SENDING SENSOR DATA TO THE CLOUD

- **Objectives:**

1. To create a system that measures temperature using a sensor (DS18B20).
2. To upload the temperature data to ThingSpeak for real-time monitoring and analysis.
3. To visualize the temperature data on the ThingSpeak dashboard.
4. To gain practical knowledge of Internet of Things (IoT) technologies, including sensors, microcontrollers, and cloud platforms.

- **Outcomes:**

1. **Functional Temperature Monitor**: A working prototype that successfully measures and displays temperature.
2. **Data Logging:** Temperature readings are logged in real-time on the ThingSpeak cloud platform.
3. **Dashboard Visualization:** Ability to visualize and analyze temperature data using ThingSpeak charts and graphs.
4. **IoT Understanding:** Improved understanding of IoT concepts, data transmission, and cloud computing.

- **Hardware Requirements:**

1. **ESP32**: Microcontroller to connect to the Wi-Fi network and send data to ThingSpeak.
2. **Temperature Sensor (DS18B20)**: For measuring temperature.
3. **Breadboard and Jumper Wires**: For making connections.
4. **Power Supply**: USB power supply or battery for the ESP32/ESP8266.
5. **Computer**: To upload code and monitor data via the Serial Monitor.

➢ Software Requirements:

1. Arduino IDE
2. Wokwi Online Simulator
3. ThingSpeak Account
    a. Arduino Libraries:
    b. DallasTemperature
    c. OneWire
    d. WiFi

➢ Theory:

1. Overview of IoT and Its Components

The Internet of Things (IoT) refers to a network of interconnected devices that communicate and exchange data over the internet. IoT devices typically consist of:

- **Sensors**: Devices that collect data from the environment (e.g., temperature, humidity, light).
- **Microcontrollers**: Small computers that process the sensor data and control other components (e.g., ESP32, ESP8266).
- **Communication Modules**: Components that enable devices to connect to the internet (e.g., Wi-Fi, LoRa, Zigbee).
- **Cloud Platforms**: Services that store, analyze, and visualize data collected from IoT devices (e.g., ThingSpeak, AWS IoT).

2. Working Principle of Temperature Measurement

In this project, a temperature sensor (like the DS18B20) is used to measure the ambient temperature. The working principle involves:

- **Sensing Element**: The DS18B20 sensor uses a thermistor, which changes resistance based on temperature. This change is measured and converted into a digital signal.
- **Data Conversion**: The microcontroller (ESP32/ESP8266) reads the digital signal from the sensor, converting it into a temperature value, usually in degrees Celsius.

3. Microcontroller Functionality

The ESP32/ESP8266 microcontroller plays a critical role in the project by:

- **Initialization**: Setting up the microcontroller, including configuring the Wi-Fi connection.
- **Data Acquisition**: Continuously reading temperature data from the sensor at defined intervals.
- **Data Processing**: Processing the temperature data for transmission, ensuring it is formatted correctly for the cloud.

4. Communication with ThingSpeak

The process of sending data from the microcontroller to ThingSpeak involves several steps:

- **Wi-Fi Connection**: The microcontroller connects to a Wi-Fi network using the provided SSID and password.
- **HTTP POST Request**: Once connected, the microcontroller sends an HTTP POST request to the ThingSpeak server. This request contains:
    - **Channel ID**: Identifies the specific channel in ThingSpeak where the data will be stored.
    - **Write API Key**: An authentication token that allows the microcontroller to write data to the specified channel.
    - **Data Fields**: The temperature value, usually formatted as a URL-encoded string.

5. Data Storage and Visualization in ThingSpeak

Upon receiving the data, ThingSpeak stores it in real-time. The key processes include:

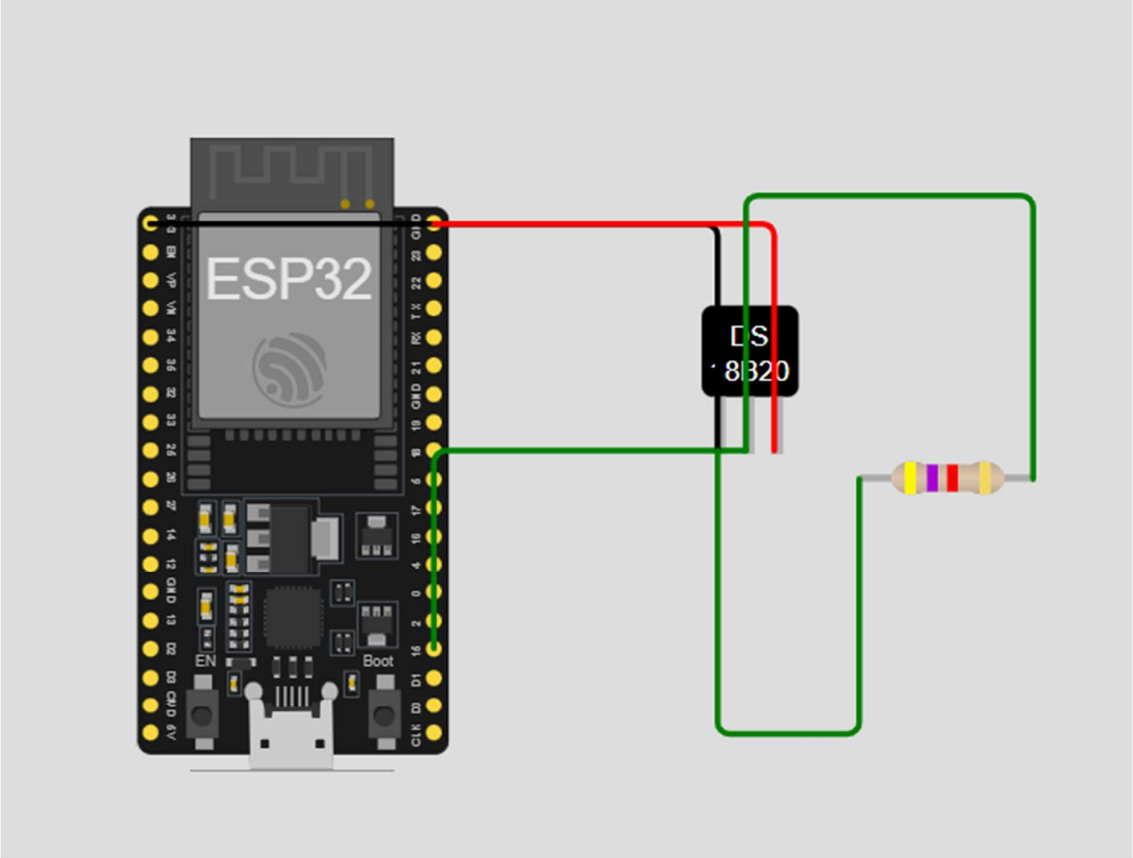
- **Data Logging**: Each data point is recorded with a timestamp, allowing users to track changes over time.
- **Visualization**: Users can create various visual representations of the data, such as:
    - **Line Charts**: To show temperature trends over time.
    - **Bar Charts**: To compare different readings.
    - **Grid Views**: For a comprehensive overview of multiple data points.

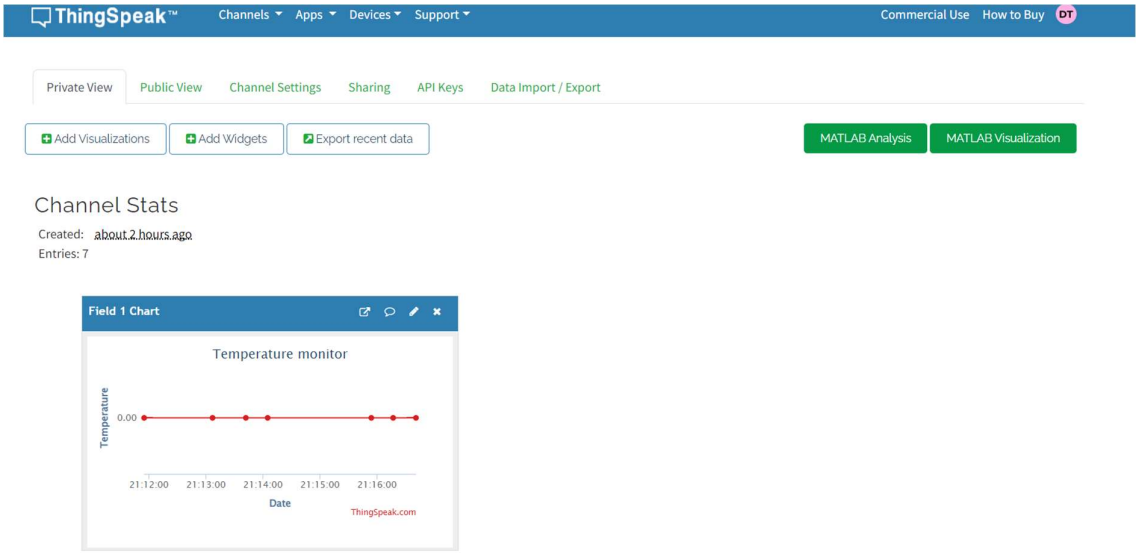
6. Data Analysis and Further Applications

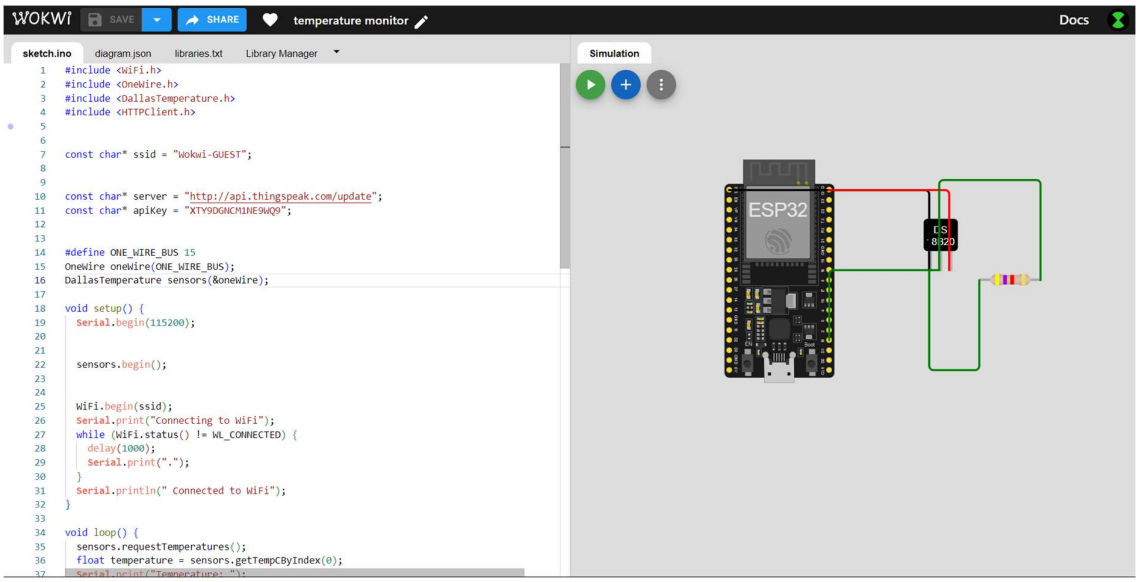
ThingSpeak allows users to analyze the collected data using MATLAB or other analytical tools. Potential applications include:

- **Alerts and Notifications**: Set up triggers to send alerts if the temperature exceeds a certain threshold.
- **Historical Analysis**: Examine historical data to identify trends or anomalies.
- **Integration with Other IoT Devices**: Combine temperature data with other environmental sensors for comprehensive monitoring (e.g., humidity, air quality).

➢ Output:


ESP32
DS
18B20
EN
Boot


ThingSpeak™
Channels
Apps
Devices
Support
Commercial Use
How to Buy
Private View
Public View
Channel Settings
Sharing
API Keys
Data Import / Export
Add Visualizations
Add Widgets
Export recent data
MATLAB Analysis
MATLAB Visualization
Channel Stats
Created: about 2 hours ago
Entries: 7
Field 1 Chart
Temperature monitor
Temperature
0.00
21:12:00
21:13:00
21:14:00
21:15:00
21:16:00
Date
ThingSpeak.com



Program :

```
#include <WiFi.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <HTTPClient.h>


const char* ssid = "Wokwi-GUEST";


const char* server = "http://api.thingspeak.com/update";
const char* apiKey = "XTY9DGNCM1NE9WQ9";


#define ONE_WIRE_BUS 15
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

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


  sensors.begin();


  WiFi.begin(ssid);
  Serial.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
```

```
    Serial.print(".");
  }
  Serial.println(" Connected to WiFi");
}

void loop() {
  sensors.requestTemperatures();
  float temperature = sensors.getTempCByIndex(0);
  Serial.print("Temperature: ");
  Serial.println(temperature);


  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String url = String(server) + "?api_key=" + apiKey + "&field1=" +
String(temperature);
    http.begin(url);
    int httpCode = http.GET();
    if (httpCode > 0) {
      Serial.println("Data sent to ThingSpeak!");
    } else {
      Serial.println("Error sending data");
    }
    http.end();
  }

  delay(20000);
}
```

➢ Conclusion

The temperature monitoring system utilizing ThingSpeak integrates various IoT components to create a functional and scalable solution. It provides real-time data collection, visualization, and analysis, showcasing the power of cloud computing in IoT applications. This project not only demonstrates fundamental IoT concepts but also encourages further exploration into more complex applications and data management strategies.