ASSIGNMENT NO: 03

TITLE:

To upload data from environmental sensor to android phone App

AIM:

To Design and implement real time monitoring system using android phone (Blynk App.) such as'soil parameter monitoring'.

THEORY:

When you hear the term ‘smart garden’, one of the things that comes to your mind is a system that measures soil moisture and irrigates your plants automatically.

With this type of system, you can water your plants only when needed and avoid over-watering or under-watering.

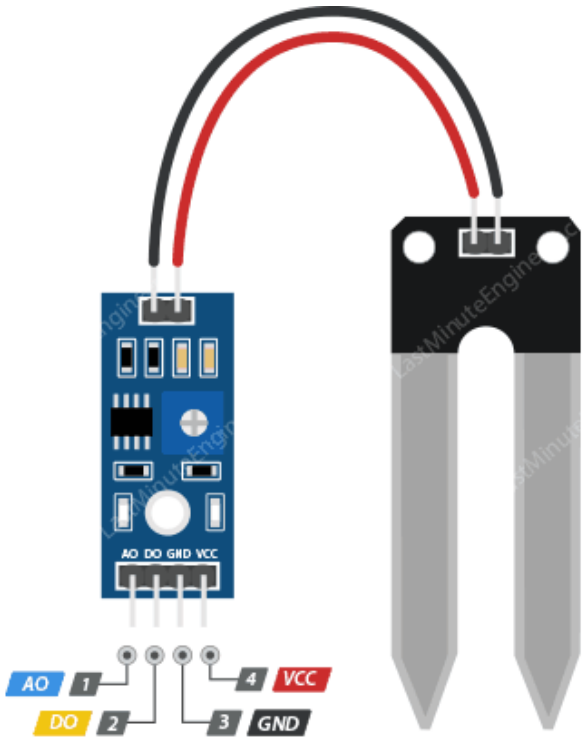
Moisture Sensor:

How Soil Moisture Sensor works?

This resistance is inversely proportional to the soil moisture:

The more water in the soil means better conductivity and will result in a lower resistance. *The less water in the soil means poor conductivity and will result in a higher resistance.*

The sensor produces an output voltage according to the resistance, which by measuring we can determine the moisture level. The fork-shaped probe with two exposed conductors, acts as a variable resistor (just like a potentiometer) whose resistance varies according to the water content in the soil. Simply insert this rugged sensor into the soil to be tested, and the volumetric water content of the soil is reported in percent.



- A0 (Analog Output) pin gives us an analog signal between the supply value to 0V.
- DO (Digital Output) pin gives Digital output of internal comparator circuit.
- VCC pin supplies power for the sensor. It is recommended to power the sensor with between 3.3V – 5V. GND is a ground connection.

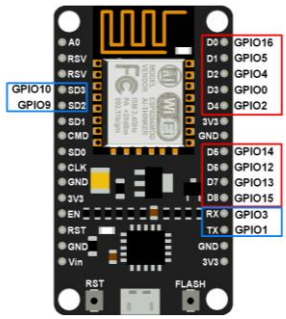
The Module

- The sensor also contains an electronic module that connects the probe to the Arduino.
- The module produces an output voltage according to the resistance of the probe and is made available at an Analog Output (AO) pin.
- The same signal is fed to a LM393 High Precision Comparator to digitize it and is made available at an Digital Output (DO) pin.

- The module has a built-in potentiometer for sensitivity adjustment of the digital output (DO).
- You can set a threshold by using a potentiometer; So that when the moisture level exceeds the threshold value, the module will output LOW otherwise HIGH.
- This setup is very useful when you want to trigger an action when certain threshold is reached. For example, when the moisture level in the soil crosses a threshold, you can activate a relay to start pumping water.

COMPONENTS REQUIRED:

- ESP8266 (NodeMCU)
- Soil Moisture Sensor module
- Breadboard
- Connecting Wires

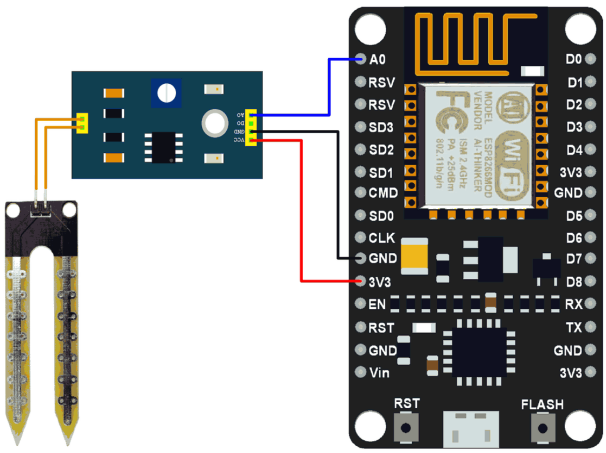


Circuit Diagram

In this experiment, we are going to log and monitor Soil Moisture data over internet using Blynk app IoT server. And we can view the logged data over time on Blynk app. System is made using ESP8266 WiFi module and Soil Moisture Sensor. ESP8266 WiFi chip reads the current Soil Moisture and sends it to Blynk server for live monitoring.

In this experiment, **Soil Moisture Sensor** can be used with esp8266 or Nodemcu and program from Arduino IDE, the Soil Moisture Sensor has used to send data to Blunk app and it was really productive there is no sudden spike, it was gradual and results are quite good. Same Soil Moisture Sensor output can be observed on serial monitor of ardunio IDE.

Connection:



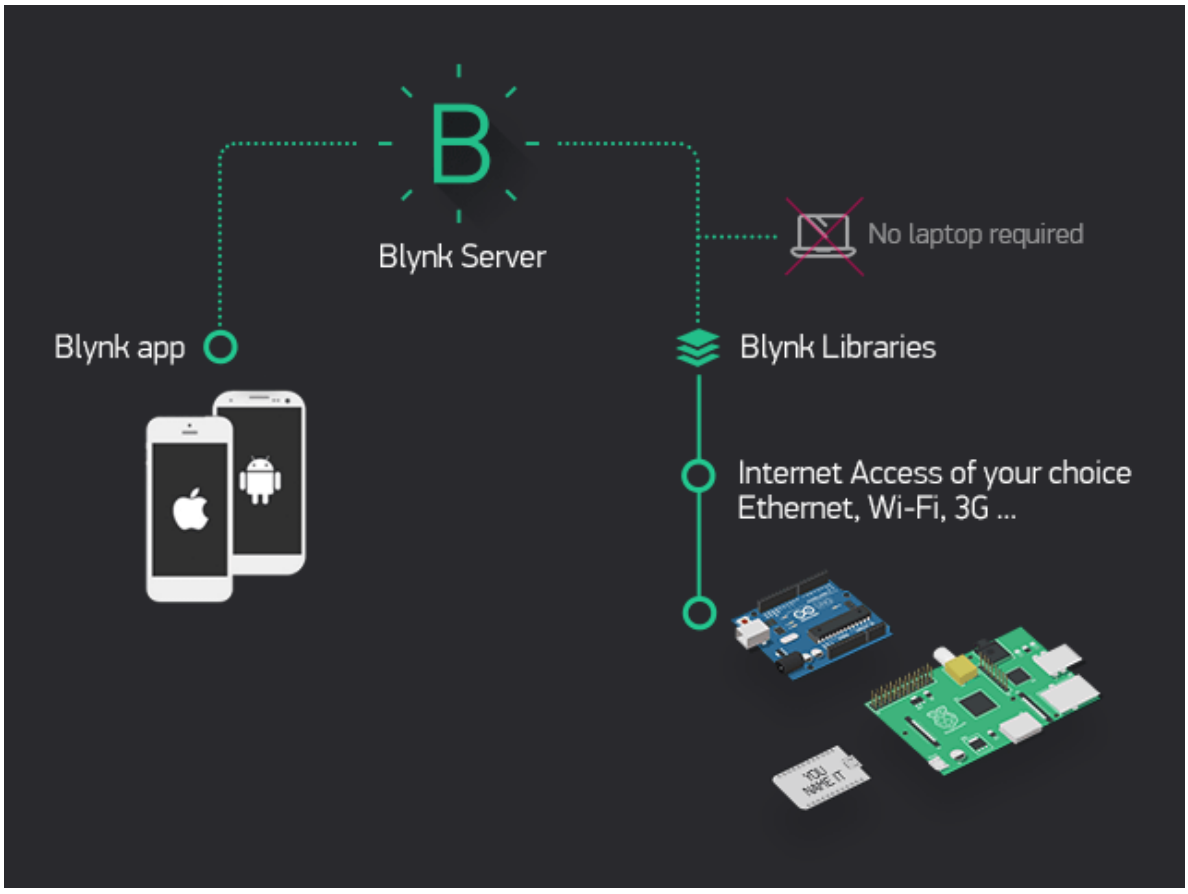
Blynk App:

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, vizualize it and do many other cool things.

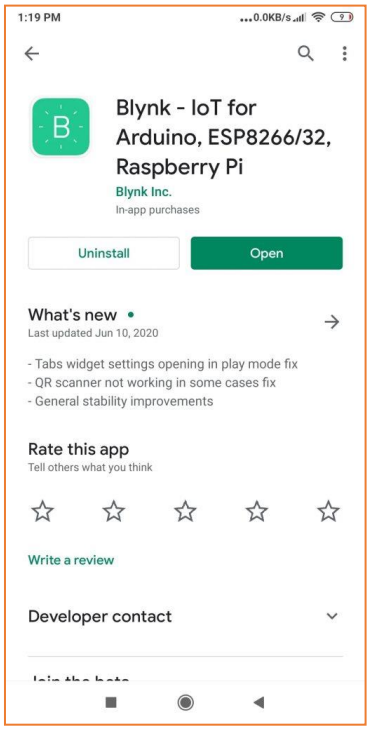
There are **three** major components in the platform:

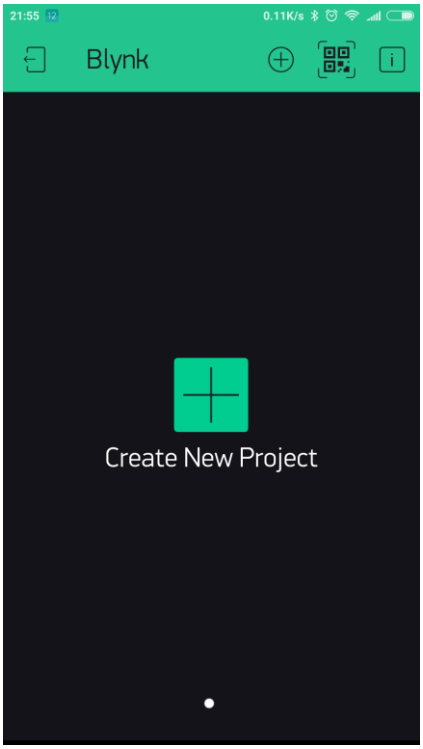
- **Blynk App** - allows to you create amazing interfaces for your projects using various widgets we provide.
- **Blynk Server** - responsible for all the communications between the smartphone and hardware. You can use the Blynk Cloud or run your private Blynk server locally. It's open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.
- **Blynk Libraries** - for all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

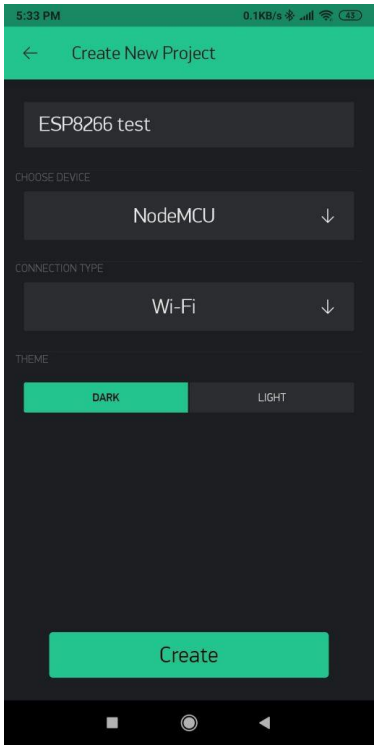
Now imagine: every time you press a Button in the Blynk app, the message travels to the Blynk Cloud, where it magically finds its way to your hardware. It works the same in the opposite direction and everything happens in a blynk of an eye.

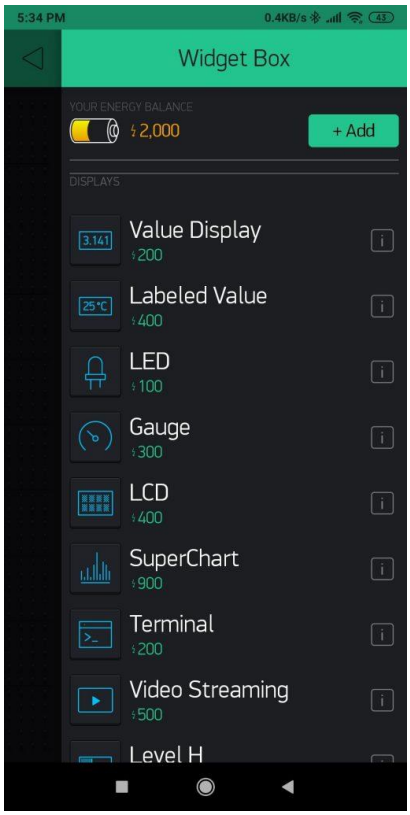


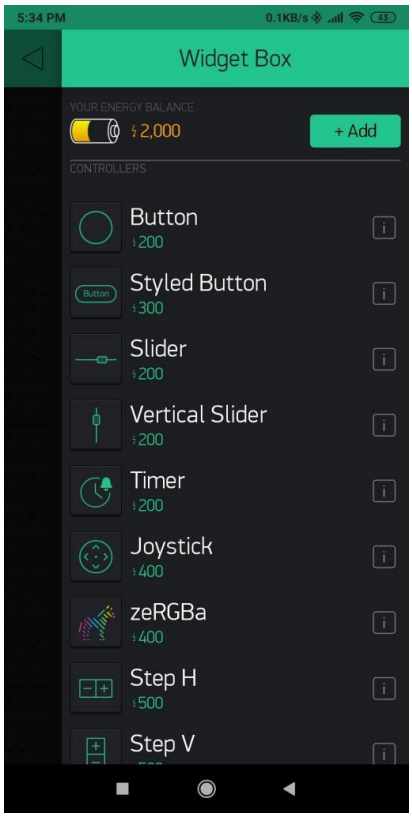
Steps to use Blynk App:





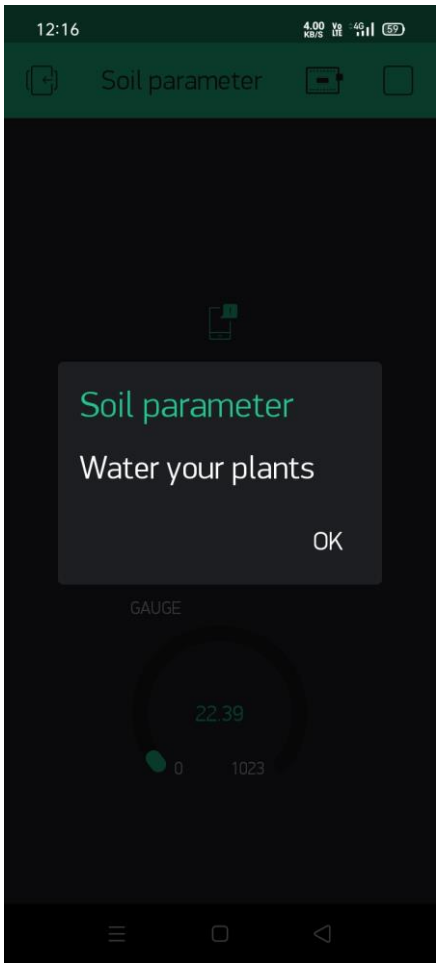






Blynk works over the Internet. This means that the hardware you choose should be able to connect to the internet. Some of the boards, like Arduino Uno will need an Ethernet or Wi-Fi Shield to communicate, others are already Internet-enabled: like the ESP8266, Raspberri Pi with WiFi dongle, Particle Photon or SparkFun Blynk Board. But even if you don't have a shield, you can connect it over USB to your laptop or desktop (it's a bit more complicated for newbies, but we got you covered). What's cool, is that the list of hardware that works with Blynk is huge and will keep on growing.

The Blynk App is a well-designed interface builder. It works on both iOS and Android.



Program:

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "_Xt3mzWJfi-rCdB04cwJDJw4r0-RGwak";
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "OPPO_pcl";
char pass[] = "pclatane123";
float moisture_percentage;
const int sensor_pin = A0;  /* Soil moisture sensor O/P pin */
BlynkTimer timer;
// This function sends Arduino's up time every second to Virtual Pin (5).
// In the app, Widget's reading frequency should be set to PUSH. This
means that you define how often to send data to Blynk App.
void sendSensor()
{
 Blynk.virtualWrite(V5, moisture_percentage);
}
 void setup()
{
  // Debug console
  Serial.begin(9600);
  pinMode(D0, OUTPUT);  /* Relay for water pump, use D0 on Nodemcu */
  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 8442);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8442);
  // Setup a function to be called every second
  timer.setInterval(1000L, sendSensor);
}
```

```
void loop()
{
 Blynk.run();
 timer.run();
 moisture_percentage = ( 100 - ((analogRead(A0)/1023.00) * 100));
 if ( moisture_percentage > 30)
 {
 digitalWrite(D0, HIGH);
 Serial.println("does not need water");
 delay(1000);
 }
 else
 {
 digitalWrite(D0, LOW);
 Serial.println("needs water, send notification");
 Blynk.notify("Water your plants");
 delay(1000);
 }
 Serial.print("Moisture Percentage = ");
 Serial.print(moisture_percentage);
 Serial.print("%\n");
 delay(1000);
}
```

CONCLUSION:

After the study of this assignment we are familiar with the Blynk App and how to send soil moisture on cloud.