**Documentación del Proyecto - Recomendaciones Agrónomas**

**Descripción del Proyecto**

Este proyecto implementa un Sistema de Recomendaciones Agrónomas que utiliza **TensorFlow.js** para proporcionar recomendaciones en tiempo real para el cuidado de plantas. El sistema toma parámetros como la humedad del aire, la humedad del suelo, la temperatura, y ahora incluye datos adicionales de API externas (como **Trefle API**) para generar sugerencias precisas sobre el cuidado de plantas. El modelo predictivo se ejecuta directamente en el navegador, lo que permite procesar los datos de manera local, rápida y sin depender de servidores remotos.

**Tecnologías Utilizadas**

* **React.js**: Framework para la creación de interfaces interactivas.
* **TensorFlow.js**: Para la creación y ejecución de modelos de aprendizaje automático en el navegador.
* **Node.js**: Backend del proyecto, maneja las peticiones y sirve los archivos.
* **Bootstrap**: Framework CSS para mejorar la presentación de la interfaz.
* **Chart.js**: Para la creación de gráficos interactivos que muestran datos climáticos y parámetros de la planta.
* **Axios**: Para consumir datos de la API externa **Trefle**.
* **npm**: Utilizado para gestionar las dependencias del proyecto.

**Dependencias**

El proyecto incluye las siguientes dependencias, que están listadas en el archivo package.json:

* **tensorflow/tfjs**
* **express**
* **dotenv**
* **cors**
* **axios**
* **chart.js**
* **react-router-dom**

**Estructura del Proyecto**

La estructura del proyecto es la siguiente:

* **/public**: Archivos estáticos como HTML, CSS y JavaScript para la interfaz gráfica.
* **/src**: Contiene los scripts para la lógica del frontend, incluyendo los componentes de React.
* **/models/**: Modelos de **TensorFlow.js** utilizados para las recomendaciones de cuidados de las plantas.
* **server.js**: Script principal que inicializa el servidor en Node.js.
* **TrefleAPI.js**: Código para consumir la API de Trefle y mostrar datos de plantas.

**Funcionalidades Principales**

1. **Recomendaciones Basadas en Parámetros de Plantas**: El sistema toma datos como humedad, luz y temperatura y genera recomendaciones para el riego y cuidados específicos.
2. **Predicciones en Tiempo Real**: A través de **TensorFlow.js**, el modelo se ejecuta en el navegador, ofreciendo resultados instantáneos sin necesidad de conectarse a servidores remotos.
3. **Interfaz de Usuario Intuitiva**: La interfaz gráfica está diseñada para que el usuario ingrese fácilmente los datos y reciba sugerencias de forma rápida y eficiente.
4. **Consumo de API Externa (Trefle API)**: Se ha implementado una integración para consumir datos de plantas a través de la API de **Trefle**.
5. **Gráficos Dinámicos**: Utilización de **Chart.js** para crear gráficos interactivos que muestran la evolución de parámetros como la humedad y temperatura.

**Recomendaciones para Mejoras Futuras**

1. **Ampliación del Modelo**: Se puede ampliar el modelo para incorporar más parámetros, como el tipo de suelo o fertilizantes específicos.
2. **API Externa para Datos Climáticos**: Integrar una API que proporcione datos climáticos en tiempo real podría mejorar la precisión de las recomendaciones.
3. **Aplicación Móvil**: Convertir el proyecto en una aplicación móvil para mejorar la accesibilidad y el uso en campo.
4. **Mejora en la Interfaz de Usuario**: Añadir elementos visuales más detallados para mejorar la experiencia del usuario al recibir las recomendaciones.
5. **Soporte para Varios Idiomas**: Internacionalizar la aplicación para admitir más idiomas.

**Requisitos**

* **Node.js**
* **npm** o **yarn**

**Instalación**

1. Clona el repositorio:

bash

Copiar código

git clone https://github.com/tokien736/recomendaciones-agronomas.git

1. Navega al directorio del proyecto:

bash

Copiar código

cd recomendaciones-agronomas

1. Instala las dependencias:

bash

Copiar código

npm install

o

bash

Copiar código

yarn install

**Uso**

1. Inicia el servidor de desarrollo:

bash

Copiar código

npm start

o

bash

Copiar código

yarn start

1. Abre tu navegador y navega a <http://localhost:3000> para ver la aplicación en acción.
2. Introduce los datos de la planta (como la humedad del suelo, la luz recibida y la temperatura) y obtén recomendaciones personalizadas para su cuidado.

**Ejemplo de Código**

A continuación se muestra un ejemplo de cómo se puede utilizar **TensorFlow.js** para crear un modelo de recomendaciones:

javascript

Copiar código

import \* as tf from '@tensorflow/tfjs';

// Definir el modelo

const model = tf.sequential();

model.add(tf.layers.dense({ units: 32, activation: 'relu', inputShape: [inputFeatures] }));

model.add(tf.layers.dense({ units: 1, activation: 'linear' }));

// Compilar el modelo

model.compile({ optimizer: 'adam', loss: 'meanSquaredError' });

// Entrenar el modelo

async function trainModel(data) {

const { inputs, labels } = data;

await model.fit(inputs, labels, {

epochs: 50,

batchSize: 32,

});

}

// Predecir recomendaciones

async function predict(inputData) {

const prediction = model.predict(tf.tensor2d([inputData], [1, inputFeatures]));

return prediction.dataSync()[0];

}

**Actualizaciones Recientes**

1. **Migración de HTML a React**: Se ha migrado la interfaz de HTML estático a componentes **React.js**, permitiendo una mayor modularidad y flexibilidad en la construcción de la aplicación.
2. **Consumo de Trefle API**: Se implementó la integración con la API de Trefle para obtener datos de plantas en tiempo real, como sus nombres comunes y científicos, y las imágenes de las plantas.
3. **Gestión de Errores y CORS**: Se manejaron los errores relacionados con CORS y problemas de autorización para asegurar que la API funcione correctamente en el entorno de desarrollo.