BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE (BITS), PILANI

Lab Oriented Project (LOP): IoT Based Intelligent Farm Monitoring Systems

END SEMESTER REPORT, 2nd SEMESTER 2021-22

Submitted To: Dr. Satyendra Kumar Mourya
Submitted By: Nishtha Pareek
ID: 2019B1A81044P

Acknowledgements

I would like to thank Dr Sashikant Sadistap (CEERI, Pilani) and Dr Satyendra Kumar Mourya (BITS Pilani) for allowing me to do a project under their guidance and giving helpful suggestions throughout the progress of the project. I would also like to thank AgroElectronics Group, CEERI and the Department of Electronics and Electrical Sciences, BITS Pilani for giving me a wonderful opportunity to do this project as a Lab-Oriented Project.

Abstract

Population explosion, urbanization, rising pollution levels, food scarcity, water depletion and looming danger of annual crop damage due to changing climate is forcing us to devise a new methodology for agriculture. Vertical Farming is an emerging agricultural technology which aims to minimize land use and optimization of resources. With the advent of renewable energy, IoT, and sophisticated architecture, Vertical Farming is a promising technology to build a sustainable future.

Reports have shown that the use of IoT for monitoring environmental conditions in Vertical Farms has resulted in lower water and power consumption, thus reducing the overall cost of Vertical Farming setup in the long run. Therefore this report contains details of a prototype of IoT Based monitoring system for Hydroponics Vertical Farms. The report mentions why Hydroponics can be a good alternative to conventional farming in arid and semi-arid regions. It then discusses IoT implementation covering modular and electrical components, their purpose in module and their operation. The next section discusses suitable crops for Vertical Farms. It also mentions specifics and growth parameters and cycles of lettuce, one of the widely cultivated hydroponic crops. The following section discusses the budget and components that need to be ordered for developing the fully scaled and functional prototype. Then, the report discusses the hardware implementation of the project using Arduino Uno, and temperature, pH, and distance sensors. This part contains the prototype of the electronics module which needs to be connected to water pumps and the modular setup for a complete IoT monitored hydroponics setup. This is followed by the future scope of work and appendix. The appendix contains a demonstration of small projects with the same electronic components. These small projects were then integrated to develop the main electronics module. At last, the references to develop this project are mentioned.

Table of Contents

1. Agricultural Problems in Rajasthan

Rajasthan is the largest state in the country in terms of geographical area. It has an area of 34.22 million hectares which is 10.41% of the total area of the country. The state can be divided into four major zones geographically: the western desert with barren hills, level rocky/sandy plains, the Aravalli hills and the south-eastern plateau. The annual temperature in the state ranges from the highest 50℃ in summers to 0℃ in winters. The annual rainfall varies between 150 mm-480mm in arid regions and 1000mm in the south-eastern plateau. The population of Rajasthan is 80.78 millions (Census 2021) and the livestock population is around 57.73 million (Livestock Census 2019).

1.1 Land Quality and Degradation

The land use of Rajasthan is peculiar in two ways: 1. Large desert cover 2. The Aravali range of hills divides the state into two distinct regions. The west of Aravali is arid and semi-arid whereas the east of Aravali is humid and sub-humid. According to the degraded and wetland statistics of Rajasthan, 67% of area is affected by desertification and/or land degradation where the wind erosion (44.2%) is the maximum contributor followed by water (11.2%), vegetal degradation (6.25%) and salinization (1.07%). Land degradation and desertification is mainly attributed to:

(i) Wind erosion – Landforms in Thar desert such as sand dunes are vulnerable to wind erosion. High human and livestock pressure with historical dry climates has also contributed to localized wind erosion or soil reactivation. The practice of mechanized deep ploughing and increase in the net sown area are also accelerating the Aeolian
processes,
(ii)Water erosion – parts of Aravali hill ranges, eastern margin of Thar desert are experiencing accelerated rill and gully erosion.
(iii) Water logging and salinity – excess irrigation and wrong drainage planning have caused water logging and salinity build up in canal command areas.

1.2 Irrigation

The average annual rainfall of western arid region is 317 mm. The rainfall is highly variable and erratic. The number of rainy days varies from 10 to 25. Groundwater is very deep, saline at many places and expensive to use. Indiscriminate use of water on undulating highly permeable sandy soils through conventional irrigation results in fall of groundwater by 0.6-1m annually. Rainfed

farming is adversely affected by low and erratic rainfall coupled with high evaporative demand and low moisture retention by light textured soils. On the other hand indiscriminate use of scarce water through conventional irrigation management practices have led to exhaustion of ground water resources amd also water logging and salinity in the canal command areas.

1.3 Drought and Calamities

In Rajasthan, there have been 48 drought years of varied intensity during last century (i.e. from 1901 to 2002), which means that the chance of occurrence of a meteorological drought in the state is 47 per cent (Rathore, 2004). The state has the maximum probability of occurrence of droughts in India (Singh et al., 2010). A 22 detailed analysis has revealed that during the past 9 years period, none of the districts in the state were affected by drought. The number of severe and very severe drought years is larger in the western and southern part of Rajasthan.

2. How Hydroponics can solve these problems

In Hydroponics or Vertical Farming, the crops are grown in vertical stacks in high-rise buildings. he productivity per foot of Vertical Farm is reported to be 130 times superior to soil-based settings and ten times higher than a greenhouse. Since, vertical farming does not use soil for growth medium, it also reduces the land degradation due to agriculture.

Aerofarms reported in 2018 that Vertical Farms utilize 95% less water than traditional farms due to water and nutrients directly supplied to roots. 40% and 98% less water than hydroponics and the conventional Farm, respectively. From various case studies of the crops grown in Vertical farming systems, it is clear that they have a much shorter plant life cycle than their traditional counterparts.

Since plants are grown in controlled environmental conditions in a Vertical farm, loss of crop yields due to climate is wholly eliminated. Implementation and control of pest and disease management systems are also more simplified and effective in the Vertical Farming systems.30% of the harvested crop annually is wasted during storage and transport. Since vertical farms are situated in cities, it eliminates the transportation and long storage requirements, saving energy and costs and providing fresher produce to the consumers.

3. IoT Monitoring in Hydroponics

Smart Farming refers to the application of technological resources to help in various stages of crop production. It derives from precision agriculture techniques and optimizes crop production by improving the nutrient application, reducing water and electricity consumption, and monitoring disease and pest growth. For instance, the monitoring system in Vertical Farming collects data about the status of crop and environmental conditions in the Vertical Farm through various sensors, which can then be transmitted over communication networks and then analyzed by management information systems. This example depicts an Internet of Things (IoT framework) where a network of interconnected intelligent devices communicate with each other to generate information about the environment. (Navarro et al., 2020) present an IoT Solution Architecture consisting of Perception, Transport, Processing, and Application as four layers encompassing the main components of IoT architecture: devices, network, services, and application.

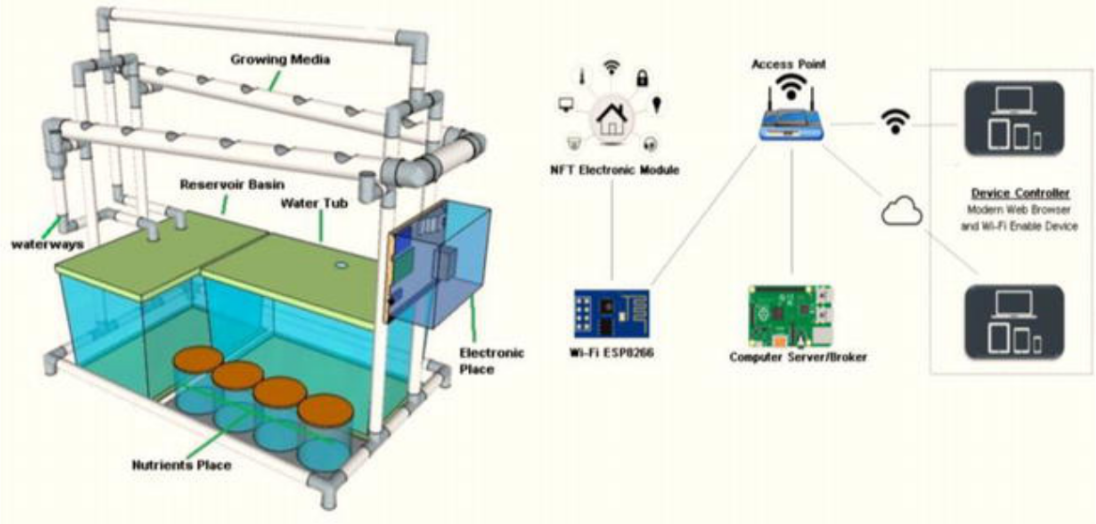
The perception layer consists of devices that collect data from the environment, such as sensors or unmanned aerial vehicles (UAV), enabling inter-device communication. The transport layer comprises various application and network protocols to transmit data between the perception and processing layer via wireless communication between sensor nodes and applications. Each protocol is characterized by range, data exchange rate, and power consumption. To ensure compatibility between IoT and non-IoT devices and transmit information to cloud platforms, application protocols such as MQTT (Message Queue Telemetry Transport) and CoAP (Constrained Application Protocol) are used. The processing layer store, visualize, and process. Artificial intelligence and machine learning models are employed to detect patterns and correlations among the unorganized and complex data to develop decision support systems and automation of the farming process.

It has been found out that Integration of hydroponics and IoT framework has improved the yield by 17% to earlier hydroponics systems and 87% to the soil-based system. Water monitoring is also more efficient with less water consumer per kg per crop. The development time of a single crop is also being reported to be reduced in the IoT monitored hydroponics system.

3.1 IoT framework for hydroponics with the justification of parameters to be measured

The proposed setup's objective is to provide automated monitoring of a lettuce producing hydroponics system. The setup is divided into major three modules:

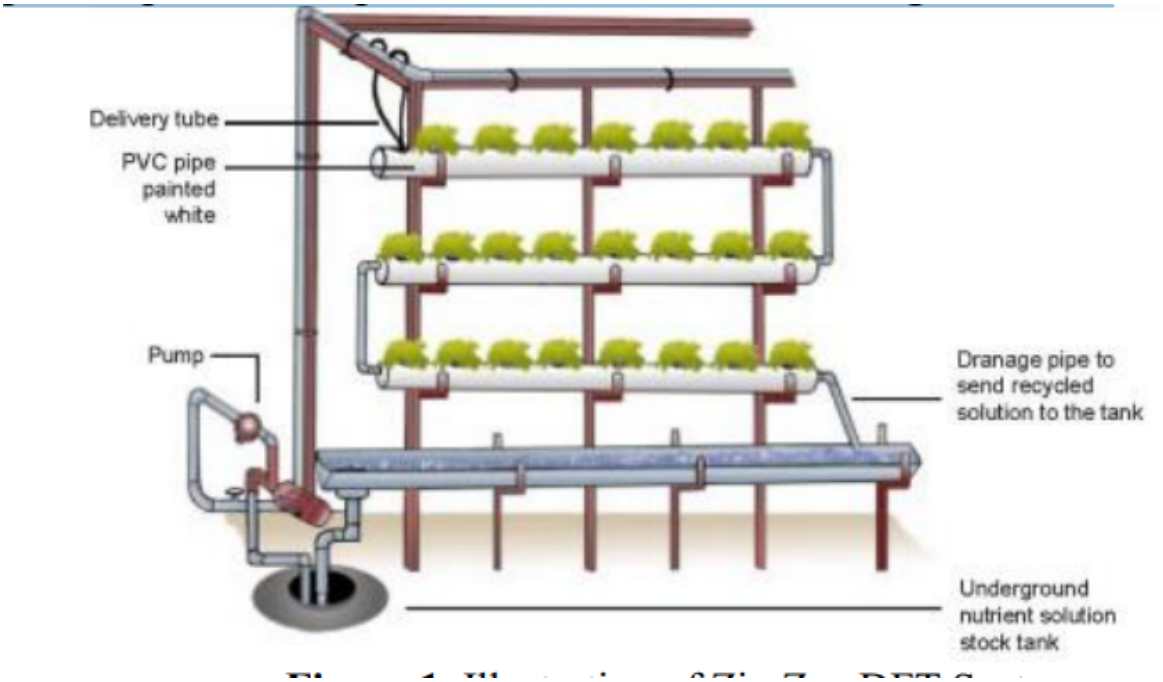
1. **Modular Structure:** It will consist of PVC pipe and aluminium frame to support the hydroponic system.
2. **Electronic Circuit:** For monitoring the environmental parameters in the hydroponics tank and pumping the nutrient solution.
3. **Sensors/Actuators:** Measure and control all the data and parameters of the system.



3.1.1 Modular Structure:

The modular structure will employ a Deep Flow Technique for hydroponics system.It is a method of cultivating hydroponic plants by placing plant roots in deep water layers. The working principle of the DFT hydroponic system is to circulate the plant nutrient solution continuously for 24 hours. This hydroponic technique is categorized as a closed hydroponic system.

Generally, the application of this hydroponic technique is used in the cultivation of leaf and fruit vegetable plants. In the pipe system DFT technique, the flow of nutrients with a depth of 2-3 cm flows on a PVC pipe with a diameter of 10 cm and the plant is placed in a plastic pot, so the plant will receive the flowing nutrients.



Implementation: There are some online sources from where Deep Flow hydroponics PVC modules can be procured.

Sources where similar module can be brought:

1. Hydroponic System For 32 Plants - DFT (Deep Flow Technique): The DFT system is an all-inclusive kit that comes with net pots, clay balls, water pump, reservoir, nutrients, pH Down, grow cube, pH test strip and even seeds. The instructions booklet is also provided with the setup.
   Green leafy plants can be grown easily. Cost is around Rs. 5499 on discount.
   Installation Demo: https://www.youtube.com/watch?v=9KwBtvTkeUQ

3.1.2 Hydroponics Sensors

This module will allow us to determine the state of plants by measuring several parameters.

1. Time Clock Module (DS3231): The Time Clock Module (or DS3231) is a module that measures the time, dependently or independently of his Arduino card through of his cell. The Arduino card measures the elapsed time since the module was turned on (in ms). The module comes assembled ready-to-use, with a battery (supplied) that lasts up to 3 years. This module will be used to measure time since the start of the experiment and based on its output time, nutrients will be supplied and pumps will be turned on/off.
2. DHT22 AM2302 Digital Temperature Sensor: It is a temperature and humidity sensor that will give the readings of air temperature and humidity that helps to track the climate conditions.

Biological Significance: Temperature controls the rate of plant growth. Generally, as temperatures increase, chemical processes proceed at faster rates. Most chemical processes in plants are regulated by enzymes which, in turn, perform at their best within narrow temperature ranges. Above and below these temperature ranges, enzyme activity starts to deteriorate and as a result, chemical processes slow down or are stopped. At this point, plants are stressed, growth is reduced, and, eventually, the plant may die. The temperature of the plant environment should be kept at optimum levels for fast and successful maturation. Both the air and the water temperature must be monitored and controlled.

3. DS18B20 Water Proof Temperature Sensor Probe: It is a water-proof temperature sensor probe to give readings of water temperature inside hydroponics.

4. Analog PH Sensor Kit for Arduino: The pH of nutrient-enriched water must be taken care of. Readings are taken from the water and will track the effective pH.
   Biological Significance: The pH of a solution is important because it controls the availability of fertilizer salts. A pH of 5.8 is considered optimum for the described lettuce growing system, however, a range of 5.6- 6.0 is acceptable. Nutrient deficiencies may occur at ranges above or below the acceptable range.

5. SeeedStudio Grove EC Sensor Kit (DJS1CBlack): Electrical Conductivity of nutrient solution is due to different ions such as nitrate, potassium and other mineral ions. Different concentrations of these ions are needed at different phases of growth for the plant. Therefore, this sensor will measure the EC of the nutrient solution in a hydroponics medium.
   Biological Significance: Electrical conductivity (EC) is a measure of the dissolved salts in a solution. As nutrients are taken up by a plant, the EC level is lowered since there are fewer salts in the solution. Alternately, the EC of the solution is increased when water is removed from the solution through the processes of evaporation and transpiration. If the EC of the solution increases, it can be lowered by adding pure water, e.g., reverse osmosis water). If the EC decreases, it can be increased by adding a small quantity of a concentrated nutrient stock solution.

6. Dissolved Oxygen Sensor: It will be connected to the Arduino module to measure the dissolved oxygen content in the water. One operational challenge is the high cost of this sensor.
   Biological Significance: Lettuce will grow satisfactorily at a DO level of at least 4 ppm. If no oxygen is added to the pond, DO levels will drop to nearly 0 ppm. The

absence of oxygen in the nutrient solution will stop the process of respiration and seriously damage and kill the plant. Pure oxygen is added to the recirculation system in the ponds. Usually the level is maintained at 8 (7-10, no advantage to 20) ppm. For sufficiently small systems, it is possible to add air to the solution through an air pump and aquarium air stone but the dissolved oxygen level achieved will not be as high as can be achieved with pure oxygen.

3.1.3 Actuators

1. **Water Pump:** The pumps will be connected to motors, which will pump the nutrient solution in the hydroponics tank. The microcontroller will instruct the pumps to be turned off in certain hours of days and also increase/decrease the rate of nutrient liquid flow based on the pH and EC of the nutrient solution in the hydroponics column.
2. **Growing Light:** These will be the LED lights that will fulfill the lighting requirements of the crop being grown in the hydroponics module. The power requirements of LED lights will be determined based on the wavelength of light needed and the models available online.
3. **Nutrient Feeder:** This module will contain powdered nutrients in plastic/glass bottles and will be mixed with the nutrient solution in the reservoir tank depending on the nutrient levels.

3.1.4 Electronics Module

1. **Arduino MEGA:** It will be the microcontroller to implement the automation. The microcontroller will be connected to all the sensor modules and based on the coded instructions, will rotate the motors so as to maintain the optimum amount and concentration of the nutrient solution. It is also connected to wifi and Bluetooth module to relay data to the ThingSpeak which will perform data analysis.

   Arduino MEGA was chosen over Arduino Nano or Uno because of more number of I/O pins, more memory space, ability to control multiple devices, and extensive use in building automation systems.
2. **ESP8266 WiFi Module:** Arduino MEGA after collecting data from the sensors will it send to ESP8266 for uploading it on Thingspeak for data analysis
3. **LCD Display:** It will be used to display the value from the sensors without having to connect to the hydroponics cultivation wirelessly
4. **Power Supply:** Arduino MEGA runs optimally on a 12V power supply (recommended power supply is 7-12 V whereas power supply limits are 6-20 V). 12 V batteries can be used to supply power. In case of using an external AC supply from the main switch, we will use12V a power adaptor for Arduino and for motors, AC-DC Converter will be used.

3.1.5 Other Miscellaneous Modules:

1. Pumice Stone or Air Pump: For oxygen level maintenance

3.1.6 Data Analysis - ThingSpeak

1. ThingSpeak is an IoT Cloud platform where sensor data can be sent to the cloud. It can also be used to analyze and visualize the data with MATLAB or other software, including making its own applications.
2. ThingSpeak includes a Web Service (REST API) that lets you collect and store sensor data in the cloud and develop Internet of Things applications. It works with Arduino, Raspberry Pi and MATLAB (premade libraries and APIs exists) But it should work with all kind of Programming Languages since it uses a REST API and HTTP.
3. Another popular IoT platform taken into consideration was Blynk. But, ThingSpeak service since being operated by MathWorks is free for BITS mail accounts even for advanced versions whereas, for Blynk, only basic features are available for analysis with a free account. Apart from it, ThingSpeak offers Visual analytics, and the interface is more intuitive

3.1.7 Implementation

1. <u>Automation using Arduino MEGA included:</u>
    a. Water cycles by timers
    b. Light by timers
    c. Data measurement
    d. Nutrition by measurement (electric conductivity) and corrective action via nutrient dosing
    e. Pump control
    f. Water quality (pH) by measurement
    g. IoT Integration
    h. Data collection
    i. Recording of sensor measurements and action.
2. <u>Arduino Procedure</u>
    a. Keeps the time
    b. Reads sensors
    c. Communicates with the ThingSpeak via wifi module:- Sends data (pH, conductivity, water temperature, air temperature, air humidity, arduino

time, pump status and light status) - Receives the update of the time from the ThingSpeak.

d. Turns the pump on / off for certain minutes every hour
e. Turn the lights on/off
f. Transmits data to ThingSpeak at 15-minute intervals (pH, conductivity, water temperature, air temperature, air humidity.

4. Crop to be grown in Hydroponics - Lettuce

Lettuce is one of the most commonly grown hydroponics vegetables. The plants can be grown into a simple liquid nutrient solution, supported by an inert medium called aggregate culture. Lettuce has shorter life cycles and mature plants grow within 35-50 days. Lettuce cultivars are easily available. That is why lettuce can be a suitable candidate for this experiment.

Set-points for environmental parameters for optimum lettuce growth

S. No	Environmental Parameter	Set Points	Remedies to increase/decrease in hydroponics
1	Air Temperature	24 C Day/19 C Night (75 F/65 F)	It can be cooled by keeping in shade/AC. Temperature can be increased by keeping in warm conditions.
2	Water Temperature	No higher than 25C, cool at 26C, heat at 24C	Water temperature can be lowered by increasing the size of reservoir, or aquarium chillers. Aquarium heaters can be used for increasing the temperature.
3	Relative Humidity	minimum 50 and no higher than70%	Can be controlled by air fans
4	Light Intensity	17 mol m-2 d-1 combination of solar and supplemental light	It can be maintained by increasing/decreasing the number of HPS or LEDs and changing the timing when

			they are turned on/off
5	Dissolved Oxygen	7 mg/L or ppm, crop failure if less than 3 ppm	Can be maintained by supplying pure oxygen or adding an aquarium air stone or air pump.
6	pH	5.6-6	If the pH reading is high, add phosphoric acid, citric acid, vinegar or pH down products slowly. If the pH is low, add potassium hydroxide, potassium carbonate or a pH up product slowly.
7	Electrical Conductivity	1150-1250 µS/cm above the source water	If the reading is higher than the optimum solution, we dilute the solution by adding more water. If reading is lower than the optimum solution, add nutrient concentrate until optimum level is reached.

Lettuce Production:

The production of the lettuce crop is separated into two growing areas. Seeds are started in a germination area where they germinate and grow for 11 days. They should be shaded from the full sun on the first day after germination, but can then be exposed to full light (17 mol/m 2 /d) or slightly greater. On Day 11, the plants are transported to the greenhouse and transplanted into the pond area where they are grown until re-spacing on day 21 and finally harvested on Day 35.

1. **Germination Area Stage:** Scheduled for Production Days 0-11 and may occur in a growth chamber.
    a. Sowing: Day0: Sowing the seeds on germination media of 1” rockwool cubes that are 10 x 20 cells per sheet. Germination takes place in plug trays. One lettuce seed is placed into each plug. Rockwool should be moistened with nutrient solution that has a relatively low pH such as 4.5 to remove pockets of high pH contaminants. For the initial 24 hours, lighting

is maintained at 50 µmol/m 2 /s with a photoperiod (day length) of 24 hours to ensure good germination if a germination room is used. The temperature is set for 20C (68F) in the germination room. The seed trays may be covered with plastic humidity covers to ensure a high relative humidity which prevents desiccation

b. Environmental Adjustment: Day1: A fertilizer solution is added to the top or sub-irrigation water 24 hours after sowing. The EC of the water is maintained at 1200 µS/cm above the source water EC. The pH of the solution is adjusted to 5.8 with possible addition of a base, potassium hydroxide (KOH) and nitric acid when it is too high. The temperature is raised to 25C and the lights increased to 250 µmol/m2/s. These environmental factors are maintained for the remainder of the crops' time in the germination area. Sub-irrigation continues for 1/4 hour every 12 hours until Day 6. The photoperiod remains at 24 hours. If hand-watering is used the same watering frequency does not need to be used but care must be taken so that the media does not dry out.
c. Humidity Decreasing: Day2: The humidity covers in place on Days 0 and 1 are removed on Day 2. At this time, the seed has germinated and the radicle has started to penetrate into the soil, as can be seen in the above photo. High humidity levels during the first two days of germination are to ensure the seed does not desiccate. Low light levels during the first 24 hours work in conjunction with the high humidity to prevent excessive seed drying.
d. Removing Double Seedlings: Day3: Any double seedlings should be removed from the plugs on Days 3 or 4 to ensure a uniform crop. Any seedlings that are particularly large should be removed so they do not suppress the growth of neighbouring plants. Also, germination percentage can be determined to monitor seed quality and proper growing conditions at this stage. It is critical to have consistent environmental conditions and consistent plant growth during this stage.
e. Increasing water frequency: Day6: The sub-irrigation system if using an ebb and flood table is scheduled for flooding four times per day, or every six hours, for 1/4 hr (15 min)

2. Transplanting

Transplanting On Day 11, the seedlings are transported to the greenhouse and transplanted into the pond. Prior to transplanting, the seedlings are thoroughly sub-irrigated. Transplanting can be scheduled to follow normal sub-irrigation periods in order to prevent desiccation during transfer. The seedling plugs float in the pond in Styrofoam floats. Each float is hand-drilled from 1” insulation. A

wooden template placed over the Styrofoam board to be drilled hastens the drilling process. A drill press may be used if board geometry allows. Several holes can be drilled simultaneously if a clever drill press apparatus is created.

3. **Preparation of Nutrient Solution:** The lettuce nutrient solution formula comprises the following components: fertilizer A: 286g/T of calcium nitrate; fertilizer B: 454g/T of potassium nitrate, 163g/T of magnesium sulfate and 87g/T of ammonium dihydrogen phosphate; fertilizer: 22g/T of EDTA iron, 3.1g/T of boric acid, 2.2g/T of manganese sulfate, 0.55g/T of zinc sulfate, 0.09g/T of copper sulfate and 0.1g/T of ammonium molybdate; and organic acid: 410-460g/T of citric acid or 280-320g/T of malic acid or 280-320g/T of tartaric acid. According to the nutritional needs of lettuce hydroponics, a large number of elements and trace elements are used in balance; miscellaneous preparation processes are simplified according to the actual operation needs, and nutrient solution raw materials are all in the solid package, and the nutrient solution is prepared on the spot and is more appropriate for the requirements of family culture

5. Budget

5.1 List of Components to be Ordered:

1.

S.No	Component	Status of order and payment
1.	Hydroponic System For 32 Plants - DFT (Deep Flow Technique)	Order Pending
2.	Time Clock Module (DS3231):	Order Pending
3.	DHT22 AM2302 Digital Temperature Sensor	Ordered and payment done
4.	DS18B20 Water Proof Temperature Sensor Probe	Ordered and payment done
5.	Analog PH Sensor Kit for Arduino:	Ordered and payment done
6.	SeeedStudio Grove EC Sensor Kit (DJS1CBlack):	Order Pending
7.	Dissolved Oxygen Sensor:	Order Pending

8.	ESP8266 WiFi Module	Ordered and Payment Done
9.	LCD Display	Ordered and Payment Done
10.	Gravity Analog TDS meter	Order Pending
11.	Arduino Mega*	Order Pending
12.	Fertilizer	Order Pending
13.	Lettuce Seeds	Order Pending

*Regarding Arduino Mega, I am not sure if its needed essentially. Since, I have an Arduino Uno and it can support Analog pH sensor, DS18B20 water temperature and DHT22 temperature and humidity sensor. I am not sure if it has computation power to support TDS meter, dissolved oxygen sensor and TDS meter though. Therefore, its star-marked and is not included in essential component list

5.2 Estimated Budget

Component	Units	Cost/Unit	Total Cost
**Hydroponic System For 32 Plants - DFT (Deep Flow Technique)	1	₹5,499.00	₹5,499.00
Gravity: Analog Dissolved Oxygen Sensor/Meter Kit for Arduino	1	₹15,112.90	₹15,112.90
**SeeedStudio Grove EC Sensor Kit (DJS1CBlack)	1	₹ 3,699.00	₹ 3,699.00
**Gravity Analog	1	₹ 1,060.00	₹ 1,060.00

TDS meter			
**DS3231 Real Time Clock (RTC) Module	1	₹ 341.02	₹ 341.02
Arduino Mega 2560 R3 Board - Clone Model	1	₹ 2,004.82	₹ 2,004.82
**Lettuce Seeds	1	₹ 165	₹ 165
**Fertilizer	1	₹ 449	₹ 449
		Total	₹ 28,330.74
		Total (Only Essential Compnents)	₹ 11,213.02

6. Hardware Implementation

6.1 Apparatus

1. Arduino Uno
2. Breadboard
3. HC-SR04 Ultrasonic Sensor
4. DS18B20 Water Proof Temperature Sensor
5. Analog pH Sensor Kit
6. DC Motors
7. Battery 9V
8. L239D Motor Driver
9. Jumper Wires

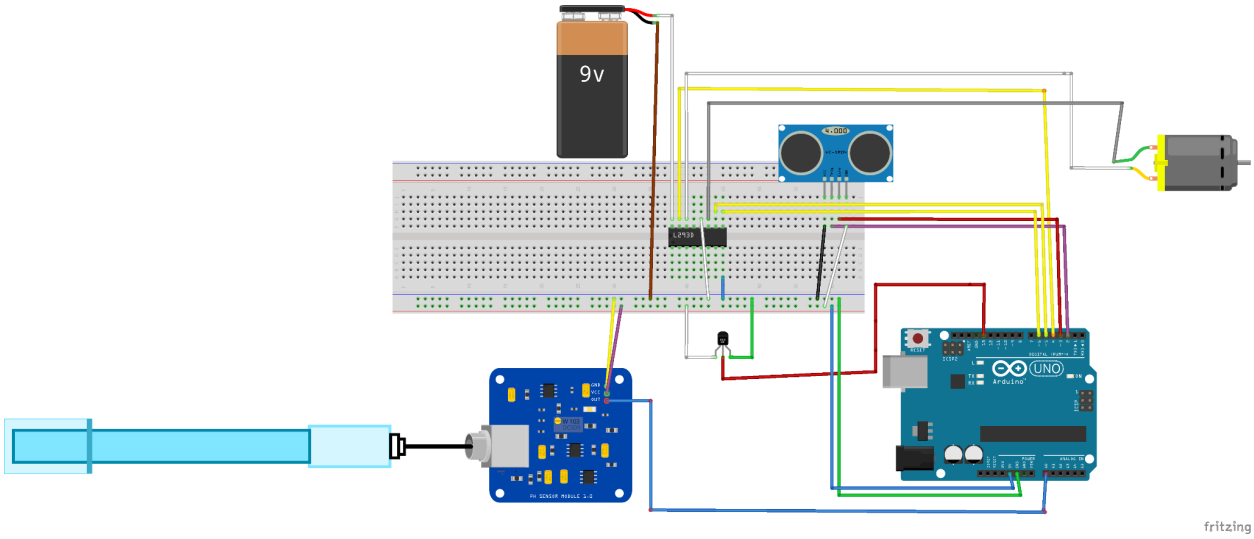
6.2 Software Used:

1. Arduino IDE

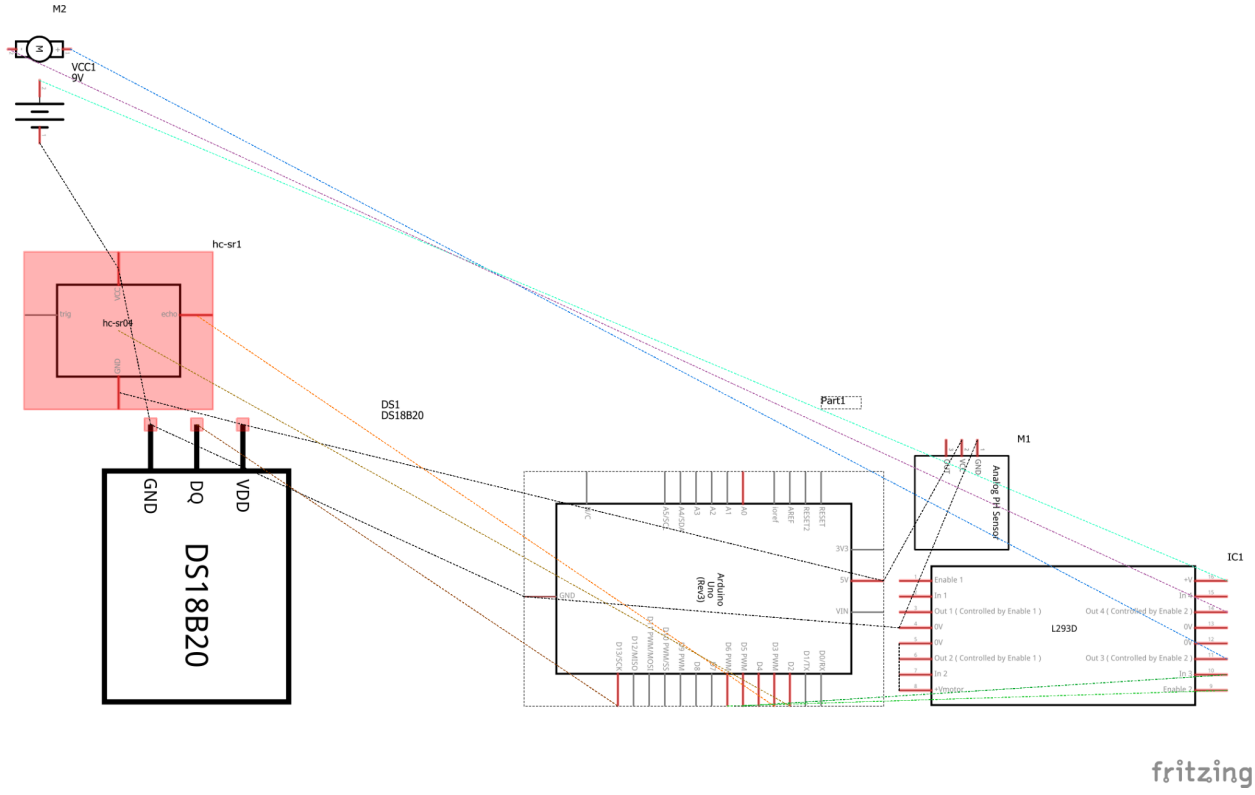
2. MATLAB: For plotting data
3. CoolTerm: For storing serial monitor data of Arduino IDE into text, spreadsheet or CSV format.
4. Fritzing: For drawing circuit diagrams

6.3 Circuit Diagram

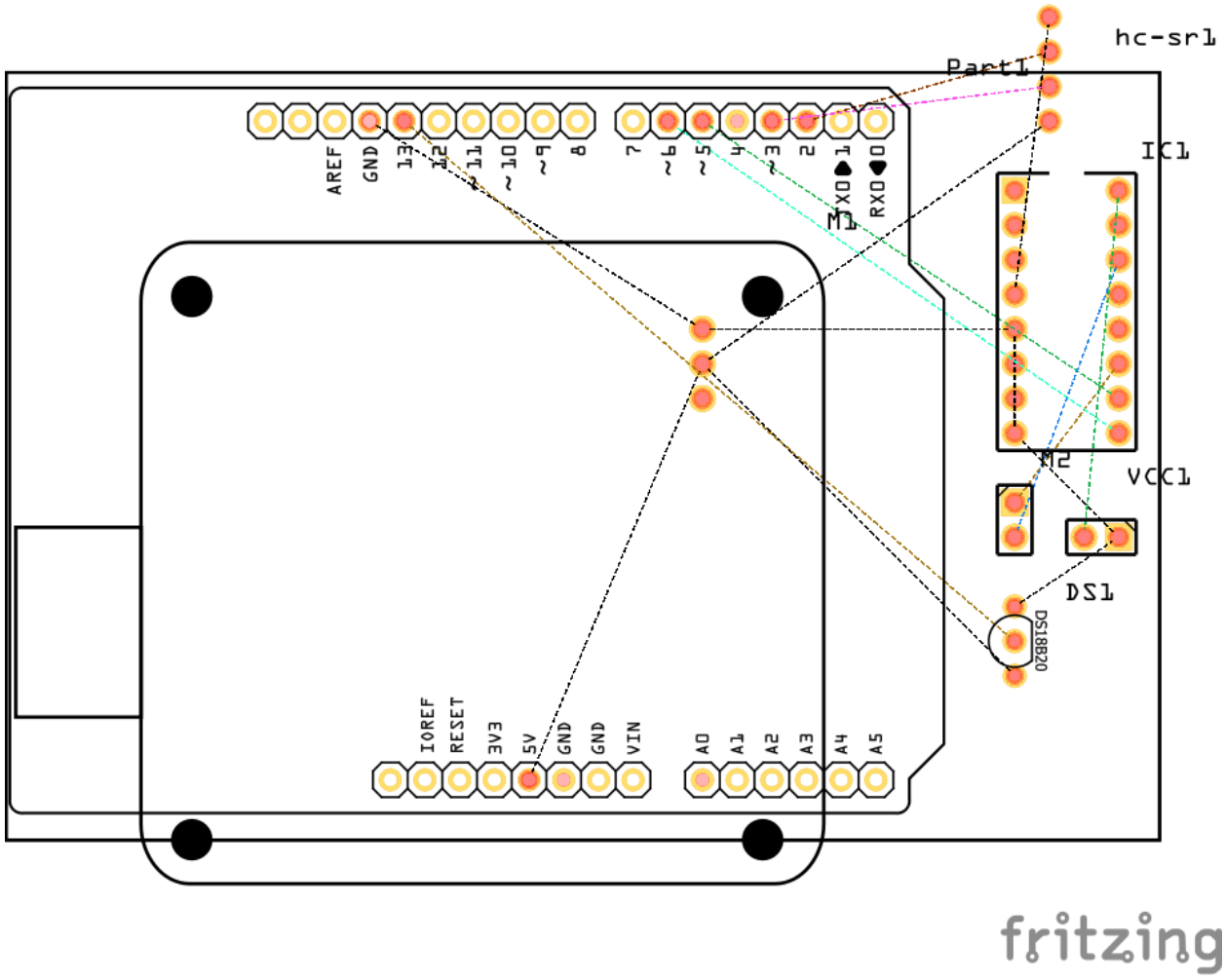
1. Breadboard



2. Schematic



3. PCB



6.4 Code Working:

Arduino

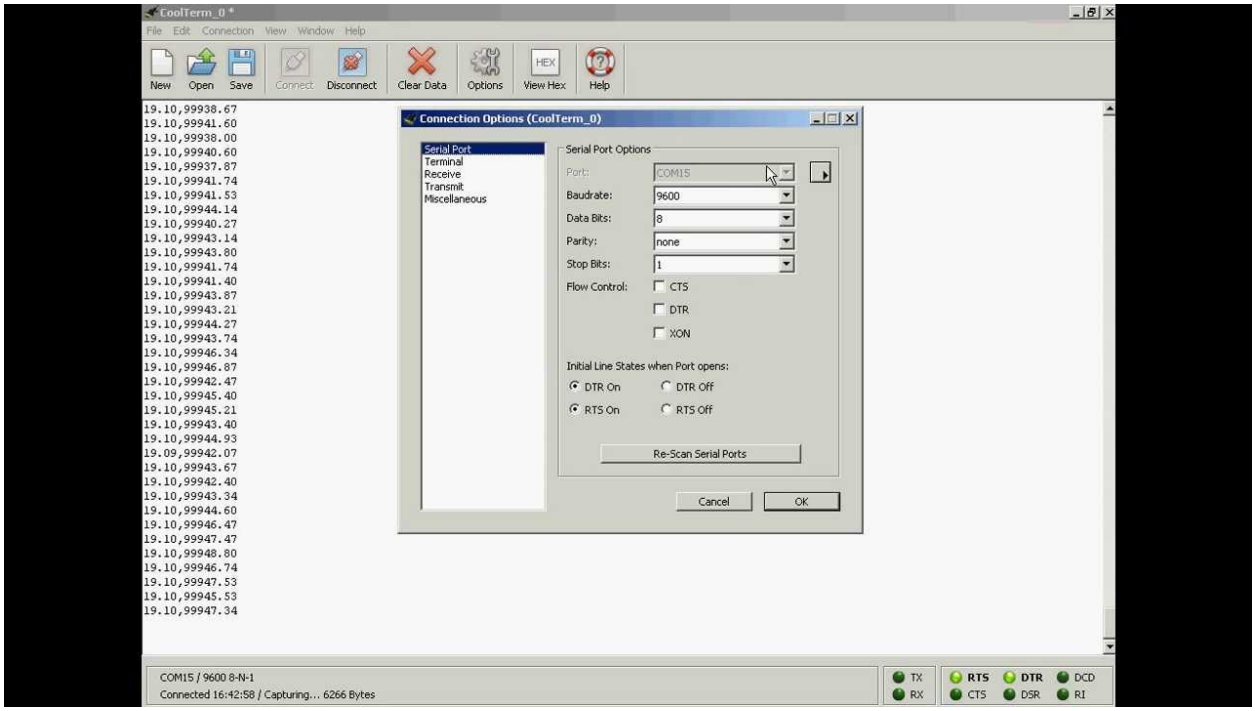
1. Include all the relevant libraries for various sensors using Sketch > Include LIbraries.
2. Choose the input pin and output pin and declare it for sensors.
3. Define variables for the input sensors and store the data from sensors in the variables.
4. Define the Arduino pin connections to motor

5. Define condition of motor rotation. Here, the motor rotates if any of the following condition is true:
    a. If the pH goes below minimum pH (5.5)
    b. If the pH goes above the maximum pH (7)
    c. If the temperature goes below 30 ℃
6. Print the values from sensors on serial monitor.

MATLAB

1. Install MATLAB Arduino Hardware Support Package. MATLAB > Add-Ons > Get Hardware Support Packages > Install from Internet > Next > Accept license agreement and download the packages.
2. Once the packages are installed, connect Arduino board to PC and type type the following command in MATLAB command window to ensure MATLAB can communicate with the arduino board.
3. >> a = arduino()
4. After Arrduino is successfully able to communicate with the MATLAB, declare pins which data need to be plotted using readVoltage or readDigitalPin commands.
5. Declare time variable to measure time elapsed.
6. Plot the graph of sensor data vs time using plot() function
7. Specify the axis labels and othe properties of graph and run the code.

CoolTerm

1. 

6.5 Code:

```
/* Library for DS18B20 Water Proof Sensor */

#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 12     /* Arduino pin which is receiving data
from the sensor */
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);


 float Celcius=0;           /*Declaring temperature variable */

 const int potPin=A0;       /*Arduino analog pin receiving data from
pH meter */
 float ph;                 /*Declaring pH variable */
 float Value=0;
 float pHMin = 5.5; /* the temperature to start the buzzer  */
 int pHMax = 7;

/* Declaring pins for Ultrasonic Distance Sensor */
#define echoPin 3 /* attach pin D4 Arduino to pin Echo of HC-SR04
*/
#define trigPin 2 /* attach pin D2 Arduino to pin Trig of HC-SR04
*/

/* defines variables for Ultrasonic sensor */
long duration; /* variable for the duration of sound wave travel */
int distance; /* variable for the distance measurement */


/* Motor connections to Arduino pins */
int enA = 4;
int in1 = 5;
int in2 = 6;


void setup() {

  Serial.begin(9600);
```

```
  /* Set the pH Sensor */
  pinMode(potPin,INPUT);

  /* Set all the motor control pins to outputs */
  pinMode(trigPin, OUTPUT); /* Sets the trigPin as an OUTPUT */
  pinMode(echoPin, INPUT); /* Sets the echoPin as an INPUT */
  pinMode(enA, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);

  /* Turn off motors - Initial state */
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);

}

void loop() {

  /* Obtaining values from DS18B20 temperature sensor */
   sensors.requestTemperatures();
  Celcius=sensors.getTempCByIndex(0)+137;


/* Find pH */
  Value= analogRead(potPin);
  float voltage=Value*(3.3/4095.0);
    ph=(3.3*voltage)+8;


 /* Operating Motor */

  if (ph>pHMax or ph<pHMin or Celcius<30 )
  {
    directionControl();
    delay(1000);
   speedControl();
   delay(1000);
  }

    /* Clears the trigPin condition */
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
```

```
  /* Sets the trigPin HIGH (ACTIVE) for 10 microseconds */
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
   /* Reads the echoPin, returns the sound wave travel time in
microseconds */
  duration = pulseIn(echoPin, HIGH);
  /* Calculating the distance   */
   distance = duration * 0.034 / 2; /* Speed of sound wave divided
by 2 (go and back) */



 /* Display the values on serial monitor */
  Serial.print(" Temperature in Celsius ");
  Serial.println(Celcius);
  Serial.print("pH: ");
  Serial.println(ph);
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  delay(10);
  /*Printing output on serial monitor*/

}


/* This function lets you control spinning direction of motors */
void directionControl() {
  /* Set motors to maximum speed */
  /* For PWM maximum possible values are 0 to 255*/
  analogWrite(enA, 255);

  /* Turn on motor A & B */
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  delay(2000);

  /* Now change motor directions */
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  delay(2000);
```

```
  /* Turn off motors */
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}

/* This function lets you control speed of the motors */
void speedControl() {
  /* Turn on motors */
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  /* Accelerate from zero to maximum speed */
  for (int i = 0; i < 256; i++) {
    analogWrite(enA, i);
    delay(20);
  }

  /* Decelerate from maximum speed to zero  */
  for (int i = 255; i >= 0; --i) {
    analogWrite(enA, i);
    delay(20);
  }

  /* Now turn off motors  */
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}
```

6.6 MATLAB Code

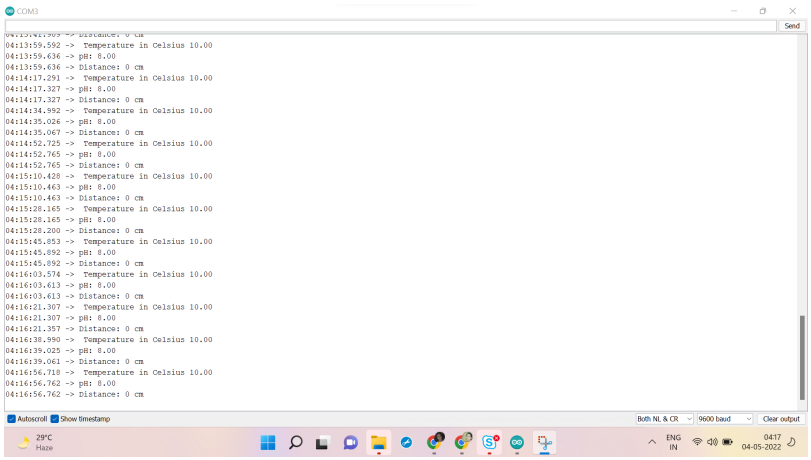
```
clc
clear all
global a;       % declaring global variable
if ~isempty(instrfind)    % finding serial port
fclose(instrfindall);
delete(instrfind);
end;
a = arduino;
interv = 200;   % declaring a for fetching data from Arduino
init_time=1;     % declaring initial time from which data collection
happens
pH= 0; % declaring pH for ANalog pH Kit
voltage = 0;
```

```
value = 0;
temp =40;          % declaring temperature in celsius for DS18B20
waterproof temperature sensor
distance = 0;   % declaring distance for HC-SR04 ultrasonic sensor
while(init_time<interv)
   b = readVoltage(a,'A0') + 8;    % reading voltage from A0 pin to
which Anaog pH kit is connected
   subplot(3,1,1)
   voltage = b*(3.3/4095.0);      % calculating pH from Voltage
   value =(3.3*voltage)+8
   pH = [pH,value];
   ph1 = plot(pH)                    % plotting pH
   set(ph1, 'Linewidth', 1);
   temp2=xlabel('Time,[s]');
   set(temp2, 'FontSize',12.5);
   temp3=ylabel('pH');
   set(temp3,'FontSize',12.5);
   ph4=title('pH vs. time');
   set(ph4,'color',[0.6 0.3 0.4],'FontSize',12);
   grid on;
   subplot(3,1,2)
    c = readDigitalPin(a,'D12');      % reading temperature from D12
pin
   temp = [temp,c];
   temp1= plot(temp);                % plotting temperature
   set(temp1, 'Linewidth', 1);
   temp2=xlabel('Time,[s]');
   set(temp2, 'FontSize',12.5);
   temp3=ylabel('Temperature in Celsius');
   set(temp3,'FontSize',12.5);
   temp4=title('Temperature vs. time');
   set(temp4,'color',[0.6 0.3 0.4],'FontSize',12);
   grid on;
    subplot(3,1,3)
   d = readDigitalPin(a,'D10');    % reading distance from D10 pin
   distance = [distance,c];
   distance1= plot(distance);        % plotting distance
   set(distance1, 'Linewidth', 1);
   distance2=xlabel('Time,[s]');
   set(distance2, 'FontSize',12.5);
   distance3=ylabel('Distance');
   set(distance3,'FontSize',12.5);
   distance4=title('Distance vs. time');
```

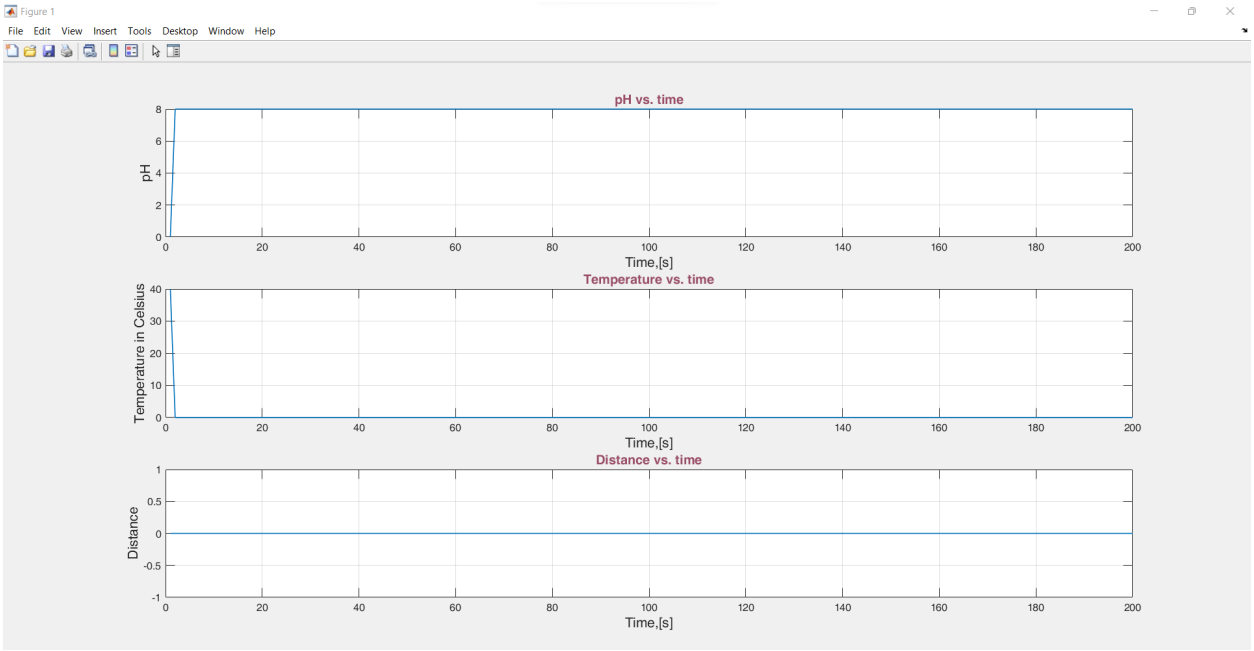
```
    set(distance4,'color',[0.6 0.3 0.4],'FontSize',12);
    grid on;
     init_time = init_time +1;
    drawnow
end
```

6.7 Output

1. Serial Monitor Output

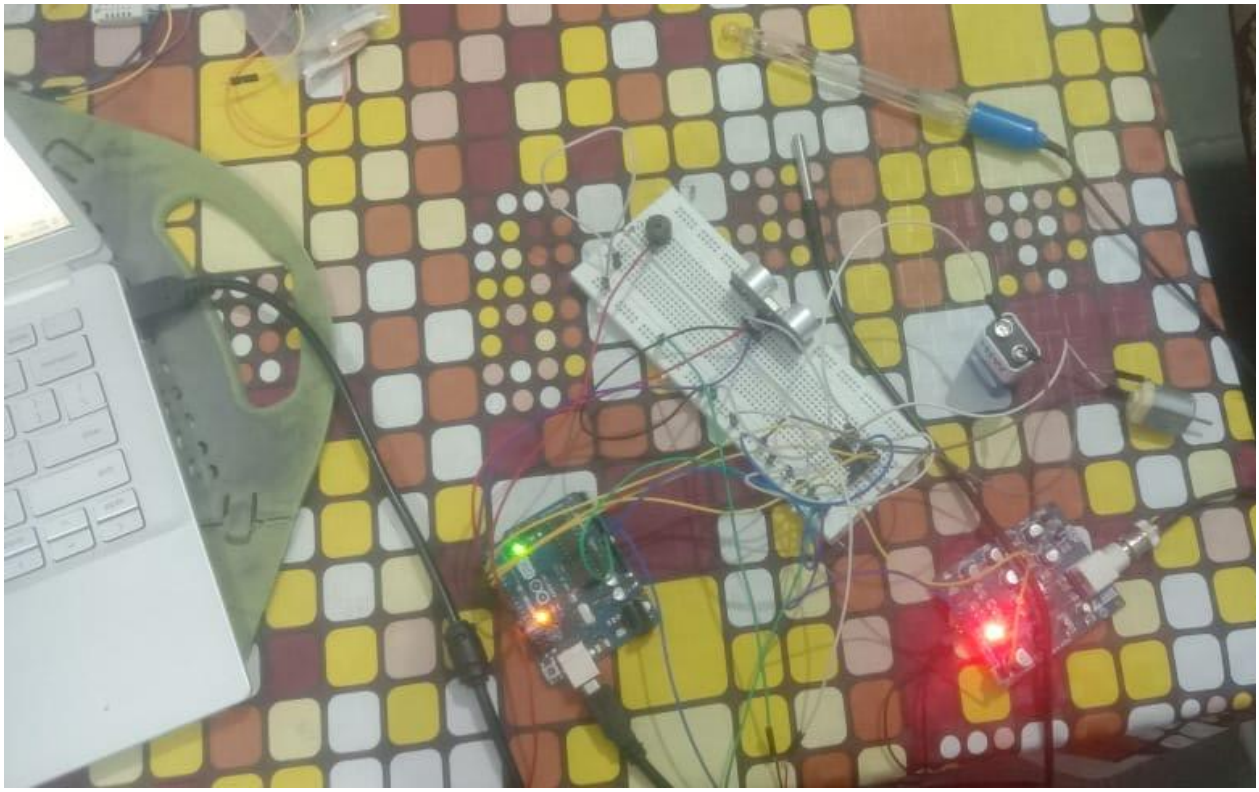


2. MATLAB



3. CoolTerm

6.8 Images



7. Future Work

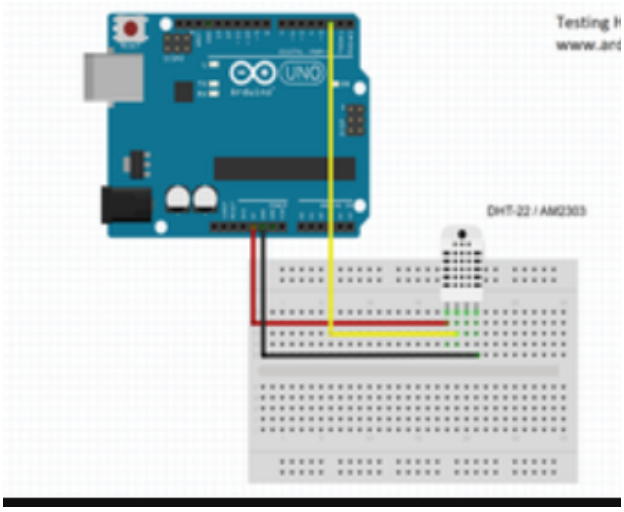
1. Addition of ES8266 WiFi Module and ThingSpeak with Arduino. Presently, the code has some errors related to getting uploaded on board and identification of ESP8266 by Arduino.
2. Ordering of the rest of the components which include modular setup, TDS and EC Sensors, Arduino Mega, Water Pumps, Seeds and Fertilizers
3. Setting up the hydroponics module. The set-up will be kept in the tissue culture facility, Department of Biological Sciences, BITS Pilani.
4. Growing the crops: Till the seed germinates and plantlets develop, the plant would be stationed in small tubs with soil as the growing medium. As the plantlet has developed, it would be transferred to the hydroponics module.
5. Integration of EC sensor and TDS Meter with existing electronics setup. Connecting motors with the water pumps.
6. Development of web app for data analytics using Android Studio/Firebase
7. Literature Review of IC Design for Internet of Farming based systems

8. Appendix

This section contains small programs which were integrated together to complete the hardware implementation above. It also contains programs to test and calibrate the sensors. Lastly, it contains some erroneous programs with ESP8266 module and NOdeMCU which will be solved and covered in future implementation of the project.

8.1 DHT22 Temperature Sensor

a. Circuit Diagram

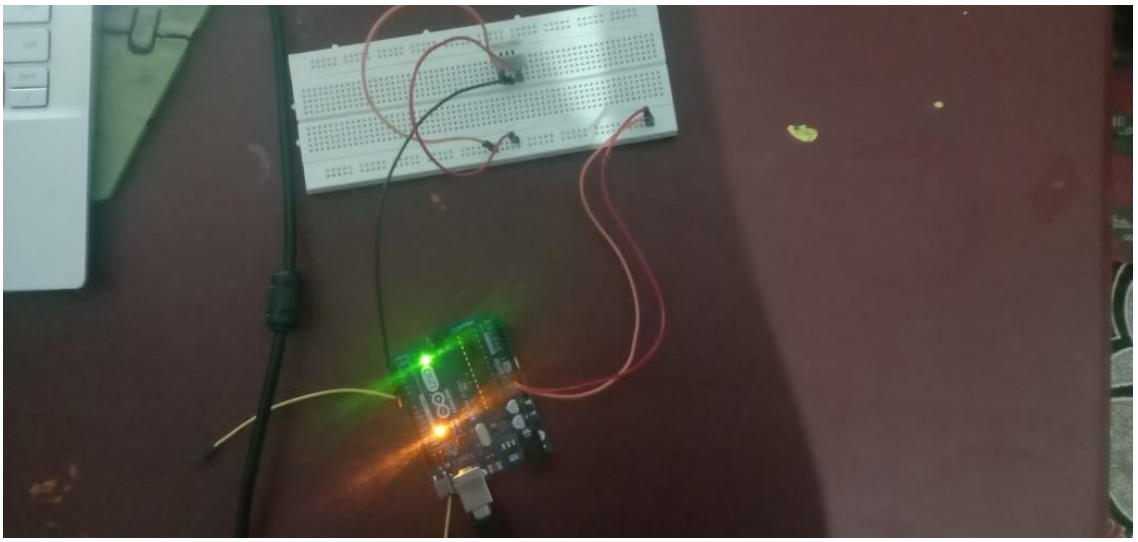


b. Code:

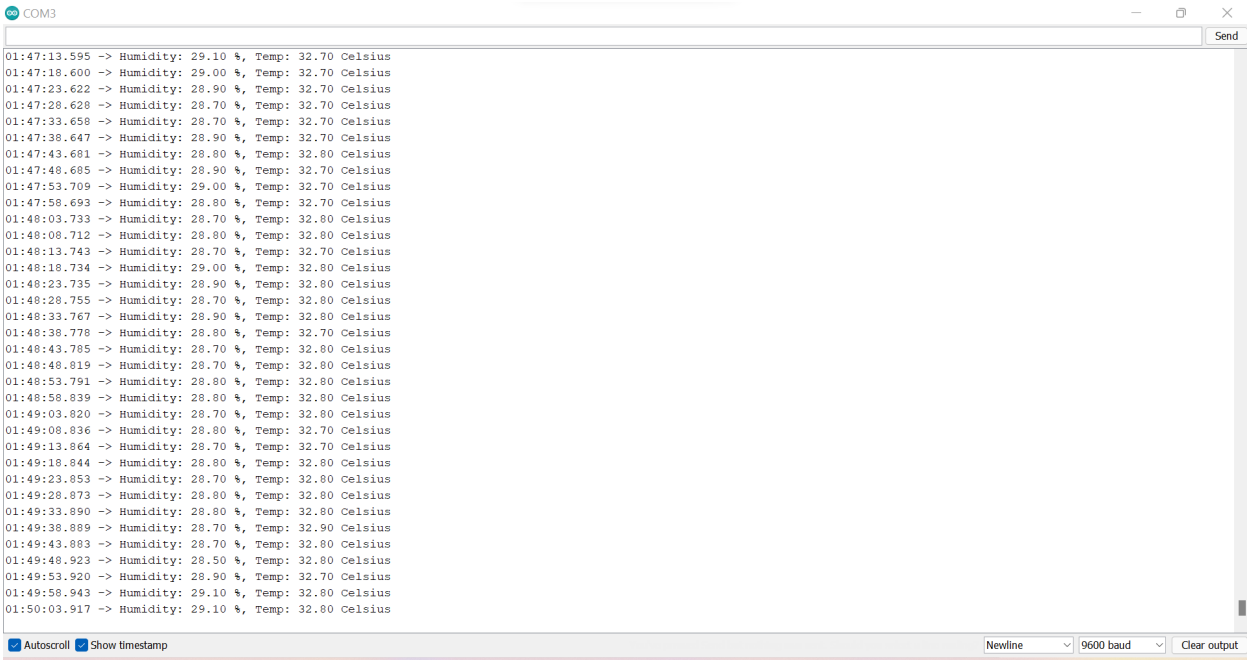
```
//Libraries
#include <DHT.h>

//Constants
#define DHTPIN 7           //what pin we're connected to
#define DHTTYPE DHT22       //DHT 22  (AM2302)
DHT dht(DHTPIN, DHTTYPE);    //Initialize DHT sensor for
normal 16mhz Arduino


//Variables
float hum;  //Stores humidity value
float temp; //Stores temperature value

void setup()
{
  Serial.begin(9600);
  dht.begin();
}
```

```
void loop()
{
  //Read data and store it to variables hum and temp
  hum = dht.readHumidity();
  temp= dht.readTemperature();

  //Print temp and humidity values to serial monitor
  Serial.print("Humidity: ");
  Serial.print(hum);
  Serial.print(" %, Temp: ");
  Serial.print(temp);
  Serial.println(" Celsius");

  delay(5000); //Delay 2 sec.
}
```

c. Implementation

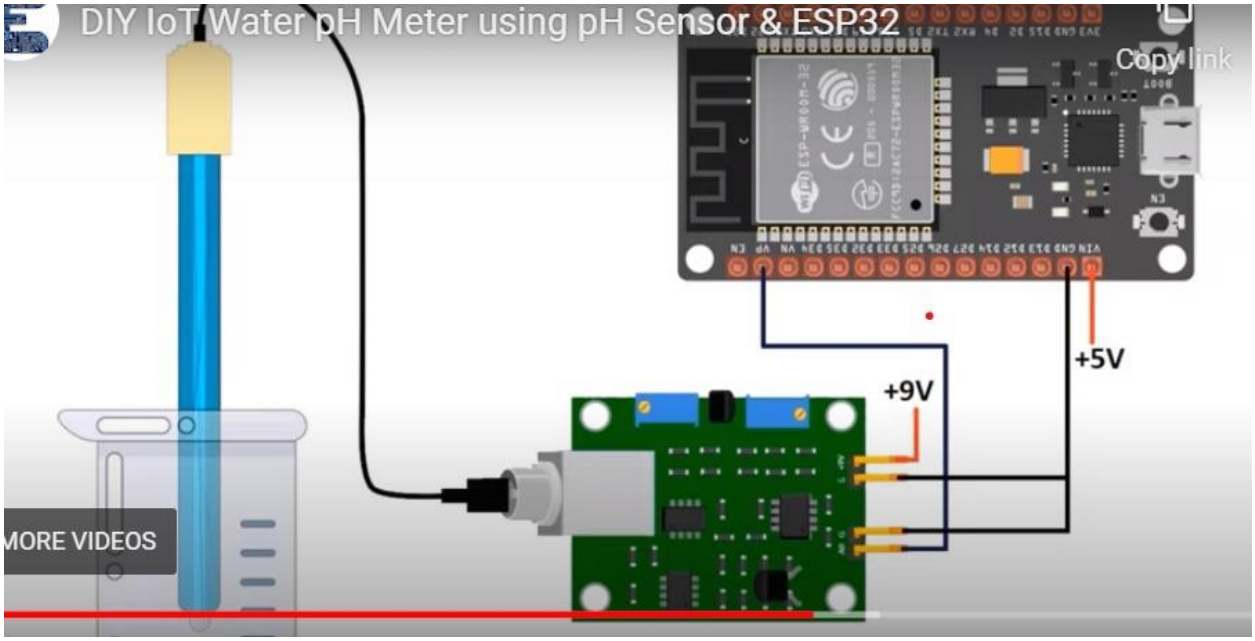


d. Output



8.2 pH electrode probe

a. Circuit Diagram

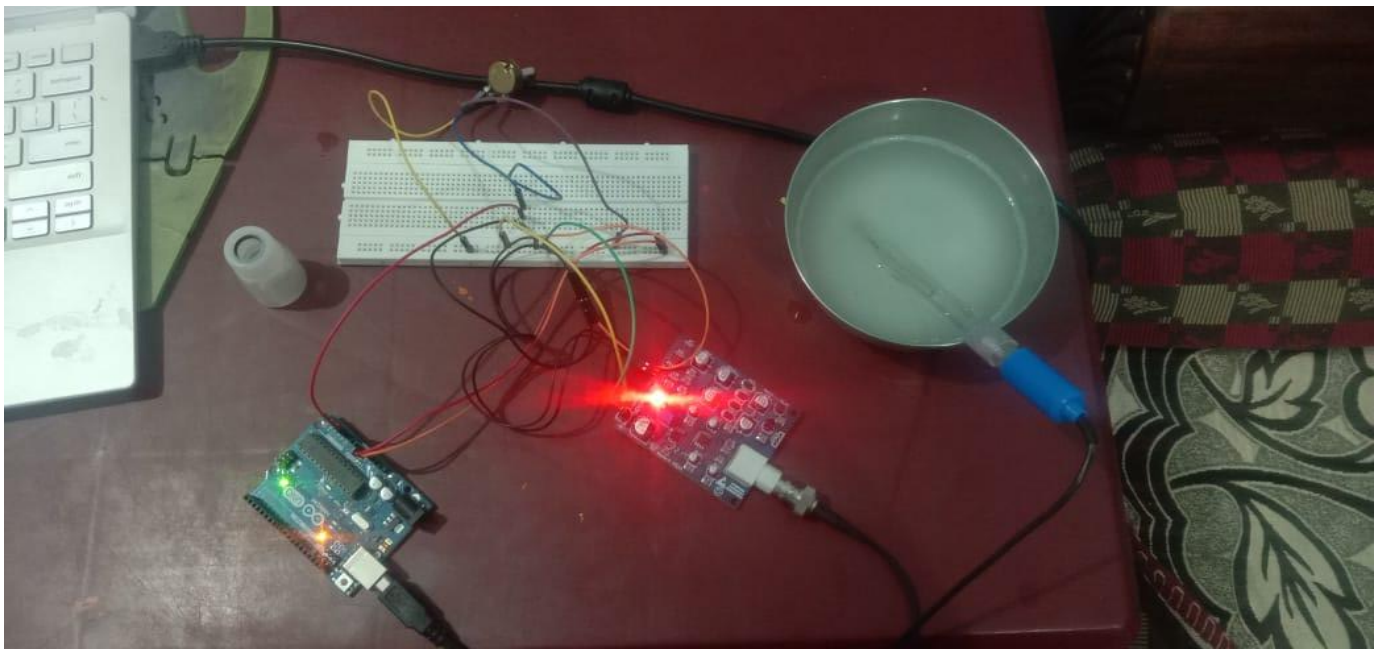


b. Code

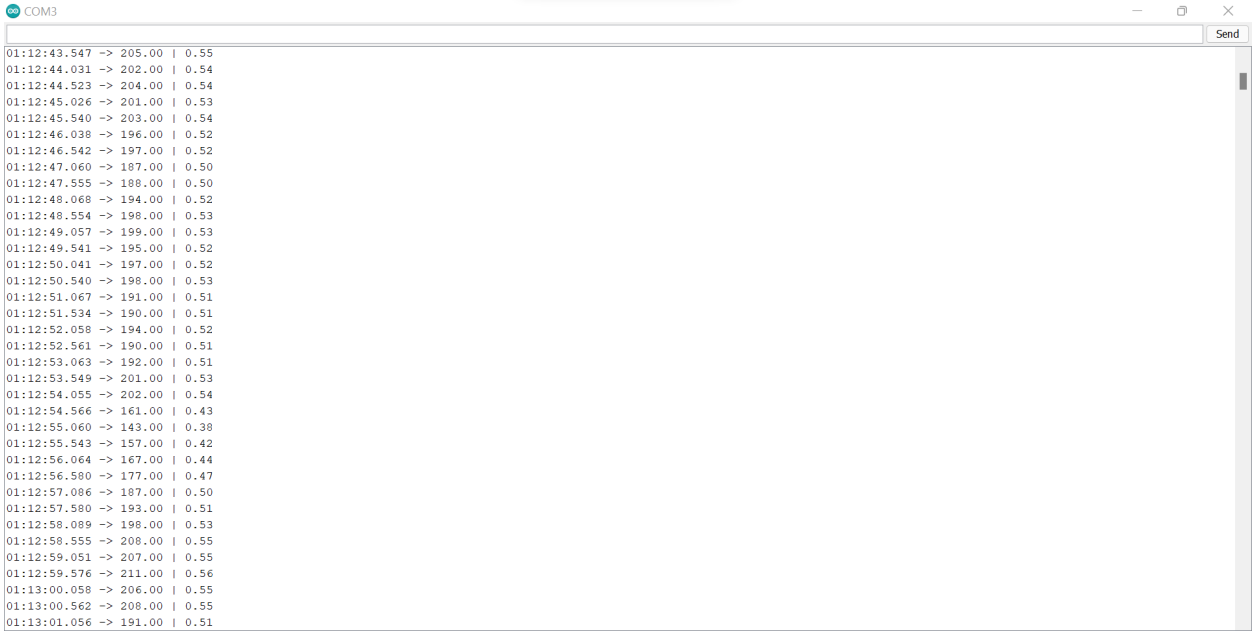
```
const int potPin=A0;
float ph;
```

```
float Value=0;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(potPin,INPUT);
  delay(1000);
}
void loop(){

    Value= analogRead(potPin);
    Serial.print(Value);
    Serial.print(" | ");
    float voltage=Value*(3.3/4095.0);
    ph=(3.3*voltage);
    Serial.println(ph);
    delay(500);
}
```

c. Hardware Implementation

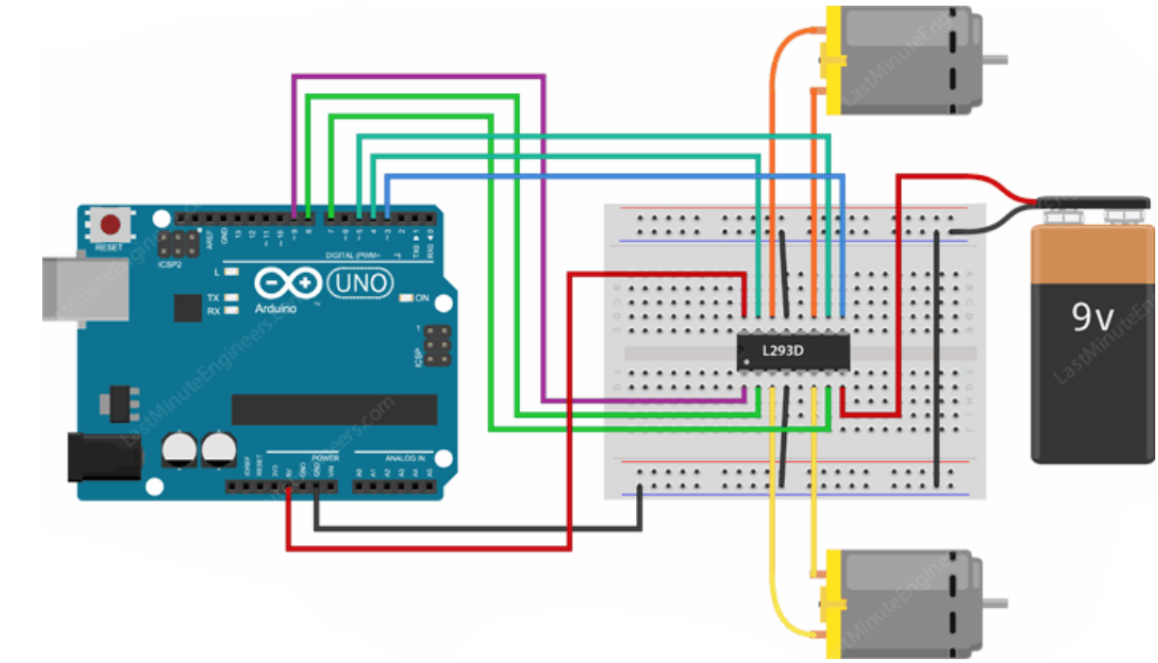


d. Output



8.3. Rotation of motor

Circuit Diagram



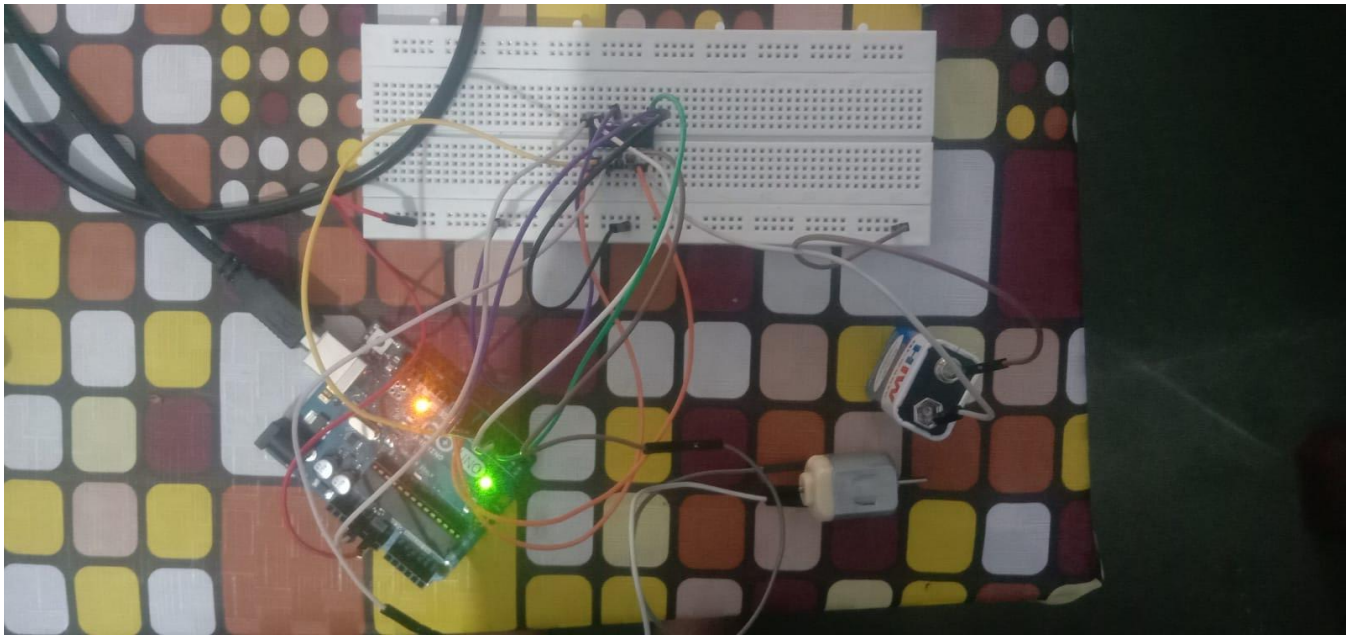
Code

```
// Motor A connections
int enA = 9;
```

```
int in1 = 8;
int in2 = 7;
// Motor B connections
int enB = 3;
int in3 = 5;
int in4 = 4;

void setup() {
      // Set all the motor control pins to outputs
      pinMode(enA, OUTPUT);
      pinMode(enB, OUTPUT);
      pinMode(in1, OUTPUT);
      pinMode(in2, OUTPUT);
      pinMode(in3, OUTPUT);
      pinMode(in4, OUTPUT);

      // Turn off motors - Initial state
      digitalWrite(in1, LOW);
      digitalWrite(in2, LOW);
      digitalWrite(in3, LOW);
      digitalWrite(in4, LOW);
}

void loop() {
      directionControl();
      delay(1000);
      speedControl();
      delay(1000);
}

// This function lets you control spinning direction of motors
void directionControl() {
      // Set motors to maximum speed
      // For PWM maximum possible values are 0 to 255
      analogWrite(enA, 255);
      analogWrite(enB, 255);

      // Turn on motor A & B
      digitalWrite(in1, HIGH);
      digitalWrite(in2, LOW);
      digitalWrite(in3, HIGH);
      digitalWrite(in4, LOW);
      delay(2000);
```

```
      // Now change motor directions
      digitalWrite(in1, LOW);
      digitalWrite(in2, HIGH);
      digitalWrite(in3, LOW);
      digitalWrite(in4, HIGH);
      delay(2000);

      // Turn off motors
      digitalWrite(in1, LOW);
      digitalWrite(in2, LOW);
      digitalWrite(in3, LOW);
      digitalWrite(in4, LOW);
}

// This function lets you control speed of the motors
void speedControl() {
      // Turn on motors
      digitalWrite(in1, LOW);
      digitalWrite(in2, HIGH);
      digitalWrite(in3, LOW);
      digitalWrite(in4, HIGH);

      // Accelerate from zero to maximum speed
      for (int i = 0; i < 256; i++) {
            analogWrite(enA, i);
            analogWrite(enB, i);
            delay(20);
      }

      // Decelerate from maximum speed to zero
      for (int i = 255; i >= 0; --i) {
            analogWrite(enA, i);
            analogWrite(enB, i);
            delay(20);
      }

      // Now turn off motors
      digitalWrite(in1, LOW);
      digitalWrite(in2, LOW);
      digitalWrite(in3, LOW);
      digitalWrite(in4, LOW);
}
```

Hardware Implementation



8.4. LCD Printing pH and Temperature

Code:

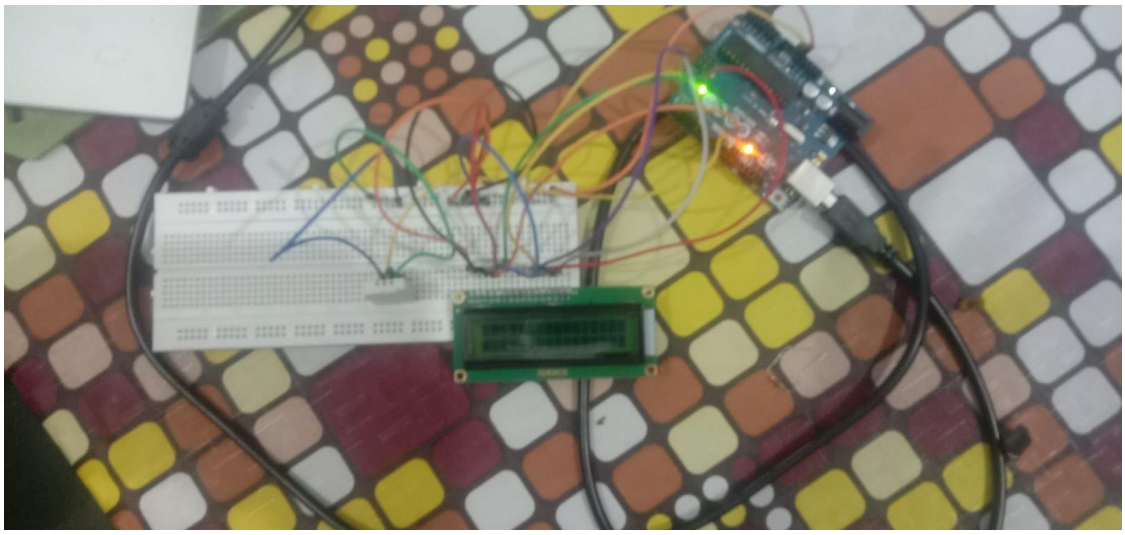
```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

/* Temperature humidity sensor */
#include <DHT.h>
#define datapin 7     // Digital pin we're connected to
#define DHTTYPE DHT22
DHT dht(datapin, DHTTYPE);

 float air_temp;
 float humidity;
 const int potPin=A0;
 float ph;
 float Value=0;
 float pHMin = 1.5; // the temperature to start the buzzer
 int pHMax = 2;
 /* Motor A connections */
 int enA = 10;
```

```
 int in1 = 8;
 int in2 = 7;

void setup() {
  Serial.begin(9600);
  dht.begin();
  lcd.begin(16, 2);
  pinMode(potPin,INPUT);
  // Set all the motor control pins to outputs

}

void loop() {

  /* Find Temperature & Humidity */
 air_temp = dht.readTemperature();
 humidity = dht.readHumidity();
  Value= analogRead(potPin);
  float voltage=Value*(3.3/4095.0);
    ph=(3.3*voltage);

  /* Controlling motor rotation */
  if (ph <=pHMax)
  {
    directionControl();
    delay(1000);
   speedControl();
   delay(1000);
  }
  /* Print Output on LCD Screen */
  lcd.setCursor(0,0);
  lcd.print(String("Temp. ") + String(air_temp));
  lcd.setCursor(0,1);
  lcd.print(String("Humidity ") + String(humidity));
  lcd.print(String("pH: ")+String(ph));
  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.print(" %, Temp: ");
  Serial.print(air_temp);
  Serial.println(" Celsius");
  Serial.println("pH: ");
  Serial.println(ph);
```

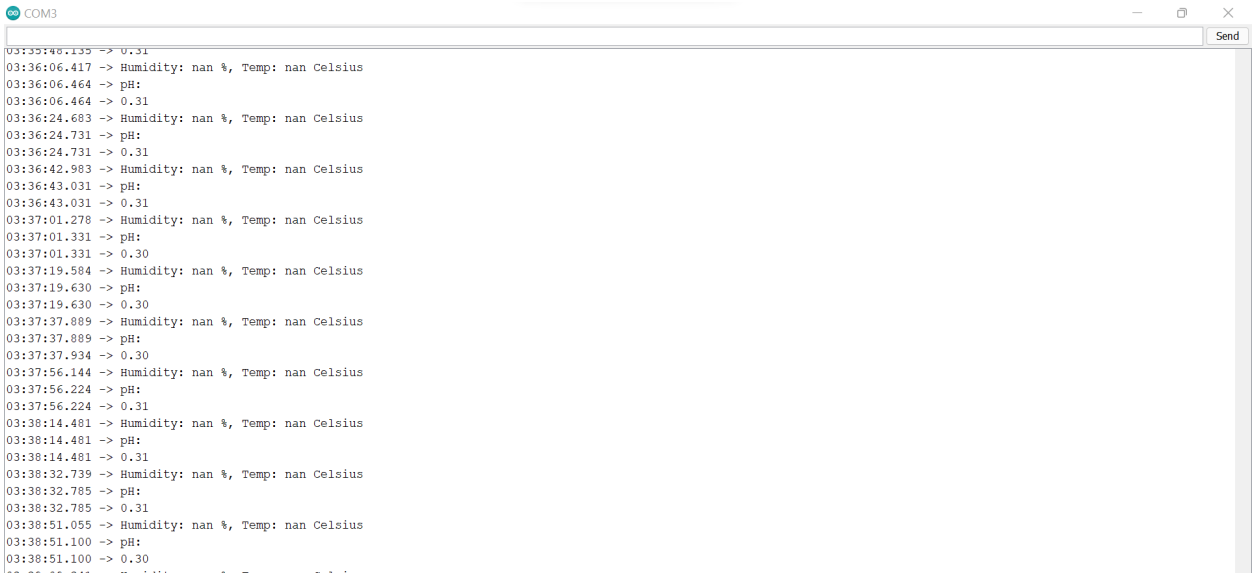
```
  delay(2000);

}

// This function lets you control spinning direction of motors
void directionControl() {
  // Set motors to maximum speed
  // For PWM maximum possible values are 0 to 255
  analogWrite(enA, 255);


  // Turn on motor A
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  delay(2000);

  // Now change motor directions
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  delay(2000);

  // Turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}

// This function lets you control speed of the motors
void speedControl() {
  // Turn on motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);


  // Accelerate from zero to maximum speed
  for (int i = 0; i < 256; i++) {
    analogWrite(enA, i);
    delay(20);
  }

  // Decelerate from maximum speed to zero
  for (int i = 255; i >= 0; --i) {
    analogWrite(enA, i);
    delay(20);
```

```
  }

  // Now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}
```

Hardware Implementation:

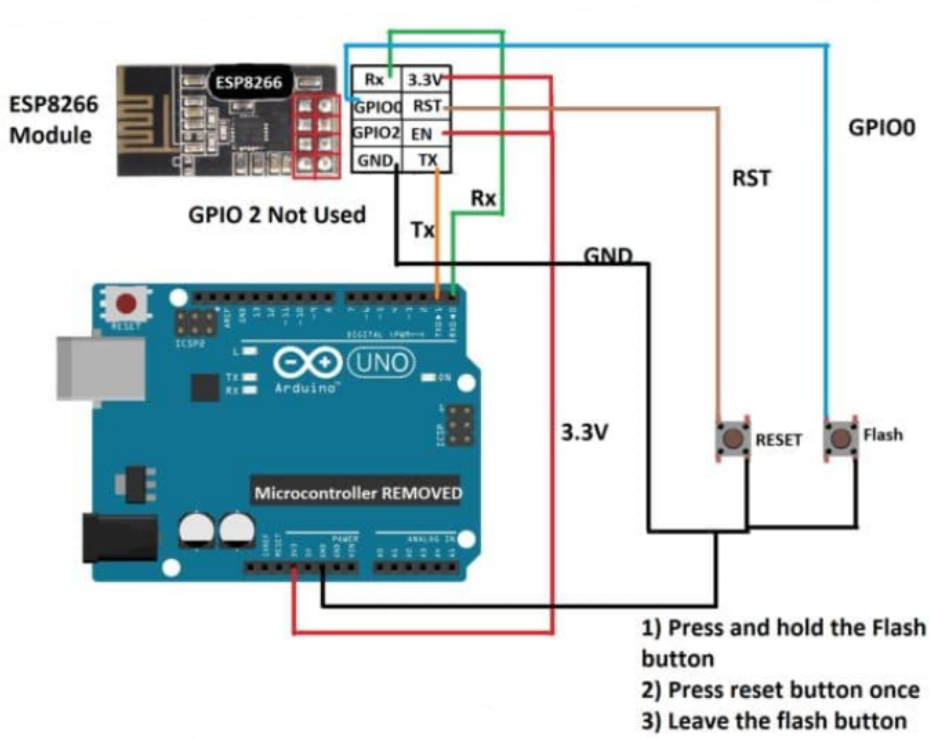


Output:



8.5. Sending DHT-22 sensor data to ThingSpeak using WiFi ESP8266

a. Circuit



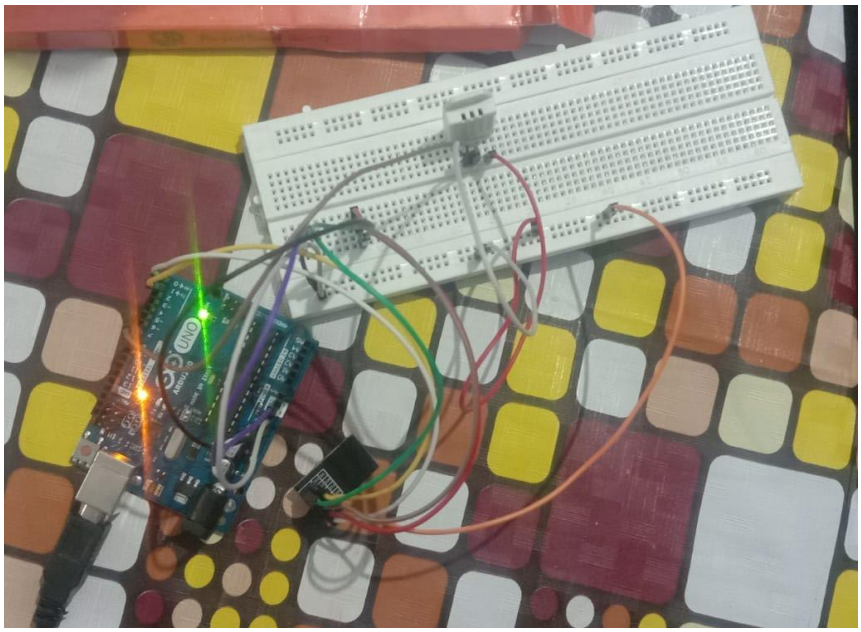
b. Code

```
#include "ThingSpeak.h"
#include <ESP8266WiFi.h>

//------- WI-FI details ----------//
char ssid[] = "xxxxxxxxx"; //SSID here
char pass[] = "yyyyyyyyy"; // Passowrd here
//--------------------------------//

//----------- Channel details ----------------//
unsigned long Channel_ID = 12345; // Your Channel ID
const char * myWriteAPIKey = "ABCDEF12345"; //Your write API key
//--------------------------------------------//

const int Field_Number_1 = 1;
const int Field_Number_2 = 2;
```

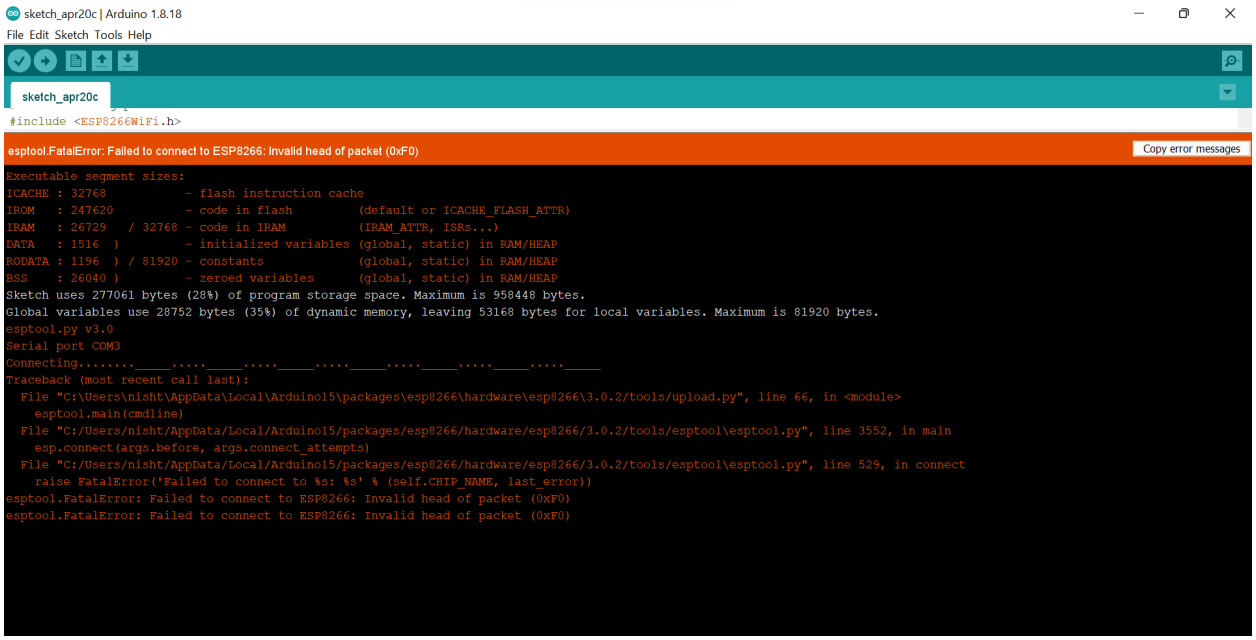
```
String value = "";
int value_1 = 0, value_2 = 0;
int x, y;
WiFiClient  client;

void setup()
{
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);
  internet();
}

void loop()
{
  internet();
  if (Serial.available() > 0)
  {
    delay(100);
    while (Serial.available() > 0)
    {
      value = Serial.readString();
      if (value[0] == '*')
      {
        if (value[5] == '#')
        {
             value_1 = ((value[1] - 0x30) * 10 + (value[2] -
0x30));
             value_2 = ((value[3] - 0x30) * 10 + (value[4] -
0x30));
        }
      }
    }
  }
  upload();
}

void internet()
{
  if (WiFi.status() != WL_CONNECTED)
  {
    while (WiFi.status() != WL_CONNECTED)
    {
```

```
      WiFi.begin(ssid, pass);
      delay(5000);
    }
  }
}

void upload()
{
   ThingSpeak.writeField(Channel_ID,  Field_Number_1,  value_1,
myWriteAPIKey);
  delay(15000);
   ThingSpeak.writeField(Channel_ID,  Field_Number_2,  value_2,
myWriteAPIKey);
  delay(15000);
  value =

}
```

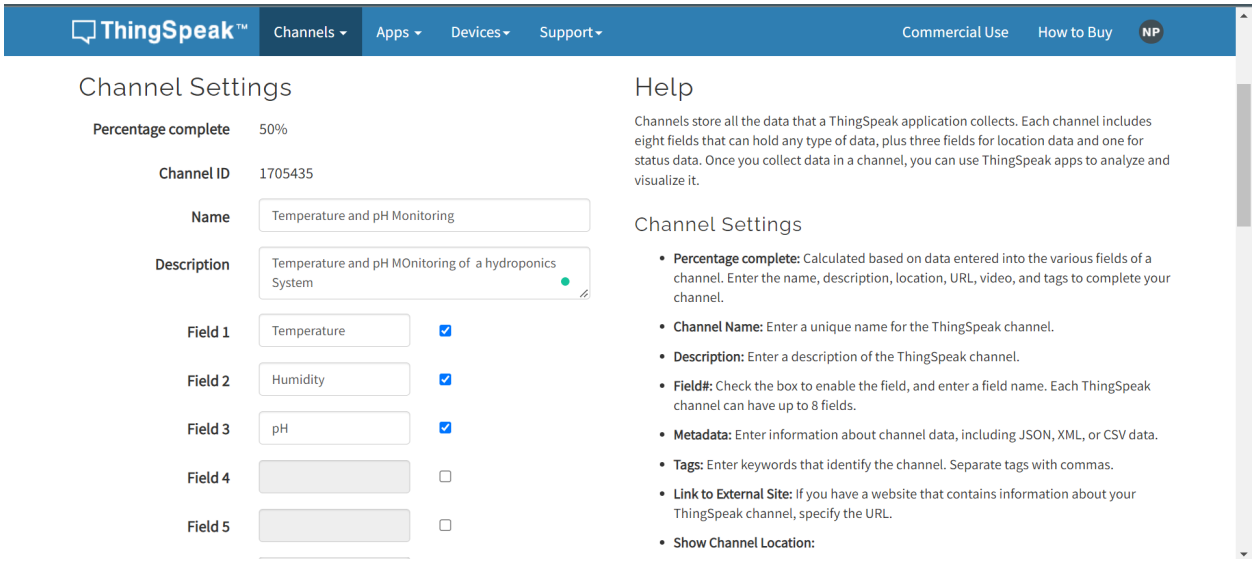
c. Hardware Implementation

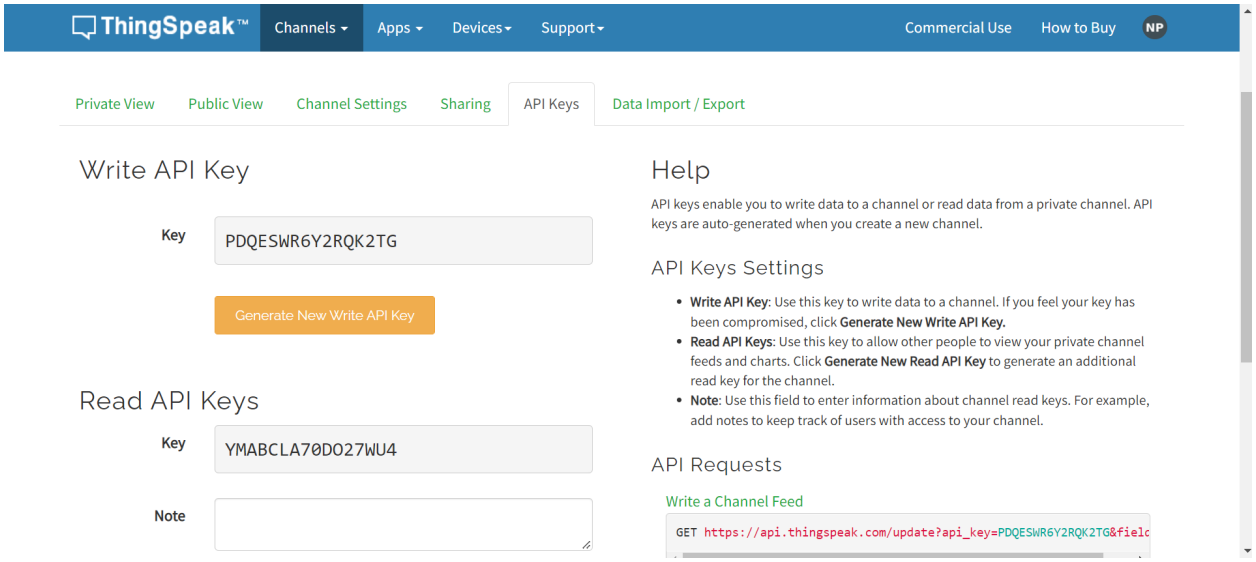


d. Output



e. ThingSpeak





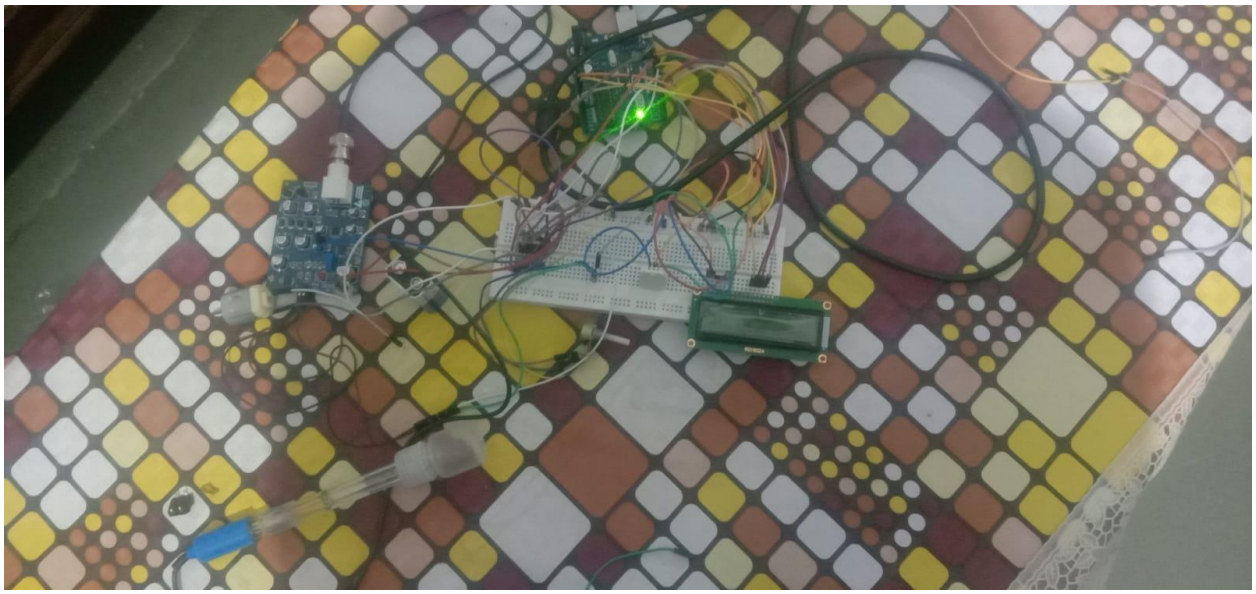
8.6. Integration (Hydroponics System with motor control for fans and pumps)

Code:

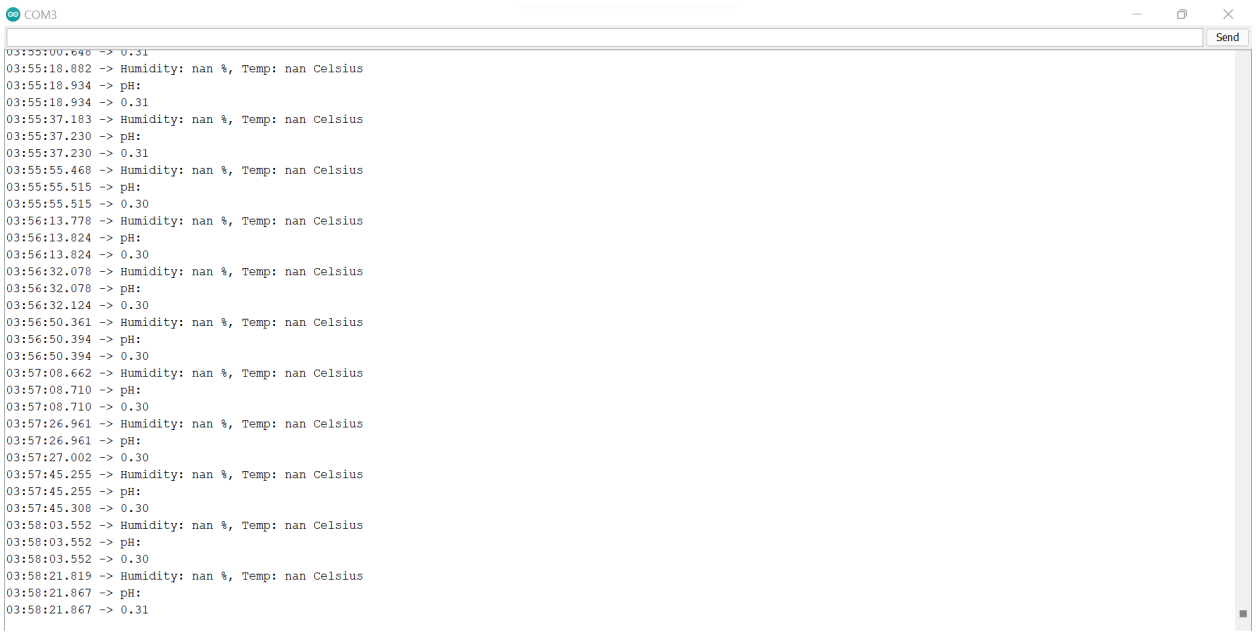
```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

/* Temperature humidity sensor */
#include <DHT.h>
#define datapin 7     // Digital pin we're connected to
#define DHTTYPE DHT22
DHT dht(datapin, DHTTYPE);

 float air_temp;
 float humidity;
 const int potPin=A0;
 float ph;
 float Value=0;
 float pHMin = 1.5; // the temperature to start the buzzer
 int pHMax = 2;
 /* Motor A connections */
 int enA = 10;
 int in1 = 8;
 int in2 = 7;

void setup() {
  Serial.begin(9600);
```

```
  dht.begin();
  lcd.begin(16, 2);
  pinMode(potPin,INPUT);
  // Set all the motor control pins to outputs

}

void loop() {

  /* Find Temperature & Humidity */
 air_temp = dht.readTemperature();
 humidity = dht.readHumidity();
  Value= analogRead(potPin);
  float voltage=Value*(3.3/4095.0);
    ph=(3.3*voltage);

  /* Controlling motor rotation */
  if (ph <=pHMax)
  {
    directionControl();
    delay(1000);
   speedControl();
   delay(1000);
  }
  /* Print Output on LCD Screen */
  lcd.setCursor(0,0);
  lcd.print(String("Temp. ") + String(air_temp));
  lcd.setCursor(0,1);
  lcd.print(String("Humidity ") + String(humidity));
  lcd.print(String("pH: ")+String(ph));
  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.print(" %, Temp: ");
  Serial.print(air_temp);
  Serial.println(" Celsius");
  Serial.println("pH: ");
  Serial.println(ph);

  delay(2000);

}

// This function lets you control spinning direction of motors
void directionControl() {
  // Set motors to maximum speed
```

```
  // For PWM maximum possible values are 0 to 255
  analogWrite(enA, 255);


  // Turn on motor A
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  delay(2000);

  // Now change motor directions
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  delay(2000);

  // Turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}

// This function lets you control speed of the motors
void speedControl() {
  // Turn on motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);


  // Accelerate from zero to maximum speed
  for (int i = 0; i < 256; i++) {
    analogWrite(enA, i);
    delay(20);
  }

  // Decelerate from maximum speed to zero
  for (int i = 255; i >= 0; --i) {
    analogWrite(enA, i);
    delay(20);
  }

  // Now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}
```



Serial Monitor Output



References:

1. M. C. Moreno, O. J. Suarez and A. P. Garcia, "IoT-based Automated Greenhouse for Deep Water Culture Hydroponic System," *2021 2nd Sustainable Cities Latin*

America Conference (SCLA), 2021, pp. 1-6, doi: 10.1109/SCLA53004.2021.9540187.

2. Sihombing, P., Karina, N.A., Tarigan, J.T., & Syarif, M.I. (2018). Automated hydroponics nutrition plants systems using arduino uno microcontroller based on android.
3. Chowdhury, M.E.H.; Khandakar, A.; Ahmed, S.; Al-Khuzaei, F.; Hamdalla, J.; Haque, F.; Reaz, M.B.I.; Al Shafei, A.; Al-Emadi, N. Design, Construction and Testing of IoT Based Automated Indoor Vertical Hydroponics Farming Test-Bed in Qatar. *Sensors* 2020, *20*, 5637. https://doi.org/10.3390/s20195637
4. Lakshmiprabha, KE, Govindaraju, C. Hydroponic-based smart irrigation system using Internet of Things. *Int J Commun Syst*. 2019;e4071. https://doi.org/10.1002/dac.4071
5. Manav Mehra, Sameer Saxena, Suresh Sankaranarayanan, Rijo Jackson Tom, M. Veeramanikandan,
6. IoT based hydroponics system using Deep Neural Networks, Computers and Electronics in Agriculture, https://doi.org/10.1016/j.compag.2018.10.015.
7. https://in.mathworks.com/matlabcentral/answers/326800-plotting-data-from-arduino-with-multiples-sensors-analogs-and-digital-in-real-time
8. https://forum.arduino.cc/t/arduino-matlab-dht22-sensor-how-to-plot-the-graph/386654
9. https://create.arduino.cc/projecthub/highvoltages/arduino-real-time-plotting-with-matlab-f8985b
10. https://create.arduino.cc/projecthub/hbolanos2001/wifi-esp8266-and-dht22-sensor-09d455
11. https://www.instructables.com/ESP8266-IOT-Using-Arduino-and-ThingSpeak/
12. https://randomnerdtutorials.com/esp8266-nodemcu-thingspeak-publish-arduino/
13. https://create.arduino.cc/projecthub/neveroffttheinternet/thingspeak-arduino-weather-station-70b4bb
14. https://electronics-project-hub.com/send-data-to-thingspeak-using-esp8266/
15. https://thingspeak.com/channels/public?page=94&tag=esp8266
16. https://randomnerdtutorials.com/guide-for-ds18b20-temperature-sensor-with-arduino/
17. https://create.arduino.cc/projecthub/iotboys/how-to-use-ds18b20-water-proof-temperature-sensor-2adecc
18. https://circuitdigest.com/microcontroller-projects/arduino-ph-meter
19. https://create.arduino.cc/projecthub/abshah15/dht22-sensor-with-16x2-lcd-screen-59a6e9
20. https://create.arduino.cc/projecthub/mafzal/temperature-monitoring-with-dht22-arduino-15b013

21. https://how2electronics.com/diy-iot-water-ph-meter-using-ph-sensor-esp32/#Basic_Test_Code
22. https://lastminuteengineers.com/l293d-dc-motor-arduino-tutorial/
23. https://create.arduino.cc/projecthub/abdularbi17/ultrasonic-sensor-hc-sr04-with-arduino-tutorial-327ff6
24. https://how2electronics.com/diy-iot-water-ph-meter-using-ph-sensor-esp32/#Basic_Test_Code
25. https://forum.arduino.cc/t/controlling-a-dc-motor-with-ultrasonic-sensor/226563
26. https://docs.arduino.cc/learn/electronics/lcd-displays
27. https://create.arduino.cc/projecthub/ryujenny3/servo-motor-ultrasonic-sensor-f951fe
28. https://learn.sparkfun.com/tutorials/terminal-basics/coolterm-windows-mac-linux
29. https://create.arduino.cc/projecthub/luisantoniomartinnuez/arduino-controlled-smart-hydroponic-modular-system-0d65ad
30. https://www.instructables.com/Arduino-Controlled-Smart-Hydroponic-Modular-System/
31. https://www.hackster.io/Sulton/monitoring-of-hydroponic-with-iot-system-b81f84
32. https://www.almanac.com/plant/lettuce
33. https://whyfarmit.com/hydroponic-lettuce/
34. https://www.treehugger.com/how-to-grow-hydroponic-lettuce-5195338
35. https://www.wikihow.com/Grow-Hydroponic-Lettuce
36. https://risehydroponics.in/how-to-grow-lettuce-hydroponically/