| | | |
|---|---|---|
| Logotipo universidad  | Nombre de la Universidad "FUNDACIÓN UNIVERSITARIA DEL AREA ANDINA" | Fecha: |

| **PROPONENTES** | | |
|---|---|---|
| NOMBRE: Carlos Alfonso Delgado Santos | DOCUMENTO 1022372023 | FIRMA |
| Sergio Andres Martinez Barreto | 1030664001 | |

| **TITULO DEL PROYECTO** |
|---|
| **PET INFORMATION** |

| **AGRADECIMIENTOS** |
|---|
| El agradecimiento de este proyecto va dirigido en primera instancia a Dios, por la oportunidad que nos brindó de poder formarnos como ingenieros, luego a la fundación universitaria del Área andina de ofrecernos los conocimientos necesarios para lograr la culminación de la carrera y por último, pero no menos importante, el apoyo que recibimos de nuestras familias durante todo el proceso de formación.<br><br>También agradecemos todo el acompañamiento brindado por nuestros docentes y coordinadores de programa en el transcurso de la carrera.<br><br>Gracias a todos. |

## DEDICATORIA

Quiero dedicar este proyecto a mi bella madre, ya que ella siempre me brindo en todo momento su apoyo para lograr mi sueño de ser un profesional.
También lo dedico a mi esposa e hija ya que también estuvieron presentes en el desarrollo de mi estudio y más aún durante su finalización.

-Carlos Alfonso Delgado Santos.

Este proyecto está dedicado primeramente a Dios quien me ha brindado lo necesario para absorber nuevos conocimientos y formarme como profesional.
De igual manera dedico este proyecto a mi familia quienes siempre han creído en mí y en mis capacidades donde siempre fueron mi mayor motivación para afrontar cada reto y ser mejorar continuamente como profesional y como persona.

-Sergio Andrés Martínez Barreto.

## TABLA DE CONTENIDO

## INDICE DE IMAGENES

## INDICE DE TABLAS



## INTRODUCCIÓN

El cuidado que conllevan las mascotas, desde consultas médicas y la importancia que tienen para nuestros hogares, hacen que sus dueños se preocupen por su bienestar, las consultas a veterinarias sea por motivos de salud o algo estético puede llegar a ser algo complejo ya sea por la cantidad de clientes o por la variedad de servicios que ofrecen, para gestionar y ofrecer una buena atención al usuario, es necesario contar con un sistema para poder controlar toda la información de sus clientes y mascotas.

Pet information es un software desarrollado para gestionar la información de las veterinarias, ya que permite un mejor control, organización y además brinda herramientas para la gestión de citas con sus clientes. La veterinaria en la que será implementado el sistema hoy en día no cuenta con herramientas para el manejo de su información, y no tiene publicidad en la web, por lo cual Pet information suplirá esa necesidad permitiendo posicionar el negocio.

## PLANTEAMIENTO DEL PROBLEMA

### DESCRIPCIÓN DEL PROBLEMA

Según la normativa de la ley 1774 del 6 de enero del 2016, sobre la protección de los animales, indica que el individuo responsable o dueño de ellos, debe garantizar como mínimo que la mascota no padezca hambre ni sed, que no sufra de forma injustificada

malestar físico o dolor, y que no le sean generadas enfermedades por descuido o negligencia.

No llevarlos a condiciones de estrés o miedo y dejarlos comportarse de manera natural, estos son algunos de los aspectos que se deben tener en cuenta a la hora tener una mascota. (Bogota.gov.co, 2020)

Hoy en día las personas deben ser más conscientes ante el cuidado de los animales, y normalmente en los hogares puede haber por lo menos una mascota.

En la actualidad el negocio de clínica veterinaria está creciendo y cada vez son más comunes en la sociedad, la competencia es fuerte en este ámbito puesto que existen varias veterinarias y podemos encontrar clínicas que no cuentan con un sistema de gestión que les permita manejar de una manera más organizada la información de sus clientes y especialmente la asignación de citas para la atención de sus mascotas.

El proyecto se desarrollará para la Veterinaria Monster Pet, ofreciendo al usuario una aplicación web con una interfaz gráfica agradable, sencilla y práctica al momento de interactuar con los servicios de la veterinaria. El sistema permitirá ofrecer sus productos y servicios por medio de una aplicación Web, facilitando la solicitud y asignación de citas online para mayor comodidad de sus clientes.

El servicios de asignación de citas de forma telefónica como se maneja actualmente en la clínica veterinaria, hace que se requiere dedicar tiempo en este proceso para la atención de los clientes, cuando este proceso se puede agilizar más con la asignación y cancelación de citas en línea, llevar de forma ordenada y facilitar la actualización del historial clínico de sus clientes, almacenado en un repositorio de datos con la seguridad necesaria que garantice la protección de la información. adicional a esto el portal web le permitirá informar de forma inmediata a sus clientes información relacionada con campañas de vacunación, publicidad, promociones, etc.

## FORMULACIÓN DEL PROBLEMA

El proyecto tiene como finalidad gestionar y mejorar los procesos en la atención de los clientes de la veterinaria por medio de una aplicación web en la cual podrán realizar la solicitud de citas para sus mascotas. El software está dirigido principalmente a microempresas en el sector de las clínicas veterinarias siendo este un mercado potencial para la aplicación del sistema.

¿La implementación de una aplicación web para la veterinaria Monster Pet le permitirá mejorar los procesos internos de asignación de citas y la atención de sus clientes?

## JUSTIFICACION

Las tecnologías hoy en día presentan unas grandes ventajas para las empresas ya que están en pro de mejorar sus sistemas de información, logrando con esto ser un poco más competitivas en el mercado, además de que facilitan la administración de los procesos en cada una de sus dependencias.

El proyecto está enfocado en dar solución a las necesidades que presenta actualmente la veterinaria Monster pet, además de generar más participación de la empresa en el sector, y aportará a mejorar el servicio de la atención de los clientes.

Para lo cual los principales pilares de este proyecto será reducir los tiempos en la asignación de citas y evitar el registro en aplicaciones como “Microsoft Word” o “Microsoft Excel”, ya que no siempre son la mejor forma de gestionar la información por lo que no es posible acceder a todos los archivos y datos creados de manera ágil e inmediata.

## OBJETIVO GENERAL DEL PROYECTO

Desarrollar e implementar, una aplicación para la clínica veterinaria Monster Pet que facilite la asignación de citas y control de la información de sus clientes.

## OBJETIVOS ESPECÍFICOS

- Implementar roles de acceso a la aplicación para registrar y gestionar de forma apropiada la información.
- Elaborar un módulo en el cual se pueda brindar información acerca de los productos y servicios que ofrece la clínica veterinaria.
- Diseñar e implementar el módulo de gestión de citas ofreciendo a los clientes un servicio ágil en la solicitud y asignación de citas en la clínica veterinaria.
- Crear repositorio de almacenamiento para la información de los clientes y sus mascotas garantizando la seguridad de los datos.

- Elaborar y diseñar la interfaz gráfica del aplicativo web por medio de tecnologías que facilite la visualización de la información de la veterinaria.

## MARCO DE REFERENCIA

### 1.1 MARCO TEÓRICO

Para sustentar el desarrollo del proyecto, es importante indagar sobre cada uno de los elementos teóricos.

#### 1.1.1 Programación orientada a objetos

Según (Aguilar, 1998) La POO se detalla como un paradigma de la programación, una forma de programar específica, en la que se organiza el código en unidades bautizadas clases, de las cuales se generan objetos que se relacionan entre sí para conseguir los objetivos de las aplicaciones.

La programación orientada a objetos es una manera singular de programar, más apropiada a cómo reflejaremos las cosas en la vida real.

Al programar bajo el paradigma de POO tendremos que estudiar y pensar cómo resolver los problemas de una manera diferente a como lo realizamos anteriormente, en la programación estructurada. En este caso tendremos que escribir nuestras aplicaciones en forma de objetos, clases, propiedades, métodos y las características principales que cumple este tipo de programación como son la abstracción, herencia, encapsulamiento y el polimorfismo.

Las clases son las plantillas para la creación de objetos, también se definen como abstracciones de objetos. Esto indica que la definición de un objeto es la clase. Cuando creamos un objeto y establecemos sus características y funcionalidades, en realidad lo que estamos haciendo es generar una clase.

Los objetos son muestras de una clase. Cuando creamos una muestra tenemos que especificar la clase a partir de la que se creará. La acción de generar un objeto a partir de una clase se llama instanciar el objeto. (Pérez, J. 2015)

#### 1.1.2 Bases de datos

Las Bases de Datos Según (Davies, 2014). Son un conjunto de datos relacionados y organizados, estos son recogidos, almacenados y gestionados por los sistemas de información de una compañía.

SGBD

Los Sistemas de Gestión de Base de Datos (DataBase Management System) son un tipo de software específico, que brinda una interfaz entre el usuario, la Base de datos y las aplicaciones que la utilizan. Se conforma de un lenguaje de definición de datos, de un lenguaje de manejo de datos y de un lenguaje de consulta.

### 1.1.3 Aplicaciones Web

Las aplicaciones web comúnmente se componen de tres niveles:
Nivel superior el que interactúa con el usuario (Navegador).
Nivel intermedio el que procesa la información (Servidor web).
Nivel inferior el que suministra los datos (Base de datos).

Las aplicaciones web son un modelo de software que se desarrolla en un lenguaje soportado por los navegadores web y su ejecución es realizada por el navegador en Internet, por tal razón reciben el nombre de aplicación Web.

Una página web puede tener varios elementos que permiten una comunicación entre el usuario y la información, permitiendo que el usuario acceda a los datos de manera rápida e interactiva, puesto que la página web se encargará de responder todas las peticiones que éste ejecute, por ejemplo, ingresar a varios gestores de bases de datos, como también publicar e interactuar con los contenidos, llenar y enviar formularios, etc.

Las aplicaciones web están profundamente relacionadas con el almacenamiento de información en la nube, puesto que toda la información se registra de forma permanente en servidores web, los que además de albergar dicha información, la envía a los dispositivos móviles como también a equipos de cómputo en cada momento requerido, generando copias temporales de estos envíos dentro de los dispositivos y equipos que utilicemos.

Las compañías que adquieren estos espacios en los servidores web son conocidas actualmente como empresas o servicios de Hosting.

Las aplicaciones web son muy acogidas y apetecidas por las siguientes razones:

- La operatividad a la hora de ejecutarlas en los navegadores web.

- No depende del sistema operativo que se use en el equipo de cómputo o dispositivo móvil.
- La comodidad de actualizar y mantener aplicaciones web sin la necesidad de tener que distribuir el software.
- La libertad de acceso que pueden tener los usuarios en cualquier momento y dispositivo, con tan solo tener la conexión a una red de internet. (Luján, 2002)

### 1.1.4 FrameWork

Un Framework o marco de trabajo, es la estructura que se establece y que se utiliza para desarrollar y disponer un software en concreto. La definición podría resumirse como el entorno pensado para realizar más sencilla la programación de un sistema o aplicación.

Este sistema genera varias ventajas para los desarrolladores, puesto que automatiza varios procesos y también facilita la programación. Es bastante útil para evitar el tener que repetir código y para realizar funciones constantes en un rango de herramientas, como por ejemplo puede ser el acceder a bases de datos o realizar llamadas a Internet. Todas las tareas se realizan de forma mucho más fácil cuando se trabaja dentro de un framework.

Propone demasiadas ventajas y, además, permite lograr también tareas mucho más complejas en la que por otros medios, serían imposibles de plantear siquiera a la hora de programar. No obstante, su implementación es algo que depende del tipo de programa y contexto en el que vaya a emplear.

Un Framework en esencia es útil para lograr escribir código o desarrollar una aplicación de manera más fácil. Permite una mejor organización y control de todo el código trabajado, así como una probable reutilización en el futuro. De acuerdo a lo anterior, asegura una mayor productividad que los métodos más convencionales y un recorte del coste al optimizar las horas de trabajo empleadas en el desarrollo.

Su implementación es algo que afecta también a los errores, puesto que los minimiza considerablemente. En conclusión, es algo que brinda una ayuda general y más que considerable al desarrollador, haciendo que sus tareas sean mucho más sencillas. (Rockcontent 2020).

### 1.1.5 Sistema de información

Un sistema de información (SI) se define como un conjunto ordenado de mecanismos que tienen como objetivo la administración de datos e información, de tal forma que puedan ser procesados fácil y rápidamente.

Cualquier sistema de información está compuesto de una variedad de recursos interconectados y en interacción, aptos del modo más conveniente en base al propósito informativo establecido, como puede ser alcanzar información personal, procesar estadísticas, organizar archivos, etc. Estos recursos pueden ser:

- Recursos humanos. Personas con diferentes roles..
- Actividades. Métodos, pasos a seguir o estaciones de trabajo.
- Datos. Cualquier tipo de información masiva que requiere ordenarse.
- Recursos informáticos. Determinados por la tecnología.

Es importante tener claro que no es lo mismo un sistema de información que un sistema informático, si bien estos últimos representan constantemente el grueso de los recursos de un SI. (Rey, 2011)

### 1.1.6 MYSQL

MySQL es conocido como un sistema de gestión de base de datos relacional de código abierto el cual se basa en un lenguaje de consulta estructurado (SQL).(Quintana, 2014)

### 1.1.7 Arquitectura de software

Se conoce como los patrones o lineamientos que cumplen a ayudar en la construcción de un programa o aplicación.
Los lineamientos permiten tener una orientación para los developers, analistas, arquitectos de software y cargos relacionados para así lograr el objetivo de cumplir con cada uno de los requerimientos planteados para cada aplicación. (Campderrich, 2013).

### 1.1.8 Modelo Vista controlador

MVC más conocido como Model View Controller es un patrón de arquitectura de software bastante conocida y usada por gran parte de desarrolladores ya que permite organizar el código de tal manera que mantiene distintas capas que se encargan de hacer una tarea en concreto ofreciendo diversos beneficios de optimización de código.
MVC usualmente es usado en sistemas que se requiera el uso de interfaces de usuario, aunque eso no quiere decir que no sea posible la práctica de este patrón de arquitectura en distintos tipos de aplicaciones.

Surge tras la necesidad de crear los sistemas con un ciclo de vida más adecuado y robusto en donde se pueda realizar de manera fácil y ágil el mantenimiento, incluso la reutilización de código y segmentación de conceptos. (López, 2015).

## METODOLOGIA

Metodología RUP

Conocido también por sus siglas como Proceso de Desarrollo Unificado, donde se compone por un conjunto de metodologías completamente adaptables a la necesidad de la organización.

Principios de Metodología RUP:

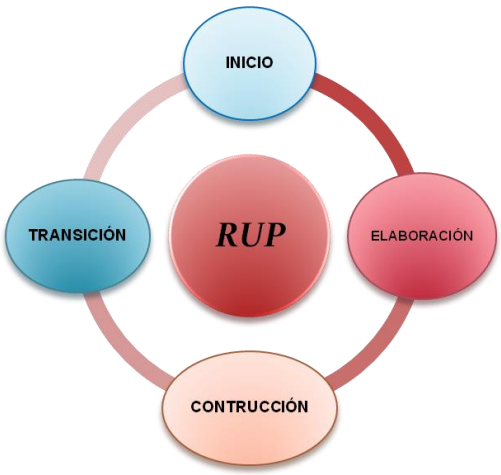
1. Adaptación del proceso
2. Valanciar prioridades
3. Demostración del valor iterativamente
4. Cooperación entre equipos definidos.
5. Enfoque en la calidad.
6. Elevación de niveles de abstracción.

**CARACTERÍSTICAS:**

- Es una forma organizada de asignar responsabilidades y tareas
- Manejo de control de cambios
- Arquitecturas basadas en componentes
- Proporciona muchas ventajas al brindar énfasis en los requerimientos de diseño.
- No se requiere de un grupo tan numeroso de desarrolladores para la aplicación de esta metodología.
- Permite rediseñarse constantemente.
- Permite usar frameworks de testeo para validación de pruebas.
- Desarrollo iterativo. (Báez y Suárez, M. I. 2013).

**Fases del ciclo de vida del RUP:**

*Figura 1: Ciclo de vida RUP*




Recuperado de https://iutoms7024grupo7.wordpress.com/2015/03/02/metodologia-rup-proceso-racional-unificado/

**1.Fase de Inicio:** Tiene como intención establecer la dimensión que abarca el proyecto, y también poder identificar los riesgos asociados al proyecto, de igual manera proponer un alcance general de la arquitectura del software, y generar un plan de trabajo para las fases posteriores.

Durante esta primera fase se realizarán diferentes entrevistas y encuestas con las personas directamente involucradas en la veterinaria, de la cual se obtendrán los requerimientos que se necesitan para la construcción del sistema de información.

**2.Fase de elaboración:** La fase de elaboración se establecen los casos de uso que permitirán definir la arquitectura del sistema los cuales se construirán en esta fase. Se genera la especificación de los casos de uso seleccionados y el primer análisis del problema, se diseña la solución preliminar.

Durante esta fase se diseñarán los casos de usos necesarios para el desarrollo del Software y se implementará un bosquejo de la interfaz gráfica del aplicativo para así tener una idea clara de la estructura.

**3.Fase de Desarrollo:** En la fase 3 se completa la funcionalidad del sistema, por lo cual se deben aclarar los requerimientos pendientes, administrar los cambios de acuerdo con las pruebas realizadas por los usuarios y se realizan las mejoras pertinentes para el proyecto.

En esta fase se realizará la programación en código PHP y HTML con su respectiva conexión a la base de datos creada en XAMPP, se realizará la creación del módulo de asignación de citas y se mejorará la interfaz gráfica implementando Boostrap el cual es un Framework de CSS para maquetar profesionalmente la capa de presentación de la aplicación.

**4.Fase de Cierre:** En la fase de cierre se asegura que el software esté disponible para los usuarios finales, se ajustan los errores y defectos encontrados en las pruebas realizadas, también se debe capacitar a los usuarios en el uso del aplicativo. Se debe comprobar que el producto cumpla con las especificaciones establecidas por las personas involucradas en el proyecto.

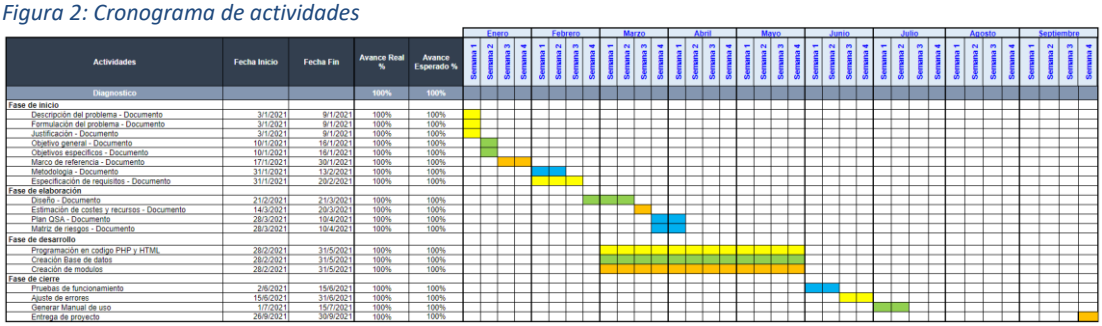
Culminando en esta fase se realizarán las distintas pruebas como la asignación de una cita, registro y navegación de la interfaz para determinar que no quede ningún error, y que el software fluya de una manera eficiente y satisfaga las necesidades para las cuales fue desarrollado. (rupandcmmi.blogspot, 2017).

## PLANIFICACIÓN DEL PROYECTO

Cronograma de actividades

Ver anexo A cronograma de actividades.

*Figura 2: Cronograma de actividades*



| Actividades | Fecha Inicio | Fecha Fin | Avance Real % | Avance Esperado % |
|---|---|---|---|---|
| Diagnostico | | | 100% | 100% |
| Fase de inicio | | | | |
| Descripción del problema - Documento | 3/1/2021 | 9/1/2021 | 100% | 100% |
| Formulación del problema - Documento | 3/1/2021 | 9/1/2021 | 100% | 100% |
| Justificación - Documento | 3/1/2021 | 9/1/2021 | 100% | 100% |
| Objetivo general - Documento | 10/1/2021 | 16/1/2021 | 100% | 100% |
| Objetivos específicos - Documento | 10/1/2021 | 16/1/2021 | 100% | 100% |
| Marco de referencia - Documento | 17/1/2021 | 30/1/2021 | 100% | 100% |
| Metodología - Documento | 31/1/2021 | 13/2/2021 | 100% | 100% |
| Especificación de requisitos - Documento | 31/1/2021 | 20/2/2021 | 100% | 100% |
| Fase de elaboración | | | | |
| Diseño - Documento | 21/2/2021 | 21/3/2021 | 100% | 100% |
| Estimación de costes y recursos - Documento | 14/3/2021 | 20/3/2021 | 100% | 100% |
| Plan QSA - Documento | 28/3/2021 | 10/4/2021 | 100% | 100% |
| Matriz de riesgos - Documento | 28/3/2021 | 10/4/2021 | 100% | 100% |
| Fase de desarrollo | | | | |
| Programación en codigo PHP y HTML | 28/2/2021 | 31/5/2021 | 100% | 100% |
| Creación Base de datos | 28/2/2021 | 31/5/2021 | 100% | 100% |
| Creación de modulos | 28/2/2021 | 31/5/2021 | 100% | 100% |
| Fase de cierre | | | | |
| Pruebas de funcionamiento | 2/6/2021 | 15/6/2021 | 100% | 100% |
| Ajuste de errores | 15/6/2021 | 31/6/2021 | 100% | 100% |
| Generar Manual de uso | 1/7/2021 | 15/7/2021 | 100% | 100% |
| Entrega de proyecto | 26/9/2021 | 30/9/2021 | 100% | 100% |

Fuente: Propia

## E.R.S Especificación de requisitos

### Requerimientos funcionales

| Id | Nombre | Requerimiento funcional | Descripción |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **Rf 001** | Administrador | El sistema debe contar con una cuenta de administrador | El administrador es el único usuario con los privilegios de registrar, editar y eliminar los registros del sistema |
| **Rf 002** | Gestionar usuarios | El sistema debe permitir administrar las cuentas de usuario. | El sistema permite registrar, actualizar, eliminar y buscar datos de usuario. |
| **Rf 003** | Roles de usuario | El sistema permite asignar los roles a cada tipo de usuario | El sistema debe asignar los roles de usuario según su función como: Administrador, empleado y cliente. |
| **Rf 004** | Gestionar citas | El sistema debe permitir gestionar citas para los usuarios que estén registrados. | El sistema permite al usuario cliente crear cita y asignarla de acuerdo con la agenda de la veterinaria. De acuerdo a la solicitud del cliente se genera cita según la necesidad. El Administrador y cliente tienen la posibilidad de cancelar la cita. |
| **Rf 005** | Gestionar mascotas | El sistema le debe permitir al usuario cliente, gestionar la información de las mascotas. | El sistema permite que el cliente pueda gestionar la información de las mascotas. Agregar foto y descripción detallada de la mascota. |
| **Rf 006** | Ver historial | El sistema debe permitir visualizar el historial de las mascotas | El sistema permite al cliente, administrador y empleado poder visualizar el historial de las mascotas. |
| **Rf 0007** | Buscar usuarios | El sistema le permite al administrador buscar usuarios registrados en el sistema. | El sistema debe listar información del usuario según código asignado en el sistema. |

*Tabla 1:Requerimientos funcionales.*

## Requerimientos No funcionales

| **Id** | **Nombre** | **Requerimiento No funcional** | **Descripción** |
|---|---|---|---|
| **RNF 001** | Validar login | El sistema valida las cuentas de ingreso. | El sistema debe validar los datos del usuario para iniciar sesión. |
| **RNf 002** | Validar Registro | El sistema valida las cuentas de correo. | El sistema valida los datos del usuario registrado para evitar más de un login con el mismo usuario. |

| | | | |
|---|---|---|---|
| **RNf 003** | Validar rol de usuario | El sistema valida el rol del usuario | El sistema identifica y da privilegios a los usuarios tipo administrador, empleado y cliente. |
| **RNf 004** | Validar Cita | El sistema debe Validar cita | El sistema debe validar la información seleccionada y registrada por el usuario tipo cliente para evitar duplicar la asignación de la cita. |
| **RNf 005** | Verificar Cita | El sistema de verificar cita | El sistema le permite al usuario tipo cliente verificar la asignación de su cita. |
| **RNf 006** | interfaz gráfica amigable | El sistema debe tener una interfaz gráfica. | El sistema debe tener una interfaz gráfica de fácil acceso y manejo para el usuario. |
| **RNf 007** | Mantener sesión activa | El sistema debe permitir mantener la cuenta del usuario activa una vez iniciada sesión | El sistema debe permitir mantener la cuenta de cualquier usuario activa una vez se haya iniciado sesión. |
| **RNf 008** | Interacción | El sistema debe responder de una forma óptima a la solicitud del cliente. | El sistema debe dar una respuesta rápida a las peticiones del cliente. |

*Tabla 2: Requerimientos no funcionales.*

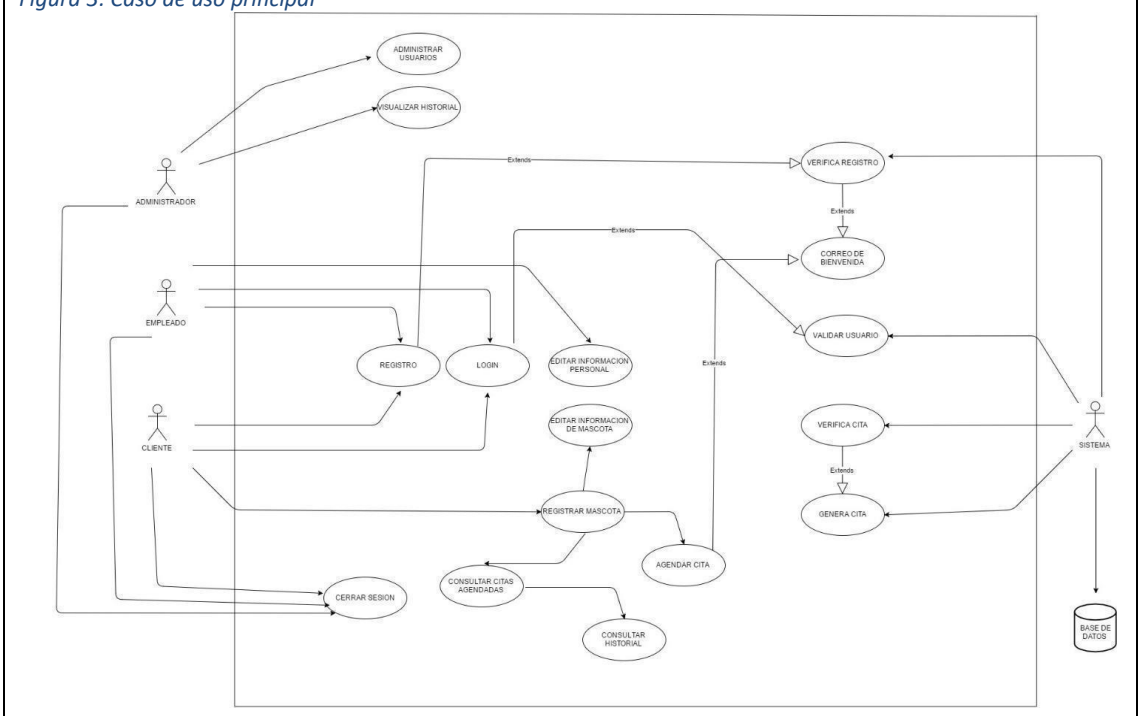
## DISEÑO

### DIAGRAMAS

### CASOS DE USO

Los casos de uso son las descripciones de las actividades para llevar a cabo un proceso, las entidades que participan en el diagrama de caso de uso se denominan actores. En la imagen 3 podemos visualizar el caso de uso principal del proyecto.

*Figura 3: Caso de uso principal*




Fuente: Propia

| Nombre | Registro | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Registro | |
| | | |
| Precondición | Seleccionar la opción de registro | |
| Descripcion | Para complementar el registro, el usuario debe ingresas una serie de datos obligatorios por medio de un formulario, encargado de enviar el registro del usuario al sistema.<br><br>Al realizar el proceso de registro, el usuario recibe un correo de confirmación. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Ingresar datos personales |
| | 2 | Clic en el botón "registrarse" |
| | 3 | El sistema valida el usuario |
| | 4 | El sistema envía un correo de bienvenida |
| | | |

| Pos condiciones | Diligenciar los campos obligatorios |
|---|---|

| Excepciones | Paso | Acción |
|---|---|---|
| | 1 | Se debe digitar solo los tipos de valor solicitados |
| | 2 | Mensaje alerta "tipo de valor no valido" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | Mensaje "el usuario ya está registrado" |
| | 5 | Sin conexión, mensaje alerta "no se pudo establecer conexión con la base de datos" |

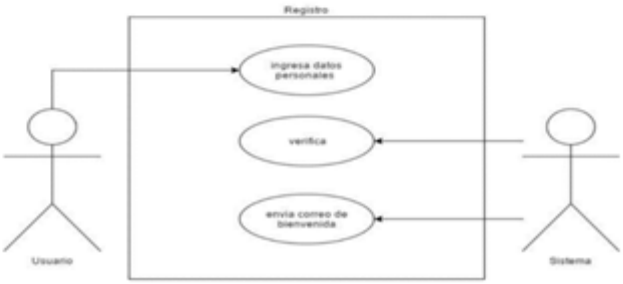
| Caso de uso |  |
|---|---|



*Tabla 3: Caso de uso registro.*

| Nombre | Logueo | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Tiene acceso a solicitar cita, extiende cerrar sesión | |
| | | |
| Precondición | El usuario está registrado en el sistema (tiene una cuenta). | |
| Descripcion | El proceso de logueo del usuario se inicia cuando el cliente empieza a diligenciar los campos de texto información requerida como nombre, usuario y contraseña, selecciona el boton de ingreso y la base de datos verifica la existencia de dicho usuario. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Ingresar usuario y contraseña |
| | 2 | Clic en el botón "ingresar" |
| | 3 | El sistema valida la existencia del usuario |
| | 4 | El usuario accede a la plataforma. |
| | | |

| Pos condiciones | Diligenciar los campos obligatorios |
|---|---|

| Excepciones | Paso | Acción |
|---|---|---|
| | 1 | Se debe digitar los datos registrados anteriormente |
| | 2 | Mensaje alerta "usuario o contraseña incorrecta" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | Sin conexión, mensaje alerta "no se pudo establecer conexión con la base de datos |
| | 5 | |

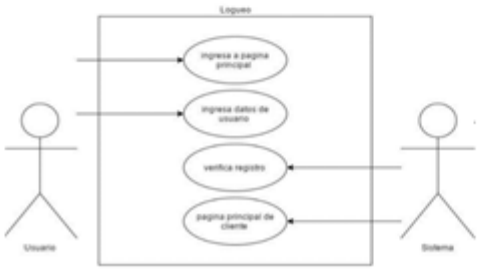
| Caso de uso |  |
|---|---|



*Tabla 4: Caso de uso logueo.*

| Nombre | Agendar cita y registrar mascota | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Incluye logueo, extiende solicitar cita | |
| | | |
| Precondición | Seleccionar la opción de solicitud de cita y tener una mascota | |
| Descripcion | Después de que el usurario se loguea tiene acceso al menú para solicitar citas, y es indispensable que el cliente tenga una mascota para completar el proceso. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Ingresar datos personales |
| | 2 | Ingresar datos de la mascota y foto opcional. |
| | 3 | Seleccionar una fecha y hora para la cita, motivo de la cita |
| | 4 | El sistema valida calendario y registra la cita solicitada |
| | | |

| Pos condiciones | | |
|---|---|---|

| Excepciones | Paso | Acción |
|---|---|---|
| | 1 | Sin conexión, mensaje alerta |
| | 2 | Mensaje error "error, no se puede solicitar cita" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | "no se pudo establecer conexión con la base de datos" |
| | 5 | |

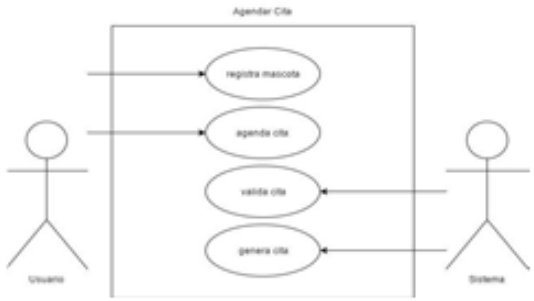
| Caso de uso |  |
|---|---|



*Tabla 5: Caso de uso agendar cita y registrar mascota.*

| Nombre | Editar información | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Incluye editar información del usuario y de la mascota | |
| | | |
| Precondición | Haberse logueado | |
| Descripcion | Cuando el usuario a ingresado con su usuario al sistema puede editar la información personal como número telefónico y actualizar la información de su mascota. | |

| | Paso | Acción |
|---|---|---|
| Secuencia normal | 1 | Ingresar al sistema. |
| | 2 | Seleccionar el módulo de citas. |
| | 3 | Ingresar a mis mascotas y a mis datos. |
| | 4 | El sistema valida información y actualiza. |
| | | |
| Pos condiciones | | |

| | Paso | Acción |
|---|---|---|
| Excepciones | 1 | Sin conexión, mensaje alerta |
| | 2 | Mensaje error "error, no se puede guardar datos" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | "no se pudo establecer conexión con la base de datos" |
| | 5 | |

Caso de uso


*Tabla 6: Caso de uso editar información.*

| Nombre | Consultar citas | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Incluye consultar historial citas agendadas y cancelarlas | |
| | | |
| Precondición | Haber solicitado cita y registrado mascota | |
| Descripcion | Cuando el usuario agenda una cita y registra su mascota, tiene acceso para visualizar las próximas citas de su mascota. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Ingresar al módulo de citas. |
| | 2 | Seleccionar el módulo de citas |
| | 3 | Ingresar a mis mascotas y visualizar en mis citas, puede visualizar las citas agenda das o cancelarla opcional. |
| | 4 | |
| | | |

| Pos condiciones | |
|---|---|

| Excepciones | Paso | Acción |
|---|---|---|
| | 1 | Sin conexión, mensaje alerta |
| | 2 | Mensaje error "no hay citas agendadas" |
| | 3 | Mensaje alerta "no se puede cancelar cita comuníquese con el administrador del sistema" |
| | 4 | "no se pudo establecer conexión con la base de datos" |
| | 5 | |

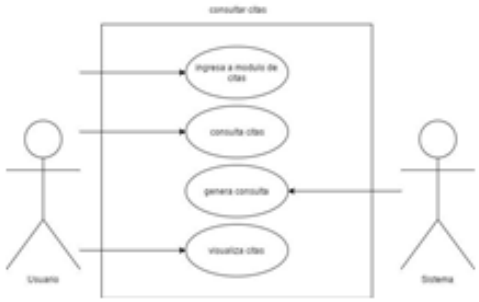
| Caso de uso |  |
|---|---|



*Tabla 7: Caso de uso consultar citas.*

| Nombre | Historial de mascotas | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Incluye editar información del usuario y de la mascota, además de visualizar su historial. | |
| | | |
| Precondición | Haberse logueado | |
| Descripcion | El usuario administrador y empleado son los únicos que tienen acceso al historial de las mascotas.<br>Estos usuarios solo son creados por el administrador del sistema. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Loguearse y dirigirse al módulo de citas |
| | 2 | Seleccionar la pestaña reportes |
| | 3 | Seleccionar clientes/mascotas y editar información |
| | 4 | Ver comentarios de la cita y su historial |
| | | |

| Pos condiciones | | |
|---|---|---|
| Excepciones | Paso | Acción |
| | 1 | Sin conexión, mensaje alerta |
| | 2 | Mensaje error "error, no se puede guardar datos" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | "no se pudo establecer conexión con la base de datos" |
| | 5 | |

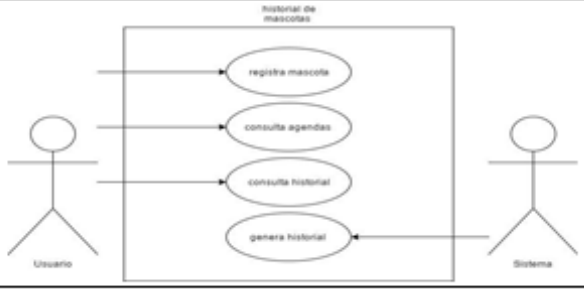
| Caso de uso |  |
|---|---|



*Tabla 8: Caso de uso historial de mascotas.*

| Nombre | Administrar usuarios | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Incluye gestionar usuarios tipo administrador en el sistema | |
| | | |
| Precondición | | |
| Descripcion | Iniciar sesión como usuario administrador, ingresar al módulo de usuario, puede crear y ver el reporte de los usuarios vinculados.<br><br>Validar el estado del usuario, eliminar y agregar usuarios. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Ingresar como administrador |
| | 2 | Ir al módulo de usuarios |
| | 3 | Creación y reporte usuarios |
| | 4 | validar estado y editar usuarios |
| | | |

| Pos condiciones | | |
|---|---|---|
| Excepciones | Paso | Acción |
| | 1 | Sin conexión, mensaje alerta |
| | 2 | Mensaje error "error, no se puede guardar datos" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | "no se pudo establecer conexión con la base de datos" |
| | 5 | |

| Caso de uso |  |
|---|---|



*Tabla 9: Caso de uso administrar usuarios.*

| Nombre | Cerrar sesión | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | Incluye salir del módulo de citas y edición de datos personales. | |
| | | |
| Precondición | | |
| Descripcion | Cerrar sesión representa dar por terminado el uso de la plataforma. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Ingresar a la plataforma. |
| | 2 | Realizar acciones como agregar mascota y asignar cita. |
| | 3 | Terminar tareas de edición e interacción con el sistema. |
| | 4 | Seleccionar módulo de usuario y salir. |
| | | |

| Pos condiciones | | |
|---|---|---|
| Excepciones | Paso | Acción |
| | 1 | Sin conexión, mensaje alerta |
| | 2 | Mensaje error "error, no se puede guardar datos" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | "no se pudo establecer conexión con la base de datos" |
| | 5 | |

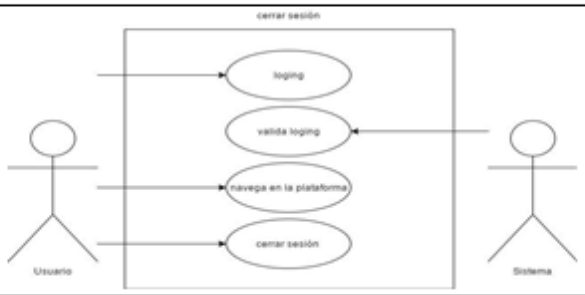
| Caso de uso |  |
|---|---|



*Tabla 10: Caso de uso cerrar sesión.*

| Nombre | Sistema | CU-01 |
|---|---|---|
| Fecha de creación | 10 de enero del 2021 | V-001 |
| | | |
| Requisito asociado | El administrador del sistema puede visualizar por medio de filtros información completa de todos los usuarios vinculados al sistema. | |
| | | |
| Precondición | | |
| Descripcion | El administrador del sistema posee acceso total a la base de datos, puede eliminar usuarios inactivos, limpiar historial, asi como visualizar movimientos recientes como citas, usuarios y mascotas nuevas.<br><br>La principal herramienta para el administrador son los filtros, por medio de palabras clave puede buscar de manera agil la información. | |

| Secuencia normal | Paso | Acción |
|---|---|---|
| | 1 | Ingresar al sistema como administrador |
| | 2 | Ingresar directamente a la base de datos |
| | 3 | Administrar información y filtrar búsqueda |
| | 4 | Realizar y guardar cambios. |
| | | |

| Pos condiciones | | |
|---|---|---|
| Excepciones | Paso | Acción |
| | 1 | Sin conexión, mensaje alerta |
| | 2 | Mensaje error "error, no se puede guardar datos" |
| | 3 | Mensaje alerta "todos los campos son obligatorios" |
| | 4 | "no se pudo establecer conexión con la base de datos" |
| | 5 | |

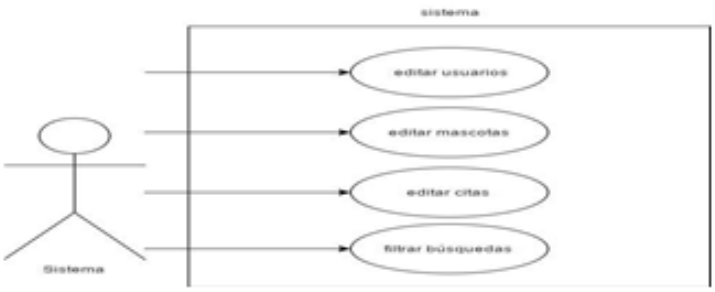
| Caso de uso |  |
|---|---|



*Tabla 11: Caso de uso sistema.*

## DIAGRAMA DE CLASES

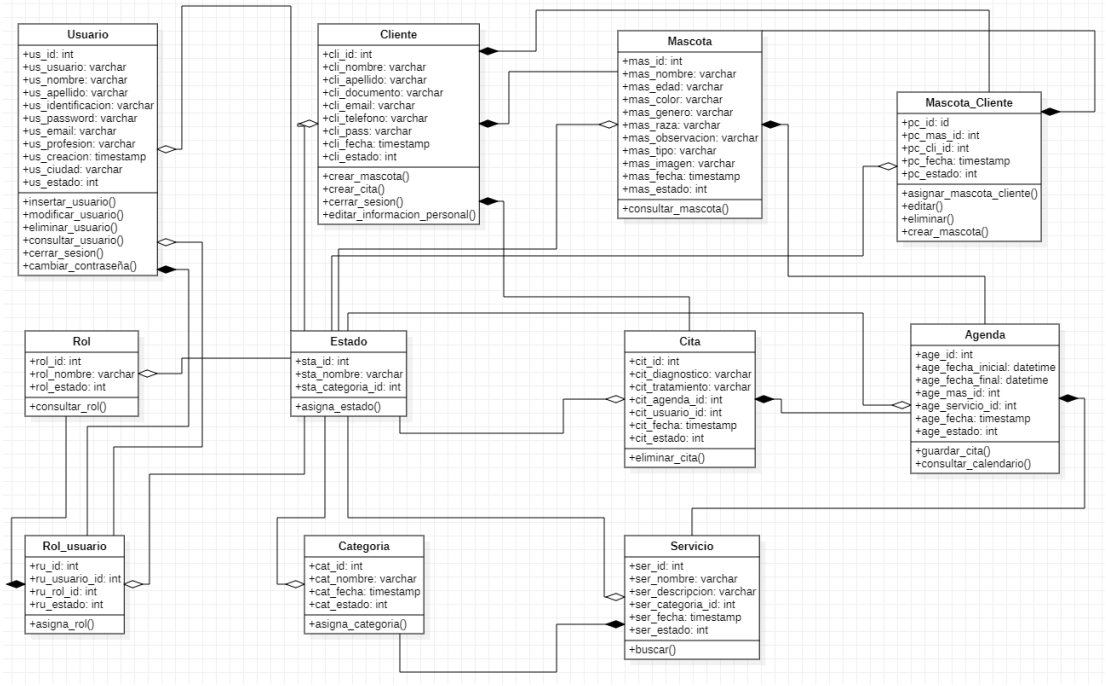
El diagrama de clases en lenguaje unificado de modelado (UML) es el diagrama que describe la estructura del sistema permitiendo visualizar las clases, atributos, operaciones y relaciones de los objetos. En la imagen 4 podemos ver el diagrama de clases del proyecto.

En el sistema encontramos 11 clases con sus respectivos atributos y operaciones, las relaciones que se encuentran son de tipo composición y agregación

Composición: Se representa con un rombo lleno en la clase cuya instancia contiene las instancias de la otra clase.

Agregación: Se representa con un rombo hueco en la clase cuya instancia es una agregación de las instancias de la otra.

*Figura 4: Diagrama de clases.*




Fuente: Propia.

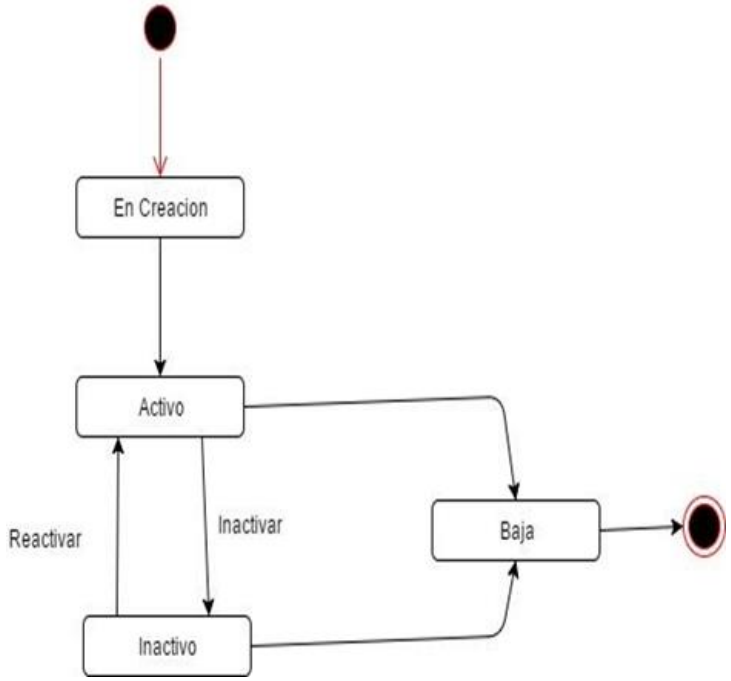
## DIAGRAMA DE ESTADOS

Los diagramas de estados describen el comportamiento de los sistemas a continuación podremos observar los desarrollados para los usuarios y citas de la aplicación.

### Diagrama de estados usuarios

En el siguiente diagrama, se puede observar que los usuarios en la aplicación contarán con 3 estados, el primero es activo al momento de crearse, el segundo se podrá inactivar si se requiere, y por último dar de baja.

*Figura 5: Diagrama de estado usuarios.*

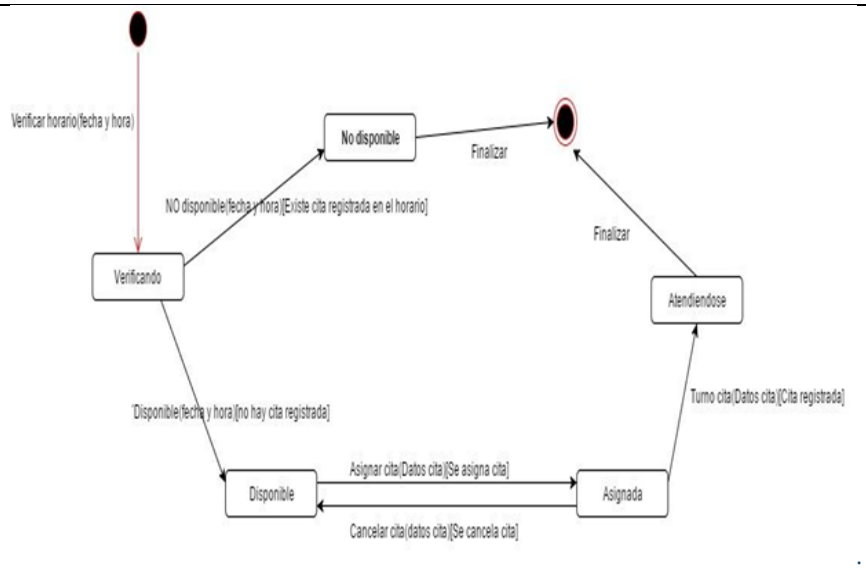



Fuente: Propia.

**Diagrama de estados cita**

En el siguiente diagrama se podrán observar los estados que tienen las citas en la aplicación. El sistema verifica si hay fecha y hora disponible para agendar, cuando no hay disponibilidad finaliza el proceso, pero si hay fecha y hora habilitada esta se podrá agendar para que posteriormente se atienda en la veterinaria y finalice.

*Figura 6: Diagrama de estados cita.*




Fuente: Propia.

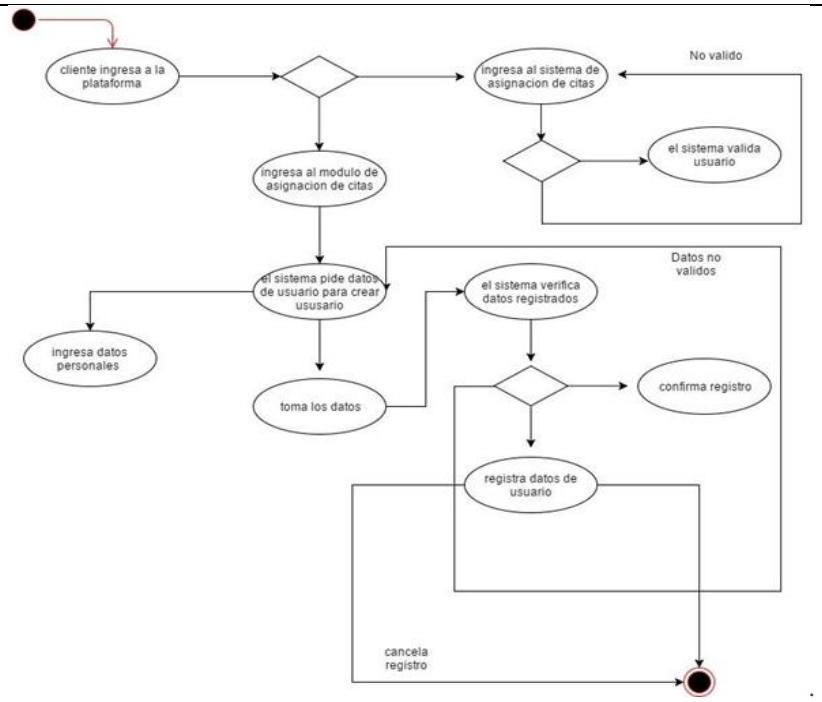
## DIAGRAMA DE ACTIVIDADES

Los diagramas de flujo son los que muestran las actividades ejecutadas por un sistema. A continuación, se podrán observar los desarrollados para el aplicativo.

### Diagrama de registro

El aplicativo permite que el usuario cliente pueda registrar mascotas al momento de agendar citas, por lo cual al momento de ingresar, el sistema valida que exista el usuario para poder continuar.

*Figura 7: Diagrama de procesos registro.*

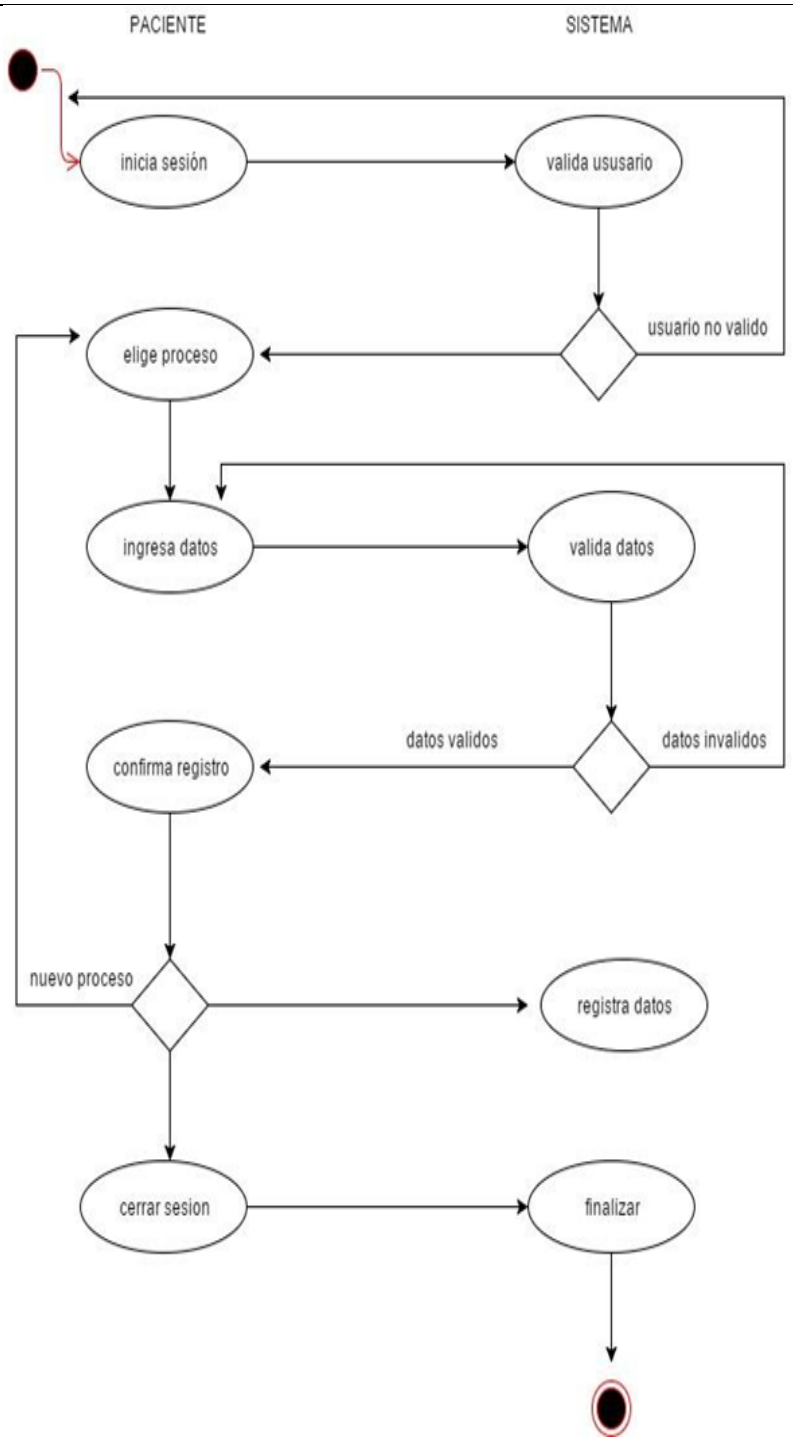



Fuente: Propia.

**Diagrama de proceso login-usuario**

El sistema valida que el usuario que inicie sesión exista en la base de datos para así permitir su ingreso, y finaliza el proceso cuando cierra sesión.

*Figura 8: Diagrama de procesos Login-Usuario.*

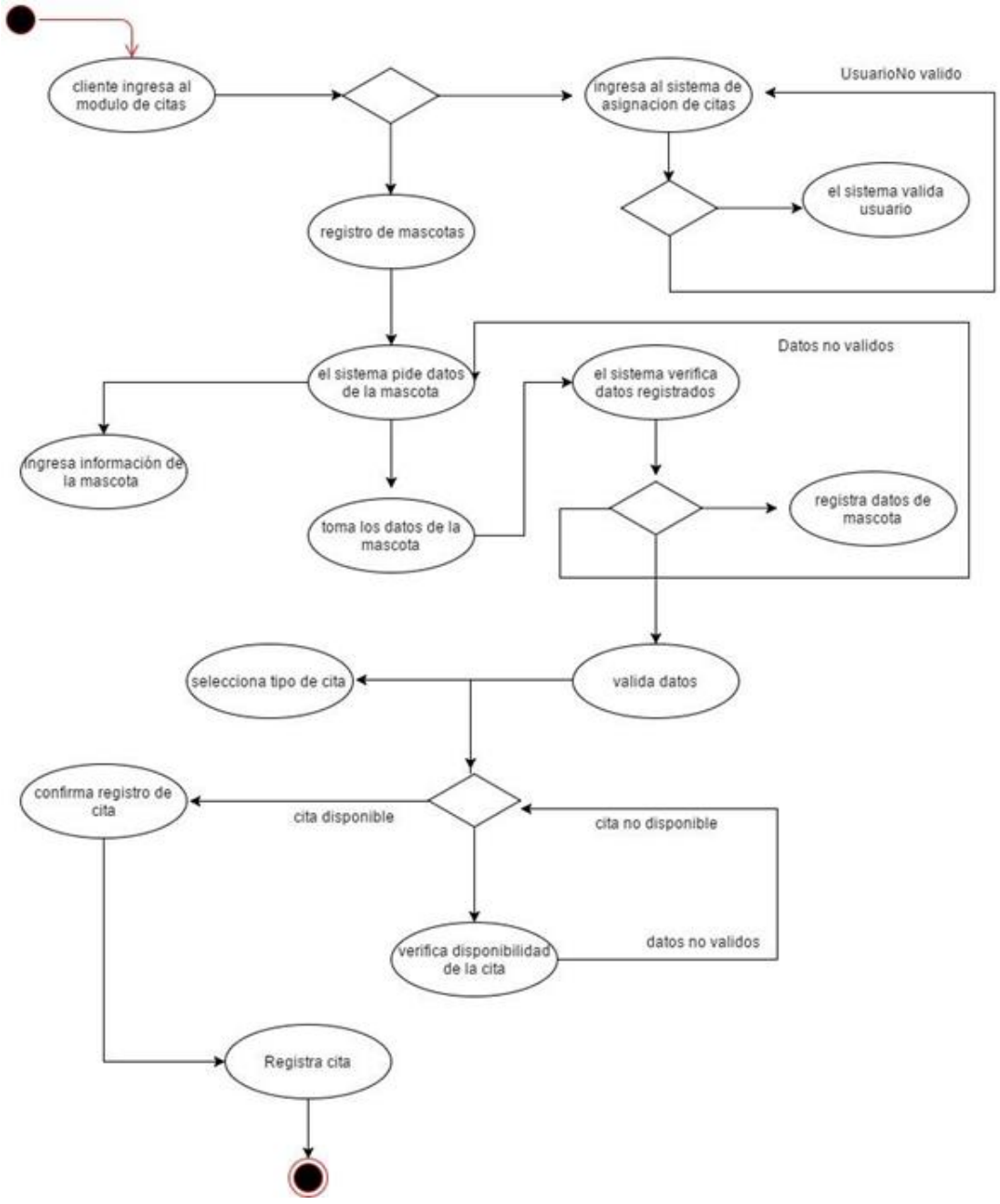



Fuente: Propia.

**Diagrama de asignación de cita**

El usuario cliente debe ingresar al módulo de citas, seguidamente debe registrar una mascota para poder continuar. Cuando seleccione la mascota podrá escoger el tipo de cita y agendarla.

*Figura 9: Diagrama de procesos asignación de citas.*



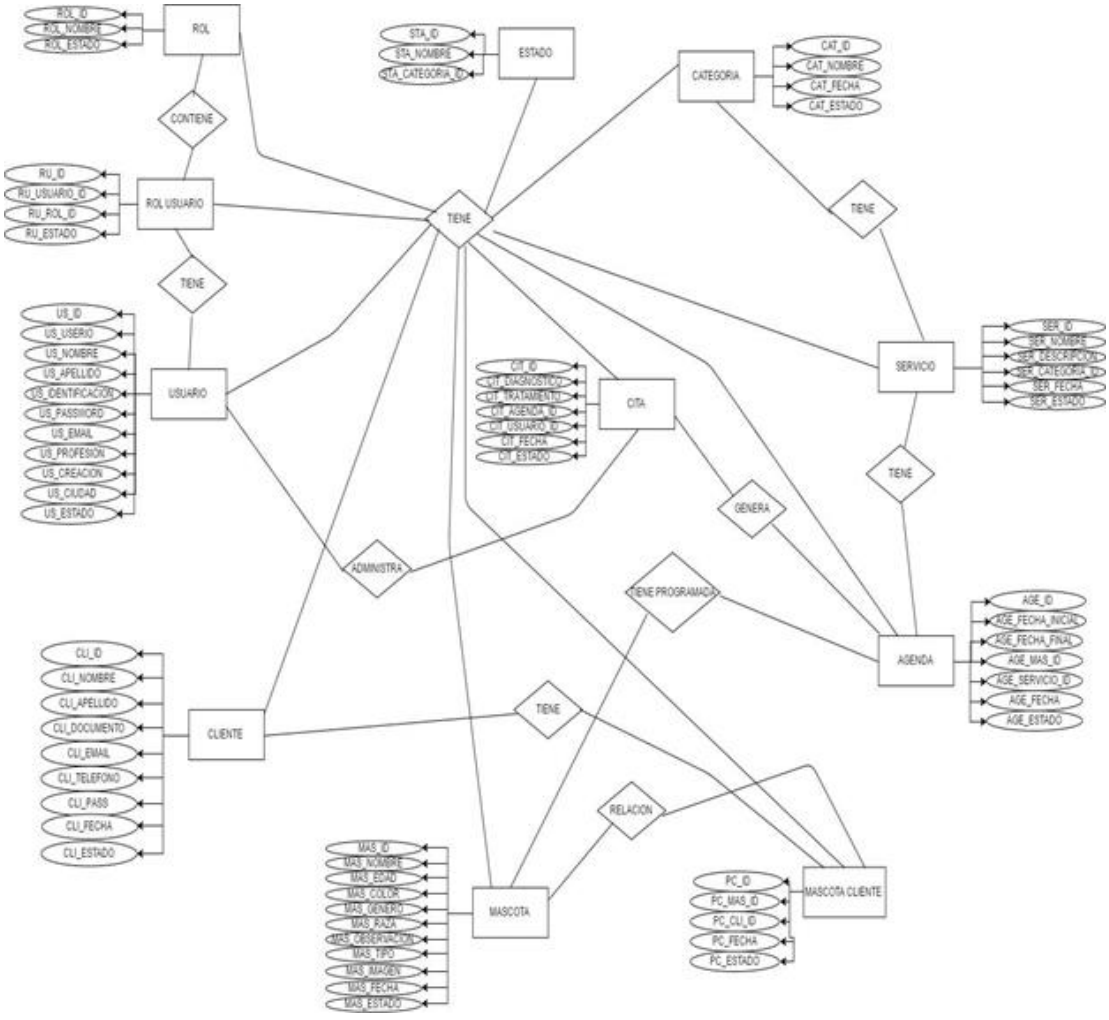

Fuente: Propia.

## MODELO DE DATOS

## MODELO ENTIDAD RELACIÓN

El modelo entidad relación es la herramienta utilizada para el modelado de datos, el cual permite representar las entidades de la base de datos. En la imagen 10 podemos observar el modelo entidad relación del proyecto.

*Figura 10: Modelo entidad relación.*

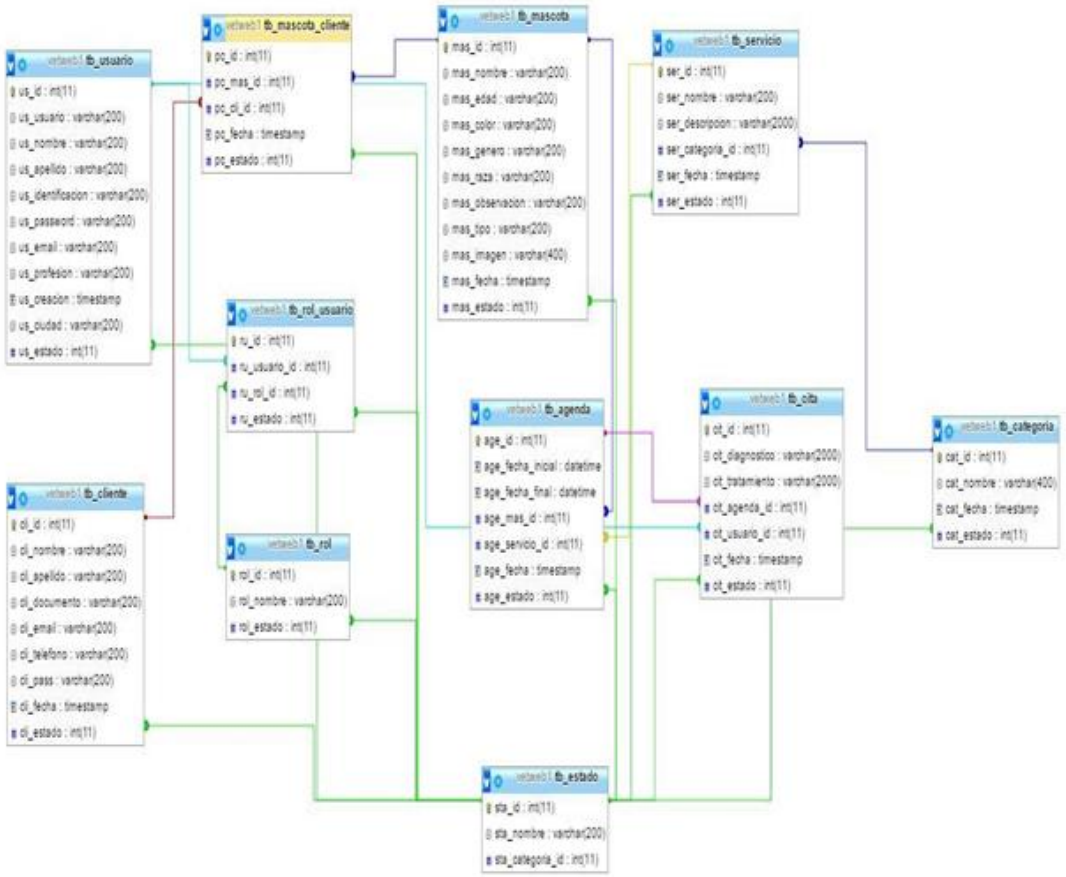



Fuente: Propia.

## MODELO RELACIONAL

En el modelo relacional representan los datos por medio de tablas relacionadas, las cuales permiten conformar la base de datos. En la imagen 11 podemos observar el modelo relacional del proyecto el cual está compuesto por 11 tablas con sus respectivos campos.

*Figura 11: Modelo relacional.*




Fuente: Propia.

# DICCIONARIO DE DATOS

## Agenda

*Figura 12: Diccionario de datos Agenda.*

### tb_agenda

Comentarios de la tabla: tabla donde se almacenan los agendamientos

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| age_id *(Primaria)* | int(11) | No | | | |
| age_fecha_inicial | datetime | No | | | |
| age_fecha_final | datetime | No | | | |
| age_mas_id | int(11) | No | | tb_mascota -> mas_id | |
| age_servicio_id | int(11) | No | | tb_servicio -> ser_id | |
| age_fecha | timestamp | No | CURRENT_TIMESTAMP | | |
| age_estado | int(11) | No | | tb_estado -> sta_id | |

**Índices**

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | age_id | 11 | A | No | |
| age_mas_id | BTREE | No | No | age_mas_id | 11 | A | No | |
| age_estado | BTREE | No | No | age_estado | 11 | A | No | |
| age_servicio_id | BTREE | No | No | age_servicio_id | 11 | A | No | |

## Categoría

*Figura 13: Diccionario de datos Categoría.*

### tb_categoria

Comentarios de la tabla: tabla donde se almacenan los tipos de formulario case, conta

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| cat_id *(Primaria)* | int(11) | No | | | |
| cat_nombre | varchar(400) | No | | | |
| cat_fecha | timestamp | No | CURRENT_TIMESTAMP | | |
| cat_estado | int(11) | No | | tb_estado -> sta_id | |

**Índices**

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | cat_id | 4 | A | No | |
| cat_estado | BTREE | No | No | cat_estado | 2 | A | No | |

## Cita

*Figura 14: Diccionario de datos Cita.*

## tb_cita

Comentarios de la tabla: Tabla donde se almacena el proceso realizado en la cita

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| cit_id *(Primaria)* | int(11) | No | | | |





| | | | | | |
|---|---|---|---|---|---|
| cit_diagnostico | varchar(2000) | No | | | |
| cit_tratamiento | varchar(2000) | No | | | |
| cit_agenda_id | int(11) | No | | tb_agenda -> age_id | |
| cit_usuario_id | int(11) | No | | tb_usuario -> us_id | |
| cit_fecha | timestamp | No | CURRENT_TIMESTAMP | | |
| cit_estado | int(11) | No | | tb_estado -> sta_id | |

### Índices

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | cit_id | 5 | A | No | |
| cit_agenda_id | BTREE | No | No | cit_agenda_id | 5 | A | No | |
| cit_usuario_id | BTREE | No | No | cit_usuario_id | 5 | A | No | |
| cit_estado | BTREE | No | No | cit_estado | 2 | A | No | |

# Cliente

*Figura 15:Diccionario de datos Cliente.*

## tb_cliente

Comentarios de la tabla: tabla donde se registran los posibles clientes

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| cli_id *(Primaria)* | int(11) | No | | | |
| cli_nombre | varchar(200) | No | | | |
| cli_apellido | varchar(200) | No | | | |
| cli_documento | varchar(200) | No | | | |
| cli_email | varchar(200) | No | | | |
| cli_telefono | varchar(200) | No | | | |
| cli_pass | varchar(200) | No | | | |
| cli_fecha | timestamp | No | CURRENT_TIMESTAMP | | |
| cli_estado | int(11) | No | | tb_estado -> sta_id | |

### Índices

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | cli_id | 3 | A | No | |
| cli_estado | BTREE | No | No | cli_estado | 3 | A | No | |

# Estado

*Figura 16: Diccionario de datos Estado.*

## tb_estado

Comentarios de la tabla: Tabla donde se almacenan los estados del sistema

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| sta_id *(Primaria)* | int(11) | No | | | |
| sta_nombre | varchar(200) | No | | | |





| | | | | | |
|---|---|---|---|---|---|
| sta_categoria_id | int(11) | No | | | |

### Índices

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | sta_id | 4 | A | No | |
| sta_categoria_id | BTREE | No | No | sta_categoria_id | 4 | A | No | |

## Mascota

*Figura 17: Diccionario de datos Mascota*

## tb_mascota

Comentarios de la tabla: Tabla donde se almacenan las mascotas

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| mas_id *(Primaria)* | int(11) | No | | | |
| mas_nombre | varchar(200) | No | | | |
| mas_edad | varchar(200) | No | | | |
| mas_color | varchar(200) | No | | | |
| mas_genero | varchar(200) | No | | | |
| mas_raza | varchar(200) | No | | | |
| mas_observacion | varchar(200) | No | | | |
| mas_tipo | varchar(200) | No | | | |
| mas_imagen | varchar(400) | Sí | *NULL* | | |
| mas_fecha | timestamp | No | CURRENT_TIMESTAMP | | |
| mas_estado | int(11) | No | | tb_estado -> sta_id | |

### Índices

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | mas_id | 9 | A | No | |
| mas_estado | BTREE | No | No | mas_estado | 2 | A | No | |

## Mascota/cliente

*Figura 18: Diccionario de datos Mascota/Cliente.*

## tb_mascota_cliente

Comentarios de la tabla: Tabla donde se relacionan las mascotas con los clientes

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| pc_id *(Primaria)* | int(11) | No | | | |
| pc_mas_id | int(11) | No | | tb_mascota -> mas_id | |
| pc_cli_id | int(11) | No | | tb_cliente -> cli_id | |
| pc_fecha | timestamp | No | CURRENT_TIMESTAMP | | |
| pc_estado | int(11) | No | | tb_estado -> sta_id | |





### Índices

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | pc_id | 4 | A | No | |
| pc_mas_id | BTREE | No | No | pc_mas_id | 4 | A | No | |
| pc_cli_id | BTREE | No | No | pc_cli_id | 4 | A | No | |
| pc_estado | BTREE | No | No | pc_estado | 2 | A | No | |

# Rol

*Figura 19: Diccionario de datos Rol.*

## tb_rol

Comentarios de la tabla: Tabla donde se almacenan los roles del sistema

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| rol_id *(Primaria)* | int(11) | No | | | |
| rol_nombre | varchar(200) | No | | | |
| rol_estado | int(11) | No | | tb_estado -> sta_id | |

### Índices

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | rol_id | 3 | A | No | |
| rol_estado | BTREE | No | No | rol_estado | 3 | A | No | |

# Rol usuario

*Figura 20: Diccionario de datos Rol Usuario.*

## tb_rol_usuario

Comentarios de la tabla: tabla donde se relacionan los usuarios y los roles

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| ru_id *(Primaria)* | int(11) | No | | | |
| ru_usuario_id | int(11) | No | | tb_usuario -> us_id | |
| ru_rol_id | int(11) | No | | tb_rol -> rol_id | |
| ru_estado | int(11) | No | | tb_estado -> sta_id | |

**Índices**

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | ru_id | 2 | A | No | |
| ru_estado | BTREE | No | No | ru_estado | 2 | A | No | |
| ru_rol_id | BTREE | No | No | ru_rol_id | 2 | A | No | |
| ru_usuario_id | BTREE | No | No | ru_usuario_id | 2 | A | No | |

## Servicio

*Figura 21: Diccionario de datos Servicio.*

## tb_servicio





Comentarios de la tabla: Tabla donde se almacenan los servicios ofrecidos

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| ser_id *(Primaria)* | int(11) | No | | | |
| ser_nombre | varchar(200) | No | | | |
| ser_descripcion | varchar(2000) | Sí | *NULL* | | |
| ser_categoria_id | int(11) | No | | tb_categoria -> cat_id | |
| ser_fecha | timestamp | No | CURRENT_TIMESTAMP | | |
| ser_estado | int(11) | No | | tb_estado -> sta_id | |

**Índices**

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | ser_id | 5 | A | No | |
| ser_estado | BTREE | No | No | ser_estado | 5 | A | No | |
| ser_categoria_id | BTREE | No | No | ser_categoria_id | 5 | A | No | |

## Usuario

*Figura 22: Diccionario de datos Usuario.*

## tb_usuario

Comentarios de la tabla: TAbla donde se almacenan los usuarios del sistema

| Columna | Tipo | Nulo | Predeterminado | Enlaces a | Comentarios |
|---|---|---|---|---|---|
| us_id *(Primaria)* | int(11) | No | | | |
| us_usuario | varchar(200) | No | | | |
| us_nombre | varchar(200) | No | | | |
| us_apellido | varchar(200) | No | | | |
| us_identificacion | varchar(200) | No | | | |
| us_password | varchar(200) | No | | | |
| us_email | varchar(200) | No | | | |
| us_profesion | varchar(200) | No | | | |
| us_creacion | timestamp | No | CURRENT_TIMESTAMP | | |
| us_ciudad | varchar(200) | No | | | |
| us_estado | int(11) | No | | tb_estado -> sta_id | |

**Índices**

| Nombre de la clave | Tipo | Único | Empaquetado | Columna | Cardinalidad | Cotejamiento | Nulo | Comentario |
|---|---|---|---|---|---|---|---|---|
| PRIMARY | BTREE | Sí | No | us_id | 2 | A | No | |
| us_estado | BTREE | No | No | us_estado | 2 | A | No | |

Fuente: Propia.

## CÓDIGO DE LA APLICACIÓN

Para el desarrollo y codificación de este proyecto, recurrimos a la instalación e implementación del Framework CodeIgniter eficaz para la construcción de aplicaciones web basado en una arquitectura de desarrollo MVC (Model - View - Controller) compuesto principalmente de la siguiente manera:

**HTML:** Es una codificación abstracta que usan las aplicaciones para representar documentos definidos por etiquetas delimitadas por los símbolos < y >.

*Figura 23: Código HTML.*

index.html

```
<!DOCTYPE html>
<html Lang="es">
    <head>
        <title>Monster Pet</title>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <link href="assets/bootstrap/css/bootstrap.css" rel="stylesheet" type="text/css"/>
        <link href="assets/font-awesome-4.7.0/css/font-awesome.css" rel="stylesheet" type="text/css"/>
        <link href="assets/css/style.css" rel="stylesheet" type="text/css"/>
        <link href="assets/fontello/css/fontello.css" rel="stylesheet" type="text/css"/>
        <link href="assets/css/responsive.css" rel="stylesheet" type="text/css"/>
        <script src="assets/jquery/jquery-3.1.1.min.js" type="text/javascript"></script>
        <script src="assets/bootstrap/js/bootstrap.js" type="text/javascript"></script>
    </head>
    <body id="myPage" data-spy="scroll" data-target=".navbar" data-offset="60">
        <nav class="navbar navbar-default navbar-fixed-top">
            <div class="container">
                <div class="navbar-header">
                    <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                    </button>
                    <a class="navbar-brand" href="#myPage">Clinica Veterinaria</a>
                </div>
                <div class="collapse navbar-collapse" id="myNavbar">
                    <ul class="nav navbar-nav navbar-right">
                        <li><a href="#about">ACERCA</a></li>
                        <li><a href="#services">SERVICIOS</a></li>
                        <li><a href="#portfolio">PRODUCTOS</a></li>
                        <li><a href="#contact">CONTACTO</a></li>
                        <li><a href="zona-de-clientes"><i class="glyphicon glyphicon-log-in"></i></a></li>
                    </ul>
                </div>
            </div>
        </nav>
        <div class="jumbotron text-center" style="padding-top: 120px">
            <h1>Monster Pet</h1>
            <p>Su mascota merece lo mejor</p>
        </div>
        <!-- Container (About Section) -->
        <div id="about" class="container-fluid">
            <div class="row">
                <div class="col-sm-6">
                    <h2>NOSOTROS</h2><br>
                    <h4>El equipo de Monster Pet comparte la filosofia de brindar el mejor cuidado y atención a
                    <p>
                        Nuestro mayor orgullo es integrar en nuestra familia a tus mascotas, con un trato person
                        Monster Pet atiende desde el año 2014, desde entonces hemos cosechado un gran número de
                    </p>
                    <br>
                </div>
```

Fuente: Propia.

**PHP:** Este proyecto está basado en la arquitectura de programación MVC por lo cual el código PHP lo tenemos segmentado de la siguiente manera:

El modelo: dentro del modelo definimos las clases en la cual cumplen como función especial el recibir, insertar, actualizar e incluso eliminar información correspondiente a la base de datos de manera organizada.

*Figura 24: Código PHP Modelo.*

RolModel.php

sys > application > models > RolModel.php > RolModel > showAll

```php
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

Class RolModel extends CI_MODEL {

    public function __construct() {

        parent::__construct();
    }

    public function showActive() {

        $sql = "SELECT
                    `rol_id` id,
                    `rol_nombre` name,
                    `rol_estado` state
                FROM
                    `tb_rol` tr
                WHERE
                    tr.rol_estado=1
                    ";

        $query = $this->db->query($sql);
        if ($query->num_rows() > 0) {
            return $query->result();
        }
    }

    //obtenemos los registros
    public function showAll() {

        $sql = "SELECT
                    `rol_id` id,
                    `rol_nombre` name,
                    `rol_estado` state
                FROM
                    `tb_rol` tr
                    where
                    rol_estado=1;
                    ";

        //$query = $this->db->get('tb_rol');
        $query = $this->db->query($sql);
        if ($query->num_rows() > 0) {

            return $query->result();
        }
    }

}

/*
*Location: application/models/crud_model.php
*/
```

Fuente: Propia.

La vista: Es todo aquello que va de cara al usuario en sus interfaces gráficas, por lo

general son documentos HTML en donde se encuentra insertado contenido PHP de manera dinámica generando diversas plantillas permitiéndonos hacer uso de ellas en distintos escenarios según la necesidad permitiéndonos la optimización de código.

*Figura 25: Código PHP vista.*

dashboard.php

sys > application > views > layouts > admin > dashboard.php > ul.nav.nav-tabs

```
<ul class="nav nav-tabs">
    <li class="dropdown">
        <a class="dropdown-toggle" data-toggle="dropdown" href="#">Usuarios
            <span class="caret"></span></a>
        <ul class="dropdown-menu">
            <li>
                <a class="ajaxProcessSingle" action="user" data="create" answer="#response" href="#">Crear</a></li>
            <li>
                <a class="ajaxProcessSingle" action="user" data="show" answer="#response" href="#">reporte</a>
            </li>
        </ul>
    </li>
    <li class="dropdown hidden">
        <a class="dropdown-toggle" data-toggle="dropdown" href="#">Campañas
            <span class="caret"></span></a>
        <ul class="dropdown-menu">
            <li>
                <a class="ajaxProcessSingle" action="campaign" data="create" answer="#response" href="#">Crear</a></li>
            <li>
                <a class="ajaxProcessSingle" action="campaign" data="" answer="#response" href="#">Reporte</a>
            </li>
        </ul>
    </li>
    <li class="dropdown">
        <a class="dropdown-toggle" data-toggle="dropdown" href="#">Reportes
            <span class="caret"></span></a>
        <ul class="dropdown-menu">
            <li>
                <a class="ajaxProcessSingle" action="client" data="generalReport" answer="#response" href="#">Clientes</a></li>
            <li>
                <a class="ajaxProcessSingle" action="pet" data="generalReport" answer="#response" href="#">Mascotas</a>
            </li>
            <li>
                <a class="ajaxProcessSingle" action="schedule" data="generalReport" answer="#response" href="#">Citas</a>
            </li>
        </ul>
    </li>
    <li class="dropdown">
        <a class="dropdown-toggle" data-toggle="dropdown" href="#"><?= ucwords($this->session->userdata('userName')) ?>
            <span class="caret"></span></a>
        <ul class="dropdown-menu">
            <li><a class="updateMyInfo puntero" data-toggle="tab" >Mis datos</a></li>
            <li><a class="logOut" data-toggle="tab" href="#logOut">Salir</a></li>
        </ul>
    </li>
</ul>
<hr />
<div id="response"><img src="<?=base_url()?>assets/images/banner.jpg" class="img-responsive" /></div>
```

Fuente: Propia.

El controlador: cumple como función mediar entre el modelo, la vista y cualquier otro recurso necesario para realizar el procesamiento de peticiones HTTP o procesar la información para generar una página web dinámica. Su comportamiento natural es recibir una petición, prosigue a validar una entrada, selecciona la vista deseada y le envía el contenido al modelo para que realice la acción deseada en la base de datos con su respectiva confirmación del proceso realizado.

*Figura 26: Código PHP Controlador.*

Pet.php

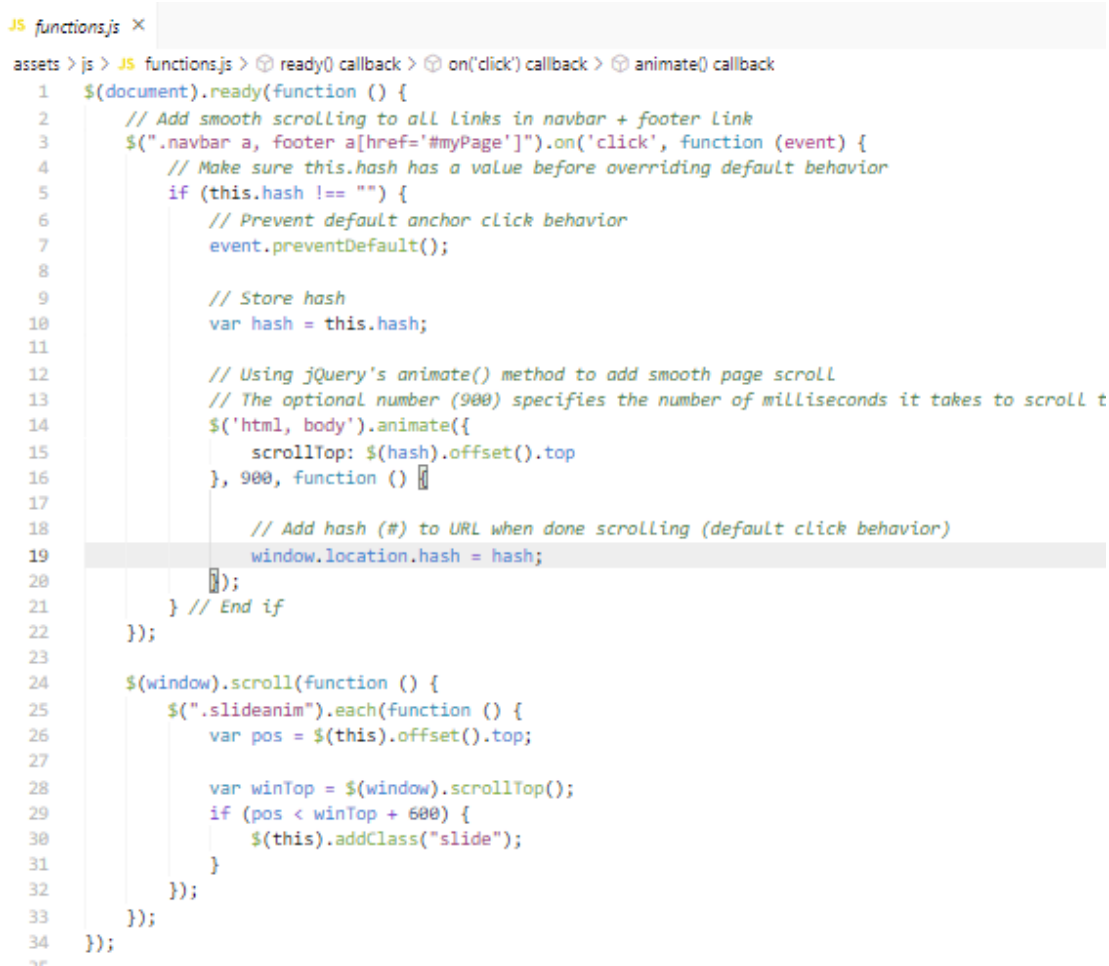
sys > application > controllers > Pet.php > Pet > edit

```
143         }
144
145         public function newPet() {
146
147             if ($this->input->is_ajax_request())
148             {
149                 $obj = (object) $this->input->post(null, false);
150                 $obj->ok = msjSuccess("Mascota guardada correctamente");
151                 $obj->okAssign = msjSuccess("Mascota vinculada correctamente");
152                 $obj->error = msjWarning("Ocurrió un error al guardar la mascota");
153                 $obj->errorAssign = msjWarning("Ocurrió un vincular la mascota");
154
155                 $obj->value = "";
156                 $obj->text = "Seleccione...";
157                 $obj->selected = "selected";
158                 $obj->age = $this->htmltags->options($obj);
159                 for ($x = 1; $x < 20; $x++) {
160                     $obj->value = $x;
161                     $obj->text = $x;
162                     $obj->selected = "";
163                     $obj->age .= $this->htmltags->options($obj);
164                 }
165
166                 $obj->layout = 'pet/edit';
167                 $obj->finalAnswer = $this->htmltags->replaceOfView(
168                         [
169                     tag("id"),
170                     tag("valSubmit"),
171                     tag("valName"),
172                     tag("valAge"),
173                     tag("valColor"),
174                     tag("valRace"),
175                     tag("valObservation"),
176                     tag("actionClass"),
177                     tag("actionToDo"),
178                         ], [
179                     "",
180                     "Registrar",
181                     "",
182                     $obj->age,
183                     "",
184                     "",
185                     "",
186                     "",
187                     ""
188                         ], $obj->layout);
189                 show($obj->finalAnswer);
190             }
191         }
192
```

Fuente: Propia.

**JAVASCRIPT:** Es un lenguaje manejado de parte del cliente en donde nos permite crear efectos, animaciones a la respuesta de eventos causados por el usuario tales como botones o modificaciones del DOM.

*Figura 27: Código JavaScript.*




Fuente: Propia.

**CSS:** Estas hojas de estilo nos permiten personalizar toda la parte de estilos tales como colores, tamaño de imágenes, tamaño de letra, adaptar nuevos tipos de letra, tamaño y ubicación de objetos, o incluso poder lograr que el el sitio sea adaptable a diversos dispositivos móviles teniendo en cuenta cada tamaño de pantalla.

*Figura 28: Código CSS.*

style.css

assets > css > style.css > .item span

```
72      border-radius:0 !important;
73      transition: box-shadow 0.5s;
74  }
75  .panel:hover {
76      box-shadow: 5px 0px 40px rgba(0,0,0, .2);
77  }
78  .panel-footer .btn:hover {
79      border: 1px solid #047bb4;
80      background-color: #fff !important;
81      color: #f4511e;
82  }
83  .panel-heading {
84      color: #fff !important;
85      background-color: #047bb4 !important;
86      padding: 25px;
87      border-bottom: 1px solid transparent;
88      border-top-left-radius: 0px;
89      border-top-right-radius: 0px;
90      border-bottom-left-radius: 0px;
91      border-bottom-right-radius: 0px;
92  }
93  .panel-footer {
94      background-color: white !important;
95  }
96  .panel-footer h3 {
97      font-size: 32px;
98  }
99  .panel-footer h4 {
100     color: #aaa;
101     font-size: 14px;
102 }
103 .panel-footer .btn {
104     margin: 15px 0;
105     background-color: #f4511e;
106     color: #fff;
107 }
108 .navbar {
109     margin-bottom: 0;
110     background-color: #047bb4;
111     z-index: 9999;
112     border: 0;
113     font-size: 12px !important;
114     line-height: 1.42857143 !important;
115     letter-spacing: 4px;
116     border-radius: 0;
117     font-family: Montserrat, sans-serif;
118 }
119 .navbar li a, .navbar .navbar-brand {
120     color: #fff !important;
121 }
122 .navbar-nav li a:hover, .navbar-nav li.active a {
123     color: #047bb4 !important;
124     background-color: #fff !important;
125 }
```

Fuente: Propia.

## INTERFACES GRÁFICAS DE LA APLICACIÓN

A continuación, se comparten algunas interfaces de usuario de la aplicación. Para visualizar de manera más completa y a detalle por favor consultar el anexo Manual Pet information.

Inicio de la aplicación web.

*Figura 29: Interfaz Inicio.*


Fuente: Propia.

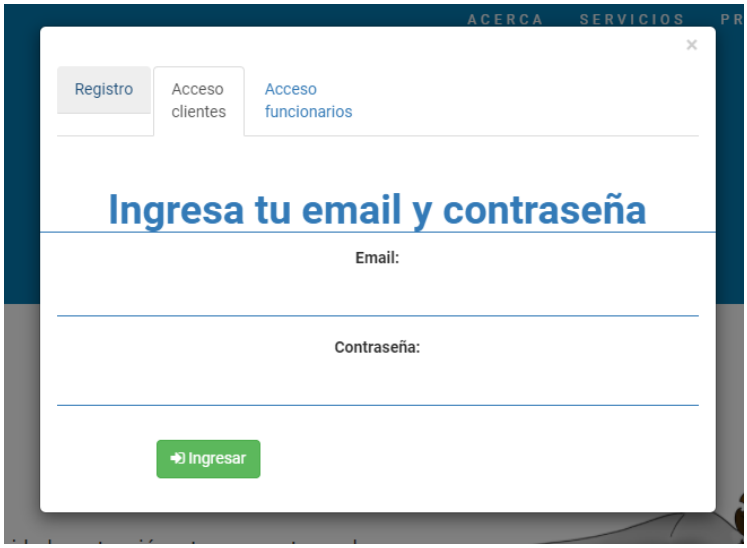
Registro de usuario

*Figura 30: Interfaz Registro de usuario.*


Fuente: Propia.

Inicio de sesión

*Figura 31: Interfaz inicio de sesión.*




Fuente: Propia.

Registro y visualización de mascotas

*Figura 32: Interfaz Registro y visualización de mascotas.*


Fuente: Propia.

Solicitar cita

*Figura 33: Interfaz Solicitar cita.*


Fuente: Propia.

Consultar citas

*Figura 34: Interfaz Consultar citas.*


Fuente: Propia.

## ESTIMACIÓN DE COSTES Y RECURSOS

En la actualidad contamos con los recursos tecnológicos y conocimientos previos para la elaboración y ejecución del proyecto, hoy en día existen herramientas como MySQL, PHP, HTML5, CCS3 las cuales son tecnologías de desarrollo para la elaboración de sistemas de información, aplicado en diferentes plataformas de páginas web.

**Recursos Tecnicos**

- ✔ Computador
- ✔ Sistema operativo Windows/ 64bits
- ✔ Procesador AMD Ryzen 3 APU/ Intel i3 o superior
- ✔ Memoria RAM 4.00 GB o superior
- ✔ Pantalla LED o LCD 24” o mayor

**Recurso Humano**

- ✔ Tecnólogos en Informática

**Recurso Financiero**

Para la realización del proyecto es importante tener en cuenta los costos fijos y variables, para darle de esta manera el valor monetario al desarrollo del software, a continuación relacionamos los costos:

*Figura 35: Recurso financiero.*

| Descripción | Costos Fijos $ | Costos Variables $ |
|---|---|---|
| Nomina | 2.500.000 | |
| Energia electrica | 75.000 | |
| Internet | 150.000 | |
| Valor del software | 2.500.000 | |
| Arriendo - Ambiente de trabajo | 160.000 | |
| Papeleria | | 20.000 |
| Total | 5.385.000 | 20.000 |
| Total fijos y variables | 5.405.000 | |
| 20% de utilidad | 1.081.000 | |
| Valor del proyecto | **6.486.000** | |

Fuente: Propia.

## PLAN QSA

El plan se desarrolló en el documento anexo B del plan de calidad del proyecto.

## MATRIZ DE RIESGOS

Ver Documento anexo C.

*Figura 36: Matriz de riesgos.*

**Convenciones**

| Probabilidad / impacto | Valor |
|---|---|
| Baja | 1 |
| Media | 3 |
| Alta | 4 |

| Nivel de riesgo | | |
|---|---|---|
| Nivel | Valor | Color |
| Muy alto | 21 - 25 | Rojo |
| Alto | 16 - 20 | Naranja |
| Medio | 10 - 15 | Amarillo |
| Bajo | 6 - 9 | Verde |
| Medio bajo | <= 5 | Azul |

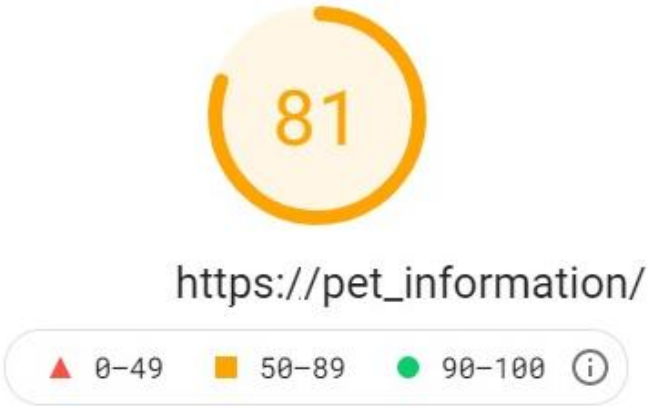
ID PROYECTO: PET INFORMATION
FECHA DE INICIO: 01/01/2021
FECHA DE FIN: 30/05/2021

| # | Nombre del riesgo | Riesgo (si) | Síntoma | Probabilidad (A/M/B) | Impacto (A/M/B) | Prioridad (1 - 12) | Estrategia prevención | Responsable de la acción de respuesta |
|---|---|---|---|---|---|---|---|---|
| 1 | La no adquisición de Hosting y Dominio | SI | Aplicación WEB no esta en linea | B | A | 25 | Explicarle al cliente los beneficios de adquirir Hosting y dominio para la aplicación | Desarrolladores |
| 2 | Cliente no requiere proyecto | SI | Cliente indica que no requiere la aplicación WEB | B | A | 22 | Explicarle al cliente la importancia de tener su aplicación WEB | Desarrolladores |
| 3 | Perdida de datos | SI | Queja de usuarios por perdida de información | M | A | 11 | Realizar Backups de la información periodicamente | Cliente |
| 4 | Diseño inadecuado | SI | Usuarios no comprenden el software, dificultando el uso | B | B | 7 | Realizar un estudio y estar a la vanguardia de las interfazes graficas | Desarrolladores |
| 5 | Problema de calidad | SI | Falla en los resultados | M | M | 10 | Ser minucioso y validar bien los requerimientos del proyecto | Desarrolladores |
| 6 | Incumplimiento del cronograma | SI | Retrasos en las entregas del software | M | B | 5 | Realizar seguimiento y cumplir con las fechas establecidas | Desarrolladores |

Fuente: Propia

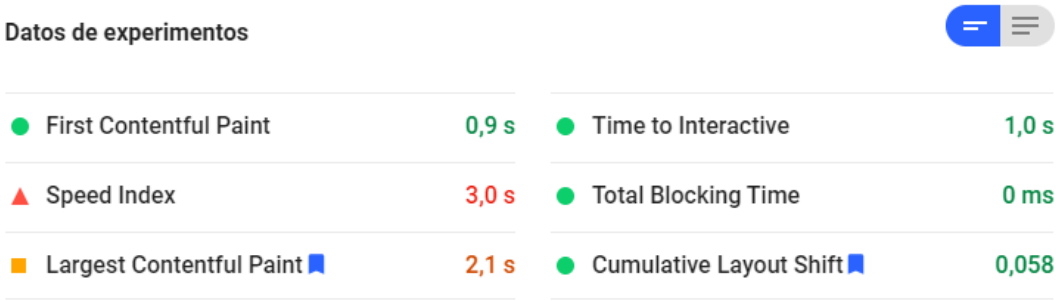
## PRUEBAS

*Figura 37: Prueba 1.*




De acuerdo a las pruebas de rendimiento realizadas por medio de la extensión de google para developers PageSpeed Insights nos arroja una puntuación de 81 sobre 100 de rendimiento el cual se encuentra basado en los siguientes criterios:

*Figura 38: Prueba 2.*




**First Contentful Paint:**

Mide cuánto tiempo le toma al navegador procesar la primera parte del contenido DOM después de que un usuario navega a su página.

El puntaje obtenido en este criterio fue de 0,9 segundos considerado rápido de acuerdo a la tabla de puntuación FCP:

| Tiempo FCP (segundos) | Código de colores | Puntuación FCP |
|---|---|---|
| 0-2 | Verde (rápido) | 75-100 |
| 2-4 | Naranja (moderado) | 50-74 |
| Más de 4 | Rojo (lento) | 0-49 |

**Time to Interactive:**

Mide el tiempo que tarda la página en volverse completamente interactiva.

El puntaje obtenido en este criterio fue de 1 segundo considerado rápido de acuerdo a la tabla de puntuación TTI:

| Tiempo FCP (segundos) | Código de colores |
|---|---|
| 0-3,8 | Verde (rápido) |
| 3,9-7,3 | Naranja (moderado) |
| Más de 7,3 | Rojo (lento) |

**Speed Index**

Mide qué tan rápido se muestra visualmente el contenido durante la carga inicial de la página

El puntaje obtenido en este criterio fue de 3.0 segundos considerado rápido de acuerdo a la tabla de puntuación:

| Tiempo FCP (segundos) | Código de colores | Puntuación FCP |
|---|---|---|
| 0-4,3 | Verde (rápido) | 75-100 |
| 44,-5,8 | Naranja (moderado) | 50-74 |
| Más de 5,8 | Rojo (lento) | 0-49 |

**Total Blocking Time**

Es una comparación del tiempo TBT de la página y los tiempos TBT para los 10.000 sitios principales cuando se carga en dispositivos móviles.

El puntaje obtenido en este criterio fue de 0 milisegundos considerado rápido de acuerdo a la tabla de puntuación TBT:

| Tiempo FCP milisegundos) | Código de colores |
|---|---|
| 0-300 | Verde (rápido) |
| 300-600 | Naranja (moderado) |
| Más de 600 | Rojo (lento) |

**Largest Contentful Paint**

Mide cuándo se presenta en la pantalla el elemento de contenido más grande de la ventana gráfica.

El puntaje obtenido en este criterio fue de 2.1 segundos considerado rápido de acuerdo a la tabla de puntuación LCP:

| Tiempo LCP(segundos) | Código de colores |
|---|---|
| 0-2.5 | Verde (rápido) |
| 2.5-4 | Naranja (moderado) |
| Más de 4 | Rojo (lento) |

**Cumulative Layout Shift**

Es una medida de la mayor ráfaga de puntuaciones de cambio de diseño para cada cambio de diseño inesperado que se produce durante toda la vida útil de la página.

El puntaje obtenido en este criterio fue de 0.058 segundos considerado rápido de acuerdo a los estándares de CLS en donde una puntuación de 0.1 o menos es bueno y entre 0.1 y 0.25 necesita mejoras.

| Tiempo CLS (segundos) | Código de colores |
|---|---|
| 0-0.1 | Verde (rápido) |
| 0.1-0.25 | Naranja (moderado) |
| Más de 0.25 | Rojo (lento) |

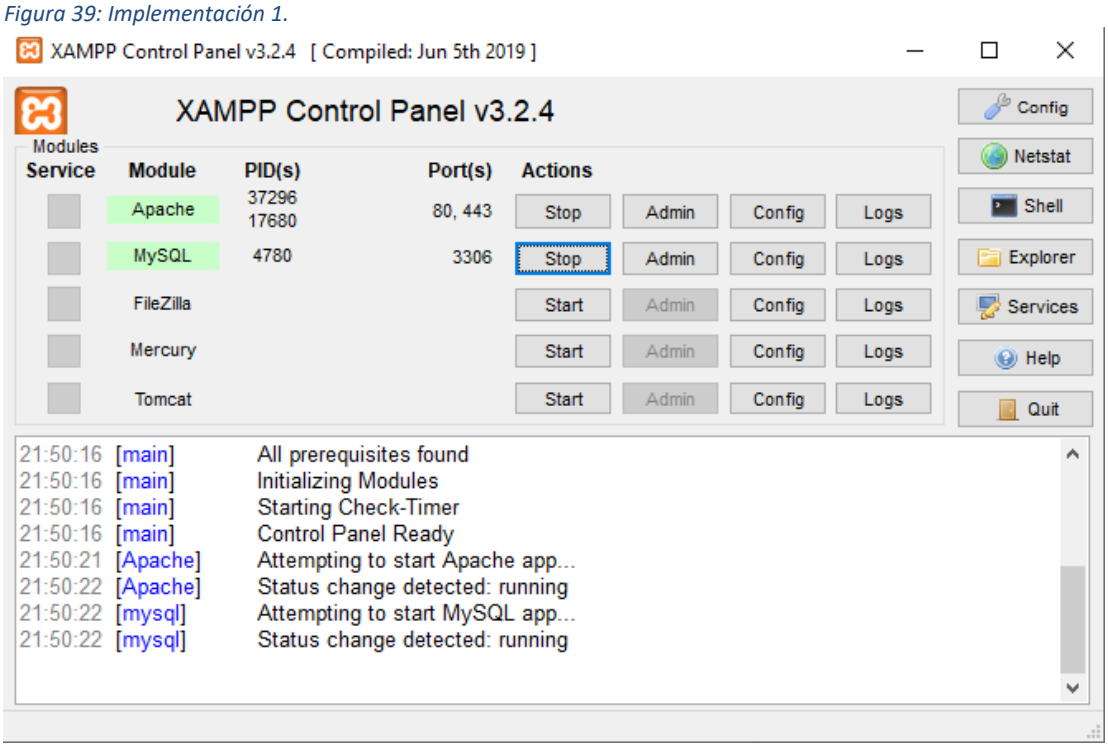
## FASE DE IMPLEMENTACIÓN

El sistema se ha implementado en primera instancia de manera local en uno de los equipos de cómputo de la veterinaria, parra fué necesario realizar el siguiente procedimiento:

1-) Descarga e instalación en el equipo de cómputo de la herramienta XAMPP el cual realizará las veces de servidor de manera local y a su vez realizará la gestión de las bases de datos.

Para que funcione correctamente la aplicación debemos entrar al panel de control de XAMPP y activar los servicios de Apache y MySQL.

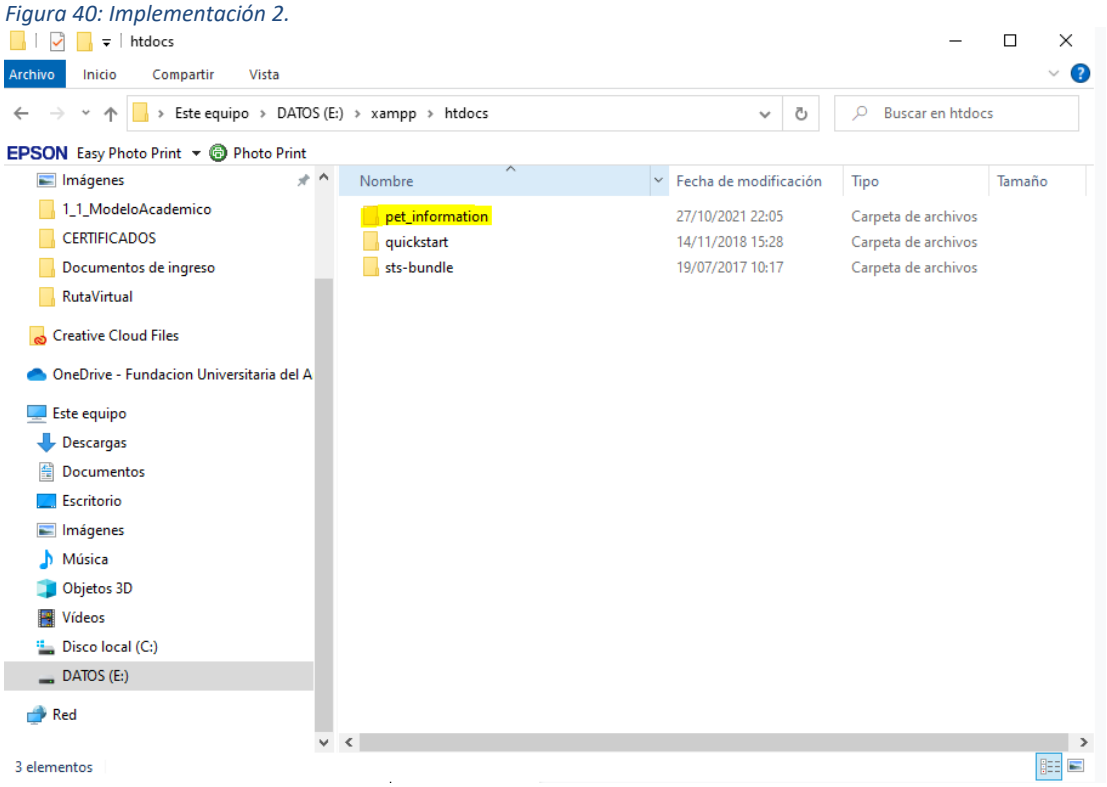
*Figura 39: Implementación 1.*




Fuente: Propia.

2-) Una vez activados los servicios vamos a la carpeta donde quedó instalado XAMPP y buscamos la carpeta htdocs en la cual es necesario trasladar toda la carpeta del código fuente de la aplicación.

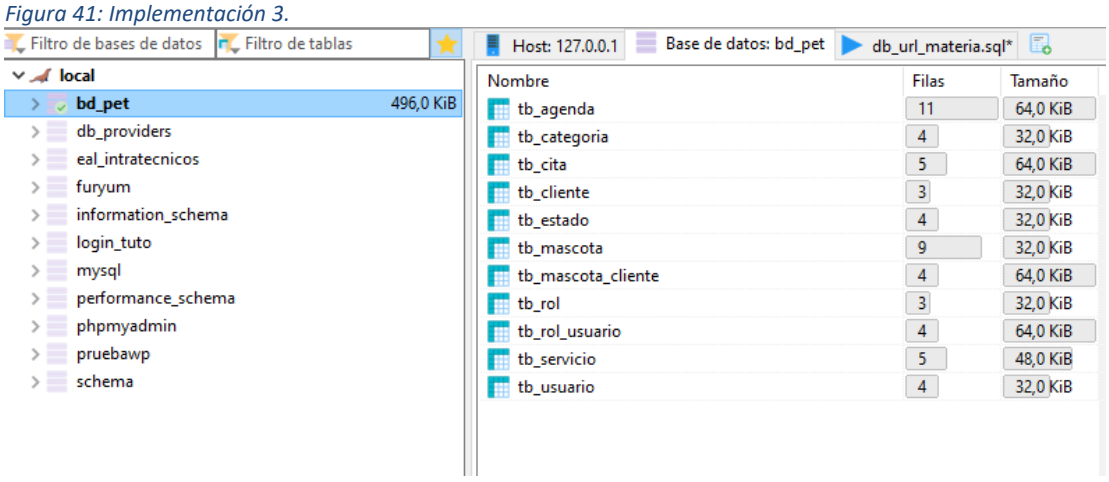
*Figura 40: Implementación 2.*




Fuente: Propia.

3-) Una vez cargado el código fuente realizaremos el cargue completo de la base de datos al nuestro servidor local.

*Figura 41: Implementación 3.*




Fuente: Propia.

4-) Al momento de tener cargado el código fuente y la base de datos con sus respectivas tablas y datos iniciales, se debe verificar en el código fuente los parámetros de conexión

a la base de datos según los definidos por el servidor local en la siguiente ruta: **pet_information/sys/application/helpers/functions_helper.php**

*Figura 42: Implementación 4.*

functions_helper.php 9+

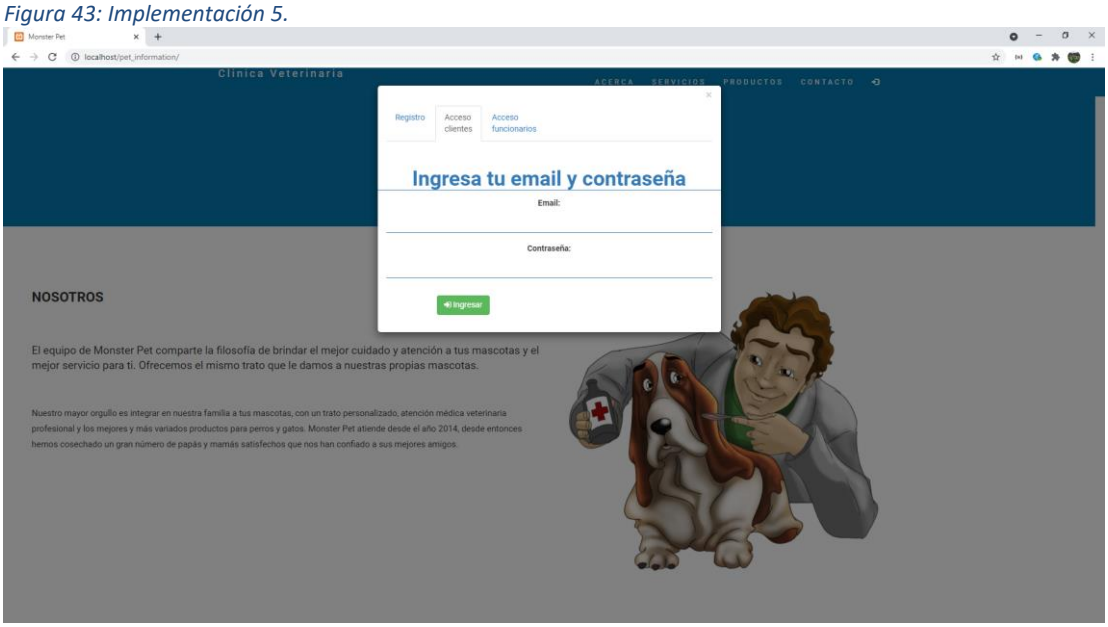
sys > application > helpers > functions_helper.php > conectionFree

```
855         return $answer;
856     }
857
858     function conectionFree($sql) {
859
860         $answer = [];
861         $servername = "localhost";
862         $username = "root";
863         $password = "";
864         $dbname = "bd_pet";
865
866         // Create connection
867         $conn = new mysqli($servername, $username, $password, $dbname);
868         // Check connection
869         if ($conn->connect_error) {
870             $answer = ["Connection failed: " . $conn->connect_error];
871             die("Connection failed: " . $conn->connect_error);
872         } else {
873             $result = $conn->query($sql);
874             if ($result->num_rows > 0) {
875
876                 while ($r = mysqli_fetch_array($result, MYSQLI_ASSOC)) {
877                     $answer[] = $r;
878                 }
879                 mysqli_free_result($result);
880             } else {
881                 $answer = [];
882             }
883             $conn->close();
884         }
885         return $answer;
886     }
887
```

Fuente: Propia.

5-) Por último ejecutamos la aplicación por medio del navegador ingresando la ruta: http://localhost/pet_information/

*Figura 43: Implementación 5.*




Fuente: Propia.

## CONCLUSIONES

De acuerdo con el trabajo realizado, como nuevos profesionales que seremos, logramos formar bases y conocimiento para el desarrollo de proyectos, en los cuales nos apoyamos bastante en la indagación y consulta bibliográfica para comprender las metodologías de desarrollo de software y todo lo relacionado con la creación del aplicativo. Pet información nació gracias a la necesidad de una veterinaria, y de acuerdo con nuestros conocimientos adquiridos en el transcurso de la carrera logramos darle forma a la solución requerida, que beneficiará específicamente a la veterinaria Monster Pet.

El desarrollo de la aplicación le permitirá a la veterinaria contar con un repositorio de datos de clientes y mascotas, también permite la gestión de citas, para llevar un control y trazabilidad de las mismas. La aplicación de igual manera permite acceder desde cualquier navegador y dispositivo.

Al implementar las tecnologías de la información en cualquier tipo de negocio como en este caso una veterinaria, se logra eliminar el uso de documentos físicos o archivos de word y excel que generaban menos control de la información.

Gracias al trabajo desarrollado en el presente proyecto se logró cumplir las necesidades de la veterinaria, logrando así una mejor gestión de la información.

## BIBLIOGRAFÍA


- Báez Pérez, C. I. y Suárez Zarabanda, M. I. (2013). Proceso de desarrollo de software: basado en la articulación de RUP y CMMI priorizando su calidad. Universidad de Boyacá. https://elibro-net.proxy.bidig.areandina.edu.co/es/lc/areandina/titulos/129062

- Beynon-Davies, P. (2014). Sistemas de bases de datos. Editorial Reverté. https://elibro-net.proxy.bidig.areandina.edu.co/es/lc/areandina/titulos/46796

- Bogota.gov.co, 20 de febrero del 2020. ¡Ojo! Estas son las sanciones y multas por maltrato animal, recuperado de https://bogota.gov.co/mi-ciudad/ambiente/sanciones-y-multas-para-el-maltrato-animal-en-colombia#:~:text=La%20Ley%201774%20de%202016,y%20multas%20por%20maltrato%20animal.&text=%E2%80%9CLos%20animales%20son%20seres%20que,6%20de%20enero%20de%202016.

- Campderrich Falgueras, B. (2013). Ingeniería del software. Editorial UOC. https://elibro-net.proxy.bidig.areandina.edu.co/es/lc/areandina/titulos/56294

- Desarrolloweb.com,Qué es la programación orientada a objetos, recuperado de https://desarrolloweb.com/articulos/499.php#:~:text=La%20programaci%C3%B3n%20Orientada%20a%20objetos%20se%20define%20como%20un%20paradigma,los%20objetivos%20de%20las%20aplicaciones.

- Joyanes Aguilar, L. (1998). Programación orientada a objetos. Madrid: Mc Graw-Hill.

- López Sanz, M. Sánchez Fúnquene, D. M. y Moreno Pérez, Á. (2015). Programación web en el entorno cliente. RA-MA Editorial. https://elibro-net.proxy.bidig.areandina.edu.co/es/lc/areandina/titulos/106486

- Luján-Mora, Sergio. (2002). Programación de aplicaciones web: historia, principios básicos y clientes web. Recuperado de: http://hdl.handle.net/10045/16995

- Moreno Pérez, J. (2015). Programación orientada a objetos. RA-MA Editorial. https://elibro-net.proxy.bidig.areandina.edu.co/es/lc/areandina/titulos/106461

- Quintana, G. (2014). Aprende SQL. Universitat Jaume I. Servei de Comunicació i Publicacions. https://elibro-net.proxy.bidig.areandina.edu.co/es/lc/areandina/titulos/53252

- Rey, Á. G. (2011). Sistemas de Información . Herramientas Prácticas Para la Gestión Empresarial. Ra-Ma.

- Rockcontent (ene 8, 20). Guía completa del Framework: qué es, cuáles tipos existen y por qué es importante en Internet. Recuperado de https://rockcontent.com/es/blog/framework/

- rupandcmmi.blogspot. Mayo 2017. CICLO DE VIDA DE RUP. Recuperado de http://rupandcmmi.blogspot.com/p/ciclo-de-vida-el-ciclo-de-vida-rup-es.html#:~:text=Fase%20de%20Inicio%3A%20Esta%20fase,y%20el%20de%20iteraciones%20posteriores.


**ANEXOS**

Anexo A. Cronograma de actividades.

Anexo B. Plan de Calidad Pet Information.

Anexo C. Matriz de Riesgos.

Anexo D. E R S. Pet information.

Anexo E. Manual Pet information.