**Planteamiento del problema**

Los dueños de mascotas en Teziutlán, Puebla; olvidan llevar a sus consultas en la veterinaria X a sus perros por falta de recordar cuando les toca su cita.

La salud de las mascotas siempre ha sido un tema importante para los dueños de estas, ya que los consideran parte de la familia, por lo que siempre quieren que tengan una buena salud para que den alegría a la familia.

Sin embargo, en Teziutlán, Puebla; en la veterinaria X se ha visto que en los últimos meses lo dueños de mascotas en este caso perros, se les olvidan las consultas que tienen programadas para llevar a sus mascotas al veterinario, lo que causa que muchos de los perros no tengan todas sus vacunas o que su enfermedad tarde más tiempo en curarse de lo que se tenían programado. Lo que origina que los perros enfermen frecuentemente por la falta de un chequeo constante y esto puede causar la muerte en muchos de ellos por la falta de atención por parte de sus dueños.

Se ha demostrado que la falta de atención a las fechas de las consultas hace que los perros tengan mas probabilidad de enfermarse y morir por la falta de cuidados adecuados o por el retraso en el chequeo.

Una condición necesaria para lograr la mejora de la salud de los perros es que los dueños recuerden cuando les toca consulta a estos o tengan algo donde puedan checar el control de su mascota.

Por lo anterior expuesto, se plantea diseñar una API en la cual los dueños de los perros puedan verificar cuando les toca cita o cuando fue la ultima a la que acudieron. Además de que en esta podrán ver el control que llevan sus mascotas en la veterinaria X desde las vacunas que tienen y las que les faltan.

**Justificación**

Diseñar una API que guarda los registro clínicos para el control de las consultas en los perros.

La salud de las mascotas siempre ha sido un tema importante para los dueños de estas, pues procuran que estén en excelentes condiciones, en este caso tomamos los datos de los perros para hacer esta API (interfaz de programación de aplicaciones).

Por lo consiguiente, en este proyecto se va a realizar una API, la cual menciona que: “Las API son mecanismos que permiten a dos componentes de software comunicarse entre sí, mediante un conjunto de definiciones y protocolos” (Amazon Web Services, 2022).

En este caso la finalidad de esta API es ver cuando le toca a tu mascota su próxima consulta y cuando fue la última.

Del mismo modo podremos checar las enfermedades que tiene y cuando le pusieron su última vacuna o si le falta alguna, incluso ver los datos del dueño, la relación que tiene con la mascota, y toda la información de su mascota.

 Esta API mantiene un impacto en los propietarios de los perros, ya que les sirve para llevar un control de las citas de sus mascotas y ver que vacuna le falta que le pongan, con esto llevan un control adecuado de la salud de su perro.

La importancia de nuestro proyecto reside en mejorar el control de las consultas de los perros, para esto es necesario guardar los registros clínicos a fin de que los dueños puedan checar los datos en caso de estar preocupados si les falta alguna vacuna o si tiene alguna enfermedad su mascota. Esto nos permitirá llevar un mejor control de cuando les toca cita a las mascotas.

**Objetivo**

Diseñar una API que guarde los registros clínicos para el control de las consultas de los perros que llevan a la veterinaria X durante el año 2022 en Teziutlán, Puebla.

**Diseño de la API**

Para empezar, daremos algunas definiciones de conceptos que se verán más adelante a fin de una mejor comprensión.

Script: (“es un documento que contiene instrucciones, escritas en códigos de programación”). (Significados.com, s.f.)

Atributos: (“son las características individuales que diferencian un objeto de otro y determinan su apariencia, estado u otras cualidades”). (Conceptos básicos de la Programación Orientada a Objetos, s.f.)

Nomenclatura Camello: (“es un estilo de escritura que se aplica a frases o palabras compuestas”). (CamelCase, s.f.)

Programa: (“es un conjunto de pasos lógicos escritos en un lenguaje de programación que nos permite realizar una tarea específica”). (Tecnologia Inclusiva, s.f.)

Class: (“es una plantilla para el objetivo de la creación de objetos de datos según un modelo predefinido”). (Kiko Palomares, s.f.)

Fields: (“es una variable de cualquier tipo que se declara directamente en una clase o struct”). (Bill Wagner, 2022)

Rest\_framework: (“es un framework que nos permite el fácil desarrollo de una API REST en Python”). (Django API REST, s.f.)

Django.urls: (“es un mapeo entre los patrones URL y las funciones de vista que deben ser llamadas por esos patrones URL”). (uniwebsidad, s.f.)

Una vez conociendo estas definiciones podemos avanzar a nuestro primer paso para crear la API.

Para crear esta API primero se elige un tema de interés, se busca una base de datos relacionada con el tema que se eligió, se ven los elementos que tiene que llevar para poder escribir el script.

Se elige un lenguaje con el cual vamos a escribir el script, en este caso tomamos el lenguaje de Python para anotar las órdenes que queremos que salgan en el proyecto y elegimos el programa de Pycharm el cual soporta este lenguaje y creamos un proyecto llamado veterinaria.

Empezando por crear ciertas carpetas las cuales van a contener el script de lo que observamos en la base de datos de la cual nos estamos guiando.

Primero se crea una carpeta la cual llevara el nombre de vet y va a contener los documentos, el primer documento que se encuentra en ella lleva por nombre models.py donde dentro de este vamos a crear las clases en las cuales vamos a meter todos los atributos que tenía la base de datos en la cual nos vamos guiando.

Empezando por la clase persona y sus atributos que pusimos fueron nombrepersona, apellidopersona, teléfono, dirección, email en estos se ponen de qué tipo de texto son si son charfiel o dataFiel, terminando esta primera clase se sigue con la segunda clase que es ClientesFamilia en la cual también se ponen sus atributos que eran PrimerApellido, CuentaBanco y teléfono y al igual que la primera especifique el tipo de texto que son, después hice la tercera clase que es la RelacionPersonasClientes y sus atributos que son IDClientesFamilia y IDPersona los cuales están relacionados con las clases anteriores lo que hace que esta solo traiga los atributos de los registros anteriores, por eso se ocupa un ForeignKey que es un tipo de llave que hace lo antes mencionado.

Luego creamos otra clase llamada PacientesMascotas en la cual es la descripción de la mascota que llevan a la veterinaria y sus atributos fueron IDClientesFamilia la cual está relacionada con una clase anterior con un ForeignKey, un AliasMascota, Especie, Raza, ColorPelo, FechaNacimiento y Vacunaciones. Posteriormente creamos otra clase llamada Vacunas la cual tiene los atributos IDPacientesMascotas, Fecha, Enfermedad y FechaProxima la cual se refiere a la próxima consulta de la mascotas que está registrada. Estos atributos están escritos en nomenclatura camello.

Después se crea otra carpeta con el nombre de admin.py en la cual vamos a poner todos los registros que pusimos en nuestra carpeta anterior tomando los nombres de las clases para regístralas en Admin y se coloca admin.site.register (Persona) cambiando Persona por el nombre de cada clase que hicimos.

Posteriormente creamos otra carpeta con el nombre serializers.py en la cual pusimos las clases que creamos antes, pero con el nombre de PersonaSerializer cambiando Persona por el nombre de cada clase, en esta creamos otra clase dentro de esta llamada class Meta en la cual se pone un fields y se pone todos los atributos que pusimos en models con comillas simples y las colocamos en forma de lista, asi con todos nuestros modelos.

Nuestro siguiente documento es views.py dentro de estas colocamos nuestras clases, pero antes le anexamos la palabra list o Detail y ponemos queryset y serializer\_class.

Posteriormente se crea otra carpeta llamada urls.py en la cual importamos rest\_framework, django.urls y vet.views, dentro de esta ponemos el nombre de nuestras clases más aparte lo que colocamos en la carpeta vistas.

Por consiguiente, abrimos la terminal y ejecutamos ciertos comando para que nos arrojara nuestra url de nuestra api. Para poder ingresar copiamos la url y la pegamos en nuestro buscador poniendo una diagonal seguido de la palabra vet diagonal el nombre de la clase que deseamos abrir.

El siguiente paso fue buscar Pythonanywhere y creamos una cuenta que tenía que ver con nuestra API, pero primero creamos un correo relacionado con nuestra API en nuestro caso de veterinaria, para que todo fuera relacionado con esta, con este correo nos creamos una cuenta con Pythonanywhere ocupando ese correo y ponemos una contraseña.

Una vez teniendo una cuenta creamos los archivos como los teníamos en Pycharm y copiamos los archivos y los pegamos en los documentos, pero ahora en Pythonanywhere, después nos vamos al apartado donde dice web apps que es donde está el url de la api para saber si ya esta subida.

Ahora nos dirigimos a cmd de la computadora donde pondremos el comando de ionic start el cual nos ayudara a crear el proyecto, eligiendo una plantilla en blanco con Angular y colocando el nombre del proyecto en nuestro caso lo llamamos AngularProject-master.

Una vez teniendo el proyecto lo abrimos en la aplicación de WebStorm y empezamos a crear carpetas que necesitaremos empezando por una llamada app la cual dentro de esta se crean más carpetas.

Adentro de la carpeta app se crea otra carpeta que llevara el nombre de pages y en esta carpeta se meterán las demás carpetas llamadas ClientesFamilia, PacientesMascotas, Persona, Pesos, RelacionPersonasClientes, Vacunas.

Todas estas carpetas antes mencionadas tendrán ciertos archivos los cuales vamos a editar de cada. Lo más importante es editar el archivo con terminación HTML en el cual pondremos los atributos que habíamos hecho anteriormente con Pycharm. Asi editando cada carpeta y poniendo en cada una sus diferentes atributos.

Una vez terminado de modificar las carpetas buscamos otra carpeta en el proyecto llamada data.service.ts en esta agregamos los link que obtuvimos en Pythonanywhere para poder obtener la información que registramos en este. Luego buscamos otra carpeta llamada pages.module.ts en donde importaremos todas las clases que pusimos en las otras carpeta.

Seguidamente nos vamos a otra carpeta llamada app-routing.module.ts en la cual vamos a agregar cada clase con una path el nombre de la clase y abajo component el nombre de la clase seguido de Component, asi con cada clase.

Posteriormente dentro de la carpeta de app se crea otra carpeta llamada components y adentro de esta se crea otra carpeta llamada menú la cual contendrá vario archivos los cuales solo vamos a modificar menu.components.ts en donde pondremos name: el nombre de nuestra clase y debajo de esta path diagonal otra vez el nombre de nuestra clase, asi con todas las clases para que queden registradas.

Guardamos todos los cambios que hicimos y abrimos la terminal donde pondremos el comando de ng serve el cual nos dará un link si todo está correcto y lo pegamos en nuestro buscador y nos tendrá que mostrar los registros que tenemos guardados de la API veterinaria.

**Implementación de la API**

Esta API se encontrará disponible en la Play Store la descarga será gratuitita

**Resultados de la API**

Esta API tuvo como resultados una buena difusión, ya que los dueños de los perros se la recomendaron a sus conocidos, puesto que esta los ayudo a recordar las citas que tenían sus mascotas en la veterinaria X, solucionando el problema de que los dueños no recordaban las consultas de sus mascotas, lo cual muchos de ellos dijeron que fue muy útil porque en cualquier momento podían ver cuando les tocaban las citas a sus mascotas.

**Evaluación de la calidad de la API**

Esta API va dirigida a los clientes de la veterinaria X que radican en Teziutlán, Puebla que tengan como mascotas a perros.

**Bibliografías**

Amazon Web Services. (s.f.). ¿Qué es una API? <https://aws.amazon.com/es/what-is/api/>

"Script". En: Significados.com. Disponible en: <https://www.significados.com/script/>