University of Bahrain

College of Engineering

MSc in Artificial Intelligence System

Implementation of an IoT-based plant monitoring system

Prepared by: Fatema Abduljabbar Jawad

ID: 20163743

Supervised by: Prof. Mohab Mangoud

Table of Contents

Introduction:

Problem Statement:

Water usage and plant diseases have a very crucial impact on the green revolution [1]. While our globe may never run out of water as a whole, it's important to recognize that pure freshwater isn't always plentiful where and when humans use it. In reality, only six countries have access to half of the world's freshwater. Over a billion people do not have access to adequate safe, clean water [2].

Plant disease has influenced food production and human civilization's evolution for thousands of years. In the early agricultural era, epidemics of plant disease were viewed as a punishment from the gods, and overt plant disease control methods were severely limited. Given poor yields and a lack of considerable food stocks, food shortages might readily emerge once disease outbreaks strike, wreaking havoc on human civilization as the Irish famine caused by potato late blight in the 1840s and the 1943 Bengal famine caused by rice brown spot did [3]. Another example of food insufficiency occurred in India between 1947 and 1960, when food production was insufficient due to an increasing population, and famine was predicted. Only 417 g of food was available per person per day [1].

Project Objectives:

The goal of the project is to protect plants from infections and conserve water by creating an Internet of Things system that consists of several sensors that monitor plant status 24/7 and determine when the plant requires watering.

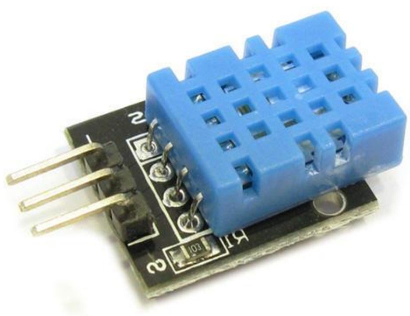
Report outline:

This report is divided into six categories, First is the Introduction where the problem is stated and objective of this project, after that in the system design section hardware and software equipment's are discussed, and then we have the system implementation, how it is done and the code are explained in details, after that the results are shown, then in the conclusion the challenges in this project and future work is stated, and at the end references are listed.

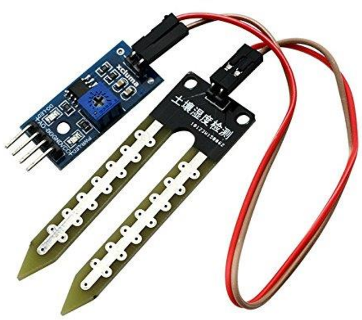
System Design:

Hardware components:

1. DHT: Temperature and humidity sensor.



2. Soil moisture sensor.



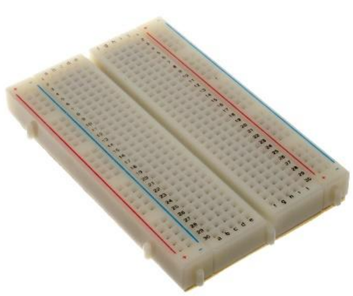
3. Esp8266: Wi-Fi module is used to connect the circuit with the Blynk application.

4. 12v solenoid water valve is used as a switch of the water.

5. 5 volt relay is an automatic switch that is used to automatically control circuit.

6. 12-volt battery for the solenoid.

7. Breadboard for connections.



8. Jumper wires for connections.

9. Power bank to power up the esp8266 Wi-Fi module.

Software components:

1. Arduino to upload code in the esp8266.
2. Blynk for cloud communication.
3. Fritzing to draw the circuit diagram.

System Implementation:

This design have two sensors which are connected to a Wi-Fi module, and that Wi-Fi module have a connection with the Blynk app where I have created an IOT Plant monitoring project and included two gauges for soil moisture and temperature, I have included also a real time Super Chart for the three measurements; Humidity (blue-line), temperature (red-line) and soil moisture (yellow-line), I have added a button as a switch for the solenoid valve, in addition to a timer that will help to open and close the water valve within an interval of time.

Table 1: Pin diagram of the circuit

ESP8266	DHT	
D4	Signal	
GND	(-)	
3V3	(+)	
ESP8266	Soil moisture	
A0	A0	
D3	D0	
GND	GND	
3V3	VCC+	
ESP8266	5V Relay	
D0	IN	
Vin	C+	
GND	C-	
Water valve	5V Relay	12V BATTERY
Volt (+)	COM	
GND (-)		GND (-)
	NO	VOLT (+)

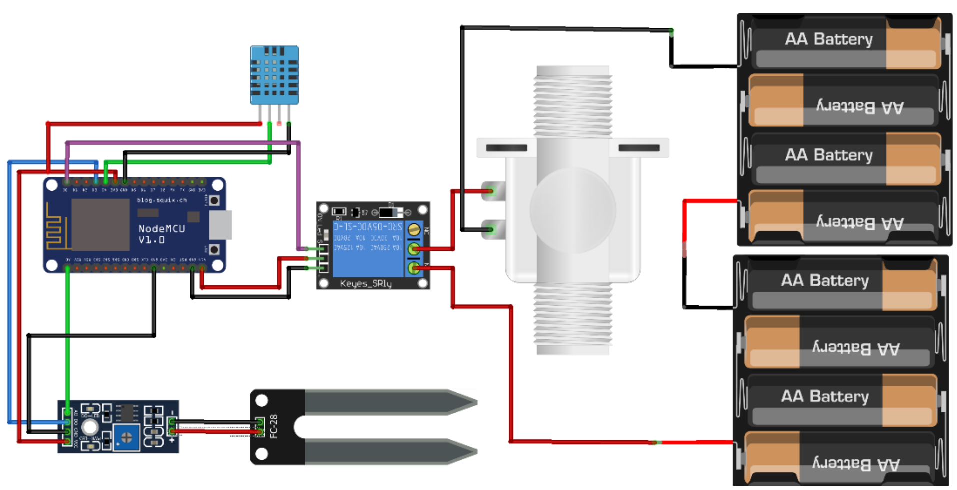


Figure 1: Circuit wiring connection

Steps of Implementation:

1. I have created a project named (IOT Plant Monitoring) in the Blynk application

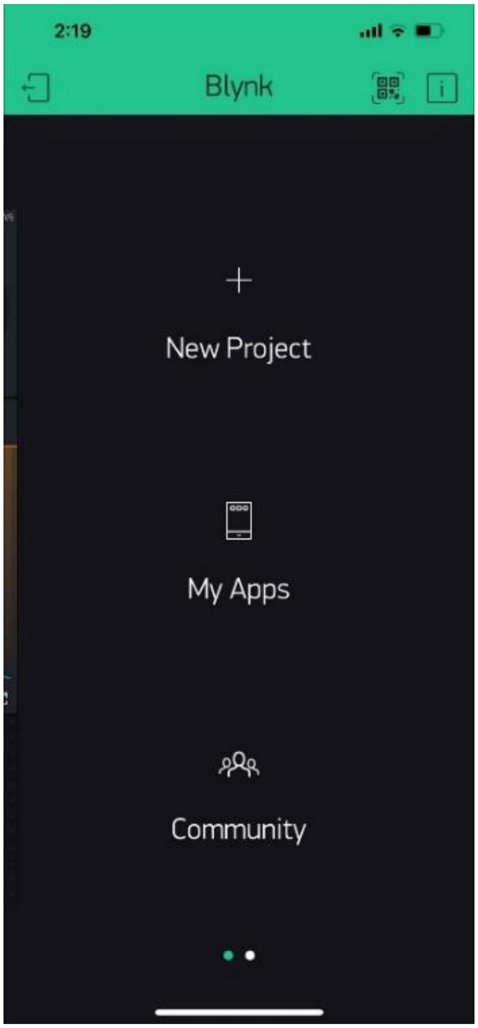


Figure 2: Creat application

2. I have Added two gauges:
   - One for the moisture readings (Virtual 2 from 0 to 1024)
   - Another one for the Temperature readings (Virtual 6 from 0 to 100)

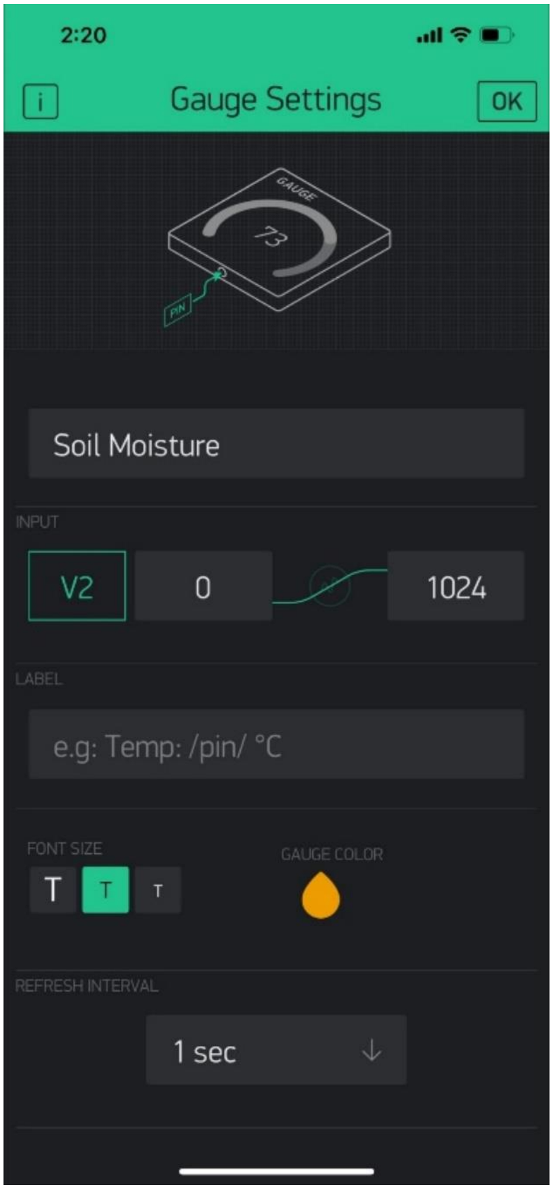


Figure 3: Soil moisture gauge

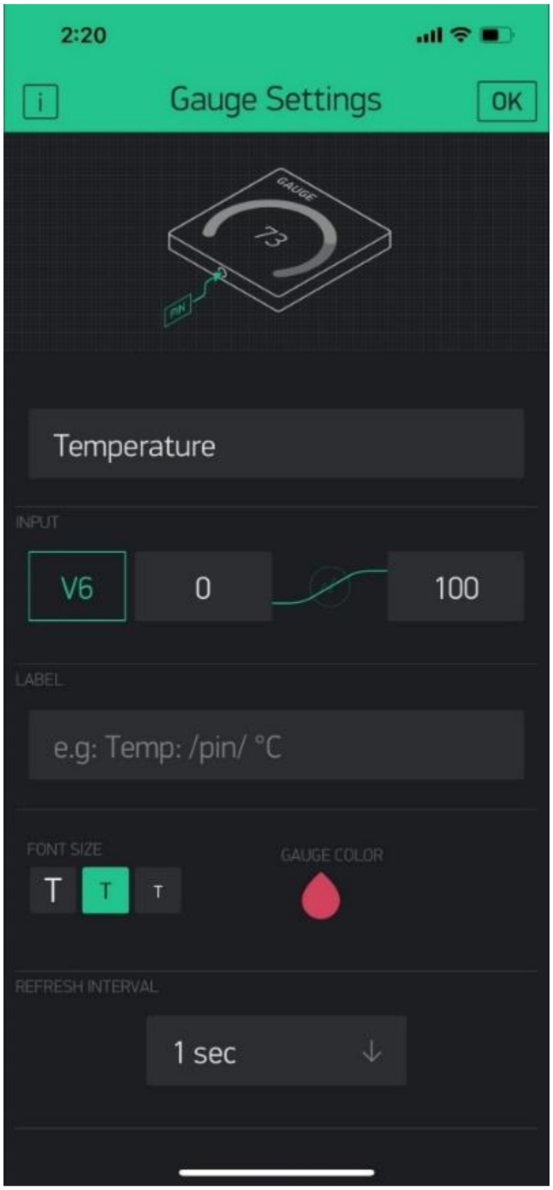


Figure 4: Temperature gauge

3. I have added SuperChart named “Soil moisture, Temp and Humidity for a real time readings

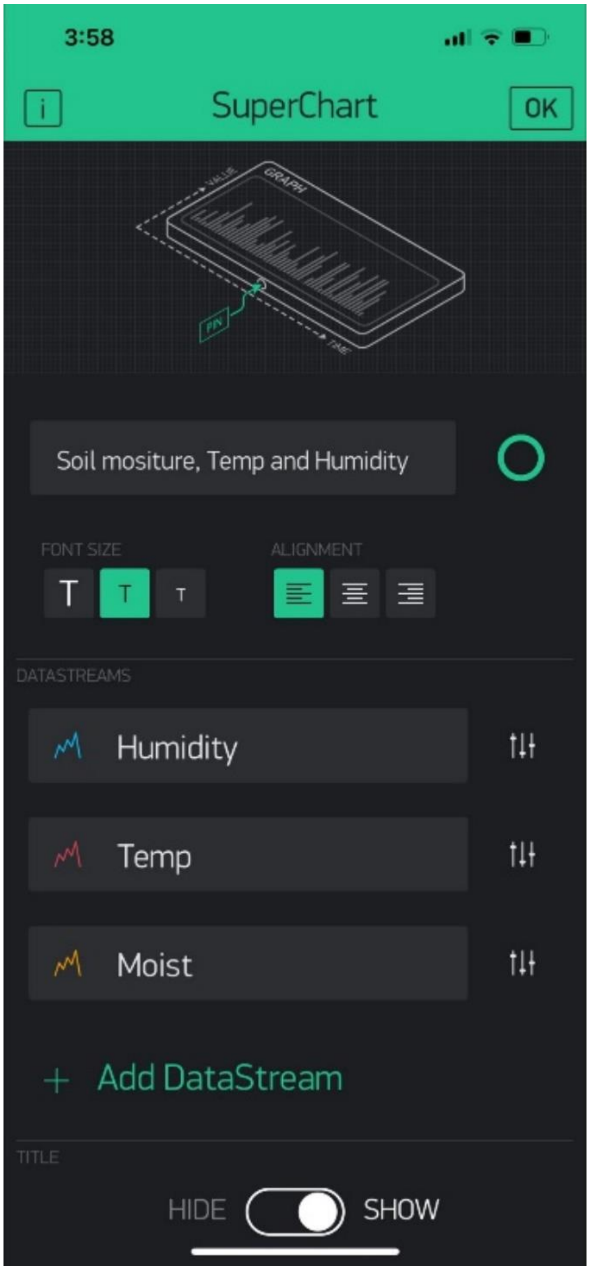


Figure 5: Sensors reading SuperChart

4. I have added a switch button named “Water” (Digital 0) which is connected to the solenoid water valve

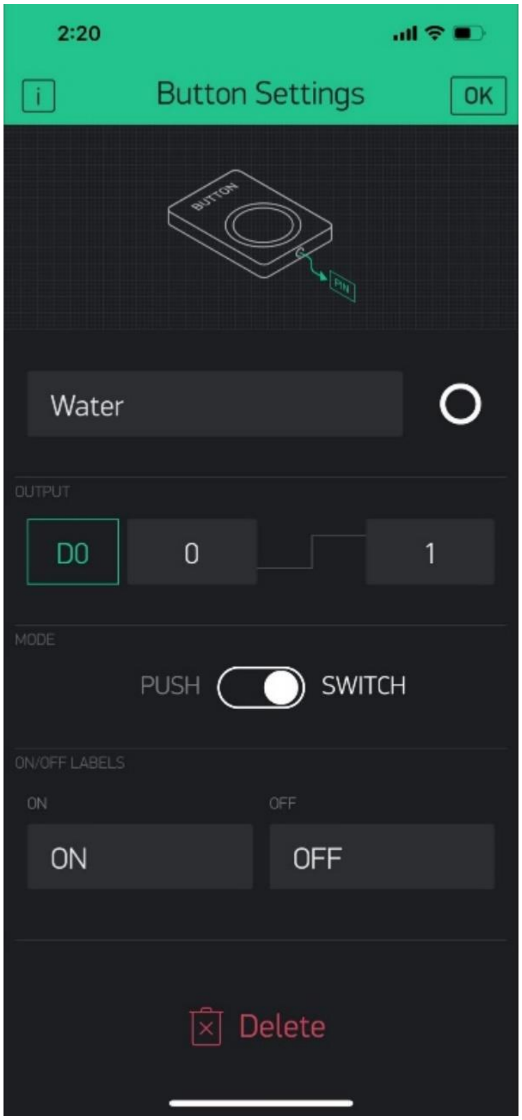


Figure 6: Solenoid water valve button

5. I have added a timer for opening and closing the solenoid water valve

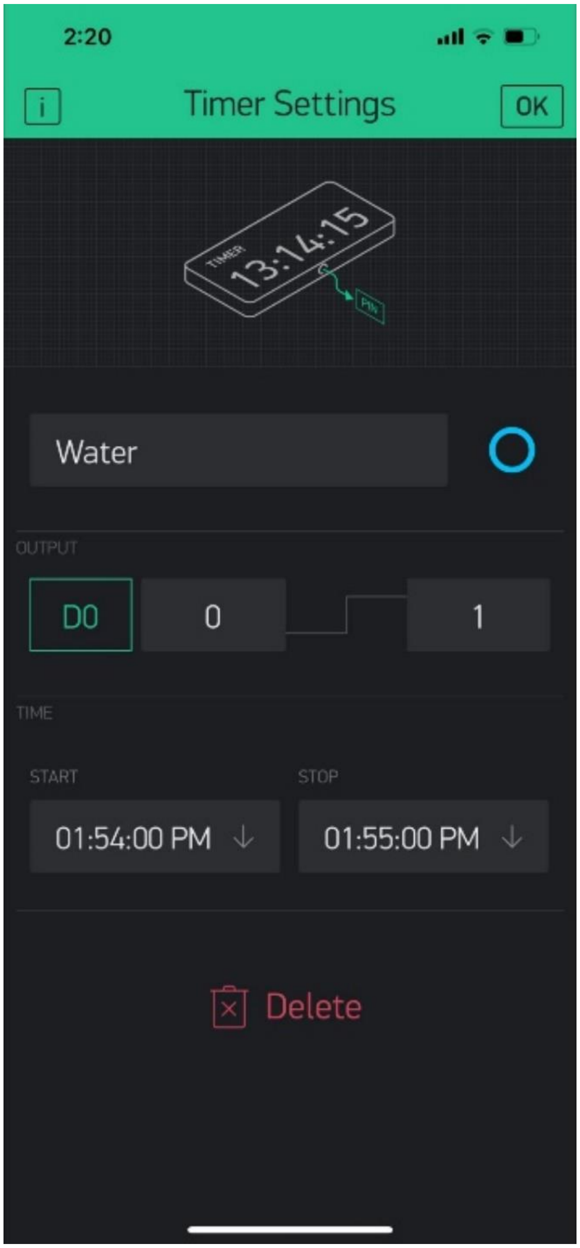


Figure 7: Timer for Solenoid

6. Each project in the Blynk application have an authentication token to be added to the software code and then uploaded to the hardware Bluetooth or Wi-fi module; in our case we used esp8266 wi-fi module, the authentication token was sent by email also.

Figure 8: Authentication token

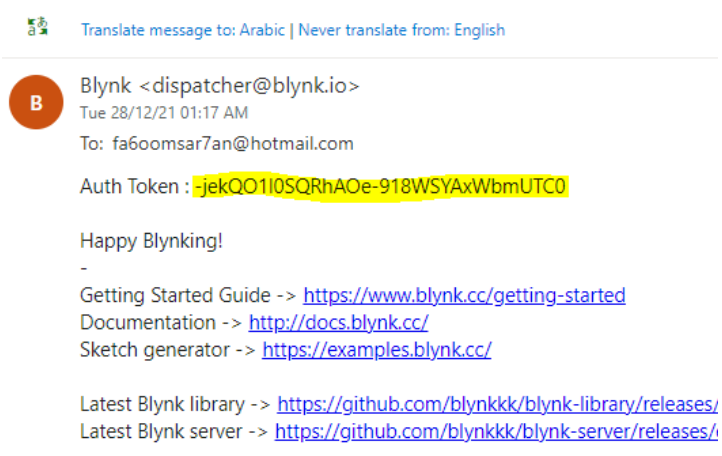


Figure 9: Authentication token - e-mail

7. I have added the authentication token to my code in Arduino and then I complied the code and uploaded it into the esp8266 module

```
// OneWire let me access one wire devices made by Dallas, such as temperature.
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS D2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

char authentication[] ="-jekQO1I0SQRhAOe-918WSYAxWbmUTC0";        //Authentication code sent by Blynk
char WiFiSSID[] = "[redacted]";                              //WiFi SSID
char PASSWORD[] = "[redacted]";                               //WiFi Password

#define MoistureSensor D3
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dHumTemp(DHTPIN, DHTTYPE);
SimpleTimer timer;
```

Figure 10: Authentication in the code

8. Then when powering the circuit, the project start reading the measurements

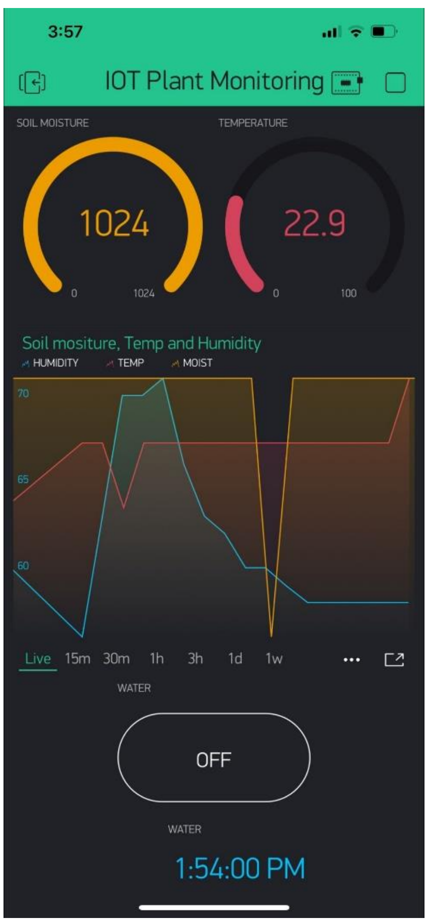


Figure 11: Interface of the project

9. From the SuperChart we can have the measurements sent by email as an excel sheet, where each data is sent separated in a different file; v5 for humidity, v6 for temperature, and v2 soil moisture.

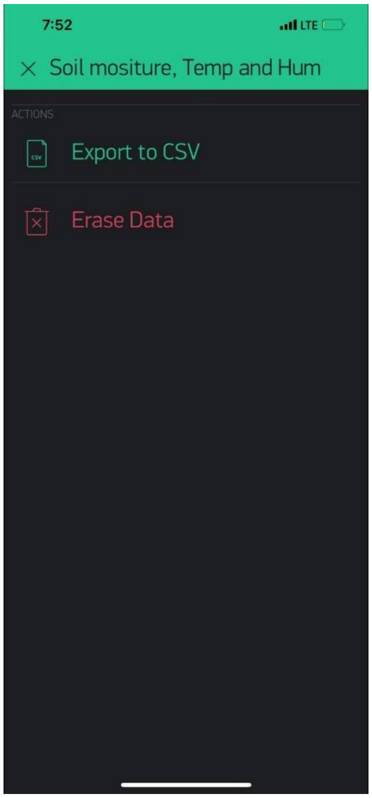


Figure 12: Export data as CSV

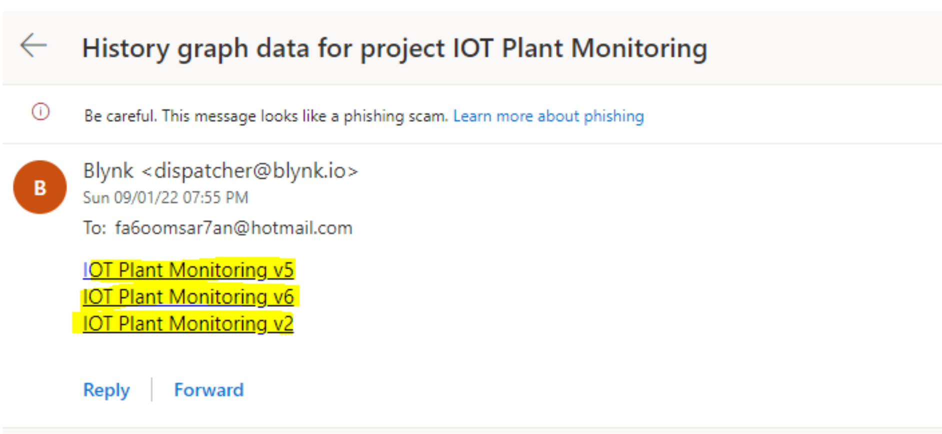


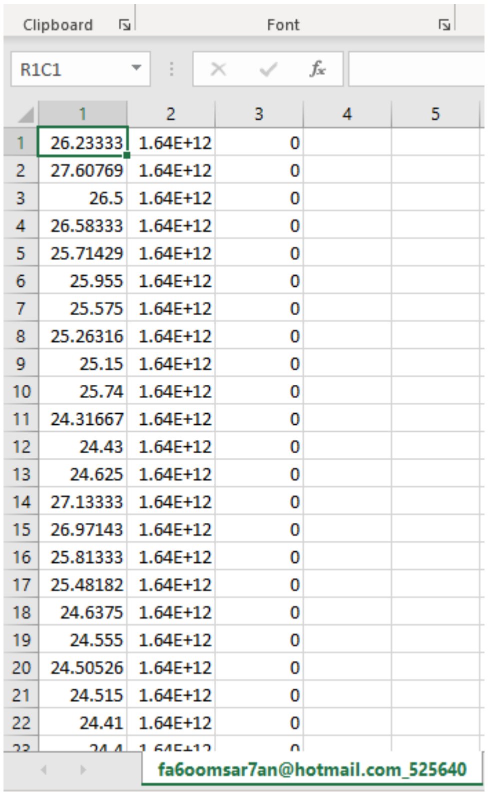
Figure 13: CSV data recieved by e-mail

	1	2	3	4	5
1	770.8	1.64E+12	0		
2	832.95	1.64E+12	0		
3	984.5789	1.64E+12	0		
4	1024	1.64E+12	0		
5	1023.9	1.64E+12	0		
6	1024	1.64E+12	0		
7	1018.438	1.64E+12	0		
8	958.15	1.64E+12	0		
9	973	1.64E+12	0		
10	978.85	1.64E+12	0		
11	987.8	1.64E+12	0		
12	994.5	1.64E+12	0		
13	989.8235	1.64E+12	0		
14	1024	1.64E+12	0		
15	1001.55	1.64E+12	0		
16	976.6	1.64E+12	0		
17	995.4615	1.64E+12	0		
18	985.8	1.64E+12	0		
19	988	1.64E+12	0		
20	1020.438	1.64E+12	0		
21	1019.455	1.64E+12	0		
22	1024	1.64E+12	0		

Figure 14: Moisture data

	1	2	3	4	5
1	77.83333	1.64E+12	0		
2	79.92308	1.64E+12	0		
3	66.5	1.64E+12	0		
4	72.16667	1.64E+12	0		
5	72.57143	1.64E+12	0		
6	61.65	1.64E+12	0		
7	61.4	1.64E+12	0		
8	62.7	1.64E+12	0		
9	63.14286	1.64E+12	0		
10	92.9	1.64E+12	0		
11	67.5	1.64E+12	0		
12	77.2	1.64E+12	0		
13	92.25	1.64E+12	0		
14	82.25	1.64E+12	0		
15	77.57143	1.64E+12	0		
16	59.46667	1.64E+12	0		
17	63.54545	1.64E+12	0		
18	61	1.64E+12	0		
19	61.55	1.64E+12	0		
20	61.89474	1.64E+12	0		
21	62.15	1.64E+12	0		
22	62.55	1.64E+12	0		

Figure 15: Humidity data



	1	2	3	4	5
1	26.23333	1.64E+12	0		
2	27.60769	1.64E+12	0		
3	26.5	1.64E+12	0		
4	26.58333	1.64E+12	0		
5	25.71429	1.64E+12	0		
6	25.955	1.64E+12	0		
7	25.575	1.64E+12	0		
8	25.26316	1.64E+12	0		
9	25.15	1.64E+12	0		
10	25.74	1.64E+12	0		
11	24.31667	1.64E+12	0		
12	24.43	1.64E+12	0		
13	24.625	1.64E+12	0		
14	27.13333	1.64E+12	0		
15	26.97143	1.64E+12	0		
16	25.81333	1.64E+12	0		
17	25.48182	1.64E+12	0		
18	24.6375	1.64E+12	0		
19	24.555	1.64E+12	0		
20	24.50526	1.64E+12	0		
21	24.515	1.64E+12	0		
22	24.41	1.64E+12	0		
23	24.4	1.64E+12	0		

Figure 16: Temperature data

Code Explanation:

First, I have included libraries needed such as ESP8266WIFI, BlynkSimpleEsp8266, OneWire and DallasTemperature libraries, I have added also the authentication token, Wi-Fi SSID and Password for connection and better to notice that both phone and esp8266 should be connected to the same network

Plant_Monitoring §

```
#define BLYNK_PRINT Serial
#include <SPI.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>
#define BLYNK_PRINT Serial
// OneWire let me access one wire devices made by Dallas, such as temperature.
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS D2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

char authentication[] ="-jekQO1I0SQRhAOe-918WSYAxWbmUTC0";        //Authentication code sent by Blynk
char WiFiSSID[] = "Jabbar";                           //WiFi SSID
char PASSWORD[] = "asfzmmai";                           //WiFi Password
```

Figure 17: Code-1

And then I have defined Pin D3 for moisture sensor to read the data and I have made a class to read temperature and humidity and send the data to Blynk application through the cloud.

Plant_Monitoring §

```
#define MoistureSensor D3
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dHumTemp(DHTPIN, DHTTYPE);
SimpleTimer timer;

// a class to send humidity and temperature to the blynk application
void sendDHTsensor()
{
  float humidity = dHumTemp.readHumidity(); // read humidity from the sensor and save it to the variable "humidity"
  float temperature = dHumTemp.readTemperature(); // read temperature from the sensor and save it to the variable "temperature"
  // use isnan function (is not a number), if the reading fails a message will occur
  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Failed!!!");
    return;
  }

  Blynk.virtualWrite(V5, humidity);  //send Humidity measurment to Virtual (V5) in blynk application
  Blynk.virtualWrite(V6, temperature);  //send Temperature measurement Virtual (V6) in blynk application
}
```

Figure 18: Code-2

Start connection to Blynk and network and start sending data to Blynk through the network

```
void setup()
{
  Serial.begin(9600);
  Blynk.begin(authentication, WiFiSSID, PASSWORD); // start connecting to the blyn
  pinMode(MoistureSensor, INPUT); // moisture sensor is set as an input
  dHumTemp.begin(); // start reading humidity and temperature

  timer.setInterval(1000L, sendDHTsensor); // calls the function "sendDHTsensor"
  Serial.begin(115200);
  Blynk.begin(authentication, WiFiSSID, PASSWORD);
   sensors.begin();
}
```

Figure 19: Code-3

Write temperature to v1 as Celsius and write moisture measurement to v2 in Blynk application

```
int moistureSense=0; // define a variable for the moisture sensor
void sendTemps()
{
moistureSense=analogRead(A0); // analog reading for soil moisture
sensors.requestTemperatures();
float temp = sensors.getTempCByIndex(0); // returns temperature in celicus scale
Blynk.virtualWrite(V1, temp); // write the temperature in celicus to Virtual 1 (V1) in the blynk application
Blynk.virtualWrite(V2,moistureSense); // write the moisture measurment to Virtual 2 (V2) in the blynk application
delay(1000);
}


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

Figure 20: Code-4

Results:

The system was working successfully without any faults. Temperature, Humidity and Moisture was measured and sent to the application through the network, solenoid water valve worked correctly through the button and through the Timer.

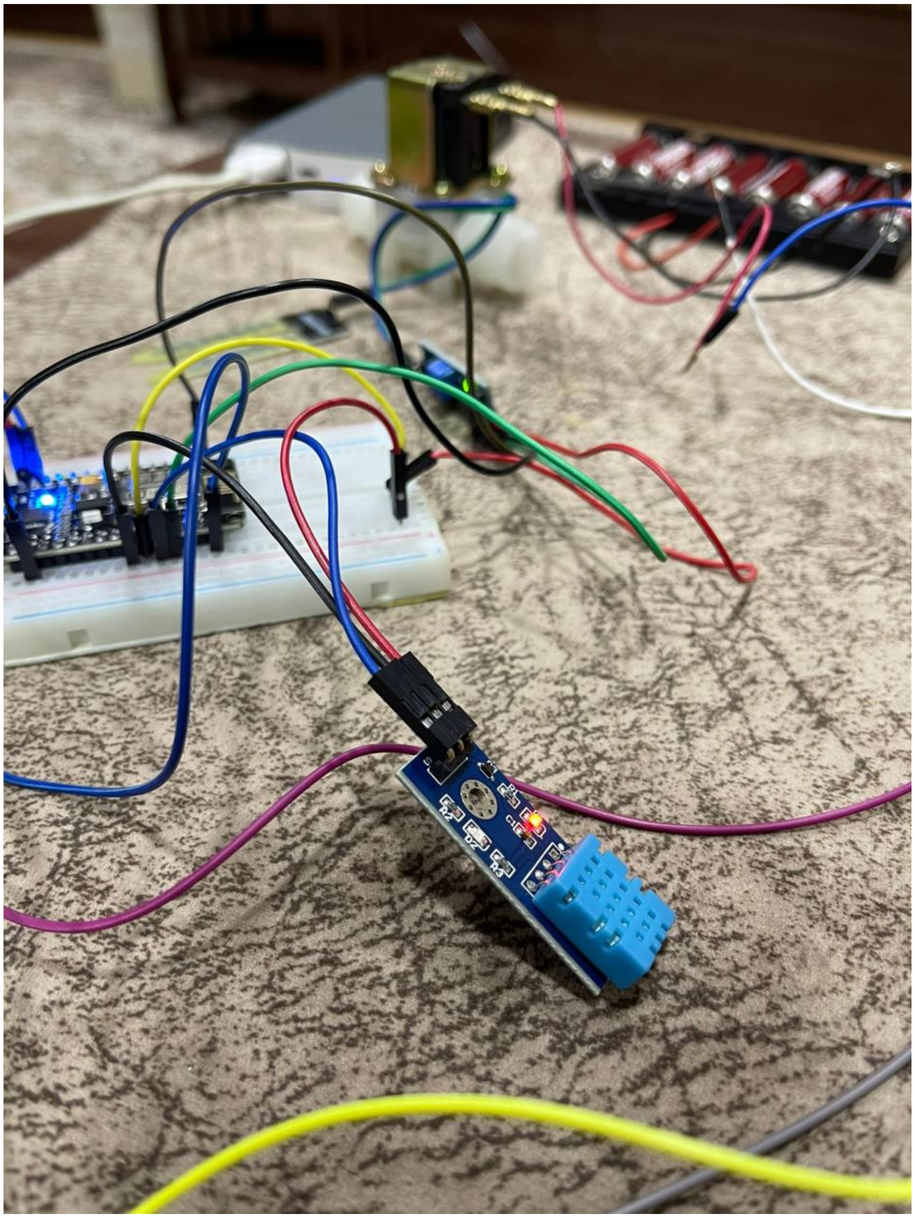


Figure 21: DHT >Temperature and Humidity

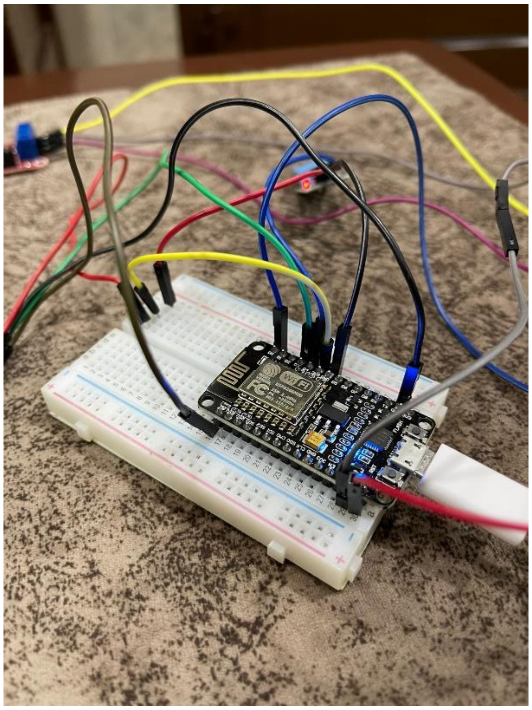


Figure 22: esp8266 Wi-Fi module

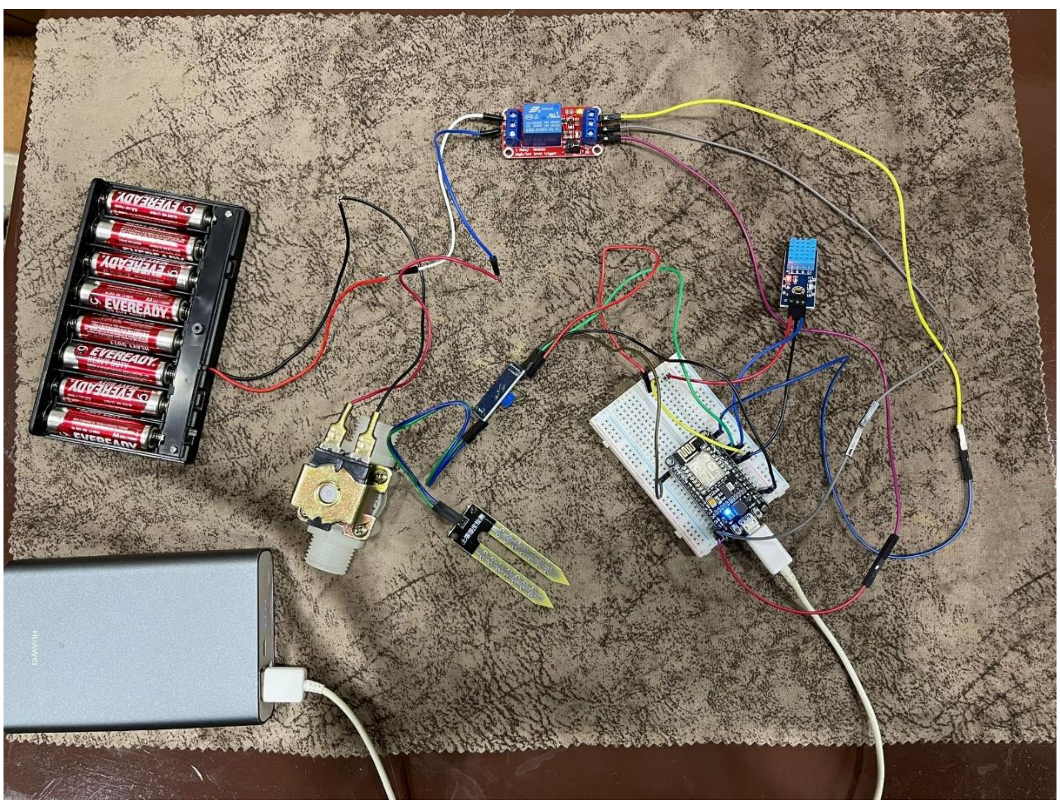


Figure 23: Circuit connection

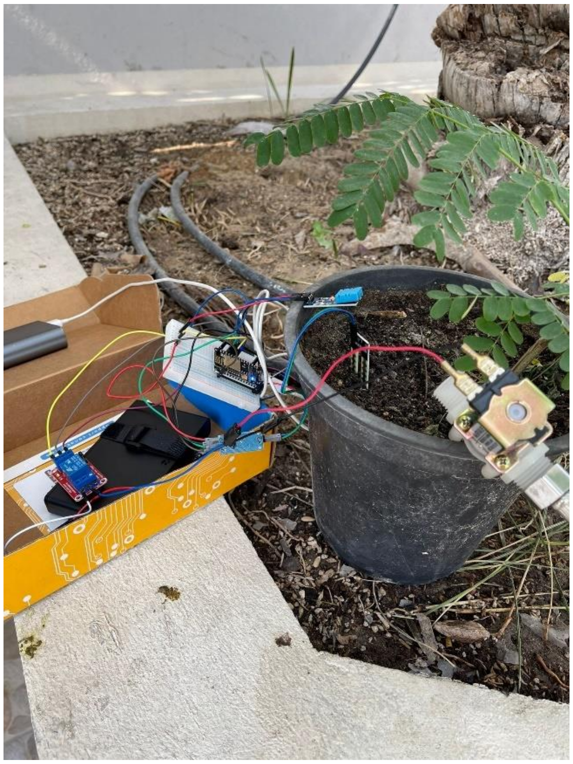


Figure 24: Implementation-1

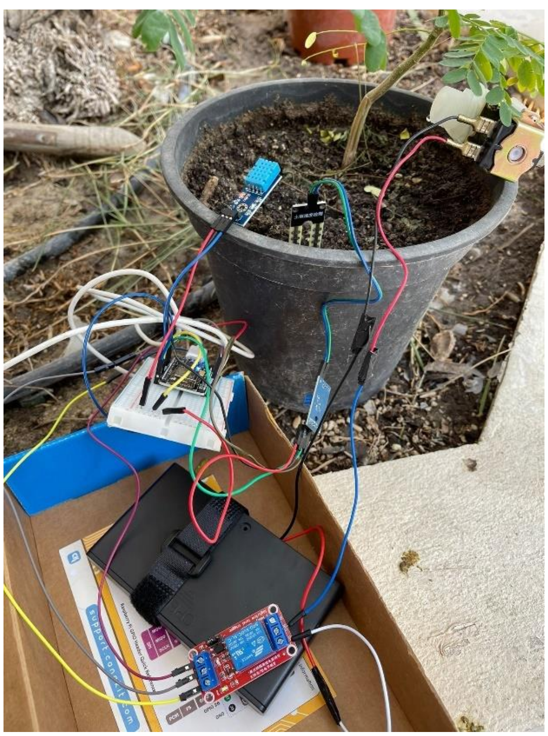


Figure 25: Implementation-2

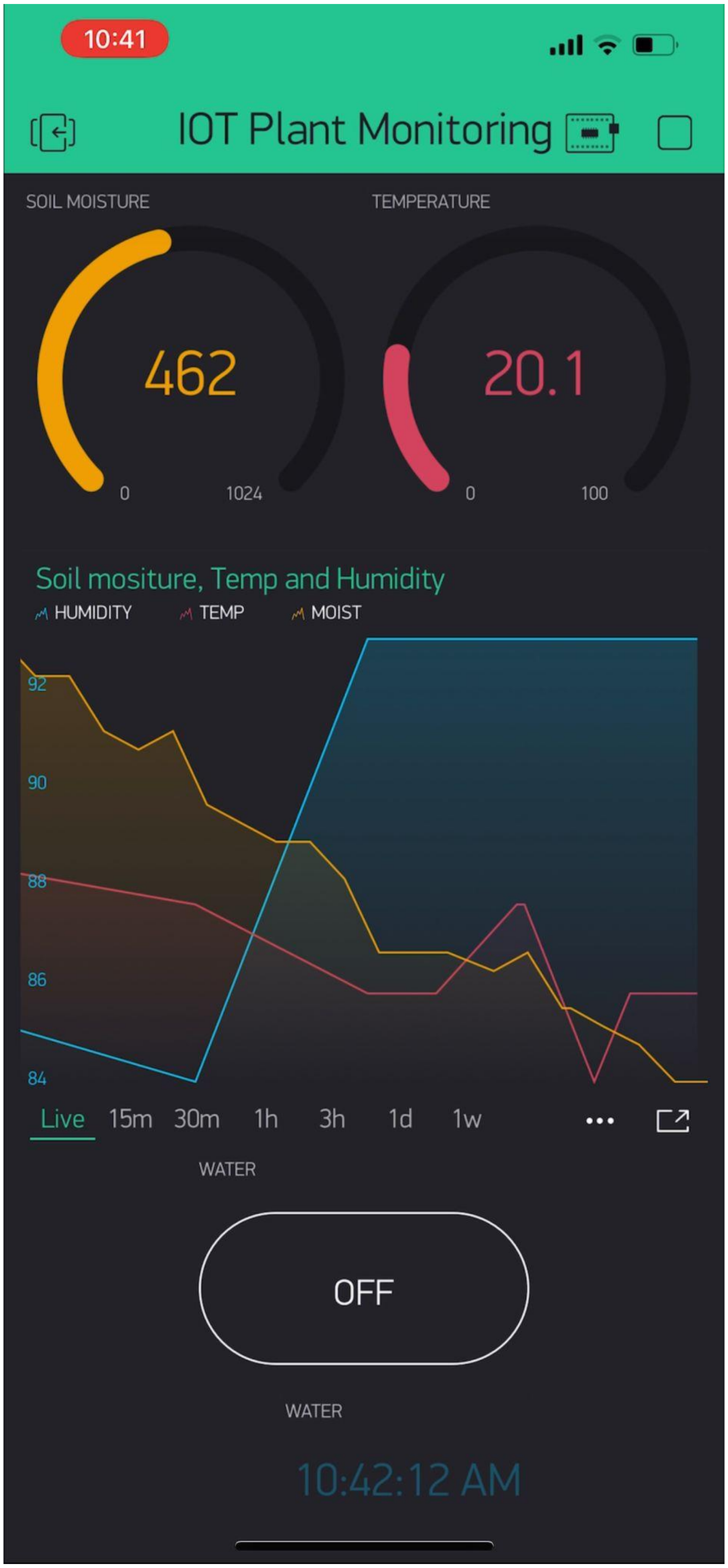


Figure 26: Implementation-3

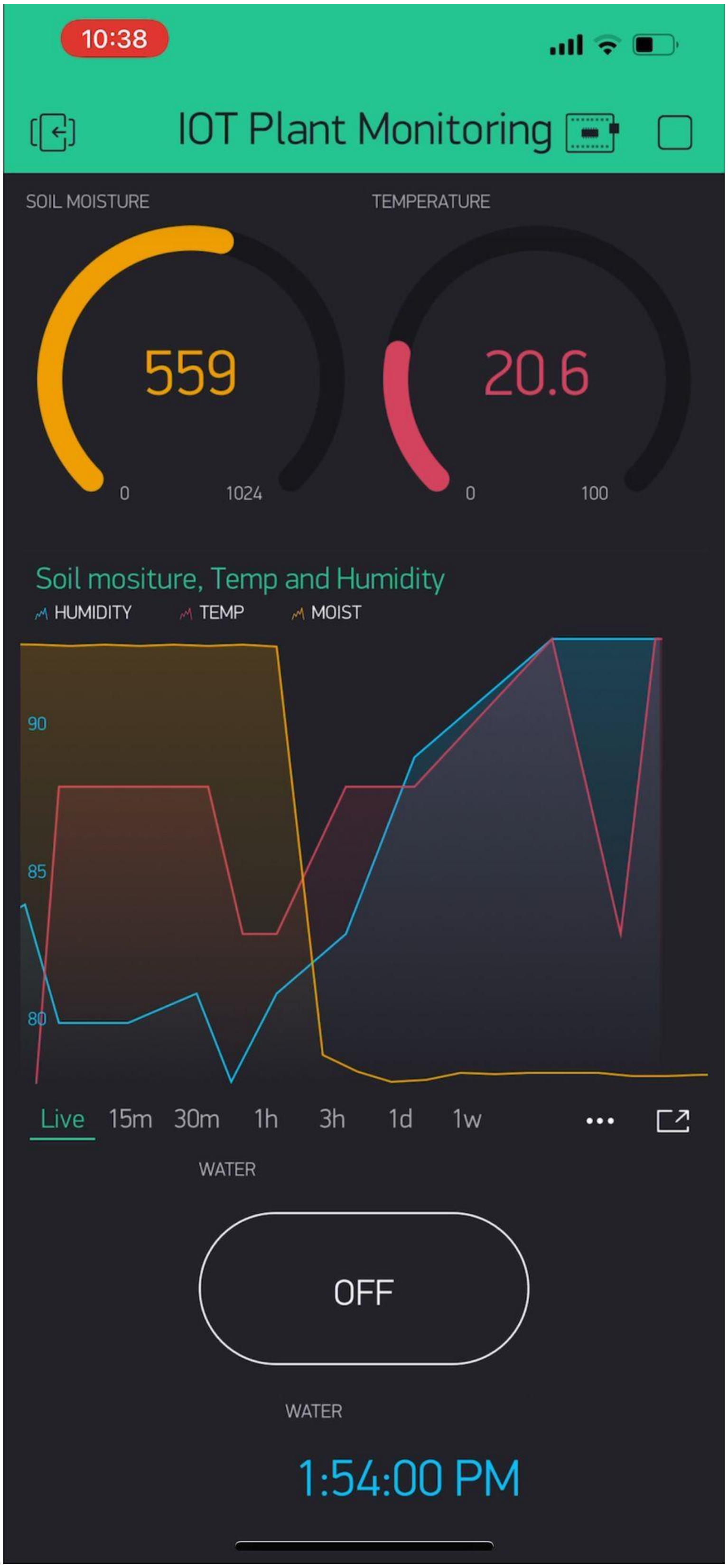


Figure 27: Implementation-4

Conclusion:

Challenges:

One of the issues I experienced during the project was uploading the code from Arduino to the esp8266 because the port was unrecognized by the computer. To address this problem, I discovered that I needed to install a CP2102 driver, in addition to finding a suitable pipe for the solenoid water valve.

Future work:

Many further improvements are possible in the future, including the following:

1. If the plant requires water, it can be watered automatically.
2. If there are any issues with the status of the plants, a notification will be sent to your phone, allowing you to respond quickly.
3. Additional sensors can be added to monitor the plant's health status.
4. Within a geographical area, the health status of all available plants can be gathered to see if the area is infected with a plant disease or if more plants are needed.

Thus the "PLANT MONITORING SYSTEM" has been designed and tested successfully. It has been developed by integrated features of all the hardware components used. Presence of every module has been reasoned out and placed carefully, thus contributing to the best working of the unit. This system has been designed to be connected and send data through the cloud. This system target is to measure the plants moisture, temperature and humidity of the weather and to water the plants from distance by your phone when you are far from your home, or to set a timer for watering the plant within an interval of time.

References:

[1] D. John and G. Babu, "Lessons From the Aftermaths of Green Revolution on Food System and Health", *Frontiers in Sustainable Food Systems*, vol. 5, 2021. Available: 10.3389/fsufs.2021.644559.

[2] 2022. [Online]. Available: https://www.amnh.org/explore/ology/earth/ask-a-scientist-about-our-environment/will-earth-run-out-of-water.

[3] D. HE, J. ZHAN and L. XIE, "Problems, challenges and future of plant disease management: from an ecological point of view", *Journal of Integrative Agriculture*, vol. 15, no. 4, pp. 705-715, 2016. Available: 10.1016/s2095-3119(15)61300-4.

[4] *Instructables.com*, 2022. [Online]. Available: https://www.instructables.com/Blynk-With-ESP8266/.

[5] *Youtube.com*, 2022. [Online]. Available: https://www.youtube.com/watch?v=1e0zIfCzxBo.

[6] "NodeMCU Relay Controlled Solenoid Valve", *Instructables*, 2022. [Online]. Available: https://www.instructables.com/NodeMCU-Relay-Controlled-Solenoid-Valve/.

[7] *Instructables.com*, 2022. [Online]. Available: https://www.instructables.com/How-to-Connect-Soil-Moisture-Sensor-and-ESP8266-to/.

[8] *Youtube.com*, 2022. [Online]. Available: https://www.youtube.com/watch?v=WLFUwyyPrKo.