Name: _____ | Class: D16 | Roll No: __

EXPERIMENT NO: 02

Aim: To interface temperature and humidity sensor with Node MCU (ESP8266) and send the data to ThingSpeakto obtain output on serial monitor with timestamp.

Apparatus: NODE MCU, Temperature and Humidity Sensor (DHT11), Micro USB Cable, Breadboard, Female to Female connectors, ThingSpeak website(https://thingspeak.mathworks.com/) for analysis

INTRODUCTION:

It is a wide requirement across the globe to obtain the information regarding a specific unit which monitors different parameters even if the users are miles apart. This necessity is facilitated by the features of ThingSpeak. ThingSpeak allows the user to obtain the information from a particular microcontroller like Node MCU which may be connected to one or more sensors and then upload the data to the cloud with a graphical representation so that the user can not only obtain the data but also analyze the changes occurring in it.

Theory:

THINGSPEAK

- ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze livedata streams in the cloud.
- One can send data to ThingSpeak from your devices, create instant visualizations of live data, and send alerts using web services like Twitter and Twilio..
- With MATLAB analytics inside ThingSpeak, you can write and execute MATLAB code to perform preprocessing, visualizations, and analyses.
- ThingSpeak enables engineers and scientists to prototype and build IoT systems without setting up a server for developing web software.

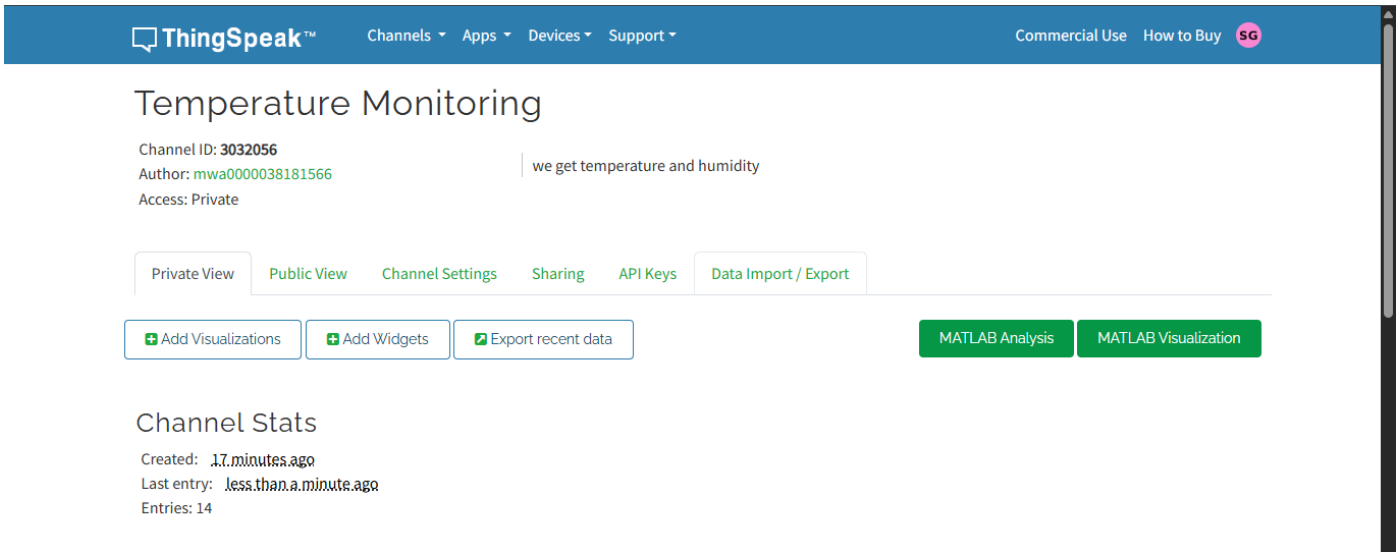

Fig. (1) Dashboard of Thingspeak for Temperature Monitoring

GETTING API KEY:

- Go to https://thingspeak.com/ and create an account.
- Create a new channel by clicking on the button.
- Enter basic details of the channel and then scroll down and save the channel
- Channel Id is the identity of your channel. Note down this. Then go to API keys, copy and paste this key in code.

PROCEDURE:

1. Interface the sensor (DHT11) with ESP8266
2. Upload the code directly to the ESP8266 Module. This means that the existing code or firmware will be erased
3. Before uploading the code, connect the GPIO0 to GND and RESET the ESP Module to enable ProgrammingMode Also, select "Generic ESP8266 Module" in the Boards section of the Arduino IDE Make sure that the correct COM PORT is selected
4. Now, set the baud rate to 9600 in the serial monitor.
5. First, it will get connected to the WiFi Network. Then, immediately, it will try to read the data from the DHT11 Humidity sensor and calculate the Temperature and Humidity values based on that data
6. After this, the values of temperature and humidity will be uploaded to the ThingSpeak API
7. If one opens their channel in the ThingSpeak, they can see the chart associated with the values from DHT11 Sensor.

CODE:

```
#include <ESP8266WiFi.h>
#include "DHT.h"

// ====== DHT Sensor Setup ======
#define DHTPIN D4 // Connect your DHT11 data pin to D4
#define DHTTYPE DHT11 // DHT11 sensor
DHT dht(DHTPIN, DHTTYPE);

// ====== WiFi and ThingSpeak Setup ======
const char* ssid = "Shreyash"; // Replace with your WiFi SSID
const char* password = "shreyash"; // Replace with your WiFi password
const char* host = "api.thingspeak.com";
const char* apiKey = "HF9RVHBJCB7XD6IC"; // Replace with your ThingSpeak Write API Key

void setup() {
  Serial.begin(115200);
  delay(10);
  dht.begin();

  Serial.println("Connecting to WiFi...");
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("\nWiFi connected");
}

void loop() {
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature(); // Celsius

  // Check if readings are valid
  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
    delay(2000);
```

```
    return;
}
  Serial.println("Sending data to ThingSpeak...");

  WiFiClient client;
  const int httpPort = 80;

  if (!client.connect(host, httpPort)) {
    Serial.println("Connection to ThingSpeak failed");
    delay(2000);
    return;
  }

  // Create ThingSpeak URL
  String url = String("/update?api_key=") + apiKey +
               "&field1=" + String(temperature) +
               "&field2=" + String(humidity);

  // Send HTTP GET request
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");

  Serial.println("Data sent: Temp = " + String(temperature) + " *C, Humidity = " + String(humidity) + " %");
  delay(15000); // ThingSpeak accepts updates every 15 seconds
}
```

OUTPUT:

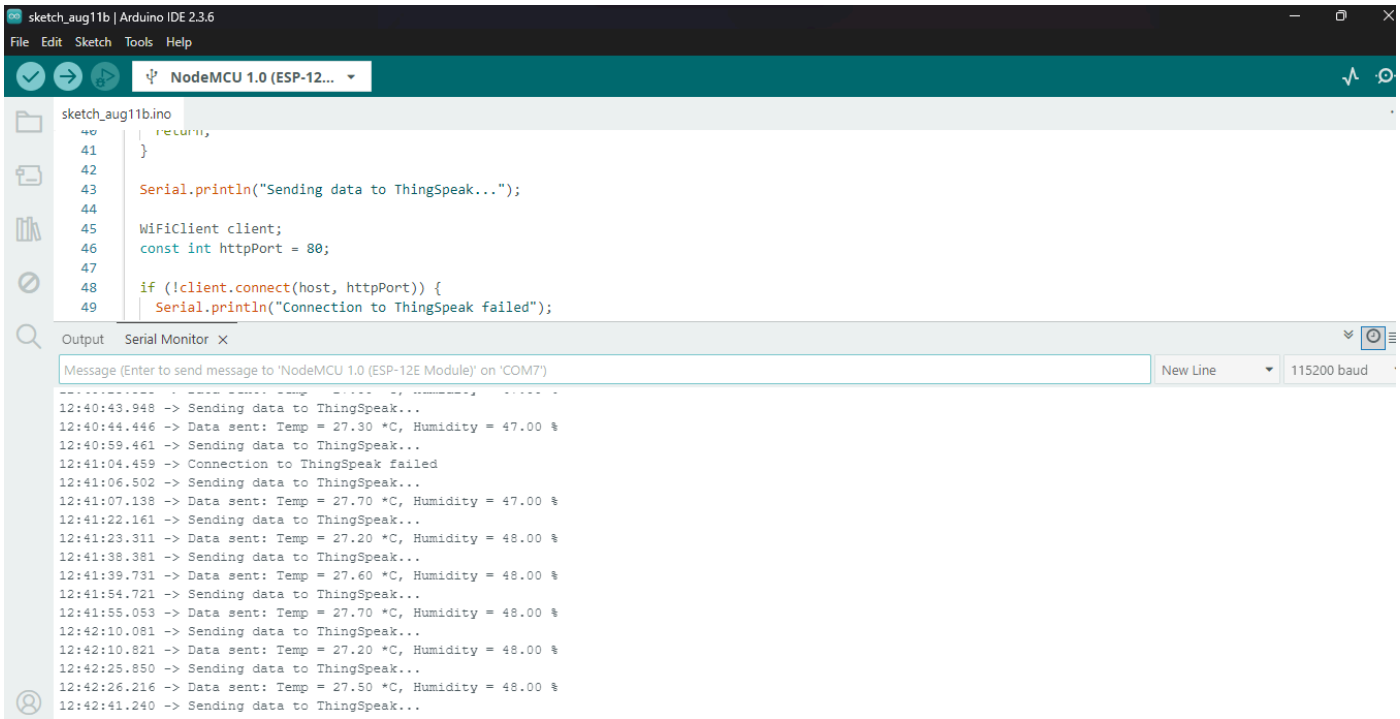

Fig. (2) Serial Monitor showing Temperature and humidity and sends data to Thingspeak

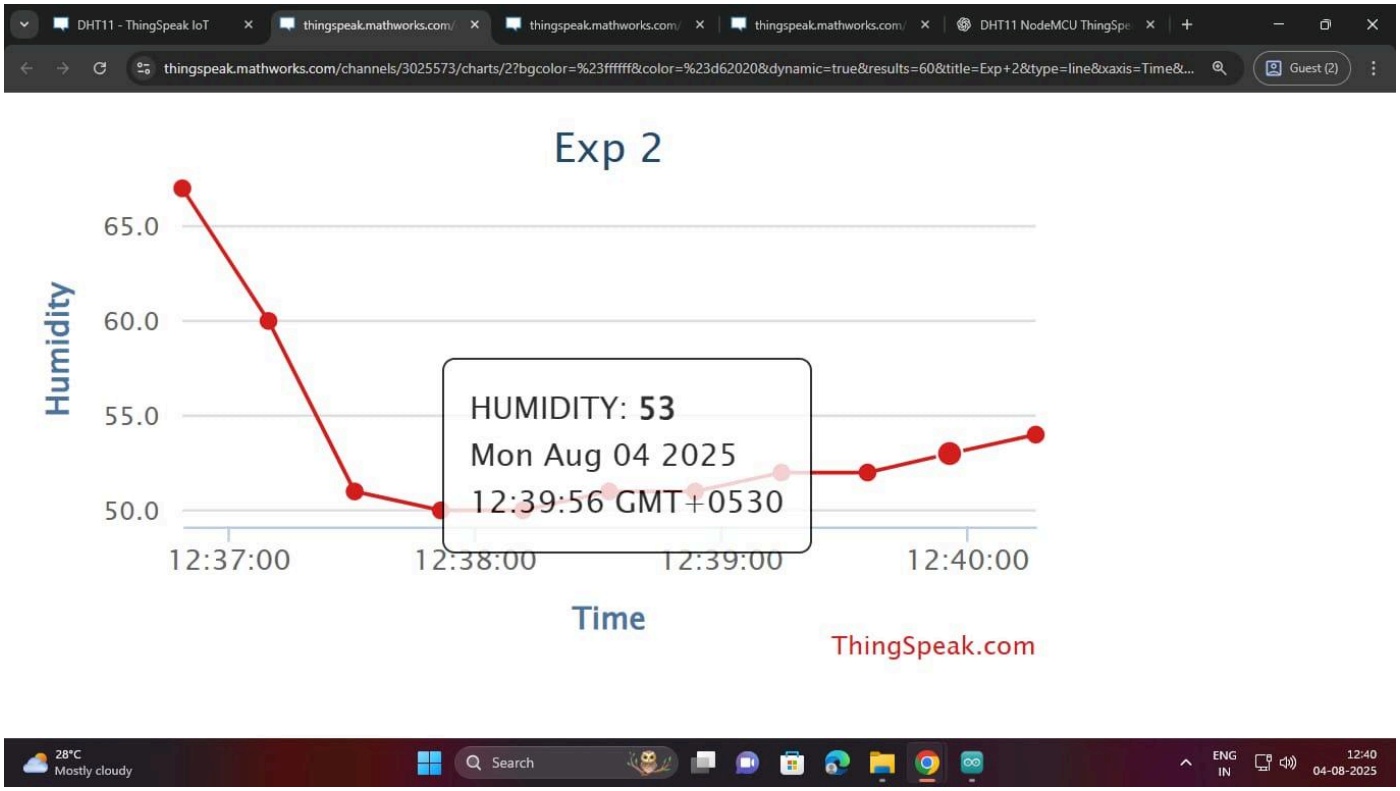

Fig. (4) Humidity Graph in Thingspeak with Timestamp

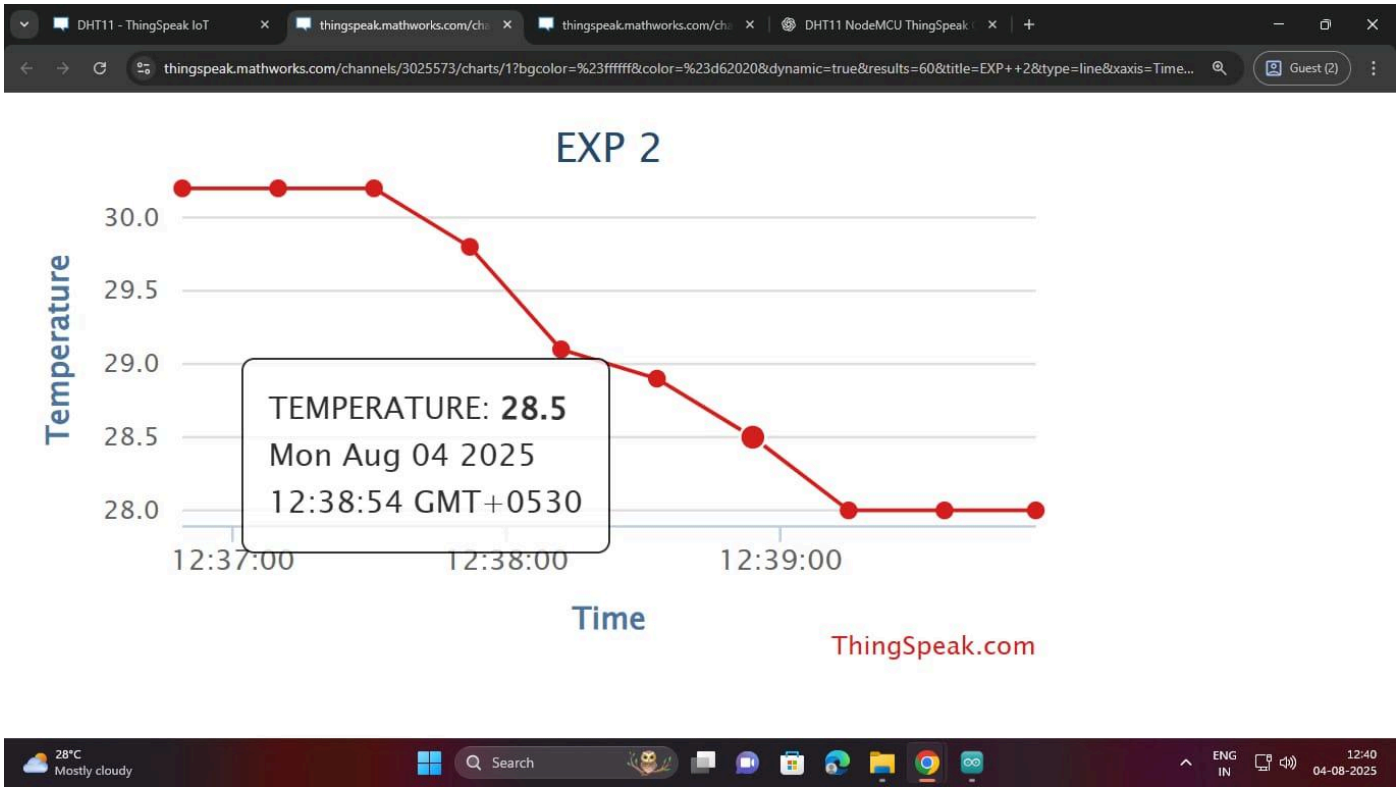

Fig. (5) Temperature Graph in Thingspeak with Timestamp

CONCLUSION:

1. Gained understanding of making POST requests.
2. Learned about ThingSpeak.
3. Discovered the application of ESP8266 in small-scale IoT projects.
4. Comprehended the use of API keys for security.
5. Recognized the significance of selecting an appropriate delay between successive requests.