**IOT BASED TEMPERATURE MONITORING SYSTEM**

Submitted in partial fulfilment of the Requirement

for the award of the Degree of

**BACHELOR OF TECHNOLOGY**

**IN**

Electronics & Communication Engineering

By

D.Ajay-1602-20-735-122

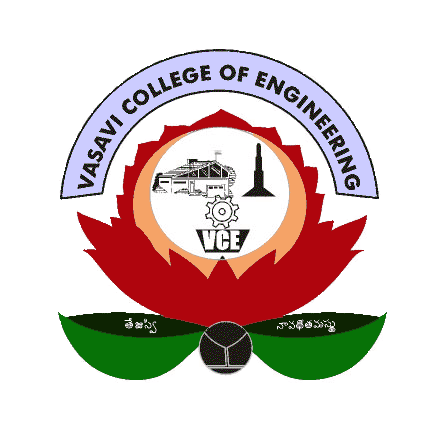
R. Sanjana-1602-20-735-157

S. Satvika-1602-20-735-159

Under the Guidance of

DEPARTMENT OF ELECTRONICS AND

COMMUNICATION ENGINNERING



# ABSTRACT

IoT (Internet of Things) plays a very important role in the present day. It is used to connect things over a network. From home automation to industries and agriculture, there are wide applications of IoT. This project aims to connect a temperature sensor over the internet.

The IoT based temperature monitoring system aims to collect data from the temperature sensor and send to a cloud platform. The cloud platform helps in remotely monitoring the data collected by hardware. This helps in easing the process of monitoring the temperature. Once connected to cloud and the data is accessed remotely, this project can be implemented in various applications such as automating the system to do a specific task like switching on AC when hot, watering the plants when soil is dry etc.

## **KEYWORDS**

Here are some keywords that are used in this project:

* **IoT–**It refers toa system of interrelated, internet-connected objects that are able to collect and transfer data over a wireless network without human intervention.
* **Cloud–** It refers to servers that are accessed over the Internet, and the software and databases that run on those servers.
* **Sensors–** It is a device which detects or measures a physical property and records, indicates, or otherwise responds to it.
* **Channel –** Channel can be said as a stream of data. It identified by a numerical channel ID using which data can be inserted or retrieved using ThingSpeak APIs.
* **Field –** Each channel is having 8 fields which can hold any type of data. For eg. you may store temperature, humidity, RFID data (alphanumeric) in each channels.
* **Status –** It is short status message to augment the data stored in a channel
* **Location –** In addition to above 8 fields we can store GPS location or coordinates. For example we can store the location of the place from where the data is coming. It is having latitude, longitude and elevation.
* **Write API Key** – A 16 digit API key code that allows an application to write data to a channel. You should not share this API key publicly, because anyone having this key can write data to your channel.
* **Read API Key** – A 16 digit API key code that allows an application to read the data stored in a channel. You should not share this API key publicly, because anyone having this key can read data from your channel.

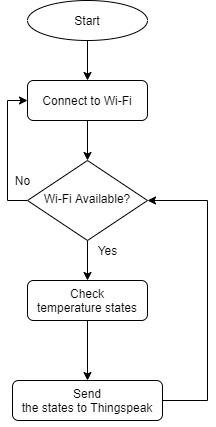
# INTRODUCTION

Temperature is very common parameter of measurement. It is used in measuring various things and has wide variety of applications like in farms, green house, medical, industries home and offices. ESP8266 Wi-Fi transceiver is one of the most popular Wi-Fi module for IoT based Applications. In this project we will use it to connect with Thing-Speak IoT cloud Platform.

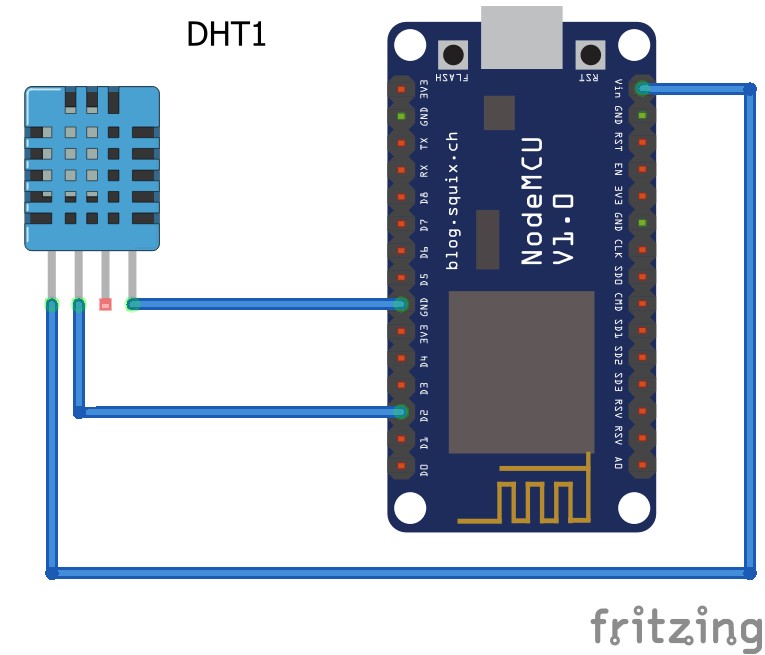
Using Internet of Things (IOT), any electronic equipment in homes and industries can be easily monitored and controlled. Moreover, data can be read from any sensor and analyze it graphically from anywhere in the world. In this project, we are going to send the data from Temperature sensor to Thing-speak using DHT11. By this method we can monitor our DHT11 sensor’s temperature data over internet using Thing-Speak IOT server, and we can view the logged data and graph over time on the Thing-Speak dashboard. NodeMCU reads the current temperature from DHT11 and sends it to Thing-Speak server for live monitoring from anywhere in the world.

Thing-Speak is a data platform for monitoring your data online, targeted to be used for IoT applications. In Thing-speak channel you can set the data as private or public according to your choice. Thing-Speak takes minimum of 15 seconds to update your readings. It’s a great and very easy to use platform for building IOT projects.

**FLOW CHART**



**BLOCK DIAGRAM**



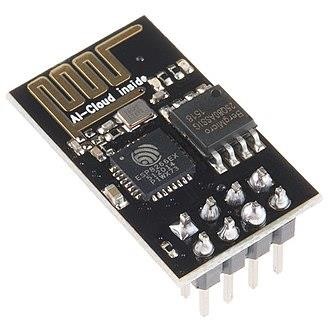
**COMPONENTS REQUIRED**

**Hardware:**

1. ESP8266
2. DHT11 temperature sensor
3. Arduino UNO
4. Jumper wires
5. Breadboard

**Software:**

* Thingspeak
* Arduino IDE



**DESCRIPTION**

**ESP8266:**

The ESP8266 is a low-cost Wi-Fi microchip, with a full TCP/IP stack and microcontroller capability. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands. The ESP8285 is an ESP8266 with 1 MB of built-in flash, allowing the building of single-chip devices capable of connecting to Wi-Fi.

The pin-out is as follows for the common ESP-01 module:

1. GND, Ground (0 V)
2. GPIO 2, General-purpose input/output No. 2
3. GPIO 0, General-purpose input/output No. 0
4. RX, Receive data in, also GPIO3
5. VCC, Voltage (+3.3 V; can handle up to 3.6 V)
6. RST, Reset
7. CH\_PD, Chip power-down
8. TX, Transmit data out, also GPIO1

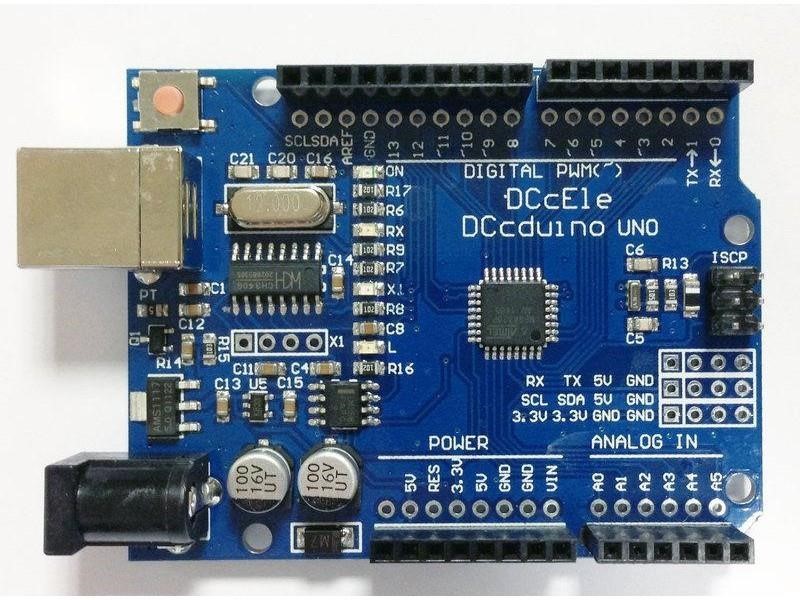
**DHT11 Sensor:**

DHT11 is a low-cost digital sensor for sensing temperature and humidity. This sensor can be easily interfaced with any micro-controller such as Arduino, Raspberry Pi etc… to measure humidity and temperature instantaneously.

DHT11 humidity and temperature sensor is available as a sensor and as a module. The difference between this sensor and module is the pull-up resistor and a power-on LED. DHT11 is a relative humidity sensor. To measure the surrounding air this sensor uses a thermistor and a capacitive humidity sensor.

DHT11 sensor has four pins- VCC, GND, Data Pin and a not connected pin. A pull-up resistor of 5k to 10k ohms is provided for communication between sensor and micro-controller.

**Arduino UNO:**

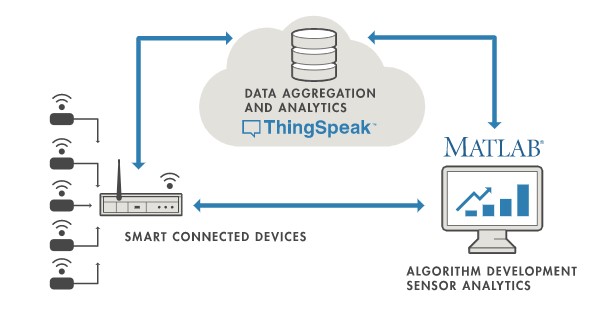
Arduino is an open-source electronics platform based on easy-to-use hardware and software. [Arduino boards](https://www.arduino.cc/en/Main/Products) are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use theArduino programming [language](https://www.arduino.cc/en/Reference/HomePage) (based on [Wiring)](http://wiring.org.co/), and [the Arduino Software (IDE),](https://www.arduino.cc/en/Main/Software) based on [Processing.](https://processing.org/)

It has a simple and accessible user experience,

Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step-by-step instructions of a kit, or sharing ideas online with other members of the Arduino community.

**ThingSpeak API:**

[ThingSpeak](https://thingspeak.com/) is a very good platform for IoT based projects. By using channels and web pages provided by ThingSpeak we can monitor any data over the Internet from anywhere and we can also control our system over the internet. ThingSpeak ‘Collects’ data from sensors, ‘Analyze and Visualize’ data and ‘Acts’ by triggering a reaction. Here we are explaining about how to setup ThingSpeak account for this project.



**PROCEDURE**

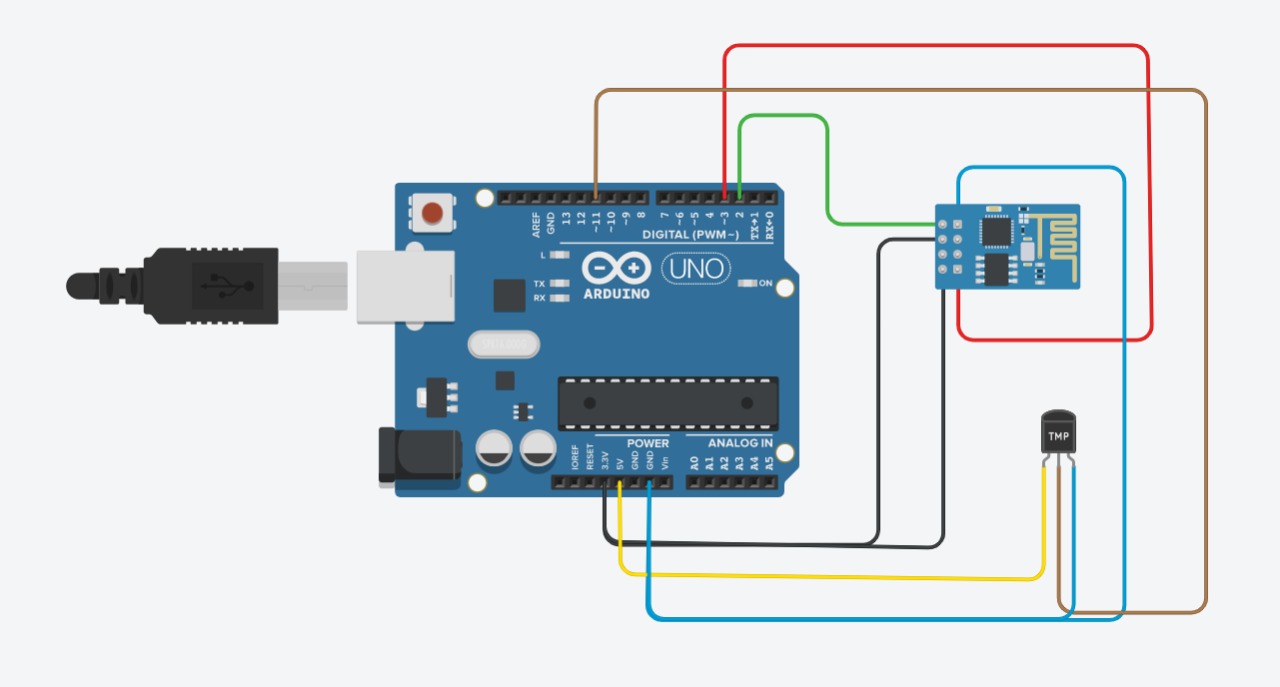
1. The hardware must be connected according to the block diagram shown above. The following table shows the ESP8266 and DHT11 connections.

|  |  |  |
| --- | --- | --- |
| **S No.** | **NodeMCU** | **DHT11** |
| 1 | Vin | VCC |
| 2 | Data | D2 |
| 3 | GND | GND |

1. Open Thingspeak, First of all, user needs to create an account on ThingSpeak.com, then Sign In and click on ‘Get Started’.
2. Now go to the ‘Channels’ menu and click on the ‘New Channel’ option in the same page. 4. Now you will see a form for creating the channel, fill the Name and Description as per your choice. Then fill ‘Temperature’ in ‘Field 1’ field. Tick the check box ‘Make Public’ option below the form and finally Save the channel. Now your new channel is ready.
3. Now click on ‘API keys’ tab and note the Write and Read API key, here we are only using Write key. You need to copy and paste this key in the code (see below).
4. Now user need to upload the program to ESP8266 using [Arduino](https://electrosome.com/arduino/) IDE.
5. After uploading, open “PRIVATE VIEW” icon in ThingSpeak website and observe the monitored temperature value on graph as shown below.
6. Open Arduino IDE, install required libraries, and enter SSID, password of Wi-Fi. Also enter the API key of the channel. Now, compile and the run the code.
7. We can observe the output temperature on the Serial monitor, by visiting Private view visualization in Thingspeak, we can have a clear picture of the temperature trends.

**OUTPUT**

Simulation in Tinker CAD:

**DESIGN**

**Simulation Code:**

#include <SoftwareSerial.h>

#include <dht11.h>

#define RX 2

#define TX 3

#define dht\_apin 11 // Analog Pin sensor is connected to

dht11 dhtObject;

String AP = "Galaxy M21"; // AP NAME

String PASS = "1234pass"; // AP PASSWORD

String API = "EVTYM7YDNOFYKQGP"; // Write API KEY

String HOST = "api.thingspeak.com";

String PORT = "80";

int countTrueCommand;

int countTimeCommand;

boolean found = false;

int valSensor = 1;

SoftwareSerial esp8266(RX,TX);

void setup() {

Serial.begin(9600);

esp8266.begin(115200);

sendCommand("AT",5,"OK");

sendCommand("AT+CWMODE=1",5,"OK");

sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",5,"OK");

}

void loop() {

String getData = "GET /update?api\_key="+ API +"&field1="+getTemperatureValue()+"&field2="+getHumidityValue();

sendCommand("AT+CIPMUX=1",5,"OK");

sendCommand("AT+CIPSTART=0,\"TCP\",\""+ HOST +"\","+ PORT,15,"OK");

sendCommand("AT+CIPSEND=0," +String(getData.length()+4),4,">");

esp8266.println(getData);delay(1500);countTrueCommand++;

sendCommand("AT+CIPCLOSE=0",5,"OK");

}

String getTemperatureValue(){

dhtObject.read(dht\_apin);

Serial.print(" Temperature(C)= ");

int temp = dhtObject.temperature;

Serial.println(temp);

delay(50);

return String(temp);

}

String getHumidityValue(){

dhtObject.read(dht\_apin);

Serial.print(" Humidity in %= ");

int humidity = dhtObject.humidity;

Serial.println(humidity);

delay(50);

return String(humidity);

}

void sendCommand(String command, int maxTime, char readReplay[]) {

Serial.print(countTrueCommand);

Serial.print(". at command => ");

Serial.print(command);

Serial.print(" ");

while(countTimeCommand < (maxTime\*1))

{

esp8266.println(command);//at+cipsend

if(esp8266.find(readReplay))//ok

{

found = true;

break;

}

countTimeCommand++;

}

if(found == true)

{

Serial.println("OYI");

countTrueCommand++;

countTimeCommand = 0;

}

if(found == false)

{

Serial.println("Fail");

countTrueCommand = 0;

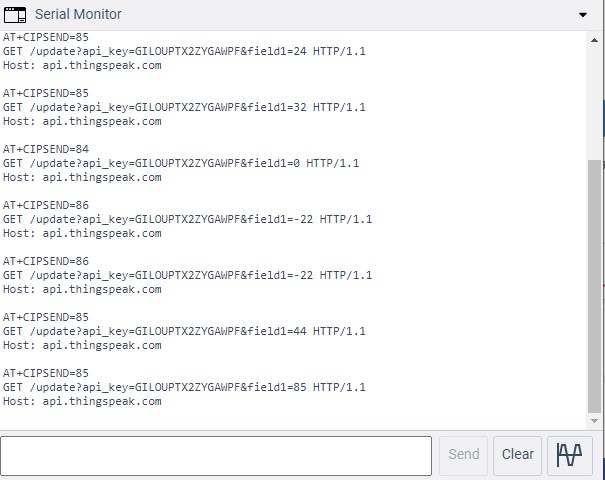
countTimeCommand = 0;

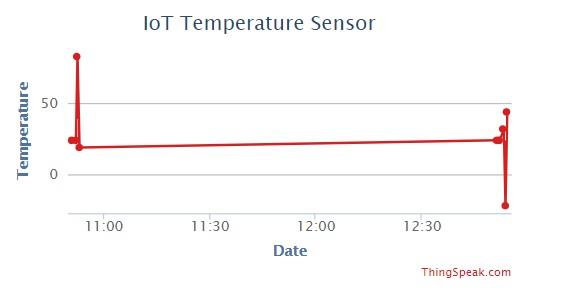
}

found = false;

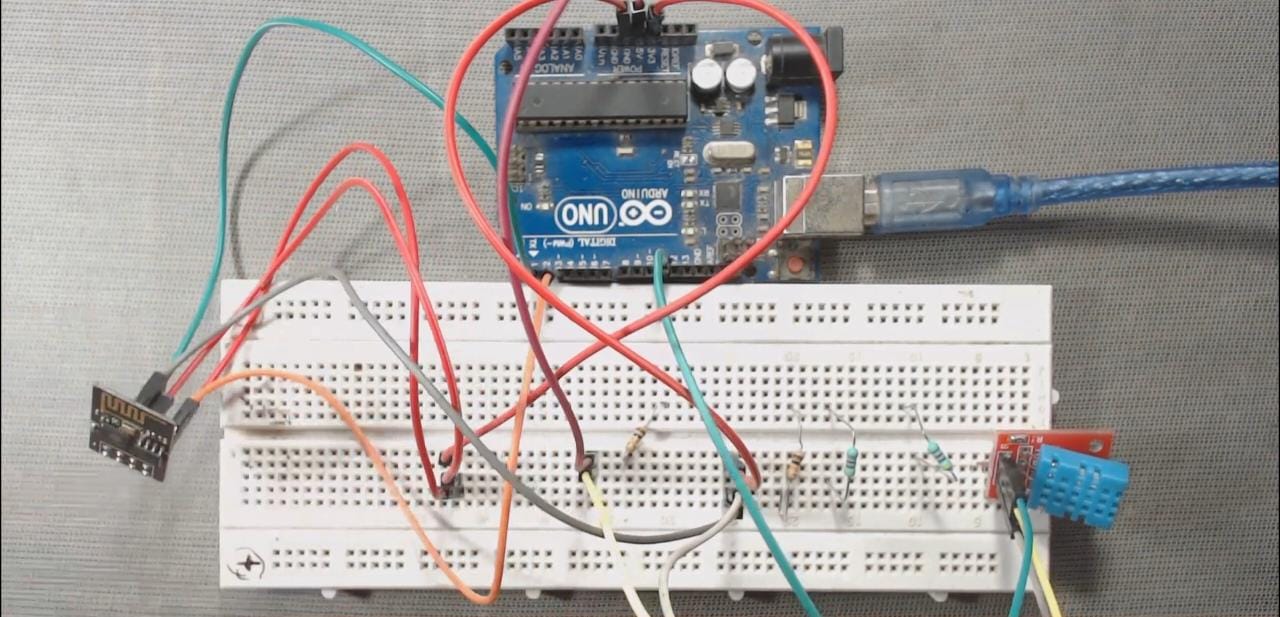
}

**Serial output**



**THINK SPEAK OUTPUT**

**WORKING MODEL**

****

**CONCLUSION**

There are various applications for an IOT based embedded system. The IOT includes communication protocols, Data Acquisition and controlling systems. The temperature has been monitored with the combination of IOT and Arduino and for every second time period the temperature has been updated in the particular IP address.

There are various business spaces where there is a need to observe temperature and update the status to the cloud. For example, the temperature must be maintained at the lowest level in food preservation process. IOT based temperature monitoring system help us to monitor the food preservation system temperature and update the data to the cloud at the regular interval. This eliminates redundant tasks like taking manual readings, thus saving time and elevating quick decision making.

**FUTURE SCOPE**

* An SMS can be sent to notify the temperature changes.
* We can also integrate with home automation i.e., if it’s exceeding a threshold temperature, it can automatically switch on the AC or, if it’s too cold, it can switch on the heater etc.
* This system can be very useful for ice-cream shop as right temperature is required for the ice-creams.