Integrated Real-World Project 4: Final Project Report

Name: Siti Jasmeen Binte Mohamad Ali

Class: PS21

Student ID: S10204916F

1. Abstract

With the increase of urban farming, people have found that simple urban farming techniques are not efficient enough on their own. For example, hydroponic systems often provide small yields and need regular human maintenance. This has provided engineers with ample opportunities to create potential solutions: the creation of systems that would allow users to make their urban farming systems more time and energy-efficient. This paper is about using a soil moisture monitoring system that would automatically water the plants when necessary. This system removes the need for a human to constantly be checking on the plants, thus saving essential time.

2. Table of Contents

3. Table of Figures

4. Introduction

With the increase of urbanization, urban farming is now very common. They are commonly placed on top of tall buildings and are often quite small, producing low yields. The efficiency of such urban farming systems is low on their own, especially considering the number of man-hours needed to conduct maintenance and care for the plant. Integrating subsystems to help conduct maintenance would reduce the hours needed to physically check the plant.

For example, having a soil moisture reading system would allow the user to ensure that their plant is growing with the optimal amount of soil moisture at all times. The system can be calibrated such that it would automatically water the plant when it detects low levels of soil moisture. In this paper, I will be covering such a system.

5. User Needs and Requirements

Plants need water, oxygen and light to grow. Oxygen and light are easily estimated from the surface for the plant’s growth. However, soil moisture is the hardest to estimate as it may seem dry at the surface, but there is still water deeper into the soil. Plants that are overwatered suffer from a lack of oxygen, which causes the roots to die. [1] Manually estimating soil moisture requires substantial man-hours, which may further decrease the efficiency of the urban farm. With this, the problem statement formulated would be “ How can we create a system that allows the user to check the soil moisture of their plant at any time of the day, as well as have automatic detection of soil moisture such that it minimizes the incidence of human error to prevent over-and under-watering? “

6. Proposed Solution

With this problem statement, a system with a sensor for measuring soil moisture that is connected wirelessly to the Internet is proposed. This system would have a sensor that detects moisture and sends this data to an Internet-of-Things (IoT) platform that would visualize the data. The user can log onto the IoT platform and obtain real-time readings/moisture levels. The system would also be calibrated such that when low soil moisture is detected, it would activate a relay that automatically waters the plant to ensure that the plant is optimally hydrated. The system would automatically stop watering the plant once it detects that the soil is watered.

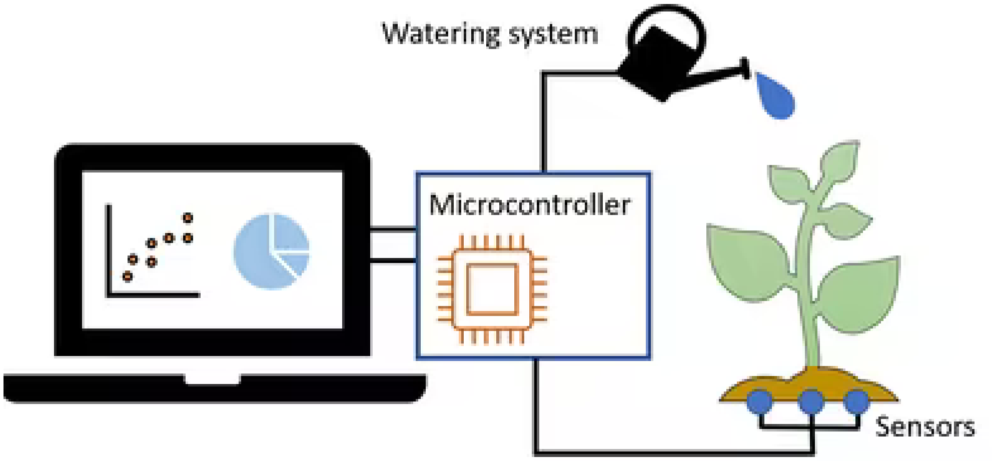


Figure 1: Simplified sample diagram of soil moisture monitoring system
Adapted from [3]

7. System Architecture

The system design has the following components:

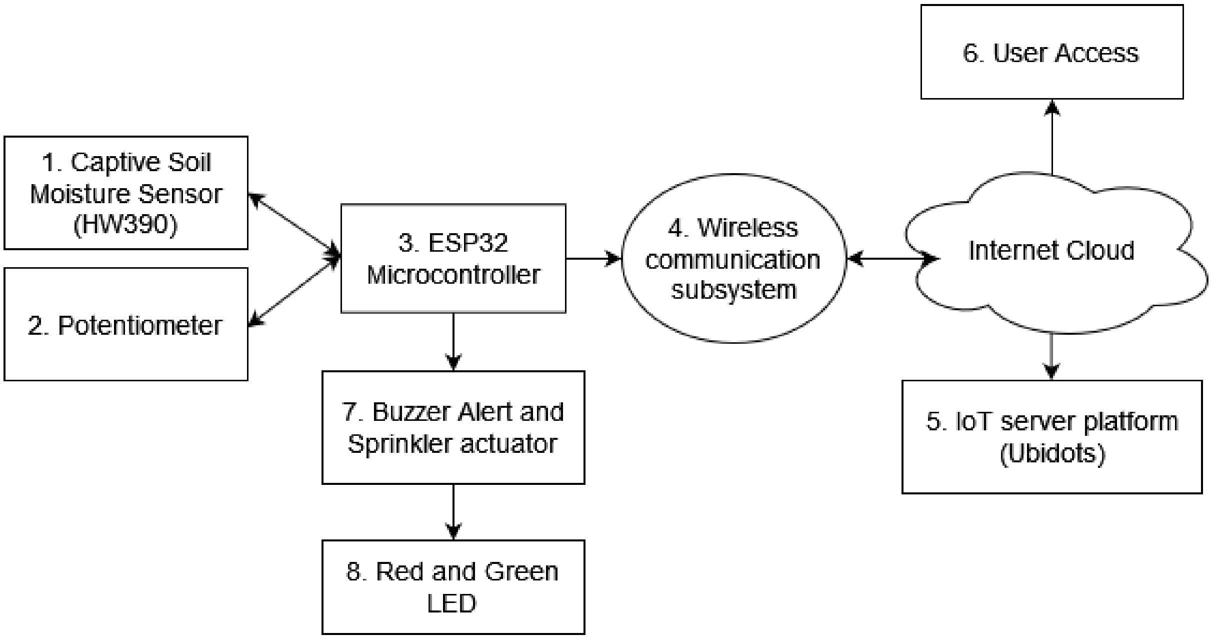

6. User Access
1. Captive Soil Moisture Sensor (HW390)
3. ESP32 Microcontroller
4. Wireless communication subsystem
Internet Cloud
2. Potentiometer
7. Buzzer Alert and Sprinkler actuator
5. IoT server platform (Ubidots)
8. Red and Green LED

Figure 2: System Architecture Diagram for IoT Soil Moisture Monitoring System

Captive Soil Moisture Sensor (HW390) is an embedded electronic module that can detect the changes in capacitance is caused by the ions that are dissolved in the soil [2]. This moisture sensor would be placed into the soil. The user must ensure that it does not come into direct contact with water, as this may affect the data collected. The raw data would be sent to an ESP32, which further processes the data to soil moisture percentage. As this system makes use of sending data to an IoT platform, the system should be set up in an area with internet connection available.

The soil moisture sensor first has to be calibrated to the surroundings of the system. The following lines of code are used to calibrate the sensor.

```
int Sensor = 27; //set the pin where you connect the sensor
void setup() {
Serial.begin(115200); // setting baud rate
}

void loop() {
int val;
val = analogRead(sensor);
Serial.println(val);
delay(100);
}
```

While running this code, the sensor is exposed to open air followed by submerging it in water. The values printed in the serial monitor of the ESP32 would be the minimum and maximum values of the sensor.

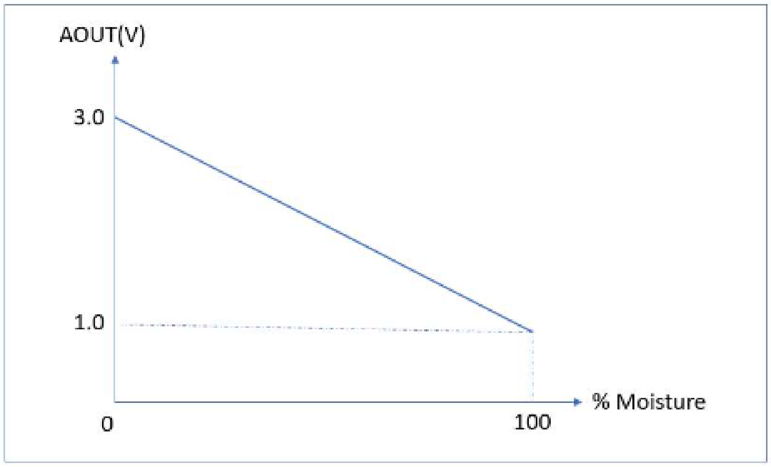
The analog reading of the soil moisture, when exposed to air, would be 3024. When exposed to pure water, the reading would be 1650. These analog values can be calculated to voltage with the following formula:

$$\text{Analog Voltage} =(x+150)*0.0008$$

Where x is the raw value read from the sensor

The voltage outputted when exposed in the air would be 2.54V, and 1.44V when exposed to water.

A formula to calculate the soil moisture percentage can be created.



Graph of Analog Output vs Moisture

Figure 3: Graph of Analog Output vs Moisture of HW390
Adapted from HW390 datasheet

With reference to the figure above, and using a straight-line equation $y = mx + c$, let y represent the output voltage, x represent the moisture percentage.
Substituting m and c,

$$y = -\frac{3-1}{100}x + 3$$

$$y = -\frac{1}{50}x + 3$$

$$x = -50y - 150$$

This equation is written in the code for the ESP32 to calculate soil moisture percentage. This value would be then passed to the IoT server platform, Ubidots. Ubidots allows the user to see visually the data that is being sent to the ESP32 microcontroller. This will allow the user to constantly keep track of their plant's soil moisture at any time (see Appendix 9.3).

The ESP32 is also calibrated such that the user can manually set the threshold soil moisture the user wants to set, depending on the plant that is being monitored, with the use of a potentiometer. The potentiometer is connected to the ESP32 board which will send an analog voltage, which the board would calculate into a moisture percentage and store it in a threshold value. This set threshold value is sent to Ubidots for ease of communication of data.

Additionally, if the ESP32 detects that the moisture is less than the set threshold moisture, it would activate the relay, as well as a buzzer. The buzzer would ring indicating that the soil is not moist enough, and the activated relay would turn on a micropump that will pump water into the soil. This activation removes the need for physical soil inspection by owners. There is also a green LED which will be on when the relay is OFF, indicating that the soil is at a good moisture percentage, and as well as a red LED that will be on when the relay is ON, indicating the soil is not at a good moisture percentage for optimal plant growth. This is so that if the relay malfunctions or if WiFi is unavailable, the user will be notified through the LEDs.

8. Conclusion

This report is about a soil moisture monitoring system that is used to make urban farms more efficient. This moisture monitoring system uses a HW390 capacitive soil moisture sensor, and the respective soil percentage threshold can be set manually by adjusting the potentiometer. This soil moisture system also sends data to an IoT platform, Ubidots, to visualise data that the user may need. Data that is sent to Ubidots is the set threshold voltage, current voltage being passed through the circuit, current soil moisture percentage and the past percentages arranged by time as well as the relay status. This is to allow the ease of communication for the user so that they are able to see the status of their urban farm at any time. The circuit is also set such that if the soil moisture detected is lower than the set threshold voltage, it would activate a relay that will turn on a water pump to water the plants. A buzzer will also be activated to alert the user to potentially check on their plants. This relay will be deactivated as soon as the sensor detects that the soil is above the threshold moisture percentage. This product allows urban farmers to worry less about having to check their plant's soil moisture, which is one of the necessities for optimal plant growth. This also eliminates the need for a physical inspection of soil moisture to determine whether the plant needs to be watered.

9. Reflection of project and service-learning

This project by far was the most interesting to work on, as the service-learning sessions were live, though it was through Microsoft Teams. I had exposure to the issue and was able to visualize what my project was being used for. During the service-learning, our partner had shown us his physical set-up of hydroponic farms and had shared with us how having IoT systems would further improve each subsystem. This allowed me to fully understand how the sensor that was provided can be used in the system. Furthermore, each practical in IRP4 had activities that slowly introduced us to components that we would be using for the project, which further helped me understand each component better at a good pace. Additionally, I felt a stronger sense of connection to this project as I myself believe that Singapore is in a vulnerable state in terms of food safety, especially because of the recent COVID-19. Climate change too is a large factor in food stability and this project allowed me to learn more about the efforts that Singapore is doing to maintain our food safety, with the 30 by 30 plan.

From the DIY kit for hydroponics we set up at the Engineering Science centre, I also learnt how I can easily grow my vegetables and create a mini hydroponics farm. With the knowledge from assessment 3 that involved researching a specific hydroponic farming type, as well as the skills I've learnt programming the soil moisture sensor, I believe that I would be able to make my own efficient hydroponic farm system at home

10. Risk Assessment in context to urban farming

Department:	For a Green Future Co.	**RA Leader:**	**SITI JASMEEN BINTE MOHAMAD ALI**	**Approved by**		**Reference Number**
Process:	Hydroponic farm maintenance	**RA Member 1:**				FGFC/M/01
Process/Activity Location:	420 Choa Chu Kang Ave 4, #01-06	**RA Member 2:**		Signature:		
Original Assessment date:	26/ January /2022	**RA Member 3:**		Name:	Dr Jaslyn Law	
Last review date:	26/ January /2022	**RA Member 4:**		Designation:	Senior Lecturer	
Next review date:	26/ January /2022	**RA Member 5:**		Date:	26/ January /2022	

Risk Assessment Form

HAZARD IDENTIFICATION				RISK EVALUATION				RISK CONTROL						
Ref	Work Activity	Hazard	Possible injury/ill-health	Existing risk controls	S	L	RPN	Additional Controls	S	L	RPN	Implementation Person	Due Date	Remarks
1	Usage of gardening/ farm tools	Careless use of rusty farm tools	Cuts	Keep a unexpired first aid kit within easy access	2	4	8	Train employees proper tool handling methods	1	2	2	JASMEEN	17 JANUARY 2022	
				Keep tools clean and store in dry place	2	1	2							
				Use appropriate tools for relevant tasks	2	3	6							
		Misuse of electronic equipment such as lights, electric pumps, etc	Unconsiousness from electrocution, possibly leading to death, Electric burns to internal tissue and orgrans or external skin	Ensure all electronics are stored and covered appropriately (e.g. Ensure wires are not exposed)	4	2	8	Ensure sockets are not overloaded	4	1	4	JASMEEN	12 JANUARY 2022	
				Ensure electrical circuit breaker	4	2	8		4	1	4			

				is clear at all times for emergency use Train employees how to use electrical appliances that are used in the urban farm correctly	4	3	12	Use ground fault circuit interrupters to ensure fast circuit break						
		Constant exposure to pesticides used for keeping pests away	Allergies and skin irritations Lung irritation	Wear masks to ensure worker does not inhale harmful gas emitted by the pesticides	2	2	4							
2	Movement in urban farm	Wet floors due to overflowing water tanks/ spillage	Slip and fall causing physical injuries (bruises)	Have workers wear non slip boots Conduct regular checks on water tanks to ensure no overflow/ structural failures	2 2	2 2	4 4	Place no running signs to remind workers not to run	2	2	4	JASMEEN	12 JANUARY 2022	

		Prolonged standing under the hot sun while manning the plants	Sunburn, Tiredness and dehydration possibly leading to sun poisoning or heat stroke	Have workers wear long sleeve white clothing and sun hats	2	2	4	Place reminders to drink up during the work day	2	1	2	JASMEEN	12 JANUARY 2022	
				Install retractable roofs that open in high	2	2	4							
				Include water dispensing services like water cooler/ vending machines	2	2	4							
				Include rest areas that are sheltered for frequent breaks during hot weather	2	2	4							
		Height Risk	Possible falling risk that may lead to serious injuries and death	Ensure railings places to prevent people from falling off the roof	2	1	2					JASMEEN	12 JANUARY 2022	

				Ensure that no farms are set up close to the edge of the roof to remove the risk of falling	2	1	2							
		Exposure to rodents and insects	Risk of the spread of diseases in plants and workers	Place fences or mesh up such that breeding of such rodents and insects like mosquitos are prevented	2	2	4	Place Mozzie Wipeout posters around the work area to serve as a reminder to ensure stagnant water sources are removed	2	1	2	JASMEEN	12 JANUARY 2022	
3	General Cleaning	Regular use of cleaning chemicals	Allergies and skin irritations	Use gloves when doing heavy cleaning tasks Use milder chemicals for other cleaning tasks Conduct constant cleans to prevent build up of dirt	1 1 1	1 1 1	1 1 1					JASMEEN	12 JANUARY 2022	

4	Storage	Improper handling of heavy and bulky objects	Strains and bruises	Use mechanical aids for carrying heavy loads	2	3	6					JASMEEN	12 JANUARY 2022	

11. References (not more than 1 page)

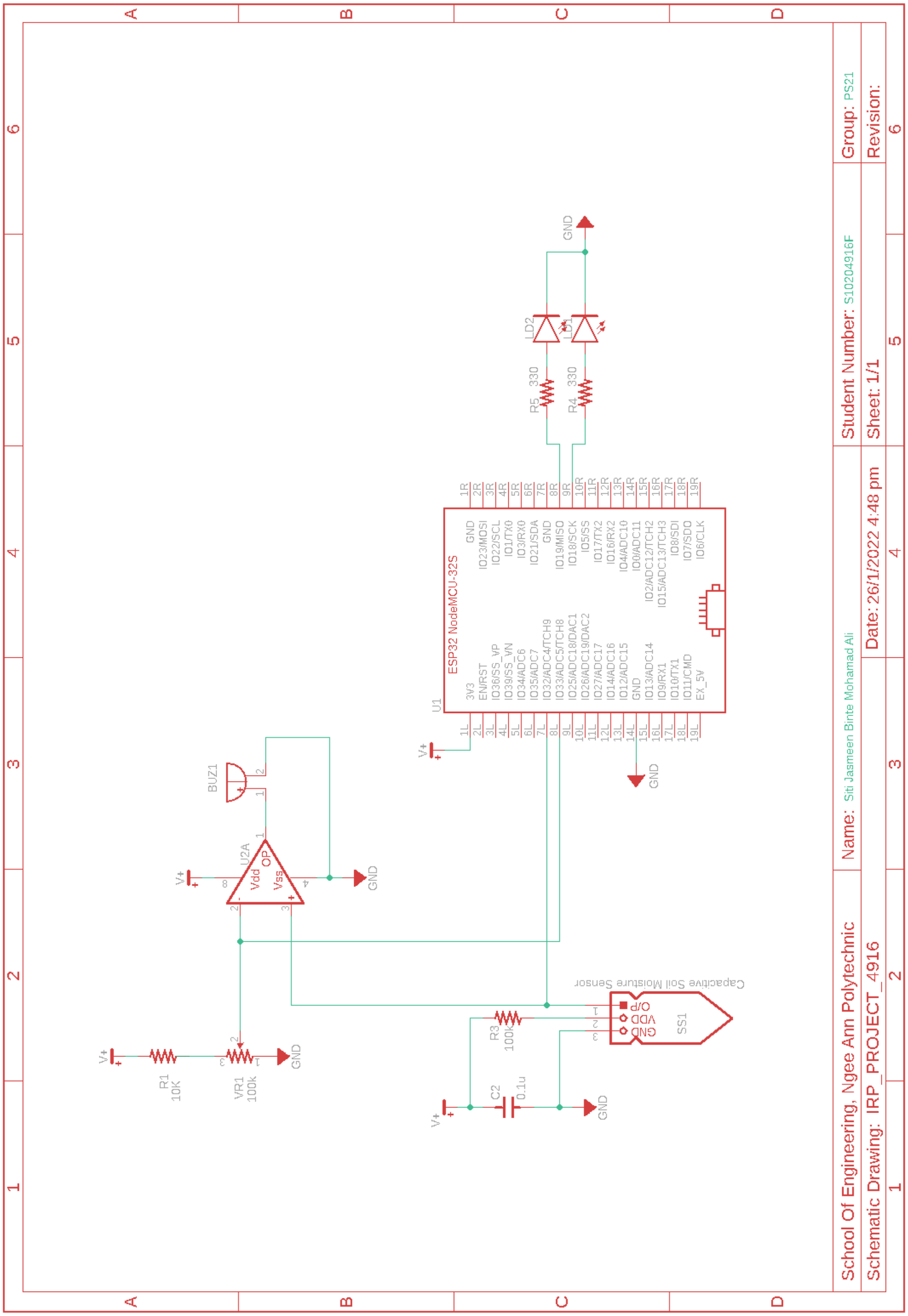
[1] “Overwatering,” *Missouribotanicalgarden.org*. [Online]. Available: https://www.missouribotanicalgarden.org/gardens-gardening/your-garden/help-for-the-home-gardener/advice-tips-resources/pests-and-problems/environmental/overwatering.aspx. [Accessed: 31-Jan-2022].

[2] “Tutorial - using capacitive soil moisture sensors on the Raspberry Pi,” *SwitchDoc Labs Blog*, 17-Jun-2020. [Online]. Available: https://www.switchdoc.com/2020/06/tutorial-capacitive-moisture-sensor-grove/. [Accessed: 01-Feb-2022].

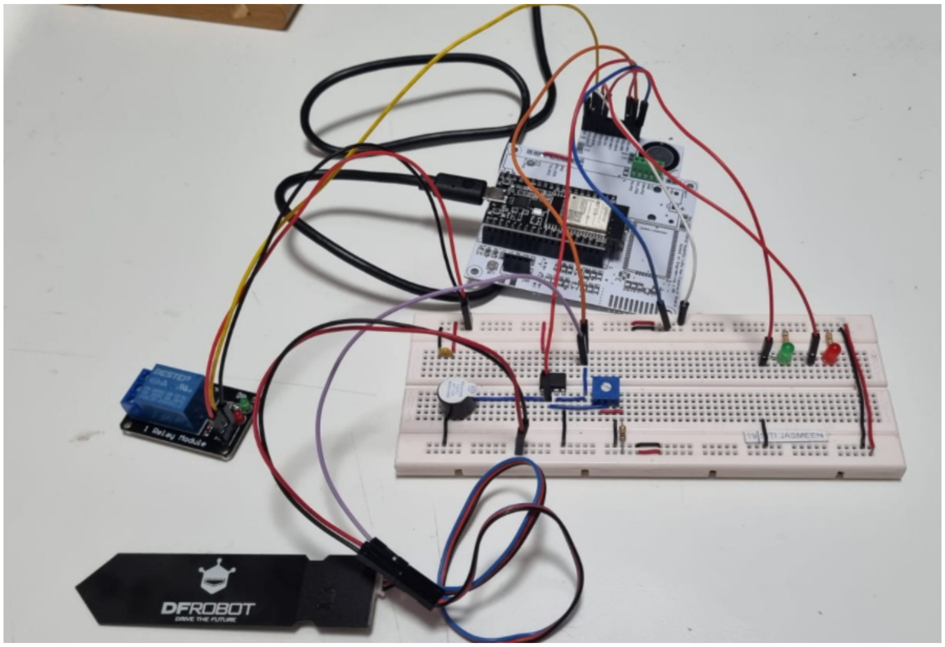
[3] “Comparing soil moisture sensors for smart irrigation systems,” *Arduino Project Hub*. [Online]. Available: https://create.arduino.cc/projecthub/antonio-ruiz/comparing-soil-moisture-sensors-for-smart-irrigation-systems-caa7aa. [Accessed: 01-Feb-2022].

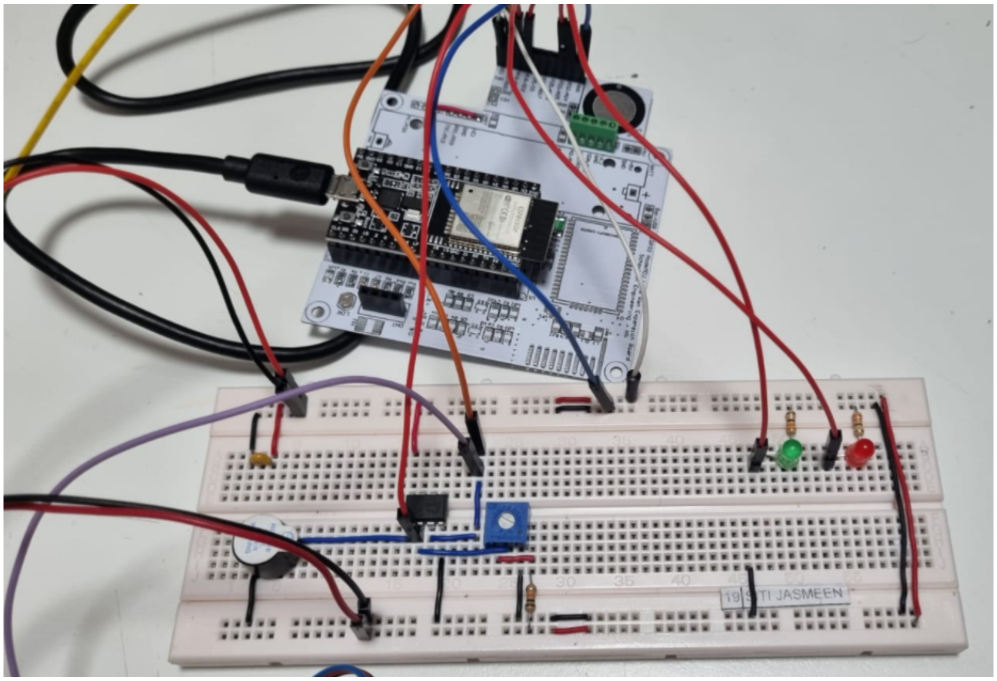
12. Appendix

9.1 Sensor Schematic

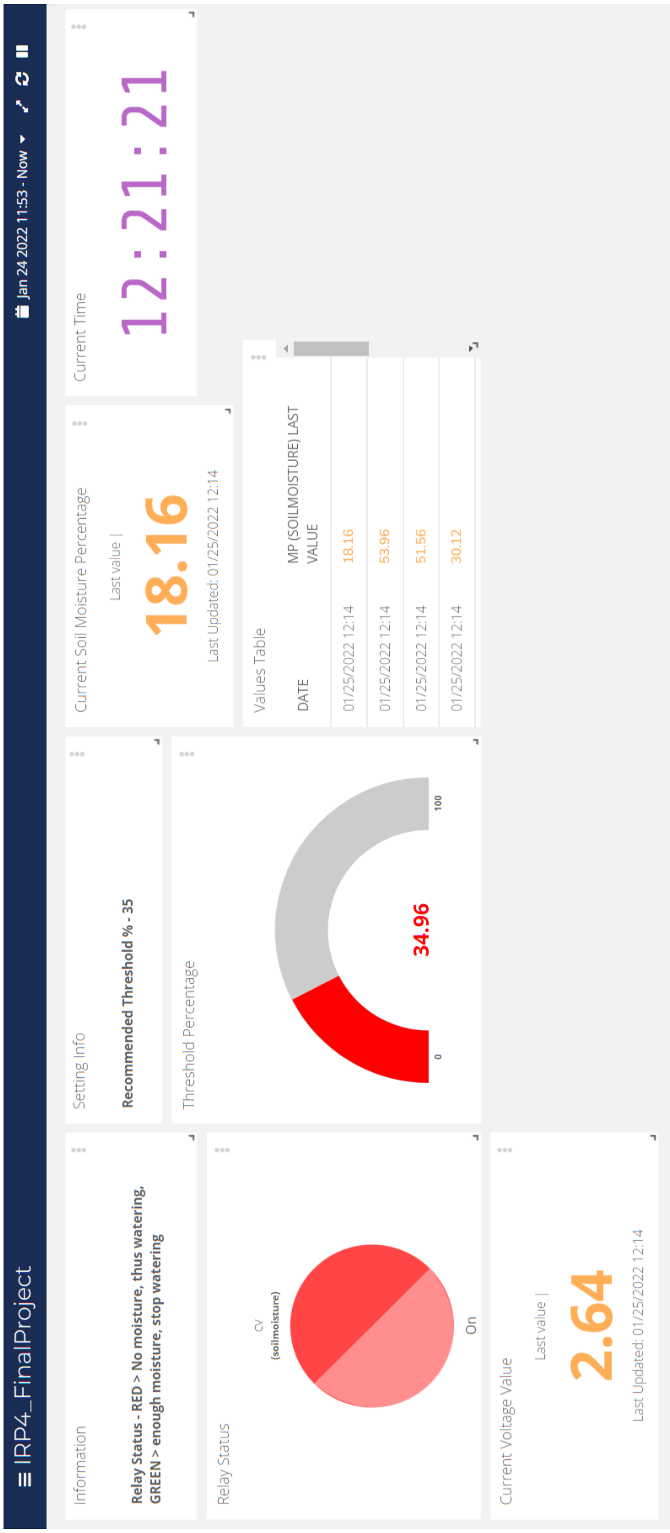


9.2 Picture of a prototype on a breadboard





9.3 Picture of the built dashboard



≡ IRP4_FinalProject

Jan 24 2022 11:53 - Now ▾

Information

Relay Status - RED > No moisture, thus watering, GREEN > enough moisture, stop watering

Setting Info

Recommended Threshold % - 35

Current Soil Moisture Percentage

Last value |

18.16

Last Updated: 01/25/2022 12:14

Current Time

12:21:21

Relay Status

Threshold Percentage

Values Table

DATE	MP (SOILMOISTURE) LAST VALUE
01/25/2022 12:14	18.16
01/25/2022 12:14	53.96
01/25/2022 12:14	51.56
01/25/2022 12:14	30.12

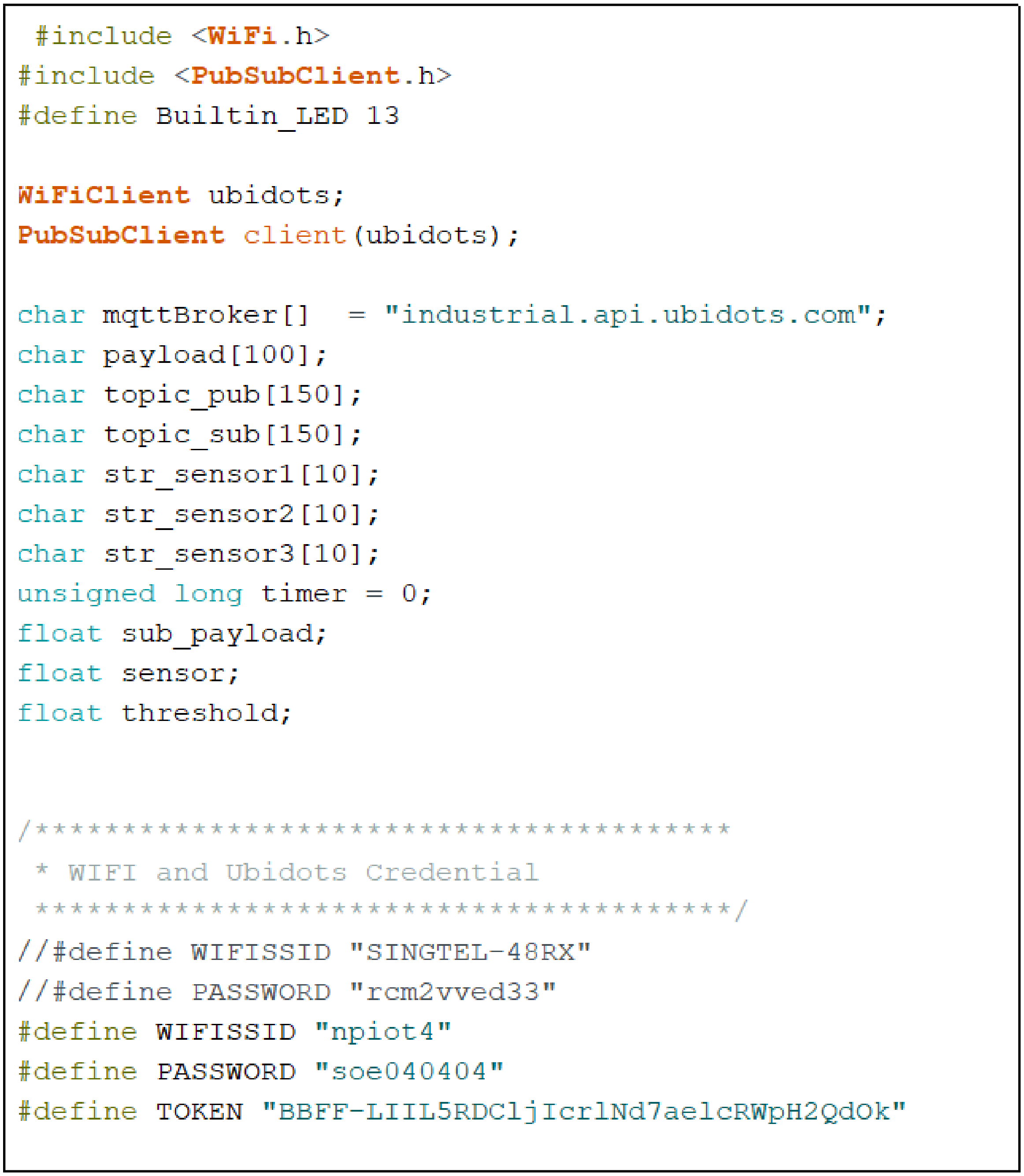
Current Voltage Value

Last value |

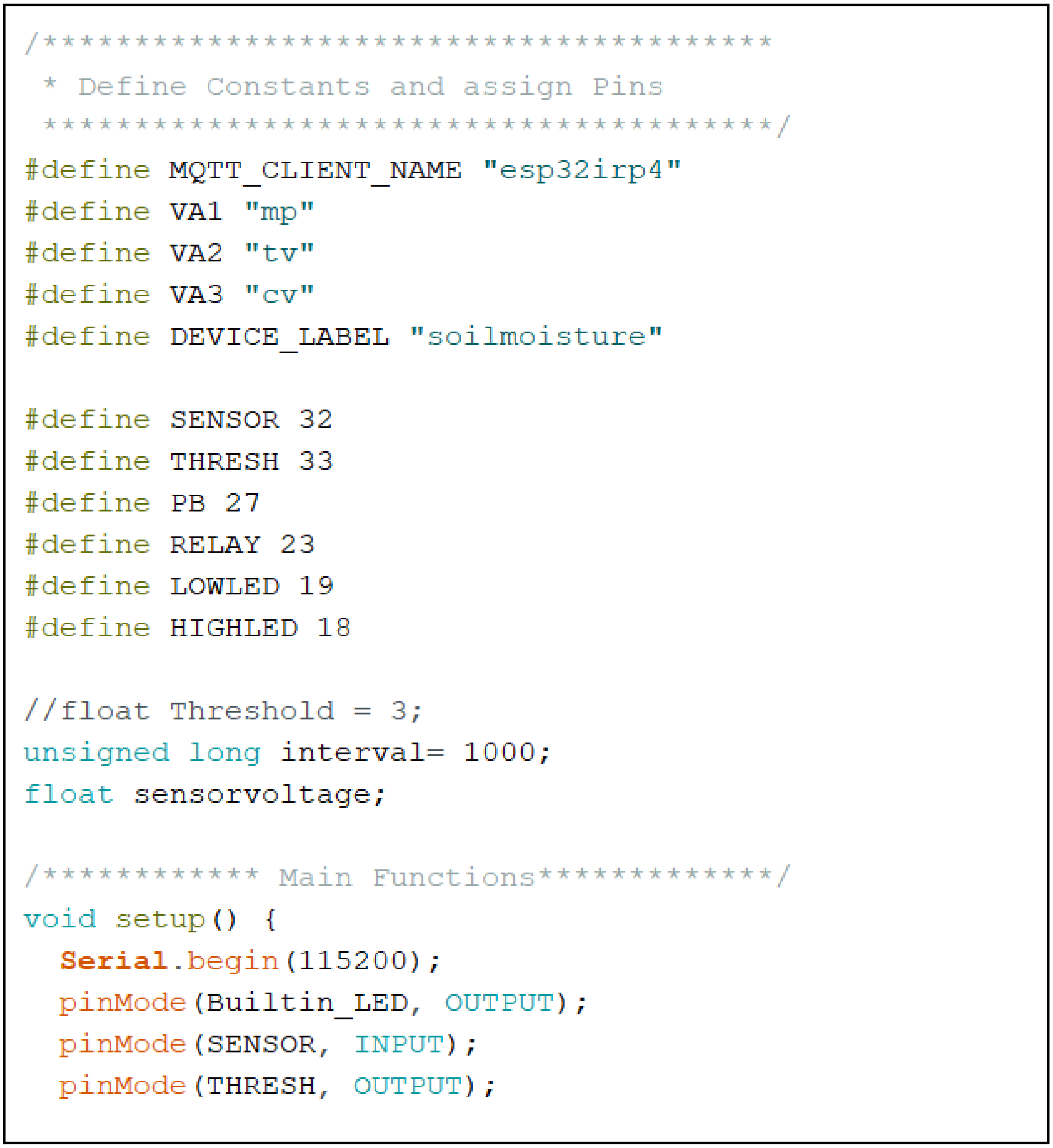
2.64

Last Updated: 01/25/2022 12:14

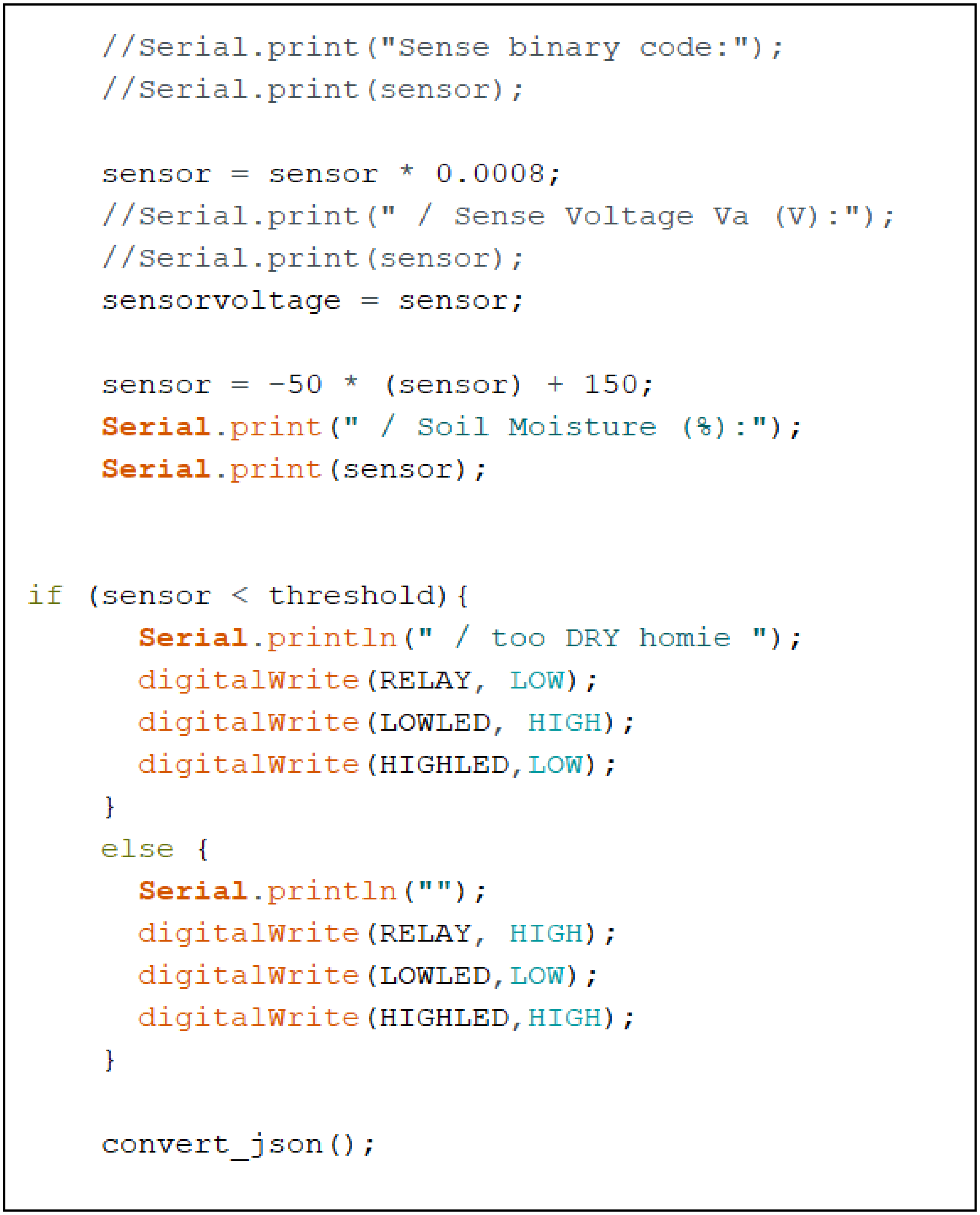
9.4 Source code for the prototype



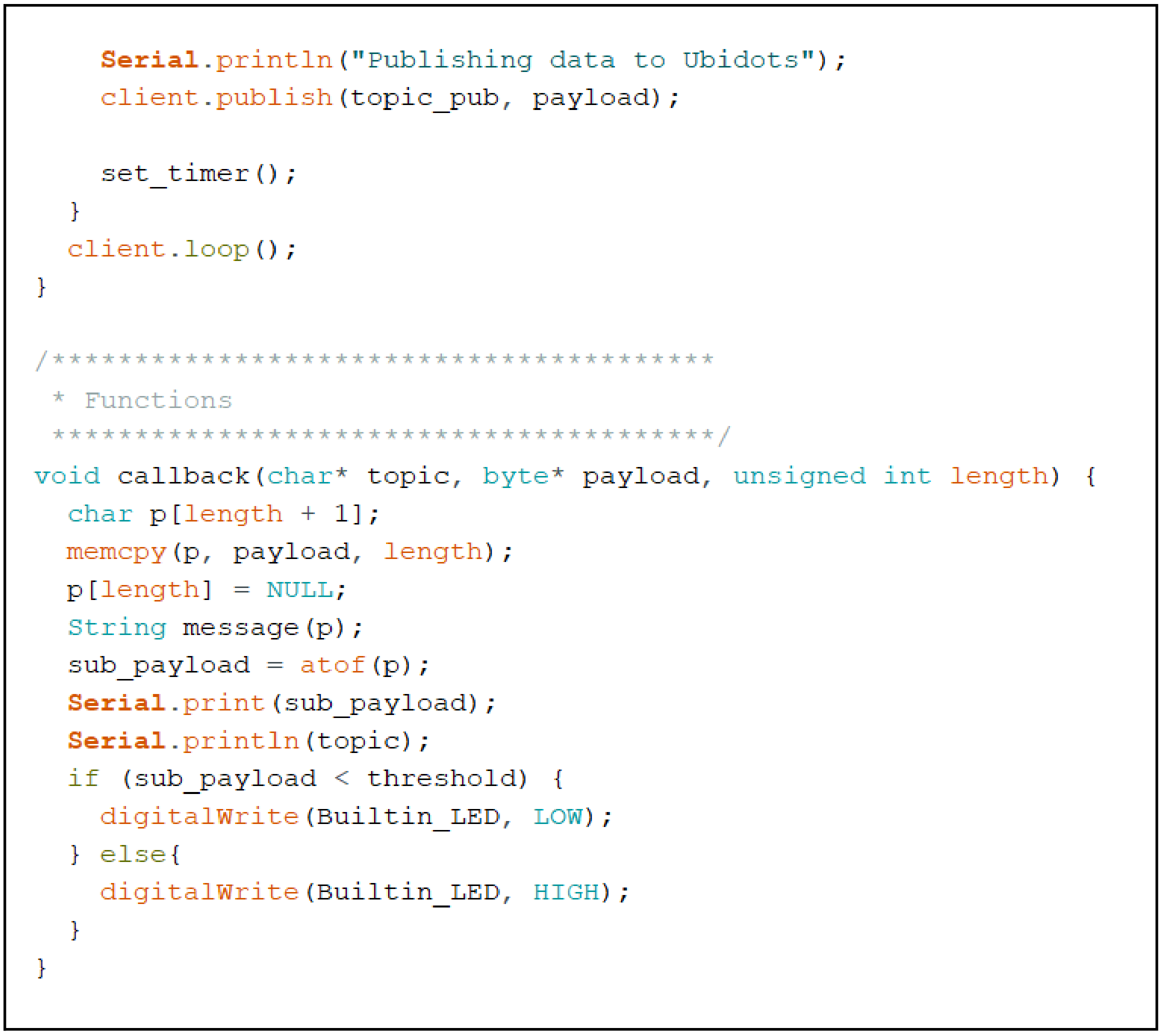
```
 #include <WiFi.h>
#include <PubSubClient.h>
#define Builtin_LED 13

WiFiClient ubidots;
PubSubClient client(ubidots);

char mqttBroker[]  = "industrial.api.ubidots.com";
char payload[100];
char topic_pub[150];
char topic_sub[150];
char str_sensor1[10];
char str_sensor2[10];
char str_sensor3[10];
unsigned long timer = 0;
float sub_payload;
float sensor;
float threshold;


/*****************************************
 * WIFI and Ubidots Credential
 *****************************************/
//#define WIFISSID "SINGTEL-48RX"
//#define PASSWORD "rcm2vved33"
#define WIFISSID "npiot4"
#define PASSWORD "soe040404"
#define TOKEN "BBFF-LIIL5RDCljIcrlNd7aelcRWpH2QdOk"
```



```
/****************************************
 * Define Constants and assign Pins
 ****************************************/
#define MQTT_CLIENT_NAME "esp32irp4"
#define VA1 "mp"
#define VA2 "tv"
#define VA3 "cv"
#define DEVICE_LABEL "soilmoisture"

#define SENSOR 32
#define THRESH 33
#define PB 27
#define RELAY 23
#define LOWLED 19
#define HIGHLED 18

//float Threshold = 3;
unsigned long interval= 1000;
float sensorvoltage;

/************ Main Functions*************/
void setup() {
  Serial.begin(115200);
  pinMode(Builtin_LED, OUTPUT);
  pinMode(SENSOR, INPUT);
  pinMode(THRESH, OUTPUT);
  pinMode(PB, INPUT);
  pinMode(RELAY, OUTPUT);
  digitalWrite(Builtin_LED, HIGH);
  pinMode(LOWLED, OUTPUT);
  pinMode(HIGHLED, OUTPUT);
  //digitalWrite(RELAY, HIGH);
  //digitalWrite(LED, HIGH);
  iot_setup();
}

void loop() {
  if (millis() > timer){
  if (!client.connected()) {
    mqttconnect();
    client.subscribe(topic_sub);
  }
//dry 35%
//wet is 65%
    threshold = analogRead(THRESH)+ 150 ;
    threshold = threshold * 0.0008;
    threshold = -50 * (threshold) + 150;

    Serial.print("Manual Set Threshold Moisture (%):");
    Serial.print(threshold);


    sensor = analogRead(SENSOR)+ 150 ;
```

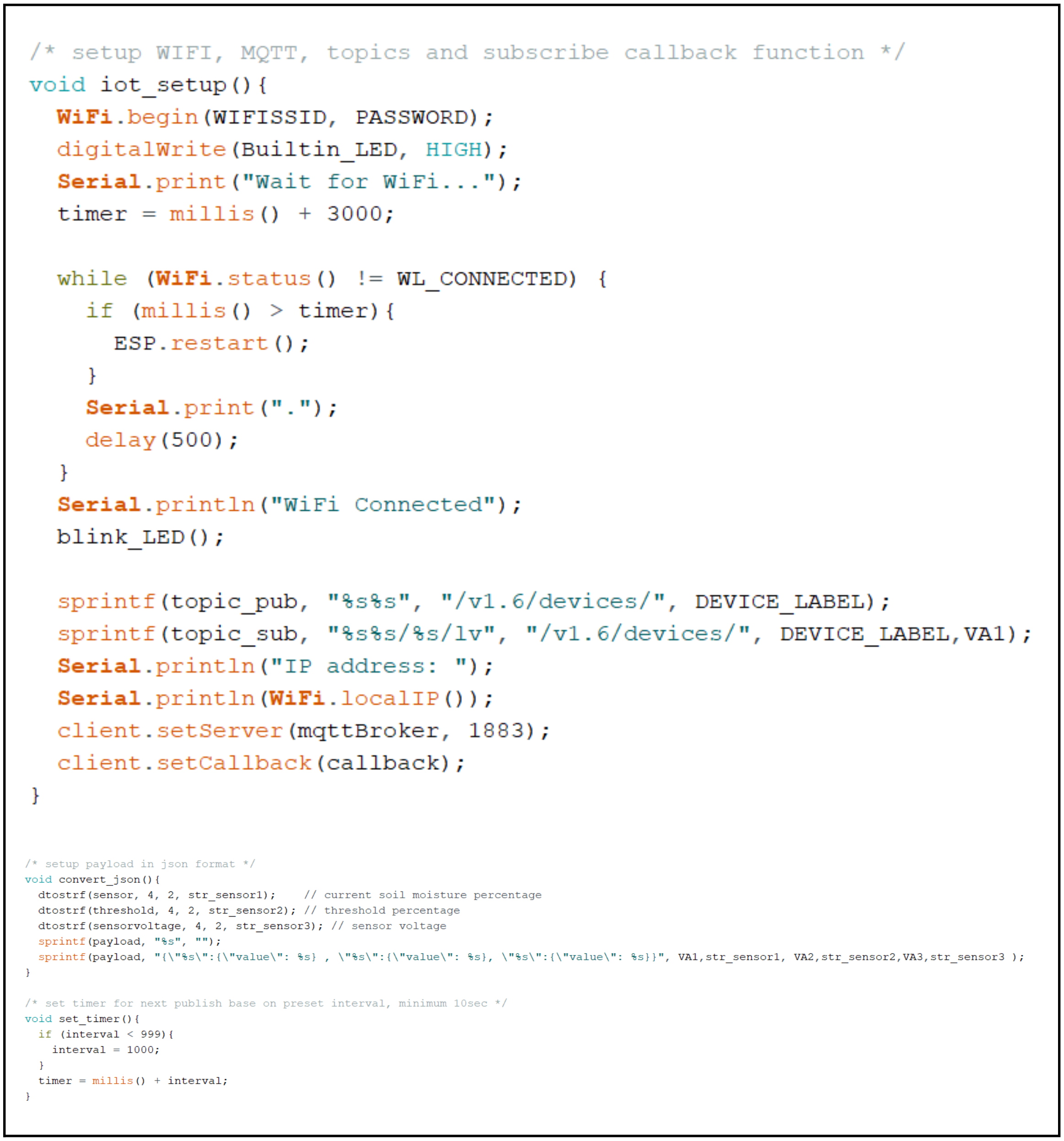


```
    //Serial.print("Sense binary code:");
    //Serial.print(sensor);

    sensor = sensor * 0.0008;
    //Serial.print(" / Sense Voltage Va (V):");
    //Serial.print(sensor);
    sensorvoltage = sensor;

    sensor = -50 * (sensor) + 150;
    Serial.print(" / Soil Moisture (%):");
    Serial.print(sensor);


if (sensor < threshold){
      Serial.println(" / too DRY homie ");
      digitalWrite(RELAY, LOW);
      digitalWrite(LOWLED, HIGH);
      digitalWrite(HIGHLED,LOW);
    }
    else {
      Serial.println("");
      digitalWrite(RELAY, HIGH);
      digitalWrite(LOWLED,LOW);
      digitalWrite(HIGHLED,HIGH);
    }

    convert_json();
```



```
    Serial.println("Publishing data to Ubidots");
    client.publish(topic_pub, payload);

    set_timer();
  }
  client.loop();
}

/****************************************
 * Functions
 ****************************************/
void callback(char* topic, byte* payload, unsigned int length) {
  char p[length + 1];
  memcpy(p, payload, length);
  p[length] = NULL;
  String message(p);
  sub_payload = atof(p);
  Serial.print(sub_payload);
  Serial.println(topic);
  if (sub_payload < threshold) {
    digitalWrite(Builtin_LED, LOW);
  } else{
    digitalWrite(Builtin_LED, HIGH);
  }
}
```

```
/* Connect to MQTT broker */
void mqttconnect() {
    while (!client.connected()) {
    Serial.println("Attempting MQTT connection...");
    if (client.connect(MQTT_CLIENT_NAME, TOKEN, "")) {
      Serial.println("Connected");
    } else {
      Serial.print("Failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 2 seconds");
      delay(2000);
    }
   }
}

/* Brink builtin_LED twice */
void blink_LED(){
  digitalWrite(Builtin_LED, LOW);
  delay(500);
  digitalWrite(Builtin_LED, HIGH);
  delay(500);
  digitalWrite(Builtin_LED, LOW);
  delay(500);
  digitalWrite(Builtin_LED, HIGH);
  delay(1000);
}
```



```
/* setup WIFI, MQTT, topics and subscribe callback function */
void iot_setup(){
  WiFi.begin(WIFISSID, PASSWORD);
  digitalWrite(Builtin_LED, HIGH);
  Serial.print("Wait for WiFi...");
  timer = millis() + 3000;

  while (WiFi.status() != WL_CONNECTED) {
    if (millis() > timer){
      ESP.restart();
    }
    Serial.print(".");
    delay(500);
  }
  Serial.println("WiFi Connected");
  blink_LED();

  sprintf(topic_pub, "%s%s", "/v1.6/devices/", DEVICE_LABEL);
  sprintf(topic_sub, "%s%s/%s/lv", "/v1.6/devices/", DEVICE_LABEL,VA1);
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  client.setServer(mqttBroker, 1883);
  client.setCallback(callback);
}
```

```
/* setup payload in json format */
void convert_json(){
  dtostrf(sensor, 4, 2, str_sensor1);    // current soil moisture percentage
  dtostrf(threshold, 4, 2, str_sensor2); // threshold percentage
  dtostrf(sensorvoltage, 4, 2, str_sensor3); // sensor voltage
  sprintf(payload, "%s", "");
  sprintf(payload, "{\"%s\":{\"value\": %s} , \"%s\":{\"value\": %s}, \"%s\":{\"value\": %s}}", VA1,str_sensor1, VA2,str_sensor2,VA3,str_sensor3 );
}

/* set timer for next publish base on preset interval, minimum 10sec */
void set_timer(){
  if (interval < 999){
    interval = 1000;
  }
  timer = millis() + interval;
}
```