ABSTRACT OF THE PROJECT:

IOT WEATHER STATION WITH NODEMCU ESP8266

IN THIS LATEST IOT PROJECT, WE WILL LEARN HOW TO MAKE AN IOT BASED SIMPLE ONLINE WEATHER STATION USING ESP8266 NODEMCU (WEMOS D1) WI-FI DEVELOPMENT BOARD .

THE NODEMCU PULLS WEATHER DATA LIKE TEMPERATURE, HUMIDITY, RAIN,VIBRATION AND DISPLAYS ON THE LOCALHOST WEBSITE CONNECT TO THE WIFI OF NODEMCU.

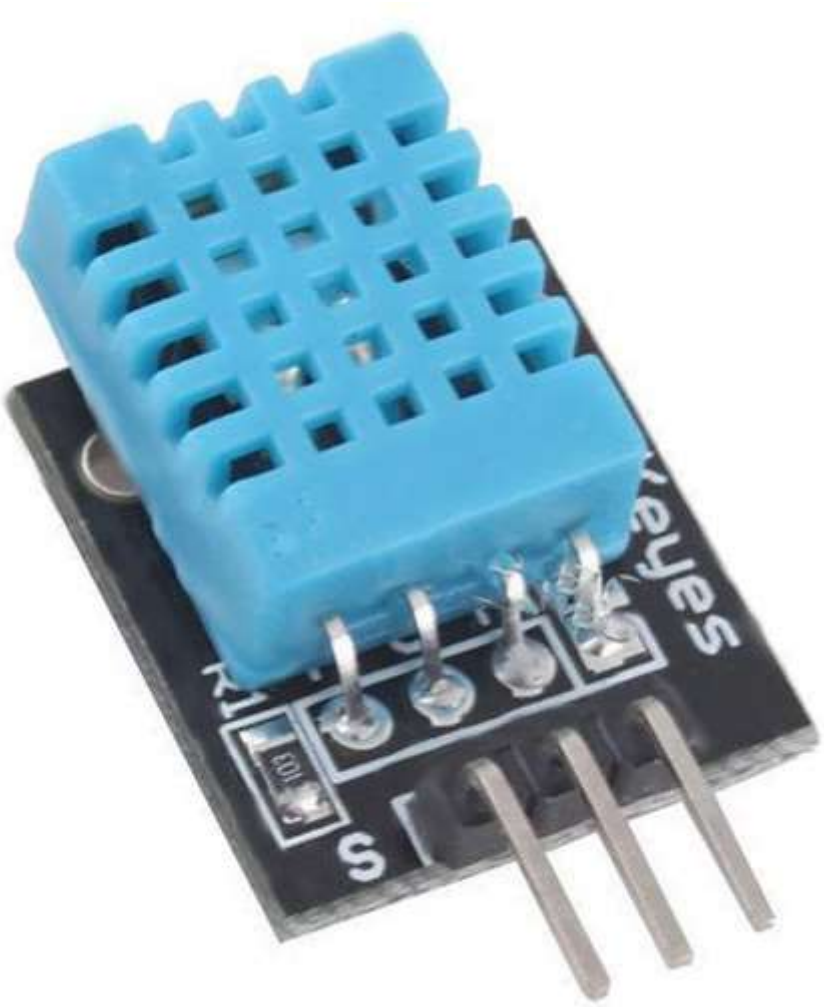
Components Required

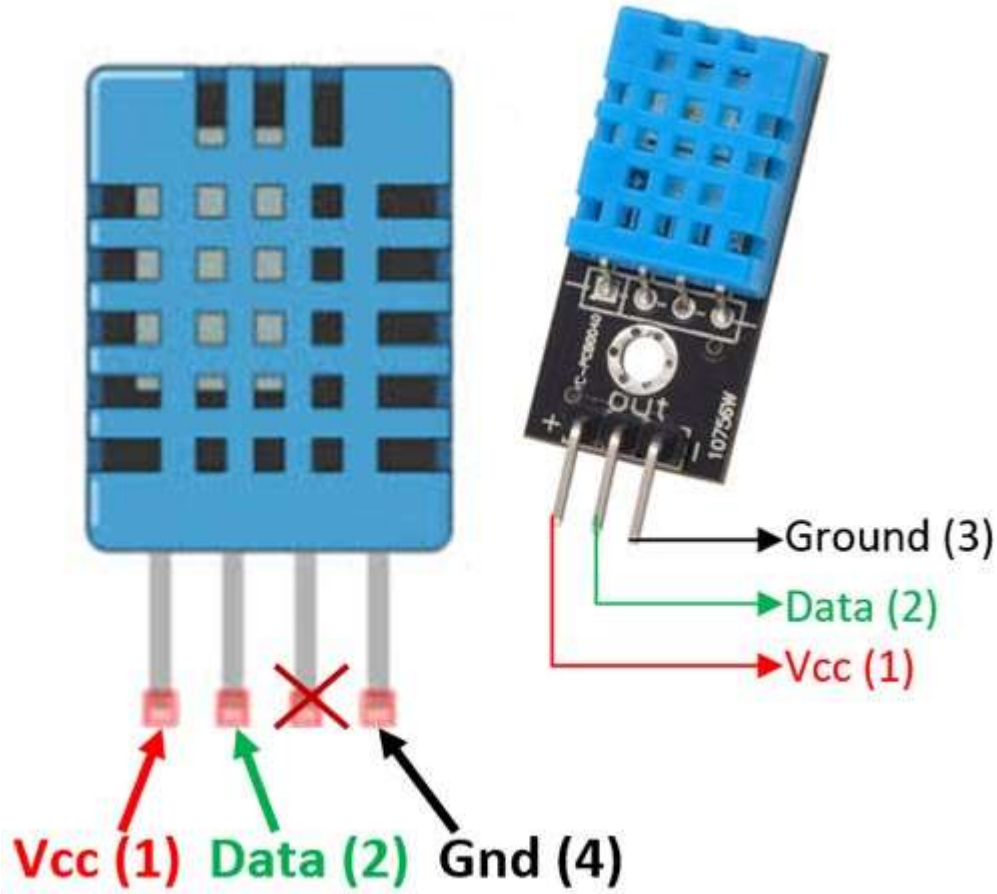
1. NODEMCU

2. DHT11 SENSOR
3. RAIN SENSOR
4. VIBRATION SENSOR
5. BREADBOARD
6. JUMPER WIRES

DHT11 Temperature and Humidity Sensor

Humidity Sensor


R1
103
S



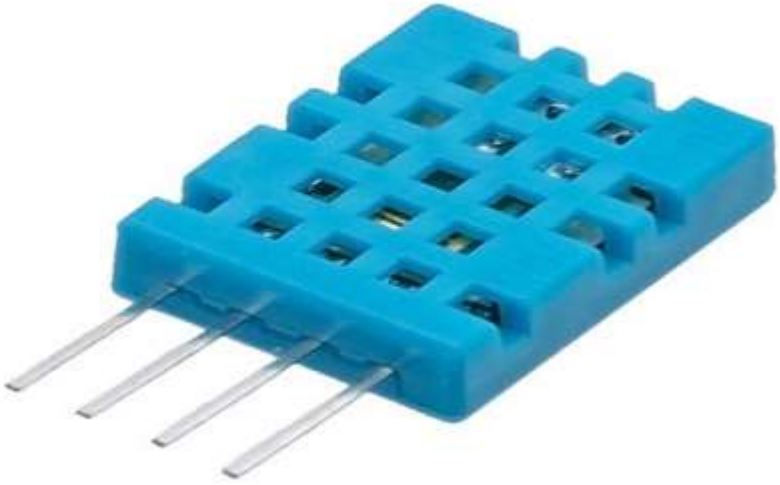
DHT11 MODULE FEATURES A HUMIDITY AND TEMPERATURE COMPLEX WITH A CALIBRATED DIGITAL SIGNAL OUTPUT MEANS DHT11 SENSOR MODULE IS A COMBINED MODULE FOR SENSING HUMIDITY AND TEMPERATURE WHICH GIVES A CALIBRATED DIGITAL OUTPUT SIGNAL. DHT11 GIVES US VERY PRECISE VALUE OF HUMIDITY AND TEMPERATURE AND ENSURES HIGH RELIABILITY AND LONG TERM STABILITY. THIS SENSOR HAS A RESISTIVE TYPE HUMIDITY MEASUREMENT COMPONENT AND NTC TYPE TEMPERATURE MEASUREMENT COMPONENT WITH AN 8-BIT MICROCONTROLLER INBUILT WHICH HAS A FAST RESPONSE AND COST EFFECTIVE AND AVAILABLE IN 4-PIN SINGLE ROW PACKAGE.

DHT11 sensor consists of a capacitive humidity sensing element and a thermistor for sensing temperature. The humidity sensing capacitor has two electrodes with a moisture holding substrate as a dielectric between them. Change in the capacitance value occurs with the change in humidity levels. The IC measure, process this changed resistance values and change them into digital form.

For measuring temperature this sensor uses a Negative Temperature coefficient thermistor, which causes a decrease in its resistance value with increase in temperature. To get larger resistance value even for the smallest

change in temperature, this sensor is usually made up of semiconductor ceramics or polymers.

The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy. Humidity range of this sensor is from 20 to 80% with 5% accuracy. The sampling rate of this sensor is 1Hz .i.e. it gives one reading for every second. DHT11 is small in size with operating voltage from 3 to 5 volts. The maximum current used while measuring is 2.5mA.



DHT11 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.

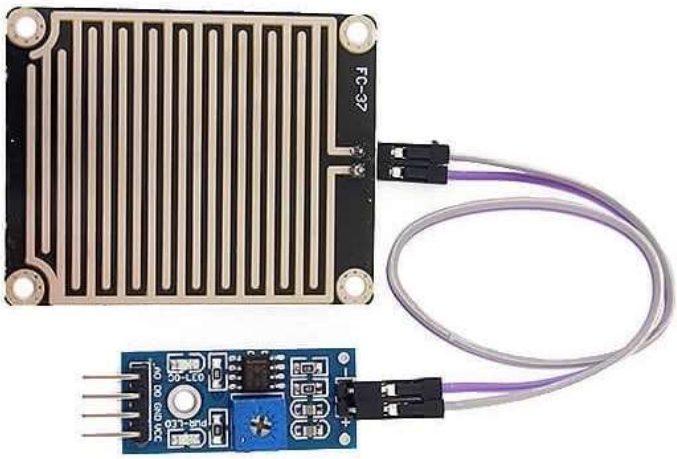
APPLICATIONS

THIS SENSOR IS USED IN VARIOUS APPLICATIONS SUCH AS MEASURING HUMIDITY AND TEMPERATURE VALUES IN HEATING, VENTILATION AND AIR CONDITIONING SYSTEMS. WEATHER STATIONS ALSO USE THESE SENSORS TO PREDICT WEATHER CONDITIONS. THE HUMIDITY SENSOR IS USED AS A PREVENTIVE MEASURE IN HOMES WHERE PEOPLE ARE AFFECTED BY HUMIDITY. OFFICES, CARS, MUSEUMS, GREENHOUSES AND INDUSTRIES USE THIS SENSOR FOR MEASURING HUMIDITY VALUES AND AS A SAFETY MEASURE.

Rain Sensor FC-37:

RAIN SENSORS ARE USED IN THE DETECTION OF WATER BEYOND WHAT A

HUMIDITY SENSOR CAN DETECT.



THE RAIN SENSOR DETECTS WATER THAT COMPLETES THE CIRCUITS ON ITS SENSOR BOARDS' PRINTED LEADS. THE SENSOR BOARD ACTS AS A VARIABLE RESISTOR THAT WILL CHANGE FROM 100K OHMS WHEN WET TO 2M OHMS WHEN DRY. IN SHORT, THE WETTER THE BOARD THE MORE CURRENT THAT WILL BE CONDUCTED.

Working Principle of Raindrop Sensor

Raindrop sensor is basically a board on which nickel is coated in the form of lines. It works on the principal of resistance. When there is no rain drop on board. Resistance is high so we gets high voltage according to V=IR. When rain drop present it reduces the resistance because water is conductor of electricity and presence of water connects nickel lines in parallel so reduced resistance and reduced voltage drop across it.

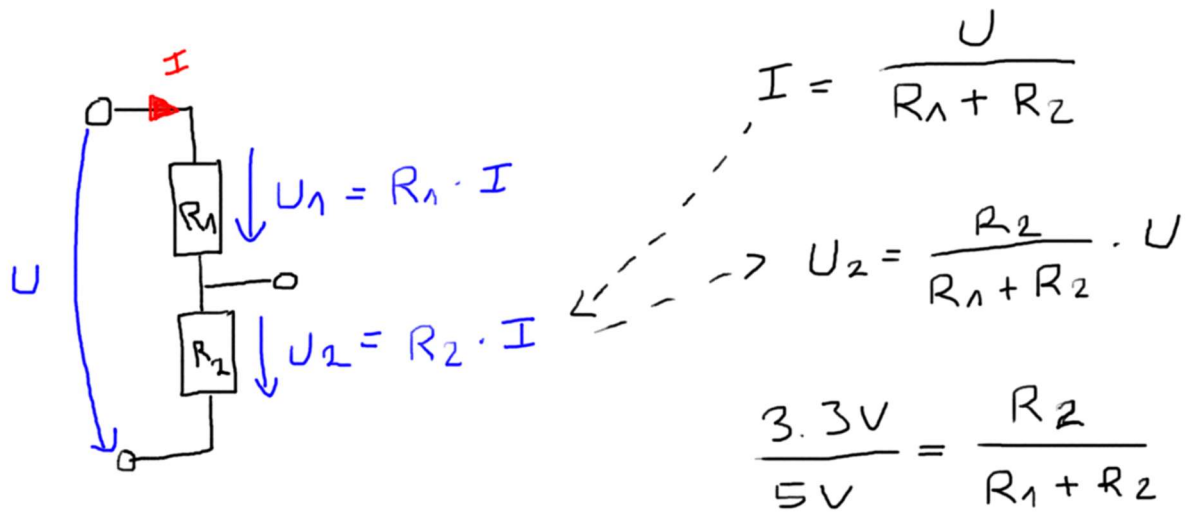
Raindrop Sensor

The analog output is used in detection of drops in the amount of rainfall. Connected to 3,3Vpower supply, the LED will turn Off when induction board has no rain drop, and output is Low. When dropping a little amount water, output is High, the switch indicator will turn on. Brush off the water droplets,

and when restored to the initial state, outputs high level.When no rain digital output is 1 and analog output gives 1023 max value. When rain is present digital output is 0 and analogue output is much less than 1023.

Functionality of a Voltage Divider

The following picture shows a voltage divider consisting of two resistors. In our case R2 is the resistor of the rain board. R1 is a reference resistor with a known resistance. From the equation you see that if we know the supply voltage U, we can calculate the voltage drop over the rain board.



But where do we find the voltage divider? This function has among other things the control board.

Functionality of the Control Board of a Rain Sensor

The control board consists of two input pins and four output pins. The input pins are connected to the rain board and the output pins to your favorite microcontroller, for example an Arduino Uno or an ESP32 NodeMCU.

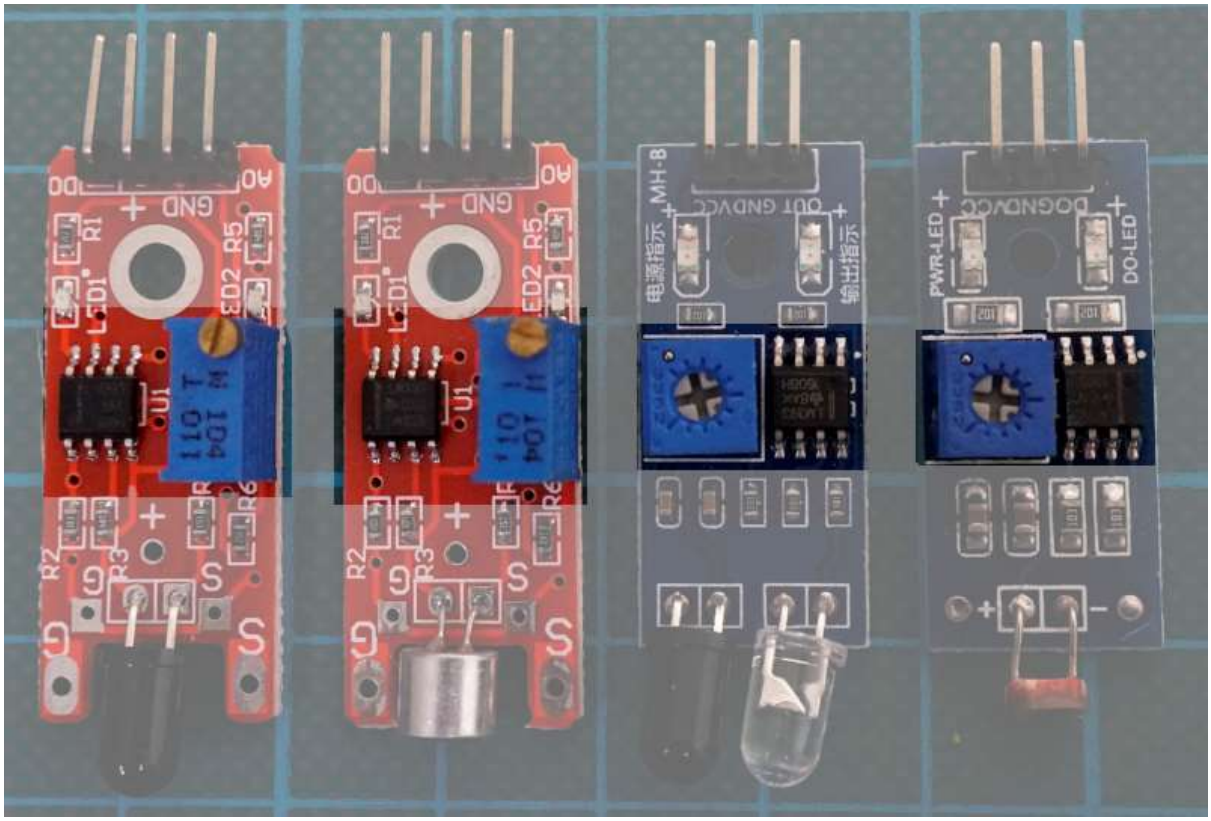
On the control board you find multiple resistors that also functions are the voltage divider to provide an analog signal for the rain

intensity. Therefore as input we get a resistance from the rain board and the control board converts this resistance into a voltage drop between the analog pin and ground. The microcontroller uses the internal analog to digital converter (ADC) to convert the voltage from the analog pin to a digital value between 0 and 1023 that can be printed to the serial output in your Arduino IDE.

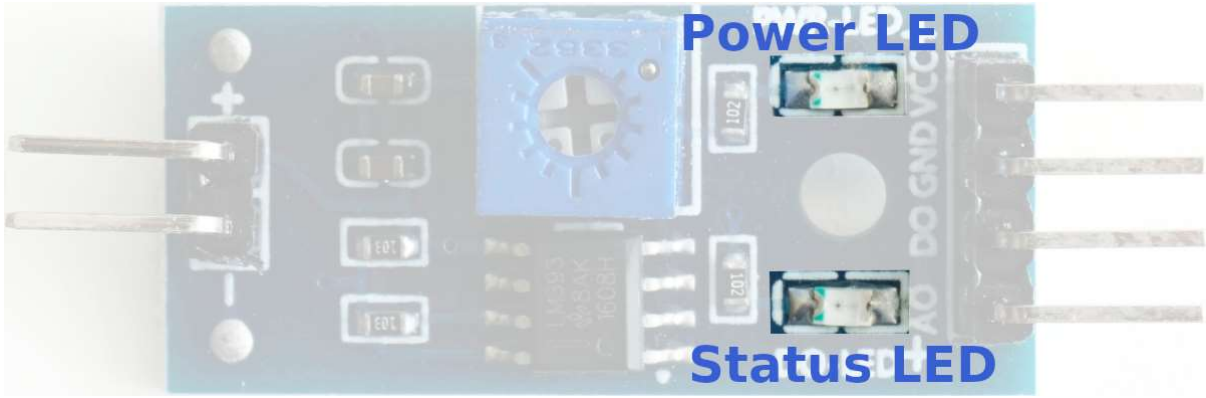
The biggest part on the control board is the potentiometer to adjust the sensitivity of the rain detector. The potentiometer is only a variable resistor whose resistance is changed with the setting wheel at the top. We need this potentiometer to compare the resistance of the potentiometer with the resistance of the rain board. If the resistance of the rain board is lower than the threshold, defined by the potentiometer, the digital output of the control board changes from 1 HIGH to 0 LOW.

But who does this comparison between the two resistances? This is done by the LM393 comparator because the voltage drop over both resistors is linear to their resistance due to ohms law. The LM393 consists of two independent precision voltage comparators and is specially designed to operate from a single power supply, in our case the microcontroller. You find more details about the LM393 on the website of Texas Instruments.

From the following picture you see that the combination of a potentiometer and the LM393 comparator is very often used for different kinds of sensors where you have to convert an analog signal to a digital signal via the threshold of the potentiometer.



The control board has also a build in LEDs that indicates the power status of the board and the status of the threshold. If the digital output of the control board changes from HIGH to LOW, indicating that it is raining, the status LED of the control board turns on.

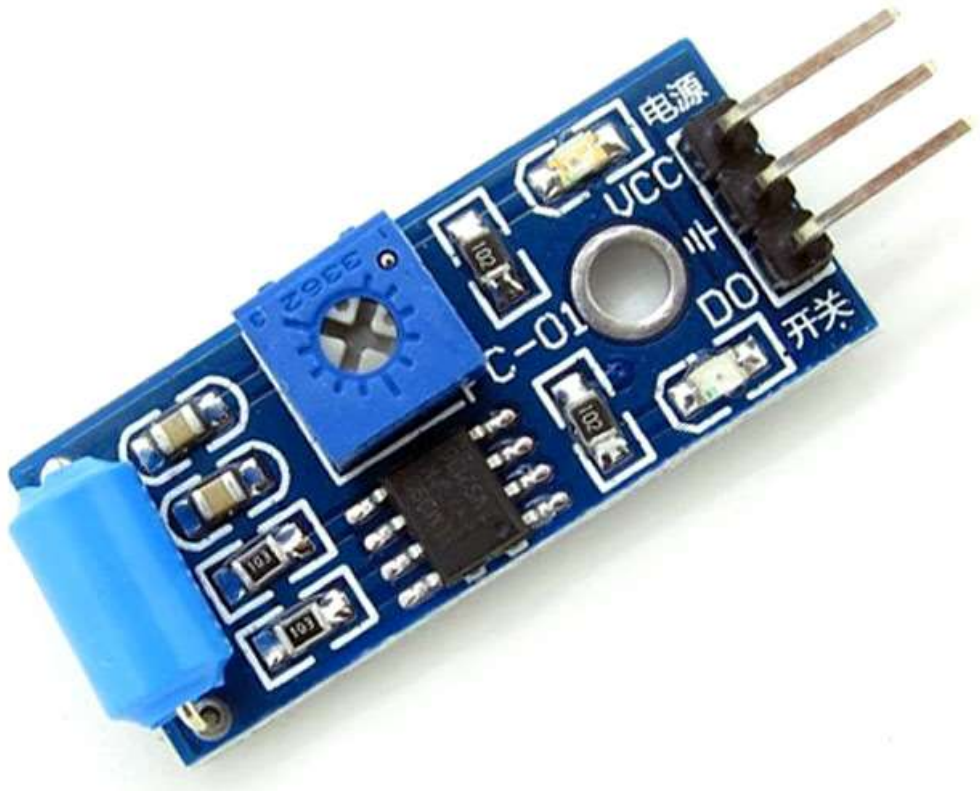


Applications

- This sensor is used as a water preservation device and this is connected to the irrigation system to shut down the system in the event of rainfall.

- This sensor is used to guard the internal parts of an automobile against the rainfall as well as to support the regular windscreen wiper's mode.
- This sensor is used in specialized satellite communications aerials for activating a rain blower over the opening of the aerial feed, to get rid of water droplets from the mylar wrap to keep pressurized as well as dry air within the waveguides.

Vibration Sensor SW-420



THE VIBRATION SENSOR MODULE BASED ON THE **VIBRATION SENSOR SW-420** AND COMPARATOR LM393 IS USED TO DETECT VIBRATIONS. THE THRESHOLD CAN ADJUST USING AN ON-BOARD POTENTIOMETER. DURING NO VIBRATION, THE SENSOR PROVIDES LOGIC LOW AND WHEN THE VIBRATION IS DETECTED, THE SENSOR PROVIDES LOGIC HIGH.

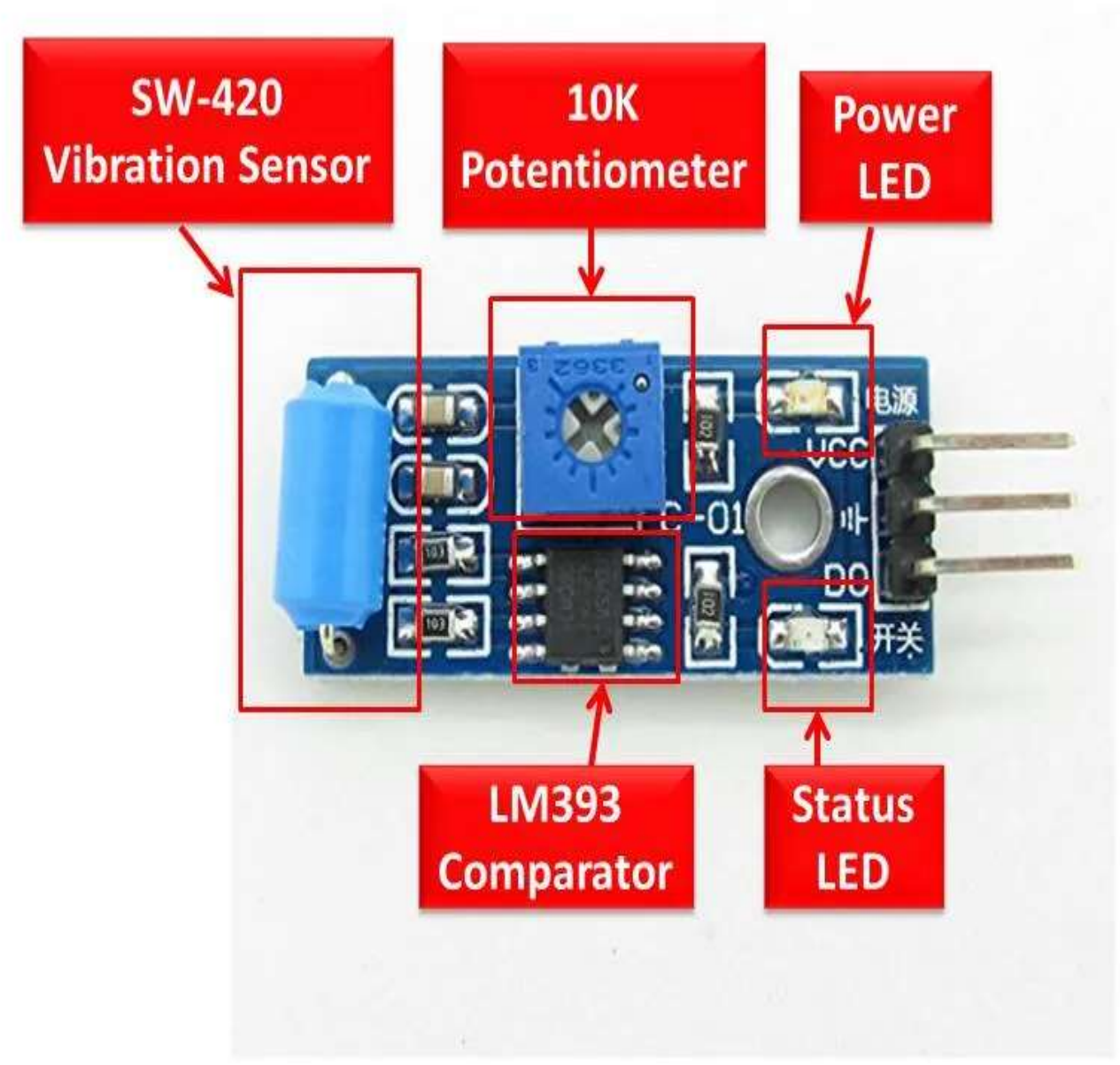
THIS VIBRATION SENSOR MODULE CONSISTS OF AN SW-420 VIBRATION SENSOR, RESISTORS, CAPACITOR, POTENTIOMETER, COMPARATOR LM393 IC, POWER, AND STATUS LED IN AN INTEGRATED CIRCUIT. IT IS USEFUL FOR A VARIETY OF SHOCKS

TRIGGERING, THEFT ALARM, SMART CAR, AN EARTHQUAKE ALARM, MOTORCYCLE ALARM, ETC.

he SW-420 vibration sensor module is integrated with the SW-420 vibration sensor, LM393 voltage comparator, a potentiometer, current limiting resistors to act as voltage dividers, therefore, control the current and capacitors as biasing elements and for noise filtering.

LM393 Voltage Comparator IC

It is a high-precise integrated circuit that compares the reference voltage and the input vibration signal. Pin 2 of the LM393 IC is connected to the adjustable potentiometer while its pin 3 is connected to the vibration sensor. The IC compares both the voltages and passes them to the digital output pin in the form of binary states.



Potentiometer

A 10K potentiometer is placed on the module to adjust the sensitivity of the sensor. The sensitivity can either be increased or decreased depending on the requirement. It is done by setting a preset or a threshold value which is given to the LM393 as a reference to compare.

SW-420 Vibration Switch

The SW-420 vibration switch senses the magnitude of the vibration in its environment. It responds to the exposed vibration either through the opening or closing of the electrical contact. The trigger switch can be an electromechanical or relay or semiconductor component.

Inbuilt LEDs

The module has two LEDs. One is to light up when the module is energized and the other is to indicate the digital output.

SW-420 Vibration Sensor Pinout

The SW-420 vibration sensor module is available in 3.2cm x 1.4cm dimensions. The pinout of the Vibration Sensor module is as shown:

SW-420 Pin Configuration

It has three headers for interfacing with any microcontroller. The pin configuration in tabular are detailed below:

Pin Name	Function
VCC	Positive power supply pin. It gives power to the sensor.
GND	Ground connection pin
D0	Digital output pin. It passes the digital output of the builtin comparator circuit.

APPLICATIONS OF VIBRATION SENSOR MODULE

- SHOCKS TRIGGERING
- THEFT ALARM
- SMART CAR
- EARTHQUAKE ALARM
- MOTORCYCLE ALARM

Display the data on the website:

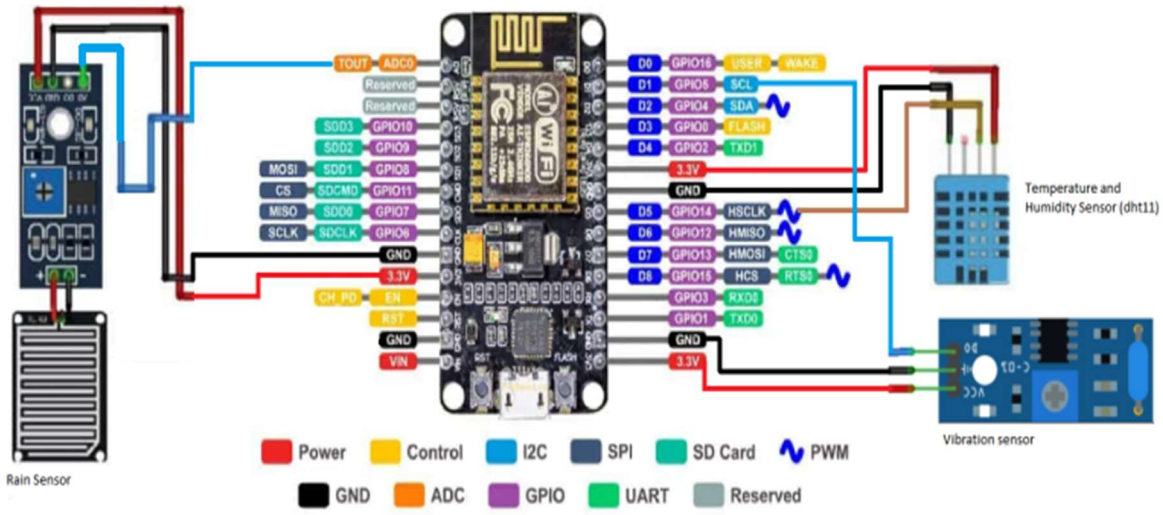
CONNECT THE NODEMCU ESP12E WITH THE LAPTOP AND CHOOSE THE BOARD AND PORT CORRECTLY AND THEN HIT THE UPLOAD BUTTON.

MAKE SURE YOUR LAPTOP OR SMARTPHONE SHARE SAME WI-FI NETWORK AS THE NODEMCU. AFTER UPLOADING THE CODE, OPEN THE SERIAL MONITOR. MAKE THE BAUD RATE OF SERIAL MONITOR AS 115200. YOU WILL SEE THE IP ADDRESS IN THE MONITOR, JUST COPY THIS IP AND PASTE IT IN THE BROWSER.

YOU WILL SEE A WEBPAGE IN YOUR BROWSER AS SHOWN BELOW. THIS PAGE WILL BE REFRESH AUTOMATICALLY AFTER EVERY 10 SECONDS. YOU CAN CHANGE THIS TIME IN CODE.

CIRCUIT DIAGRAM:

BELOW IS THE CIRCUIT DIAGRAM FOR MAKING LIVE WEATHER SATION MONITORING USING NODEMCU. THE CIRCUIT IS SHOWN IN THE FIGURE BELOW:



ARDUINO CODE:

Weather_station.ino

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

```
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <Wire.h>

#include "index.h" //Our HTML webpage contents with javascripts
#include "DHTesp.h" //DHT11 Library for ESP

#define LED 2 //On board LED
#define DHTpin 14 //D5 of NodeMCU is GPIO14
#define vibsensorPin  5 //D1 of NodeMCU is GPIO5

DHTesp dht;

//SSID and Password of your WiFi router
const char* ssid = "Redmi K20 Pro";
const char* password = "c08a0309f2a0";

ESP8266WebServer server(80); //Server on port 80

void handleRoot() {
String s = MAIN_page; //Read HTML contents
server.send(200, "text/html", s); //Send web page
}

float humidity, temperature;

void handleADC() {
char status;
double T,P,p0,a;
double Tdeg, Tfar, phg, pmb;

humidity = dht.getHumidity();
temperature = dht.getTemperature();

int rain = analogRead(A0);
rain = map(rain,1024,0,0,100);

long vib = pulseIn(vibsensorPin, HIGH);

String data =
"{\"Vibration\":\""+String(vib)+"\",\"Rain\":\""+String(rain)+"\",
```

```
\"Temperature\":\""+ String(temperature) +"\", \"Humidity\":\""+
String(humidity) +"\"}";
digitalWrite(LED,!digitalRead(LED)); //Toggle LED on data request
ajax
server.send(200, "text/plane", data); //Send ADC value, temperature
and humidity JSON to client ajax request

delay(dht.getMinimumSamplingPeriod());

Serial.print("V:");
Serial.println(vib);
Serial.print("H:");
Serial.println(humidity);
Serial.print("T:");
Serial.println(temperature); //dht.toFahrenheit(temperature));
Serial.print("R:");
Serial.println(rain);

}

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

// dht11 Sensor

dht.setup(DHTpin, DHTesp::DHT11); //for DHT11 Connect DHT sensor to
GPIO 17
pinMode(LED,OUTPUT);

pinMode(LED, OUTPUT);
pinMode(vibsensorPin, INPUT);

WiFi.begin(ssid, password); //Connect to your WiFi router
Serial.println("");

// Wait for connection
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
```

```
//If connection successful show IP address in serial monitor
Serial.println("");
Serial.print("Connected to ");
Serial.println(ssid);
Serial.print("IP address: ");
Serial.println(WiFi.localIP()); //IP address assigned to your ESP

server.on("/", handleRoot); //Which routine to handle at root
location. This is display page
server.on("/readADC", handleADC); //This page is called by java
Script AJAX

server.begin(); //Start server
Serial.println("HTTP server started");
}

void loop()
{
server.handleClient(); //Handle client requests
}
```

INDEX.H

```
const char MAIN_page[] PROGMEM = R"=====(
<!DOCTYPE html>
<html>
<head>
<title>Weather Station</title>
</head>
<style>
@import url(https://fonts.googleapis.com/css?family=Montserrat);
@import
url(https://fonts.googleapis.com/css?family=Advent+Pro:400,200);
*{margin: 0;padding: 0;}

body{
  background:#544947;
  font-family:Montserrat,Arial,sans-serif;
}
h2{
  font-size:14px;
```

```
}
.widget{
  box-shadow:0 40px 10px 5px rgba(0,0,0,0.4);
  margin:100px auto;
  height: 330px;
  position: relative;
  width: 500px;
}

.upper{
  border-radius:5px 5px 0 0;
  background:#f5f5f5;
  height:200px;
  padding:20px;
}

.date{
  font-size:40px;
}
.year{
  font-size:30px;
  color:#c1c1c1;
}
.place{
  color:#222;
  font-size:40px;
}
.lower{
  background:#00A8A9;
  border-radius:0 0 5px 5px;
  font-family:'Advent Pro';
  font-weight:200;
  height:130px;
  width:100%;
}
.clock{
  background:#00A8A9;
  border-radius:100%;
  box-shadow:0 0 0 15px #f5f5f5,0 10px 10px 5px rgba(0,0,0,0.3);
  height:150px;
  position:absolute;
  right:25px;
```

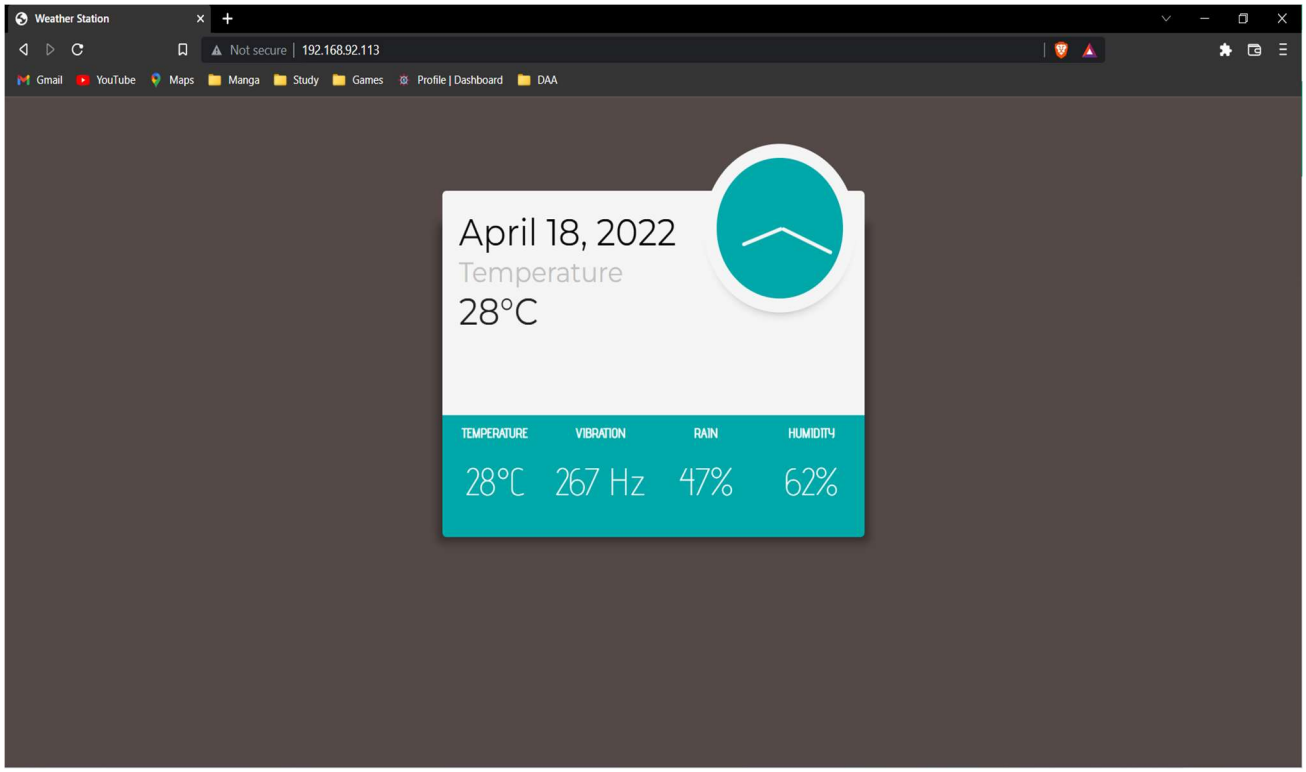
```
  top:-35px;
  width:150px;
}

.hour{
  background:#f5f5f5;
  height:50px;
  left:50%;
  position: absolute;
  top:25px;
  width:4px;
}

.min{
  background:#f5f5f5;
  height:65px;
  left:50%;
  position: absolute;
  top:10px;
  transform:rotate(100deg);
  width:4px;
}

.min,.hour{
  border-radius:5px;
  transform-origin:bottom center;
  transition:all .5s linear;
}

.infos{
  list-style:none;
}
.info{
  color:#fff;
  float:left;
  height:100%;
  padding-top:10px;
  text-align:center;
  width:25%;
}
.info span{
  display: inline-block;
```

```
  font-size:40px;
  margin-top:20px;
}
.weather p {
  font-size:20px;padding:10px 0;
}
.anim{animation:fade .8s linear;}

@keyframes fade{
  0%{opacity:0;}
  100%{opacity:1;}
}

a{
 text-align: center;
 text-decoration: none;
 color: white;
 font-size: 15px;
 font-weight: 500;
}
</style>
<body>


<div class="widget">
  <div class="clock">
    <div class="min" id="min"></div>
    <div class="hour" id="hour"></div>
  </div>
  <div class="upper">
    <div class="date" id="date">21 March</div>
    <div class="year">Temperature</div>
    <div class="place update" id="temperature">23 &deg;C</div>
  </div>
  <div class="lower">
    <ul class="infos">
      <li class="info temp">
        <h2 class="title">TEMPERATURE</h2>
        <span class='update' id="temp">21 &deg;C</span>
      </li>
      <li class="info weather">
        <h2 class="title">VIBRATION</h2>
```

```
        <span class="update" id="vibration">0 mb</span>
      </li>
      <li class="info wind">
        <h2 class="title">RAIN</h2>
        <span class='update' id="rain">0%</span>
      </li>
      <li class="info humidity">
        <h2 class="title">HUMIDITY</h2>
        <span class='update' id="humidity">23%</span>
      </li>
    </ul>
  </div>
</div>

<script>
setInterval(drawClock, 2000);

function drawClock(){
    var now = new Date();
    var hour = now.getHours();
    var minute = now.getMinutes();
    var second = now.getSeconds();

    //Date
    var options = {year: 'numeric', month: 'long', day: 'numeric' };
    var today  = new Date();
    document.getElementById("date").innerHTML =
today.toLocaleDateString("en-US", options);

    //hour
    var hourAngle = (360*(hour/12))+((360/12)*(minute/60));
    var minAngle = 360*(minute/60);
    document.getElementById("hour").style.transform =
"rotate("+(hourAngle)+"deg)";
    //minute
    document.getElementById("min").style.transform =
"rotate("+(minAngle)+"deg)";

    //Get Humidity Temperature and Rain Data
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
      if (this.readyState == 4 && this.status == 200) {
```

```
      var txt = this.responseText;

      var obj = JSON.parse(txt); //Ref:
https://www.w3schools.com/js/js_json_parse.asp
        document.getElementById("rain").innerHTML = obj.Rain + "%";
        document.getElementById("temperature").innerHTML =
Math.round(obj.Temperature) + "&deg;C";
        document.getElementById("temp").innerHTML =
Math.round(obj.Temperature) + "&deg;C";
        document.getElementById("humidity").innerHTML =
Math.round(obj.Humidity) + "%";
        document.getElementById("vibration").innerHTML =
obj.Vibration + " Hz";
      }
    };
   xhttp.open("GET", "readADC", true); //Handle readADC server on
ESP8266
   xhttp.send();
}
</script>
</body>
</html>
)=====";
```

SCREEN SHOTS OF THE OUTPUT:



REFERENCES:

[1] DHANALAXMI, B., & NAIDU, G. A. (2017, FEBRUARY). A SURVEY ON DESIGN AND ANALYSIS OF ROBUST IOT ARCHITECTURE. IN 2017 INTERNATIONAL CONFERENCE ON INNOVATIVE MECHANISMS FOR INDUSTRY APPLICATIONS (ICIMIA) (PP. 375-378). IEEE.

[2] KRISHNAMURTHI, K., THAPA, S., KOTHARI, L., & PRAKASH, A. (2015). ARDUINO BASED WEATHER MONITORING SYSTEM. INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH AND GENERAL SCIENCE, 3(2), 452-458.

CIRCUIT: