Documentación Técnica de la Aplicación Web

# 1. Resumen

Esta aplicación web es un sistema integral que combina datos de sensores ambientales con recomendaciones agrícolas. Utiliza React para el frontend, Firebase como backend en tiempo real, y Chart.js para la visualización de datos. El sistema proporciona recomendaciones de cultivos basadas en condiciones ambientales actuales y muestra mapas interactivos con información meteorológica.

# 2. Estructura del Proyecto

El proyecto está compuesto por los siguientes archivos clave:  
1. index.html  
2. condiciones\_cultivos.json  
3. dataUploader.js  
4. firebase.js  
5. graficas.js  
6. maps.js  
7. model.js  
8. recommendation.js  
9. weather.js  
10. style.css  
11. styleSensores.css  
12. App.js

## 2.1. index.html

Actúa como plantilla principal para la aplicación React.  
Incluye referencias al favicon y al archivo manifest.json.  
Contiene el <div id="root"></div> donde se montará la aplicación React.  
Incluye el script para cargar la API de Google Maps con una clave de API específica.

## 2.2. condiciones\_cultivos.json

Almacena una lista de cultivos y sus condiciones ambientales óptimas.  
Cada cultivo incluye:  
 - nombre: Nombre del cultivo.  
 - condiciones: Objeto que contiene humedad\_ambiental, humedad\_suelo, y temperatura.

## 2.3. dataUploader.js

Simula datos de sensores y los envía a Firebase periódicamente.  
 - Inicializa la aplicación Firebase y obtiene una referencia a la base de datos.  
 - generateSensorData: Genera datos aleatorios de temperatura, humedad, y humedad del suelo.  
 - sendDataToFirebase: Envía los datos generados a Firebase.

## 2.4. firebase.js

Gestiona la conexión y la comunicación con Firebase.  
 - Inicializa Firebase con la configuración proporcionada.  
 - convertirHumedadSuelo: Convierte valores de humedad del suelo de 0-1023 a 0-100%.  
 - Escucha cambios en los datos y actualiza la interfaz de usuario.

## 2.5. graficas.js

Proporciona funciones para crear gráficos utilizando Chart.js.  
 - createHumidityBarChart: Gráfico de humedad ambiental.  
 - createSoilHumidityBarChart: Gráfico de humedad del suelo.  
 - createTemperatureBarChart: Gráfico de temperatura.

## 2.6. maps.js

Maneja la inicialización y visualización de Google Maps y los datos meteorológicos.  
 - Utiliza la API de geolocalización para obtener la posición actual del usuario.  
 - Crea un mapa centrado en la ubicación del usuario y llama a getWeather para obtener información meteorológica.

## 2.7. model.js

Implementa la lógica de un modelo de aprendizaje automático con TensorFlow.js para recomendaciones de cultivos.  
 - convertirHumedadSuelo: Convierte valores de humedad del suelo.  
 - loadCultivos: Carga los datos de condiciones\_cultivos.json.  
 - createAndTrainModel: Crea y entrena un modelo de TensorFlow.

## 2.8. recommendation.js

Gestiona la renderización de mapas y gráficos para las recomendaciones.  
 - createGoogleMap: Crea y muestra un mapa de Google.  
 - renderChart: Renderiza un gráfico de ejemplo.

## 2.9. weather.js

Obtiene datos meteorológicos de la API de OpenWeatherMap.  
 - getWeather: Función que recibe latitud y longitud y devuelve datos meteorológicos.  
 - Maneja la respuesta y los errores de la solicitud.

## 2.10. style.css

Define los estilos globales y el tema de la aplicación.  
 - Utiliza variables CSS para colores y tipografías.  
 - Incluye estilos para el cuerpo, navbar, mapas, encabezados y footer.

## 2.11. styleSensores.css

Estilos específicos para los componentes de visualización de datos de sensores.  
 - Estilos para escalas de humedad, barras de progreso, y números en escalas.

## 2.12. App.js

Archivo principal de la aplicación que define las rutas y renderiza los componentes.  
 - Utiliza react-router-dom para manejar la navegación entre las rutas principales.

# 3. Notas Adicionales

Configuración de Firebase:  
- Reemplaza las claves y configuraciones de Firebase con las de tu proyecto.  
- Protege tus claves de API y evita exponerlas públicamente.  
  
Claves de API:  
- Google Maps: Reemplaza 'YOUR\_API\_KEY' con tu clave de API en index.html y maps.js.  
- OpenWeatherMap: Actualiza la clave de API en weather.js.  
  
Dependencias:  
- Instala las dependencias necesarias: React, Firebase, Chart.js, TensorFlow.js.  
  
Pruebas:  
- Utiliza dataUploader.js para simular datos y verifica el correcto funcionamiento de los componentes.

# 4. Conclusión

Esta documentación proporciona una visión detallada de cada archivo en el proyecto, explicando su propósito y cómo interactúan entre sí. Con esta información, podrás mantener y extender la funcionalidad de tu aplicación web.