**1. Definición de objetivos**

Mediante el desarrollo de este software se busca aportar una ayuda a las veterinarias, para este caso se enfocara en la veterinaria "Fieles Angelitos", dicha ayuda se realizara mediante el desarrollo de una aplicación web, el cual busca mejorar las ventas que realiza la empresa al ofrecer sus productos en internet, al igual que mejorar su sistema de manejo de citas al poder realizarse de manera virtual y expandirá su horizonte brindando información de la misma, el aplicativo contara con una interfaz amigable con el usuario y tiempos rápidos de carga.

* Desarrollar la base de datos que contendrá información relevante como: productos, clientes, citas.
* Esquematizar una interfaz amigable y atractiva para el usuario bajo el estándar ISO 9241-151.
* Implementar aplicación tipo API REST que integre pasarela de pagos.
* Evaluar grado de usabilidad y desempeño de la aplicación.

**2. Identificación de requisitos**

**Funcionales**

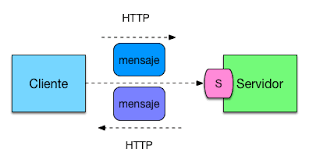
* + El sistema debe permitir a los veterinarios y al personal de la clínica registrar a nuevos pacientes.
  + Debe recopilar información como nombre, especie, raza, edad, historial médico y datos de contacto del propietario.
  + Los usuarios deben poder programar citas en línea a través del sitio web.
  + Debe haber un calendario donde los usuarios puedan ver las fechas y horas disponibles para las citas.
  + El sistema debe mantener un registro digital de cada paciente, incluyendo diagnósticos anteriores, tratamientos y medicamentos recetados.
  + Los veterinarios deben poder acceder al historial médico durante las consultas.
  + El sistema debe enviar recordatorios automáticos de citas a los propietarios de mascotas a través de correo electrónico.
  + Debe existir un módulo que permita llevar un control del inventario.
  + El sistema debe generar facturas automáticas basadas en los servicios y ventas realizadas.
  + Los propietarios deben poder realizar pagos en línea a través de métodos seguros.
  + Debe existir un módulo para gestionar los datos de los empleados, como horarios de trabajo, roles y registros de tiempo.
  + El sistema debe generar informes y estadísticas sobre la cantidad de pacientes atendidos, tipos de tratamientos más comunes, entre otros datos relevantes.
  + El módulo de empleados puede agregar, modificar y eliminar el personal que puede ingresar a manipular información.

**No funcionales**

* + El sistema debe ser intuitivo y fácil de usar para los usuarios tanto con experiencia en tecnología como sin ella.
  + La interfaz de usuario debe ser amigable y adaptarse a diferentes dispositivos, como computadoras y dispositivos móviles.
  + El sistema debe ser capaz de manejar múltiples usuarios simultáneamente sin ralentizarse ni perder funcionalidad.
  + El tiempo de carga de las páginas no debe exceder los 5 segundos.
  + Los datos de los pacientes y propietarios deben ser almacenados de forma segura para prevenir accesos no autorizados.
  + Se deben implementar medidas de autenticación y autorización para restringir el acceso a información confidencial.
  + El sistema debe estar disponible las 24 horas del día, los 7 días de la semana.
  + El sistema debe ser escalable para manejar un aumento en el número de pacientes, citas y registros a medida que la clínica crece.
  + El código del sistema debe estar bien documentado y estructurado para facilitar futuras actualizaciones y mantenimiento.
  + Las actualizaciones y correcciones de errores deben poder implementarse sin interrumpir el funcionamiento normal del sistema.
  + El sistema debe ser compatible con diferentes navegadores web modernos.
  + Debe poder integrarse con sistemas externos como sistemas de facturación.
  + El sistema debe cumplir con las regulaciones de privacidad de datos.

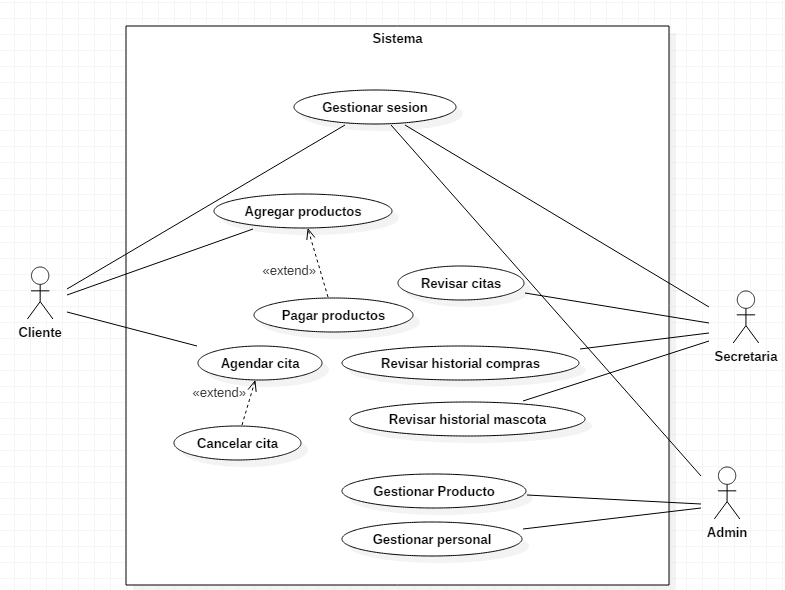
**3. Diseño de la arquitectura**

Arquitectura REST

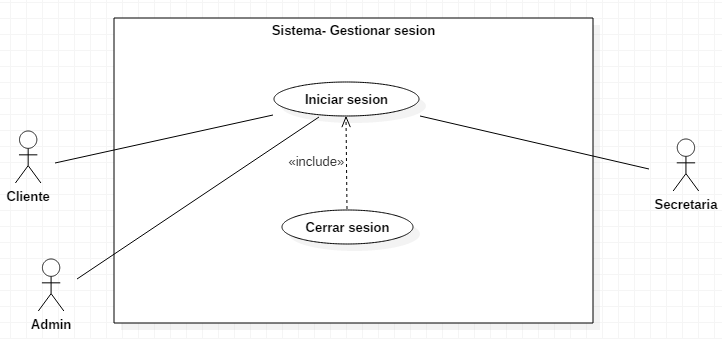
****

La arquitectura REST se basa en que el cliente envía peticiones para recuperar o modificar recursos, y el servidor responde con el resultado, que puede ser con los datos que hemos pedido o el estado de la petición. La principal ventaja de esta arquitectura es que ha aportado a la web una mayor escalabilidad, es decir, dan soporte a un mayor número de componentes y las interacciones entre ellos.

**Diagrama de caso de uso**



**Diagrama caso de uso gestionar sesión**



**Diagrama caso de uso gestionar producto**

Diagrama

Descripción generada automáticamente

**Diagrama caso de uso gestionar personal**

Diagrama

Descripción generada automáticamente

**Modelo Base de datos**

Diagrama

Descripción generada automáticamente

4. Desarrollo de prototipos

El software que se desarrollo es un prototipo.

5. Diseño de experimentos

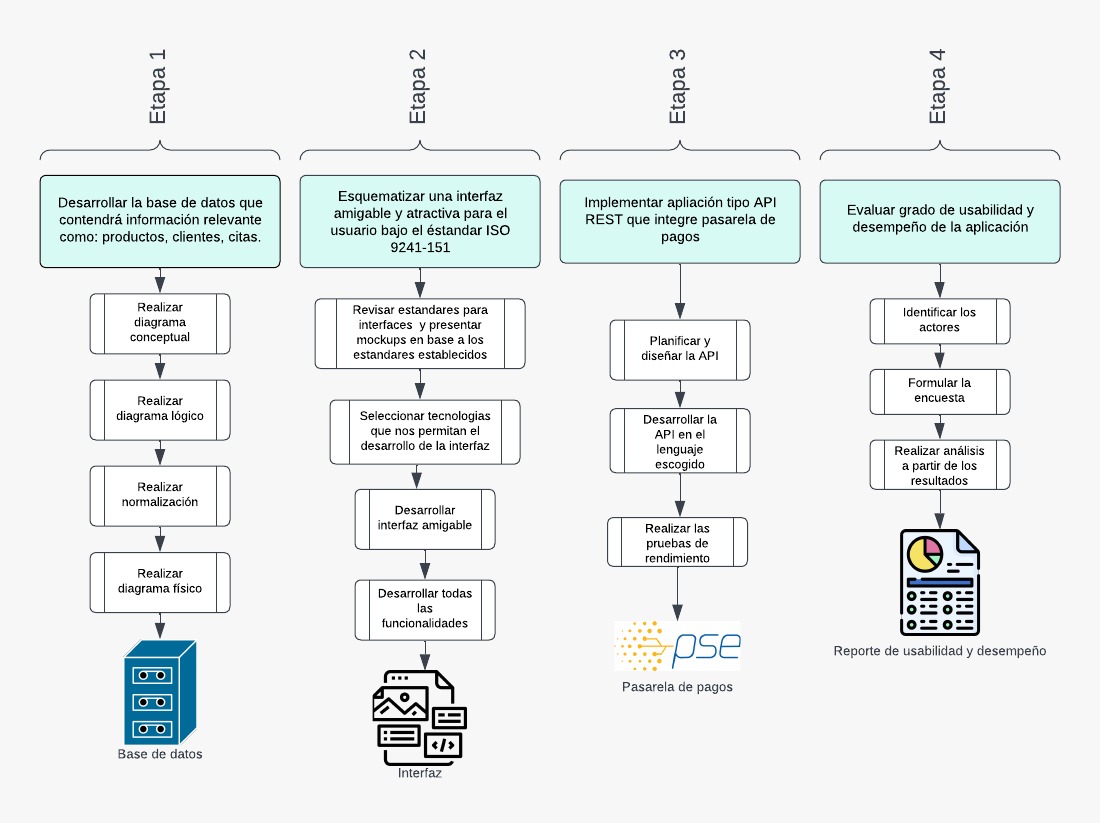
Al software se le probaran aspectos de rendimiento y usabilidad, ya que, se busca que les sea fácil a los usuarios hacer uso de la aplicación.

6. Definición de métricas

Las métricas que se manejaran serán el tiempo de respuesta a las peticiones que se generen desde el front-end al back-end, en cuanto a la usabilidad y experiencia de usuario, se realizara un reporte de usabilidad y desempeño.

7. implementación del software

El desarrollo del software se realizará siguiendo la metodología FDD, allí se dividió en etapas el desarrollo del software, en la primera etapa se desarrollo el modelo de la base de datos, para la segunda etapa se desarrollaron las interfaces de la aplicación junto con la lógica que las acompaña y en la tercera etapa se desarrolló la conexión con una pasarela de pagos.

****

**Análisis de resultados**

El análisis de resultados se dará a partir de lo obtenido por el reporte de utilidad y desempeño de la aplicación web.