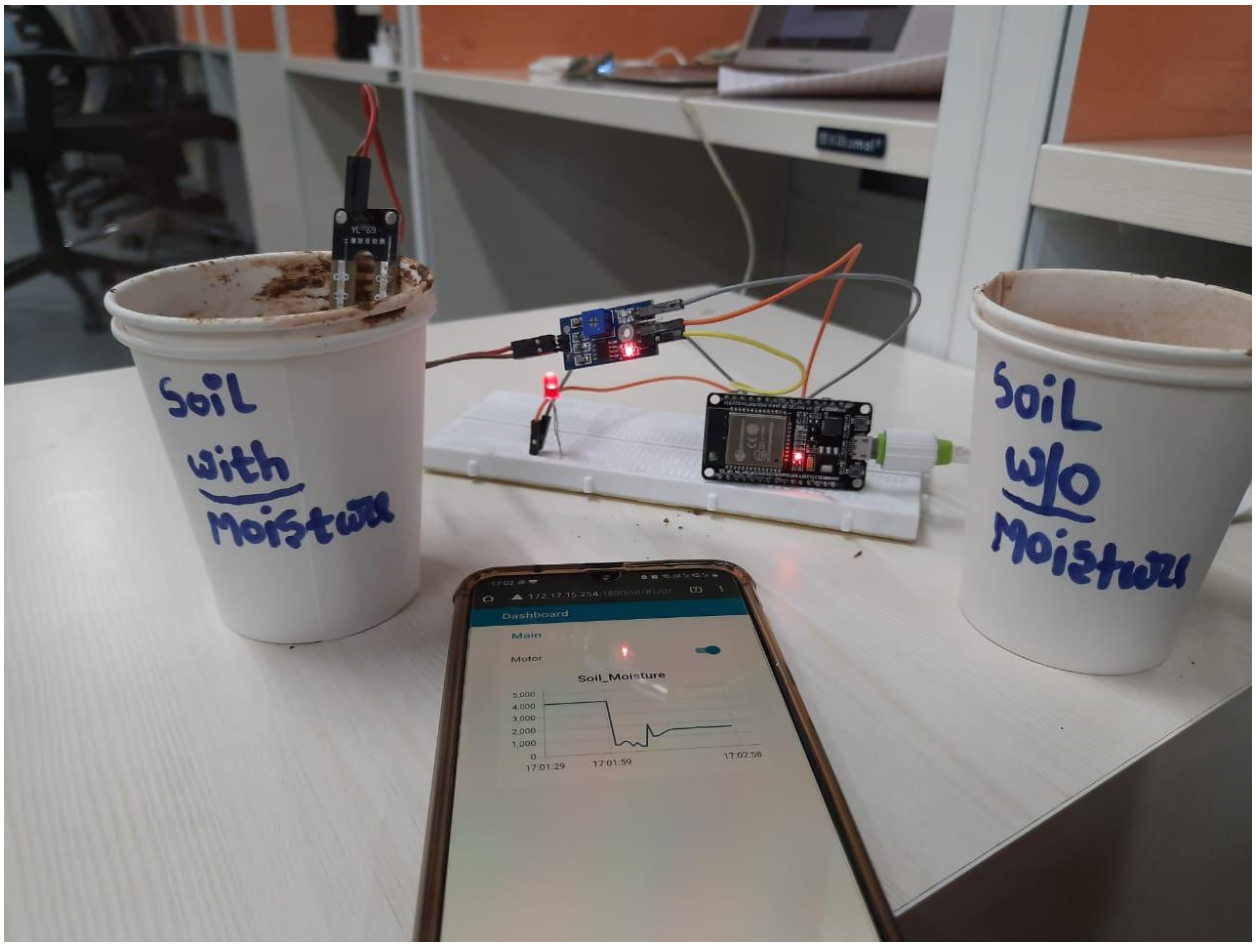
Smart Agriculture System using MQTT

Hello,
This is our IoT project which is based on the MQTT protocol.
We have built a smart agriculture system which senses the soil moisture data in the soil and displays it in graph form on our smart phone so that we can turn on/off the motor.

Project video link:
https://drive.google.com/file/d/1szn2ybsCA_tsbvLAwr8LRfOp3k_tIyB4/view?usp=sharing



Components used in the the project are:-

1.Resberry pi:- we use raspberry pi for our local server or as our MQTT broker by installing Mosquitto on our Raspberry pi.

2.ESP32 board:- we use ESP32 as it already has a built-in wifi in it. Esp32 is our client-1 which publishes data in the topic name ‘soil_moisture’ on our broker and it also subscribes to the topic name ‘Motor_State’.

3.Node-Red:- Node red is nothing but a flow based development tool which is also installed on our raspberry and used as client-2.It publishes the current Motor state(on/off) onto the topic name ‘Motor_state’.It is subscribed to the topic name ‘soil_moisture’.

And Soil_moisture sensor,jumper wires.

This whole project is working on the IITH network, So all the devices which are connected through the IIT network can access the soil_moisture data and give commands to turn on or off the Motor.

You can see the live working of our project in the attached video:-

https://drive.google.com/file/d/1szn2ybsCA_tsbvLAwr8LRfOp3k_tIyB4/view?usp=sharing

Below is the code explanation and step by step guide on how we built this project.

Make Raspberry pi as as a broker(locally):

To install the Mosquitto Broker on Raspberry type the below commands on the raspberry terminal:-

pi@raspberry:~ $ sudo apt-get update
(this command update the raspberry)

pi@raspberry:~ $ sudo apt install -y mosquitto mosquitto-clients(this command install mosquitto client on our raspberry pi).

pi@raspberry:~ $ sudo systemctl enable mosquitto.service

(This command is to make Mosquitto auto start on boot up enter)

Node-Red dashboard as our client-2:(we used node-red dashboard asv)

Node-RED is a powerful open source tool for building Internet of Things (IoT) applications with the goal of simplifying the programming component.

It uses visual programming that allows you to connect code blocks, known as nodes, together to perform a task.

The nodes when wired together are called flows.

Getting Node-RED installed in your Raspberry Pi is quick and easy. It just takes a few commands.

Having an SSH connection established with your Raspberry Pi, enter the following commands to install Node-RED:

pi@raspberry:~ $ bash <(curl -sL https://raw.githubusercontent.com/node-red/raspbian-deb-package/master/resources/update-nodejs-and-nodered)

pi@raspberry:~ $ sudo systemctl enable nodered.service

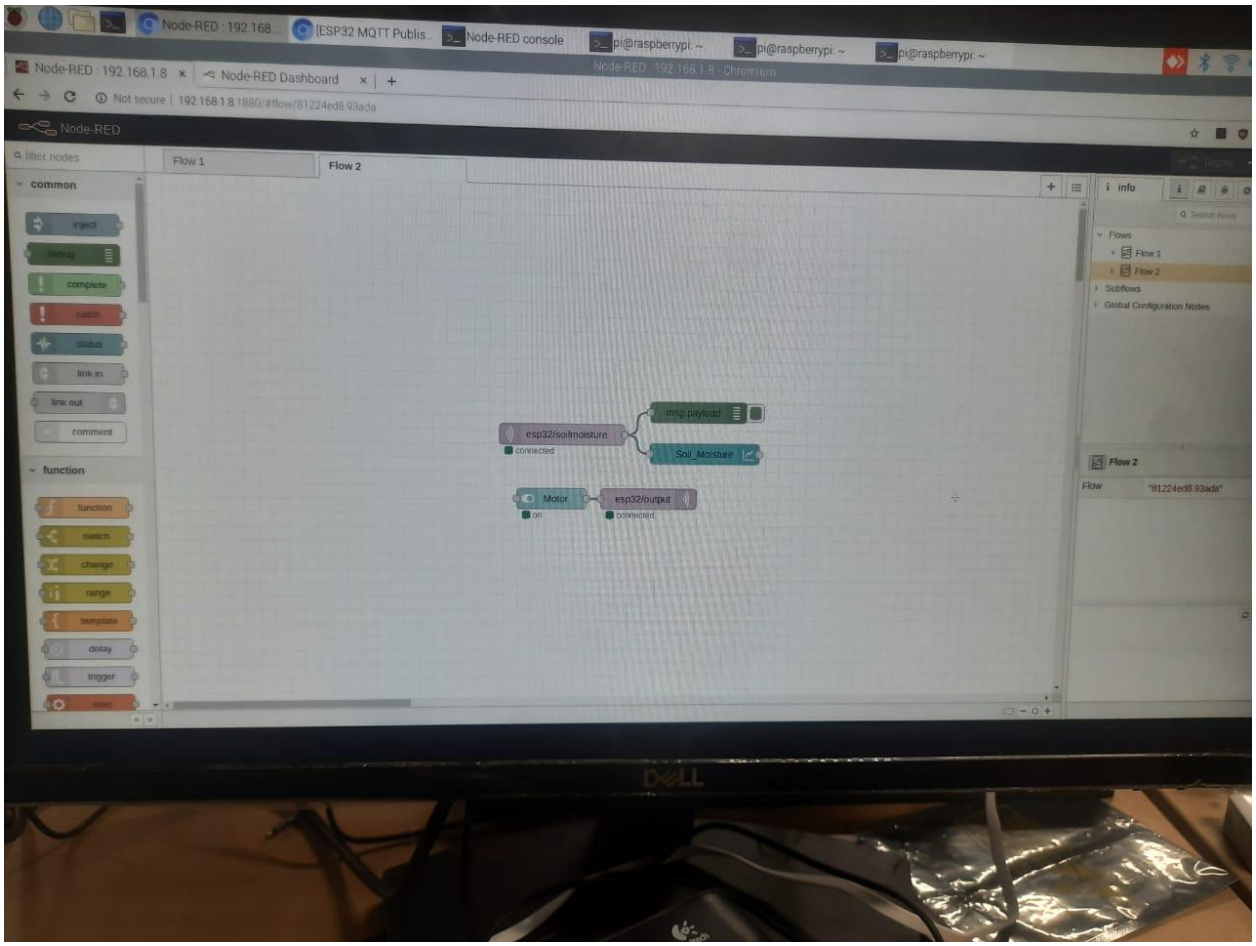
(This command make node-red autostart on boot)

After that reboot your raspberry so that all the installation gets permanent and the effect shows.

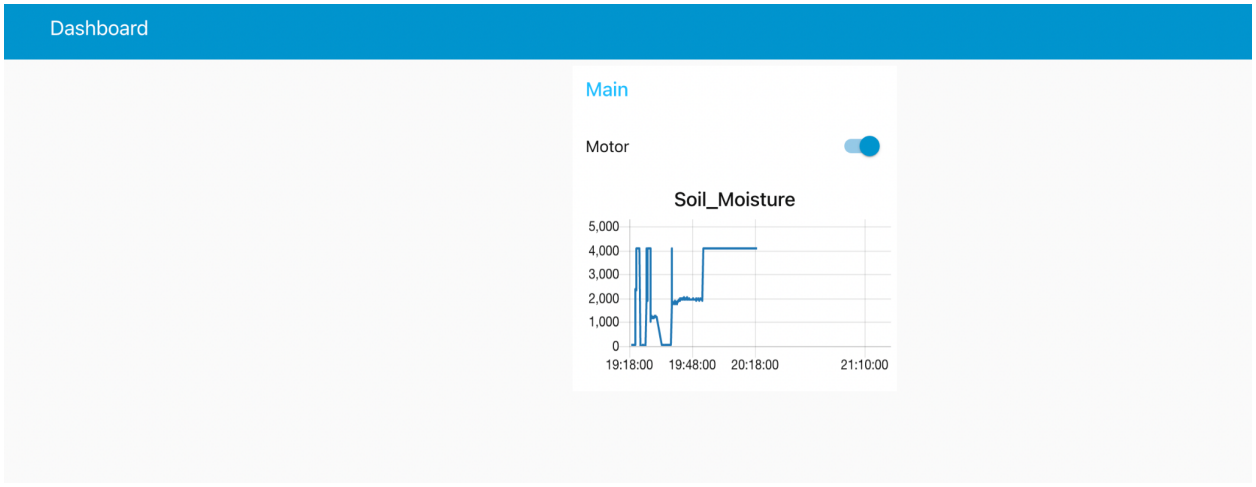
When your Pi is back on, you can test the installation by entering the IP address of your Pi in a web browser followed by the 1880 port number:

To get the ip of your raspberry pi you can type the command ifconfig in your terminal.

This will open node_red gui where we can build our graphical interface of your Node-Red dashboard that shows the Soil moisture graph and give us a Toggle button for Motor on and off.



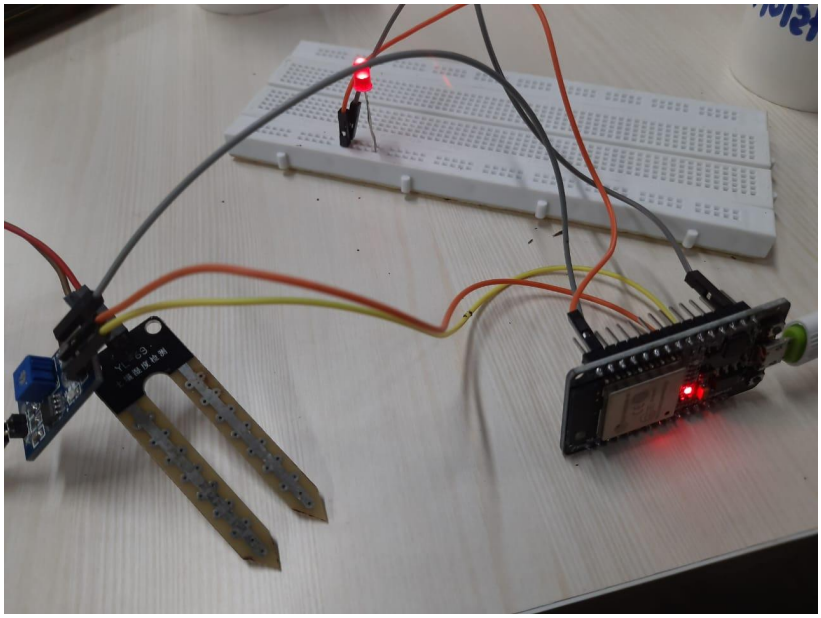
After creating it looks like this..



Esp-32 as our client-1:

we use ESP32 DEVKIT V1, we mention the full model because your ESP might have different pin numbers.

So, the connection is as follows:-



Esp32 pins:	Soil_moisture sensor:
vin	vcc
Grn	Gnd
D32(analog input)	A0

ESP32 pins:	LED
D22(digital output pin)	Positive LED pin
Gnd	Gnd

ARDUINO CODE:

```
#include <WiFi.h>

//This library allows you to send and receive MQTT messages

#include <PubSubClient.h>

#include <Wire.h>



//ssid and password of our router

char*ssid = "IITH-Guest-PWD-IITH@2019";

const char* password = "IITH@2019";



//broker ip in our case we made local broker using raspberry pi so it's ip of our raspberry pi

//You can get the ip of raspberry by writing ifconfig command in the raspberry terminal

const char*  mqtt_server = "172.17.15.254";



//we will declare an object of class WiFiClient, which allows to establish a connection to a specific IP and port

WiFiClient espClient;

//We will also declare an object of class PubSubClient, which receives as input of the constructor the previously defined WiFiClient.

PubSubClient client(espClient);
```

```
//our soil moisture gives analog output to esp32 on pin32. I got an error on pin13 becoz wifi max limit exceed.

int sensor_pin=32;

//output_value is the analog output that our esp32 get

int soil_moisture;

// LED Pin

const int ledPin = 22;



void setup() {

  Serial.begin(9600);

  setup_wifi();

  client.setServer(mqtt_server, 1883);

  client.setCallback(callback);



  pinMode(ledPin, OUTPUT);

}



void setup_wifi() {
```

```
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

```
void callback(char* topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
  }
  Serial.println();

  // If a message is received on the topic esp32/output, you check if the message is either "on" or "off".
  if (String(topic) == "esp32/output") {
    Serial.print("Changing output to ");
    if(messageTemp == "on"){
      Serial.println("on");
      digitalWrite(ledPin, HIGH);
```

```
    }
    else if(messageTemp == "off"){
      Serial.println("off");
      digitalWrite(ledPin, LOW);
    }
  }
}


void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      // Subscribe
      client.subscribe("esp32/output");
    } else {
      Serial.print("failed, rc=");
```

```
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

    soil_moisture=analogRead(sensor_pin);
    //Serial.println(soil_moisture);
    Serial.print("Soilmoisture: ");
    Serial.println(soil_moisture);
    //sm=(((soil_moisture)/40.95)-100.0)*-1.0;
    //Serial.print(sm);
```

```
  //Serial.println("%");
  char soilmoisture_String[8];
  dtostrf(soil_moisture, 1, 2, soilmoisture_String);
  client.publish("esp32/soilmoisture", soilmoisture_String);
  delay(2000);



}
```