Santo Tomas

Informe de proyecto detector fuga de gas

Profesor responsable:
Christian Rivero Morales
Estudiantes:
Benjamín Montoya Oporto
Maximiliano Esparza Rifo

11 – 2024

Índice

Introducción

El proyecto pretende desarrollar un sistema de monitoreo de gas y temperatura con el microcontrolador ESP32. Este sistema está diseñado para detectar niveles peligrosos de gas en el ambiente y medir la temperatura, proporcionando alertas visuales, sonoras y enviando datos a una base de datos a través de Node-RED

El proyecto combina tecnologías de sensores, redes inalámbricas, siendo ideal para aplicaciones en hogares, industrias o cualquier otro entorno que requiera monitoreo de seguridad y comodidad.

Objetivos

Objetivo general

- Diseñar un sistema que monitoree niveles de gas y temperatura, emita alertas en tiempo real y registre datos en una base de datos utilizando Node-RED.

Objetivos específicos

- Implementar el sensor **MQ-135** para detectar concentraciones de gas y humo en el ambiente.
- Incorporar el sensor **DHT11** para medir temperatura y humedad.
- Desarrollar una interfaz visual con un **LED RGB** para indicar niveles seguros o peligrosos de gas.
- Integrar un **buzzer** para emitir alertas sonoras.
- Establecer comunicación entre el ESP32 y Node-RED mediante **MQTT** para el envío de datos en tiempo real.
- Configurar Node-RED para almacenar y visualizar datos en un dashboard o base de datos.

Materiales Utilizados

Componentes	Cantidad	Descripción
ESP32	1	Microcontrolador con conexión a Wi-Fi.
Sensor MQ-135	1	Sensor para detección de gases nocivos.
Sensor DHT11	1	Sensor de temperatura y humedad.
LED RGB	1	Indicador visual de estado.
Buzzer	1	Genera las señales sonoras.
Resistencia	2	Limitador de corriente para LED RGB.
Protoboard y cables	-	Para conexiones y pruebas del circuito.

Metodología

Diseño del circuito:

- Los sensores MQ-135 y DHT11 se conectaron al ESP32 para realizar lecturas de gas y temperatura.
- El LED RGB se configuró para cambiar de color según el nivel de gas detectado (verde para niveles de gas seguros, rojo para niveles de gas s peligrosos)
- Se añadió un buzzer para alertas sonoras en caso de condiciones críticas.

Programación del ESP32:

- Se escribió un programa en Arduino IDE para leer datos de los sensores, controlar el LED RGB y el buzzer, y enviar datos a Node-RED utilizando el protocolo MQTT.

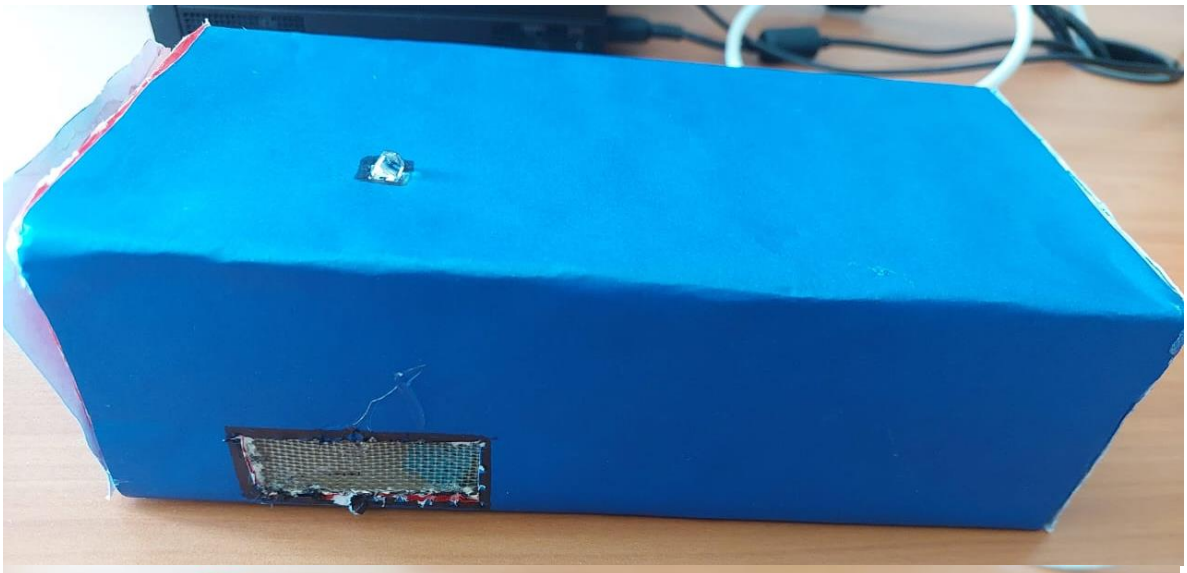
Configuración de Node-RED:

- Se configuró un flujo en Node-RED para recibir datos de temperatura y gas desde el ESP32.
- Los datos se almacenaron en una base de datos.

Pruebas y Validación:

- Se realizaron pruebas en diferentes niveles de gas simulados para validar el comportamiento del sistema.
- Se verificó la correcta conexión y envío de datos a Node-RED.

Evidencia fotográfica del proyecto



Documentación Código

```
1   #include <WiFi.h>
2   #include <PubSubClient.h>
3   #include <DHT.h>
```

Esta parte del código se encarga de incluir el uso de librerías para conectarse a Wi-Fi, conectase a MQTT y permite el uso de DHT11

```
6   #define DHTPIN 14
7   #define DHTTYPE DHT11
8   DHT dht(DHTPIN, DHTTYPE);
```

Estas líneas permiten la configuración del DHT11

```
10  // Configuración del MQ-135
11  #define MQ135_PIN 34
```

Línea que define el pin par el uso del sensor MQ135

```
13  // Configuración del LED RGB
14  #define LED_R 25          // Canal rojo conectado al GPIO 25
15  #define LED_G 26          // Canal verde conectado al GPIO 26
```

Líneas que definen los pines para los colores del led RGB

```
17  // Configuración del Buzzer
18  #define BUZZER_PIN 23     /
```

Línea para utilizar el Buzzer y definir el pin de uso

```
23  // W1-F1
24  const char* ssid = "SensorGas";
25  const char* password = "sensor";
```

Líneas que permiten al esp32 conectarse a una red Wi-fi.

```
27  // MQTT
28  const char* mqtt_server = "mqtt.eclipseprojects.io"; // Dirección del broker MQTT
29  const char* topic_gas = "sensor/gas";
30  const char* topic_temp = "sensor/temperature";
```

Líneas de código que donde configuramos el MQTT

```
46    connectWiFi();
47
48
49    client.setServer(mqtt_server, 1883);
50  }
```

Líneas donde nos permita conectarnos a internet y también configurar el MQTT para conectarnos con él.

```
52  void loop() {
53
54    if (!client.connected()) {
55      reconnectMQTT();
56    }
57    client.loop();
58
59
60    float temperatura = dht.readTemperature();
61    int mq2Value = analogRead(MQ2_PIN);
```

Con la línea de código 54 vemos si nuestro cliente esta conecta y así asegurarnos, después en la línea 60 y 61 podremos leer los datos de nuestro DHT11 y nuestro MQ135.

```
69    client.publish(topic_temp, String(temperatura).c_str());
70    client.publish(topic_gas, String(mq2Value).c_str());
```

Con estas dos líneas de código mandamos los datos hacia el NODE-RED.

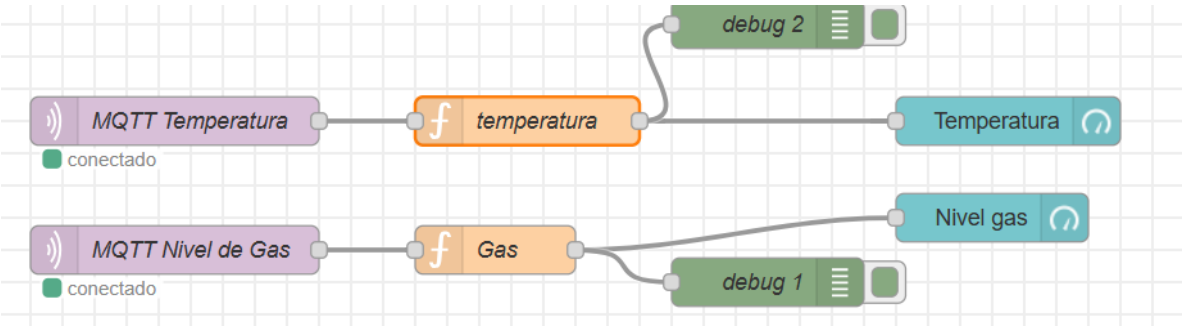
```
74    if (mq2Value > UMBRAL_GAS) {
75      setColor(255, 0);
76      activateBuzzer();
77    } else {
78      setColor(0, 255);
79      deactivateBuzzer();
80    }
81
82    delay(2000);
83  }
```

En este if vemos la configuración de nuestro led y el buzzer para realizar una acción a los distintos niveles de gas, en este caso el led estaría en verde si el nivel de gas es moderado, en caso de que sea peligroso el led se colocara rojo sonando el buzzer, todo con un delay de 2 segundo de reacción.

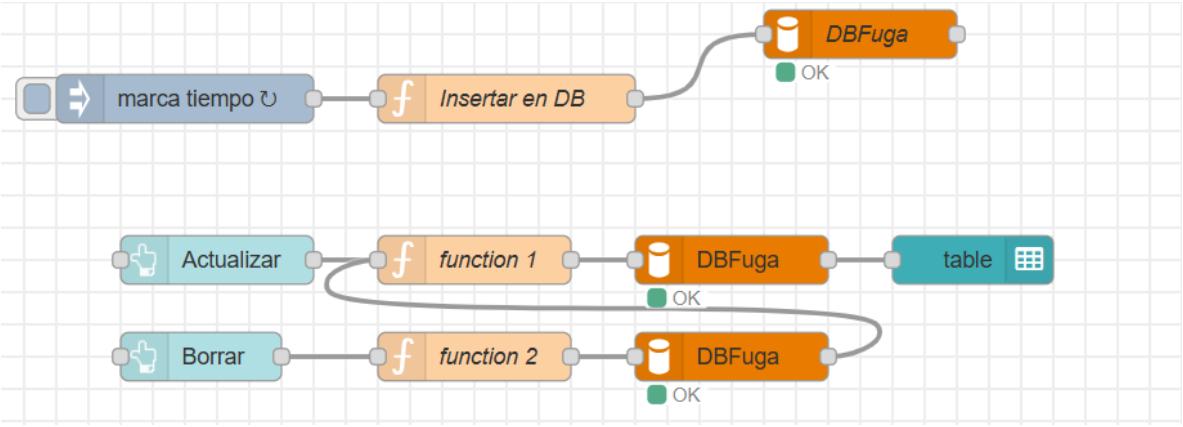
```
void connectWiFi() {
  Serial.print("Conectando a Wi-Fi");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWi-Fi conectado.");
}
```

En este void vemos una función para conectarnos a wifi, el while es un bucle que nos muestra si mientras no esté conectado seguirá intentando conectarse imprimiendo un punto con un delay de 500 milisegundos, pero si está conectado nos imprimirá un “\nWi-Fi conectado. “.

Código de Node-Red:



Este flujo de node red más que nada funciona para conectar e ingresar los datos que toma nuestro sensor en el esp32, al node mediante MQTT para luego mostrarlo en el dashboard con un medidor de datos.



Despues vendría un insert para que los datos que se envían del Arduino hacia el mqtt se guarden en una base de datos de sql, actualizándose a tiempo real.

Luego estaría un botón actualizar y borrar para la tabla de contenidos, más que nada son un select y un delete los distintos botones, para luego todo eso mandarlo en una tabla conectada a Sql

Resultados Obtenidos

Monitoreo Ambiental:

- El sistema detectó cambios en los niveles de gas con precisión, mostrando colores correspondientes en el LED RGB (verde para seguro, rojo para peligroso).

- El buzzer se activó correctamente en situaciones de peligro, generando alertas audibles.

Conexión a Node-RED:

- Los datos de temperatura y gas se transmitieron con éxito a Node-RED mediante MQTT.
- Se visualizó la información en un dashboard en tiempo real y los datos se almacenaron en la base de datos.

Linkografía

Apoyo de IA: https://chatgpt.com

Conocer el sensor MQ135: https://naylampmechatronics.com/blog/42_tutorial-sensores-de-gas-mq2-mq3-mq7-y-mq135.html

Como utilizar esp32: https://programarfacil.com/esp8266/programar-esp32-ide-arduino/

Como conectar mqtt en nodered: https://ine4celectronics.com/introduccion-mqtt-con-node-red-y-arduino/

Conclusiones

En este proyecto logramos solucionar los problemas que nos ha dado en los meses, a pesar de las pocas capacidades de codificación, logramos la idea que tuvimos, buscamos información de distintos lugares para lograr nuestro proyecto. Pensar en lograr un proyecto que podría salvar las vidas de las personas nos hace ver que somos capaces de lograr algo si

nos lo proponemos, vimos el potencial de cada uno al conseguir este proyecto, vimos que podemos salvarle la vida a muchas personas, quizás nos faltó más tiempo para lograr hacer un prototipo mejor y más llamativo, pero la idea de hacer este detector de fuga se llegó, el potencial de nuestro proyecto puede salvar vidas, supimos darle un uso al Arduino, avanzamos en conocimiento que nos puede ayudar a motivarnos para crear y generar más proyectos aun. Este trabajo abre puertas a futuros proyectos que abran en mente para otra ocasión ya que más que nada es un prototipo logrado en un par de meses, el objetivo fue logrado y el resultado de nuestro detector de fuga de gas fue mejor de lo que pensábamos, no teníamos muchas esperanzas, pero a pesar de todo salió mejor de lo esperado