Mini-Project Report

Project GROW- Self Sustaining Irrigation System

B.Tech.

By

Preity M(108118074) – ECE

Vaisnav A(108118105) - ECE

Vaibhav A(110118094) - ICE

S Rohit(111118103) - Mech

Department of Electronics and Communication Engineering

National Institute of Technology

Tiruchirappalli – 620 015

BONAFIDE CERTIFICATE

This is to certify that the mini-project titled ‘Project GROW-Predictive Self Sustaining Irrigation System’ is a bona fide record of the work done by

Preity M(108118074) – ECE
Vaisnav A(108118105) – ECE
Vaibhav A(110118094) – ICE
S Rohit (111118103) - Mech

for the elective course ECPE 34 Internet of Things/ ECOE 26 Internet of Things/ ICPE 41 IoT System Design offered in the Department of Electronics and Communication Engineering, National Institute of Technology, Tiruchirappalli – 620 015, during January 2021 Session.

(Dr. M. Bhaskar)
Professor, ECE

Faculty In-charge

ABSTRACT/PROBLEM STATEMENT

The World is currently facing a very big problem in the name of water scarcity.

Especially in India, where 60-70% economy depends on agriculture, there is a great need to modernize conventional agricultural practices for better productivity.

Through this project, we aim to solve the big problem of water scarcity in rural areas, by implementing a self-sustaining irrigation system using Sensors and IoT Technology.

Our project aims to maintain the optimal moisture level in the soil, hence curbing the problem of over watering plants which results in wastage of water as well as damage to the plant.

Hence by doing this project, we attempt to counter the problem of wastage of water(overwatering) in agriculture, by using weather forecasts as well as the current soil moisture level to water plants only when soil moisture level goes below a particular threshold.

OBJECTIVE OF THE PROJECT/INTRODUCTION

The main objective of this project is to provide an automated irrigation system to minimize the utilization of water in agriculture by combining technologies like Internet of Things (IoT), optimization tools and cloud computing.

The scarcity of clean water resources around the globe has generated a need for their optimum utilization.

Irrigation done using canal systems affects crop health and produces a poor yield because some crops are too sensitive to water content in soil.

Another major issue present is over-watering of plants, which leads to plant damage as well as wastage of water.

For effective and optimum utilization of fresh water in irrigation, it becomes essential to develop smart irrigation systems based on dynamic prediction of soil moisture pattern of the field and precipitation information of upcoming days.

Our aim is to design a modest, easy to install methodology IoT based device to monitor and indicate the soil moisture level and using weather prediction, that is regularly organized and automated.

IoT based smart irrigation systems can help in achieving optimum water-resource utilization in the precision farming landscape.

We use soil moisture and temperature sensors to monitor soil moisture and use weather forecasting to predict rains, to water plants only when needed and to not water plants the day before rains.

COMPONENTS REQUIRED AND SPECIFICATIONS

Components and their Specifications:

1. Board-Arduino Uno:

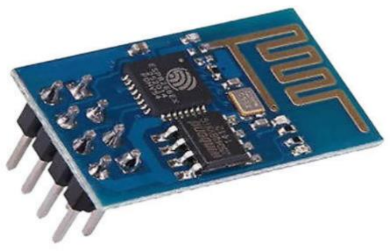
   Technical Specifications:

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

2. ESP8266 WiFi Module:

Technical Specifications:

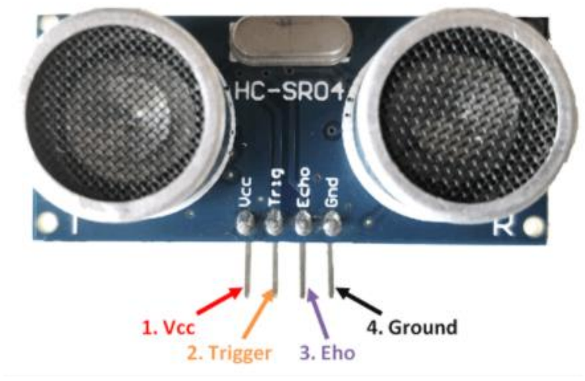
- 2.4 GHz Wi-Fi (802.11 b/g/n, supporting WPA/WPA2).
- General-purpose input/output (16 GPIO).
- Inter-Integrated Circuit (I^2C) serial communication protocol.
- Analog-to-digital conversion (10-bit ADC).
- Serial Peripheral Interface (SPI) serial communication protocol.
- I^2S (Inter-IC Sound) interfaces with DMA(Direct Memory Access) (sharing pins with GPIO).
- UART (on dedicated pins, plus a transmit-only UART can be enabled on GPIO2).
- Pulse-width modulation (PWM).



3. Ultrasonic Sensor (HC-Sr04):

   Module Pin Configuration:

   Technical Specifications:

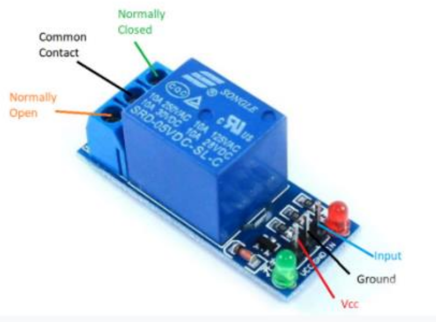


- Working Voltage: DC 5V
- Working Current: 15m
- Working frequency: 40 Hz
- Max Range:4m
- Min Range:2cm
- Measuring Angle:15 degree
- Trigger Input Signal: 10µS TTL pulse
- Echo Output Signal Input TTL lever signal and the range in proportion
- Dimension:45*20*15 mm

Pin Number	**Pin Name**	**Description**
1	Vcc	The Vcc pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

4. Relay Module:

   Module Pin configuration:



   Technical Specifications:

- Supply voltage – 3.75V to 6V
- Quiescent current: 2mA
- Current when the relay is active: ~70mA
- Relay maximum contact voltage – 250VAC or 30VDC
- Relay maximum current – 10A

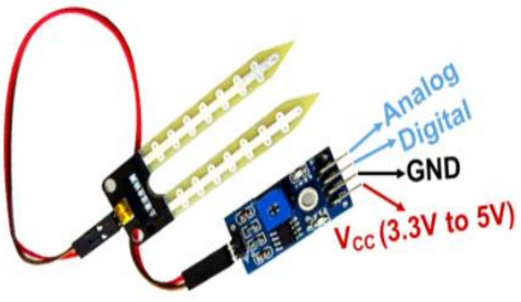
Pin Number	Pin Name	Description
1	Relay Trigger	Input to activate the relay
2	Ground	0V reference
3	VCC	Supply input for powering the relay coil
4	Normally Open	Normally open terminal of the relay
5	Common	Common terminal of the relay
6	Normally Closed	Normally closed contact of the relay

5. Moisture Sensor Unit:

Module Pin Configuration:

Technical Specifications:

- Operating Voltage: 3.3V to 5V DC
- Operating Current: 15mA
- Output Digital - 0V to 5V, Adjustable trigger level from preset
- Output Analog - 0V to 5V based on infrared radiation from fire flame falling on the sensor
- LEDs indicating output and power
- PCB Size: 3.2cm x 1.4cmLM393 based design



Pin Name	Description
VCC	The Vcc pin powers the module, typically with +5V
GND	Power Supply Ground
DO	Digital Out Pin for Digital Output.
AO	Analog Out Pin for Analog Output

6. Solenoid valve:

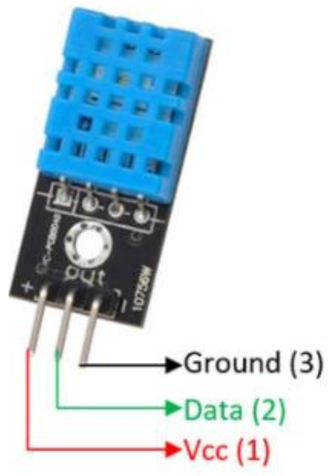
Technical Specifications:

- Rated Operating Voltage: 12V DC
- Rated Current: 0.6A
- Operation Mode: NC (Normally Closed)
- Pressure: 0.02 – 0.8MPa
- Max fluid temperature: 100°C

- Power Consumption: 8W
- Dimensions: Body: 84 x 57mm (L x H), Coil size: 34 x 23mm (W x H)
- Weight: 100gm
- Inlet and Outlet diameters: Hose barbs for ½’’ outer diameter hose.

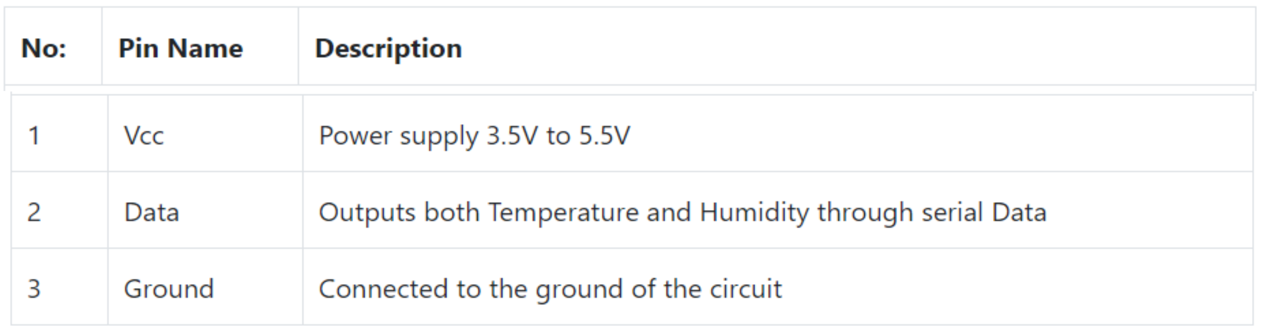
7. DHT 11 Temperature Sensor Module:

Module Pin Configuration:



Technical Specifications:

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%



No:	Pin Name	Description
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	Ground	Connected to the ground of the circuit

9. Jumper Wires.

10. Ultrasonic Sensor Bracket – For holding Ultrasonic Sensor.

BILL OF MATERIALS:

	COMPONENT	COST
Robu	Jumper wire set	75
		65
		60
Robu	Solenoid valve - 12V 0.6A	420
Robu	Ultrasonic sensor	75
Robu	Ultrasonic bracket	50
Robu	Arduino	400
Robu	Relay module	190
Ecomp	Soli moisture sensor	60
		125
Robu	DHT11 Temp	130
EComp	ESP 8266 Wifi module	180
EComp	Soil Moisture 3 way	325
EComp	12V 2A Adapter	150
Ecomp	LCD display	240
		155
	Miscellaneous(to buy)	
	Total	2700

SOFTWARE USED:

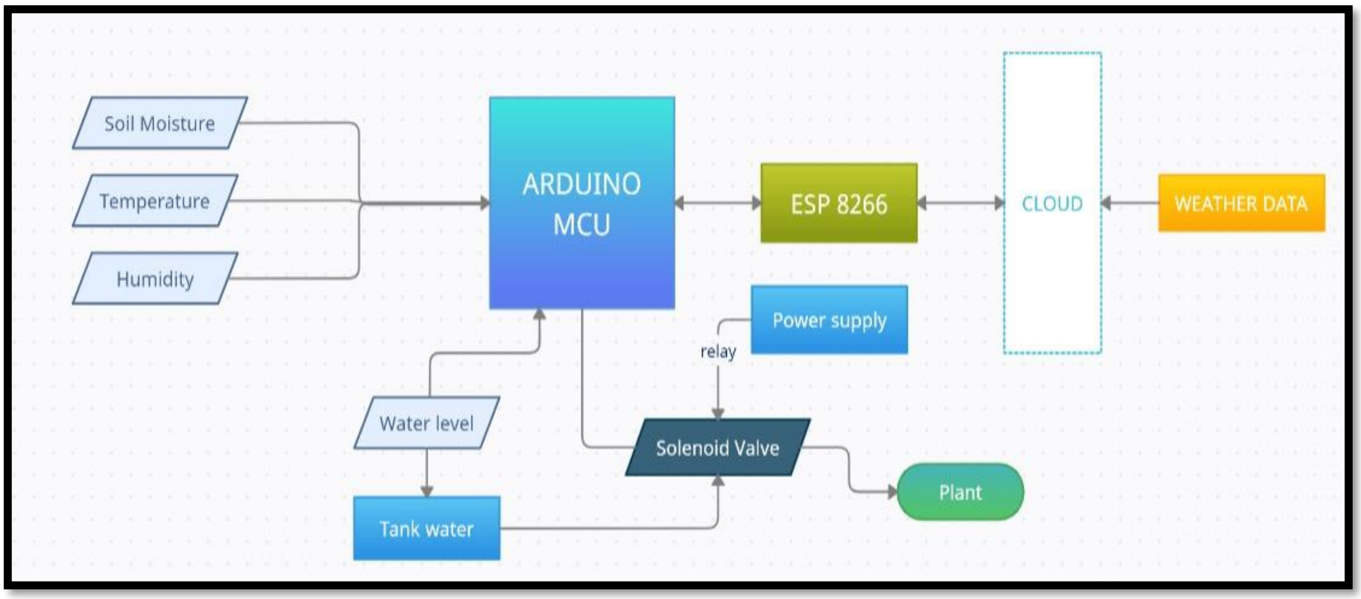
1) Arduino IDE: The Arduino IDE is used for creating codes(a.k.a sketches) to be uploaded to the Arduino UNO Board.

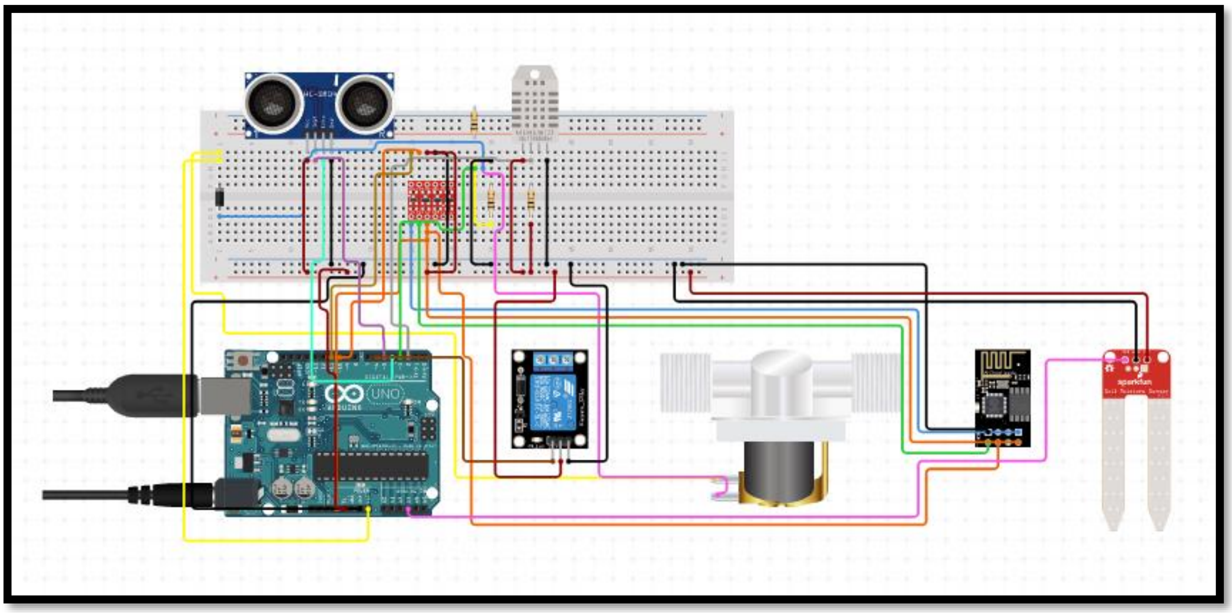
2) ThingSpeak: ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak from your devices, create instant visualization of live data, and send alerts. We send the data to ThingSpeak and get visualization of the data with time. So, it acts as a cloud with inbuilt cloud computing features

3) Atom: It is an open source hackable text editor used by developers to develop software. Atom was used to build the frontend website of this project, and also to integrate weather forecast and ThingSpeak API's with website.

4) Github: GitHub is a website for developers and programmers to collaboratively work on code. We host our website in github.

BLOCK SCHEMATIC:





The following are the various components that will be used in out circuit. There will be an external power supply from 11.1V Li-Ion battery which is connected to 7805 and 7803 IC for obtaining 5V and 3.3V, respectively. The solenoid valve will be actuated using a relay module that decouples the circuit from high power devices. The placement of the Ultrasonic sensor will be in the top of the drum while the moisture sensor is placed inside the soil. The DHT11 measures the air temperature and humidity in the surrounding area.

IMPLEMENTATION STEPS/IMPLEMENTATION:

- Connect the circuit as shown in the above figure
- In the breadboard, make sure your connections are firm and always use a multimeter to test the same
- For indication, you can use an LED as well
- When being externally powered, make sure you are having common ground for external battery and Arduino to ensure proper sensor readings.
- Mount the Ultrasonic sensor rigidly ensuring it does not fall into the water during measurement – using double side tape and tag wires.
- The relay is used for isolating the solenoid valve and Arduino circuit due to high power consumption of power by valve.
- Make appropriate connections to the ESP8266 module. – since max rating is 3.3V and Tx pin of Arduino outputs 5V, use a voltage divider.
- Update the firmware of the ESP8266 to ensure it is up to date.
- Prepare different functional blocks for different sensors and write individual codes for the same.
- Ensure proper libraries are available and added for the code implementation such as dht11.
- Debug and make sure code run for different training scenarios.
- Build the circuit in Hardware and run the software code in Arduino.
- Check if the values are being properly updated in the Thingspeak server.

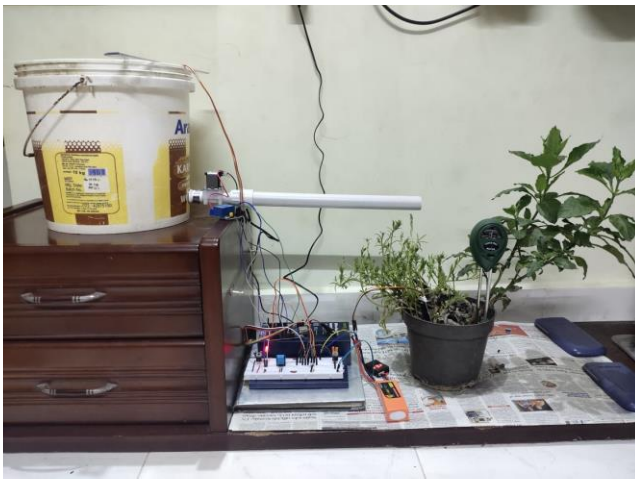
OUTPUTS/SIMULATION DETAILS/PROGRAM CODES:

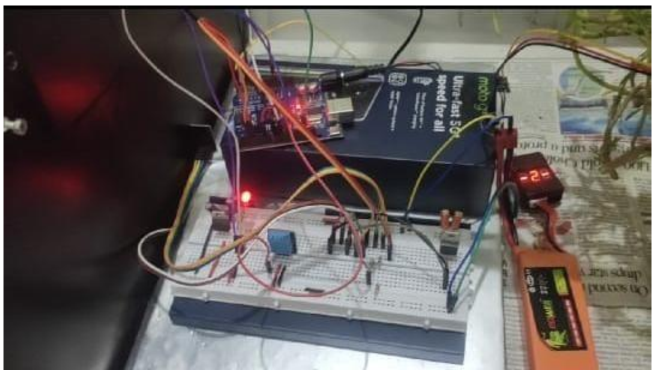
```
    #include <SoftwareSerial.h>
    #define RX 2
    #define TX 3
    String AP = "TP-Link_41EC";       // AP NAME
    String PASS = "rand_pass"; // AP PASSWORD – not original
    String API = "5MANDCF04CXNGYUR";   // Write API KEY
    String HOST = "api.thingspeak.com";
    String PORT = "80";
    int countTrueCommand;
    int countTimeCommand;
    boolean found = false;
    int valSensor = 1;
    SoftwareSerial esp8266(RX,TX);

   //TEMPERATURE AND HUMIDITY SENSOR

    #include <dht.h>
    #define dht_apin A1
    dht DHT;
    int air_temp = 0;
    int air_hum = 0;

   // SOIL MOISTURE SENSOR

    #define soil_moist A0
    const int AirValue = 620;
    const int WaterValue = 310; //calibration units
    int soilMoistureValue = 0;
    int soil_moisture = 0;

  // ULTRASONIC SENSOR

    const int EchoPin = 8;
    const int TrigPin = 9;
    int water_level = 0;

  // RELAY MODULE - SOLENOID VALVE

    const int relay = 11;
    volatile byte relayState = LOW;
    int relay_time = 0;


//---------SETUP_CODE--------//

void setup()
```

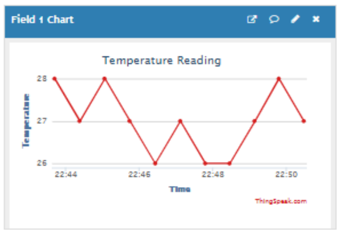
```
{
  // SETTING UP ESP_8266 //

  Serial.begin(9600); // open serial port, set the baud rate to 9600 bps
  esp8266.begin(115200); // set baud rate of esp8266 to 115200

  sendCommand("AT",5,"OK");
  sendCommand("AT+CWMODE=1",5,"OK");
  sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",20,"OK");

 // ULTRASONIC SENSOR //

   pinMode(TrigPin, OUTPUT);
   pinMode(EchoPin, INPUT);

 // RELAY MODULE //

   pinMode(relay, OUTPUT);
}

//---------LOOP_CODE--------//
void loop()
{

//-----GETTING SENSOR VALUES-----//

 // SOIL MOISTURE SENSOR //

   soilMoistureValue = analogRead(A0);
   soil_moisture = Soil_mois(soilMoistureValue);

 // ULTRASONIC SENSOR //

   water_level = Ultrasonic_dist();

 // TEMPERATURE AND HUMIDITY //

   air_temp = temperature();
   air_hum = humidity();

 // RELAY MODULE //

   relay_time = Relay_state(soilMoistureValue, water_level, air_temp, air_hum);
   if(relay_time > 0)
      {
       digitalWrite(11,HIGH);
       delay(relay_time);
       digitalWrite(11,LOW);
      }
```

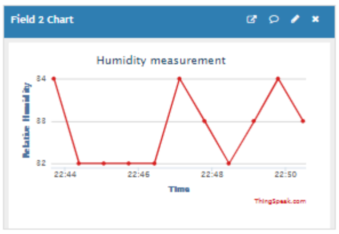
```
//SENDING DATA TO THINGSPEAK SERVER
   {
     String getData = "GET /update?api_key="+ API
+"&field1="+air_temp+"&field2="+air_hum+"&field3="+soil_moisture;
     sendCommand("AT+CIPMUX=1",5,"OK");
     sendCommand("AT+CIPSTART=0,\"TCP\",\""+ HOST +"\","+ PORT,15,"OK");
     sendCommand("AT+CIPSEND=0," +String(getData.length()+4),4,">");
     esp8266.println(getData);
     delay(1500);
     countTrueCommand++;
     sendCommand("AT+CIPCLOSE=0",5,"OK");
  }
}


//---------FUNCTIONS--------//

// ULTRASONIC
int Ultrasonic_dist()
{
        long distance = 0;
        long duration = 0;
        long height = 0;
        digitalWrite(TrigPin,LOW);
        delayMicroseconds(2);

        digitalWrite(TrigPin,HIGH);
        delayMicroseconds(10);
        digitalWrite(TrigPin,LOW);

        duration = pulseIn(EchoPin, HIGH);
        distance = 0.017*duration;
        Serial.print("Distance: ");
        Serial.println(distance);

        // distance is measured from top
        // distance > 15 - LOW
        // 5 < distance < 15 - MEDIUM
        // distance < 5 - HIGH

return height; //RETURN DISTANCE IN CENTIMETER
}

// SOIL MOISTURE
int Soil_mois(int soilMoistureValue)
{
    int soilmoisturepercent = 0;

    Serial.println(soilMoistureValue); //serial monitor print
    soilmoisturepercent = map(soilMoistureValue, AirValue, WaterValue, 0, 100);
    if(soilmoisturepercent >= 100)
      {
        Serial.println("100 %");
      }
    else if(soilmoisturepercent <=0)
      {
        Serial.println("0 %");
      }
    else if(soilmoisturepercent >0 && soilmoisturepercent < 100)
      {
        Serial.print(soilmoisturepercent);
        Serial.println("%");
      }
   delay(250);

return soilmoisturepercent; //RETURN MOSITURE AS PERCENTAGE
}

// AIR TEMPERATURE
int temperature()
{
    int temp;
    DHT.read11(dht_apin);
    temp = DHT.temperature;

return temp;
}

// AIR HUMIDITY
int humidity()
{
    int hum;
    DHT.read11(dht_apin);
    hum = DHT.humidity;

return hum;
}

// RELAY MODE - (Sample shown to present report)
int Relay_state(int soilMoistureValue,int water_level, int air_temp, int air_hum)
{
   if(soil_moisture<20 && water_level>5 && air_temp<=35 && air_hum>=50)
     {
        relay_time = 3000;
         //return the amount of time the valve neeeds to be turned on;
     }
   else if(20<soil_moisture<60 && water_level>10 && air_temp<=35 && air_hum>=50)
         //additional constraints can be added
     {
        relay_time = 1000;
      }
    else if(soil_moisture>80 && water_level>=0 && air_temp<=35 && air_hum>=60)
     {
          relay_time = 0;
     }
return relay_time;
}

//SEND COMMAND TO ESP8266
void sendCommand(String command, int maxTime, char readReplay[])
{
  Serial.print(countTrueCommand);
  Serial.print(". at command => ");
  Serial.print(command);
  Serial.print(" ");
  while(countTimeCommand < (maxTime*1))
  {
    esp8266.println(command);//at+cipsend
    if(esp8266.find(readReplay))//ok
    {
      found = true;
      break;
    }

    countTimeCommand++;
  }

  if(found == true)
  {
    Serial.println("OYI");
    countTrueCommand++;
    countTimeCommand = 0;
  }

  if(found == false)
  {
    Serial.println("Fail");
    countTrueCommand = 0;
    countTimeCommand = 0;
  }
  found = false;
 }
```

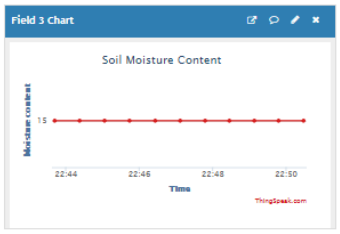
DETAILED PICTURE OF THE HARDWARE/SOFTWARE DEVELOPED:

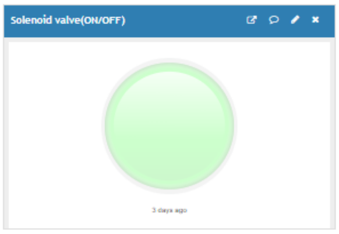


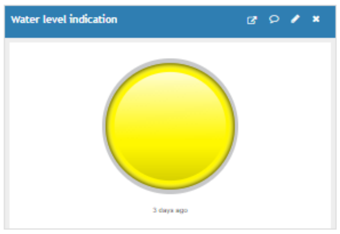


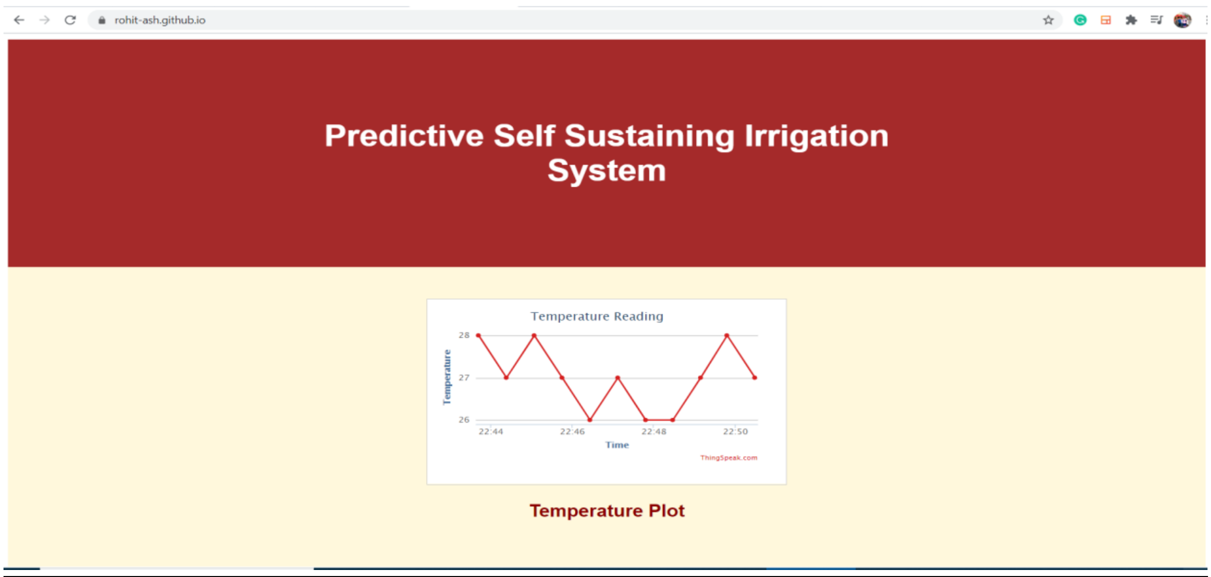






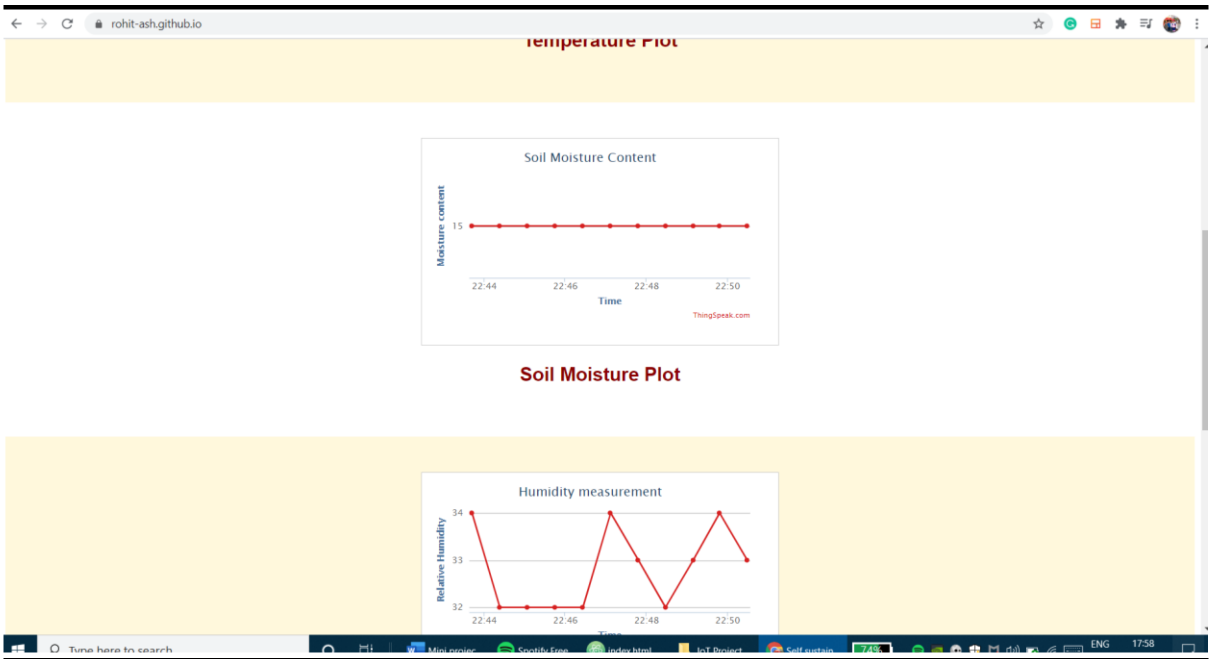






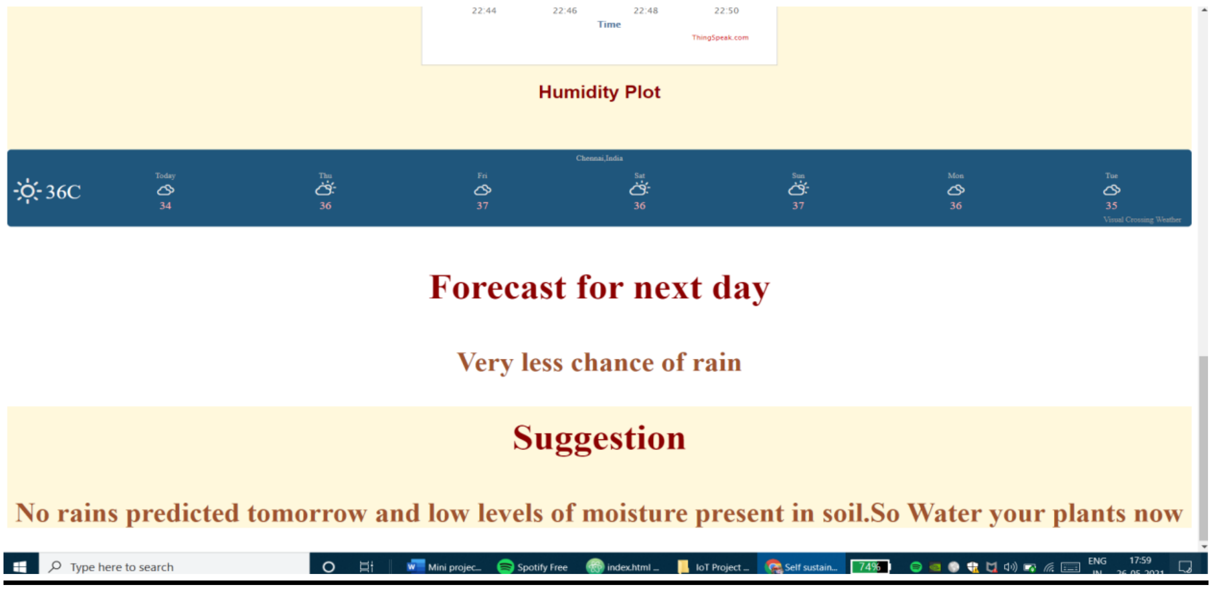
Predictive Self Sustaining Irrigation System

Temperature Plot



Temperature Plot

Soil Moisture Plot



Humidity Plot

Forecast for next day

Very less chance of rain

Suggestion

No rains predicted tomorrow and low levels of moisture present in soil.So Water your plants now

Website Link: https://rohit-ash.github.io/

REAL TIME APPLICATIONS:

We can use this project(device) to monitor the water levels of agricultural land and irrigate land (provided we scale up), appropriately watering the plants, instead of using the canal system of irrigation.

Also, the applications of our project can be extended to monitor water levels and irrigate/water home gardens as well as greenhouses.

Our project can also be extended to be used in aquaponics/hydroponics setups, where plants are grown without using soil.

FUTURE SCOPE:

We can use this project(device) to monitor the water levels of agricultural land and irrigate land (provided we scale up), appropriately watering the plants, instead of using the canal system of irrigation.

Also, the applications of our project can be extended to monitor water levels and irrigate/water home gardens as well as greenhouses.

Our project can also be extended to be put to use in aquaponics/hydroponics setups, where plants are grown without using soil.

The following points sum up the future scope/additions that can be made to the project to enhance its capabilities:

- We can devise methods to enable automatic actuation of valve based on output based on weather data, instead of just based on soil moisture.
- Appropriate software backend architecture can be developed for giving notification to user only when watering plants is required.
- The chemical composition of soil can be analyzed and the fertility of soil can be monitored remotely.
- Specific properties of plants can be studied and the irrigation of the crops can be done in a more judicious way.
- Machine learning and deep learning techniques can be employed for classifying crops automatically (using camera) and watering it as per its requirement.
- A similar model can be developed for the use of fertilizers as well.

CONTRIBUTION OF EACH STUDENT:

- **PREITY -** Documentation + Interfacing Ultrasonic Sensor
- **VAISNAV –** ESP8266 and DHT sensor interfacing + Code Debugging
- **VAIBHAV -** Hardware Interfacing + Hardware Implementation
- **ROHIT –** Website front end + Weather forecast API integration with website + Interfacing Soil Moisture Sensor

REFERENCES:

1. https://create.arduino.cc/projecthub/abdularbi17/ultrasonic-sensor-hc-sr04-with-arduino-tutorial-327ff6
2. https://create.arduino.cc/projecthub/muhammed-shameel-k-v/how-to-use-a-relay-with-arduino-e04e3c
3. https://create.arduino.cc/projecthub/user16726/configuring-the-esp8266-using-an-arduino-0ab2e6
4 .https://how2electronics.com/interface-capacitive-soil-moisture-sensor-arduino/
5. https://create.arduino.cc/projecthub/pibots555/how-to-connect-dht11-sensor-with-arduino-uno-f4d239
6. https://www.youtube.com/watch?v=5f_wOVnBb4g
7. https://www.visualcrossing.com/resources/documentation/weather-api/how-do-i-add-weather-forecast-to-my-webpage/