APLICACIÓN WEB - PETAPP

DANNY RUBEN GONZÁLEZ
OSCAR ALONSO GONZÁLEZ MUÑOZ

Corporación Universitaria Minuto de Dios
Rectoría Antioquia y Chocó
Sede Bello
Programa Tecnología en Informática
Noviembre de 2020

APLICACIÓN WEB - PETAPP

DANNY RUBEN GONZÁLEZ
OSCAR ALONSO GONZÁLEZ MUÑOZ

Trabajo de Grado presentado como requisito para optar al título de Tecnólogo en Informática y Tecnólogo en Gestión de Redes y Comunicaciones

Asesora:
Laura Valderrama López

Corporación Universitaria Minuto de Dios
Rectoría Antioquia y Chocó
Sede Bello
Tecnología en Informática
Noviembre de 2020

Resumen

La tendencia en la sociedad actual de tener una mascota en casa ha ido aumentando de manera sorprendente, en la mayoría de los casos las prefieren antes que a tener un hijo, todo esto ha llevado a las empresas a reconvertir un poco su manera de vender productos y servicios en este campo, lo cual ha permitido la evolución del servicios funerarios, de bienestar (SPA), guarderías, cuidadores entre otros, ampliándolos no solo para los humanos sino también para éstas; dentro de esa evolución, los dispositivos de tracking lo han hecho también, ya las empresas no solo se limitan a rastrear sus bienes sino que ahora ofrecen servicios de ubicación en tiempo real e identificación de las mismas, incurriendo en costos por dispositivos electrónicos y limitando su utilización a personas que tengan buenas fuentes de ingreso, pero, ¿qué pasa con los que no?; para ellos se desarrolló esta aplicación web, pensada en ayudar a encontrar las mascotas extraviadas, la cuál con la ayuda de un smartphone con lector de códigos QR y la lectura de la respectiva placa de la mascota, permite identificar los datos básico de ésta y poder contactar al dueño para brindar información sobre su paradero para su recuperación, además de tips de cuidado, alimentación, entre otros, brindando con esto bienestar tanto para la mascota como para su dueño.

Palabras clave: mascotas, código QR, *APLICACIÓN WEB - PETAPP*

Abstract

The trend in today's society to have a pet at home has been increasing surprisingly, in most cases they prefer them over having a child, all this has led companies to reconvert their way of selling a bit products and services in this field, which has allowed the evolution of funeral services, wellness services (SPA), nurseries, caregivers among others, expanding them not only for humans but also for them; Within this evolution, tracking devices have done it as well, and companies are not only limited to tracking their goods but now offer real-time location services and identification of them, incurring costs for electronic devices and limiting their use of people with good sources of income, but what about those who are not ?; This WEB application was developed for them, designed to help find lost pets, which with the help of a smartphone with a QR code reader and reading the respective pet's license plate, allows identifying the pet's basic data and being able to contact to the owner to provide information about their whereabouts for their recovery, as well as tips for care, feeding, among others, thus providing well-being for both the pet and its owner.

Keywords: pets, QR code, WEB application PETAPP

Tabla de contenido

Lista de tablas

Lista de figura

Introducción

En el mundo actual, la tecnología juega un papel determinante en la evolución del hombre, esos cambios constantes y el acelerado crecimiento de los desarrollos electrónicos ha permitido avances significativos en muchos aspectos de la cotidianidad, entre ellos la manera en que se le puede brindar mejor calidad de vida a esos seres no humanos que acompañan todos esos momentos y la forma como ante determinadas situaciones se puede proceder para evitar no solo una posible pérdida del animal, si no la forma como se pueda recuperar, evitando con esto altos costos en la solución, la probabilidad de un accidente no solo para la mascotas sino para un posible conductor, además de los problemas psicológicos que pueda causar en su dueño por alguno de los casos anteriores.

Según Paulina Pulgarín Serna, Ingeniera Ambiental y activista de AnimalNaturalis, en Medellín en el año 2018 se extraviaron más de 8 animales por día, y en la mayoría de las situaciones no era posible encontrar a la mascota.

"La pérdida del animal de compañía es un momento muy doloroso para la familia, y sobre todo para el perro o gato, pues muchas veces nunca se reencuentran, es común que los animales hogareños no puedan sobrevivir en las calles y sean atropellados fácilmente o maltratados." (Serna, 2018)

Dado lo anotado anteriormente, sumado al constante crecimiento de internet y la facilidad que existe al día de hoy respecto al diseño e implementación de servicios alojados en la nube brinda posibilidades para realizar proyectos de toda clase, es por ello que basándose en todo lo que brinda la tecnología y más en el campo del desarrollo web, se procedió a realizar una plataforma web que permita registrar a usuarios dueños de mascotas, los datos de los animales, generar un código QR para imprimir, brindando una solución para localizarlas, en caso de extravío.

Este documento recoge el proceso de implementación del proyecto, el cual está estructurado en los siguientes capítulos:

El primero contiene la descripción del problema, antecedentes, justificación, objetivos y alcance del proyecto.

En el segundo se consignan los referentes teóricos que dan sustento al proyecto.

El tercero abarca el desarrollo del proyecto, en su parte metodológica.

En el cuarto se describen los resultados obtenidos.

Finalmente se registran las conclusiones y recomendaciones del trabajo de grado.

Capítulo 1. Descripción del proyecto

Planteamiento el problema

Con los cambios de la sociedad actual ha habido una creciente necesidad humana de tener cualquier tipo de mascotas en sus hogares, al punto que las han empezado a humanizar o al extremo de optar por no tener hijos a cambio de una de ellas; en medio del afán de hacerla cada vez más un integrante de la familia, se han dado infinidad de casos en donde su amo o dueño está dispuesto a pagar sumas considerables de dinero en afiliaciones a medicina prepagada, spa para mascotas a tal punto de llegar a servicios funerarios. Hasta aquí todo normal, pero cuando alguna de ellas se pierde por cualquier circunstancia, ¿qué hacer para recuperarla?, aunque no es solo ese el problema que aqueja la situación ya que anímicamente sus dueños empiezan a decaer, trayendo dificultades de salud, también con esta pérdida se presentan problemas de salubridad no solo para la mascota sino también para los individuos de su entorno y si a todo lo anterior se le suma el alto grado de accidentalidad que pueden generar al estar por ahí sueltas y sin alguien que las pueda cuidar y los costos que pueda representar la adquisición de un dispositivo para ubicación en tiempo real.

Antecedentes

La tecnología ha hecho que la vida de las personas sea más fácil y práctica, solucionando en muchas oportunidades los problemas que se les presenta: compras y pagos en línea, agendas y reuniones virtuales, son algunas de las opciones que, a través de 'gadgets', dispositivos y aplicaciones, los avances tecnológicos han dejado para ayudar al hombre con sus tareas diarias. Entre todas esas renovaciones que la tecnología ha hecho, se destaca en los últimos tiempos la opción de apoyar a los dueños de mascotas con el cuidado de las mismas, permitiendo que ahora sea mucho más fácil el tener animales domésticos en casa y poder disfrutar de la compañía de perros y gatos al interior del hogar (el Tiempo, 2019).

En el mundo se han desarrollado proyectos con objetivos similares a éste, a continuación, se resumen algunos de ellos:

Humane Tribe Pet Locator Tag, es un dispositivo que cuenta con tecnología NFC, con un chip externo y además cuenta con código QR de URL corta para facilitar su ubicación, si bien cuenta con mucha tecnología, el costo no es demasiado alto, tan solo 20 dólares aproximadamente. El servicio se presta en los Estados Unidos. (Humane tribe, 2020), (kaveri, 2020).

Garmin no se podía quedar atrás y con Garmin T5 GPS Dog Collar, permite a los dueños saber la ubicación en un rango de 15 kms, con materiales resistentes al agua y a los impactos y un costo aproximado de 250 dólares. (Garmin, 2020).

Páginas como https://geekflare.com/pet-trackers/, permiten tener una mirada amplia sobre el estado actual de la tecnología en el área de las mascotas.

Justificación

La implementación de este proyecto permite que las personas dueñas de mascotas puedan de una manera económica, facilitar su ubicación en caso de pérdida y ofrecer tips para el cuidado y bienestar de las mismas. De otro lado, se evita recurrir a procesos invasivos, de implantación de chips en los animales, permitiendo de manera rápida, con la lectura del QR y una conexión a internet, la pronta ubicación de su dueño, incrementando la probabilidad de ser recuperada y posiblemente evitar un accidente y muerte en las calles, además del bajo costo para la implementación, pues se basaría en el registro, generación del QR y la compra de un collar en acero con el código grabado.

Objetivos.

Objetivo general:

Desarrollar una aplicación Web que permita identificar una mascota en caso de pérdida, empleando un código QR, con información básica, para que se pueda ubicar mas fácilmente a su dueño y adicionalmente se brinden consejos de cuidado y bienestar animal.

Objetivos específicos:

- Diseñar la aplicación Web, de manera que permita cumplir con la funcionalidad establecida.
- Implementar la aplicación en el framework seleccionado.
- Realizar pruebas de usuario para verificar la funcionalidad de la aplicación.

Alcance

Con el desarrollo de este proyecto, el usuario final contará con una aplicación Web, de fácil uso, en donde podrá realizar su respectiva creación de usuario para el ingreso al mismo, la creación y modificación de los datos de su mascota para luego poder generar su código QR, el cual podrá ser consultado desde cualquier smartphone para obtener los datos básicos para ser contactado en el caso que sea requerido, además de poder encontrar tips para el cuidado y prevención de enfermedades para la misma.

Capítulo 2. Marco Teórico

2.1 Aplicación Web

Hoy en día es posible utilizar herramientas de tipo virtual accediendo a un equipo ubicado en una organización de manera local o a una distancia considerable gracias a internet, una tecnología que cambio al mundo para bien, y es que nadie pensaba que una tecnología pensada con fines militares en la década de los 60 permitiría interactuar con cada ciudad del planeta de manera casi instantánea, en donde gracias a los aportes iniciales de tres norteamericanos (Leonard Kleinrock, J.C.R. Licklider, Robert W. Taylor,) surgiría lo que más tarde se conocería como Arpanet (Advanced Research Projects Agency Network), entonces esos fueron los primeros pasos de lo que hoy se conoce como la nube, aunque esto no sería posible aún sin los aportes de Robert Kahn y Vinton Cerf, quienes desarrollaron el protocolo TCP/IP, el cual permitió que las maquinas se comunicaran a nivel global. Sin embargo, es hasta el desarrollo de html y la creación de www que se conocería a internet como tal de la mano de Tim Berners-Lee y Robert Cailliau, el propósito del sr Berners era mostrar un ejemplo de cómo vincular información entre sí.

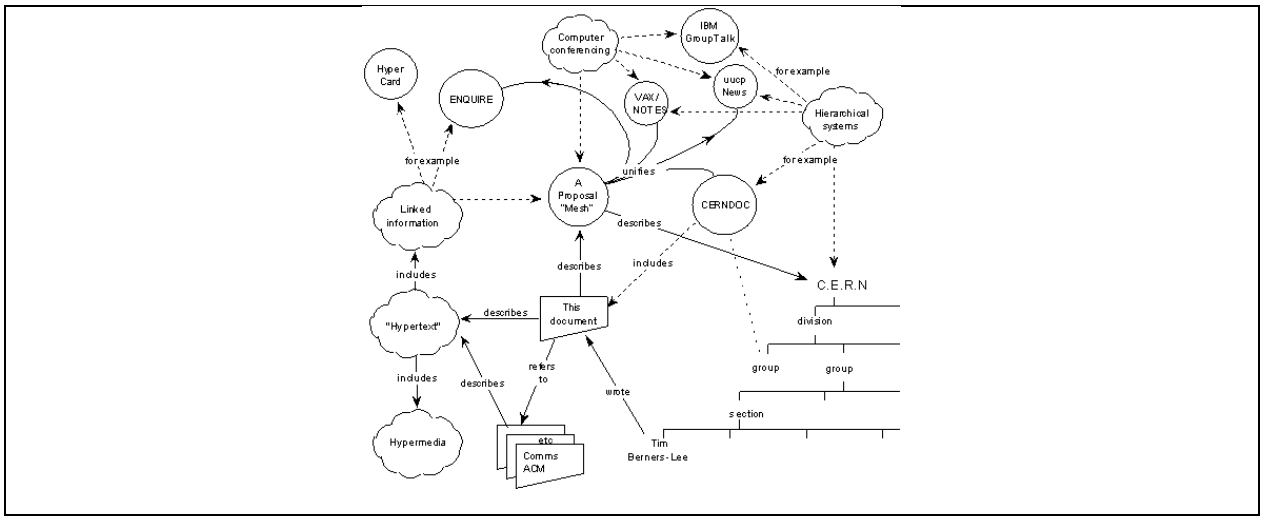

Figura 1

(Tim Berners-Lee, CERN March 1989, May 1990)

This proposal concerns the management of general information about accelerators and experiments at CERN. It discusses the problems of loss of information about complex evolving systems and derives a solution based on a distributed hypertext system.) (Berners-Lee, 1998)

En la figura 1 se observa la propuesta desarrollada por Tim Berners .

Pero es con la llegada de PHP (**Personal Home Page Tools**) que todo mejoraría (PHP tal y como se conoce hoy en día es en realidad el sucesor de un producto llamado PHP/FI. Creado en 1994 por Rasmus Lerdorf, la primera encarnación de PHP era un conjunto simple de ficheros binarios Common Gateway Interface (CGI) escritos en el lenguaje de programación C.) (https://www.php.net/, 2201-2020)

Las aplicaciones web reciben este nombre porque se ejecutan en internet. Es decir que los datos o los archivos en los que trabajas son procesados y almacenados dentro de la web. Estas aplicaciones, por lo general, no necesitan ser instaladas en tu computador (Juan Avella, s.f.).

Características y requerimientos de una aplicación web:

- Utilizan un sitio web con un front-end.
- Una aplicación web puede ser una aplicación de Internet o una intranet
- Solicitud y validación de http: las aplicaciones de EB se comunican con el usuario a través del protocolo http.
- El manejo de solicitudes http lo realiza un servidor web.
- Existe una sesión única y una relación entre un usuario web y una aplicación web.
- Comunicarse con otros recursos web como bases de datos (IBM, udb, db2, ORACLE, Microsoft SQL y MySQL).
- Admiten la entrada, transacción y presentación de datos interactivos a través de un navegador web estándar
- Existe una aplicación web como una jerarquía estructurada de directorios.

(Wei-Dong Jackie Zhu, 20 agosto 2004)

Tipos de aplicaciones web:

- Aplicación web estática: son sencillas y no requieren muchos cambios. Están desarrolladas en HTML y CSS además permite agregar algo de publicidad mediante banners y contenido multimedia, su edición es algo engorrosa debido a que el código debe descargarse para hacer las modificaciones y volverlo a subir para ser publicado.
- Aplicación web dinámica: tienen un mayor grado de dificultad a la hora de desarrollarla, su contenido se actualiza con cada visita realizada por el usuario, es en este punto donde se recurre a php y javascript para lograr la interacción de la página con el cibernauta. también se puede modificar el diseño de la web.
- Tienda virtual: se le conoce como e-commerce son desarrolladas para vender artículos o/y servicios, requieren más características aquí ya se utiliza el protocolo https ya que se debe gestionar información sensible de cada usuario, además se debe implementar métodos de pago, asegurando total confidencialidad y seguridad durante la transacción ejecutada por el usuario.

2.2 Lenguaje de desarrollo PHP

Es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. (Achour, 2020).
PHP o Pre-Procesador de Hipertexto, usado para diseñar aplicaciones web y hacerlos mas intuitivos e interesantes. Se ha vuelto popular por años como un lenguaje de secuencia de comandos del lado servidor, es facil de usar, poderoso. Trabaja sobre multiples sistemas operativos y puede soportar multiples servidores. (David Carr, 2018)pag 19.

2.3 Framework Laravel

En la actualidad existen varios frameworks que trabajan con php con el fin de simplificar los procesos para llegar al resultado deseado.
Laravel es un framework de código abierto que brinda facilidad para trabajar con PHP creada en el año 2011. Un framework es un entorno o marco de trabajo. Es un conjunto de conceptos, de prácticas y criterios estandarizados a seguir (Palomares, s.f.)

Laravel es un marco de aplicación web con una sintaxis elegante y expresiva. Permite a los desarrolladores crear sin preocuparse por las pequeñas cosas. Intenta aprovechar lo mejor de otros frameworks y aprovechar las características de las últimas versiones de PHP (Otwell, s.f.)
Laravel ofrece una herramienta importante para poder interactuar, conocida como Artisan el cual se basa en el componente Console de SYMFONY, en donde por medio de un conjunto de comandos permite realizar diferentes tareas incluso si la aplicación está en línea o en producción. (Laravel, 2020).
Según Wikipedia un framework es una estructura conceptual y tecnológica de soporte definido, normalmente con módulos de software concretos, que puede servir de base para el desarrollo de software, gracias a esto se puede evitar escribir código repetitivo, ya que el framework se encargará degestionar el código común, como el necesario para enlazar la base de datos, os datos de validación para los permisos, permitiendo que el usuario se enfoque en el desarrollo de la aplicación, además los frameworks se estandarizan en el modelo MVC, Modelo-Vista-Controlador,

- Modelo: relaciona a los datos que componen la aplicación y sus normas
- Vista: es la manera que los datos serán presentados a los usuarios
- Controlador: es el componente de la aplicación que gestiona las peticiones de los clientes y administra todo lo relacionado a la ejecución del sistema.

El modelo MVC permite que todo se gestione de manera ordenada, y no se puede evitar mencionar que un framework permite hacer forma sencilla una aplicación, ya que sin su uso sería muy complejo hacerla, sin embargo, es importante aclarar que se requieren habilidades en el área de desarrollo para comprender el funcionamiento del framework además de las cualidades que ofrece.
¿Pero que framework se debe elegir? La respuesta es sencilla, se debe elegir uno que tenga buen soporte por parte de la comunidad relacionada con el mismo, además en los foros se encontraran usuarios con buena experiencia que brindaran tips para sacarle provecho al framework, por otro lado, es esencial que este ofrezca seguridad ya que sin este factor el proyecto quedaría expuesto a cualquier ataque una vez publicado.
¿Es laravel una buena opción? Si lo ya que es existen varias razones que demuestran su versatilidad y manejo para el desarrollo de una aplicación
Modularidad: Laravel se construyó sobre más de 20 bibliotecas diferentes y es en sí mismo dividido en módulos individuales. Estrechamente integrado con Composer, se puede actualizar con facilidad.
Capacidad de prueba: desarrollado desde cero para facilitar las pruebas, Laravel gestiona varias Ayudas que le permiten visitar las rutas de sus pruebas, rastreando así el código resultante, Garantizar que los métodos son llamados en ciertas clases, sean seguros y así denegar el servicio a usuarios falsos tratando de autenticarse.

Enrutamiento: Laravel te da mucha flexibilidad a la hora de definir las rutas de tu solicitud. Por ejemplo, puede vincular manualmente una función anónima simple a una ruta con un verbo HTTP, como GET, POST, PUT o DELETE. Esta característica es inspirado en micro-frameworks, como Sinatra (Ruby) y Silex (PHP).
Además, es posible adjuntar funciones de filtro que se ejecutan en rutas particulares

Gestión de la configuración: la mayoría de las veces, su aplicación se ejecutará en diferentes entornos, lo que significa que la base de datos o el servidor de correo electrónico.

La configuración de las credenciales o la visualización de mensajes de error será diferente cuando la aplicación se ejecuta en un servidor de desarrollo local que cuando se ejecuta en un servidor de producción. Laravel permite definir configuraciones para cada entorno y luego selecciona automáticamente la configuración correcta dependiendo de dónde se esté ejecutando la aplicación.

2.4 PHP MyAdmin

Es una herramienta de software gratuita escrita en PHP, destinada a manejar la administración de MySQL a través de la Web. También permite administrar la base de datos Maria DB. (Achour, 2020).

2.5 Código QR.

Un código QR es un código de barras bidimensional cuadrada que puede almacenar los datos codificados. La mayoría del tiempo los datos es un enlace a un sitio web. (UNITAG, s.f.). ¿Como se genera un código QR en laravel?, para ello se requiere añadir un paquete mediante composerjson

```
"require": {
        "php": "^7.2.5",

        "simplesoftwareio/simple-qrcode": "~3",
```

Luego se procede a instalar el paquete mediante el comando “composer update”, una vez realizado este procedimiento se podrá utilizar el paquete para la generación del Código Qr.

2.6 Servidor Web

Es un programa que recibe solicitudes en una red, ejecuta alguna logica basada en los parametros de la solicitud y retorna el resultado a la aplicación del cliente, la respuesta generada por el servidor web es visualizada en el buscador. (Wei-Dong Jackie Zhu, 20 agosto 2004), Pag 5.

2.7 SQL para MySQL

Fue desarrollado inicialmente por IBM a principios de los 70´s, fue formalizado por la ANSI en 1986. La norma ha sido revisada siete veces. Los ejemplos de este libro fueron probados con MySQL 5.6, que cumple con el estándar SQL: 2008. Este estándar es una revisión anterior a SQL: 2011, el estándar más reciente. SQL consta de un lenguaje de definición de datos (DDL) y un lenguaje de manipulación (DML). El DDL se utiliza para crear, eliminar y modificar la estructura de una tabla y otros objetos de la base de datos. El DML se utiliza para insertar, recuperar y actualizar datos en una tabla o tablas. (Darmawikarta, 2014) pag 7-8

Capítulo 3. Desarrollo de la propuesta

Metodología

Para este proyecto, se siguió una metodología ágil de desarrollo, fundamentada en Scrum. Se dividió en 3 Sprints, tal como se detalla en la tabla del Cronograma.

Cronograma

Atendiendo a los objetivos propuestos, se dividió el proyecto en las actividades que registran a continuación:

Sprint	Actividades	Fecha
1. Diseño de la aplicación	Selección del framework, diseño de la arquitectura	10 /03/2020 – 24 /03/2020
	Modelado de la base de datos	
	Diseño de la IU	
2. Implementación	Funcionalidad registro usuarios	7/04/2020 - 12/05/2020
	recopilación información sobre situaciones relacionadas con animales extraviados	
	configuración servidor para iniciar la fase de Desarrollo basada en el framework seleccionado	
	adecuación area de trabajo, (servicios de red, internet, servidor, cliente,)	
	Desarrollo Código Fuente para crear las vistas, seleccion de api, y scripts que facilitaran el proceso.	
3. Pruebas de usuario	Pruebas de usuarios1:	3/11/2020

	Modificación según las pruebas:	27/11/2020
	Pruebas de usuarios2:	
	Modificación según las pruebas:	
	Pruebas de usuarios3:	
	Modificación según las pruebas:	
	Pruebas de usuarios4:	
	Modificación según las pruebas	
	Pruebas de usuarios5	
	Modificación según las pruebas	
	Generación de códigos QR	
	Verificación de funcionalidad de los códigos	

Tabla 1 Cronograma

Presupuesto

Recurso Humano				
Encargado	Rol	Valor hora	# Horas proyecto	Valor total

Danny Ruben Gonzalez	Desarrollador	20000	60	1200000
Oscar Alonso Gonzalez	Técnico Mantenimiento y redes	20000	40	800000

Recursos físicos			
Tipo	# Meses	Valor mensual	Valor total
Internet	4	60000	240000
Servicios públicos	4	30000	120000
Total	360000		

Recursos software y hardware			
Tipo	Cantidad	Valor unitario	Valor total
Equipo de cómputo	2	3000000	6000000
Total	6000000		

Tabla 2 Presupuesto

Valor total del proyecto: $8’ 360.000.

Capítulo 4. Resultados

Para este proyecto se uso una arquitectura MVC (Modelo, Vista Controlador), en el Modelo (Mysql) en el controlador (PHP - Laravel) y en la Vista (HTML y CSS)
En la figura se puede observar la vista para diseñador de la base de datos basada en Mysql

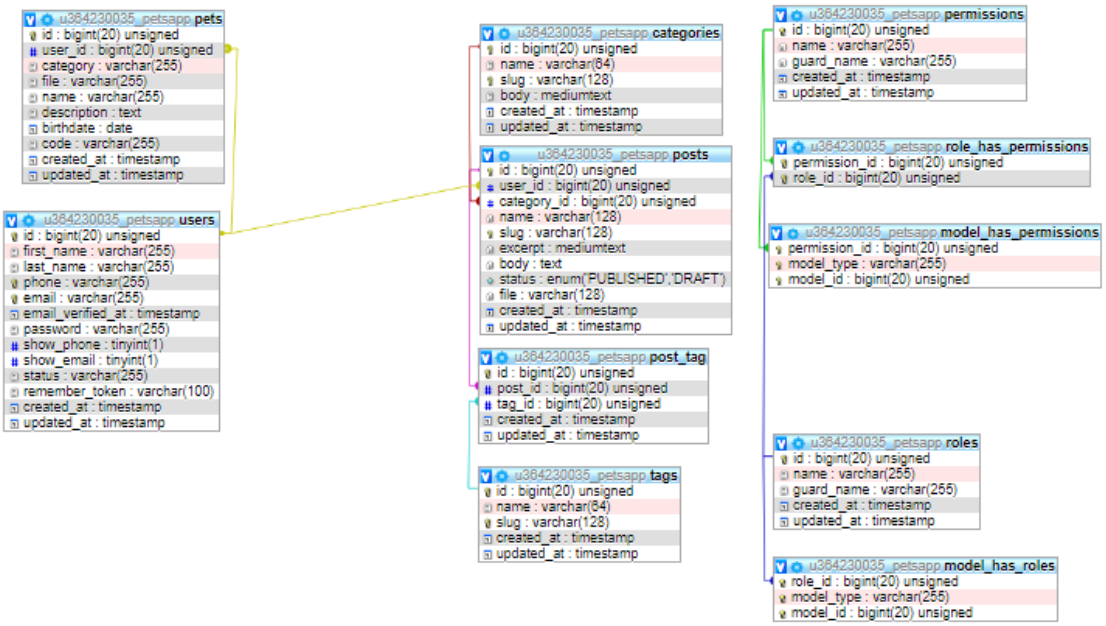

Figura 2 Arquitectura

En el modelo de la base de datos encontraremos 11 tablas
USERS, PETS, POSTS, CATEGORIES, POST_TAG, TAGS
ROLES, MODEL_HAS_ROLES, MODEL_HAS_PERMISSIONS, PERMISSIONS

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

iones

	id	first_name	last_name	phone	email	email_verified_at	password	show_phone
Editar Copiar Borrar	1	Admin		012345678	admin@petapp.com	NULL	$2y$10$6eZYV7DqjclWRtn/elnmkuhBg198qpYB1Jnqgn0PkFE...	1
Editar Copiar Borrar	2	Andrea	Barrera	123456789	andreabarrera@mail.com	NULL	$2y$10$jL3Vt2mNQEDjxnDqdDWr/.mp4b/9Wyvx57RDSVMEJzZ...	1
Editar Copiar Borrar	3	José	Hernández	00234567	josehernandez@mail.com	NULL	$2y$10$4EmWn1STufEzi4braBUW2exUj4.EbmMKt.ZjlKodhR7...	1
Editar Copiar Borrar	4	Oscar Alonso	González Muñoz	3502406412	oagonzalez@misena.edu.co	NULL	$2y$10$d0POKtGhBj1cR838uA0l2eZ76CM54.8krMHEZa3Q8L1...	1
Editar Copiar Borrar	6	Danny	Gonzalez	3173812355	dondannys@gmail.com	NULL	$2y$10$qvto39ga58JgAr/5RNBIlODMGvTkXsl5mCR8TrTRpVr...	1
Editar Copiar Borrar	8	Lorena	Gil	3167713414	Lorenitagf23@hotmail.com	NULL	$2y$10$rC2ECAmHZao3EEZt6QWj2eB880wDsZ0eBTrku1YrAD5...	0
Editar Copiar Borrar	10	Joseph	Bakhos	04242748346	josephbakhos@gmail.com	NULL	$2y$10$dWlHrnVb6ZRDt4MD0XCqPekeMXOY95nfBAmgBKq5VOM...	1
Editar Copiar Borrar	11	Kely	Guerra	3509878756	kyguerra@misena.edu.co	NULL	$2y$10$yrOlKYhOEwdWQNYKtONxZef1lH721Xhl/J5plv.3GRj...	1
Editar Copiar Borrar	14	Pepito	Perez	3504567898	pepitoperez@petaps.com.co	NULL	$2y$10$E1zp4bQq0sCznid6YAW0PeUtkC/Y0vlfbEB7wL9NVse...	0
Editar Copiar Borrar	15	Fulanito	de Tal	3129876541	Fulanito@petaps.com.co	NULL	$2y$10$XmsXkv/SGLw87j0N/jNybOC3tEwhreWOxxROSmADU7m...	1
Editar Copiar Borrar	16	Sutano	Vélez	(4) 2383201	sutanito@petaps.com.co	NULL	$2y$10$AdmdCQzt6HslhVLNFCHS2uhVODi7fX00fX/BLWUILtR...	1
Editar Copiar Borrar	17	Andrés	Londoño	3225327305	andres14l@hotmail.com	NULL	$2y$10$Hw/y5qnzka2Spt3atELGTuCjOS3s/8NsRJhz3qUAVPm...	0

Figura 3

Tabla Users

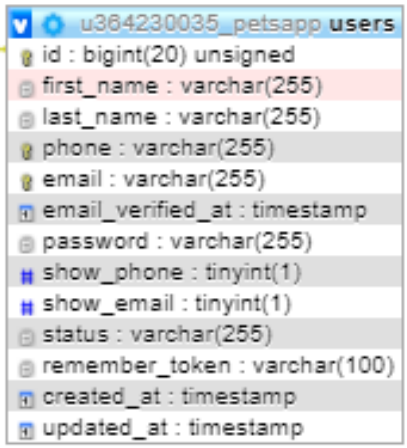

Figura 4

La tabla Users permite almacenar la información básica de cada usuario registrado incluido contraseña para realizar el login en la página, teléfono y correo electrónico, cuenta con una clave primaria para el id el cual es auto incremental.

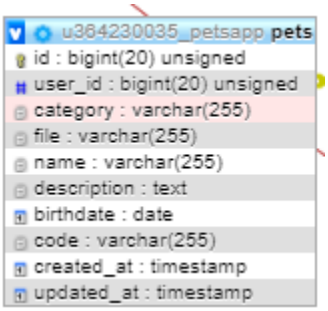

Figura 5

La tabla pets permite almacenar la información básica de la mascota registrada por el usuario, en esta tabla se encuentra como clave primaria el id asignado para la mascota, y cuenta conuna clave foránea que conecta con la tabla users, la casilla categoría permite seleccionar la especie, la casilla file permite adjuntar la ruta donde se aloja la imagen que corresponde a la mascota, también se encuentran las casillas name y description, en donde se puede describir informacion sensible sobre la mascota, como rasgos, estado de salud y su nombre de pila, es importante destacar que estas tablas pueden ser editadas por el usuario en caso de querer adjuntar una imagen más actualizada o una mejor descripción de los rasgos correspondientes a la mascota.

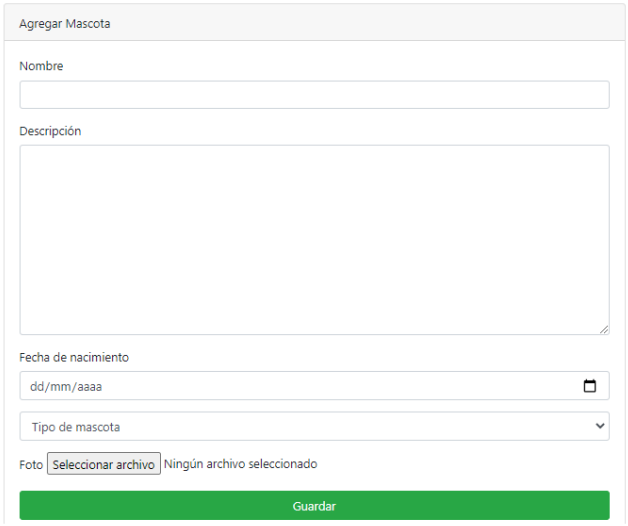

Figura 6

En la figura 7 se puede observar la interfaz gráfica que relaciona la tabla pets para ingresar la información que será alojada en la base de datos.

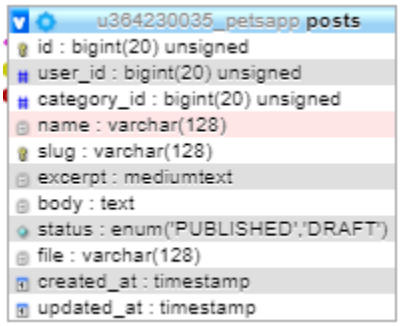

Figura 7

La tabla posts cuenta con los siguientes campos: el campo id permitirá que la consulta almacenada con el numero generado pueda ser visualizada al realizar la petición por parte del usuario, esta es la clave primaria que permitirá interactuar con otras tablas, también se encuentra el campo user_id, el cual permitirá interactuar con la tabla user, ya que este es quien realiza las consultas. El siguiente campo es category e interactúa con la tabla category donde se almacena las categorías creadas por los usuarios para luego ser consultada, también están los campos relacionados con la creación de un post, en donde se ingresarán datos de interés, adjuntar una imagen y un campo para permitir dar de alta el post en la interfaz web.

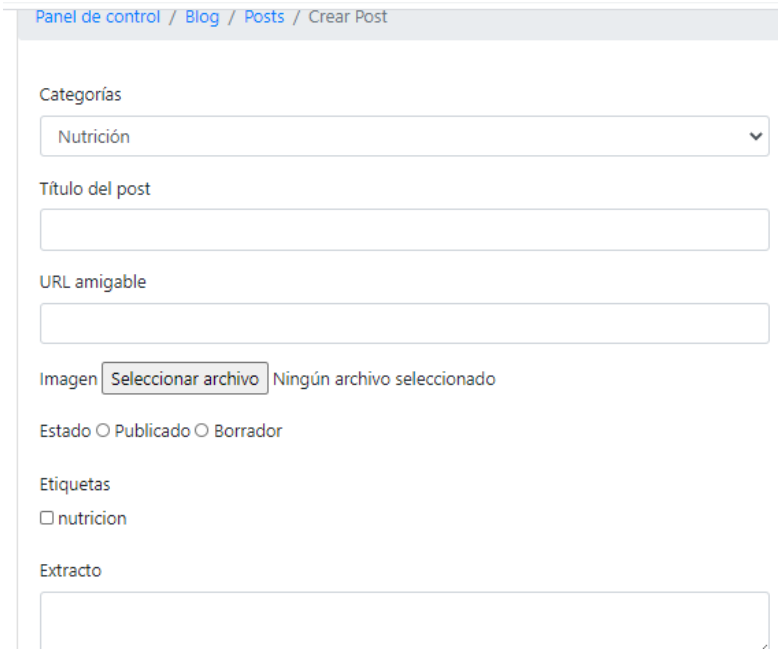

Figura 8

En la figura se puede visualizar la interfaz web para la creación de un post.

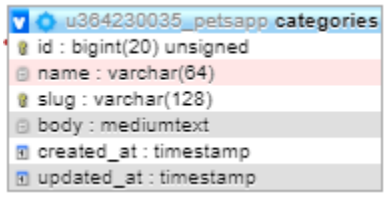

Figura 9

La tabla categories permitirá crear una categoría para el post antes creado, esta tabla cuenta con el campo id que funciona como clave primaria, además será la clave foránea para la tabla posts, a esta se le podrá asignar un nombre, también cuenta con el campo body para agregar una breve descripción, el campo slug que permitirá agregar el enlace dentro de la página,

Figura 10

La tabla post_tag, permite interactuar tanto conla tabla post mediante las clave foránea post_id y tag_id, en donde esta tabla permitirá acceder a los datos alojados en las otras tablas antes mencionadas mediante consultas.

Figura 11

La tabla tags permite ingresar las etiquetas para los post que se realicen en la página, y al igual que sus antecesoras cuenta con un campo id el cual es incremental, un campo name, y un campo slug.

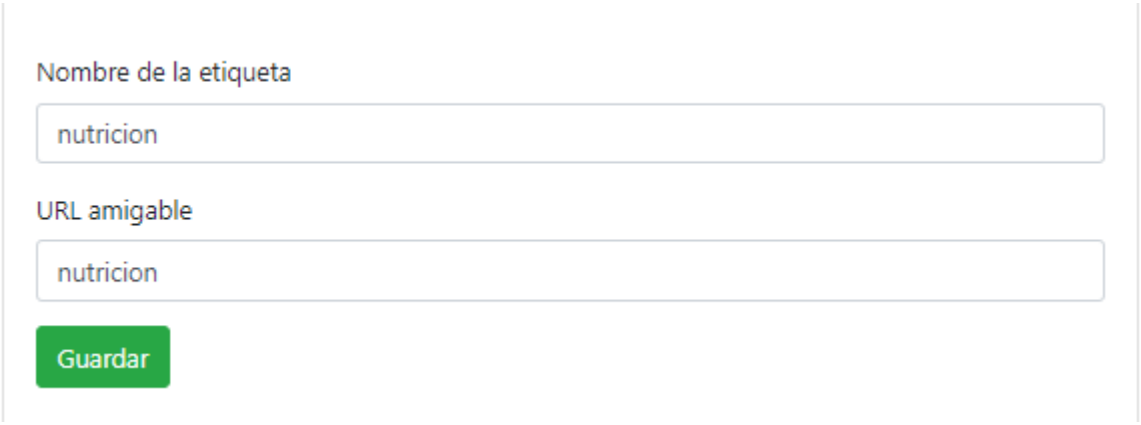

Figura 12

En la figura 13 se puede visualizar la interfaz para ingresar informacion en los campos correspondientes a la tabla tags

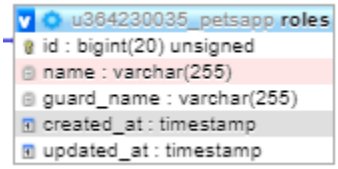

Figura 13

La tabla roles permite darle un rol usuario o admin a las personas registradas, esta tabla cuenta con el campo id, que funciona como clave primaria, el campo name, que asignara usuario o admin según los requerimientos.

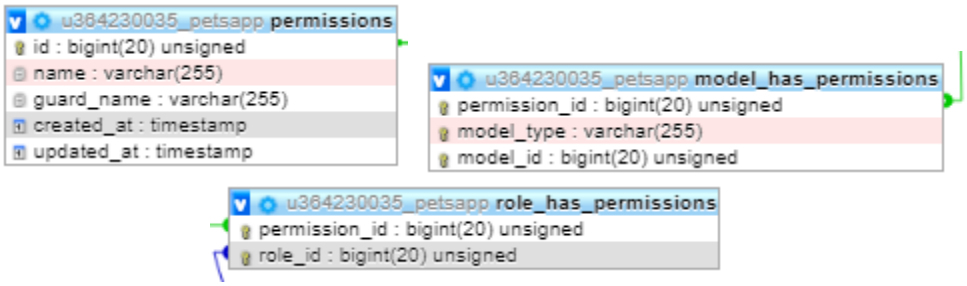

Figura 14

Las tablas permissions, model_hsa_permissions y role_has_permissions, interactúan entre si con el fin de permitir o denegar según el administrador el acceso a las secciones de la aplicación, en este caso el usuario cuenta con permisos de edición o eliminación de los datos relacionados con su mascota

El código fuente utilizado y la división de los módulos se presenta asi:
En la figura 15 se observa la estructura de Laravel la cual se compone de las siguientes carpetas

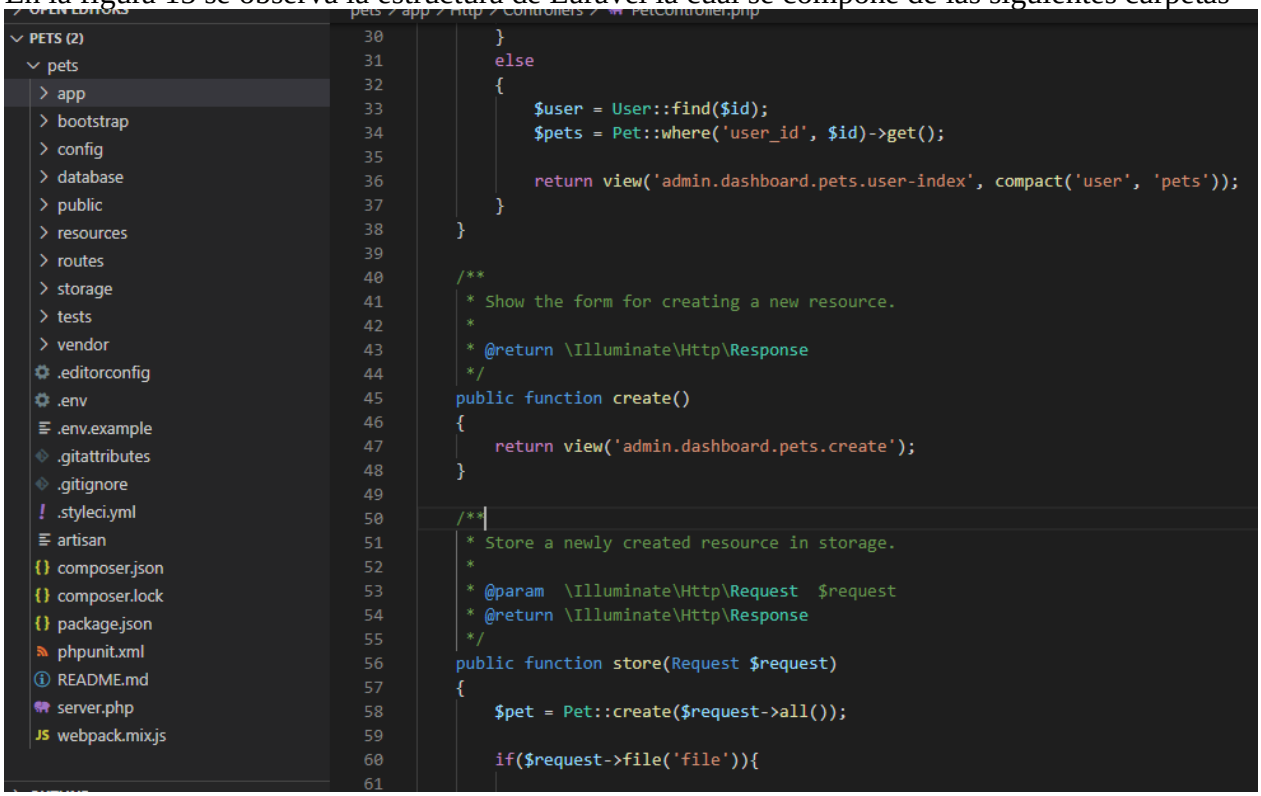

Figura 15 Estructura

Carpeta App: en esta carpeta se aloja la estructura principal del código para el proyecto aquí se encontrarán los directorios, console, http los cuales tendrán funciones y procedimientos para interactuar con otras carpetas correspondientes a la aplicación

Carpeta Bootstrap: en esta carpeta se alojan componentes que conforman el framework css con el fin de brindar una interfaz amigable e intuitiva al usuario final,

Carpeta Config: en esta carpeta se configuran los arreglos PHP que trabajaran para hacer funcional la aplicación.

Carpeta database: en esta sesión se encuentra una subcarpeta que se llama migraciones, desde ahí se arma la base de datos y será reenviada a un motor de bases de datos, en esta sesión es donde se configura todo lo referente a las tablas de la base de datos para la aplicación.

Carpeta public: el contenido de esta carpeta es la que el usuario final podrá visualizar en la interfaz web, imágenes, fuentes, estilos css e interacciones con la aplicación.

Carpeta Resources: en esta sesión se guardan las vistas y los raw assets, en otras palabras, esta carpeta es esencial para visualizar la interfaz web de manera amigable con el usuario.

Carpeta routes: esta carpeta es esencial en los procesos de laravel ya que se encarga de recibir solicitudes y enviar respuestas relacionada con las peticiones del usuario, entre sus subcarpetas encontraremos routes/web.php, que se encargara de entrar a cada sesión de la aplicación web.

También se cuenta con el directorio composer.json que es donde se pueden declarar las librerías que serán utilizadas para el proyecto en cuestión.

En la figura 16 se puede demostrar el código fuente para home controller, el cual permitirá ingresar al usuario registrado, en donde según el id le asignará un rol.

```php
<?php

namespace App\Http\Controllers;

use App\User;
use Illuminate\Http\Request;

class HomeController extends Controller
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('auth');
    }

    /**
     * Show the application dashboard.
     *
     * @return \Illuminate\Contracts\Support\Renderable
     */
    public function index()
    {
        $id = auth()->id();
        $user = User::find($id);

        if ($user->id != 1)
        {
            $user->assignRole('User');
        }
```

```
        return view('admin.dashboard.index');
    }
}
```

Figura 16 Home Controler

En este apartado se puede observar el código fuente para permitir el registro de un nuevo usuario

```
class RegisterController extends Controller
{
    /*
    |--------------------------------------------------------------------------
    | Register Controller
    |--------------------------------------------------------------------------
    |
    | This controller handles the registration of new users as well as their
    | Validation and creation. By default this controller uses a trait to
    | provide this functionality without requiring any additional code.
    |
    */

    use RegistersUsers;

    /**
     * Where to redirect users after registration.
     *
     * @var string
     */
    protected $redirectTo = RouteServiceProvider::HOME;

    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('guest');
    }

    /**
     * Get a validator for an incoming registration request.
     *
     * @param  array  $data
     * @return \Illuminate\Contracts\Validation\Validator
     */
    protected function validator(array $data)
```

```
    {
        return Validator::make($data, [
            'first_name' => ['required', 'string', 'max:255'],
            'last_name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users']
,
            'password' => ['required', 'string', 'min:8', 'confirmed'],
            'phone' => ['required', 'string', 'max:12'],
            'show_phone' => ['boolean'],
            'show_email' => ['boolean'],
        ]);
    }

    /**
     * Create a new user instance after a valid registration.
     *
     * @param  array  $data
     * @return \App\User
     */
    protected function create(array $data)
    {
        if (! Isset( $data['show_phone'] ))
            {
               $data['show_phone'] = "0";
            }

        if (! Isset( $data['show_email'] ))
            {
               $data['show_email'] = "0";
            }
        return User::create([
            'first_name' => $data['first_name'],
            'last_name' => $data['last_name'],
            'email' => $data['email'],
            'password' => Hash::make($data['password']),
            'phone' => $data['phone'],
            'show_phone' => $data['show_phone'],
            'show_email' => $data['show_email'],
        ]);
    }
}
```

Figura 17 Reg User

En la figura 18 se detalla el código fuente correspondiente al controlador de usuario, el cual permitirá visualizar y editar los datos mostrados por la interfaz web.

```php
<?php

namespace App\Http\Controllers;

use App\Pet;
use App\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Storage;

class UserController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $id = auth()->id();

        if ($id == 1)
        {
            $users = User::where('id', '!=', 1)->get();
            return view('admin.dashboard.users.admin-index', compact('users'));
        }
        else
        {
            $user = User::find($id);
            return view('admin.dashboard.users.user-index', compact('user'));
        }
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        //
    }
```

```php
    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        //
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        $user = User::find($id);
        $pets = Pet::where('user_id', $id)->get();

        return view('admin.dashboard.users.profile', compact('user', 'pets'));
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        $user = User::find($id);
        return view('admin.dashboard.users.edit-user', compact('user'));
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
```

```
    public function update(Request $request, $id)
    {
        $this->validate($request, [
            'password' => ['required', 'confirmed']
        ]);

        $user = User::find($id);

        if ($request['password'] != NULL)
        {
            $request['password'] = Hash::make($request->password);
            $user->password = $request['password'];
            $user->save();
        }

        $user->fill($request->except('password'))->save();


        return redirect()->route('users.index')-
>with('info', 'Usuario actualizado con éxito.');
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        $user = User::find($id)->delete();

        return redirect()->route('users.index');

    }

    public function status(Request $request)
    {
        $user = User::find($request->id);

        $user->fill($request->all())->save();


        return redirect()->route('users.index')-
>with('info', 'Estado actualizado con éxito');
```

```
    }


    public function password(Request $request)
    {
        return view('admin.dashboard.users.edit');
    }

    public function changePassword(Request $request)
    {
        $this->validate($request, [
            'password' => ['required', 'confirmed']
        ]);

        $id = auth()->id();

        $user = User::find($id);

        if ($id == 1 && $request['password'] != NULL)
        {
            $request['password'] = Hash::make($request->password);
            $user->password = $request['password'];
            $user->save();
        }

        return redirect()->route('users.index')-
>with('info', 'Contraseña actualizada correctamente.');
    }
}
```

Figura 18 User Controler

En la figura 19 se observa el código fuente relacionado con la mascota en donde se puede crear, editar o eliminar los datos relacionados con el animal.

```
<?php

namespace App\Http\Controllers;


use Image;
use QrCode;
use App\User;
use App\Pet;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Storage;
```

```php
class PetController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $id = auth()->id();

        if ($id == 1)
        {
            $pets = Pet::all();

            return view('admin.dashboard.pets.admin-index', compact('pets'));
        }
        else
        {
            $user = User::find($id);
            $pets = Pet::where('user_id', $id)->get();

            return view('admin.dashboard.pets.user-
index', compact('user', 'pets'));
        }
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('admin.dashboard.pets.create');
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
```

```
    {
        $pet = Pet::create($request->all());

        if($request->file('file')){

            $file = $request->file('file');
            $img = Image::make($request->file('file'))->resize(120, 120);
            $path = Storage::disk('public')->put('img/app/pets/',  $file);
            $pet->fill(['file' => $path])->save();
            $img->save($path);
        }

        return redirect()->route('pets.index')-
>with('info', 'Mascota creada con éxito');
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        $pet = Pet::where('id', $id)->first();
        $user = User::find($pet->user_id);

        return view('web.profile', compact('user', 'pet'));
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        $pet = Pet::find($id);
        return view('admin.dashboard.pets.edit', compact('pet'));
    }

    /**
     * Update the specified resource in storage.
     *
```

```
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
        $pet = Pet::find($id);

        $pet->fill($request->all())->save();

        if($request->file('file')){

            $file = $request->file('file');
            $img = Image::make($request->file('file'))->resize(120, 120);
            $path = Storage::disk('public')->put('img/app/pets/',  $file);
            $pet->fill(['file' => $path])->save();
            $img->save($path);
        }

        return redirect()->route('pets.index')-
>with('info', 'Mascota actualizada con éxito.');
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        $pet = Pet::find($id)->delete();

        return redirect()->route('pets.index')-
>with('info', 'Mascota eliminada con éxito.');
    }

    public function qr($id)
    {

        $pet = Pet::find($id);
        return view('admin.dashboard.pets.qr', compact('pet'));
    }
}
```

Figura 19 Pet User

Este apartado cuenta con varios subindices:
Dentro de este controlador (ver fig 20) se encuentran las clases que permitirán registrar, iniciar sesión, reiniciar contraseña de usuario

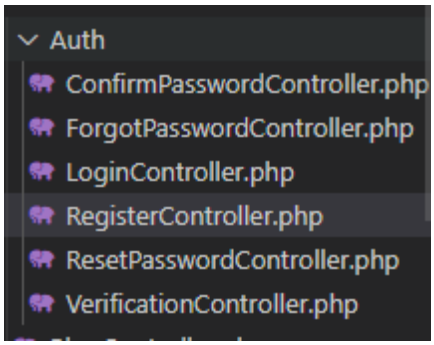

Figura 20 Class registros

En la siguiente figura se evidencia el Código para registro de usuario (fig 21)

```
* Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('guest');
    }

    /**
     * Get a validator for an incoming registration request.
     *
     * @param  array  $data
     * @return \Illuminate\Contracts\Validation\Validator
     */
    protected function validator(array $data)
    {
        return Validator::make($data, [
            'first_name' => ['required', 'string', 'max:255'],
            'last_name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users']
,
            'password' => ['required', 'string', 'min:8', 'confirmed'],
            'phone' => ['required', 'string', 'max:12'],
            'show_phone' => ['boolean'],
            'show_email' => ['boolean'],
        ]);
    }
```

```php
protected function create(array $data)
    {
        if (! isset( $data['show_phone'] ))
            {
               $data['show_phone'] = "0";
            }

        if (! isset( $data['show_email'] ))
            {
               $data['show_email'] = "0";
            }
        return User::create([
            'first_name' => $data['first_name'],
            'last_name' => $data['last_name'],
            'email' => $data['email'],
            'password' => Hash::make($data['password']),
            'phone' => $data['phone'],
            'show_phone' => $data['show_phone'],
            'show_email' => $data['show_email'],
        ]);
    }
}
```

Figura 21 cod reg user

En la figura 22 se puede observer el codigo relacionado con la funcion para generar el qr que permitira identificar a la mascota a traves de una interfaz web relacionada con dicho codigo

```php
<?php

namespace App\Http\Controllers;


use Image;
use QrCode;
use App\User;
use App\Pet;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Storage;

class PetController extends Controller
{
    /**
     * Display a listing of the resource.
     *
```

```
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $id = auth()->id();

        if ($id == 1)
        {
            $pets = Pet::all();

            return view('admin.dashboard.pets.admin-index', compact('pets'));
        }
        else
        {
            $user = User::find($id);
            $pets = Pet::where('user_id', $id)->get();

            return view('admin.dashboard.pets.user-
index', compact('user', 'pets'));
        }
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('admin.dashboard.pets.create');
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        $pet = Pet::create($request->all());

        if($request->file('file')){

            $file = $request->file('file');
```

```
            $img = Image::make($request->file('file'))->resize(120, 120);
            $path = Storage::disk('public')->put('img/app/pets/',  $file);
            $pet->fill(['file' => $path])->save();
            $img->save($path);
        }

        return redirect()->route('pets.index')-
>with('info', 'Mascota creada con éxito');
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {
        $pet = Pet::where('id', $id)->first();
        $user = User::find($pet->user_id);

        return view('web.profile', compact('user', 'pet'));
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        $pet = Pet::find($id);
        return view('admin.dashboard.pets.edit', compact('pet'));
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
```

```
        $pet = Pet::find($id);

        $pet->fill($request->all())->save();

        if($request->file('file')){

            $file = $request->file('file');
            $img = Image::make($request->file('file'))->resize(120, 120);
            $path = Storage::disk('public')->put('img/app/pets/',  $file);
            $pet->fill(['file' => $path])->save();
            $img->save($path);
        }

        return redirect()->route('pets.index')-
>with('info', 'Mascota actualizada con éxito.');
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        $pet = Pet::find($id)->delete();

        return redirect()->route('pets.index')-
>with('info', 'Mascota eliminada con éxito.');
    }

    public function qr($id)
    {

        $pet = Pet::find($id);
        return view('admin.dashboard.pets.qr', compact('pet'));
    }
}
```

Figura 22 gen QR

Despúes de todo lo relacionado con el desarrollo de la aplicacion Web se realizo registro de los usuarios (ver figura 23),con sus respectivas mascotas (figura 24), en cada uno, se valida la información, se hacen ediciones,y cambios hasta de contraseña, al igual que los datos de las mascotas, para validar la funcionalidad de la aplicacion.

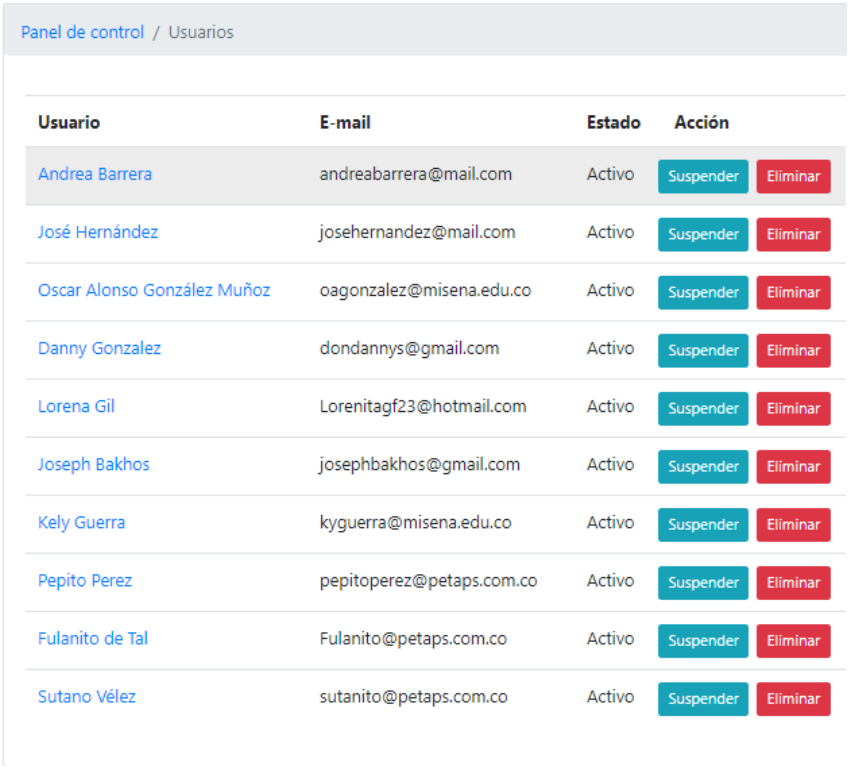

Figura 23 Usuarios creados

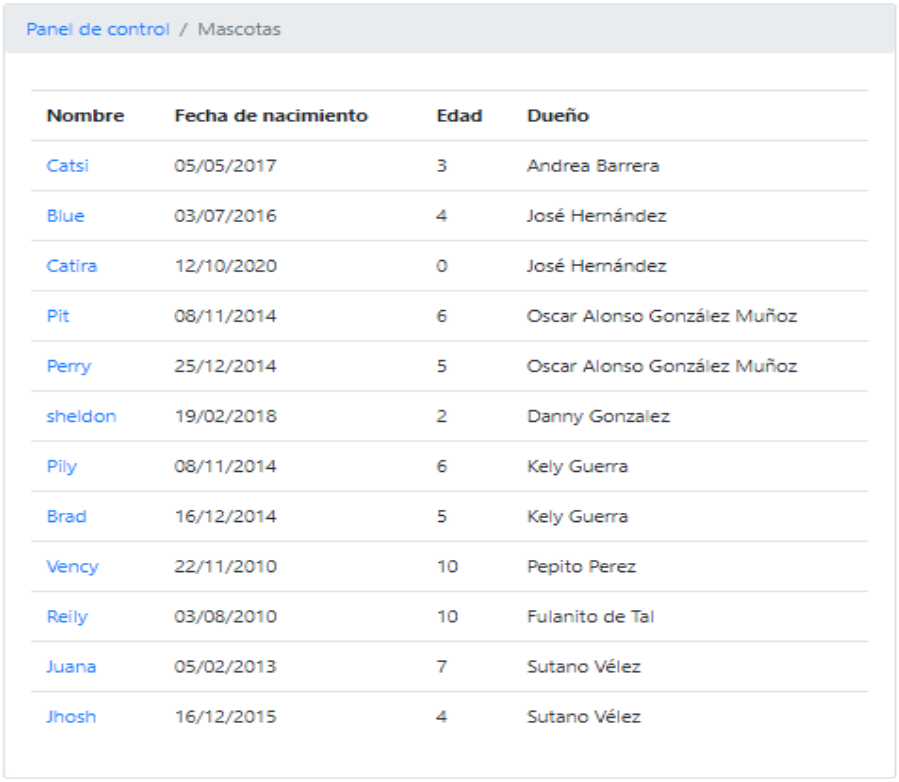

Figura 24 mascotas

Se suspenden y habilitan usuarios, siendo satisfactorios (fig 25 y 26)

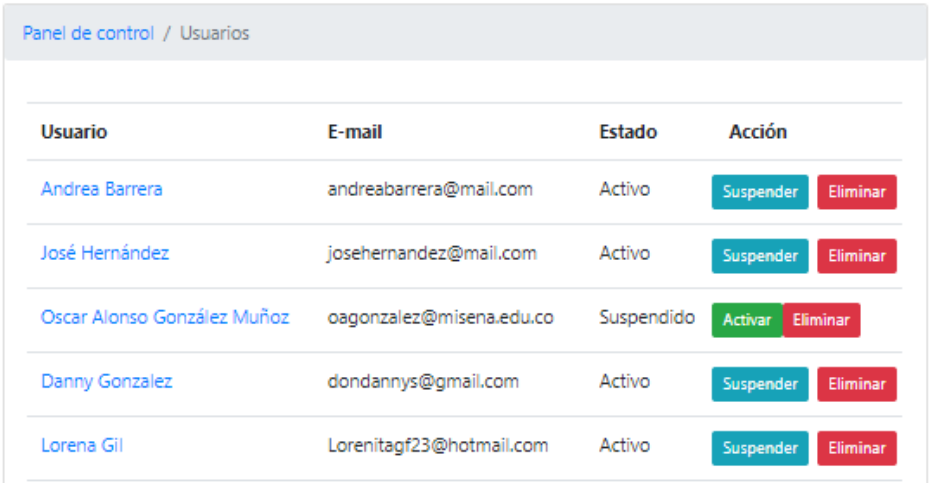

Figura 25 Usuario suspendido

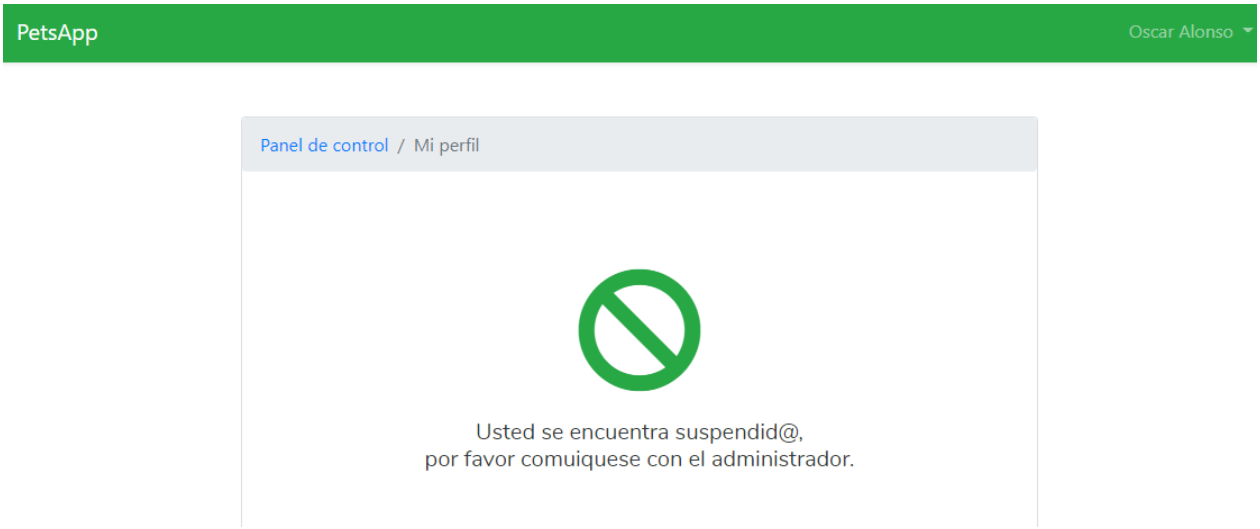

Figura 26 User suspendido

La siguiente figura nos muestra el codigo Qr generado

Figura 27 Qr generado

Se genera Código QR de cada una de las mascotas desde la aplicación web (fig 28)

Figura 28 Qr

Se verifica con el smartphone el enlace a la URL respectiva, cumpliendo con los objetivos propuestos para el proyecto (fig 29 y 30).

Figura 29 ir URL

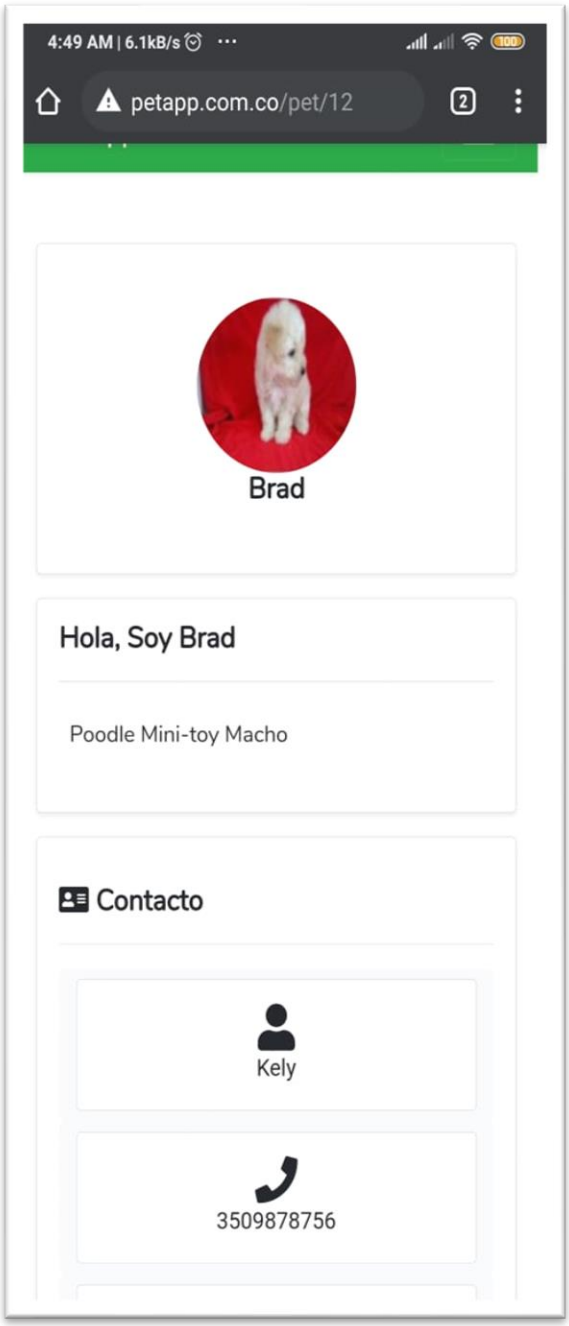

Figura 30 Qr encontrado

La funcionalidad de la aplicación es basada en entornos amigables y entendibles para el usuario final, el cual a la hora de registrarse debe consignar unos datos básicos de contacto con los cuales obtendrá un usuario y una contraseña para el ingreso a la aplicación, luego después de su respectivo registro y logueo, podrá agregar a su perfil las mascotas que quiera, con información básica como nombre, tipo de animal, edad, se puede almacenar, foto y generar un código QR, que puede ser impreso con laser o plasma en medallas de acero inoxidable o aluminio, y con la ayuda de un collar sujetarse al cuelo de la animal, para en caso de ser requerido para identificar, con

solo poner el lector de códigos del celular y un acceso a internet podrá disponer de la información básica del dueño para su respectiva recuperación.

Después de tener claro el panorama de desarrollo, se realizaron pruebas de funcionalidad en donde se crearon 5 usuario ficticios, todos ellos con sus respectivas mascotas, a las cuales se les valido y modifico la información como cambio de fecha de nacimiento, y foto, igual que con datos como el nombre, el número de contacto, o el email, todos ellos satisfactorios, como resultado de este trabajo, se pueden generar los códigos Qr, que son los que se deben grabar en metal para que posteriormente sean leídos por la Cámara del celular y direccionar el navegar a la URL solicitada.

Capítulo 5. Conclusiones y Recomendaciones

Se desarrolló una aplicación Web que cumple con el propósito de brindar una alternativa a los dueños de mascotas, para identificarlas mediante un código QR. El proceso es simple y económico, sólo basta con registrar los datos básicos de la persona en conjunto con los del animal. Adicionalmente la página cuenta con información sobre cuidados de las mascotas, que es alimentada en formato blog, lo cual permite actualizarla fácilmente.

El uso del framework Laravel, hace que los módulos de la aplicación tengan responsabilidades diferenciadas y únicas.

Su implementación hace que puedan adicionarse más funcionalidades, tal como se tiene previsto a futuro, dado que su arquitectura permite que sea extensible.

Lista de referencias

Achour, M. (27 de 07 de 2020). *My php.net*. Obtenido de www.php.net/manual/es/intro-whatis.php

Berners-Lee, T. (1998). *https://www.w3.org.* Obtenido de https://www.w3.org: https://www.w3.org/History/1989/proposal.html

el Tiempo, P. (22 de 05 de 2019). la tecnología al servicio de los animales. Obtenido de https://www.eltiempo.com/contenido-comercial/la-tecnologia-al-servicio-de-los-animales-362552

Garmin, L. (2020). *Garmin*. Recuperado el 2020, de https://buy.garmin.com/en-US/US/p/160887

https://www.php.net/. (2201-2020). *https://www.php.net/.* Obtenido de https://www.php.net/: https://www.php.net/manual/es/history.phttps://www.php.net/hp.php

Humane tribe. (2020). Obtenido de https://m.humanetribe.com/

Juan Avella. (s.f.). *GCF Gobal*. Recuperado el 11 de 2020, de https://edu.gcfglobal.org/es/informatica-basica/que-son-las-aplicaciones-web/1/

kaveri. (2020). *Geekflare*. Obtenido de https://geekflare.com/pet-trackers/

Laravel. (2020). *The PHP Framework for Web Artisans*. Recuperado el 10 de 2020, de http://laravel.com

Palomares, K. (s.f.). *Kiko Palomares*. Obtenido de https://kikopalomares.com/que-es-un-framework-en-programacion-diccionario-del-programador/

Serna, P. P. (01 de 2018). *El palpitar*. Obtenido de http://www.elpalpitar.com/opinion/2018/01/medellin-se-pierden-mas-ocho-animales-dia/

UNITAG. (s.f.). *UNITAG*. Obtenido de https://www.unitag.io/es/qrcode/what-is-a-qrcode

Wei-Dong Jackie Zhu, J. A. (20 agosto 2004). Implementig Web Applications with CM Information Integrator for Content and OnDemand Web Enablement Kit. En J. A.-L. Wei-Dong Zhu, *Implementig Web Applications with CM Information Integrator for Content and OnDemand Web Enablement Kit* (pág. 618 pages). IBM.COM/redbooks.