Ex: 14 Temperature Sensor Monitoring with NodeMCU

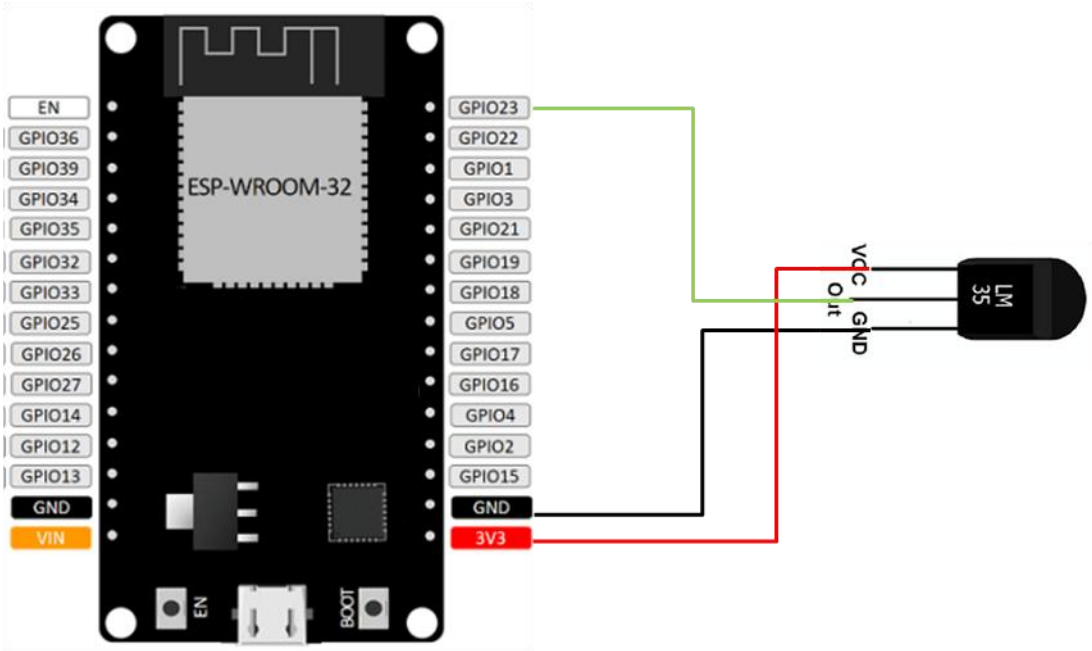
Aim:

To implement Temperature monitoring and storing in ThingSpeak cloud using ESP32.

Components Required:

- ESP 32
- LM 35 Temperature Sensor
- Jumpers
- WiFi Network
- Channel ID and API Key from www. thingspeak.com
    - Create a login using your email.
    - Create a new channel in your login.

Circuit Connection:



Pin Connections between LM32 and ESP32

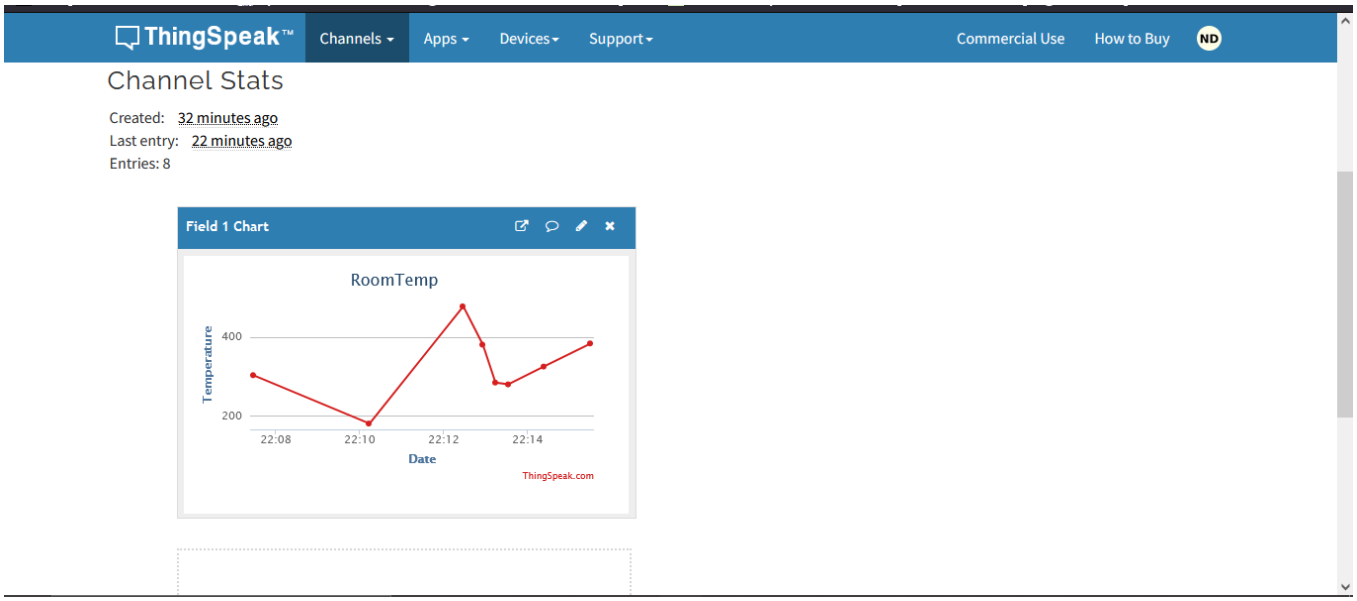
LM35	Arduino UNO Pin
Ground	Ground
OUT	GPIO23
VCC	3V3

Sketch

```
#include <WiFi.h>
#include <ThingSpeak.h> // always include thingspeak header file after other header files

#define SECRET_SSID "SAMPLE"    // replace with your WiFi network name
#define SECRET_PASS "SAMPLE" // replace with your WiFi password

#define SECRET_CH_ID 0000000   // replace 0000000 with your channel number
#define SECRET_WRITE_APIKEY "XYZ"   // replace XYZ with your channel write API Key

char ssid[] = SECRET_SSID;   // your network SSID (name)
char pass[] = SECRET_PASS;   // your network password
int keyIndex = 0;            // your network key Index number (needed only for WEP)
WiFiClient  client;
unsigned long myChannelNumber = SECRET_CH_ID;
const char * myWriteAPIKey = SECRET_WRITE_APIKEY;
int temppin=32;
float pinval;
float temp=0.00;

void setup()
{
  pinMode(temppin,INPUT);
  Serial.begin(115200);  //Initialize serial
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo native USB port only
  }
    WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);  // Initialize ThingSpeak
}

void loop()
{
  // Connect or reconnect to WiFi
  if(WiFi.status() != WL_CONNECTED)
  {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(SECRET_SSID);
    while(WiFi.status() != WL_CONNECTED)
    {
      WiFi.begin(ssid, pass); // Connect to WPA/WPA2 network.
      Serial.print(".");
      delay(5000);
    }
```

```
  Serial.println("\nConnected.");
 }
 pinval=analogRead(temppin);
 temp=pinval*(0.122);
 Serial.println(temp);

 // Write to ThingSpeak.
 int x = ThingSpeak.writeField(myChannelNumber, 1, temp, myWriteAPIKey);
 if(x == 200)
 {
  Serial.println("Channel update successful.");
 }
 else
 {
  Serial.println("Problem updating channel. HTTP error code " + String(x));
 }
 delay(1000); // Wait 7 seconds to update the channel again
}
```

Output:



Result:

Thus ESP32 board is utilized to collect temperature data and upload it into Thinkspeak cloud.