INTEGRATING TEMPERATURE AND HUMIDITY SENSOR WITH NODE MCU

Anand Kumar Sahu [201020405]
Kshitij Kumar [201020426]
Sanskar Chandra [201020449]
Aditya Goel [201020402]

Guided by- Dr. Shrivishal Tripathi

B.Tech 1st year (2020- 2024)

Dr. Shyama Prasad Mukherjee International Institute of Information Technology

ABSTRACT

Temperature controlling is required in many places such as server rooms, houses, industries, etc. So, this project can be very useful in understanding how to control the temperature at your home autonomously. We have done this with a DHT11 temperature and humidity sensor. It can be done in various other ways as well. With a thermistor or other sensors like a contactless temperature sensor. But thermistor usually need contact with the surface and contactless sensors can be costly. While DHT11 is a cost-effective sensor.

INTRODUCTION

Home Automation has become a popular topic in modern-day technology. Each person tends to use automated devices in his or her daily life due to multiple reasons ranging from safety to ease of handling. The small-scale industrialists, therefore, have focused on increasing their market to the public in a more efficient way.

Many researchers are working on different methodologies to achieve their objectives. For examples, S. Shimamura and his team worked on Smart fan robot, which automatically detects and tracks a person using thermal camera and RGB-D camera.Another research work was done on human tracking system via ultrasonic distance sensors to detect a person only up to 3 meters.PWM technique is also used to control the fan speed by changing the duty cycle according to room temperature.In addition, Md Mozasser Rahman, Mohd Fahrul and Shahrul Sidek developed a smart fan, which operates based on the presence of human, ambient temperature and the position of humans.Another project is conducted to control home appliances using smart phone with Wi-Fi as communication protocol and Raspberry Pi as a server system.

Most of these research work are on human tracking and controlling the speed of fan but not focused on human comfort. Heat Index ,also known as ,apparent temperature, tells what the temperature feels like to the human body when relative humidity is combined with the air temperature. This has important considerations for the human body's comfort. When the body gets too hot, it begins to perspire or sweat to cool itself off. If the perspiration is not able to evaporate, the body cannot regulate its temperature. Hence , Temperature and Humidity are key to Human Comfort.

In this project we have collected temperature and humidity to make a temperature-controlled fan using NodeMCU. With this project, we will be able to change the fan speed in our home or any place according to the room temperature and also display the current temperature, humidity and fan status on the Blynk application remotely.

OBJECTIVE

Integrating Temperature & Humidity Sensor with Node MCU to read current Environment values. DHT11 is a Humidity and Temperature Sensor, which generates calibrated digital output. It shows the temperature, humidity and heat index value. The heat index is an index value that combines the temperature and relative humidity, to propose a human-perceived equivalent temperature.

LIST OF COMPONENTS

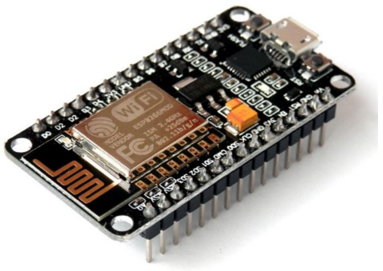
- ESP8266
- DHT11 Humidity and Temperature sensor
- IRF520 MOSFET Driver Module
- Bulb
- A Fan
- Jumper – male to female wires for connections
- Micro-USB cable
- Power bank for power supply.

Arduino IDE has been used to compile the code.

USAGE OF COMPONENTS

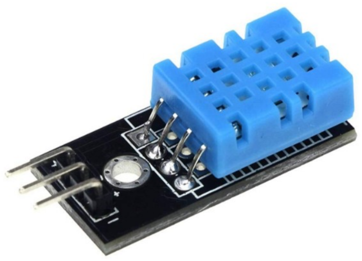
1. **NodeMCU ESP8266** -

The ESP8266 is a system on a chip (SOC) Wi-Fi microchip for Internet of Things (IoT) applications produced by Espressif Systems. The ESP8266 module enables microcontrollers to connect to 2.4 GHz Wi-Fi, using IEEE 802.11 b/g/n. It can be used with ESP-AT firmware to provide Wi-Fi connectivity to external host MCUs, or it can be used as a self-sufficient MCU by running an RTOS-based SDK. The module has a full TCP/IP stack and provides the ability for data processing, reads and controls of GPIOs.



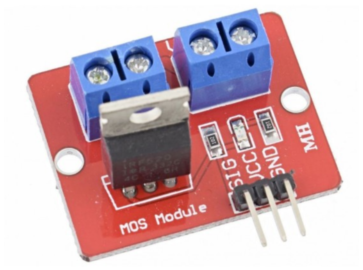
2. **DHT11 sensor**-

DHT11 sensor measures and provides humidity and temperature values. The DHT11 is chosen because it is lab calibrated, accurate and stable and its signal output is digital. Most important of all, it is relatively inexpensive for the given performance. The digital output signal by the sensor can be read by any microcontroller or microprocessor for further analysis.

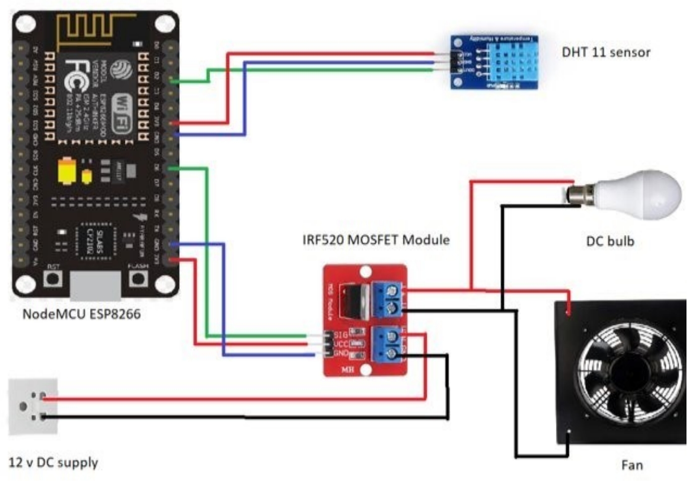


3. **IRF520 MOSFET Module** -

The module is designed to switch heavy DC loads from a single digital pin of your microcontroller. Its main purpose is to provide a low-cost way to drive a DC motor for robotics applications, but the module can be used to control most high current DC loads. Screw terminals are provided to interface to your load and external power source. An LED indicator provides a visual indication of when your load is being switched.



CIRCUIT DIAGRAM



Working Principle

1. Firstly, the DHT 11 sensor detects the temperature and humidity of the system (surrounding).

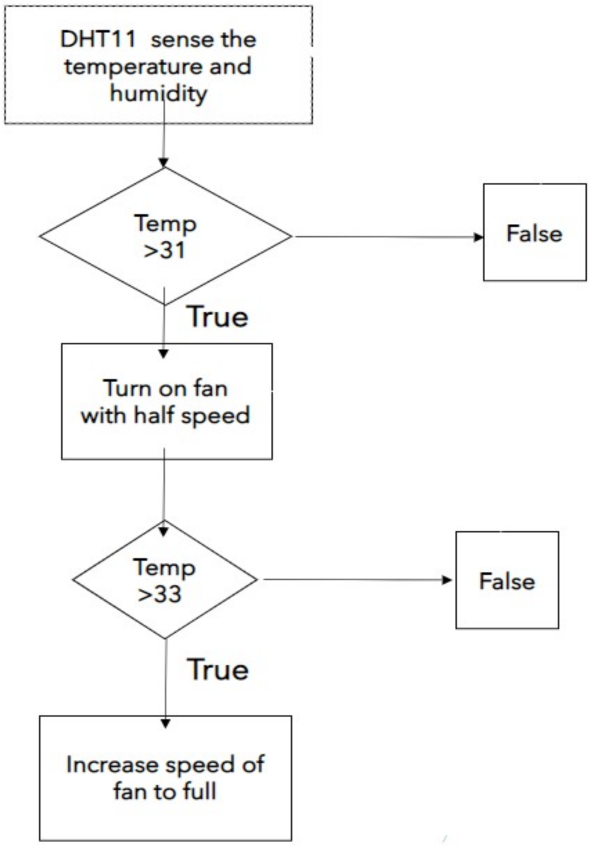
2. Initially, the fan is switched off. But, when the temperature of the system (surrounding) is between 31°C and 33°C, then the fan is switched on automatically with 50% of its speed.

3. As soon as the temperature of the system (surrounding) becomes greater than 33°C then the fan rotates with its full speed. Thus, cooling the system.

4. Thus the entire process works automatically.

5. The temperature and humidity data collected during the entire process can also be accessed through a mobile or web interface.

ALGORITHM (FLOWCHART)


DHT11 sense the temperature and humidity
Temp >31
False
True
Turn on fan with half speed
Temp >33
False
True
Increase speed of fan to full

The data collected from the DHT11 sensor has been integrated to the thingspeak cloud service and to the Blynk application for the user preview. The data can be seen on both platforms and it is real-time. The project output is to show the surrounding environment temperature, humidity, and the Heat index value.

RESULT

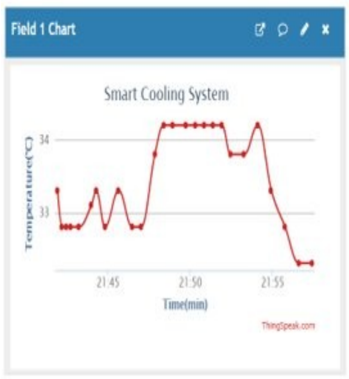
This proposed system provides a convenient method for effective monitoring of temperature and humidity in real time. We get the surrounding environment temperature, humidity, and the Heat index value. The data can be seen on both Blynk app and thingspeak cloud and it is real-time.

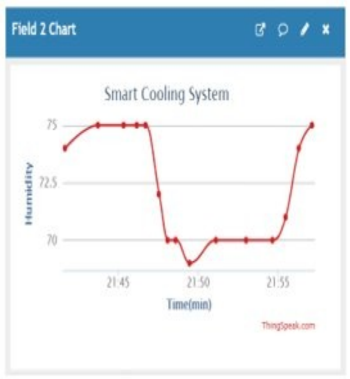
The unique feature which our system has is that it automatically turns the fan ON/OFF based on the temperature. If the temperature is high above a pre-specified value, then the fan will turn ON, this way it provides comfort to the person using this system. The user can also view the temperature, humidity and the heat index value on either of the two platforms - Blynk Application and Thingspeak.

When viewing the current temperature and other readings, the interface also shows the readings within some time range, which means we can see the reading of a certain time before also.

This way the user can see and analyse the data if they want and make certain decisions accordingly.

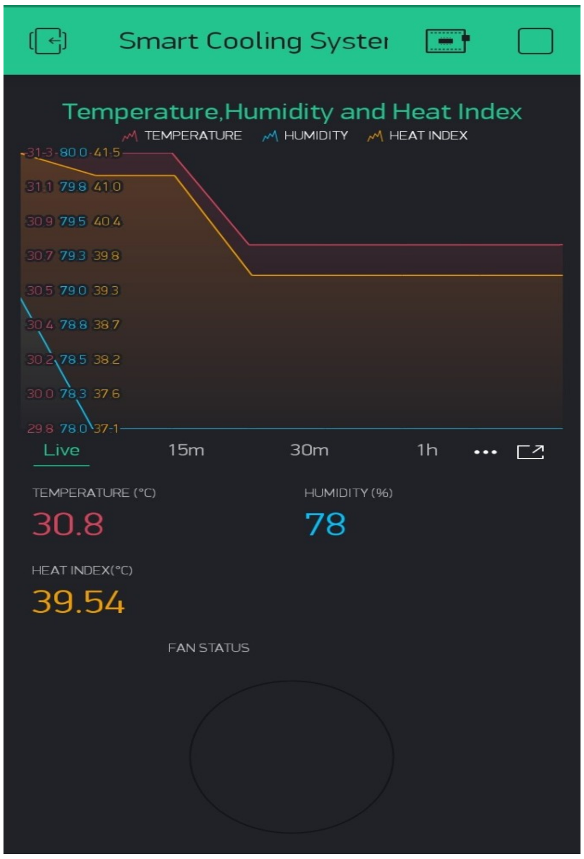
Thingspeak interface is shown below :





CASE 1- WHEN TEMPERATURE IS LESS THAN 31°C

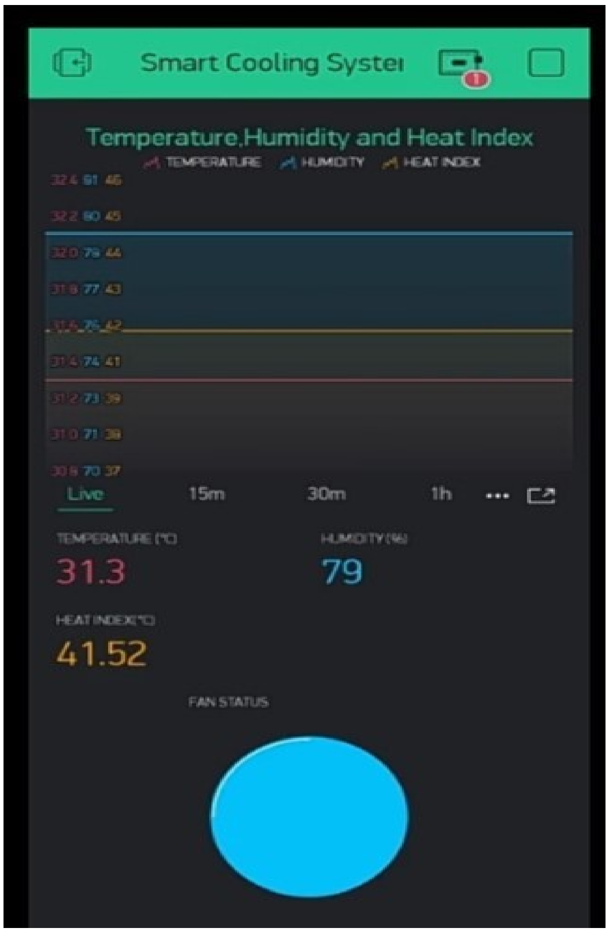
When the temperature goes down 31°C, the fan and indicator remain switched off, since there is no need at low temperature.



TEMPERATURE (°C) 30.8

HUMIDITY (%) 78

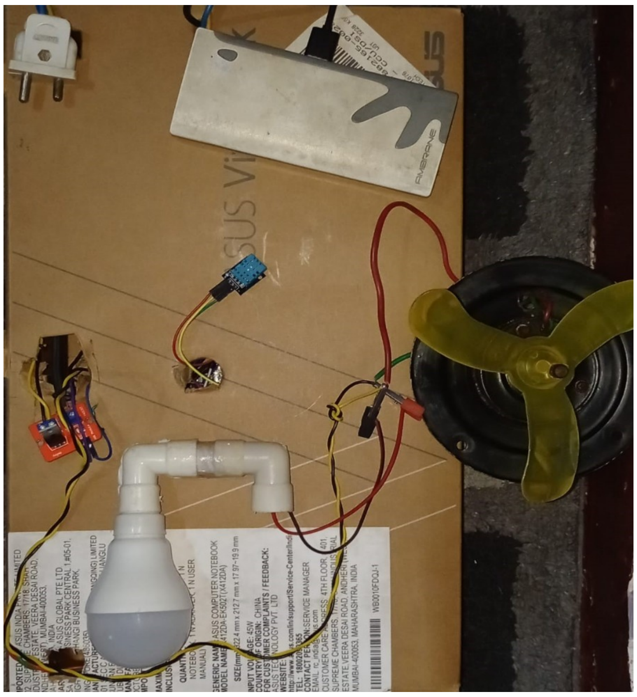
HEAT INDEX(°C) 39.54

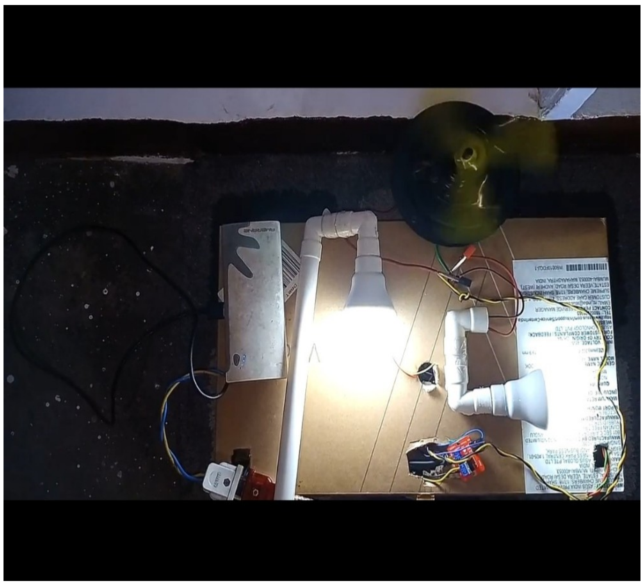


TEMPERATURE (°C) 31.3

HUMIDITY (%) 79

HEAT INDEX(°C) 41.52



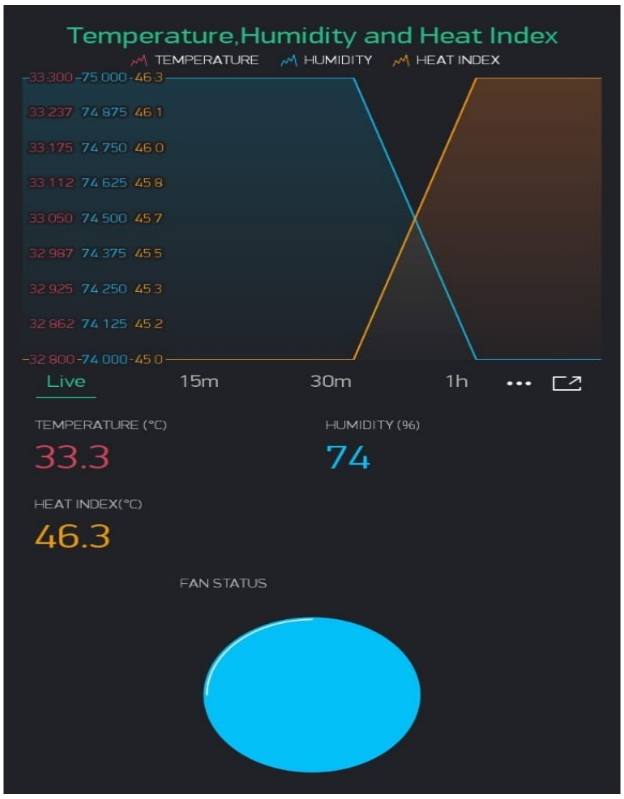


CASE 2 - WHEN THE TEMPERATURE GOES BETWEEN 31°C AND 33°C

Fan rotates at its 50% capacity when temperature is between 31°C and 33°C

CASE 3 - WHEN THE TEMPERATURE GOES ABOVE 33°C

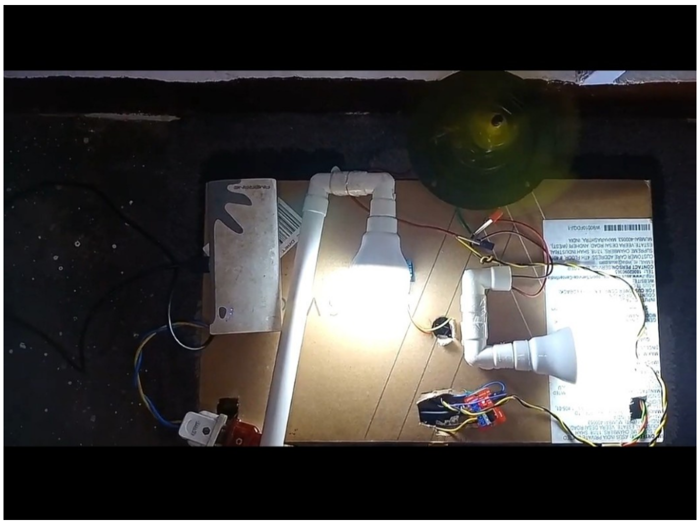
Fan speed reaches full capacity when temperature goes above 33°C to cool down temperature.



TEMPERATURE (°C)
33.3

HUMIDITY (96)
74

HEAT INDEX(°C)
46.3



Comparison With The Existing Solution

1. In the previous projects, there is one thing that is missing, and that is automation. They showed the current temperature and humidity level, but no work is being done to raise or decrease these values.
   Thus, we integrated a fan with the project, which will get on when the temperature rises above a certain level, and its speed will further increase when the temperature is high.

2. We have also linked our project to thingspeak cloud so that anyone can access the data.

Applications

This can be used as a **Smart Cooling system** at various places. One major application is that this system can be used in ventilation systems.
DHT11 Relative Humidity and Temperature Sensor can be used in many applications like:

- HVAC (Heating, Ventilation and Air Conditioning) Systems.

  This prototype can really help in the heating and ventilation systems due to its appropriate application.

- Weather Stations
- Medical Equipment for measuring humidity (temperature) and cooling the system
- Home Automation Systems
- Automotive and other weather control applications

ACKNOWLEDGEMENT

All the group members would like to acknowledge our professor Dr. Shrivishal Tripathi for his guidance during the whole process of project making and its implementation. We thank everyone for their best wishes and helping us to make this project a success. We are very grateful to one’s who dedicated their valuable time to guiding us throughout this project.
It brings us great pleasure for an opportunity to work and submit our IT Workshop-2 project on **Integrating Temperature And Humidity Sensor With Node Mcu**.
For this we are sincerely thankful to Dr. Shrivishal Tripathi for his help and invaluable guidance and encouraging us throughout the course of the current project. Finally, we are deeply thankful to our parents and friends who helped and inspired us in this project.

REFERENCES

1. https://iotdesignpro.com/projects/temperature-humidity-monitoring-over-thingspeak-using-arduino-esp8266
   For connecting the data with Thingspeak cloud service
2. https://www.jncet.org/Manuscripts/Volume-8/Issue-4/Vol-8-issue-4-M-73.pdf
   To get to know about the different pins of DHT11 Temperature and Humidity Sensor.
3. https://www.electronicshub.org/dht11-humidity-sensor-arduino/
   Working of the DHT11 sensor and how the data is acquired.

APPENDIX

SOURCE CODE-

```
//Here we go with the code:
#include <ESP8266WiFi.h>;
#include<WiFiClient.h>;
#define BLYNK_PRINT Serial                          // Enables Serial Monitor
#include <BlynkSimpleEsp8266.h>                    //For Blynk interface
#include <ThingSpeak.h>;                           //For thingspeak interface
#include "DHT.h"

const char* ssid = "wifi name";
const char* pass = "wifi password";

unsigned long myChannelNumber = 1430786;                       //Thingspeak channel
number
const char * myWriteAPIKey = "XOCER4ZMO34ty57v";            //Thingspeak API key

#define DHTPIN 4
// Digital pin connected to the DHT sensor D2
#define DHTTYPE DHT11

char auth[] = "1mKYOsvaY-C3cc0gSZAex_RIs5At_9eC";       //Blynk authorisation token

DHT dht(DHTPIN, DHTTYPE);

#define FAN_PIN 12 // D6
WidgetLED FAN(V4);
//Sending the status of fan that the fan is on or off

WiFiClient client;

void setup()
{
   Blynk.begin(auth, ssid, pass);
   Serial.begin(9600);
   pinMode(FAN_PIN, OUTPUT);
   Serial.println("Speed Depends on Temperature");
   dht.begin();
   ThingSpeak.begin(client);
}
```

```
 void loop()
{
  Blynk.run();
  float h = dht.readHumidity();
 // Read humidity (the default)
  float t = dht.readTemperature();
// Read temperature as Celsius (the default)

  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
        return;
  }

  //Thingspeak field number to send data on it
  ThingSpeak.writeField(myChannelNumber, 1, t, myWriteAPIKey);
  ThingSpeak.writeField(myChannelNumber, 2, h, myWriteAPIKey);
  ThingSpeak.writeField(myChannelNumber, 3, hic, myWriteAPIKey);

  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print(F("°C  Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.println(" ");

  //When temperature greater than 33 fan rotates with its full speed
  if (t > 33) {
    digitalWrite(FAN_PIN,HIGH);
    analogWrite(FAN_PIN, 255);
    FAN.on();
  }
  //When temperature lies between than 31 and 33 fan rotates with its half speed
  else if (t >= 31 && t <= 33) {
    digitalWrite(FAN_PIN,HIGH);
    analogWrite(FAN_PIN, 255 / 2); //speed reduces to 50%
    FAN.on();
  }
  //When temperature less than 31 fan does not rotates
  else if (t < 31)
  {
   digitalWrite(FAN_PIN,LOW);
   FAN.off();
  }
  Blynk.virtualWrite(V1,t);
//sending temperature of environment to Blynk
  Blynk.virtualWrite(V2,h);
//sending humidity of environment to Blynk
  Blynk.virtualWrite(V3,hic);
//sending Heat Index of environment to Blynk
      delay(500);
}
```