18/12/2023

Sistema de seguimiento y recomendación de plantas

Universidad Autónoma de Sinaloa

Facultad de Ingeniería Mochis

Integrantes:

- Sandoval Lizárraga Carlos Fernando.
- García Cinco Axel.
- Ruiz Elizalde Yilma Lizbeth.
- Sánchez López Axel Osbaldo.

Maestra: Rocío Jacqueline Becerra Urquidez

Materia: Inteligencia Artificial.

Grupo: 503-Software.

Índice

Índice de Figuras

Introducción

En este documento se presentarán aspectos generales de nuestro proyecto haciendo referencia a los siguientes puntos:

- Descripción del proyecto: donde presentaremos de manera general la idea principal del proyecto a realizar, así como las acciones que se realizará para cubrir ciertas necesidades.
- Descripción del problema: en este punto se centrará y se explicará el por qué se realizará este proyecto
- Justificación: en este apartado vamos a especificar los puntos a los que nos lleva a realizar este proyecto.
- Explicación de la solución: se explicará de manera general a como se llegó a la solución del problema y que herramientas se utilizaron.

Dichos puntos nos ayudaran a dar solución a nuestro proyecto y dar a conocer el alcance que tendrá al momento de realizarse, en dicho documento se explicara cómo funciona la aplicación y sobre las herramientas se utilizaron para llegar a la resolución del proyecto.

Descripción General del Proyecto

Las plantas nos proporcionan alimentos, medicinas, madera, combustible y fibras. Además, brindan cobijo a multitud de otros seres vivos, producen el oxígeno que respiramos, mantienen el suelo, regulan la humedad y contribuyen a la estabilidad del clima. Hace muchos años la población debía conocer para qué servía cada vegetal, árbol o arbusto que crecía en su entorno. Hoy en día estas costumbres se han perdido y la mayoría de nosotros no sabemos en qué nos puede servir una planta en el frente o patio de nuestras casas.

Creemos que es importante que las personas reconozcan los usos que se les puede dar a las plantas que tenemos en la ciudad, ya sea las que nos sirven para uso medicinal, para comidas, o incluso saber las que pueden ser peligrosas en algunos casos. Creemos que este proyecto ayudará a incrementar el interés hacia las plantas que nacen en la ciudad y de igual manera a hacer conciencia sobre el debido cuidado que debemos darles.

El producto de este proyecto es un sistema que permite reconocer una planta y dar las características de esta, esto se hará con una aplicación móvil que por medio de la cámara o una imagen nos reconocerá una planta y nos dará sus características, por ejemplo: nombre, distancia a la que se encuentra de nosotros, familia, género, si es tóxica, comestible, etc.

En la interfaz principal se mostrará un mapa con los registros de plantas de los cuales por medio del GPS integrado al celular y el giroscopio describirá las plantas que están alrededor del usuario, notificando al usuario por medio de texto a voz qué plantas están cerca del usuario y qué plantas le podrían interesar, haciendo así un recorrido para que el usuario pueda ver las plantas ya registradas.

El sistema registrará el recorrido que hizo el usuario y el tiempo empleado.

Especificación REAS

Agente	Rendimiento	Entorno	Actuadores	Sensores
Sistema de Seguimiento y recomendación de plantas.	• Amigable • Preciso • Rápido	• Personas • Plantas • Camino	• Interfaz Gráfica, • Altavoz, • Reporteador	• GPS, • Cámara • Giroscopio • Micrófono

Especificaciones de entorno de trabajo

Entornos de trabajo	Observable (total o parcial)	Estocástico determinista	Episódico o secuencial	Estático o dinámico	Discreto o continuo	Tipo de agente
Sistema de Seguimiento y recomendación de plantas.	Totalmente.	Determinista.	Secuencial.	Estático.	Discreto	Agente individual.

Definición de la estructura del agente

El sistema de seguimiento y recomendación de plantas se comprende de una aplicación móvil para que los usuarios puedan utilizar fácilmente en sus teléfonos celulares, está aplicación servirá para llevar control del recorrido hecho, así como brindar información sobre las plantas que localice.

Se tiene también una aplicación web que es de uso exclusivo para el administrador, donde se podrán realizar las actividades de gestión general del sistema. Además, este módulo para el administrador incluirá un reporteador que mostrará las estadísticas de más importancia con respecto al uso y al seguimiento del sistema.

El agente cuenta con una Base de conocimiento en Prolog que junto con los datos obtenidos se dedica a la toma de decisiones para el funcionamiento del sistema.

Como medio de comunicación entre los diferentes componentes del agente tenemos otra que es una API REST, para que las aplicaciones tanto como web y móvil puedan consultar a la base de datos del sistema y a la base de conocimiento de prolog.

Especificación de los componentes del problema

Problema: Sistema de seguimiento y recomendación de plantas

Estados: Registrar y recomendar plantas

Estado inicial: El sistema precarga plantas en caso de que este registrada si no se precarga del mapa vacío

Función sucesora: registrar, ver el mapa, puede consultar plantas por medio de la voz, puede consultar recorridos por filtros de fecha

Test objetivo: El sistema corrobora por medio de la foto, si es planta y nos busca información sobre la planta

Costo del camino: cada costo es individual, cada persona puede registrar diferentes plantas en la aplicación.

Especificación de Recursos. Explicación detallada y gráfica de cada recurso utilizado (cantidad, costos, calidad, tiempo etc.)

Humanos

Para la intervención en este proyecto se formó un equipo de 4 personas, estudiantes de la carrera de Lic. en Ingeniería de Software de quinto año.

Como guía para el desarrollo del proyecto se contó con el apoyo de la profesora de la asignatura de Inteligencia Artificial, Rocío Jacqueline Becerra Urquidez.

De Hardware

Teléfono celular para instalación y pruebas de la aplicación móvil con sistema operativo Android o IOS.

Cámara para el reconocimiento de las plantas que enfoque el usuario. (será utilizar la misma que tiene integrada el teléfono)

Micrófono que se usa para identificar las instrucciones que el usuario da con la voz.

Altavoz, el agente usará este dispositivo para brindar información al usuario en forma de audio.

Sistema de Posicionamiento Global (GPS), el cual servirá para definir las ubicaciones de cada una de las plantas registradas, así como para saber en todo momento la ubicación del usuario mientras está en uso la aplicación.

De software

MySQL

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base de datos

Se utilizará para almacenar la información del sistema como los usuarios, puntos de las plantas, información de las plantas etc.

Ilustración 1 Imagen de MySQL

Swi-Prolog 9.0.4

SWI-Prolog es una implementación en código abierto del lenguaje de programación Prolog. Su autor principal es Jan Wielemaker. En desarrollo ininterrumpido desde 1987, SWI-Prolog posee un rico conjunto de características, bibliotecas, herramientas y una documentación extensiva

Se utilizará para las consultas inteligentes de la App, como saber que plantas o zonas son las más visitadas de mi región entre otras consultas.

Ilustración 2 Logo de SWI-Prolog

Flutter 3.13.6

Flutter es un SDK de código fuente abierto de desarrollo de aplicaciones móviles creado por Google. Suele usarse para desarrollar interfaces de usuario para aplicaciones en Android, iOS y Web, así como método primario para crear aplicaciones para Google Fuchsia

Se utilizará para desarrollar la aplicación móvil para los recorridos de plantas.

Ilustración 3 Logo de Flutter

Laravel 10.2.8

Laravel es un framework de código abierto para desarrollar aplicaciones y servicios web con PHP 5, PHP 7 y PHP 8. Su filosofía es desarrollar código PHP de forma elegante y simple, evitando el "código espagueti". Fue creado en 2011 y tiene una gran influencia de frameworks como Ruby on Rails, Sinatra y ASP.NET MVC.

Se utilizará para desarrollar la API de la App Móvil y Web, la cual contendrá servicios del sistema y consumirá a Apis de tercero como Trefle y Plantsnet que serán de mucha utilidad para el reconocimiento e información de las plantas.

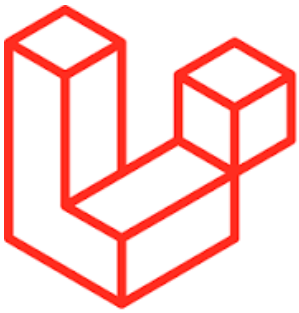


Ilustración 4 Logo de Laravel

Angular V16

Angular es un framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página.

Se utilizará para desarrollar el administrador de la aplicación, la cual contendrá un dashboard del mapa con los puntos de las plantas registradas y sus respectivos reportes

Ilustración 5 Logo de Angular

Otros

Para el desarrollo del proyecto se utilizaron varios servicios los cuales fueron muy importantes para el reconocimiento de plantas y sobre su información.

Trefle

Es un sitio web que ofrece servicios gratuitos de información sobre plantas, géneros, etc.

Ilustración 6 Logo de Trefle

Pl@ntsNet

Pl@ntNet es una herramienta para ayudar a identificar plantas a través de fotografías está organizado en diferentes temáticas y floras geográficas.

Ilustración 7 Logo de Pl@ntsNet

Desarrollo

La aplicación cuenta con 4 módulos:

Administrador Web

El administrador web cuenta con las siguientes funcionalidades

- Gestión de Usuarios
- Gestión de Plantas
- Gestión de Comandos de voz.
- Configuración de la Aplicación.
- Configuración del Usuario
- Inicio de Sesión
- Recuperación de Cuenta

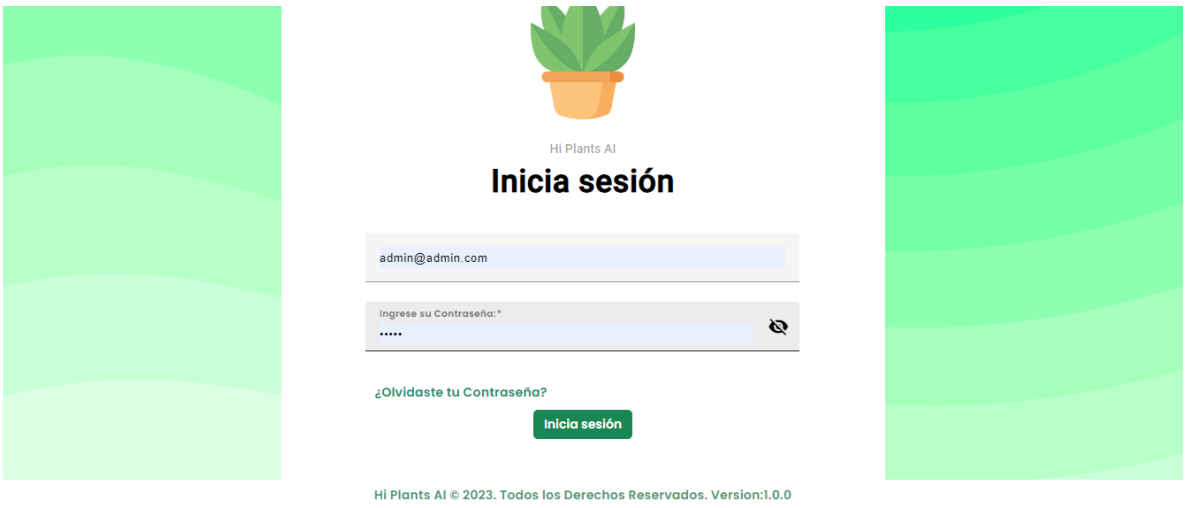


Ilustración 8 Iniciar Sesión Web

Ilustración 9 Recuperar Cuenta Web.

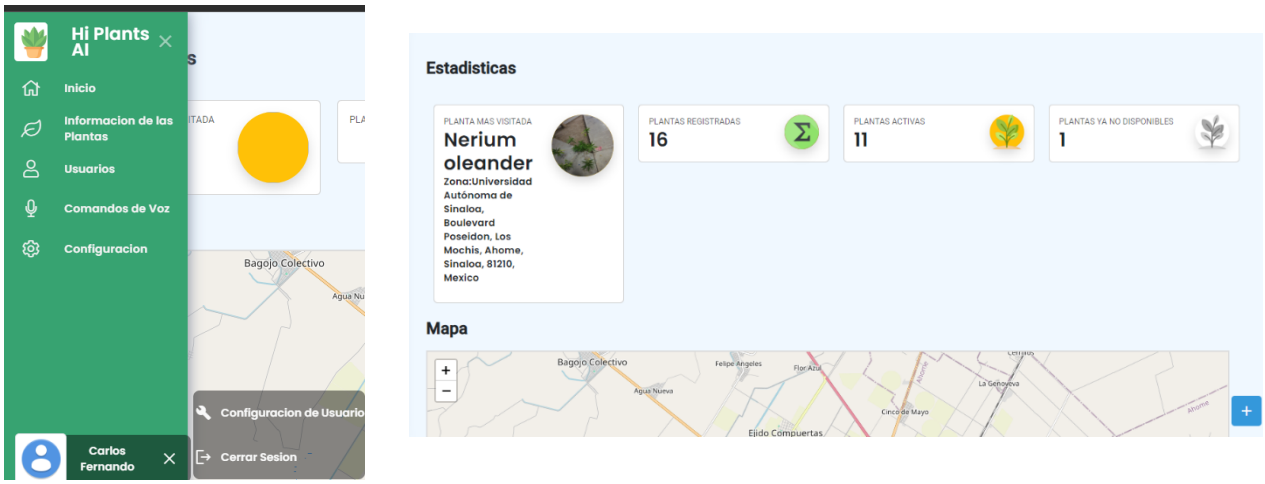


Ilustración 10 Barra Lateral con Estadísticas

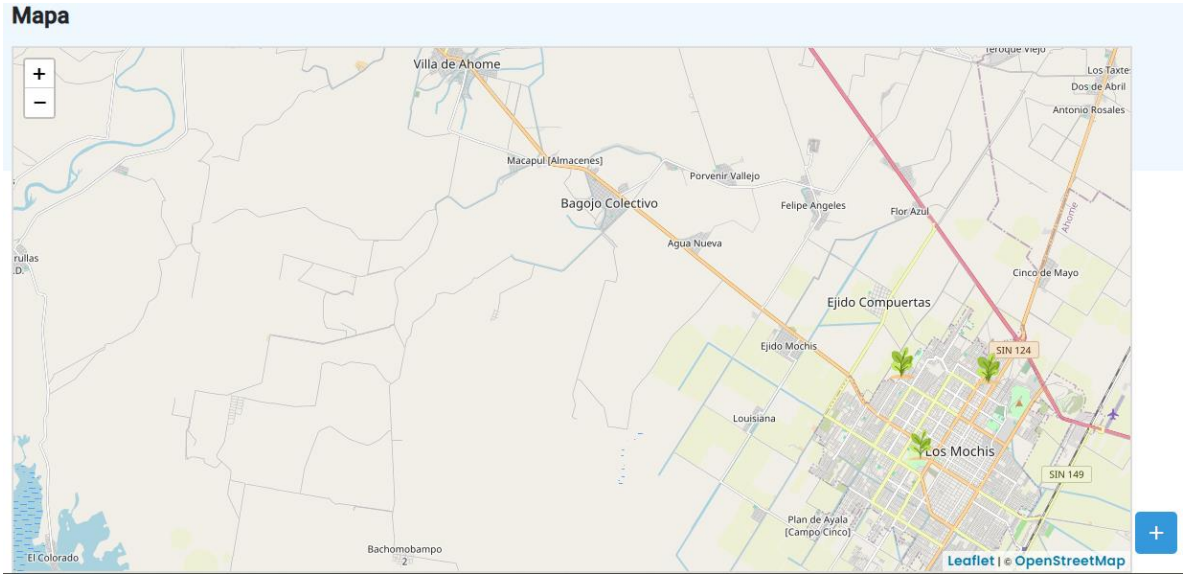


Ilustración 11 Mapa con Plantas Registradas Web

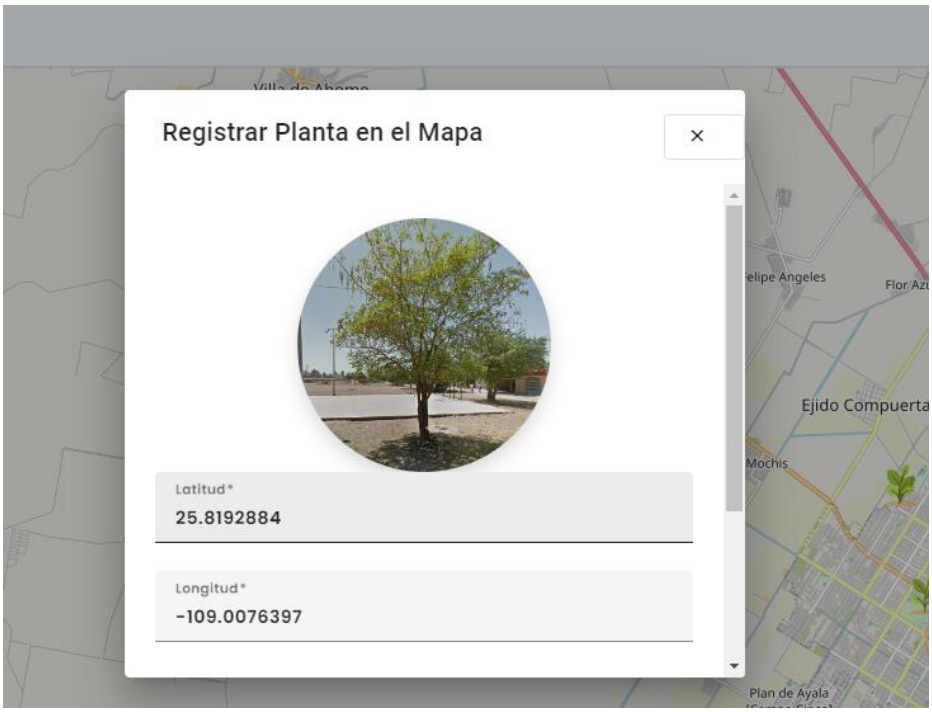


Ilustración 12 Registrar Planta en el Sistema

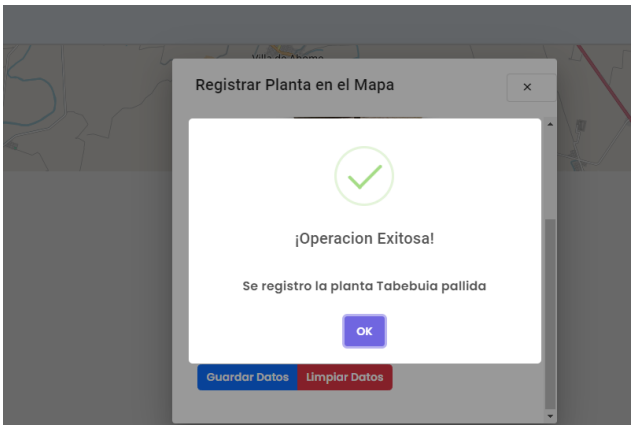


Ilustración 13 Registrando Planta Web

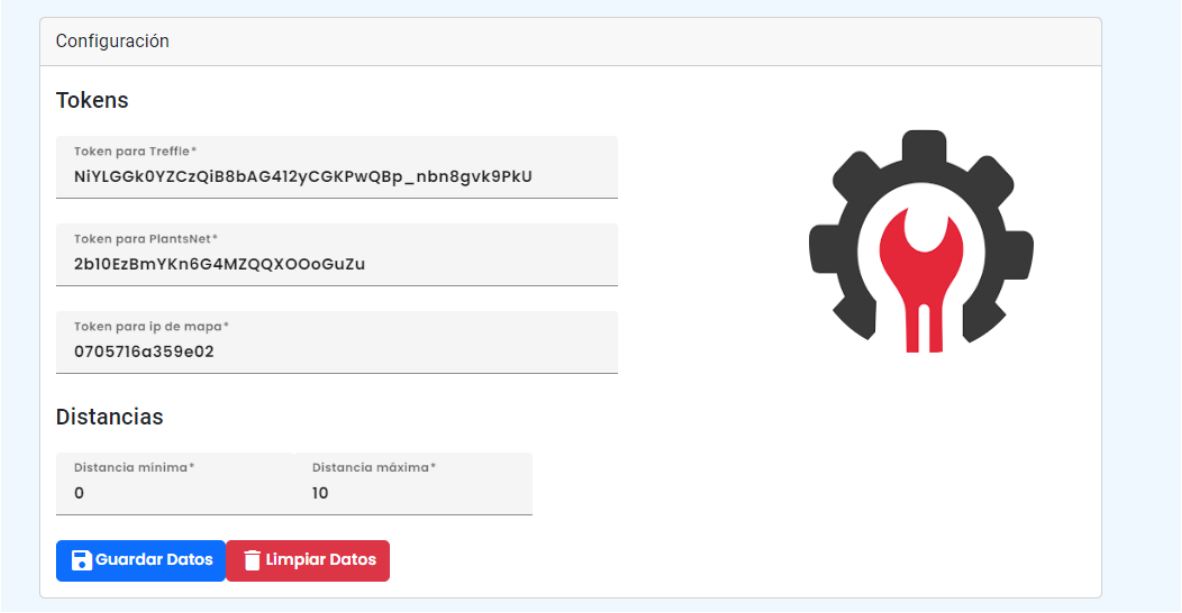


Ilustración 14 Configuración de Aplicación Web

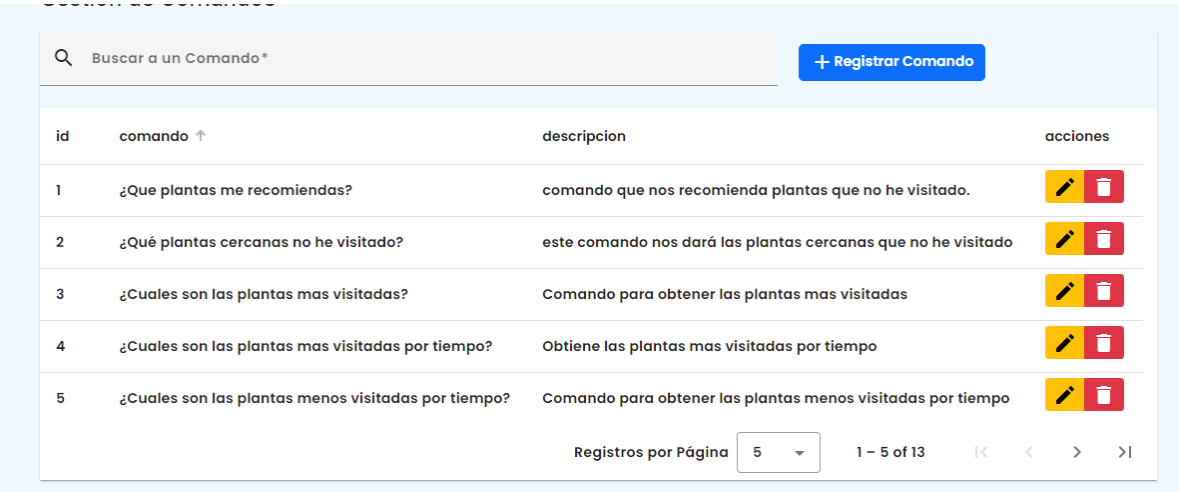


Ilustración 15 Comandos de Voz Disponibles para App Movil

App Móvil

La aplicación móvil cuenta con varias funcionalidades importantes como:

- Registrarse
- Registrar Plantas
- Ver las Plantas Registradas de acuerdo con
- la posición del usuario
- Comandos de voz.
- Recorridos que el usuario ha hecho a las plantas
- Configuración del Usuario

Ilustración 16 Iniciar Sesión Móvil.

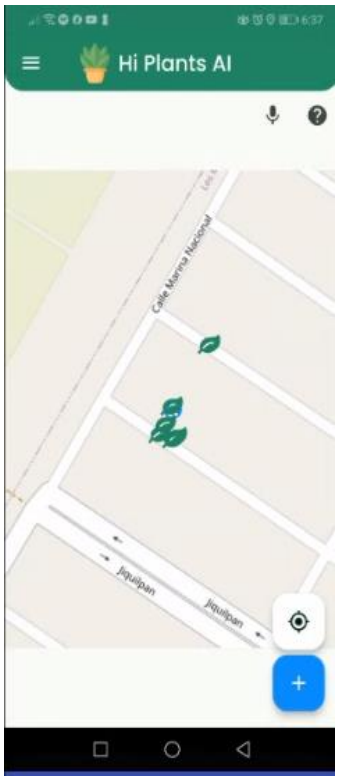


Ilustración 17 Mapa con Plantas Móvil

Ilustración 18 Barra Lateral

Ilustración 19 Ajustes del Usuario

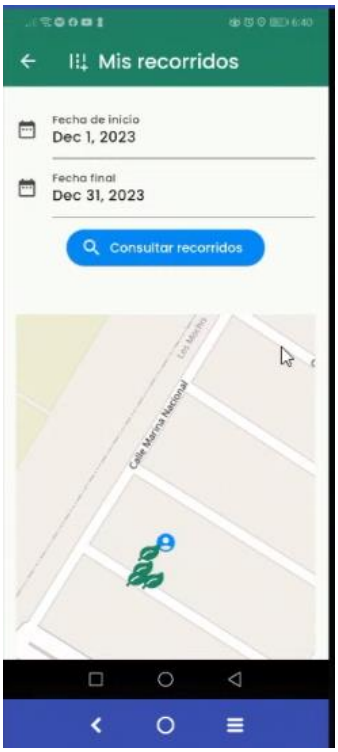


Ilustración 20 Recorridos

App móvil 4 configu

Ilustración 21 Información de Planta Recorrida

Ilustración 22 Registro de Plantas

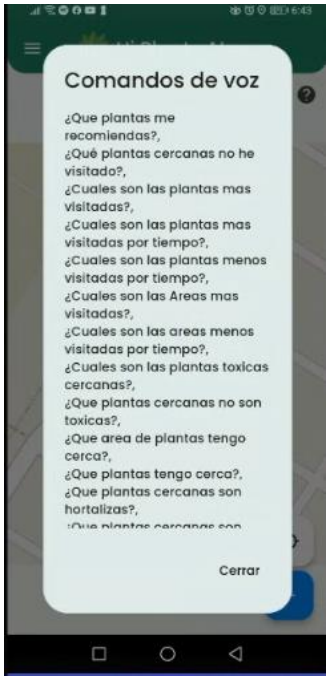


Ilustración 23 Comandos de Voz

API Rest

La API Rest Cuenta con los siguientes servicios

```php
public function IdentificarPlantaConImagen($imagen)
    {
        $token = Configuracion::find(1);

        $PROJECT = "all";
        $API_URL = 'https://my-api.plantnet.org/v2/identify/' . $PROJECT . '?api-key=';
        $API_PRIVATE_KEY = $token->tokenplantsnet;
        $API_SIMSEARCH_OPTION = '&include-related-images=true';
        $API_LANG = '&lang=es';

        $client = new GuzzleHttp\Client();
        $apiRequest = $client->request(
            'POST',
            $API_URL . $API_PRIVATE_KEY . $API_SIMSEARCH_OPTION . $API_LANG,
            [
                'multipart' => [
                    [
                        'name'     => 'images',
                        'contents' => fopen($imagen, 'r')
                    ],
                ]
            ]
        );
        $response = json_decode($apiRequest->getBody());

        return $response;
    }
```

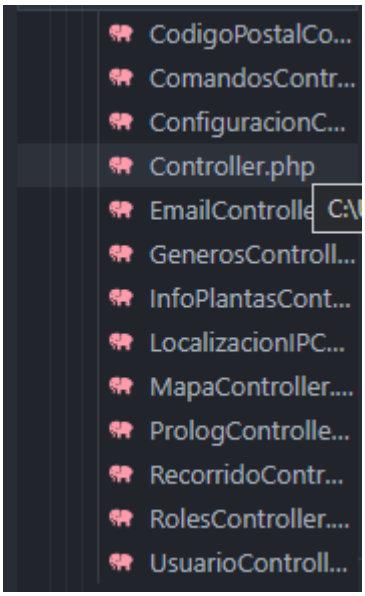


Ilustración 24 Controladores de la API

Ilustración 25 Servicio de Identificar Planta con Imagen con PlantNet

```php
public function ObtenerInformacionEspecie($especie)
    {
        $token = Configuracion::find(1);

        $sanitizeEspecie = urlencode($especie);

        $response = Http::withHeaders([
            'Authorization' => $token->tokentreffle
        ])->get(env("TreffleURL") . "/v1/species/$sanitizeEspecie");

        return json_decode($response, true);
    }
```

Ilustración 26 Obtener Información de la Especie de la planta identificada

Web Service en Prolog con Base de conocimiento.

El web service contara con varios servicios que consumirán a la base de conocimiento

```
:- http_handler('/plantasNoVisitadas', buscar_planta_handler, [method(get)]).
:- http_handler('/plantasNoVisitadas/Cercanas', buscar_planta_cercana_handler, [method(get)]).
:-http_handler('/plantaMasVisitada',buscar_planta_mas_visitada_handler,[method(get)]).
:-http_handler('/plantaMasVisitadaTiempo',buscar_planta_mas_visitada_tiempo_handler,[method(get)]).
:-http_handler('/plantaMenosVisitadaTiempo',buscar_planta_menos_visitada_handler,[method(get)]).
:-http_handler('/plantasCercanas',buscar_plantas_cercanas_handler,[method(get)]).
:-http_handler('/plantasCercanasToxicas', buscar_plantas_cercanas_toxicas_handler,[method(get)]).
:-http_handler('/plantasCercanasNoToxicas', buscar_plantas_cercanas_no_toxicas_handler,[method(get)]).
:-http_handler('/plantasCercanasComestibles', buscar_plantas_cercanas_comestibles_handler,[method(get)]).
:-http_handler('/plantasVegetables', buscar_plantas_vegetables_handler,[method(get)]).
:-http_handler('/plantasVegetablesCercanas',buscar_plantas_cercanas_vegetables_handler,[method(get)]).
```

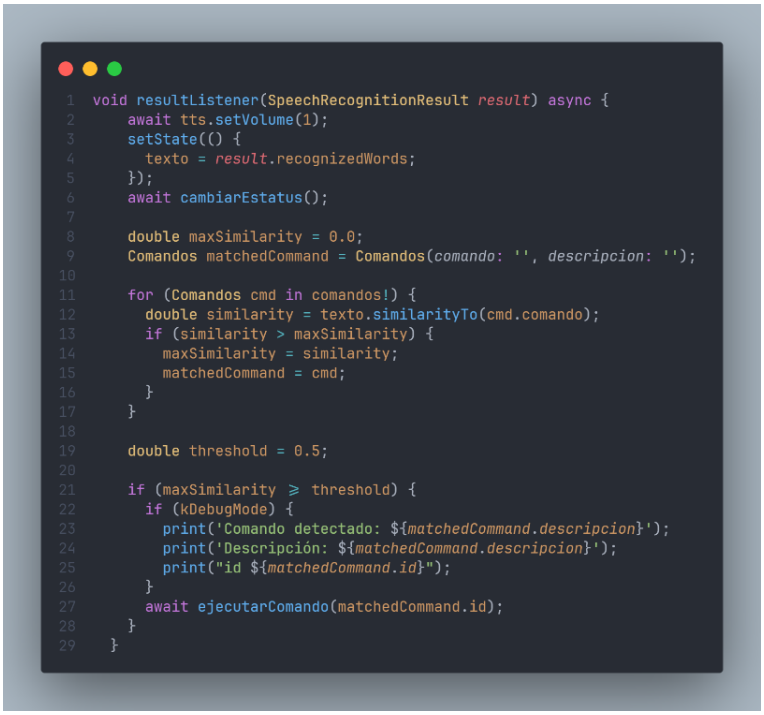
Ilustración 27 Endpoints del Web Service de Prolog

Comunicando Prolog con Laravel

```
public function ObtenerPlantasNoVisitadasCercanas($text = false, $Lat, $Long)
    {
        $id = auth()->user()->id;
        $Lat = urlencode($Lat);
        $Long = urlencode($Long);
        $id = urlencode($id);
        $url = "{$this->prologURL}/plantasNoVisitadas/Cercanas?lat=$Lat&long=$Long&idusuario=$id";
        $response = json_decode(Http::get($url));
        if (boolval($text) == 0)
            return response()->json(Message::success($response->resultado));
        else {
            $mensaje = "¡No se encontraron plantas no visitadas cercanas!";

            $numeroPlantas = count($response->resultado);
            $nombres = implode(', ', array_column($response->resultado, 2));

            if ($numeroPlantas > 0)
                $mensaje = "Se " . ($numeroPlantas > 1
                    ? 'encontraron'
                    : 'encontro'
                    )
                    . " " . $numeroPlantas . " " .
                    ($numeroPlantas > 1
                            ? 'Plantas no visitadas cercanas, '
                            : 'Planta no visitada cercana,')
                    . $nombres;

            return response()->json(Message::success($mensaje));
        }
    }
```

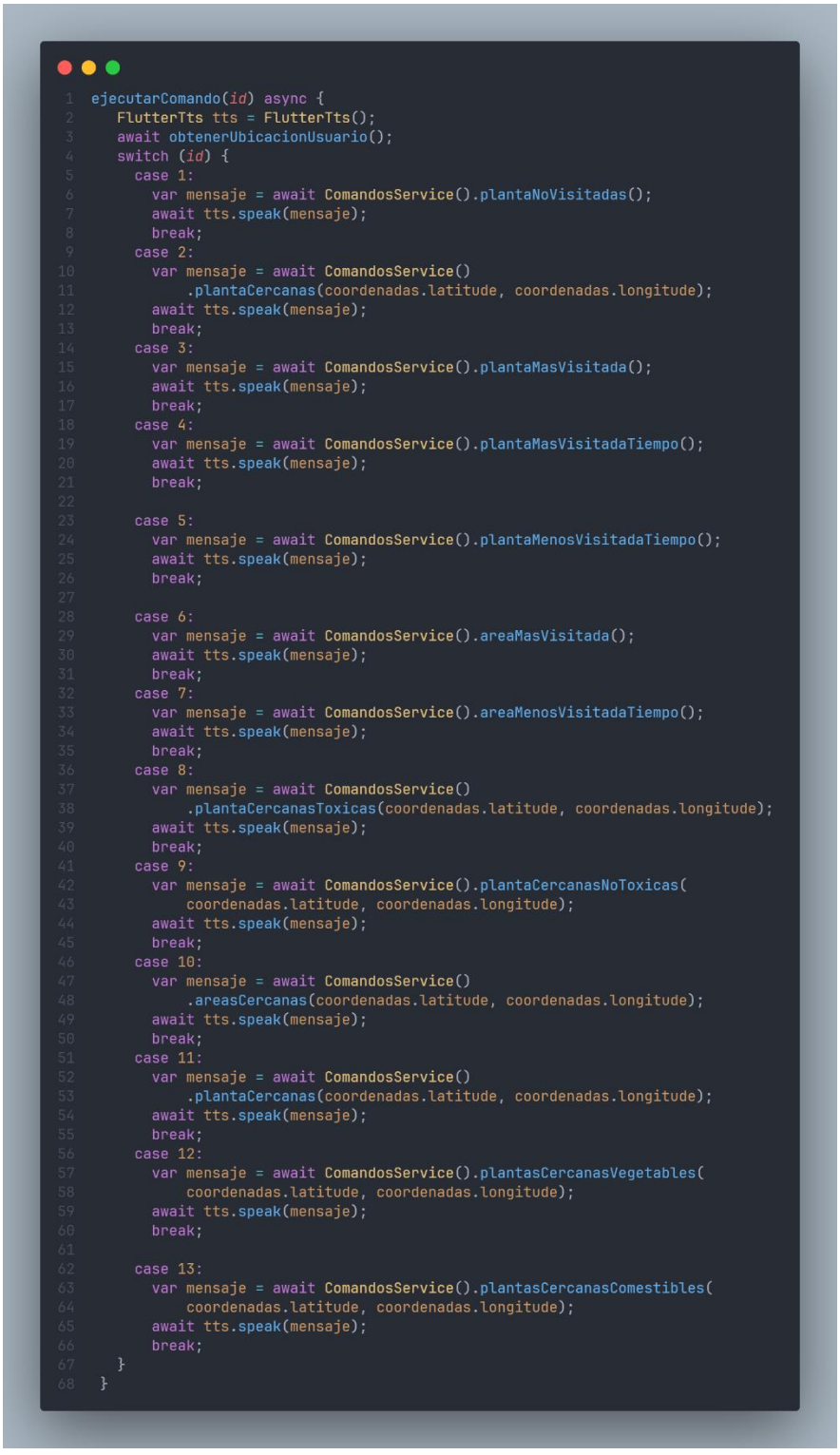
Ilustración 28 Endpoint de laravel consumiento a prolog

Consumiendo Servicios de laravel con prolog en Flutter



```dart
void resultListener(SpeechRecognitionResult result) async {
    await tts.setVolume(1);
    setState(() {
      texto = result.recognizedWords;
    });
    await cambiarEstatus();

    double maxSimilarity = 0.0;
    Comandos matchedCommand = Comandos(comando: '', descripcion: '');

    for (Comandos cmd in comandos!) {
      double similarity = texto.similarityTo(cmd.comando);
      if (similarity > maxSimilarity) {
        maxSimilarity = similarity;
        matchedCommand = cmd;
      }
    }

    double threshold = 0.5;

    if (maxSimilarity >= threshold) {
      if (kDebugMode) {
        print('Comando detectado: ${matchedCommand.descripcion}');
        print('Descripción: ${matchedCommand.descripcion}');
        print("id ${matchedCommand.id}");
      }
      await ejecutarComando(matchedCommand.id);
    }
  }
```

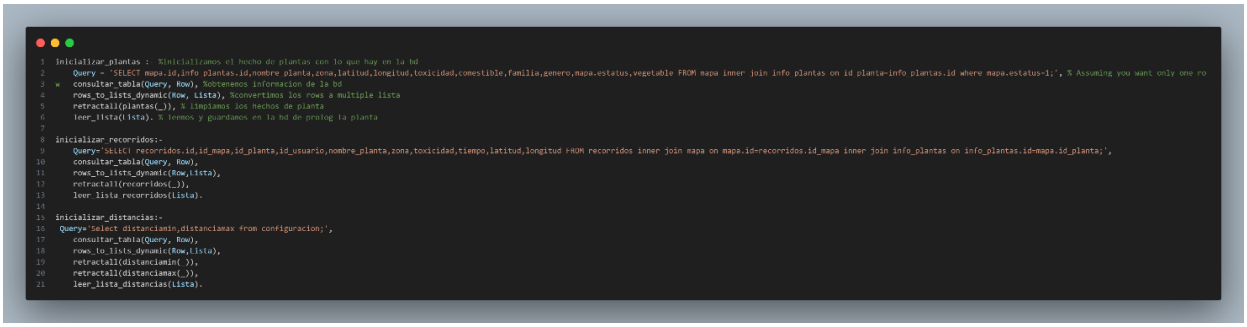
Ilustración 29 Servicio de Reconocimiento de Voz



```
ejecutarComando(id) async {
   FlutterTts tts = FlutterTts();
   await obtenerUbicacionUsuario();
   switch (id) {
     case 1:
       var mensaje = await ComandosService().plantaNoVisitadas();
       await tts.speak(mensaje);
       break;
     case 2:
       var mensaje = await ComandosService()
           .plantaCercanas(coordenadas.latitude, coordenadas.longitude);
       await tts.speak(mensaje);
       break;
     case 3:
       var mensaje = await ComandosService().plantaMasVisitada();
       await tts.speak(mensaje);
       break;
     case 4:
       var mensaje = await ComandosService().plantaMasVisitadaTiempo();
       await tts.speak(mensaje);
       break;

     case 5:
       var mensaje = await ComandosService().plantaMenosVisitadaTiempo();
       await tts.speak(mensaje);
       break;

     case 6:
       var mensaje = await ComandosService().areaMasVisitada();
       await tts.speak(mensaje);
       break;
     case 7:
       var mensaje = await ComandosService().areaMenosVisitadaTiempo();
       await tts.speak(mensaje);
       break;
     case 8:
       var mensaje = await ComandosService()
           .plantaCercanasToxicas(coordenadas.latitude, coordenadas.longitude);
       await tts.speak(mensaje);
       break;
     case 9:
       var mensaje = await ComandosService().plantaCercanasNoToxicas(
           coordenadas.latitude, coordenadas.longitude);
       await tts.speak(mensaje);
       break;
     case 10:
       var mensaje = await ComandosService()
           .areasCercanas(coordenadas.latitude, coordenadas.longitude);
       await tts.speak(mensaje);
       break;
     case 11:
       var mensaje = await ComandosService()
           .plantaCercanas(coordenadas.latitude, coordenadas.longitude);
       await tts.speak(mensaje);
       break;
     case 12:
       var mensaje = await ComandosService().plantasCercanasVegetables(
           coordenadas.latitude, coordenadas.longitude);
       await tts.speak(mensaje);
       break;

     case 13:
       var mensaje = await ComandosService().plantasCercanasComestibles(
           coordenadas.latitude, coordenadas.longitude);
       await tts.speak(mensaje);
       break;
   }
 }
```

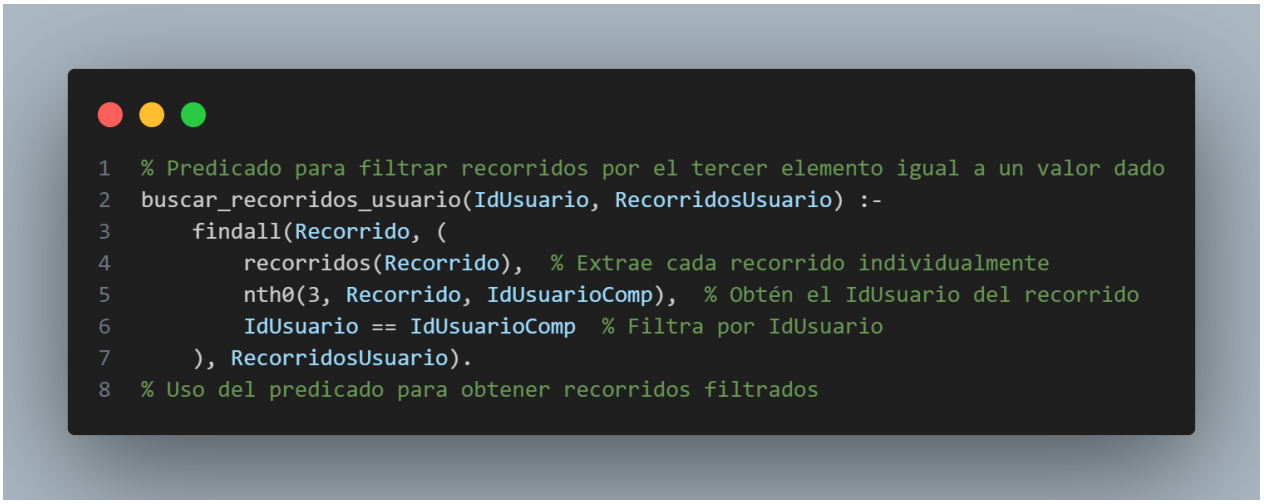
Ilustración 30 Método que Ejecuta el Comando de voz que consume a Prolog

Base de conocimiento de Prolog



```
inicializar_plantas :  %inicializamos el hecho de plantas con lo que hay en la bd
    Query = 'SELECT mapa.id,info_plantas.id,nombre_planta,zona,latitud,longitud,toxicidad,comestible,familia,genero,mapa.estatus,vegetable FROM mapa inner join info_plantas on id_planta=info_plantas.id where mapa.estatus=1;', % Assuming you want only one ro
    consultar_tabla(Query, Row), %obtenemos informacion de la bd
    rows_to_lists_dynamic(Row, Lista), %convertimos los rows a multiple lista
    retractall(plantas(_)), % limpiamos los hechos de planta
    leer_lista(Lista). % leemos y guardamos en la bd de prolog la planta

inicializar_recorridos:-
    Query='SELECT recorridos.id,id_mapa,id_planta,id_usuario,nombre_planta,zona,toxicidad,tiempo,latitud,longitud FROM recorridos inner join mapa on mapa.id=recorridos.id_mapa inner join info_plantas on info_plantas.id=mapa.id_planta;',
    consultar_tabla(Query, Row),
    rows_to_lists_dynamic(Row,Lista),
    retractall(recorridos(_)),
    leer_lista_recorridos(Lista).

inicializar_distancias:-
Query='Select distanciamin,distanciamax from configuracion;',
    consultar_tabla(Query, Row),
    rows_to_lists_dynamic(Row,Lista),
    retractall(distanciamin(_)),
    retractall(distanciamax(_)),
    leer_lista_distancias(Lista).
```

Ilustración 31Obtención de los datos existentes en la BD

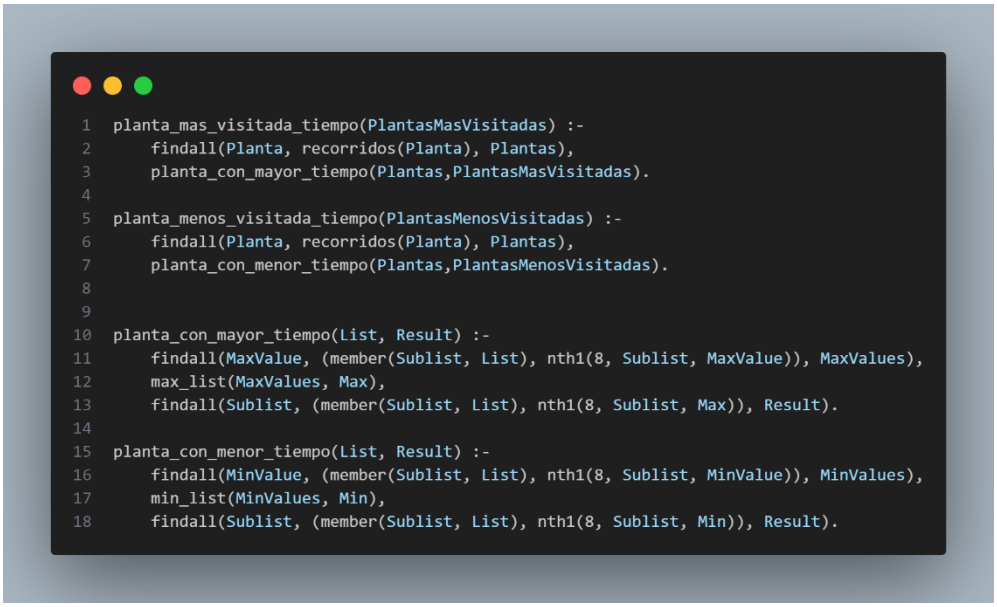


```
% Predicado para filtrar recorridos por el tercer elemento igual a un valor dado
buscar_recorridos_usuario(IdUsuario, RecorridosUsuario) :-
    findall(Recorrido, (
        recorridos(Recorrido),  % Extrae cada recorrido individualmente
        nth0(3, Recorrido, IdUsuarioComp),  % Obtén el IdUsuario del recorrido
        IdUsuario == IdUsuarioComp  % Filtra por IdUsuario
    ), RecorridosUsuario).
% Uso del predicado para obtener recorridos filtrados
```

Ilustración 32Búsqueda de los recorridos de un usuario

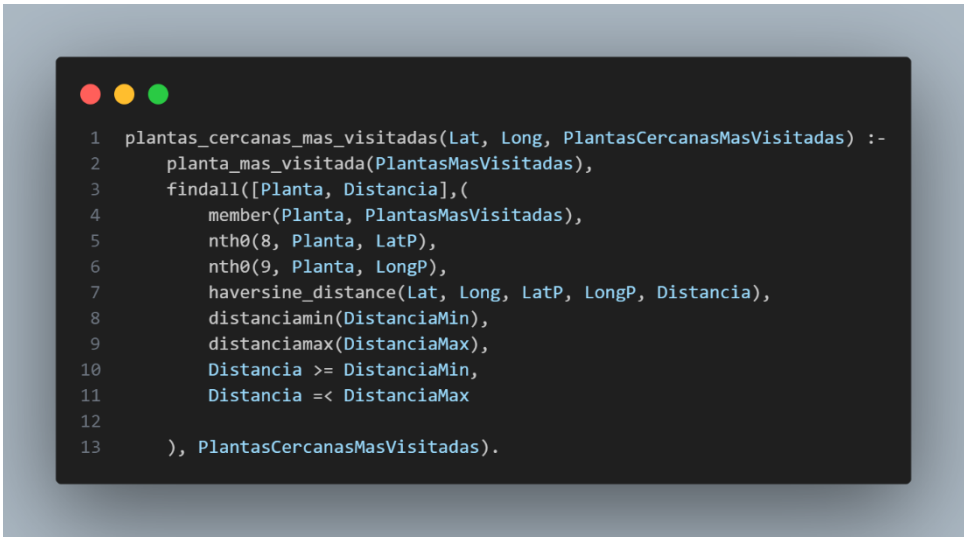
```
plantas_no_visitadas_por_usuario(IdUsuario, PlantasNoEnRecorridos) :-
    buscar_recorridos_usuario(IdUsuario, Recorridos),
    findall(Planta, plantas(Planta), PlantasRegistradas),
    plantas_no_en_recorridos(PlantasRegistradas, Recorridos, PlantasNoEnRecorridos).

plantas_no_visitadas_cercanas_por_usuario(Lat, Long, IdUsuario, PlantasNoEnRecorridos) :
-   plantas_no_visitadas_por_usuario(IdUsuario, PlantasCercanas),
    findall([Planta, Distancia], (
        member(Planta, PlantasCercanas),
        nth0(4, Planta, LatP),
        nth0(5, Planta, LongP),
        haversine_distance(Lat, Long, LatP, LongP, Distancia),
        distanciamin(DistanciaMin),
        distanciamax(DistanciaMax),
        Distancia >= DistanciaMin,
        Distancia =< DistanciaMax
    ), PlantasNoEnRecorridos).

plantas_no_en_recorridos(Plantas, Recorridos, PlantasNoEnRecorridos) :-
    findall(Planta, (
        member(Planta, Plantas),
        nth0(1, Planta, IdPlanta),
        \+ (member(Recorrido, Recorridos),
            nth0(2, Recorrido, IdPlantaRecorrido),
            IdPlantaRecorrido == IdPlanta
       )
    ), PlantasNoEnRecorridos).
```

Ilustración 33 Búsqueda de las plantas visitadas por un usuario



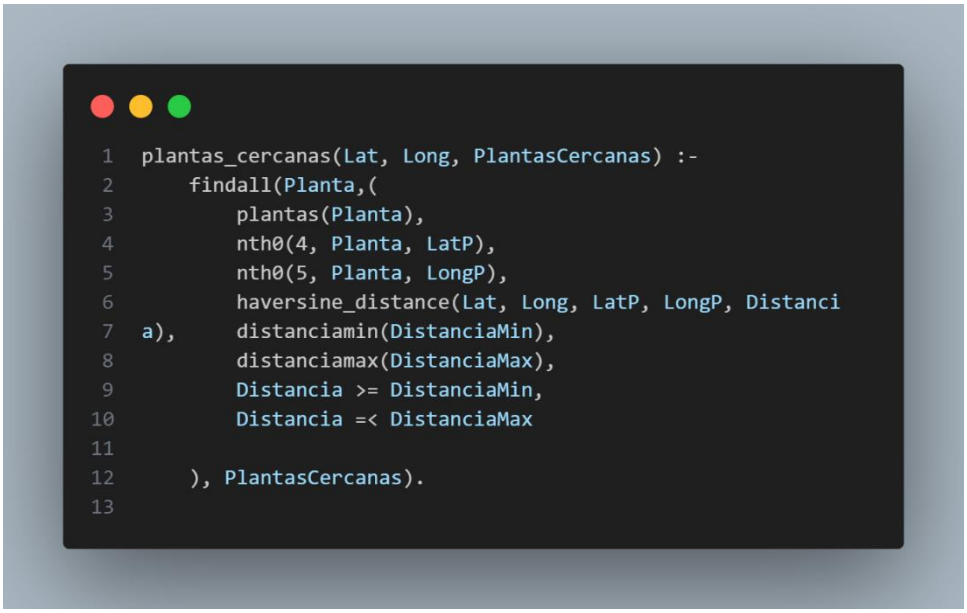
```
planta_mas_visitada_tiempo(PlantasMasVisitadas) :-
    findall(Planta, recorridos(Planta), Plantas),
    planta_con_mayor_tiempo(Plantas,PlantasMasVisitadas).

planta_menos_visitada_tiempo(PlantasMenosVisitadas) :-
    findall(Planta, recorridos(Planta), Plantas),
    planta_con_menor_tiempo(Plantas,PlantasMenosVisitadas).


planta_con_mayor_tiempo(List, Result) :-
    findall(MaxValue, (member(Sublist, List), nth1(8, Sublist, MaxValue)), MaxValues),
    max_list(MaxValues, Max),
    findall(Sublist, (member(Sublist, List), nth1(8, Sublist, Max)), Result).

planta_con_menor_tiempo(List, Result) :-
    findall(MinValue, (member(Sublist, List), nth1(8, Sublist, MinValue)), MinValues),
    min_list(MinValues, Min),
    findall(Sublist, (member(Sublist, List), nth1(8, Sublist, Min)), Result).
```

Ilustración 34 Obtención de las plantas más y menos visitadas



```
plantas_cercanas_mas_visitadas(Lat, Long, PlantasCercanasMasVisitadas) :-
    planta_mas_visitada(PlantasMasVisitadas),
    findall([Planta, Distancia],(
        member(Planta, PlantasMasVisitadas),
        nth0(8, Planta, LatP),
        nth0(9, Planta, LongP),
        haversine_distance(Lat, Long, LatP, LongP, Distancia),
        distanciamin(DistanciaMin),
        distanciamax(DistanciaMax),
        Distancia >= DistanciaMin,
        Distancia =< DistanciaMax

    ), PlantasCercanasMasVisitadas).
```

Ilustración 35 Obtención de las plantas cercanas más visitadas en general



```
plantas_cercanas(Lat, Long, PlantasCercanas) :-
    findall(Planta,(
        plantas(Planta),
        nth0(4, Planta, LatP),
        nth0(5, Planta, LongP),
        haversine_distance(Lat, Long, LatP, LongP, Distanci
a),     distanciamin(DistanciaMin),
        distanciamax(DistanciaMax),
        Distancia >= DistanciaMin,
        Distancia =< DistanciaMax

    ), PlantasCercanas).

```

Ilustración 36 Obtención de las plantas cercanas a mi ubicación

```
plantas_cercanas_toxicas(Lat, Long, PlantasCercanasToxicas) :-
    plantas_toxicas(PlantasToxicas),
    findall(Planta,(
        member(Planta, PlantasToxicas),
        nth0(4, Planta, LatP),
        nth0(5, Planta, LongP),
        haversine_distance(Lat, Long, LatP, LongP, Distancia),
        distanciamin(DistanciaMin),
        distanciamax(DistanciaMax),
        Distancia >= DistanciaMin,
        Distancia =< DistanciaMax

    ), PlantasCercanasToxicas).


plantas_cercanas_no_toxicas(Lat, Long, PlantasCercanasNoToxicas) :
-   plantas_no_toxicas(PlantasNoToxicas),
    findall(Planta,(
        member(Planta, PlantasNoToxicas),
        nth0(4, Planta, LatP),
        nth0(5, Planta, LongP),
        haversine_distance(Lat, Long, LatP, LongP, Distancia),
        distanciamin(DistanciaMin),
        distanciamax(DistanciaMax),
        Distancia >= DistanciaMin,
        Distancia =< DistanciaMax

    ), PlantasCercanasNoToxicas).
```

Ilustración 37 Búsqueda de las plantas toxicas y no toxicas

Video de Funcionamiento del Sistema: https://youtu.be/PiLG6ZlXheM

Conclusiones

La realización de este proyecto fue muy enriquecedora en muchos aspectos, pudimos aprender muchos conocimientos nuevos sobre el área de estudio de la inteligencia artificial, así como reforzar conocimientos adquiridos anteriormente referentes a todas las etapas del desarrollo de un producto de software.

Durante este proyecto desarrollamos un agente que consta de dos aplicaciones que pueden ser usadas por dos tipos de usuarios.

Creemos que nuestro Sistema de Seguimiento y recomendación de plantas puede ser muy útil si se aplica de la manera correcta en diferentes ámbitos.

Haciendo uso de Prolog, realizamos una base de conocimiento que nos ayudara a procesar los datos ya existentes y a partir de estos obtener otros nuevos, pero teniendo una mayor cantidad de datos para procesar se puede crear un conocimiento cada vez más grande haciendo que este tipo de proyectos tomen mayor importancia; tal como mencionamos anteriormente, creemos de gran importancia que la población comience a conocer más sobre las plantas que nacen en la comunidad y sobre todo que conozcan las formas en que pueden usarlas para sacarles provecho, esto siempre en busca de preservar los espacios verdes para aportar a la mejora del medio ambiente.

Esta aplicación puede ser reforzada con muchas más funcionalidades que con el tiempo vayan haciendo que el agente sea cada vez más completo, de este modo se podrán registrar las plantas que hay en un área mucho más amplia.

De poderse usar por los habitantes de toda una comunidad puede haber una base de datos mucho más rica que aporte a la creación de nuevo conocimiento.

Esta fue solo una idea que tuvimos un grupo de 4 estudiantes, pero creemos que existen muchos más proyectos relacionados y pensamos que hay muchas más personas que piensen que con ayuda de nuevas tecnologías, como la inteligencia artificial, podemos ayudar a que las personas se interesen en el cuidado del medio ambiente.

Axel Cinco:

El haber participado en este proyecto me hizo aprender nuevas herramientas tecnológicas que se están utilizado hoy en día, una de ella fue la inteligencia artificial donde aprendimos conceptos sobre ella y cómo podemos implementarla en nuestros proyectos, también otra de las cosas que aprendí fue a conocer diversos tipos de plantas y características de ellas, beneficios, y por qué no un poco de historia sobre ellas.

El haber implementado nuevas herramientas en el proyecto como lo es prolog fue una tarea complicada al no tener muchos conocimientos sobre la implementación de dicha herramienta en nuestro proyecto, pero también fue gratificante el haber utilizado dicha herramienta porque me hizo conocer su funcionamiento, la creación de hechos y reglas entre otras cosas que ayudaron al proyecto hacer más efectivo para la realización de consultas y búsquedas de plantas.

Carlos Sandoval:

La ejecución de este proyecto fue tanto interesante como desafiante, destacándose la implementación gradual y metódica de algoritmos de geolocalización de plantas y el uso de la base de conocimientos en Prolog. Esta combinación reveló la complejidad en la intersección de la biología vegetal y la tecnología avanzada. Explorar el mundo de las plantas mediante la integración de sensores y algoritmos amplió mi perspectiva, enfocándome no solo en la importancia vital de las plantas, sino también en cómo la tecnología puede fortalecer la capacidad de estudio y preservación de la biodiversidad. La resolución de desafíos fue constante fuente de aprendizaje y crecimiento, especialmente al enfrentar retos significativos en la creación de algoritmos y el manejo de datos geoespaciales con la aplicación de la lógica en Prolog.

Yilma Ruiz:

En el transcurso de esta asignatura tuvimos la oportunidad de conocer varios de los usos más comunes que se le da a la inteligencia artificial hoy en día, que abarcan desde tareas muy simples como responder una pregunta hasta aquellas más complejas como hacer reconocimiento y la generación de contenido textual o visual altamente específico.

La exploración del campo de la inteligencia artificial y su aplicación práctica en el cuidado del medio ambiente muestra una combinación fascinante de tecnología y conciencia ecológica. Me parece muy importante usar la tecnología más nueva para algo que cuente con el potencial de impactar positivamente a la comunidad y al entorno. En resumen, parece que este proyecto no solo ha ampliado nuestro conocimiento técnico sobre algunas herramientas, sobre todo con el lenguaje de programación Prolog, sino que también ha plantado la semilla para futuras ideas innovadoras y sostenibles.

Axel Sanchez:

El desarrollo de este proyecto fue muy enriquecedor para mí en muchos aspectos, tanto para reforzar temas técnicos como programación, análisis, diseño y muchos otros aspectos que se tienen que tomar en cuenta a la hora de desarrollar un producto de software, así como también aspectos sociales como el trabajo en equipo, la tolerancia, la planeación y formación de estimaciones, etc.

Este proyecto en particular fue muy interesante de hacer, ya que pude aprender sobre el desarrollo en el campo de la inteligencia artificial, espero poder seguir aprendiendo con el paso del tiempo durante mi etapa profesional.

Referencias Bibliográficas

https://trefle.io/

https://identify.plantnet.org/es

https://flutter.dev/

https://angular.io/

https://github.com/dlutton/flutter_tts

https://github.com/baseflow/flutter-geolocator/tree/main/geolocator

https://github.com/fleaflet/flutter_map

https://leafletjs.com/

Flutter (software) - Wikipedia, la enciclopedia libre