Project Report
On
IoT-based Gas and Flame Detection System using ESP32 and Blynk.

Course: Embedded Systems & IoT Lab

Course Code: SWE 466

Submitted to:

Nawshad Ahmed Chowdhury
Department Head [EEE]
Metropolitan University, Sylhet

Submitted by:

Group Members:

- **Shafat Alam** (ID: 213-134-001)
- **Marjana Begum** (ID: 221-134-020)
- **Tanvir Mahmud Ove** (ID: 221-134-023)
- **Tahmid Samin** (ID: 221-134-032)
- **Moumita Das Mou** (ID: 221-134-017)

Title: IoT-based Gas and Flame Detection System using ESP32 and Blynk.

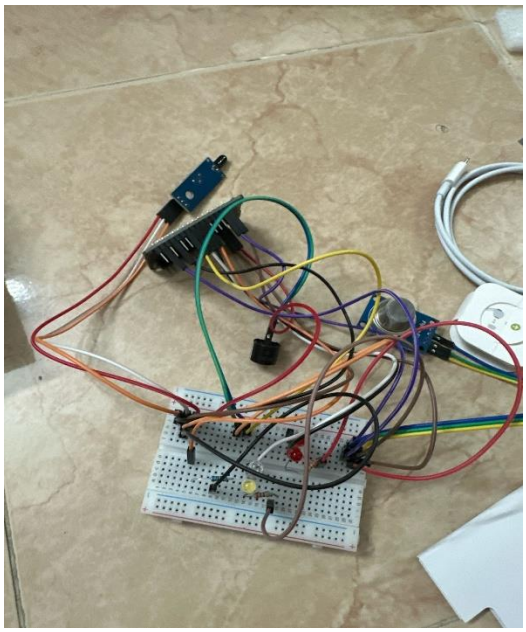
1. Objective

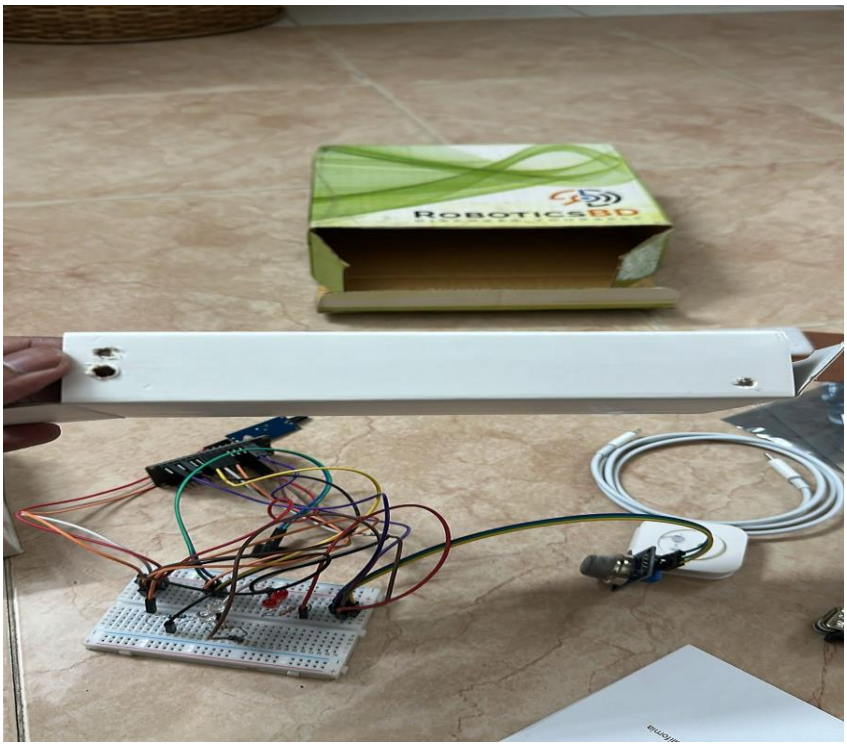
The goal of this project is to develop a smart and affordable Fire and Gas Detector System using the **ESP32 microcontroller** and **IoT integration via Blynk**. The system detects hazardous events such as gas leaks or fire and triggers:

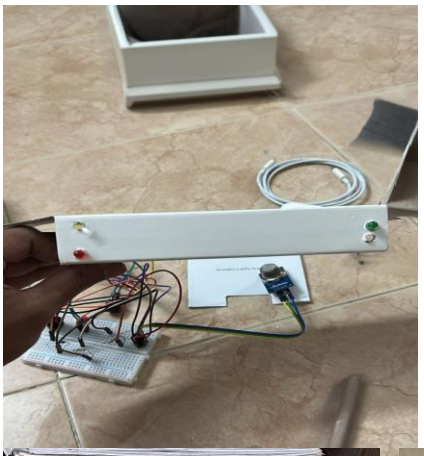
- Real-time alerts to mobile devices through the Blynk app.
- Visual alerts via color-coded LEDs.
- Audio alerts using a buzzer.

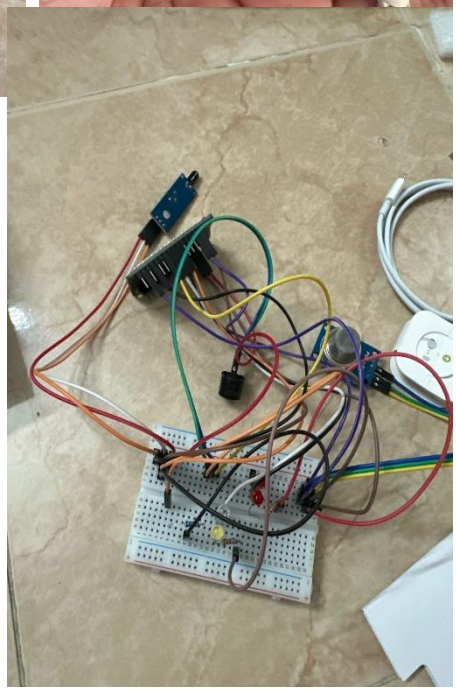
This system is aimed at enhancing safety in residential and industrial environments by providing immediate responses to dangerous conditions.

2. Project Photograph









3. Procedure

Component Integration

The following components were used:

- **ESP32** – Wi-Fi-enabled microcontroller
- **MQ2 Gas Sensor** – Detects smoke, LPG, and methane gases
- **Flame Sensor** – Detects infrared signals from flames
- **Red, Green, Yellow LEDs** – Indicate sensor status
- **Buzzer** – Audio warning for detected threats
- **Resistors, Breadboard, Jumper wires**

Circuit Design

- MQ2 sensor connected to analog pin 36 of ESP32.
- Flame sensor connected to digital pin 34.
- LEDs connected to GPIOs: RED (25), GREEN (26), YELLOW (27).
- Buzzer connected to GPIO 33.
- Onboard LED at GPIO 2 used for system status.

Programming

The code was developed using **Arduino IDE**. Key highlights:

- Wi-Fi credentials and **Blynk Auth Token** were configured.
- **Blynk.virtualWrite** used to push sensor readings to the Blynk app.
- Threshold values used to determine presence of gas or flame.
- If flame is detected or gas level exceeds 700, the system activates:
    - **Buzzer**
    - **Red LED**
    - **Mobile notification**
- If safe, **Green** or **Yellow** LED is lit based on intermediate conditions.

IoT Integration (Blynk)

- Uses Blynk virtual pins **V1 (Smoke)**, **V2 (Flame)**, and **V3 (Alert)**.
- Real-time data and warning notifications appear in the Blynk app.

4. Problem Solving

Each team member played a crucial role in solving specific challenges during development:

- **Circuit design and stability** were managed by **Marjana Begum** and **Moumita Das Mou**, who overcame wiring noise issues by organizing connections cleanly on the breadboard.
- **Flame detection** configuration was handled by **Tahmid Samin**, who tackled false readings by fine-tuning the sensitivity and avoiding direct light interference.
- **Gas sensor tuning** was led by **Tanvir Mahmud Ove**, who calibrated threshold values and smoothed sensor data to ensure consistent gas detection.
- **Code development and Blynk integration** were done by **Shafat Alam**, who addressed Wi-Fi instability, optimized the sensor polling, and set up real-time communication with the Blynk app using BlynkTimer.

This collaborative effort resulted in a responsive and reliable fire and gas monitoring system.

5. Appendix

Hardware List

- ESP32 board
- MQ2 Gas Sensor
- Flame Sensor Module
- LEDs (Red, Green, Yellow)
- Buzzer
- Resistors (220Ω)
- Jumper wires
- Breadboard
- USB power or battery supply

Software & Tools

- Arduino IDE
- Blynk IoT App
- Blynk Cloud Platform

Reference Resources

- ESP32 Documentation
- Blynk Docs

Code

Controlling Servo Motor

```
#define BLYNK_TEMPLATE_ID "TMPL6TXV8POGb"
#define BLYNK_TEMPLATE_NAME "Gas and Flame Monitor"
#define BLYNK_AUTH_TOKEN "VubJ5G_tlAhG8NhLf4PjKgzq2eTksgKn"

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

BlynkTimer timer;

// Wi-Fi Credentials
char ssid[] = "Shafat";
char pass[] = "shafat22";

// GPIO Pins
#define MQ2_SENSOR       36  // AO of MQ2 smoke sensor
#define FLAME_SENSOR_D   34  // D0 of Flame sensor
#define RED_LED          25
#define GREEN_LED        26
#define BUZZER_PIN       33
#define ONBOARD_LED      2
#define YELLOW_LED       27

// Blynk Virtual Pins
#define VPIN_SMOKE V1
#define VPIN_FLAME V2
#define VPIN_ALERT V3

int smokeValue = 0;
int flameValue = 1;  // Digital: 0 = detected
bool buzzerOn = false;
bool wifiConnected = false;

unsigned long lastReconnectAttempt = 0;
const unsigned long reconnectInterval = 10000; // every 10s
```

Controlling Servo Motor

```
void readSensors() {
  smokeValue = map(analogRead(MQ2_SENSOR), 0, 4095, 0, 100);
  flameValue = digitalRead(FLAME_SENSOR_D);  // LOW = flame

  bool flameDetected = (flameValue == LOW);
  bool smokeDetected = (smokeValue > 20); // threshold adjustable

  // Debug print
  Serial.print("Smoke: ");
  Serial.print(smokeValue);
  Serial.print("% | Flame: ");
  Serial.println(flameDetected ? "YES" : "NO");

  // LED indicators
  digitalWrite(RED_LED, smokeDetected ? HIGH : LOW);
  digitalWrite(GREEN_LED, flameDetected ? HIGH : LOW);

  // Buzzer control
  if (smokeDetected || flameDetected) {
    if (!buzzerOn) {
      tone(BUZZER_PIN, 2000);
      buzzerOn = true;
    }
    digitalWrite(ONBOARD_LED, HIGH);  // Turn on onboard LED when alert
  } else {
    if (buzzerOn) {
      noTone(BUZZER_PIN);
      buzzerOn = false;
    }
    digitalWrite(ONBOARD_LED, LOW);  // Turn off onboard LED when safe
  }

  // Blynk updates
  if (wifiConnected) {
    Blynk.virtualWrite(VPIN_SMOKE, smokeValue);
    Blynk.virtualWrite(VPIN_FLAME, flameDetected ? "🔥 Flame Detected" : "✅ No Flame");

    if (smokeDetected && flameDetected) {
      Blynk.virtualWrite(VPIN_ALERT, "🔥 Smoke + Flame Detected!");
      Blynk.logEvent("gas_and_flame", "🔥 Gas + Flame Detected!");
    } else if (smokeDetected) {
      Blynk.virtualWrite(VPIN_ALERT, "💨 Smoke Detected");
      Blynk.logEvent("gas", "Smoke Detected!");
    } else if (flameDetected) {
      Blynk.virtualWrite(VPIN_ALERT, "🔥 Flame Detected");
      Blynk.logEvent("flame", "Flame Detected!");
    } else {
      Blynk.virtualWrite(VPIN_ALERT, "✅ Safe");
    }
  }
}
```

Controlling Servo Motor

```
void setup() {
  Serial.begin(115200);

  pinMode(MQ2_SENSOR, INPUT);
  pinMode(FLAME_SENSOR_D, INPUT);
  pinMode(RED_LED, OUTPUT);
  pinMode(GREEN_LED, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(ONBOARD_LED, OUTPUT);
  pinMode(YELLOW_LED, OUTPUT);
  noTone(BUZZER_PIN);

  digitalWrite(RED_LED, LOW);
  digitalWrite(GREEN_LED, LOW);
  digitalWrite(YELLOW_LED, LOW);
  digitalWrite(ONBOARD_LED, LOW);

  // Initial Wi-Fi connection
  Serial.println("📶 Connecting to Wi-Fi ...");
  WiFi.begin(ssid, pass);

  int attempts = 0;
  while (WiFi.status() ≠ WL_CONNECTED && attempts < 40) {
    Serial.print(".");
    delay(200);
    attempts++;
  }

  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("\n✅ Wi-Fi Connected!");
    Serial.print("📡 IP: ");
    Serial.println(WiFi.localIP());

    wifiConnected = true;
    digitalWrite(YELLOW_LED, HIGH);     // Turn on yellow when connected
    digitalWrite(ONBOARD_LED, HIGH);    // Blue LED steady = connected
    Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
    timer.setInterval(2000L, readSensors);
  } else {
    Serial.println("\n❌ Wi-Fi failed. Running in offline mode.");
    wifiConnected = false;
    timer.setInterval(2000L, readSensors);
  }
}
```

Controlling Servo Motor

```
void loop() {
  if (wifiConnected && WiFi.status() == WL_CONNECTED) {
    Blynk.run();
    timer.run();
  } else {
    // Retry Wi-Fi every 10s
    unsigned long now = millis();
    if (now - lastReconnectAttempt > reconnectInterval) {
      Serial.println("🔄 Reconnecting Wi-Fi ... ");
      WiFi.begin(ssid, pass);
      lastReconnectAttempt = now;

      if (WiFi.status() == WL_CONNECTED) {
        Serial.println("✅ Reconnected to Wi-Fi!");
        digitalWrite(YELLOW_LED, HIGH);
        digitalWrite(ONBOARD_LED, HIGH);
        wifiConnected = true;
        Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
      }
    }

    readSensors();  // Local monitoring even without Wi-Fi
  }
}
```

6. Final Remarks

This project demonstrates how embedded systems and IoT can be practically applied to build life-saving technology. The Fire and Gas Detector System provides real-time safety alerts, combining local and remote response mechanisms, and is a step toward smarter, safer living environments.