1.HEALTH SENSOR

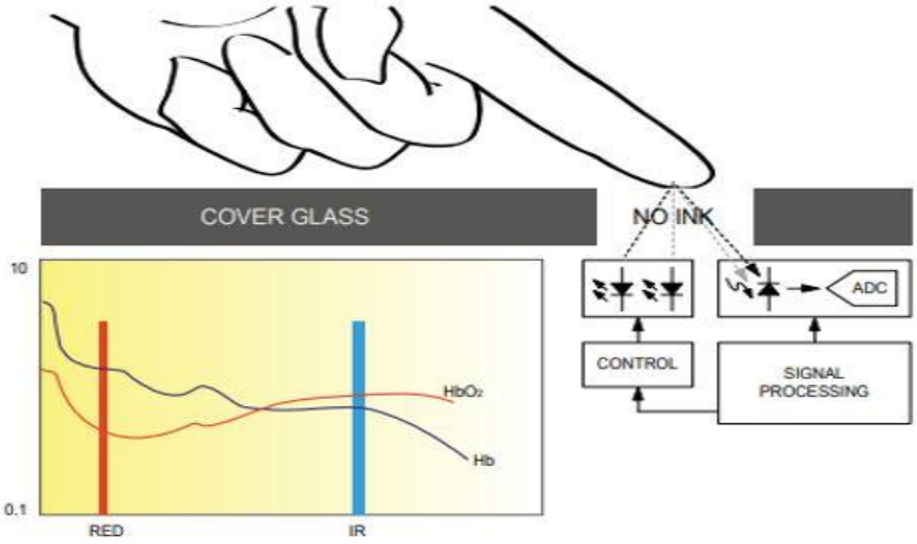
MAX30102 Sensor Working

In this section, let us discuss how the MAX30102 heartbeat monitor and pulse oximeter actually works.

Pulse Oximeter

To find the blood oxygen concentration (%), it is first important to know that inside our blood haemoglobin is responsible for carrying oxygen. When a person holds a pulse oximeter, light from the device passes through the blood in the fingers. This is used to detect the amount of oxygen by measuring the changes in light absorption in both oxygenated and deoxygenated blood.

As we already mentioned before, the MAX30102 sensor consists of two LEDs (Red and IR) and a photodiode. Both of these LEDs are used for SpO2 measurement. These two LEDs emit lights at different wavelengths, ~660nm for the red led and ~880nm for the IR LED. At these particular wavelengths, the oxygenated and deoxygenated hemoglobin have vastly different absorption properties.



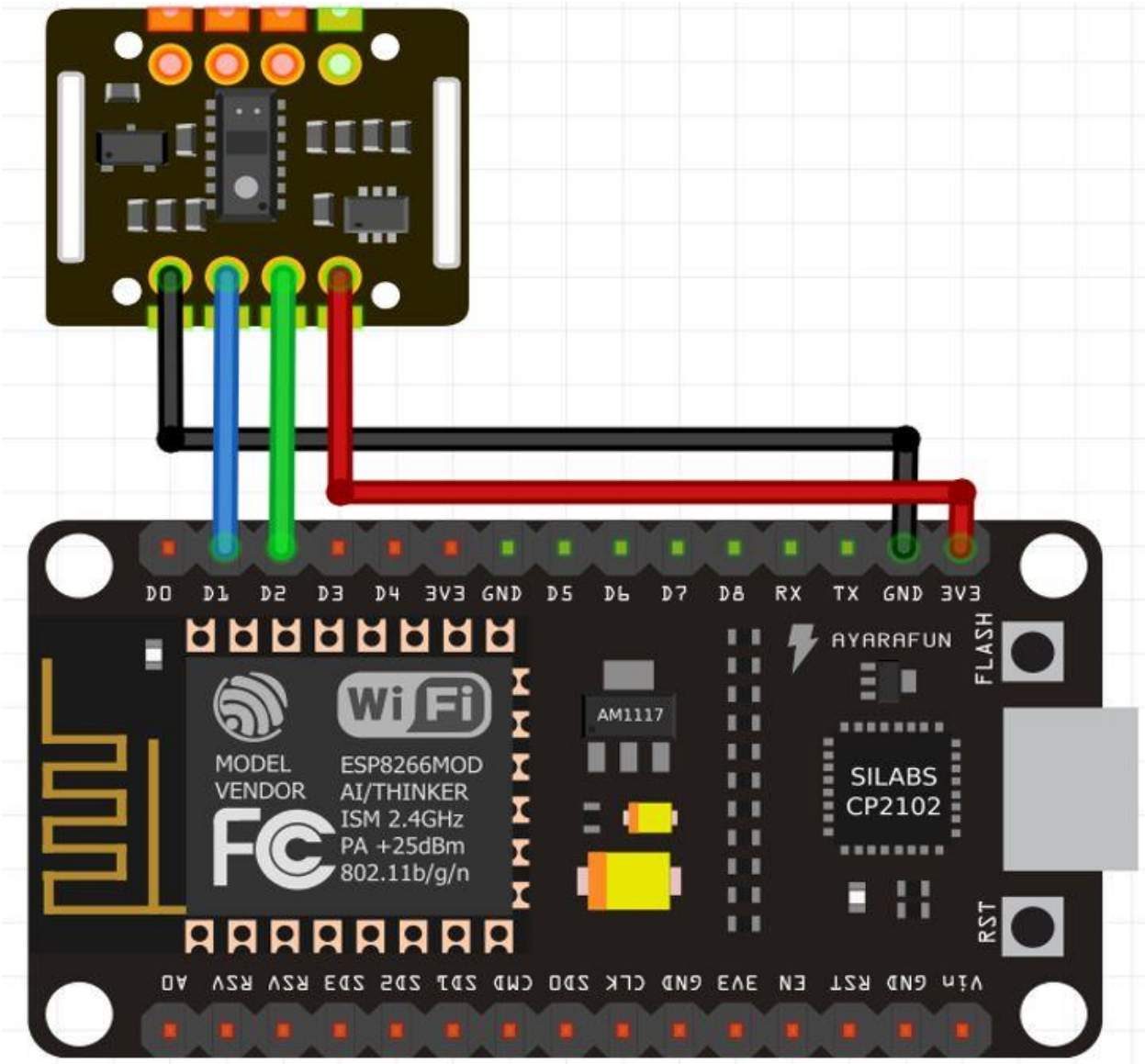
Heart Rate Measurement

To measure the heart rate, we do not require the Red LED, only the IR LED is needed. This is because oxygenated hemoglobin absorbs more infrared light.

The heartbeat rate is the ratio of time between two consecutive heartbeats. Similarly, when the human blood is circulated in the human body then this blood is squeezed in capillary tissues. As a result, the volume of capillary tissues is increased but this volume is decreased after each heartbeat. This change in volume of capillary tissues affects the infrared light of the sensor, which transmits light after each heartbeat

Interfacing MAX30102 BPM and SpO2 Sensor with ESP8266 NodeMCU

MAX30102 Module	ESP8266
VCC	3.3V
SCL	D1
SDA	D2
GND	GND



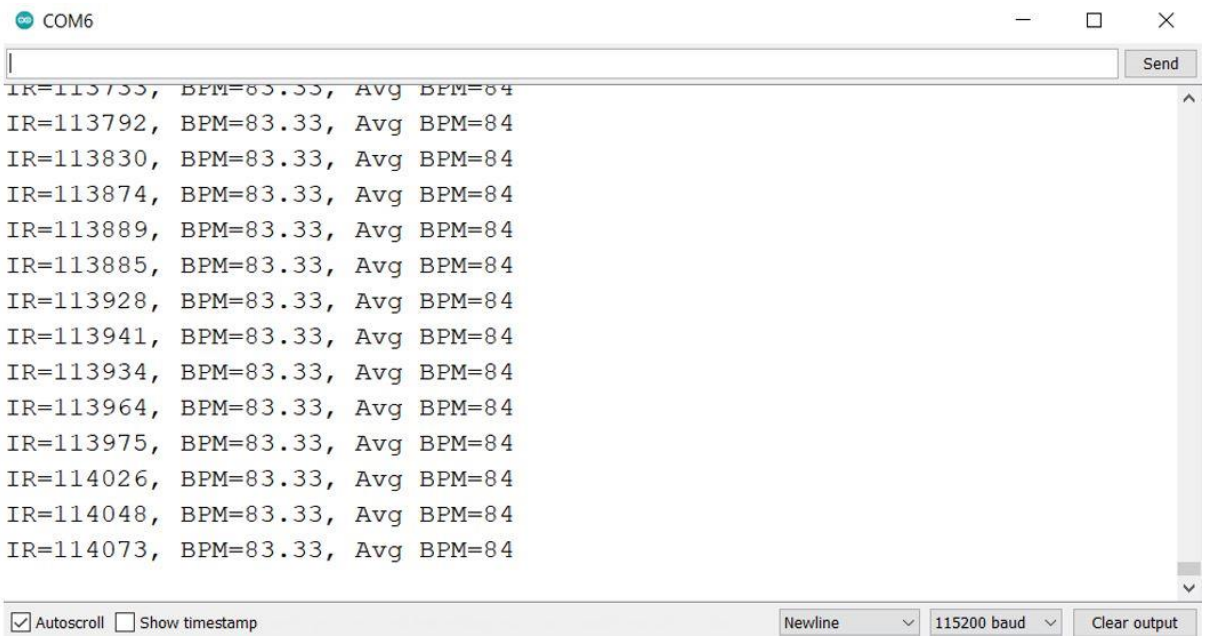
Connection Diagram

Program:

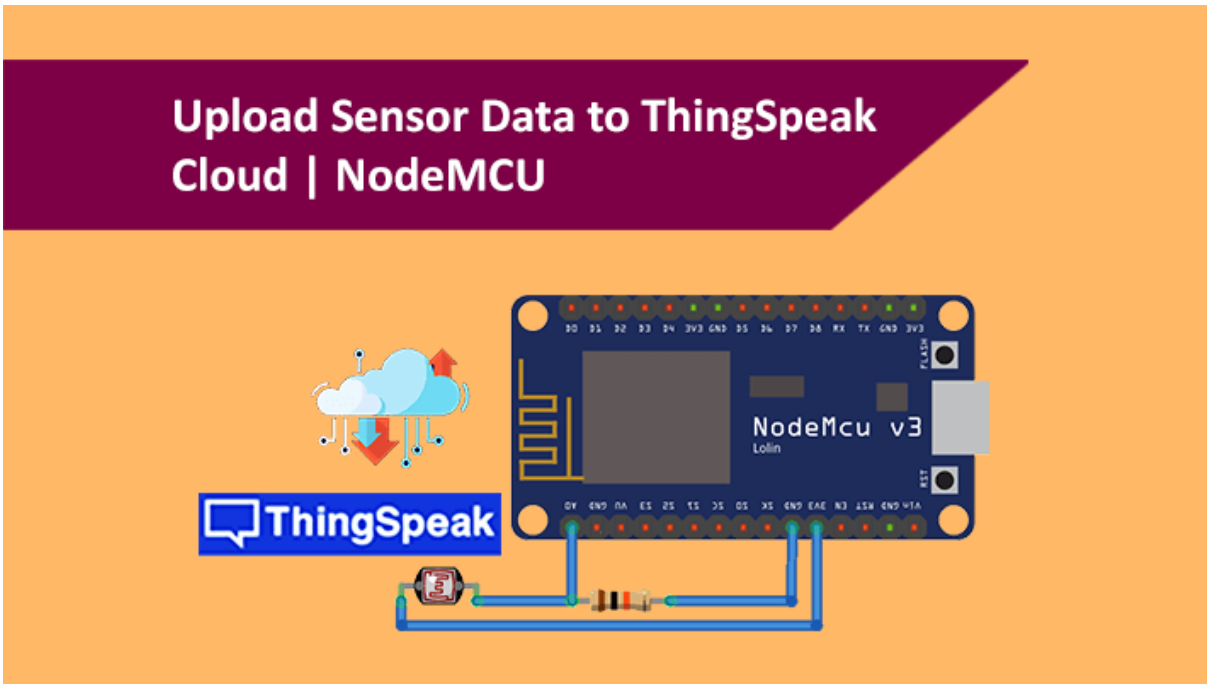
```
#include <Wire.h>
#include "MAX30105.h"
#include "heartRate.h"
MAX30105 particleSensor;
const byte RATE_SIZE = 4; //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE]; //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0; //Time at which the last beat occurred
float beatsPerMinute;
int beatAvg;

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

	// Initialize sensor
```

```
        if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use default I2C port, 400kHz speed
        {
        Serial.println("MAX30105 was not found. Please check wiring/power. ");
        while (1);
        }
  Serial.println("Place your index finger on the sensor with steady pressure.");
  particleSensor.setup(); //Configure sensor with default settings
  particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running
  particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED
}
 void loop()
{
        long irValue = particleSensor.getIR();

        if (checkForBeat(irValue) == true)
                {
                        //We sensed a beat!
                        long delta = millis() - lastBeat;
                        lastBeat = millis();
                        beatsPerMinute = 60 / (delta / 1000.0);
                if (beatsPerMinute < 255 && beatsPerMinute > 20)
                        {
                        rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array
                        rateSpot %= RATE_SIZE; //Wrap variable
//Take average of readings
                        beatAvg = 0;
                        for (byte x = 0 ; x < RATE_SIZE ; x++)
                        beatAvg += rates[x];
                        beatAvg /= RATE_SIZE;
                        }
                }

        Serial.print("IR=");
        Serial.print(irValue);
        Serial.print(", BPM=");
        Serial.print(beatsPerMinute);
        Serial.print(", Avg BPM=");
        Serial.print(beatAvg);
        if (irValue < 50000)
        Serial.print(" No finger?");
        Serial.println();
}
```

Output:



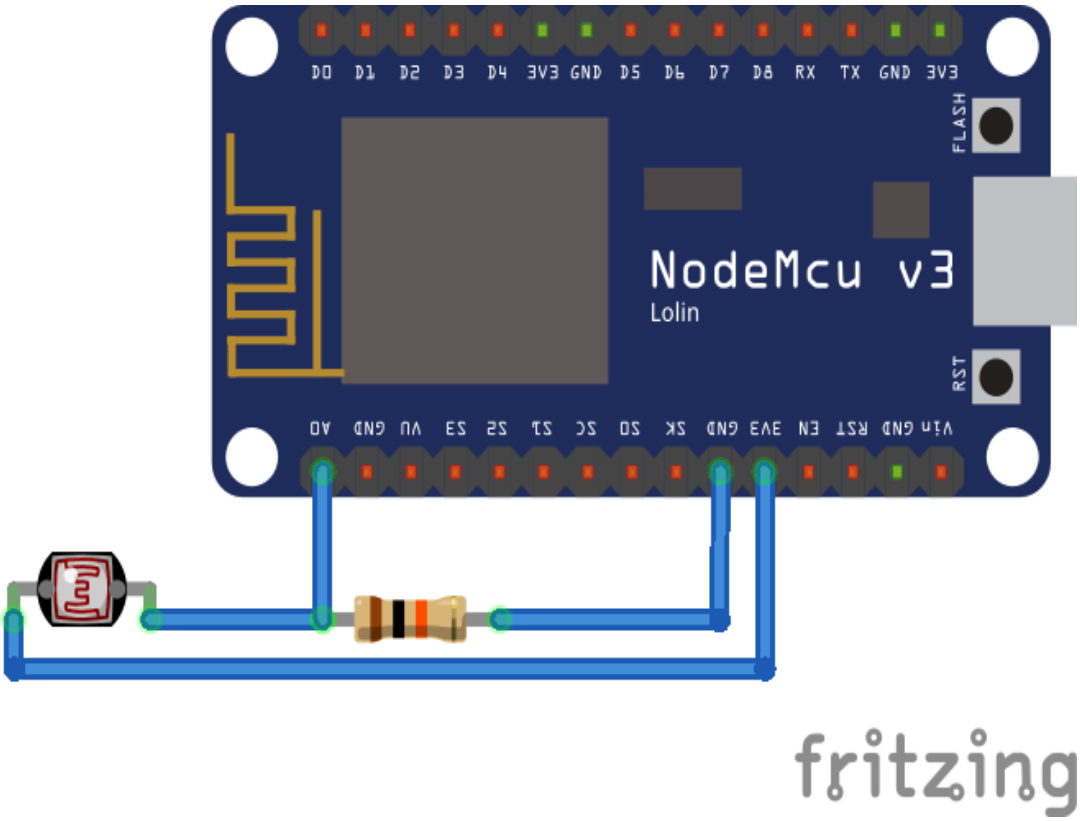
2.Upload Sensor Data to ThingSpeak using NodeMCU



In this tutorial, we will be using an LDR to plot its light Intensity level on ThingSpeak using NodeMCU. We will program the NodeMCU to read and store the LDR data into a variable and then upload it to ThingSpeak using its channel name and API key. The NodeMCU should be connected to the internet via Wi-Fi. We will see how to create ThingSpeak Channels and configure it on NodeMCU.

Hardware Required:

- NodeMCU
- LDR Module
- 10K Ohm Resistor
- Jumper Wires
- Breadboard (Optional)



Once the hardware is set up, We can go ahead and create our ThingSpeak Channel.

Step 1: Go to https://thingspeak.com/ and create your ThingSpeak Account if you don't have. Login to Your Account.

Step 2: Create a Channel by clicking 'New Channel'.

step 3: Enter the channel details.

Name: Any Name

Description: Optional

Field 1: Light Intensity LDR – This will be displayed on the analytics graph. If you need more than 1 Channels you can create for additional Sensor Data.Save this setting.

Step 4: Now you can see the channels. Click on the 'API Keys' tab. Here you will get the Channel ID and API Keys. Note this down.

Step 5: Open Arduino IDE and Install the ThingSpeak Library. To do this go to Sketch>Include Library>Manage Libraries. Search for ThingSpeak and install the library.

Step 6: Now we need to modify the program with your credentials.

In the below code you need to change your Network SSID, Password and your ThingSpeak Channel and API Keys.

Replace the following content in the code,

'*Your SSID Here*' – Your Wi-Fi Name

'*Your Password Here*' – Your Wi-Fi Password

'*YYYYYY*' – Your ThingSpeak Channel Number (without Quotes)

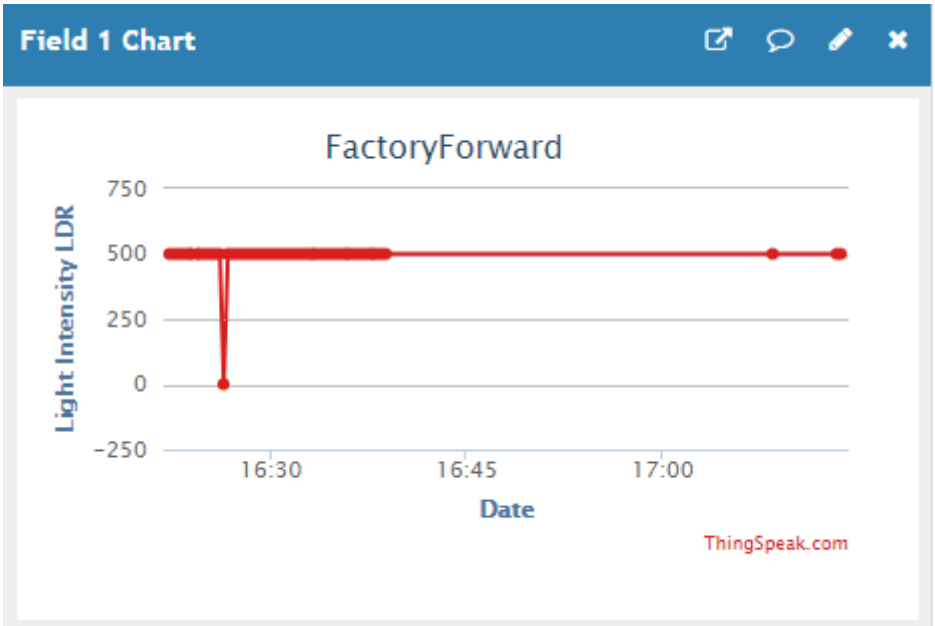
'*XXXXXXXXXX*' – Your Thing Speak API Key.

Program:

```
#include <ESP8266WiFi.h>;

#include <WiFiClient.h>;

#include <ThingSpeak.h>;

const char* ssid = "Your SSID Here"; //Your Network SSID

const char* password = "Your Password Here"; //Your Network Password

int val;

int LDRpin = A0; //LDR Pin Connected at A0 Pin

WiFiClient client;

unsigned long myChannelNumber = 123456; //Your Channel Number (Without Brackets)

const char * myWriteAPIKey = "XXXXXXXXXXXXXXX"; //Your Write API Key

void setup()

{

Serial.begin(9600);

delay(10);

// Connect to WiFi network

WiFi.begin(ssid, password);

ThingSpeak.begin(client);

}
```

```
void loop()
{
val = analogRead(LDRpin); //Read Analog values and Store in val variable
Serial.print(val); //Print on Serial Monitor
delay(1000);
ThingSpeak.writeField(myChannelNumber, 1,val, myWriteAPIKey); //Update in ThingSpeak
delay(100);
}
```

Output:



3. IoT Smart Agriculture with Automatic Irrigation System

Capacitive Soil Moisture Sensor

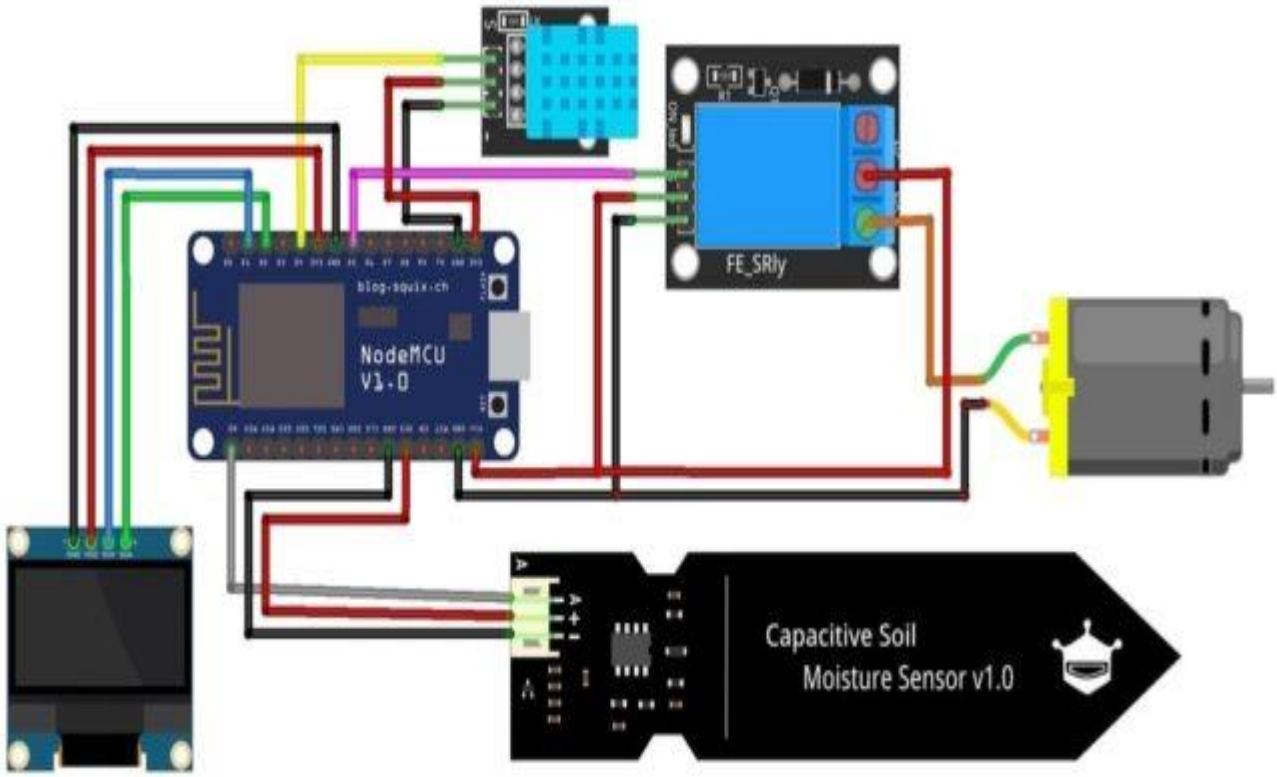
This is an **analog capacitive soil moisture sensor** which measures soil moisture levels by **capacitive sensing**. This means the capacitance is varied on the basis of water content present in the soil. You can convert the capacitance into voltage level basically from 1.2V minimum to 3.0V maximum. The advantage of Capacitive Soil Moisture Sensor is that they are made of a **corrosion-resistant material** giving it a long service life.

DC 3-6V Micro Submersible Mini Water Pump

The **DC 3-6 V Mini Micro Submersible Water Pump** is a low cost, small size Submersible Pump Motor. It operates from a **2.5 ~ 6V power supply**. It can take up to 120 liters per hour with a very low current consumption of **220mA**. Just connect the tube pipe to the motor outlet, submerge it in water, and power it.

Circuit Diagram & Connection

Let us see the schematic of the IoT Smart Agriculture & Automatic Irrigation System project. I use Fritzing to make an schematic for most of my projects. All you need is to place and connect a component that is super easy.



Connect the soil moisture sensor to A0 of Nodemcu. The motor connects to Relay. To control the relay, we use the D0 Pin of NodeMCU

Program:

```
#define BLYNK_TEMPLATE_ID "TMPLgPMR_iH7"
#define BLYNK_DEVICE_NAME "Switch"
#define BLYNK_AUTH_TOKEN "c3TFJ-CKLp7UDd5dnbSSejuNk1_wKmbc"
#define BLYNK_PRINT Serial
#define motor D0
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "mi";
char pass[] = "Sharath@123";
const int AirValue = 620;   //you need to replace this value with Value_1
const int WaterValue = 310;  //you need to replace this value with Value_2
int soilMoistureValue = 0;
int soilmoisturepercent=0;
int sensor = A0;
BlynkTimer timer;
void myTimerEvent()
        {
          Blynk.virtualWrite(V0,digitalRead(motor));
        Blynk.virtualWrite(V1,soilmoisturepercent);
        }
void setup()
        {
                Serial.begin(9600);
                pinMode(motor,OUTPUT);
                Blynk.begin(auth, ssid, pass);
```

```
                timer.setInterval(1000L, myTimerEvent);
}
void loop()
 {
          Blynk.run();
          timer.run();
          soilMoistureValue = analogRead(sensor);  //put Sensor insert into soil
          Serial.print("sensor value");
          Serial.println(soilMoistureValue);
          soilmoisturepercent = map(soilMoistureValue, AirValue, WaterValue, 0, 100);
          if(soilmoisturepercent >= 100)
          {
                    Serial.print("moisture is ");
                     Serial.println("100 %");
          }
else if(soilmoisturepercent <=0)
          {
                    Serial.print("moisture is ");
                    Serial.println("0 %");
          }
else if(soilmoisturepercent >0 && soilmoisturepercent < 100)
          {
                    Serial.print("moisture is ");
                    Serial.print(soilmoisturepercent);
                     Serial.println("%");
  }
          if(soilmoisturepercent <= 30)
```

```
        {
          digitalWrite(motor,LOW);
         Serial.println("MOTOR IS ONN");
          }
  if(soilmoisturepercent >= 80)
        {
        digitalWrite(motor,HIGH);
        Serial.println("MOTOR IS OFF");
        }
         delay(1000);
}
```