**Smart Weather Monitoring System using ESP32 and Blynk Cloud**

# Objective

To build a smart weather monitoring system using ESP32 that senses temperature, humidity, hydrogen gas concentration, and rain levels. The system displays the data on an I2C LCD, sends real-time values to Blynk Cloud, and provides alerts via buzzer, LED, and mobile notification in case of dangerous gas levels.  
  
**Background**

With the growing need for environmental monitoring, integrating IoT-based systems for weather tracking has become essential. Traditional weather systems often lack real-time alerts and remote accessibility. This project addresses these gaps by combining sensor data collection with cloud integration using ESP32 and Blynk, making it ideal for smart environments and safety monitoring.

**Methodology**

The methodology followed includes hardware interfacing of multiple sensors with ESP32, coding the logic in Arduino IDE, and connecting to the Blynk Cloud for real-time monitoring. Sensor data is read periodically, processed, displayed on an LCD, and transmitted to the mobile application. Alerts are generated based on threshold values set in the code.  
  
**Tools and Techniques Used**

* ESP32 Microcontroller
* Arduino IDE for programming
* Blynk Cloud and Mobile App for IoT integration
* DHT11 for temperature and humidity sensing
* MQ-8 Hydrogen Gas Sensor
* Analog Rain Sensor
* I2C LCD Display
* Serial Monitor for debugging
* Wi-Fi for real-time data transfer

### ****Scope****

This project is suitable for educational demonstrations, smart home weather monitoring, and basic safety systems in small industrial or residential areas. It showcases real-time IoT applications and builds a strong foundation in sensor integration and cloud-based monitoring.

### ****Significance****

The project demonstrates the application of embedded systems and IoT in solving real-world problems. It enables users to monitor environmental conditions remotely and receive immediate alerts, thus contributing to both safety and awareness in potentially hazardous situations.

# Components Used

1. ESP32 Development Board (with Type-C or Micro-USB cable)
2. DHT11 Temperature and Humidity Sensor
3. MQ-8 Hydrogen Gas Sensor
4. Rain Sensor Module
5. I2C 16x2 LCD Display
6. Buzzer
7. LED
8. Jumper Wires
9. Breadboard  
   10.Power Bank (optional for portability)

**Wiring Details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Pin on Component** | **Connect to ESP32 GPIO** | **Notes** |
| DHT11 | VCC | 3.3V | Power |
| GND | GND | Ground |
| DATA | GPIO 4 | Data output |
| MQ-8 Gas Sensor | VCC | 5V | Requires 5V |
| GND | GND | Ground |
| AO | GPIO 35 | Analog gas reading |
| Rain Sensor | VCC | 3.3V or 5V | Power supply |
| GND | GND | Ground |
| AO | GPIO 34 | Analog rain detection |
| Buzzer | + (VCC) | GPIO 14 | Buzzer control |
| - (GND) | GND | Ground |
| LED | Anode (+) | GPIO 26 | Use 220Ω resistor in series |
| Cathode (-) | GND | Ground |
| I2C LCD Display | VCC | 5V | Power |
| GND | GND | Ground |
| SDA | GPIO 21 | I2C data |
| SCL | GPIO 22 | I2C clock |

**Working Logic**

- DHT11 provides temperature and humidity readings.  
- MQ-8 measures hydrogen gas levels. If value exceeds 2500, buzzer and LED are activated, and a Blynk alert is sent.  
- Rain Sensor gives analog value for moisture detection. Lower values = higher rainfall.  
- LCD shows real-time environmental data.  
- Blynk Cloud is used to send real-time sensor data to the mobile app and trigger gas alerts.  
  
  
**Description of the Process**

The ESP32 reads data from DHT11, MQ-8, and the rain sensor every 2 seconds. It sends this data to the Blynk cloud using Wi-Fi. Users can view the data on the Blynk mobile app through virtual pins (V1 to V4). The LCD continuously displays all sensor values in real time. If hydrogen gas exceeds a safe threshold, a buzzer and LED are activated, and a push notification is sent to the user via Blynk.

# Code Summary (Blynk Enabled)

**Code:-**   
#define BLYNK\_TEMPLATE\_ID "TMPL37diIJvAO"  
#define BLYNK\_TEMPLATE\_NAME "Smart Weather Monitoring System"  
#define BLYNK\_AUTH\_TOKEN " ps6xU0w293ccSUMKwWUJWAEtwB0A2ihj"  
  
#include <WiFi.h>  
#include <BlynkSimpleEsp32.h>  
#include "DHT.h"  
#include <Wire.h>  
#include <LiquidCrystal\_I2C.h>  
  
char auth[] = BLYNK\_AUTH\_TOKEN;  
char ssid[] = "Your\_SSID";  
char pass[] = "Your\_PASSWORD";  
  
#define DHTPIN 4  
#define DHTTYPE DHT11  
#define MQ8PIN 35  
#define RAINPIN 34  
#define BUZZER 14  
#define LED 26  
  
DHT dht(DHTPIN, DHTTYPE);  
LiquidCrystal\_I2C lcd(0x27, 16, 2);  
bool alertSent = false;

void setup() {  
 Serial.begin(115200);  
 Blynk.begin(auth, ssid, pass);  
 Wire.begin(21, 22);  
 lcd.init();  
 lcd.backlight();  
 dht.begin();  
 pinMode(BUZZER, OUTPUT);  
 pinMode(LED, OUTPUT);  
  
 lcd.setCursor(0, 0);  
 lcd.print("Weather Monitor");  
 delay(2000);  
 lcd.clear();  
}  
  
void loop() {  
 Blynk.run();  
 static unsigned long lastUpdate = 0;  
 if (millis() - lastUpdate >= 2000) {  
 lastUpdate = millis();  
  
 float temp = dht.readTemperature();  
 float hum = dht.readHumidity();  
 int gasValue = analogRead(MQ8PIN);  
 int rainValue = analogRead(RAINPIN);  
  
 if (!isnan(temp) && !isnan(hum)) {  
 Blynk.virtualWrite(V1, temp);  
 Blynk.virtualWrite(V2, hum);  
 Blynk.virtualWrite(V3, gasValue);  
 Blynk.virtualWrite(V4, rainValue);  
 }  
  
 lcd.setCursor(0, 0);  
 lcd.print("T:"); lcd.print(temp, 1);  
 lcd.print(" H:"); lcd.print(hum, 1);  
 lcd.setCursor(0, 1);  
 lcd.print("G:"); lcd.print(gasValue);  
 lcd.print(" R:"); lcd.print(rainValue);  
  
 if (gasValue > 2500 && !alertSent) {  
 digitalWrite(BUZZER, HIGH);  
 digitalWrite(LED, HIGH);  
 Blynk.logEvent("gas\_alert", "High gas level detected!");  
 alertSent = true;  
 } else if (gasValue <= 2500) {  
 digitalWrite(BUZZER, LOW);  
 digitalWrite(LED, LOW);  
 alertSent = false;  
 }  
 }  
}

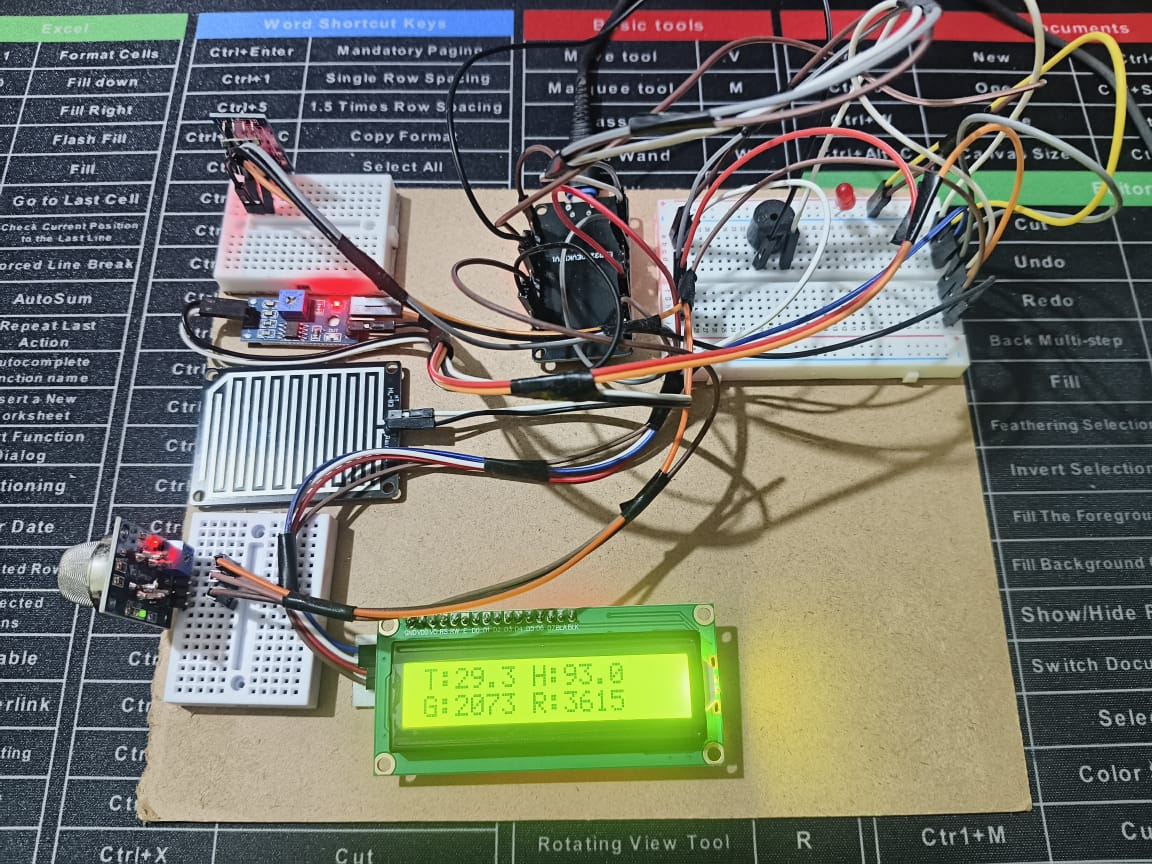
# Thresholds

- Gas Sensor (MQ-8): Value > 2500 triggers buzzer, LED, and Blynk notification.  
- Rain Sensor: Dry ≈ 1023, Wet ≈ 300 (display use only).

# LCD Output Format

Line 1: T:<temperature> H:<humidity>  
Line 2: G:<gas\_value> R:<rain\_value>

# Output on LCD

Example:  
 T:28.3 H:67.1  
 G:3120 R:1023  
  
Project Prototype (Original Image)  
  
 **Project Image**

A screenshot of a phone

AI-generated content may be incorrect.

**Blynk IOT Mobile Dashboard**

A screenshot of a computer

AI-generated content may be incorrect.

**Blynk IOT Web Dashboard**

# Notes

1. Ensure all GND pins share a common rail.  
2. Use an I2C Scanner to confirm your LCD address.  
3. Replace WiFi and Blynk credentials as needed.  
4. Use resistors where needed (especially for the LED).  
5. Use Blynk App to monitor sensor values on virtual pins V1-V4.  
6. The Blynk cloud offers seamless real-time data sync.  
7. LCD output helps with local monitoring.  
8. MQ-8 sensor responds effectively to hydrogen concentration.  
9 .Rain sensor values vary smoothly with moisture levels.  
10. The system is reliable for small-scale weather monitoring setups.  
  
Result

The system monitors environmental data in real-time, displays readings on LCD, updates values on Blynk Cloud, and provides audio-visual and mobile alerts when gas levels are hazardous. This makes it ideal for IoT-based weather stations and smart safety systems.  
The system successfully monitored all environmental parameters and displayed them accurately on both the LCD and the Blynk app. The gas alert mechanism worked reliably, triggering both local (buzzer and LED) and remote (app-based) alerts when unsafe levels were detected. All sensors functioned as expected.  
  
**Future Directions**

* Integrate solar power for off-grid deployments.
* Add more sensors like wind speed, UV index, or air quality (PM2.5).
* Use local data logging via SD card or EEPROM.
* Implement a dashboard on Blynk for data trends and history.
* Deploy multiple units for a distributed monitoring network.  
   **Recommendations**
* Ensure strong and stable Wi-Fi for uninterrupted data updates.
* Calibrate sensors periodically for accurate readings.
* Use weatherproof casing for outdoor use.
* Use resistors and power regulation circuits to protect components.

**Summary**  
The Smart Weather Monitoring System is an IoT-based project using the ESP32 microcontroller to measure environmental conditions like temperature, humidity, hydrogen gas levels, and rainfall. It uses DHT11, MQ-8, and a rain sensor to collect data, which is displayed on a 16x2 I2C LCD. The system connects to the Blynk Cloud to send real-time sensor values to a mobile app. If the gas level crosses a safe limit, it activates a buzzer, LED, and sends a push notification through Blynk. This project helps in remote weather tracking and ensures safety in hazardous environments.