Microprocessor and Embedded System Project Report
Sylhet Engineering College

School of Applied Science and Technology
Shahjalal University of Science and Technology

Course code: EEE-704
Project: IoT Based Data Logger

Project Submitted By

Ajijul Haque Sakib Opu (2019338501)
Ashraf Ahmed (2019338502)
Amit Kanti Ray Badhon (2019338503)
Amit Das Raj (2019338504)
Safwan Bin Waris (2019338505)

BSc in Electrical and Electronics Engineering
4th year 1st semester
Session: 2019-2020
Sylhet Engineering College

Project Submitted To

Md. Janibul Alam Soeb
Assistant Professor

Faculty of Agricultural Engineering and Technology
Sylhet Agricultural University
Sylhet -3100, Bangladesh

Project title:

To build an IoT based data logger for measuring environment parameters.

Required Component:

1) ESP32
2) MQ-135
3) DHT11
4) TEMT6000
5) BMP180
6) Vero Board and jumper cables
7) Power Supply

Working Principle:

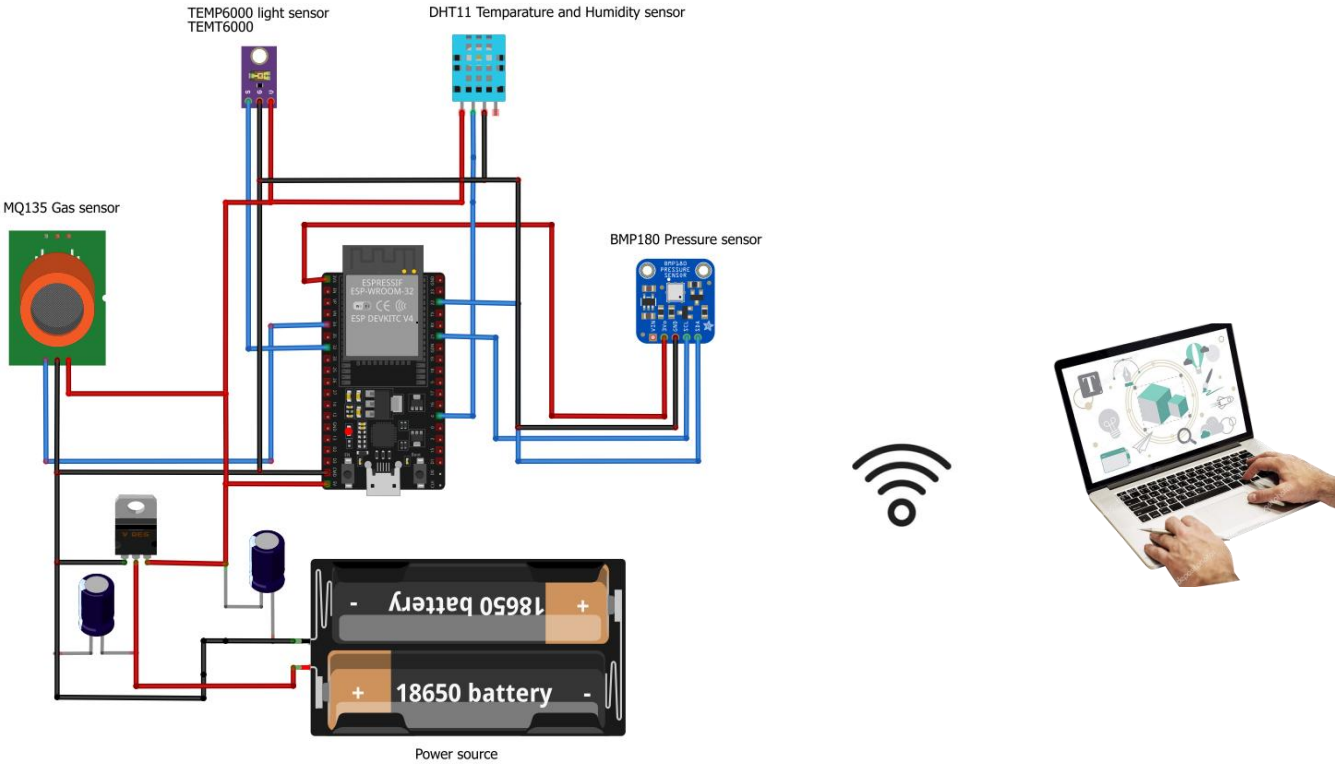
The IoT-based data logger employs an ESP32 Dev-Kit microcontroller interfaced with multiple sensors for real-time environmental data acquisition and transmission. The system components include the MQ-135 gas sensor, DHT11 temperature and humidity sensor, TEMT6000 light sensor, BMP180 pressure sensor, a Vero board with jumper cables for connectivity, and a stable power supply.

Sensor data is collected periodically, with the MQ-135 monitoring air quality, the DHT11 measuring temperature and humidity, the TEMT6000 recording ambient light intensity, and the BMP180 tracking atmospheric pressure and altitude. These sensors generate analog or digital outputs, which are processed by the ESP32.

The ESP32's onboard Wi-Fi module enables seamless data transmission. A pre-configured cloud-based script (Google Apps Script) facilitates data upload to Google Sheets every 5 seconds, ensuring continuous and synchronized logging. The Vero board organizes connections, enhancing system reliability.

This implementation offers a robust, cost-effective solution for environmental monitoring with real-time data storage accessible remotely for analysis.

Circuit Diagram:



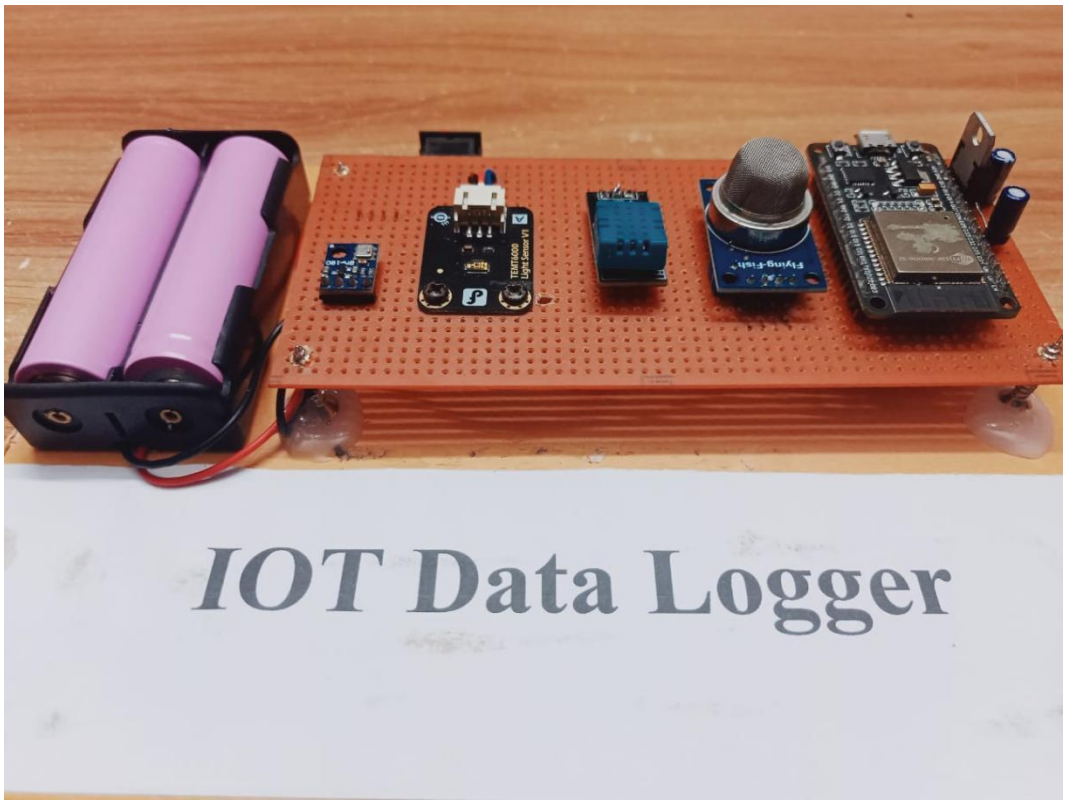
Firmware:

```
#include <WiFi.h>
#include <HTTPClient.h>
#include "DHT.h"

// Wi-Fi credentials
const char* ssid = "ROOM302";
const char* password = "ROOM_605";

// Google Apps Script Web App URL
const char* serverName = "https://script.google.com/macros/s/AKfycbztrrsGnEYjJ3V0-
pjrA31lWgU_PAFGxP5CVWihdzuRJzX3XwobBgLGGW0UyBKSGowlBg/exec";

// Timer variables
unsigned long lastTime = 0;
unsigned long timerDelay = 10000;  // Send data every 10 seconds

// DHT11 settings
#define DHTPIN 4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);



// MQ135 settings
#define MQ135PIN 34

// TEMT6000 settings
#define TEMTPIN 32

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

  // Initialize the DHT sensor
  dht.begin();

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  Serial.print("Connecting to Wi-Fi...");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("\nConnected to Wi-Fi");
```

```
}

void loop() {
  if ((millis() - lastTime) > timerDelay) {
    // Read DHT11 data
    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();

    // Read MQ135 data
    int gasValue = analogRead(MQ135PIN);

    // Read TEMT6000 data
    int lightValue = analogRead(TEMTPIN);

    // Check if DHT readings failed
    if (isnan(temperature) || isnan(humidity)) {
      Serial.println("Failed to read from DHT sensor!");
      return;
    }



    // Log sensor readings to Serial Monitor
    Serial.println("Sensor Readings:");
    Serial.printf("Temperature: %.2f °C, Humidity: %.2f %%\n", temperature, humidity);
    Serial.printf("Gas: %d, Light: %d\n", gasValue, lightValue);

    // Send data to Google Sheet if connected to Wi-Fi
    if (WiFi.status() == WL_CONNECTED) {
      HTTPClient http;

      // Prepare the URL with sensor data
      String url = String(serverName) +
                   "?temperature=" + String(temperature) +
                   "&humidity=" + String(humidity) +
                   "&gas=" + String(gasValue) +
                   "&light=" + String(lightValue);

      // Send HTTP GET request
      http.begin(url.c_str());
      int httpResponseCode = http.GET();

      if (httpResponseCode > 0) {
        Serial.println("Data sent successfully!");
        String response = http.getString();
        Serial.println(response);
```

```
      } else {
        Serial.print("Error sending data: ");
        Serial.println(httpResponseCode);
      }

      // End HTTP connection
      http.end();
    } else {
      Serial.println("Wi-Fi not connected");
    }

    // Update the timer
    lastTime = millis();
  }
}
```

Result:



IoT Data Logger

File Edit View Insert Format Data Tools Extensions Help

H17

	A	B	C	D	E	F	G	H
1	Date & Time	Temperatue (C)	Humidity (%)	GAS	Light (lux)	Pressure (hPa)		
2	19/11/2024 19:08:38	27.1	69	0	37	1011		
3	19/11/2024 19:08:52	27.1	69	0	0	1009		
4	19/11/2024 19:09:07	27.8	95	0	15	1010		
5	19/11/2024 19:09:21	27.8	95	0	16	1011		
6	19/11/2024 19:09:40	27.5	83	0	10	1011		
7	19/11/2024 19:09:58	27.5	78	0	13	1011		
8	19/11/2024 19:10:12	27.5	73	0	13	1011		
9	19/11/2024 19:10:25	26.8	72	0	15	1011		
10	19/11/2024 19:10:42	26.8	71	0	16	1011		
11	19/11/2024 19:10:58	27.5	71	0	16	1011		
12	19/11/2024 19:11:11	26.8	70	0	16	1011		
13	19/11/2024 19:11:24	26.2	71	0	0	1011		
14	19/11/2024 19:11:37	26.2	71	600	6	1009		
15	19/11/2024 19:11:51	26.2	71	430	13	1011		
16	19/11/2024 19:12:04	26.2	70	300	16	1011		
17	19/11/2024 19:12:19	26.2	70	200	16	1011		
18	19/11/2024 19:12:36	25.9	71	0	13	1011		
19	19/11/2024 19:12:50	25.9	71	0	1145	1010		
20	19/11/2024 19:13:04	26.2	71	0	3980	1011		
21	19/11/2024 19:13:18	25.6	71	0	150	1011		
22	19/11/2024 19:13:34	25.6	71	0	11	1011		
23	19/11/2024 19:13:47	25.6	72	0	14	1010		
24								
25								
26								
27								
28								
29								
30								
31								