Soil Analyzer IoT

Project Report submitted in partial fulfillment of
The requirements for the degree of

MASTER OF COMPUTER APPLICATION

Of

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY

By

Subhodeep Sarkar, Roll no: 29171021011
Birupaksha Dey, Roll no: 29171021028
Soutik Mahanta, Roll no: 29171021036
Subhajyoti Mondal, Roll no: 29171021039

Under the guidance of

Prof. Piyali Mukherjee

DEPARTMENT OF COMPUTER APPLICATION

NETAJI SUBHASH ENGINEERING COLLEGE
TECHNO CITY, GARIA, KOLKATA – 700 152

2022-2023

CERTIFICATE

This is to certify that this project report titled Soil Analyzer IoT submitted in partial fulfillment of requirements for award of the degree Master of Computer Application of West Bengal University of Technology is a faithful record of the original work carried out by,

Subhodeep Sarkar, **Roll no**. 29171021011, **Regd. No.** 212910571010014 (2021)

Birupaksha Dey, **Roll no**. 29171021028, **Regd. No.** 212910571010034 (2021)

Soutik Mahanta, **Roll no**. 29171021036, **Regd. No.** 212910571010035 (2021)

Subhajyoti Mondal, **Roll no**. 29171021039, **Regd. No.** 212910571010039 (2021)

under my guidance and supervision.

It is further certified that it contains no material, which to a substantial extent has been submitted for the award of any degree/diploma in any institute or has been published in any form, except the assistances drawn from other sources, for which due acknowledgement has been made.

Date:………..

Guide's signature

Prof. Piyali Mukherjee

Sd/__________________

Head of the Department

Department of Computer Application

NETAJI SUBHASH ENGINEERING COLLEGE
TECHNO CITY, GARIA, KOLKATA – 700 152

DECLARATION

We hereby declare that this project report titled

Soil Analyzer IoT

is our own original work carried out as a under graduate student in Netaji Subhash Engineering College except to the extent that assistances from other sources are duly acknowledged.

All sources used for this project report have been fully and properly cited. It contains no material which to a substantial extent has been submitted for the award of any degree/diploma in any institute or has been published in any form, except where due acknowledgement is made.

Student’s names:	Signatures:	Dates:
………………………..	………………………..	……………………
………………………..	………………………..	……………………
………………………..	………………………..	……………………
………………………..	………………………..	……………………

CERTIFICATE OF APPROVAL

We hereby approve this dissertation titled

Soil Analyzer IoT

carried out by

Subhodeep Sarkar, **Roll no**. 29171021011, **Regd. No.** 212910571010014 (2021)

Birupaksha Dey, **Roll no**. 29171021028, **Regd. No.** 212910571010034 (2021)

Soutik Mahanta, **Roll no**. 29171021036, **Regd. No.** 212910571010035 (2021)

Subhajyoti Mondal, **Roll no**. 29171021039, **Regd. No.** 212910571010039 (2021)

under the guidance of

Prof. Piyali Mukherjee

of Netaji Subhash Engineering College, Kolkata in partial fulfillment of requirements for award of the Master of Computer Application of West Bengal University of Technology

Date:………..

Examiners' signatures:

1. ………………………………………….
2. ………………………………………….
3. ………………………………………….

Acknowledgement and/or Dedication

Success of any project depends largely on the encouragement and guidelines of many others. We take this sincere opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project work.

We are thankful to Prof. Piyali Mukherjee for her guidance and supervision as well as for providing necessary information regarding the project and for her support for completing this project.

We are also thankful to Prof. Tanmay De and Prof. Biplab Chowdhury for their guidance and recommendations which has played a major role in extending the usability of the project.

The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

Subhodeep Sarkar

Birupaksha Dey

Soutik Mahanta

Subhajyoti Mondal

Dated:…………………

Abstract

Farmers are the backbone of India as the agriculture sector contributes to about 20% of the GDP. This is why it is very important to keep them safe from scams. In recent times people are selling land plots that are unsuitable for agriculture to farmers. When the crops are cultivated in these unsuitable lands their quality is affected which results in a fall in the selling price of these crops which leads to poverty. Along with that, the quantity of crops is also reduced drastically which leads to a shortage of crops in the market and this ultimately results in inflation. Once a farmerinvests in a plot of land it takes several years to recover the money and over the years, they incur losses. One of the main reasons for purchasing such land is because of merely relying on instincts or the words of others. The main concept is to build a soil analyzer kit using ESP-32 and various sensorswhich analyze the various parameters of the soil like atmospheric light, soil temperature, and soil moisture and matches it against the user's requirements which is provided from a web interface and shows real-time data on another web interface using WebSockets as well as on ThingSpeak cloud. All these data combined show whether that particular plot of land is suitable for the farmer or not and thus the farmer does not have to rely on mere instincts and can take the decision, based on data.

Key words: Farmers, Agriculture, ESP-32, WebSockets, ThingSpeak

CONTENTS

INTRODUCTION

Farmers have great importance in our society. They are the ones who provide us food. Since every person needs proper food for their living, so they are a necessity for the society.

There are different types of farmers, and they all have equal significance. First are the farmers who grow a crop like wheat, barley, rice, etc. Since the maximum intake in Indian households is of wheat and rice. Farmers who grow these crops are of prime importance. Second, are the ones who cultivate fruits.

These farmers have to prepare soil specially for different types of fruits. Because these fruits grow according to the season. Therefore the farmers need to have a great knowledge of fruits and crops. There are many other farmers who grow crops of other types. Furthermore, they all have to work hard to get maximum harvesting. In addition to above, the farmers contribute almost 18 per cent the Indian economy. That is the maximum of all. But still, a farmer is deprived of every luxury of society.

(1).Farming is a remarkable part of the economy in India, as it adds about 18 per cent of the absolute GDP. It gives employment to over 60 per cent of the population.

(2). Farmers are an important part of the existence of our various societies because they provide food and fiber, which gives us nutrition and cloth.

(3).Farming is an industry that relies on the natural environment.

(4).Cultivation practices often provide natural biologically active filter mechanisms for water as well as soil vegetation stabilization.

(5). Indian Farmer and farming communities provide an excellent environment to raise relations

PROBLEM STATEMENT

When a farmer is making decision to buy a new piece of land, there is a chance that he/she might buy a land which is not suitable for agriculture. For instance, the soil might not be enough moist or the soil temperature in that region surpasses the threshold required for agriculture or the soil might be too acidic or basic in nature. If any of these condition are true and the farmer is unaware of it and invests the money, it might go to vain. It will take years for the farmer to buy another piece of land. Also the crops produced will not be healthy enough and the country will suffer the consequences.

Farmers often buy a plot of land based on their instincts or influenced by others which lead to poor agriculture due to the existence of inappropriate amounts of parameters. The basic parameters that are responsible for proper agriculture are atmospheric light, soil moisture, and soil temperature. Different crops require a different amount of these parameters. This is where soil analyzing & monitoring is useful. The user can enter the required minimum and maximum values and the soil is tested against those values to see if it matches the requirement or not. In this way, the decision to buy the land or not is backed up by real-time data

Crop	Min. Temperature (°C)	Max. Temperature (°C)	Min. Moisture (%)	Max. Moisture (%)	Min. Light (%)	Max. Light (%)
Bajra	20	30	40	60	70	80
Paddy	21	37	55	80	60	73
Wheat	15	18	70	83	85	90

(Table 1: Required parameters for growing crops)

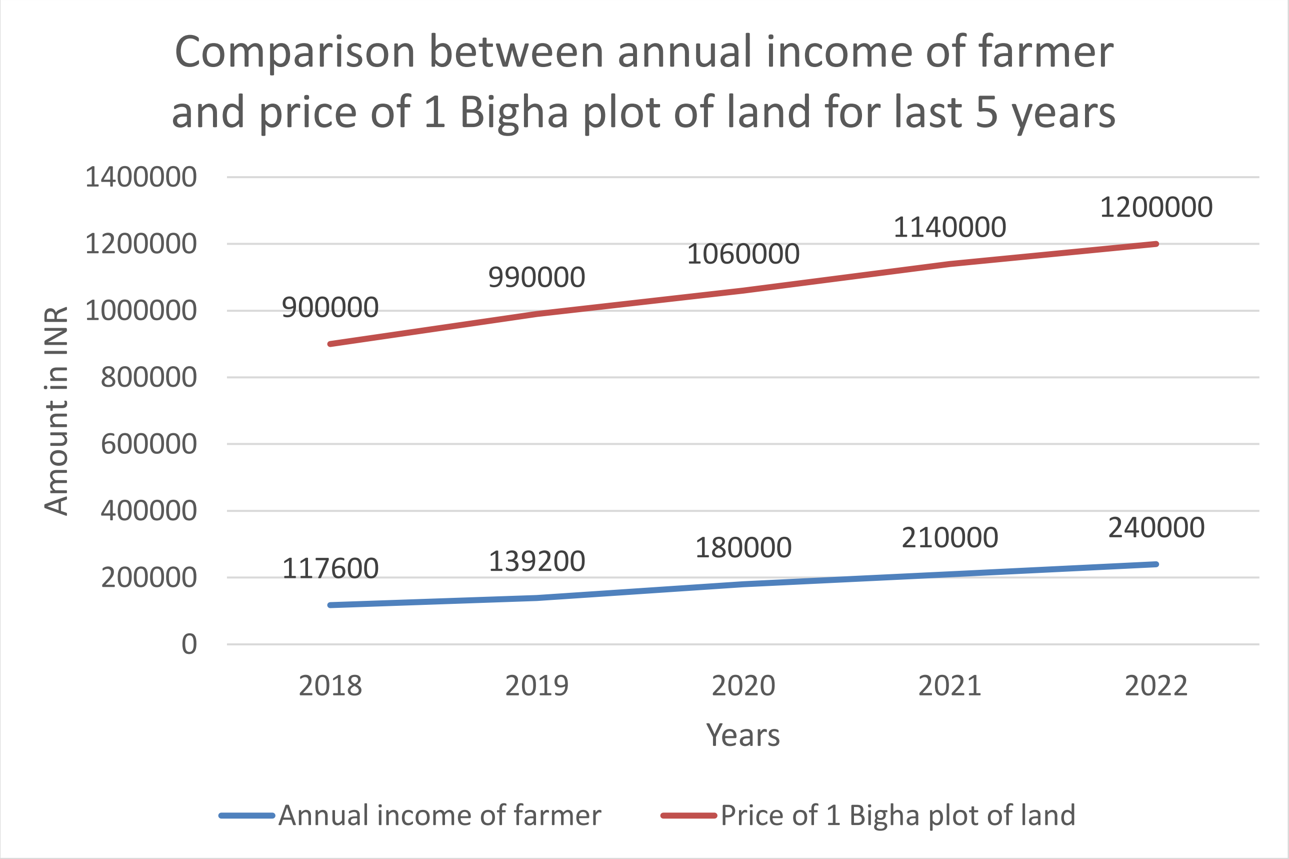
PROJECT OBJECTIVES

- To create an affordable kit which can analyze soil.
- It should analyze the atmospheric light, soil temperature and soil moisture.
- It should be able to monitor the parameters in real-time.
- It should enable a farmer to make a data-driven decision.
- Help the farmers to take correct decision based entirely on data instead of their instincts
- Help the country to get better crops due to the use of suitable land for agriculture
- Help maintaining transparency in business as it will prevent the farmers from getting cheated from cunning dealers
- Exports will increase thus GDP will rise

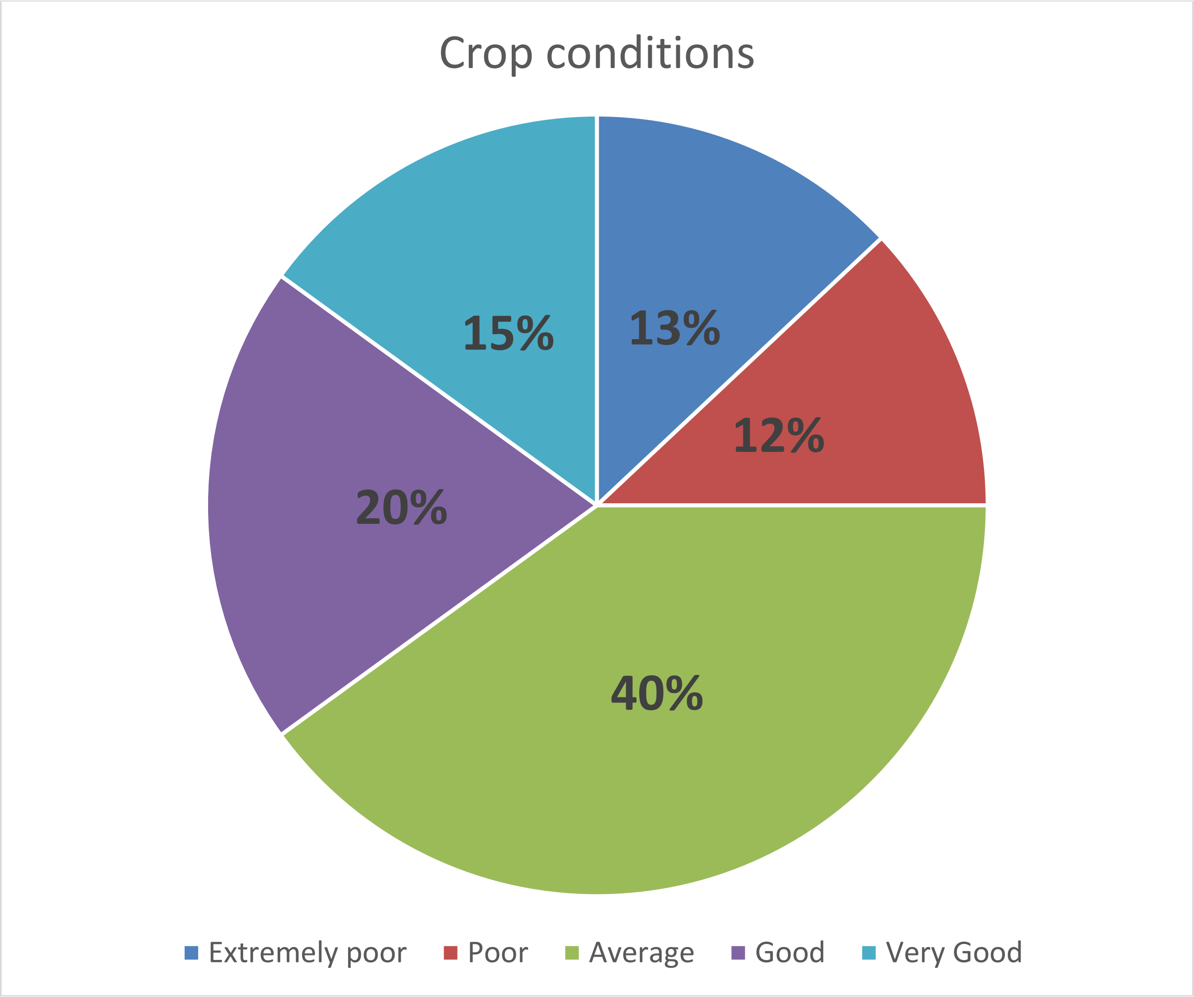
SURVEY OF FARMERS

1) Comparison between the annual income of farmers and price of land for the last 5 years
2) Crop conditions
3) Comparison between the selling price of healthy and unhealthy crops
4) Sector wise GDP in India (2021)

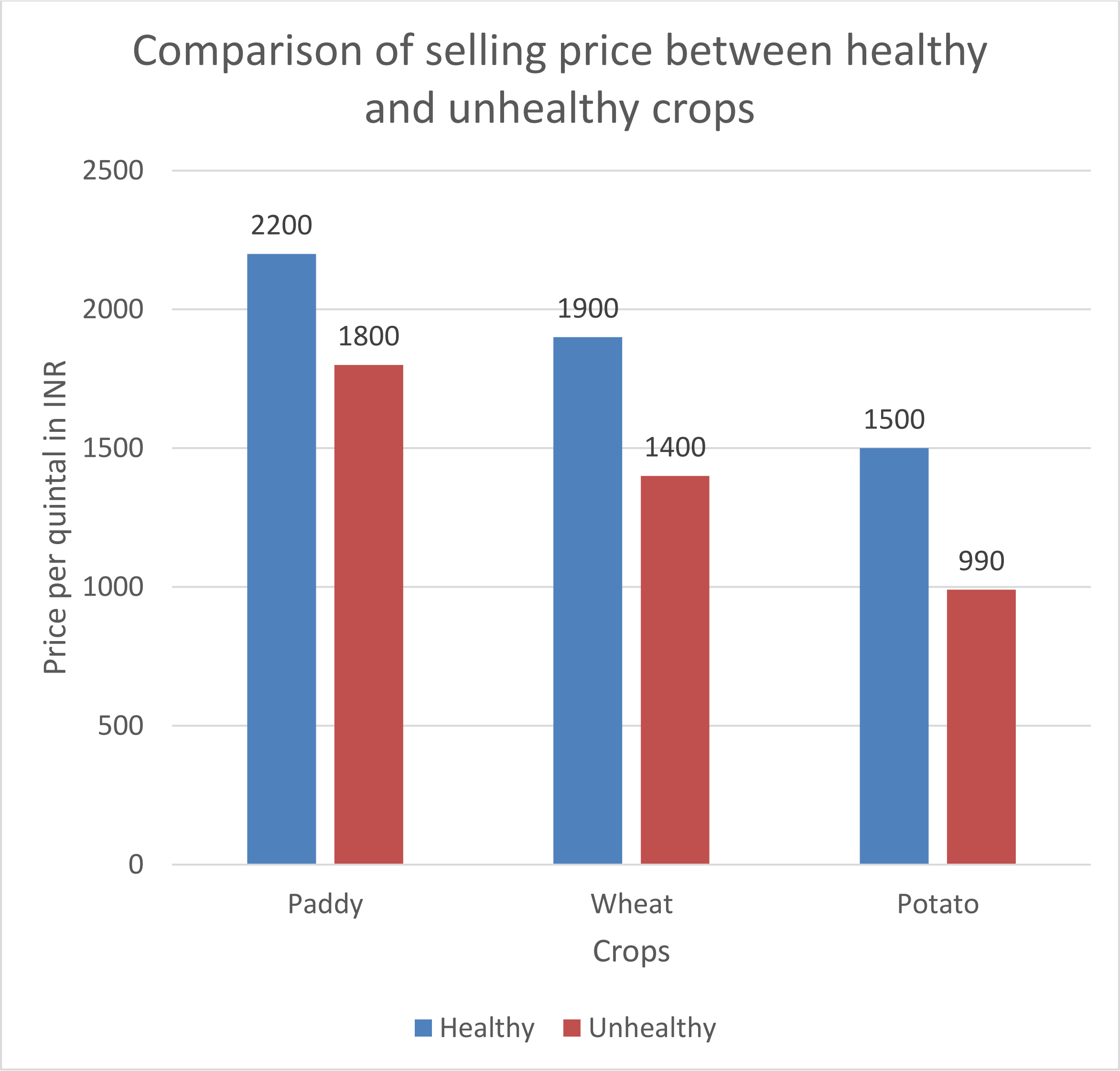
Sample population: 100
Location: Taki (India-Bangladesh border), West Bengal, India
Conducted by: Subhodeep Sarkar



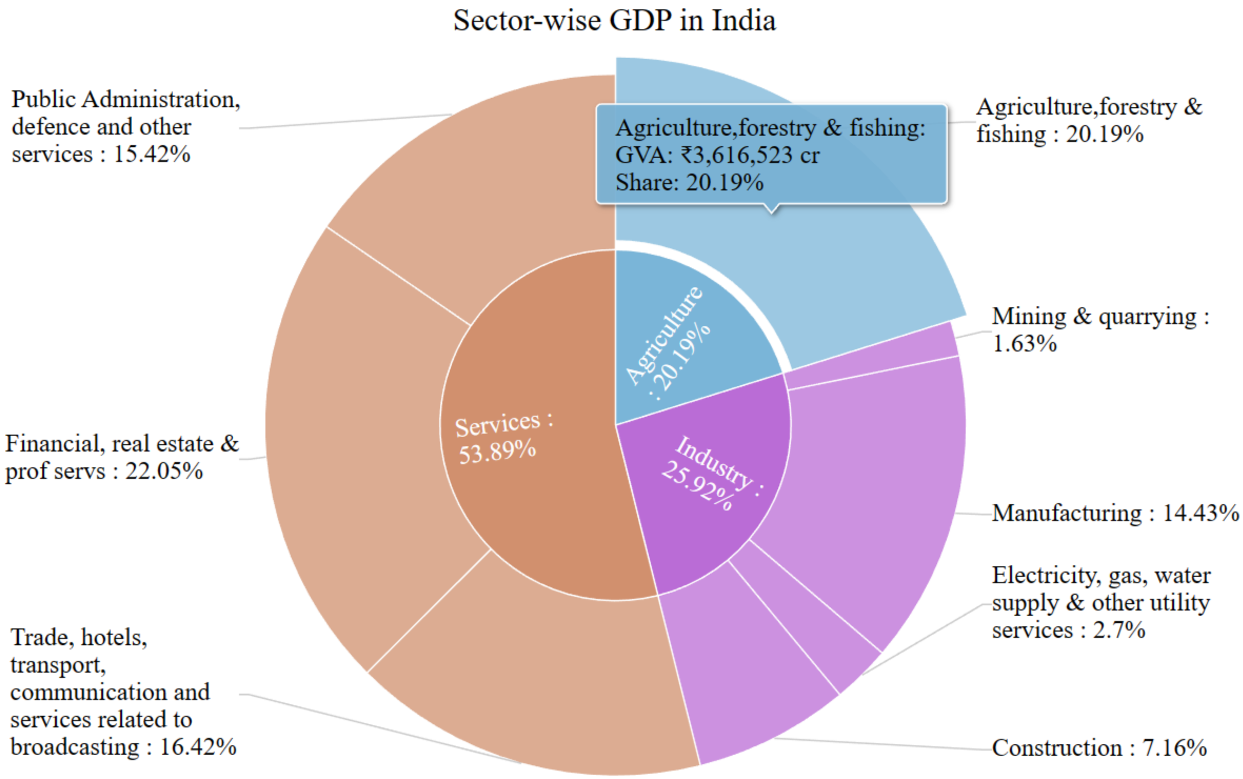
(Chart 1: Comparison between annual income of farmer and price of 1 Bigha plot of land for last 5 years)



(Chart 2: Crop conditions as reported by the farmers)



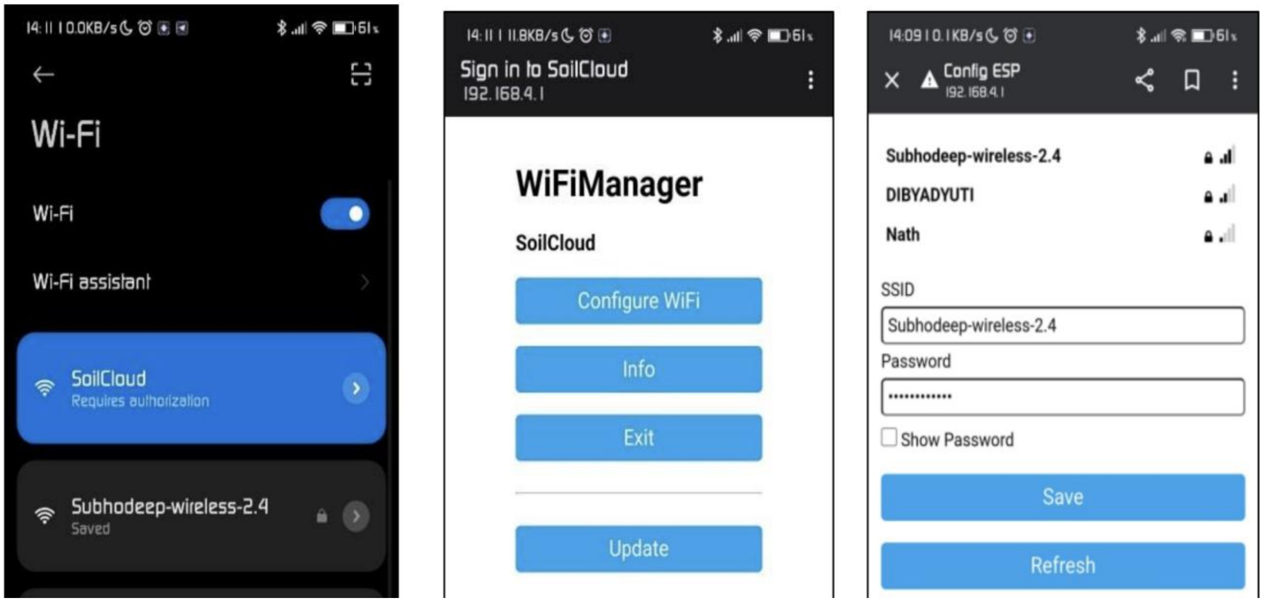
(Chart 3: Comparison of selling price between healthy and unhealthy crops)



(Chart 4: Sector wise GDP in India [Source: Statistics Time {2021}])

PROJECT METHODOLOGY

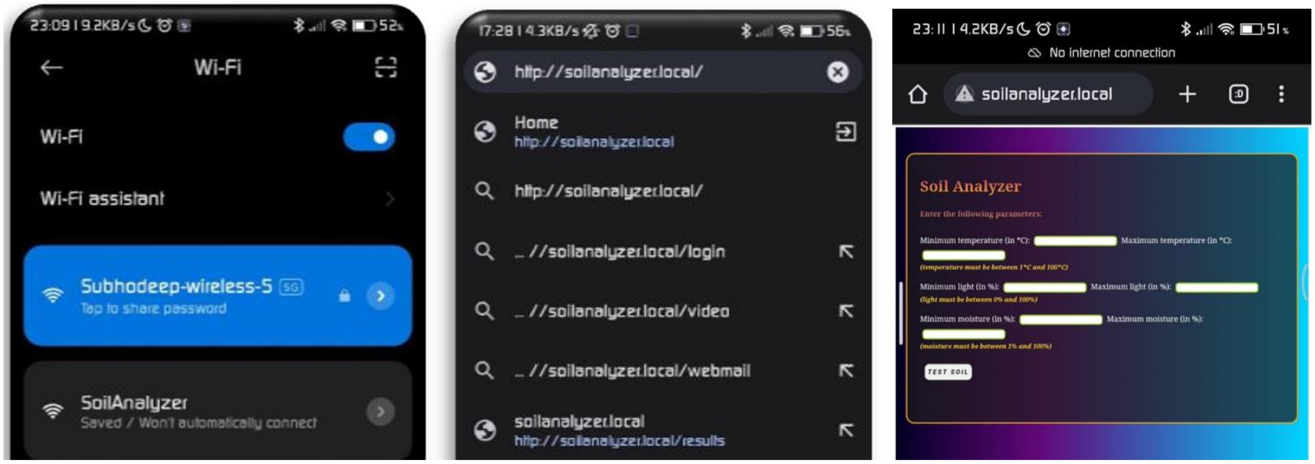
The WiFi mode of ESP-32 is set to AP_STP to allow the module to be in access point as well as station mode simultaneously.
SoilCloud AP is started where a smart device is connected to access the configuration portal to connect the ESP-32 to an internet-enabled network. It then scans nearby available WiFi networks and displays them with their SSID and Signal strength and the user has to select one of them and enter the password to connect to that network [7].



SSID
Subhodeep-wireless-2.4
Password
••••••••••••

(Figure 1: Process of connecting to internet)

Once the ESP-32 is connected to an internet-enabled network it closes SoilCloud AP. Now it starts SoilAnalyzer AP where a user connects a smart device to access the input form via browser. IP is not required as the local mDNS server can convert the domain name to IP. The domain name is http://soilanalyzer.local/


23:09 | 9.2KB/s
52
Wi-Fi
Wi-Fi
Wi-Fi assistant
Subhodeep-wireless-5 5G
Tap to share password
SoilAnalyzer
Saved / Won't automatically connect
17:28 | 4.3KB/s
56
http://soilanalyzer.local/
Home
http://soilanalyzer.local
http://soilanalyzer.local/
…//soilanalyzer.local/login
…//soilanalyzer.local/video
…//soilanalyzer.local/webmail
soilanalyzer.local
http://soilanalyzer.local/results
23:11 | 4.2KB/s
51
No internet connection
soilanalyzer.local

Soil Analyzer

Enter the following parameters:

Minimum temperature (in °C): Maximum temperature (in °C):

(temperature must be between 1°C and 100°C)

Minimum light (in %): Maximum light (in %):

(light must be between 0% and 100%)

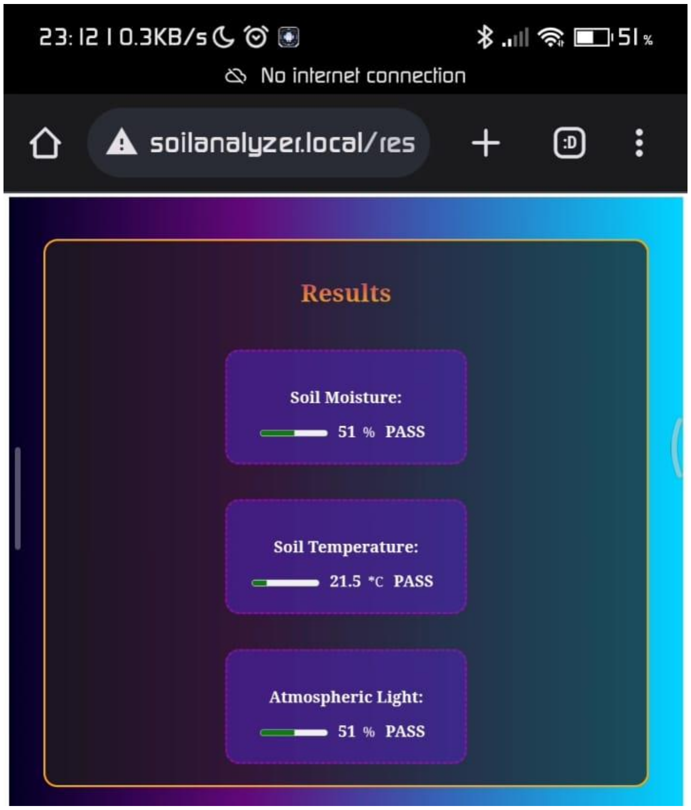
Minimum moisture (in %): Maximum moisture (in %):

(moisture must be between 1% and 100%)

TEST SOIL

(Figure 2: Navigating to home page and entering parameters)

Once the user enters all the desired parameters, data validation is done and upon success, it takes the user to another page (URL:http://soilanalyzer.local/results)



Results

Soil Moisture:

51 % PASS

Soil Temperature:

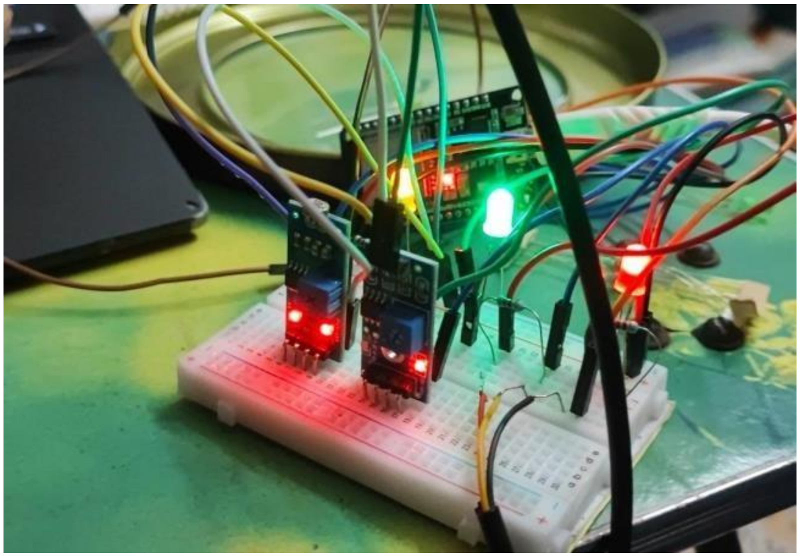
21.5 °C PASS

Atmospheric Light:

51 % PASS

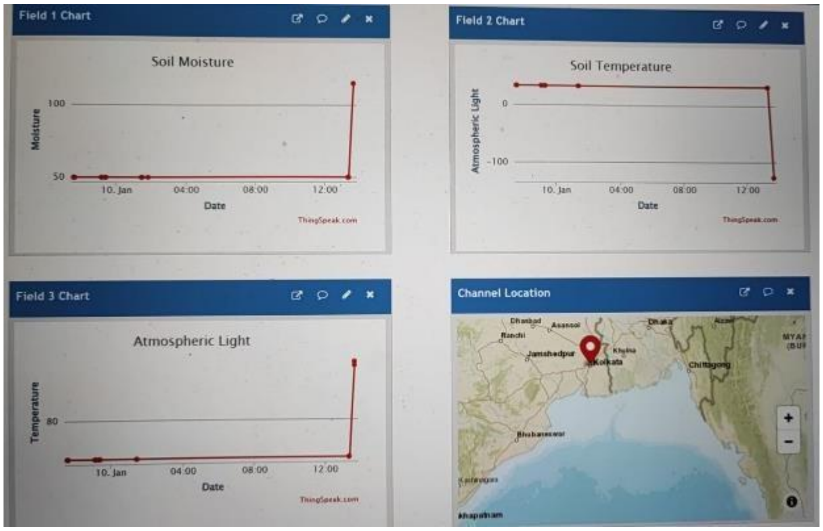
(Figure 3: Monitoring live results)

This is where the real-time data is matched against the user input and it shows whether the requirement has passed or not. If a particular parameter is passed the LED associated with it turns on.



(Figure 4: All LEDs glow as all the parameters passed)

This data is updated after every 17 seconds using WebSockets. The communication between frontend and server takes place using JSON via WebSockets [1] and the same data is uploaded to the ThingSpeak server [2] where it is plotted in graphs this data is further analyzed using MATLAB analysis tool built inside ThingSpeak [3].



(Figure 5: ThingSpeak Cloud monitoring)

Components used

- LM393 Light sensor
- DS18B20 temperature sensor
- FC28 moisture sensor
- ESP32 DEV DOIT v1 microcontroller
- 03962A/TP4056 battery charging module
- 5V Rechargeable Battery
- Resistors (220 Ω & 4.7 KΩ)
- LEDs

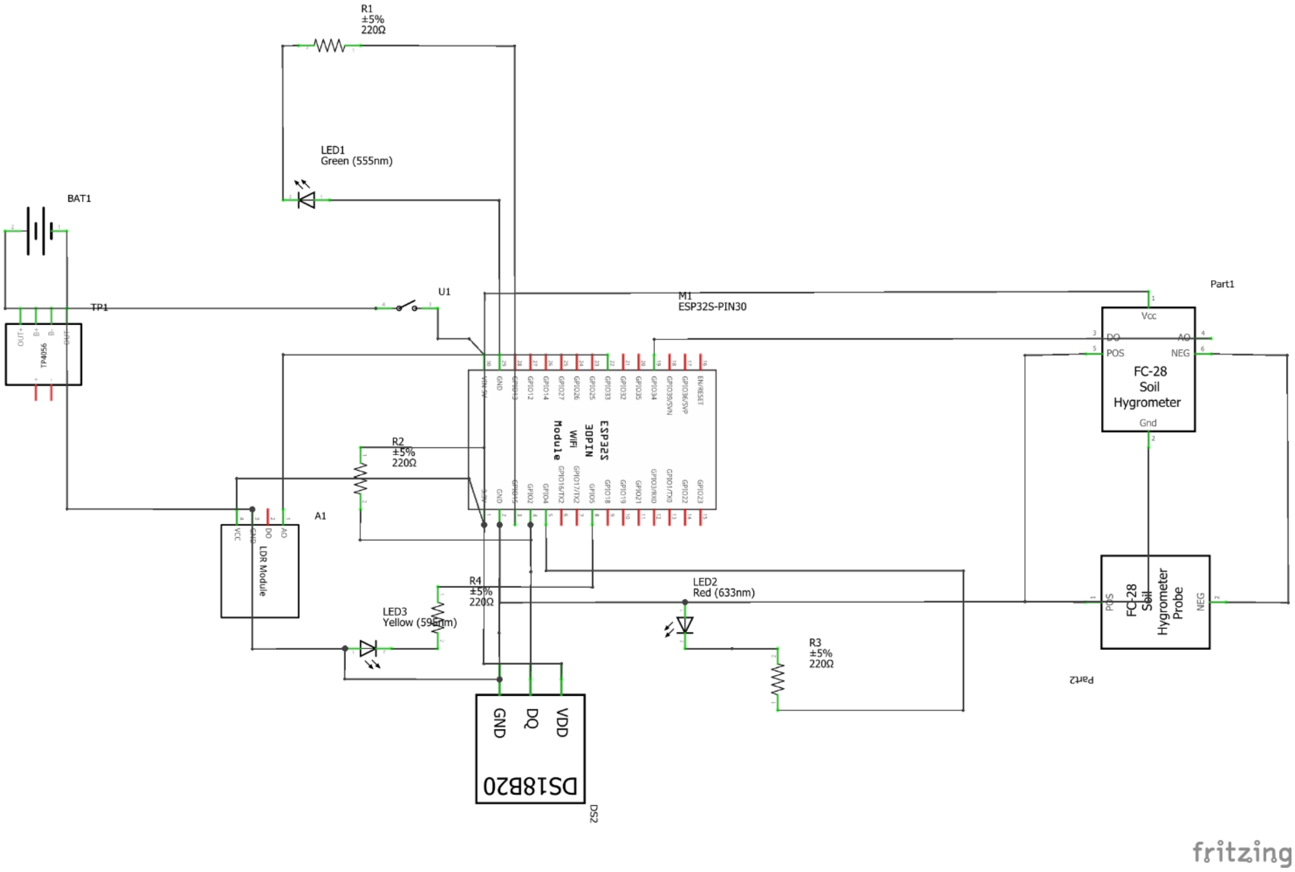
Technologies used

- WLAN Wi-Fi Hotspot
- mDNS
- HTTPv2.0 Websockets
- Asynchronous webserver
- HTML
- CSS
- JavaScript
- JSON
- C++
- ThingSpeak cloud

Tools used

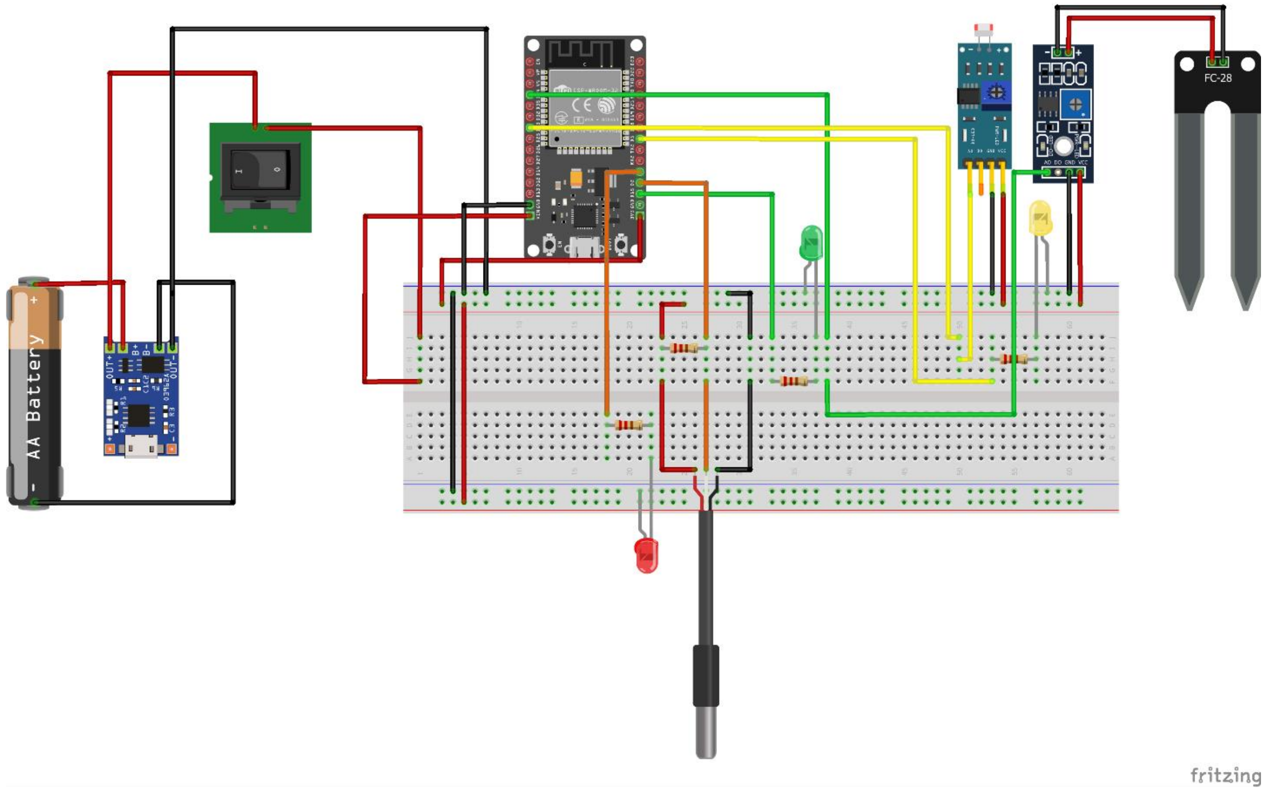
- Arduino IDE
- Google Chrome
- Fritzing

Circuit Diagram



(Figure 6: Schematic Diagram)

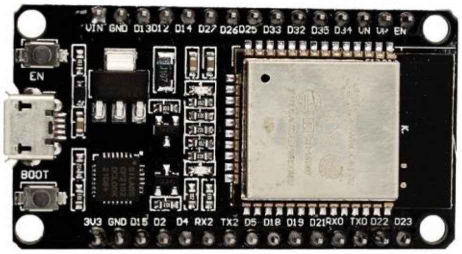
Breadboard diagram



(Figure 7: Breadboard Diagram)

Components description and how they communicate

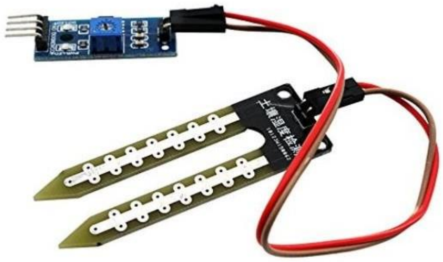
1. ESP-32


VIN GND D13 D12 D14 D27 D26 D25 D33 D32 D35 D34 VN VP EN
EN
BOOT
3V3 GND D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23

(Figure 7: ESP-32)

ESP-32 uses Xtensa® Dual-Core 32-bit LX6 microprocessor and has a clock speed ranging from 160Mhz to 240Mhz. 4 MB of flash memory is built inside it. It also has a WiFi module and a Bluetooth module built into it.

2. FC-28 Soil Moisture Sensor



(Figure 8: FC-28 Moisture Sensor)

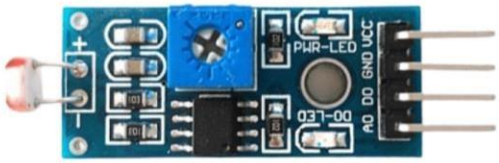
FC-28 sensor comes with two items, two-legged probes and an LM393 comparator. The probes are dug into the soil and it sends the quantity of water to the LM393 comparator which converts it to an analog voltage.

3. DS18B20 Temperature Sensor

(Figure 9: DS18B20 Temperature sensor)

DS18B20 is a waterproof temperature sensor which can be dug into the soil and it senses the temperature and converts it into voltage. It can measure temperature from -55°C to +125°C.

3. LM393 light sensor



(Figure 10: LM393 light sensor)

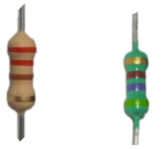
LM393 is a light sensor which can sense light via the light dependent resistor attached to it and then convert it to both analog as well as digital signal. The sensitivity of the sensor can be changed by rotating the potentiometer.

4. LEDs

(Figure 11: Red, Green and Yellow LEDs)

- The red LED will glow if the soil temperature meets the user requirements.
- The green LED will glow if the soil moisture meets the user requirements.
- The yellow LED will glow if the soil moisture meets the user requirements.

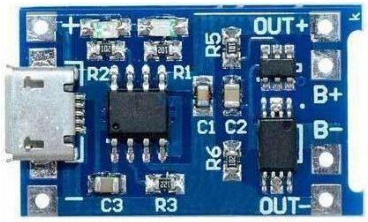
4. Resistors



(Figure 12: 220 Ω and 4.7 KΩ resistors)

220 Ω resistors are used for connecting the LEDs and 4.7 KΩ resistor is used to connect DS18B20temperature sensor.

5. TP-4056 Battery Charging Module



(Figure 13: TP-4056 battery charging module)

TP-4056 is a power supply as well as charging module. It takes power via battery and powers on theESP-32 using a switch. The connected battery must be rechargeable and it can be charged using USB. [6]

6. Rechargeable 5V AA Battery

(Figure 14: AA Battery)

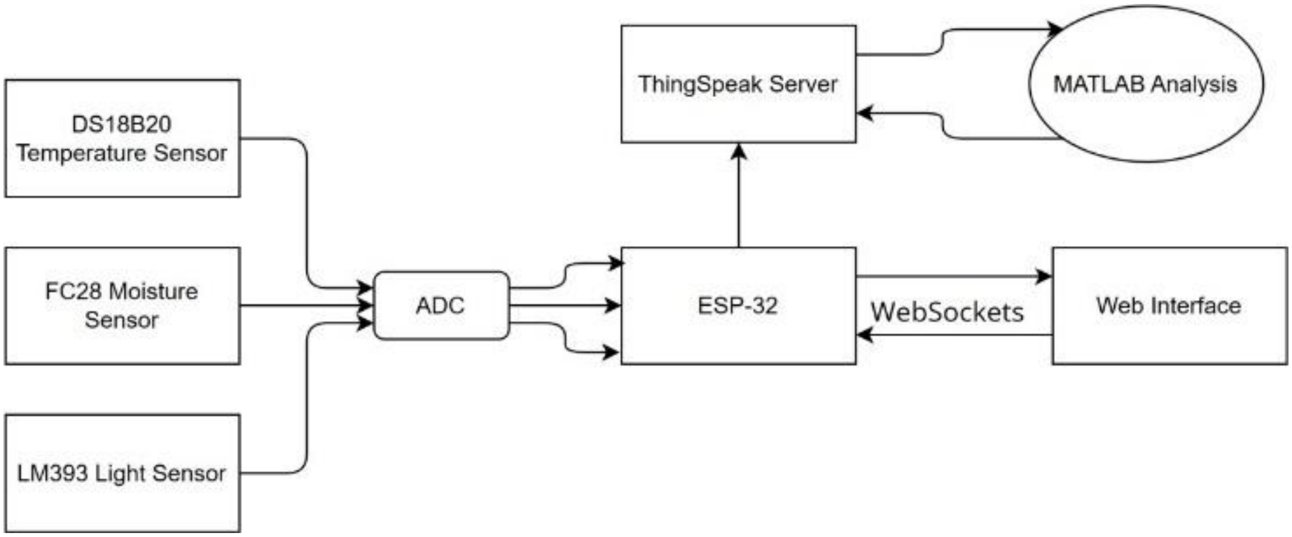
This battery actually powers the ESP-32 via TP-4056 module and can be charged via the same. [6]

7. SPST Switch

(Figure 15: SPST Switch)

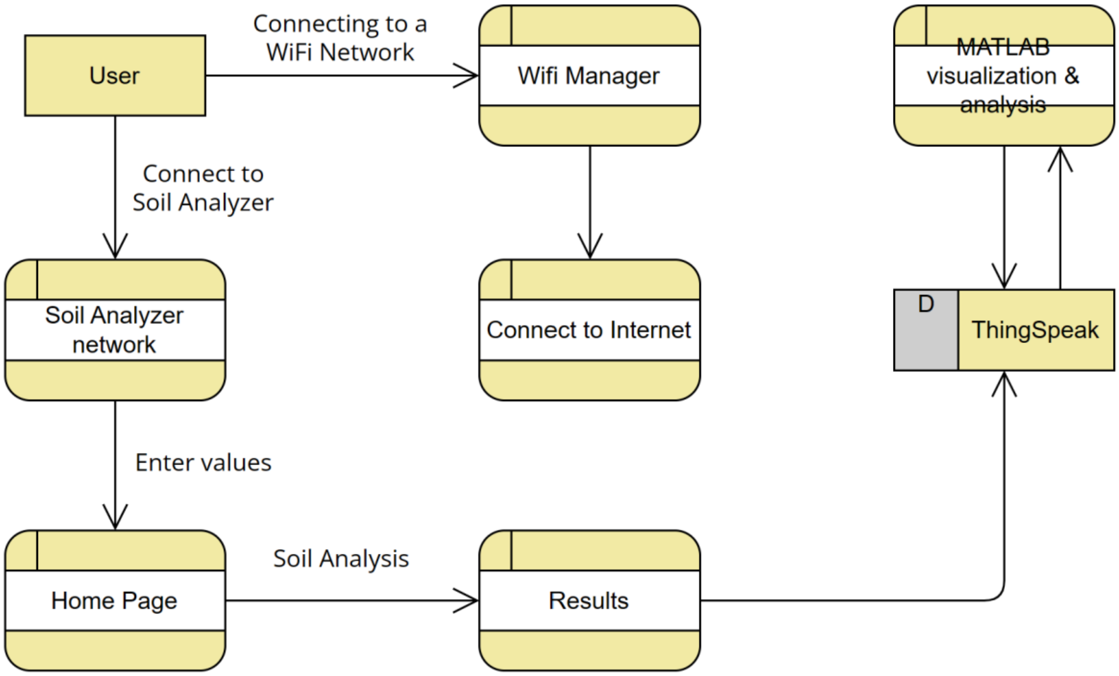
This switch is used for turning on or off the ESP-32.

Conceptual Model



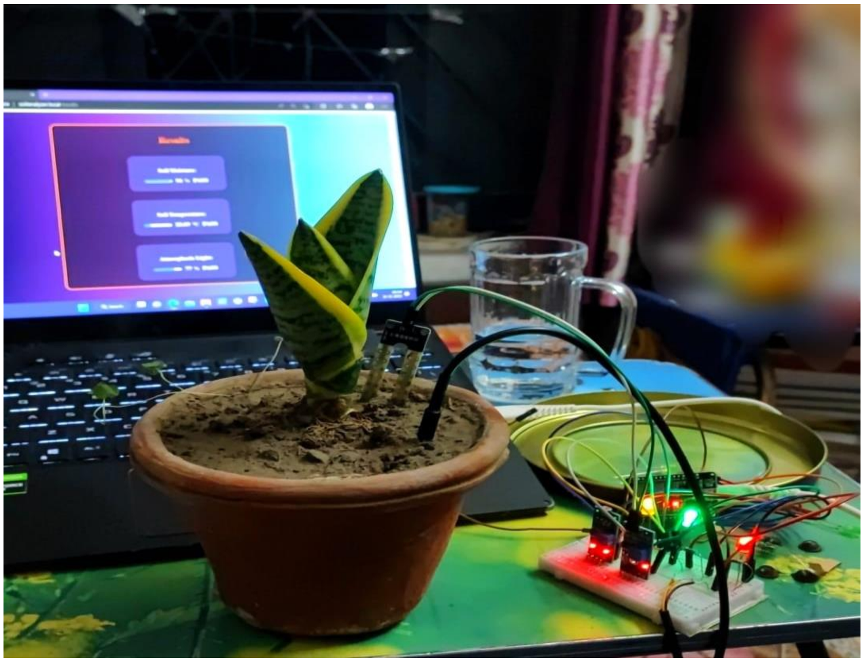
(Figure 16 : Conceptual model of the system)

Data Flow Diagram



Results

Figure 18 shows the device after assembling all the components on a breadboard. It can be seen that the parameters of the soil are displayed on the monitor of the laptop.



(Figure 18: The complete project on breadboard)

Four soil samples have been collected from different regions of West Bengal, India and the parameters of the soil have been analyzed.

Location	Temperature (°C)	Moisture (%)	Atmospheric light (%)
Taki, North 24 Parganas, West Bengal, India	15	86	75
Susunia forest, Bankura, West Bengal, India	23	49	84
Kanaipur, Malda, West Bengal, India	26	58	81
Darjeeling Tea Garden, Darjeeling, West Bengal, India	10	42	57

(Table 2: Soil Analysis sample)

Limitations

- Limited cloud features due to free version of ThingSpeak
- Does not use Zeroboard so not reliable
- Not water/dust/fire proof
- NPK sensor not installed
- pH sensor not installed

Conclusion

It is evident that the device can measure three basic parameters

- Atmospheric light
- Soil temperature
- Soil moisture

These three factors play the most important role in suitable agriculture. It can then match it against the user's requirements and check if the soil suits their need or not. This device will help the farmer to take the decision of purchasing a plot of land based on real-time data. This will safeguard them from huge losses and help prosper them. This will ensure that markets do not have a shortage of crops and the crops are healthy, thus eliminating inflation. In short, this will help the farmers, as well as the population of the country, prosper. This device can be further extended with additional sensors such as the NPK sensor and pH sensor [4] to further boost data-oriented decision-making with more data and accuracy.

References

[1] N. Mitrović, M. Đorđević, S. Veljković, D. Danković “Implementation of WebSockets in ESP32 based IoT Systems”; 2021 15th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS); E-ISBN: 978-1-6654-4442-2

[2] Parida, A. Behara, J.K. Naik, S. Pattanaik, R.S. Nanda, “Real-time environment monitoring system using ESP8266 and ThingSpeak on Internet of Things Platform”; 2019 International Conference on Intelligent Computing and Control Systems (ICCS); E-ISBN: 978-1-5386-8113-8

[3] Vijay, A.K. Saini, S. Banerjee, H. Nigam, “An IoT instrumented Smart Agricultural Monitoring and Irrigation System”; 2020 International Conference on Artificial Intelligence and Signal Processing (AISP); E-ISBN: 978-1-7281-4458-0

[4] E.R. Badakh, V.B. Malode, “Automatic Soil Testing System”; 2020 International Conference on Smart Innovations in Design, Environment, Management, Planning and Computing; E-ISBN: 978-1-7281-5970-6

Bibliography

- CCSNIAM KCG Final Report: https://bit.ly/3HAno2Z

- Power Your Projects With a Built-In Lithium Battery and a TP4056 Charger: https://bit.ly/3Dlqcid

- ESPAsyncWiFiManager Github: https://bit.ly/3j2Zu7b

Project Code

```
#include <ESPAsyncWebServer.h> //importing the Asynchronous server header
#include <ESPmDNS.h>
#include <WebSocketsServer.h>
#include <ArduinoJson.h>
#include <Ticker.h>
#include <OneWire.h>
#include <DallasTemperature.h>
//#include <WiFiManager.h> //This is incompatible with AsyncTCP library
#include <ESPAsyncWiFiManager.h> //This the alternative for WiFimanager
#include <ThingSpeak.h>
#define MOISTURE_SENSOR 34
#define MOISTURE_LIGHT 15
#define TEMP_SENSOR 2
#define TEMP_LIGHT 4
#define PHOTO_SENSOR 33
#define PHOTO_LIGHT 5
#define CHANNEL_ID 2002484
#define CHANNEL_WRITE_API_KEY "HB1XUG2HIXVI9HCR"

const int light_low = 4096;
const int light_high = 0;
const int mois_low = 4095;
const int mois_high = 550;
int timeout = 180;
char thingspeakserver[] = "www.thingspeak.com";

WiFiClient client;
AsyncWebServer server(80); //server listening at port 80 i.e HTTP port
WebSocketsServer websockets(81); //web sockets server listening at port 81
Ticker timer;
OneWire onewire(TEMP_SENSOR);
DallasTemperature tempsensor(&onewire);
DNSServer dns;

void sendSensorVal(); //this function will be called after an interval of
17 second in loop [declared below]
void ondemandap();

int mintemp,maxtemp,minlight,maxlight,minmois,maxmois;
String moistest,lighttest,temptest;
//defining function for invalid requests
void notFound(AsyncWebServerRequest *request){
  request->send(404,"text/html","Not found");
}
```

```
void webSocketEvent(uint8_t num, WStype_t type, uint8_t *payload, size_t
length){
  //num -> client number | type -> websocket type | payload -> data from
client | length -> length of payload
  switch(type){
    case WStype_DISCONNECTED: //if the client gets disconnected
    Serial.printf("[%u] Disconnected!\n",num); //print client number
  break;
  case WStype_CONNECTED: { //When a new client gets connected
    IPAddress ip = websockets.remoteIP(num); //get the IP address of the
client
    Serial.printf("[%u] Connected from %d.%d.%d.%d url:
%s\n",num,ip[0],ip[1],ip[2],ip[3],payload);
    //print connected client information
    //then send a message from server to that client
    websockets.sendTXT(num,"Connected");
  }
  break;
  case WStype_TEXT: //When the client sends data
    Serial.printf("[%u] get Text: %s\n",num,payload);
    String message = String((char*)(payload));
    Serial.println(message);

    DynamicJsonDocument doc(200);
    DeserializationError error = deserializeJson(doc,message);
    //Deserialize the JSON data into doc object
    //Store the error into error object (if any)
    if (error){ //if there is any error, print it
      Serial.print("Deserialization failed");
      Serial.println(error.c_str());
      return;
    }
    mintemp = doc["mintemp"];
    maxtemp = doc["maxtemp"];
    minlight = doc["minlight"];
    maxlight = doc["maxlight"];
    minmois = doc["minmois"];
    maxmois = doc["maxmois"];
  }
}

void setup() {
  Serial.begin(115200); //baud rate
  WiFi.mode(WIFI_AP_STA);
  // WiFi.softAP("SoilAnalyzer",""); //hotspot with SSID and password is
empty
  // Serial.println("IP: ");
  // Serial.println(WiFi.softAPIP()); //IP of the microcontroller will be
printed on serial monitor
  ondemandap();
  server.onNotFound(notFound); //calls the notFound() function upon
requesting invalid page
  pinMode(MOISTURE_LIGHT, OUTPUT);
  pinMode(TEMP_LIGHT,OUTPUT);
  pinMode(PHOTO_LIGHT,OUTPUT);
  tempsensor.begin();


  //route
  //home page
  server.on("/",[](AsyncWebServerRequest *request){
    //this webpage will be sent upon receiving the request at root
    //R"=====()=====" is raw string
    //location.hostname -> hostname of the server
    //window.location -> location redirection
    //PROGMEM -> saves the array into flash memory
    //var connection establishes the connection for websocket from client
to serveer at port 81
    //connection.send() sends message from client to server
    char webpage[] PROGMEM = R"=====(
    <!DOCTYPE html>
<html>
    <head>
        <title>Home</title>
        <style>
            input{
                margin: 5px 5px 5px 5px;
                display: inline;
            }

            button{
                height: 50px;
                font-style: italic;
                font-size: 20px;
                box-shadow: inset 0 0 0 0 purple;
                margin: 10px 10px 10px 10px;
                height: 40px;
                border-radius: 12px;
                transition: ease-out 0.5s;
                font-size: 15px;
                text-transform: uppercase;
                letter-spacing: 3px;
                font-weight: bolder;
                color:black
            }

            button:hover{
                  box-shadow: inset 300px 0 0 0 skyblue;
            }
            h1{
                padding-bottom: 10px;
                background: -webkit-linear-gradient(#cb356b, #ffd608);
                -webkit-background-clip: text;
                -webkit-text-fill-color: transparent;
            }

            h4{
                padding-bottom: 10px;
                background: -webkit-linear-gradient(#b24592, #ffd608);
                -webkit-background-clip: text;
                -webkit-text-fill-color: transparent;
            }
            #container{
                background-color: rgba(32, 29, 29, 0.7);
                color: white;
                padding: 30px 30px 30px 30px;
                text-align: left;
                border-radius: 20px;
                border: 3px solid orange;
                font-size:110%;
                display: block;
                align-items: center;
                justify-content: center;
                width: 850px;
                top: 150px;
                bottom:100px;
                left: 0;
                right: 0;
                margin: auto;
                margin-top: 50px;
            }

            #bg{
                background: rgb(2,0,36);
                background: linear-gradient(90deg, rgba(2,0,36,1) 0%,
rgba(98,9,121,1) 35%, rgba(0,212,255,1) 100%);
                background-size: cover;
                height: 95vh;
                background-position:absolute;
                padding: 10px 10px 10px 10px;
            }

            input{
                border-radius: 8px;
                border:3px solid yellowgreen
            }

            .desc{
                font-style: italic;
                color:#ffd608;
                font-weight: 600;
            }
        </style>
    </head>
    <body>
        <div id="bg">
            <div id="container">
                <h1>Soil Analyzer</h1>
                <h4>Enter the following parameters:</h4>
                <form action="#" id="form">
                    <div id="temperature">
                        Minimum temperature (in *C): <input type="text"
name="mintemp">
                        Maximum temperature (in *C): <input type="text"
name="maxtemp">
                        <br>
                        <small class="desc">(temperature must be between
1*C and 99*C)</small>
                    </div>
                    <br>
                    <div id="light">
                        Minimum light (in %): <input type="text"
name="minlight">
                        Maximum light (in %): <input type="text"
name="maxlight">
                        <br>
                        <small class="desc">(light must be between 0% and
100%)</small>
                    </div>
                    <br>
                    <div id="moisture">
                        Minimum moisture (in %): <input type="text"
name="minmois">
                        Maximum moisture (in %): <input type="text"
name="maxmois">
                        <br>
                        <small class="desc">(moisture must be between 1%
and 100%)</small>
                    </div>
                    <br>
                </form>
                <button id="submit" onclick="captureSend()">Test
Soil</button>
                <h1 id="error"></h1>
            </div>
        </div>
        <script>
        var connection = new WebSocket('ws://'+location.hostname+':81/');
        function captureSend(){
            var formdata = document.getElementById('form');
            if(formdata["mintemp"].value>0 && formdata["maxtemp"].value<100
&& formdata["minmois"].value > 0 && formdata["maxmois"].value <=100 &&
formdata["minlight"].value >=0 && formdata["maxlight"].value <=100){
              var jsondata =
'{"mintemp":'+formdata["mintemp"].value+',"maxtemp":'+formdata["maxtemp"].v
alue+',"minlight":'+formdata["minlight"].value+',"maxlight":'+formdata["max
light"].value+',"minmois":'+formdata["minmois"].value+',"maxmois":'+formdat
a["maxmois"].value+'}';
              formdata["mintemp"].value = "";
              formdata["maxtemp"].value = "";
              formdata["minlight"].value = "";
              formdata["maxlight"].value= "";
              formdata["minmois"].value = "";
              formdata["maxmois"].value = "";
              connection.send(jsondata);
              window.location = "http://"+location.hostname+"/results";
            }
            else{
              var errormsg = document.getElementById("error");
              errormsg.innerText = "Invalid parameters";
            }
        }
        </script>
    </body>
</html>
  )=====";
  //send_P -> sends the webpage saved in flash memory
    request->send_P(200,"text/html",webpage);
  });

  //results page
  server.on("/results",[](AsyncWebServerRequest *request){
      char resultpage[] PROGMEM = R"=====(
        <!DOCTYPE html>
<html>
    <head>
        <title>Results</title>
        <style>
            #moisval,#moistest,#tempval,#temptest,#lightval,#lighttest{
                display: inline;
                margin: 10px;
            }

            #bg{
                background: rgb(2,0,36);
                background: linear-gradient(90deg, rgba(2,0,36,1) 0%,
rgba(98,9,121,1) 35%, rgba(0,212,255,1) 100%);
                background-size: cover;
                height: 150vh;
                background-position:absolute;
                padding: 10px 10px 10px 10px;
            }

            #container{
                width: 750px;
                background-color: rgba(32, 29, 29, 0.7);
                color: white;
                padding: 30px 30px 30px 30px;
                text-align: center;
                border-radius: 20px;
                border: 3px solid orange;
                font-size:110%;
                display: block;
                align-items: center;
                justify-content: center;
                width: 800px;
                top: 150px;
                bottom:100px;
                left: 0;
                right: 0;
                margin: auto;
                margin-top: 50px;
            }

            #container div{
                width: 280px;
                background-color: rgba(78, 35, 180, 0.575);
                color: white;
                padding: 30px 30px 30px 30px;
                text-align: center;
                border-radius: 20px;
                border: 3px dashed rgba(255, 0, 200, 0.384);
                font-size:110%;
                display: block;
                align-items: center;
                justify-content: center;
                top: 150px;
                bottom:100px;
                left: 0;
                right: 0;
                margin: auto;
                margin-top: 50px;
            }

            h1{
                padding-bottom: 10px;
                background: -webkit-linear-gradient(#cb356b, #ffd608);
                -webkit-background-clip: text;
                -webkit-text-fill-color: transparent;
```

```
            }
        </style>
    </head>
    <body>
        <div id="bg">
            <div id="container">
                <h1>Results</h1>
                <div id="mois">
                  <h3>Soil Moisture: </h3>
                  <meter id="moismtr" value="100" min="0"
max="100"></meter>
                  <h3 id="moisval">100</h3>%
                  <h3 id="moistest">FAIL</h3>
                </div>
                <div id="temp">
                    <h3>Soil Temperature: </h3>
                    <meter id="tempmtr" value="100" min="0"
max="100"></meter>
                    <h3 id="tempval">100</h3>*C
                    <h3 id="temptest">FAIL</h3>
                </div>
                <div id="light">
                    <h3>Atmospheric Light: </h3>
                    <meter id="lightmtr" value="100" min="0"
max="100"></meter>
                    <h3 id="lightval">100</h3>%
                    <h3 id="lighttest">FAIL</h3>
                </div>
            </div>
        </div>
        <script>
            var connection = new
WebSocket('ws://'+location.hostname+':81/');
            var moisdata = 0;
            var moistest = "";

            var tempdata = 0;
            var temptest = "";

            var lightdata = 0;
            var lighttest = "";

            connection.onmessage = function(event){
                var fulldata = event.data;
                console.log(fulldata);
                var data = JSON.parse(fulldata);

                moisdata = data.moisdata;
                moistest = data.moistest;
                tempdata = data.tempdata;
                temptest = data.temptest;

                lightdata = data.lightdata;
                lighttest = data.lighttest;

                document.getElementById("moismtr").value = moisdata;
                document.getElementById("moisval").innerHTML = moisdata;
                document.getElementById("moistest").innerHTML = moistest;

                document.getElementById("tempmtr").value = tempdata;
                document.getElementById("tempval").innerHTML = tempdata;
                document.getElementById("temptest").innerHTML = temptest;

                document.getElementById("lightmtr").value = lightdata;
                document.getElementById("lightval").innerHTML = lightdata;
                document.getElementById("lighttest").innerHTML = lighttest;
            }
        </script>
    </body>
</html>
      )=====";

    request->send_P(200,"text/html",resultpage);
  });

  websockets.onEvent(webSocketEvent); //call the webSocketEvent() function
on any websocket event
  timer.attach(17,sendSensorVal); //attach timer with sendSensorVal with
interval of 17 [this function will be called after every 17 second]
  //start servers
  server.begin(); //start the web server
  MDNS.begin("soilanalyzer"); //set up local domain name server with
hostname -> soilanalyzer.local
  websockets.begin(); //start the websockets server
  WiFi.softAP("SoilAnalyzer",""); //hotspot with SSID and password is empty
  Serial.println("IP: ");
  Serial.println(WiFi.softAPIP()); //IP of the microcontroller will be
printed on serial monitor
  ThingSpeak.begin(client);  // Initialize ThingSpeak
}

void loop() {
  // put your main code here, to run repeatedly:
  websockets.loop(); //continuously monitor

}

//function declaration of sendSensorVal
void sendSensorVal(){
```

```
int moisdata_raw = analogRead(MOISTURE_SENSOR);
int moisdata = map(moisdata_raw,mois_low,mois_high,0,100);
if (moisdata>=minmois && moisdata<=maxmois && moisdata!=0){
  moistest = "PASS";
  digitalWrite(MOISTURE_LIGHT,HIGH);
}
else{
  moistest = "FAIL";
  digitalWrite(MOISTURE_LIGHT,LOW);
}

tempsensor.requestTemperatures();
float tempdata  = tempsensor.getTempCByIndex(0);
if(tempdata>=mintemp && tempdata<=maxtemp){
  temptest = "PASS";
  digitalWrite(TEMP_LIGHT,HIGH);
}
else{
  temptest = "FAIL";
  digitalWrite(TEMP_LIGHT,LOW);
}

int lightdata_raw = analogRead(PHOTO_SENSOR);
int lightdata = map(lightdata_raw,light_low,light_high,0,100);

if(lightdata>=minlight && lightdata<=maxlight){
  lighttest = "PASS";
  digitalWrite(PHOTO_LIGHT,HIGH);
}
else{
  lighttest = "FAIL";
  digitalWrite(PHOTO_LIGHT,LOW);
}

String JSON_data = "{\"moisdata\":";
        JSON_data+= moisdata;
        JSON_data+=",\"moistest\":";
        JSON_data+="\""+moistest+"\"";
        JSON_data+=",\"tempdata\":";
        JSON_data+= tempdata;
        JSON_data+=",\"temptest\":";
        JSON_data+="\""+temptest+"\"";
        JSON_data+=",\"lightdata\":";
        JSON_data+= lightdata;
        JSON_data+=",\"lighttest\":";
        JSON_data+="\""+lighttest+"\"";
        JSON_data+="}";

Serial.println(JSON_data);
```

```
  websockets.broadcastTXT(JSON_data); //send the JSON data to all clients
i.e broadcast
  if(client.connect(thingspeakserver,80)){
      //internet is available
      //digitalWrite(THINGSPEAKLED,HIGH);
      //Set the fields one by one
      ThingSpeak.setField(1,moisdata);
      ThingSpeak.setField(2,tempdata);
      ThingSpeak.setField(3,lightdata);
      //POST all the data at once (i.e in a single HTTP request)
      ThingSpeak.writeFields(CHANNEL_ID,CHANNEL_WRITE_API_KEY);
      //ThingSpeak allows new data point after 15 seconds minimum
      //3 million requests are free annually
    }
    else{
      //digitalWrite(THINGSPEAKLED,LOW);
      Serial.println("Could not connect to server");
      //ondemandap();
    }
}


void ondemandap(){
  AsyncWiFiManager wifiManager(&server,&dns);
  wifiManager.resetSettings();
  wifiManager.setTimeout(timeout);

  if (!wifiManager.startConfigPortal("SoilCloud")) {
      Serial.println("failed to connect and hit timeout");
      delay(3000);
      ESP.restart();
      delay(5000);
    }
    Serial.println("connected...yeey :)");
}
```

Appendix

Project Link: https://www.github.com/H4CK3RD33P/soilanalyzer (main branch)

Images/Diagrams link:
https://github.com/H4CK3RD33P/SoilAnalyzer/tree/main/resources (main branch)