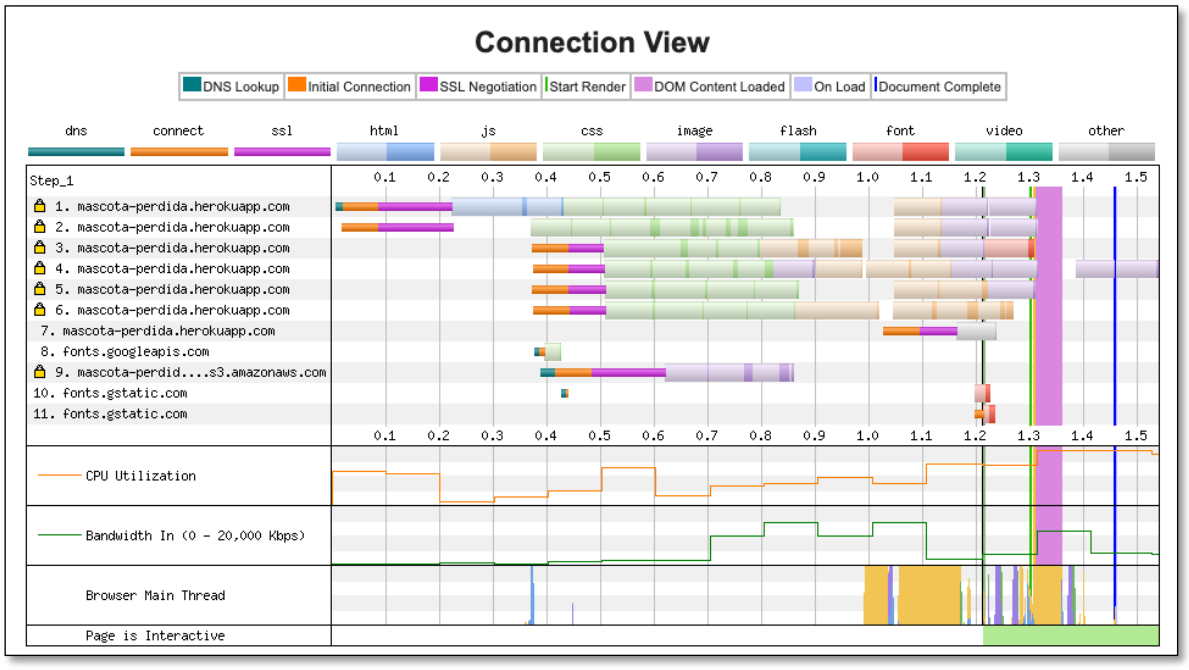


Para la búsqueda de anuncios tenemos un tiempo de carga de 1.9 segundos, de igual forma dentro de los parámetros establecidos inicialmente.

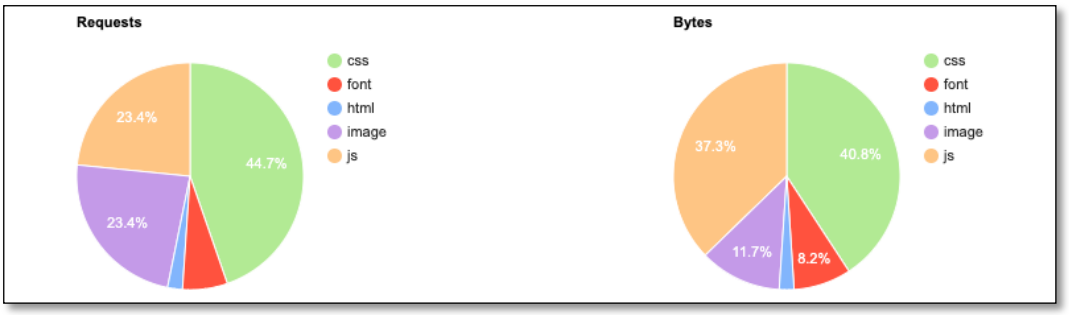


Para la búsqueda de anuncios se puede observar que la mayor cantidad de recursos es usada por los archivos CSS, seguidos por los archivos JS,

Destalles de Anuncio



La página de detalle de anuncio toma 1.5 segundos en estar lista para ser visualizada por los usuarios.



De igual forma que la página de búsqueda de anuncios los elementos que consumir mayor cantidad de recursos son tipo CSS.

Anexo 9 – Pruebas de carga

Las pruebas de carga realizadas por el equipo de trabajo se detallan a continuación.

Lista de anuncios

La página de lista de anuncios muestra un tiempo de carga de 1.4 segundos y un cargar de todos sus elementos de 2.3 segundos, lo cual indica que cumple con los paramentos definidos inicialmente.

Tester: i-0570ed0334dbb718a
First View only

Export HTTP Archive (.har)
Custom Metrics

									Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	First Contentful Paint	Visually Complete	Speed Index	Last Painted Hero	First CPU Idle	Result (error code)	Time	Requests	Bytes In	Time	Requests	Bytes In
2.227s	0.354s	1.400s	1.324s	2.100s	1.511s	1.800s	> 1.445s	0	2.227s	58	1,824 KB	2.304s	59	1,825 KB

Colordepth	domInteractive	domContentLoaded	loadEvent
24	1.393s	1.393s - 1.436s (0.043s)	2.227s - 2.227s (0.000s)

Para las pruebas de estrés se simuló una concurrencia de 25 usuarios simultáneos durante dos minutos, como se observa en la siguiente figura el sistema logro estar disponible para los 25 usuarios y en el tiempo establecido nunca dejó de funcionar o respondió con errores.

Búsqueda de anuncios

La búsqueda de anuncios obtuvo un tiempo de carga de documento de 1.8 segundos y 1.9 segundos de carga completa. Los tiempos de carga se consideran aceptables ya que son menores a los establecidos en los requerimientos iniciales.

Tester: i-0570ed0334dbb718a
First View only

Export HTTP Archive (.har)
Custom Metrics

									Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	First Contentful Paint	Visually Complete	Speed Index	Last Painted Hero	First CPU Idle	Result (error code)	Time	Requests	Bytes In	Time	Requests	Bytes In
1.868s	0.308s	1.200s	1.175s	1.800s	1.237s	1.800s	> 1.174s	0	1.868s	51	894 KB	1.950s	52	895 KB

Colordepth	domInteractive	domContentLoaded	loadEvent
24	1.337s	1.337s - 1.365s (0.028s)	1.867s - 1.867s (0.000s)

De igual forma que en las pruebas anteriores se simuló una concurrencia de 25 usuarios durante dos minutos de forma progresiva, como se observa en la siguiente grafica el servidor siempre respondió correctamente y no hubo caída del servidor o repuestas con errores.

Detalle de anuncio

Finalmente se realizó las pruebas de carga y estrés a la página que muestra los detalles de un anuncio.

En la siguiente figura se detalla los tiempos de carga, dentro de los el documento se cargó en 1.4 segundos y en 1.5 segundos la carga completa con todos sus elementos.

Tester: i-0570ed0334dbb718a
First View only

Export HTTP Archive (.har)
Custom Metrics

									Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	First Contentful Paint	Visually Complete	Speed Index	Last Painted Hero	First CPU Idle	Result (error code)	Time	Requests	Bytes In	Time	Requests	Bytes In
1.458s	0.355s	1.300s	1.213s	1.500s	1.302s	1.500s	> 1.213s	0	1.458s	48	864 KB	1.538s	49	866 KB

Colordepth	domInteractive	domContentLoaded	loadEvent
24	1.307s	1.307s - 1.360s (0.053s)	1.459s - 1.459s (0.000s)

Waterfall View

La prueba de estrés para la página de detalle de anuncio se realizó con una concurrencia de 25 usuarios conectado simultáneamente durante dos minutos, la siguiente figura muestra el desempeño del servidor y evidencia que no falló durante esta prueba, tampoco se reportaron respuestas con errores o no disponibilidad de la página.