Proyecto3A_Server_Frontend

Generado por Doxygen 1.12.0

Capítulo 1

Sistema de Monitoreo de Sensores

Este proyecto consiste en un sistema de monitoreo que utiliza un sensor de temperatura y un sensor de ozono. Los datos son enviados desde un dispositivo Arduino a una aplicación frontend, donde se visualizan en tiempo real. La aplicación se comunica con un servidor que almacena y procesa la información.

1.1. Tabla de Contenidos

1.2. Características

- Monitoreo en tiempo real de los niveles de ozono y temperatura.
- Interfaz web que muestra los datos de los sensores.
- Actualización automática de datos cada 5 segundos.
- Almacenamiento de datos en una base de datos PostgreSQL a través de una API REST.

1.3. Requisitos

- Arduino con sensor de temperatura y ozono.
- Node.js y npm para el backend.
- PostgreSQL para la base de datos.
- Un entorno web compatible con JavaScript.

1.4. Estructura del Proyecto

```
/Proyecto3A_Server_Frontend
    /public
        index.html # Archivo HTML principal
    /src
        /app
            /css
                styles.css # Estilos para la interfaz
            /img
            /js
            api.js # Lógica del frontend
```

1.5. Instalación

1. Clona el repositorio:
2. ```
   git clone https://github.com/tu-usuario/tu-repositorio.git
   cd tu-repositorio
   ```

1. Instala las dependencias del backend:
   ```
   cd backend
   npm install
   ```
2. Configura la base de datos PostgreSQL:
   - Crea una base de datos y configura las tablas necesarias.
3. Carga el código de Arduino en tu placa.

1.6. Uso

1. Inicia el servidor:
   ```
   cd backend
   node server.js
   ```
2. Abre index.html en un navegador web para ver los datos en tiempo real.

1.7. Contribuciones

Las contribuciones son bienvenidas. Por favor, sigue estos pasos:

```
1. Haz un fork del proyecto.
2. Crea una nueva rama (git checkout -b feature/nueva-característica).
3. Realiza tus cambios y haz un commit (git commit -m 'Agregué una nueva característica').
4. Haz un push a la rama (git push origin feature/nueva-característica).
5. Abre un Pull Request.
```

1.8. Licencia

Este proyecto está licenciado bajo la Licencia MIT. Consulta el archivo LICENSE para más detalles.
```

Capítulo 2

Índice de archivos

2.1. Lista de archivos

Lista de todos los archivos con breves descripciones:

Capítulo 3

Documentación de archivos

3.1. Referencia del archivo README.md

3.2. Referencia del archivo src/app/js/api.js

Funciones

- async function fetchLatestSensorData ()

  Función asíncrona para obtener los datos más recientes de ozono y temperatura del servidor.

Variables

- let previousOzoneTimestamp = null
- let previousTemperatureTimestamp = null
- window onload

  Función que se ejecuta al cargar la página.

3.2.1. Documentación de funciones

3.2.1.1. fetchLatestSensorData()

```
async function fetchLatestSensorData ()
```

Función asíncrona para obtener los datos más recientes de ozono y temperatura del servidor.

Esta función realiza una solicitud a la API del servidor para obtener los últimos datos de los sensores de ozono y temperatura. Los datos se utilizan para actualizar la interfaz de usuario con los valores actuales.

@async @function fetchLatestSensorData

Devuelve

{Promise<void>} No devuelve ningún valor.

3.2.2. Documentación de variables

3.2.2.1. onload

```
window onload
```

Valor inicial:

```
= () => {
    fetchLatestSensorData();
    setInterval(fetchLatestSensorData, 1000);
}
```

Función que se ejecuta al cargar la página.

Llama a la función `fetchLatestSensorData` para obtener los datos iniciales de los sensores y configura un intervalo para actualizar los datos cada 1 segundo.

3.2.2.2. previousOzoneTimestamp

```
let previousOzoneTimestamp = null
```

3.2.2.3. previousTemperatureTimestamp

```
let previousTemperatureTimestamp = null
```

Índice alfabético