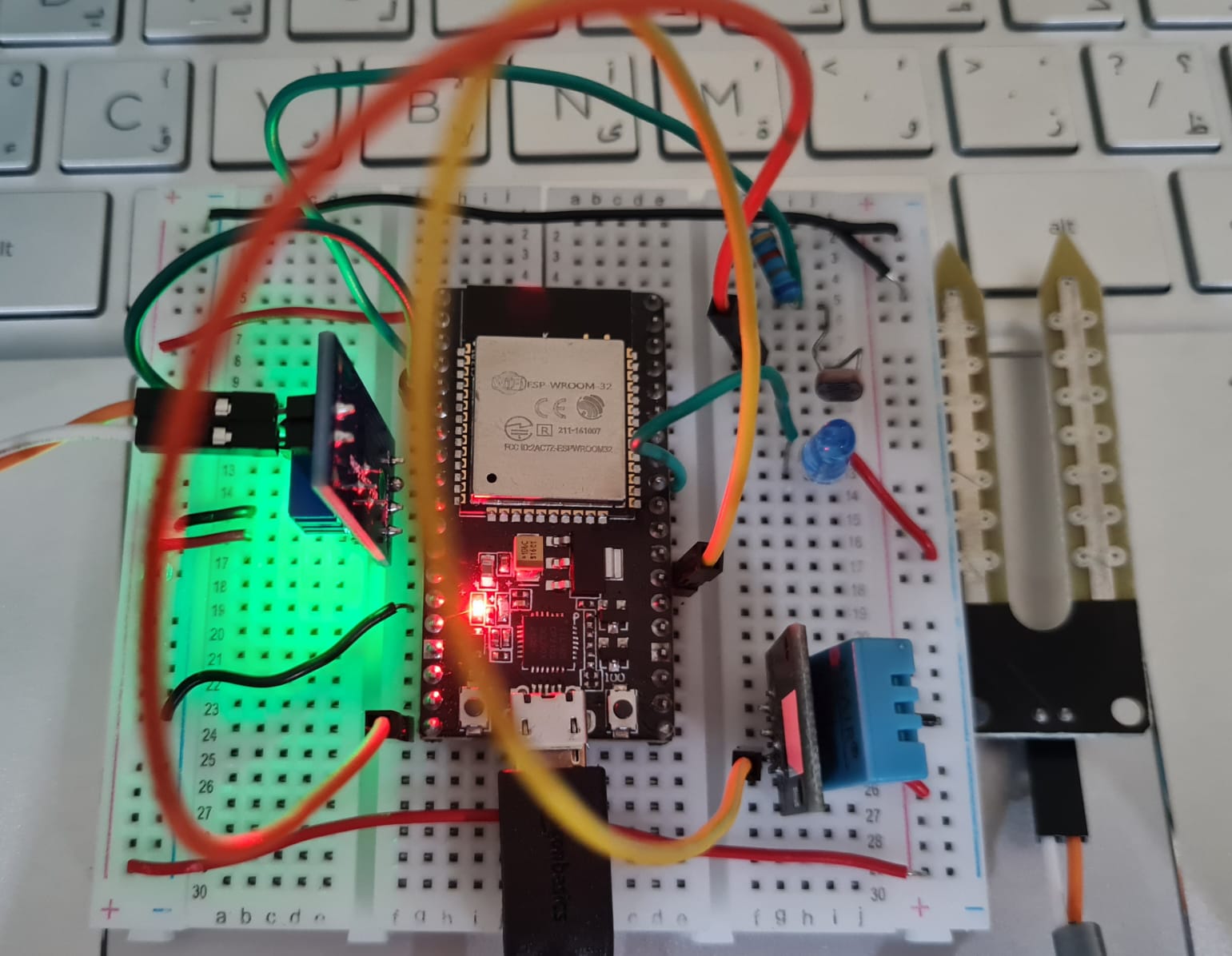
ECTE474 – Internet of things

PROJECT REPORT

SUBMITTED BY:

Razveen Kashif (7025014) | 100% Clayton D’Gama (7242293) | 100% Suraksha Kotte (6996656) | 100% Suraksha Kotte (6996656) | 100%



# Introduction

Our project's objective is to help set up an Internet of Things (IoT) solution for Jamal, a nearby farmer who is on holiday in Europe and is seeking help in remotely monitoring and controlling the conditions of his blueberry plant.

To answer Jamal's concerns, the creation of a web-based and mobile dashboard was suggested. Important environmental variables like temperature, humidity, light intensity, and soil moisture would be monitored in real time via this dashboard.

This system would be operational throughout the day to provide continuous monitoring. Additionally, a notification system is implemented, that shows a status flag that tells us if the plant is operating at optimal efficiency. For instance, the indicator will turn false, if the soil moisture falls below 30%. An ESP32S board, several sensors, which include an LED, an LDR for light detecting, a DHT11 sensor for temperature and humidity measurement, and a soil moisture sensor will all be used in the implementation of this project.

In summary, the proposed Internet of Things solution seeks to maintain the ideal growing environment for Jamal’s blueberry plant during his vacation.

# Objective

1. Provide a mobile and web-based dashboard that allows for the real-time monitoring of environmental variables such soil moisture, temperature, humidity, and light intensity.
2. Create a notification system that will notify Jamal, the farmer, if the plant's optimal circumstances are not met.
3. Establish a database that allows real-time data collection to be uploaded to a Google spreadsheet for convenient analysis and preservation.
4. Utilize the ESP32S board along with sensors such as a soil moisture sensor, DHT11 sensor for temperature and humidity, LDR for light sensing, and an LED for controlling the artificial light source.
5. Ensure the blueberry plant remains in optimal conditions, contributing to sustainable and efficient farming practices.

# Methodology

The code was created to develop a smart plant monitoring system using an ESP32S microcontroller. It measures temperature, humidity, light intensity, and soil moisture using a number of sensors. After that, the gathered data is sent via Wi-Fi to a Google Spreadsheet for analysis and central storage.  
  
The first line of the code includes the libraries required for sensor interface, HTTP communication, and Wi-Fi connectivity. it offers variables for pin assignments.

#include "WiFi.h"

#include <HTTPClient.h>

#include <DHT.h>

#define LED 18

#define soil\_pin 34

#define ldr\_pin 36

#define On\_Board\_LED\_PIN 2

const char\* ssid = "WIFI\_NAME"; //--> Your wifi name

const char\* password = "WIFI\_PASSWORD"; //--> Your wifi password

// Google script Web\_App\_URL.

String Web\_App\_URL = "https://script.google.com/macros/s/AKfycby3-vTqRDAkjdjqiTCw2de6cA-fdP1JSHjXocqVW\_5O7ajZAY8N2HTEaZ9HYraaH40/exec";

//initialize the DHT object

DHT dht(4,DHT11);

bool ideal;

The code in the setup, creates a Wi-Fi connection during initialization in order to allow seamless interaction with the internet, which is joined with a designated Wi-Fi network set up by the group. Alongside this, the defined pins are set as outputs/inputs and calling the sensors.

void setup() {

Serial.begin(9600);

//inputs

pinMode(soil\_pin,INPUT);

pinMode(ldr\_pin,INPUT);

pinMode(4,INPUT); // for DHT

//output

//initialise LED

pinMode(LED,OUTPUT);

//start DHT object

dht.begin();

//----------------------------------------Set Wifi to STA mode

Serial.println();

Serial.println("-------------");

Serial.println("WIFI mode : STA");

WiFi.mode(WIFI\_STA);

Serial.println("-------------");

//----------------------------------------

//----------------------------------------Connect to Wi-Fi (STA).

Serial.println();

Serial.println("------------");

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

The most effective environmental conditions for plant development are determined by continually monitoring sensor data and comparing them to specified limits in the main loop of the code. The soil moisture sensor and light-dependent resistor (LDR) provide analogue values, which are converted to percentages, while the DHT11 sensor measures temperature and humidity.

void loop()

{

//read soil moisture as percentage

float soil = (100 - ( (analogRead(soil\_pin)/4095.00) \* 100 ));

//read light intensity

int light = analogRead(ldr\_pin);

//read temperature and humidity

float temp = dht.readTemperature();

float humid = dht.readHumidity();

//set conditions for setting the plant in its ideal condition

//conditions to set the status as true or false

if ((light > 600) && (soil >= 30) && (temp >22) && (temp < 27) && (humid>=65) && (humid <80)){

ideal = true;

}

else {

ideal = false;

}

The below code manages the LED attached to the ESP32S which acts as an artificial light source. The LED is switched on to improve the available light if the light intensity is less than 600, which indicates low light circumstances.

//turn on the led if the brightness of the ldr is too low

if (light <= 600){

digitalWrite(LED,HIGH);

}

else if (light > 600){

digitalWrite(LED,LOW);}

The code creates a URL with the sensor data and makes an HTTP GET request to a Google Spreadsheet as soon as a successful Wi-Fi connection is made. This makes it easier to log data centrally for later analysis and to monitor remotely.

if (WiFi.status() == WL\_CONNECTED) {

// Create a URL for sending or writing data to Google Sheets.

String Send\_Data\_URL = Web\_App\_URL + "?sts=write";

//Send\_Data\_URL += "&srs=" + Status\_Read\_Sensor;

Send\_Data\_URL += "&temp=" + String(temp);

Send\_Data\_URL += "&humid=" + String(humid);

Send\_Data\_URL += "&moist=" + String(soil);

Send\_Data\_URL += "&light=" + String(light);

Serial.println();

Serial.println("-------------");

Serial.println("Send data to Google Spreadsheet...");

Serial.print("URL : ");

Serial.println(Send\_Data\_URL);

// Initialize HTTPClient as "http".

HTTPClient http;

// HTTP GET Request.

http.begin(Send\_Data\_URL.c\_str());

//http.setFollowRedirects(HTTPC\_STRICT\_FOLLOW\_REDIRECTS);

// Gets the HTTP status code.

int httpCode = http.GET();

Serial.print("HTTP Status Code : ");

Serial.println(httpCode);

// Getting response from google sheets.

String payload;

if (httpCode > 0) {

payload = http.getString();

Serial.println("Payload : " + payload);

}

http.end();

To ensure a periodic update of sensor readings, the code introduces a delay of 60 seconds before repeating the loop.

delay(60000);

# Results

The results provided by the code were able to meet the requirements of the client Jamal. Firstly, the dashboard along with the data results in the spreadsheet were able to display the real-time values.

A screenshot of a computer

Description automatically generated

Figure 1: Spreadsheet with data values and dashboard.

In the dashboard, the status/ideal condition is flagged as “False”, and the data is updated regularly (every minute) to ensure consistent monitoring over the plant. Additionally, the circuitry would also reflect the changes when the plant was not said to be in ideal condition. Where the artificial light remains on despite the external light being present.

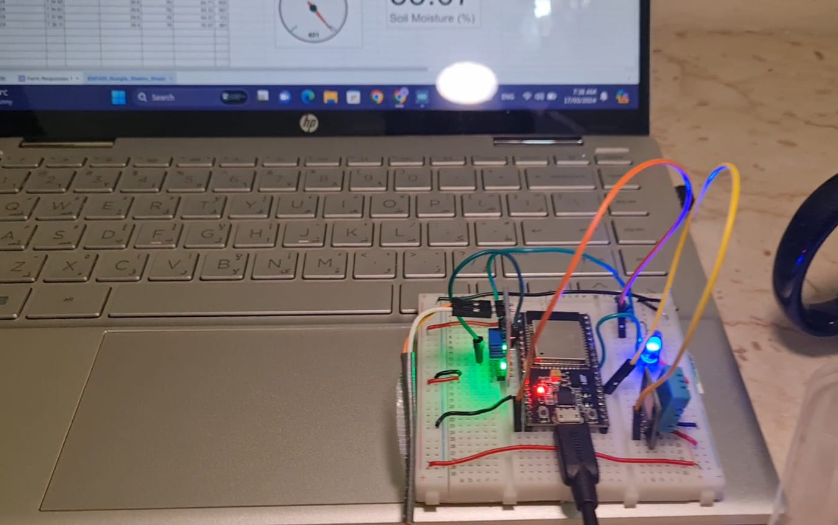


Figure 2: LED remains "ON" due to low threshold value of light.

# Flowchart

This section illustrates how the data flows within the IoT based plant monitoring system. Based on the requirements of Jamal, the sensors for the temperature, humidity (sensed by DHT11), light (LDR) and soil moisture are the inputs to the system. These are then processed and from the read values and compared to the earlier discussed thresholds. The outputs, which are the LED, and the status “Ideal” flag are set to be ON/OFF and True/False, respectively.

A white rectangular sign with black text

Description automatically generated

Figure 3: Context Diagram

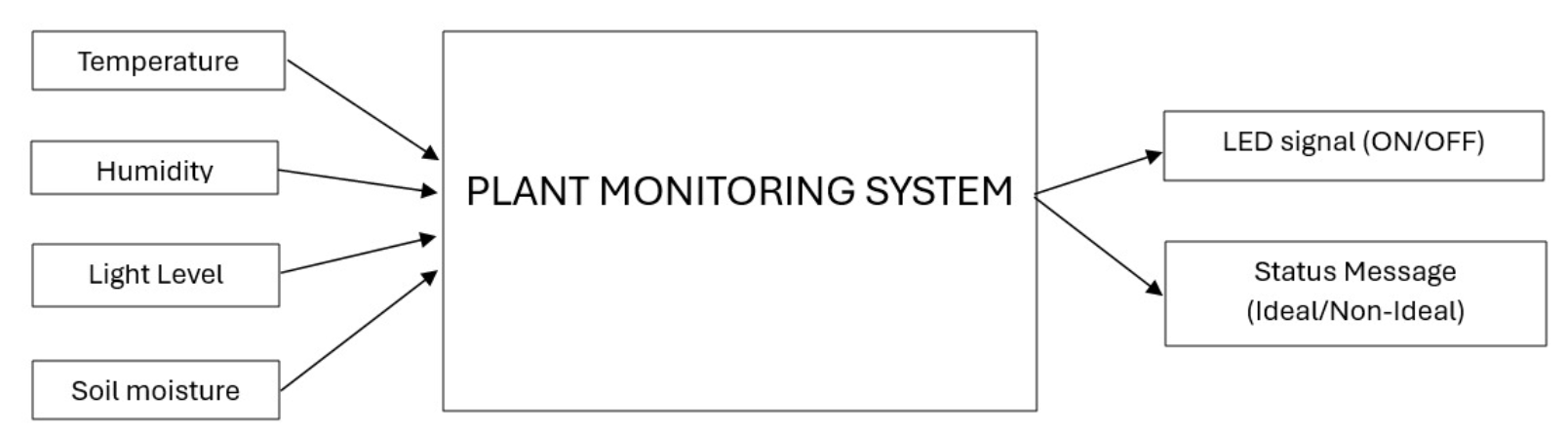


Figure 4: Detailed data flow diagram of the IoT plant monitoring system.

# Conclusion

In summary, this IoT project’s completed code gives a strong solution for plant management and monitoring, meeting the demands of the client Jamal who needed a dependable system to watch over his blueberry plant while he was on vacation. The solution successfully meets the project's needs by offering a mobile and web-based dashboard for tracking key environmental data including soil moisture, temperature, humidity, and light intensity.