DIY HARDWARE PROJECT REPORT

CS578: IOT

IoT based Smart Agriculture Monitoring System

Akshat Rana
November 3, 2020

1 PURPOSE OF THE PROJECT

The main goal of this project is to design an IoT based smart agriculture monitoring system that makes use of sensor networks that collects data from different sensors and sends it through the wireless protocol.

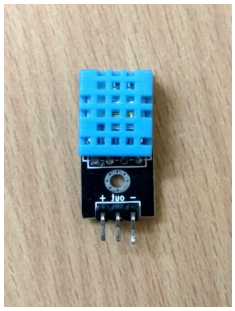
2 COMPONENTS USED

- Digital Temperature Humidity Sensor
- Water Level Sensor
- Soil sensor
- Arduino Uno
- NodeMCU
- Breadboard
- LCD Display
- Mini Exhaust Fan
- DC motor

- Smartphone
- Relays
- Jumper wires, cables
- Potentiometer-10K ohm, Resistors

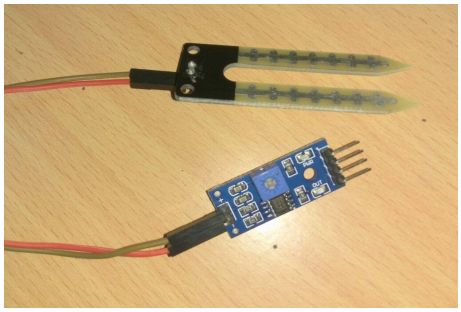
3 IMPLEMENTED ATTRIBUTES

- **Temperature and Humidity sensor:**



Temperature and humidity are essential factors in deciding when crops and fruits will get ready to cultivate or begin to produce. This parameter is measured by a digital sensor called DHT11, which can measure both temperature and humidity. Humidity is calculated by means of measuring the conductivity of liquid substrate that alters with an exchange in humidity and temperature is calculated by the usage of a thermistor. The function called read() is used to take readings from the sensor, which is included in the library.

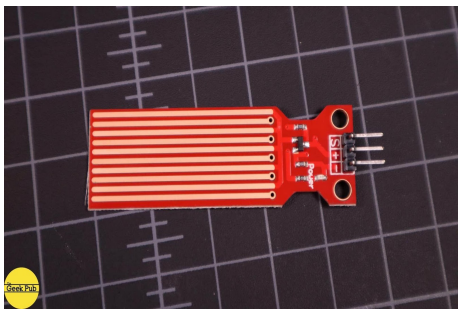
- **Soil Moisture Sensor:**



The amount of water to be irrigated every day also varies; this depends on how well the soil can hold moisture, current season, temperature and humidity. The soil moisture can be measured using the illustrated sensor, which has two prongs (electrodes) which are to be inserted on the top layer of soil. This is an analog sensor which will output analog values to Arduino.

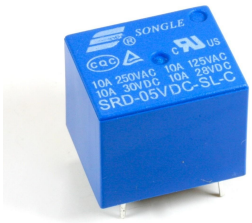
We are going to use only the analog output of this sensor, just like other analog sensors mentioned here; the output is converted to a 10-bit digital value and finally to percentage out of 100.
0% means the soil is dry 100% indicates the soil wet. But anywhere more than 50% indicates that the soil is wet enough.

- **Water Level Sensor:**



Pure water is not conductive. It's actually the minerals and impurities in water that makes it conductive. Ultimately, we're just reading the resistance of the impurities in your water. The more of the traces that are bridged, the easier current can pass.
Reading the sensor output with an Analog input pin on the Arduino will result in an integer between 0 and 500 with typical public water supplies.

- **Relay:**

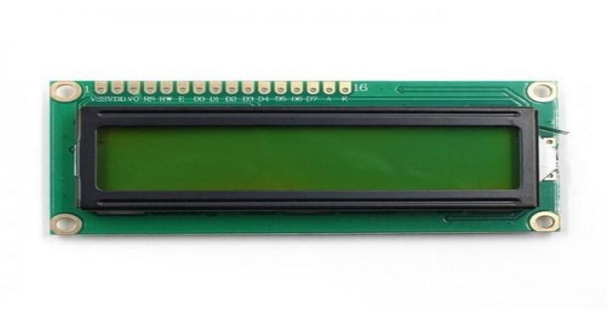


Relay It is a switching device. To mechanically control a switch many of the relays use an electromagnet, but some other fundamentals can also be used like solid-state relays. When it is crucial to operating a circuit by way of independent low power signal or if different circuits are managed by means of a single signal, then relays are used. So relay acts as an automated switch that operates on the circuit having high current using a low current signal.
In the project relay is used to turn ON & OFF the motor and the Exhaust fan according to the signal received by the Arduino Uno. .

- **DC Motor:**
When a current-carrying conductor is placed in a magnetic field, it experiences a torque

and has a tendency to move. In other words, when a magnetic field and an electric field interact, a mechanical force is produced. The DC motor or direct current motor works on that principle. It works on 3V/5V DC supply.
A DC motor is used here in this project to replicate the function of a water pump. So if the moisture sensor senses that there is no water in the soil, then the pump will turn on.

- **16x2 LCD Display**

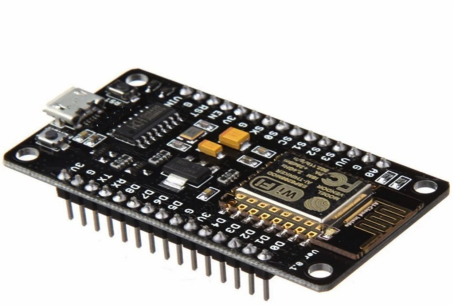


It has 16 pins, and the first one from left to right is the Ground pin. The second pin is the VCC which connects the 5 volts pin on the Arduino Board. Next is the Vo pin on which we can attach a potentiometer for controlling the contrast of the display. Next, The RS pin or register select pin is used for selecting whether we will send commands or data to the LCD. For example, if the RS pin is set on low state or zero volts, then we are sending commands to the LCD like: set the cursor to a specific location, clear the display, turn off the display and so on. And when the RS pin is set on High state or 5 volts, we are sending data or characters to the LCD.
LCD displays the water level, temperature, humidity and soil moisture at all times.

- **DC Exhaust fan**

When the temperature crosses 35°C temperature, then the fan automatically turns on. It uses a solid-state relay so is silent in operation and more durable than using a mechanical relay.

- **NodeMCU:**

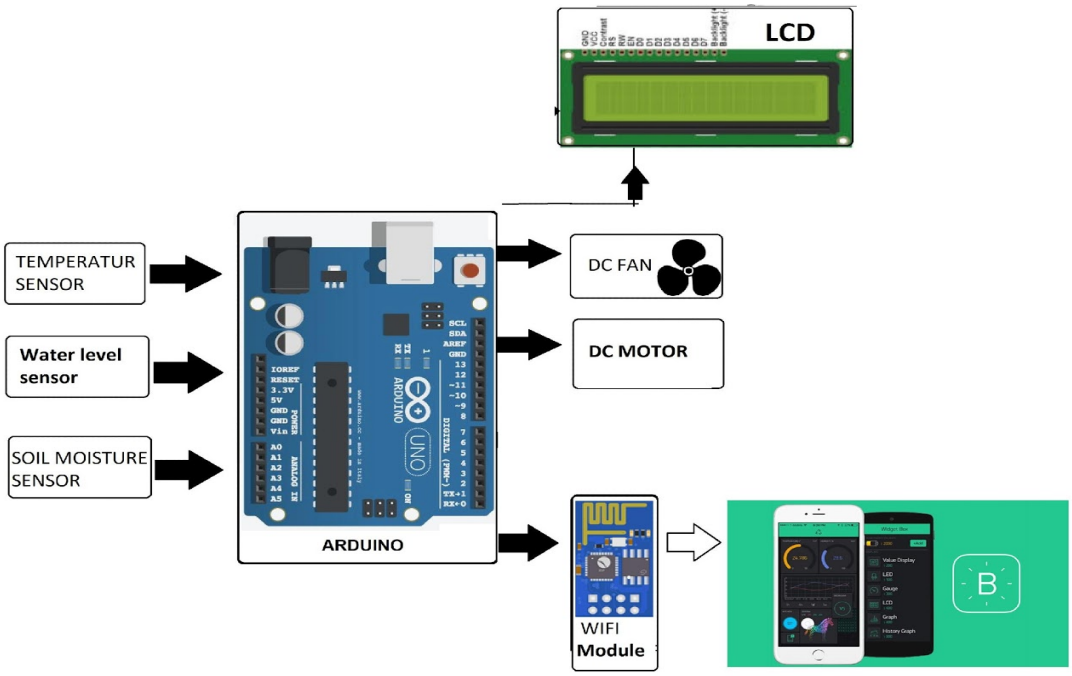


It is an open sources firmware and development kits to build IoT products. It includes firmware that runs on ESP8266 WiFiSoC and hardware that has an ESP-12 module. The kit has an analog(A0). It also has digital (D0-D8) pins on the board. It even assists serial ports communications such as SPI, UART, I2C etc. It supports the Arduino C programming language. It is used to establish a Wifi connection with the smartphone and also receives data from the Arduino through serial communication. **serial Communication** *For serial communication a single string is send from the arduino containing the temperature,humidity moisture and water level. The string is then decoded after being received at the nodeMCU to get the separate values.*

- **Blynk App:**

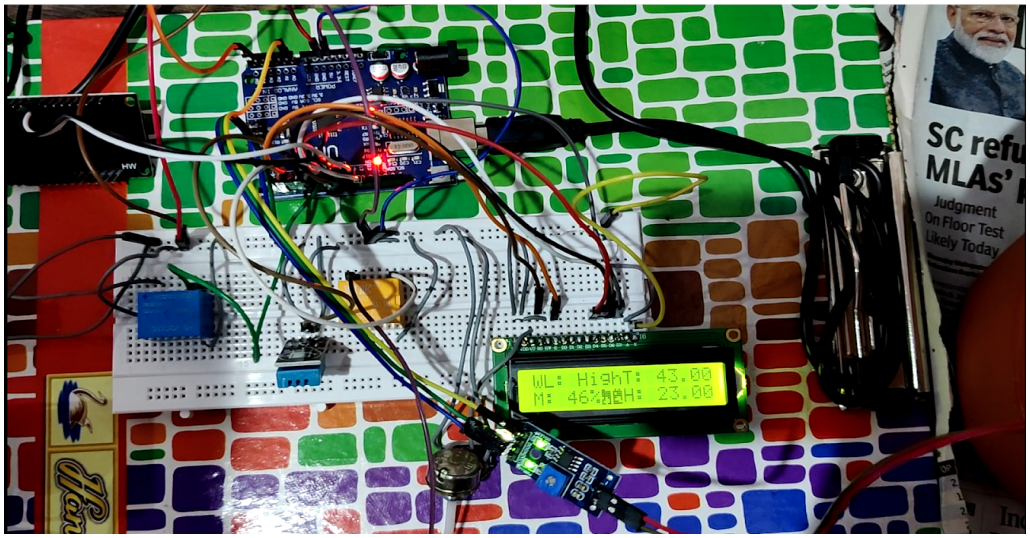
It was designed for IoT. This app has the capacity to remotely control hardware and also shows sensor information. This app also helps to visualise and store data. Blynk Server establishes a communication network between smartphone and hardware. All incoming and outgoing commands are processed and also enables the communication between server and process

4 BLOCK DIAGRAM

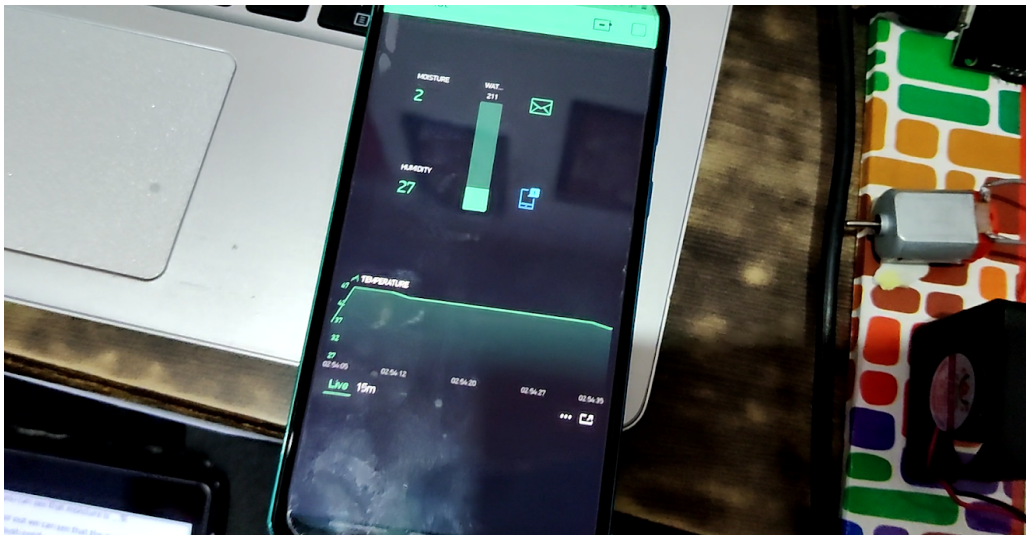


5 SAMPLE OUTPUTS

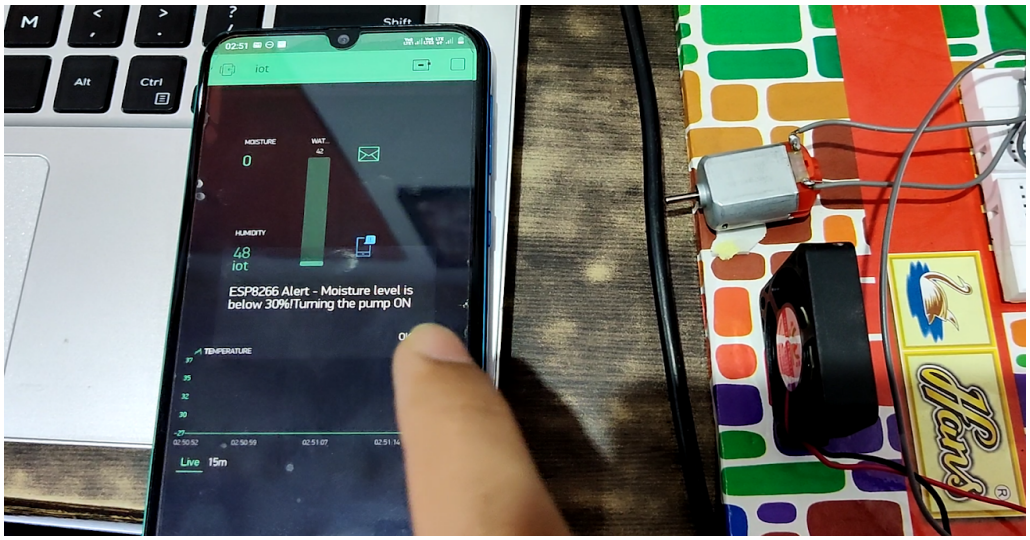
1. **LCD Display**

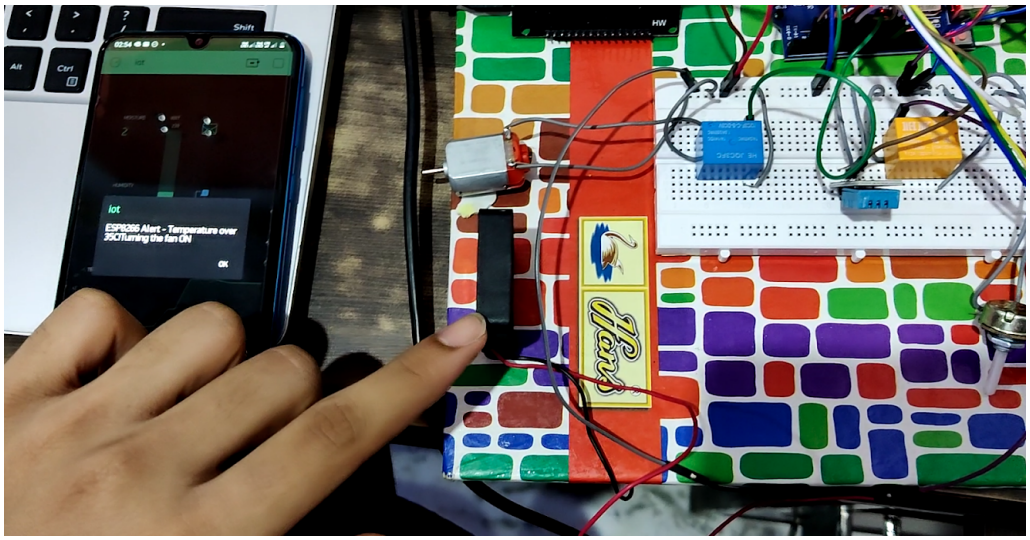


2. **Blynk app**

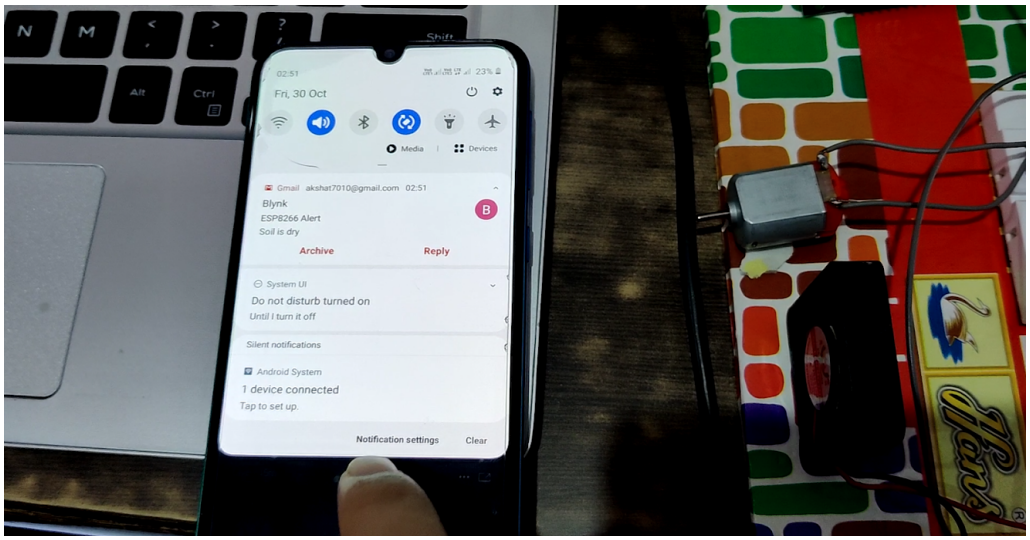


3. **Notification alert**





4. **Email alert**



5. **Automatic functioning**

6 CODES

- **ARDUINO CODE**

```
//Arduino code
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>//for displaying the LED Dsplay
#include <dht.h>
dht DHT;
SoftwareSerial s(11,10);//serial communication established on digital 11 and 10 pin
 int resval = 0;  // holds the value
int respin = A5; // sensor pin used
int sensor_pin = A0;//for soil sensor
int output_value ;
LiquidCrystal lcd(2,3,4,5,6,7);
void setup() {
  s.begin(115200);
  Serial.begin(115200);
  pinMode(9, OUTPUT);//connected to DC motor
  pinMode(12, OUTPUT);//connected to fan
  lcd.begin(16, 2);
}
void loop() {
  resval = analogRead(respin); //Read data from analog pin and
  //store it to resval variable
  lcd.setCursor(0, 0);
    lcd.clear();
  if (resval<=100){ lcd.print("WL: Empty");}
  else if (resval>100 && resval<=135){ lcd.print("WL: Low"); }
  else if (resval>135 && resval<=170){ lcd.print("WL: Medium"); }
//if water level is high then turn off the motor
  else if (resval>170){   lcd.print("WL: High"); digitalWrite(9, LOW);}
  if(DHT.temperature>35)//if temperature goes above 35 then turn on the fan
      digitalWrite(12, HIGH);
  else
      digitalWrite(12, LOW);
    int chk = DHT.read11(8);// pin 8 is used for DHT11
  Serial.print("Temperature = ");
  Serial.println(DHT.temperature);
  Serial.print("Humidity = ");
  Serial.println(DHT.humidity);
   output_value= analogRead(sensor_pin);
   output_value = map(output_value,550,0,0,100);
   if(output_value<0)
```

```
    output_value=0;
   if(output_value <30 && resval<100)//if soil is dry then turn on the motor
    {
      digitalWrite(9, HIGH);
    }
   Serial.print("Moisture : ");
   Serial.print(output_value);
   Serial.println("%");
   String str =String(DHT.temperature)+"&"+
   String(DHT.humidity)+"&"+
   String(output_value)+"&"+
   String(resval)+"&";
//send all the data in string format
   if(s.available()>0)
  {
   s.print(str);
  }
//the below lines prints on the lcd display
lcd.print("T: ");
lcd.print(DHT.temperature);
 lcd.setCursor(0, 1);
lcd.print("M: ");
   lcd.print(output_value);
   lcd.println("%");
lcd.print("H: ");
lcd.print(DHT.humidity);
  delay(3000);
}
```

- **NODEMCU CODE**

```
    //nodemcu code
#include <ESP8266WiFi.h>
#include <SoftwareSerial.h>//for serial communication
#include <BlynkSimpleEsp8266.h>//for enabling blynk app
char auth[] = "your authentication id";
char ssid[] = "proj";//my wifi id & password
char pass[] = "cool7010";
BlynkTimer timer;
SoftwareSerial s(D6,D5);//communication on D6,D5 pins
String data;
//function to separate the data from the
//string received through serial communication
String getValue(String data, char separator, int index)
```

```
{
    int found = 0;
    int strIndex[] = { 0, -1 };
    int maxIndex = data.length() - 1;

    for (int i = 0; i <= maxIndex && found <= index; i++) {
        if (data.charAt(i) == separator || i == maxIndex) {
            found++;
            strIndex[0] = strIndex[1] + 1;
            strIndex[1] = (i == maxIndex) ? i+1 : i;
        }
    }
    return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}
//function to send the data to blynk app
void sendSensor()
{
  s.write("s");
  if (s.available()>0)
    data=s.readString();
  String xval = getValue(data, '&', 0);
String yval = getValue(data, '&', 1);
String zval = getValue(data, '&', 2);
String wval = getValue(data, '&', 3);
  int h = xval.toFloat();//temperature
  int t = yval.toFloat();//humidity
  int m = zval.toFloat();//moisture
  int l = wval.toFloat();//water level
  Blynk.virtualWrite(V5, t);
  Blynk.virtualWrite(V6, h);
  Blynk.virtualWrite(V2, m);
  Blynk.virtualWrite(V3, l);
    // Send EMAIL
  // and PUSH Notification
  if(h > 35){
  //add your e-mail below
    Blynk.email("example@gmail.com", "ESP8266 Alert", "Temperature over 35C!");
    Blynk.notify("ESP8266 Alert - Temperature over 35C!Turning the fan ON");
  }
    if(m <30 && l<100)
    {
    //change it with you email
   Blynk.email("example@gmail.com", "ESP8266 Alert", "Soil is dry");
    Blynk.notify("ESP8266 Alert - Moisture level is below 30%!Turning the pump ON");
```

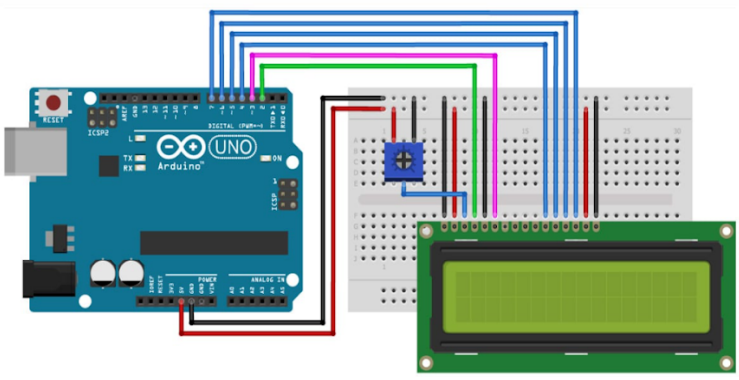
```
    }
}
}
void setup()
{
  s.begin(115200);//establish communication at 115200 baud rate
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
  timer.setInterval(1000L, sendSensor);//timer at the interval of 1 sec
}

void loop()
{
  Blynk.run();
  timer.run();
}
```

7 USER MANUAL

Follow the steps below for setting up the system:

1. Install the following libraries before running the code:
   - Liquid crystal
   - Blynk
   - DHT sensor library
   - esp8266
2. Go to preferences and paste the following URL under the Additional board manager tab: http://arduino.esp8266.com/stable/package_esp8266com_index.json
3. Connect the 16x2 LCD as follows:



Adjust the contrast with the help of 10k ohm potentiometer.

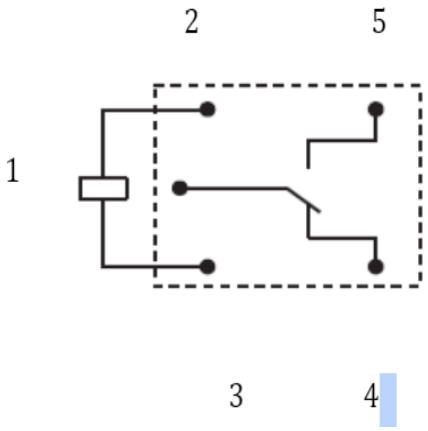
4. All the related pins have been commented out. So, if you want to use different pins (e.g.:- if you want to use the 12,13 digital pins of the Arduino instead of the 11,10 pin for serial communication) then change it in the code.

5. Output pins:
   pin 9 -> Relay->DC Motor
   pin 12 -> Relay->Fan
   you can change it in the code and the hardware configuration if you want.

6. For water sensor A5 is used.
   For Soil moisture sensor A0 is used.
   For DHT11 digital pin 8 is used.

7. Use baud rate 115200 or above as the string transmitted is large.

8. Connect the relay as follows:



   Connect 1 to +5V line
   Connect 3 to the output pin
   Connect 2 to ground
   Connect 4 to the one terminal of the motor
   Connect the other terminal of the motor to ground
   Leave 5 as it is.

9. Open the Blynk app and generate your unique user authentication key.

10. Replace the ssid and password in the code with that of your mobile hotspot and also replace the authentication id.

11. Change the Email in the code with your Email address.

12. Open the Blynk app and use the value displays, super chart and Level H/Level V to see the data. Use any virtual pins of your choice. Make changes in the following:

Blynk.virtualWrite(V5, t);//humidity
Blynk.virtualWrite(V6, h);//temperature
Blynk.virtualWrite(V2, m);//moisture
Blynk.virtualWrite(V3, l);//water level
If you use a different virtual pin, then change it in the code accordingly. Also, place the E-mail and push notification widgets on the dashboard.

13. You are all set up, now connect the NodeMCU and Arduino with any two COMS and make sure to change the board before you upload the code. To change the code, go to Tools->Boards and select the boards appropriately.

14. Make sure the Blynk app is open while the NodeMCU tries to connect with your hotspot.

15. After uploading, you should see the values displaying on the LCD display as well as on the Blynk app.

16. Finally, you can also change the values at which the alert is shown.
For example->if you want the alert when the temperature rises over 40-degree Celsius then it can be changed in the code above.