**IoT-Based Smart Farming System**

Submitted in partial fulfilment of the requirement for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**ELECTRONICS & COMMUNICATIONS ENGINEERING**

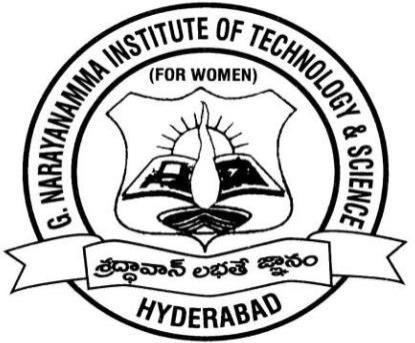
Submitted By

**SAREDDI SUJITHA (21251A0455)**

Under the esteemed guidance of

**Dr. M.Vijayalakshmi,**

**Associate Professor**, ECE, GNITS (FOR WOMEN)



**DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGG G. NARAYANAMMA INSTITUTE OF TECHNOLOGY & SCIENCE**

(for Women)

**(AUTONOMOUS)**

**(Accredited by NBA & NAAC)**

2024-25

**DEPARTMENT OF**

**ELECTRONICS & COMMUNICATION ENGINEERING**

**CERTIFICATE**

THIS IS TO CERTIFY THAT THE MINI PROJECT ENTITLED

**IoT-Based Smart Farming System**

IS THE BONAFIDE WORK OF

|  |
| --- |
|  |
| SAREDDI SUJITHA | | (21251A0455) |

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE

AWARD OF DEGREE OF BACHELOR OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING DURING THE YEAR 2023-24

Dr K Ragini GUIDE

PROF & HOD, ECE Dr. M, VIJAYALAKSHMI

ASSOC. PROF,ECE

**ACKNOWLEDGEMENTS**

We are extremely thankful and indebted to our guide **Dr.M.Vijayalakshm**i Associate Professor**, Department of ECE,** GNITS for her constant guidance, encouragement, and moral support throughout the project.

We are extremely thankful to **M.Shanthi**, Assistant Professor, ECE department

GNITS. Mini Project coordinator for her encouragement and support throughout the project

Our sincere thanks and gratitude to **Dr K Ragini**, Professor & Head, Department of ECE, GNITS for all the timely support and valuable suggestions during the period of our project.

We would like to express our sincere thanks to **Dr K. Ramesh Reddy**, Principal, GNITS, for providing the working facilities in the college.

Finally, we are extremely thankful to all the faculty members and staff of ECE Department who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

|  |  |
| --- | --- |
| **SAREDDI SUJITHA** | **(21251A0455)** |
|  |  |

**CONTENTS**

**S.No. Topic Page no.**

[Abstract I](#_Toc79926)

[List of Figures II](#_Toc79927)

[List of Tables IV](#_Toc79928)

### 1 Introduction

1.1 Introduction 1

1.2 Problem statement 1

1.3 Objective 2

1.4 Scope of project 2

1.5 Application areas 3

1.6 About the project 3

### 2. Literature Review and Background

2.1 Literature survey 4

2.2 Background 4

### 3. System Design and Implementation details

3.1 System design 6

3.2 Block diagram 6

3.3 Requirements 7

3.4 Methodology 7

3.5 Hardware

3.5.1 Physical earth condition 7

3.5.2 Temperature and Humidity level 8

3.5.3 Hardware components used 9

3.5.4 Temperature and humidity sensor 10

3.5.5 Soil moisture sensor 11

3.5.6 Water level sensor 13

3.5.7 Relay 15

3.5.8 Water pump 16

3.5.9 Arduino USB cable 17

3.5.10 Node MCUES8266 18

3.5.11 Male and Female wires 19

3.5.12 Breadboard 20

3.5.13 Multiplexer 21

3.5.14 Rainwater sensor 22

3.7.1 Arduino IDE 23

3.7.2 Thingspeak 26

3.6 Simulation details

3.6.1 Simulation of physical data 29

### 4. Code and Results

4.1 Testing and validation

4.1.1 Measures Temperature and Humidity

level through sensor DHT11 30

4.1.2 Measure water level through sensor water level 31

4.1.3 soil moisture sensor 32

4.1.4 Rainwater 34

4.2 Results 35

4.3 Code 41

### 5. Conclusion and Future recommendation

5.1 Conclusion 50

5.2 Future recommendation 50

5.3 References 51

# ABSTRACT

India is an agricultural country where the agriculture sector is the backbone of the economy, contributing to more than 20% of the country's GDP. However, the agriculture sector is facing numerous challenges such as water scarcity, climate change, and low productivity due to outdated farming practices. Therefore, there is a dire need to introduce modern technology in the agriculture sector to enhance its productivity and efficiency. This project presents an IOT Based Smart Agriculture Monitoring System aimed at increasing agricultural productivity by automating and optimizing crop management. The system uses various sensors to monitor environmental conditions in real-time. The data collected is processed by a microcontroller and transmitted wirelessly to a web application that provides farmers with visualized information about their crops. This, in turn, can lead to increased crop yields, reduced costs, and improved profitability. The project also has future implications, including the integration of machine learning and artificial intelligence technologies to further optimize crop management. With the increasing demand for food production and the need to address the challenges of climate change and food security, this project serves as a promising solution for sustainable agriculture.

## LIST OF FIGURES

**S.NO FIG.NO TITLE PAGE.NO**

1. 2.1 Old agriculture system 5
2. 2.2 Smart agriculture monitoring system 6
3. 3.1 Block diagram 7
4. 3.2 Soil moisture 8
5. 3.3 Temperature and Humidity 9
6. 3.4 Project with NODE MCUES8266 10
7. 3.5 Temperature and Humidity sensor 11
8. 3.6 Soil moisture sensor 13
9. 3.7 Water level sensor 15
10. 3.8 Relay 17
11. 3.9 Water pump 19
12. 3.10 USB cables 19
13. 3.11 NODE MCUES8266 21
14. 3.12 Male and Female wires 22
15. 3.13 Breadboard 23
16. 3.14 Analog Multiplexer 24
17. 3.15 Rainwater sensor 25
18. 3.16 Arduino IDE 26
19. 3.17 Add library menu 27
20. 3.18 Library manger menu 27
21. 3.19 Zip manger menu 28
22. 3.20 Thingspeak website 29
23. 3.21 Unfilled field on my channel 30
24. 3.22 Filled field channel 31
25. 3.23 API key 31

1. 3.24 API key channel 31
2. 3.25 API requested link 31
3. 3.26 Proteus simulation of sensors 33
4. 3.27 Email interface 34
5. 3.28 Password interface 34
6. 4.1 DHT11 sensor integrated with NODE MCU ES 35

8266

1. 4.2 DHT11 sensor reading serial monitor 35
2. 4.3 Waterlevel connected with NODE MCU ES8266 36
3. 4.4 Waterlevel readings on serial motor 37
4. 4.5 Soil moisture sensor connected with NODE MCU 37 ES8266
5. 4.6 Soil moisture readings on serial monitor 38
6. 4.7 Rainwater connected NodeMCU ES8266 38
7. 4.8 RainWater sensor Reading serial monitor 39
8. 4.9 Soil moisture,waterlevel and rainwater readings 42 in thingspeak
9. 4.10 All sensors readings on telegram 45

## LIST OF TABLES

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.NO TABLE NO.** | | |  | **TITLE** | **PAGE NO.** | |
| 1 |  | 3.1 |  | Pin configuration of DHT11 |  | 12 |
| 2 |  | 3.2 |  | Pin configuratuon of soil moisture |  | 14 |
| 3 |  | 3.3 |  | Pin configuration of waterlevel sensor |  | 15 |
| 4 |  | 3.4 |  | Pin configuration of relay |  | 17 |
| 5 |  | 3.5 |  | Pin configuration of multiplexer |  | 24 |
| 6 |  | 3.6 |  | Pin configuration Rainwater sensor | | 25 |

**1. INTRODUCTION AND PROBLEM STATEMENT**

### 1.1. INTRODUCTION

Agriculture is the backbone of many economies around the world, and its modernization is essential to meet the growing demands for food and raw materials. Traditional farming practices often rely on the farmer's experience and manual labor, which can lead to inefficiencies and increased workload. The advent of the Internet of Things (IoT) offers a transformative approach to agriculture by integrating advanced technology to monitor and manage farming activities more efficiently.

An IoT-based agriculture control system aims to revolutionize farming by leveraging real-time data and automated systems to enhance productivity and sustainability. This system utilizes a network of sensors to monitor various environmental parameters such as soil moisture, temperature, humidity, and light intensity. These sensors collect critical data, which is then processed and transmitted to a central cloud platform for analysis.

To make this system more interactive and user-friendly, a Telegram bot can be integrated to provide farmers with real-time updates and control over their agricultural operations. Telegram, a widely used messaging app, offers a robust API that can be utilized to create a bot capable of sending alerts, receiving commands, and providing insights based on the sensor data.

**1.2. Problem Statement**

To monitor agricultural variables through NODE MCU ESP8266 module based real time IoT for improving yield through use of multiple sensors. There are many main Problems related to Agriculture. Pakistan is under developed country that’s why Pakistan face many problems and these problems is increasing day by day. There are many problems we faced in field agriculture when we talk about crops. There are many types of toxic gases flow in the air. These effect on the health of crop due to these toxic gases crop growth is slowly and also grow sick due to these gases we cannot take good quality of food from crops. Also more light effect on the health of crops because crops need moderate level of light for growth. This light helps in the process of photosynthesis. Those areas where water available in huge amount farmer irrigate the crops more than sufficient amount of water to sure that’s crop received the required amount of water due to more water this water bad effect on the health of crops. Some area is hot and some areas cold. Pakistan is the country where having the four season. When season change temperature will also change. Temperature is important and deciding factor for crops. When crops will ready to

1

grow and produce food. We will make the system. In this system we will be measure different variables according to above defined different Parameters.

### 1.3. Objectives

The Objectives of an IoT based Smart Farming System is to use advanced technology to monitor and manage various environmental conditions affecting crops, such as soil moisture, temperature, light intensity, and air quality, among others. This system aims to provide farmers with real-time data and analytics to help them optimize crop yield, reduce water usage, lower costs, and enhance sustainability.

The specific objectives of an IoT based Smart Farming System may include:

* To decrease the amount of manual labour.

* To Developing an integrated system of sensors and actuators that can collect and transmit real-time data about soil moisture, temperature, light intensity, air quality, and other relevant factors.
* To take the update about the crop behaviour from anywhere when climate suddenly vary.
* To take action against toxic gases though fertilizers and antioxidant.
* To Providing farmers with access to this data through a user-friendly interface, allowing them to monitor crop conditions, analyse data trends, and make informed decisions about crop management.
* To Lowering costs by reducing waste and improving efficiency in crop management.
* To monitor the Temperature and Humidity level of the crops in the field.

### 1.4. Scope of Project

The scope of an IoT based smart agriculture monitoring system project can include several aspects such as hardware and software development, system design, testing, and validation, among others. The specific scope of the project may depend on the objectives, requirements, and resources available.

Some potential areas of scope for an IoT Based Smart Agriculture Monitoring System project could include:

* With the help of sensors, the labour need will decrease.
* The sensor allows precisely measure the moisture level of soil.
* The gas sensor will detect different toxic gases. With the help of gas sensor farmers and labours will take action on accurate time to save crops from bad growth.
* The temperature sensor detects the temperature level precisely and accurate than the manual process.
* This may involve using statistical methods and other analytical tools to process the data collected by the system and generate useful insights
* This may involve creating a user-friendly interface that allows farmers to easily access and interpret the data collected by the system.

### 1.5. Application Areas

Those places where temperature suddenly vary with the help of this we can monitor temperature easily.

* Crops where we can’t monitor the different required parameters of crops.
* All those areas were industrial toxic gases effect on crops health.

Areas where labour cannot go and unable to access, we can easily monitor crop variable on real-time.

**1.6. About the project**

The IoT-based agriculture control system aims to modernize traditional farming practices by incorporating advanced technology to monitor and manage agricultural activities more efficiently. This system utilizes a combination of IoT sensors, a microcontroller, a cloud platform, and a Telegram bot to provide real-time monitoring, data analysis, and remote control capabilities. The primary goal is to enhance farm productivity, optimize resource usage, and support sustainable farming practices.

**2. LITERATURE REVIEW AND BACKGROUND**

### 2.1. Literature Survey

Water management is paramount in countries with water scarcity. This also affects agriculture, as a large amount of water is dedicated to that use. The possible consequences of global warming lead to the consideration of creating water adaptation measures to ensure the availability of water for food production and consumption. Thus, studies aimed at saving water usage in the irrigation process have increased over the years. Typical commercial sensors for agriculture irrigation systems are very expensive, making it impossible for smaller farmers to implement this type of system. However, manufacturers are currently offering low-cost sensors that can be connected to nodes to implement affordable systems for irrigation management and agriculture monitoring. Due to the recent advances in IoT and WSN technologies that can be applied in the development of these systems, we present a survey aimed at summarizing the current state of the art regarding smart irrigation systems. We determine the parameters that are monitored in irrigation systems regarding water quantity and quality, soil characteristics and weather conditions. We provide an overview of the most utilized nodes and wireless technologies. Lastly, we will discuss the challenges and the best practices for the implementation of sensor-based irrigation systems.

### 2.2. Background

IoT technology is used by smart agriculture monitoring systems to track and manage different elements of farming, such as crop health, soil conditions, weather patterns, and more. These systems gather and analyse real-time data from agricultural fields using a network of sensors, actuators, and communication technologies. The processed and used data is then used to optimize farming operations and make educated decisions. The basis of an IoT-based Smart Agriculture Monitoring system is the rising need for productive and environmentally friendly agricultural methods. Traditional agricultural techniques encounter difficulties such erratic weather patterns, a lack of resources, and labor-intensive procedures. IoT and other contemporary technologies have been used to create smart agricultural solutions as a response to these problems. Many author proposed many idea and research about Smart Farming System. Because in Sub-continent agriculture system is very old specially in Pakistan. We yet work on old agriculture. system. Specially, if we talk about the different monitoring parameters of agriculture. We yet adopted the 19’s agriculture method.



**Figure 2.1.** Old Agriculture System

We need to improve this method and change this system into new system. We want to invent new system for agriculture where we improve the yield of crops and also improve the production of crops and also improve the quality of production. Whole world shifted toward smart agriculture for gaining more profit from the agriculture. European Countries is now shifted toward smart agriculture system.

Where they can monitor agricultures variables without need of any huge labour. Also smart agriculture system is less expensive system than manual labour. Because in manual we need huge amount of manual labour for measuring the different values according to different parameters. But in smart agriculture sensor perform these tasks more easily and efficiently than human and manual labor.



**Figure 2.2.** Smart Agriculture Monitoring System

For increasing more yields of crops, we need to adopt this system. With this system more will grow, when crops amount is increase in parallel our export will increase and it is directly affected on our economy. With this our economic condition will improve.

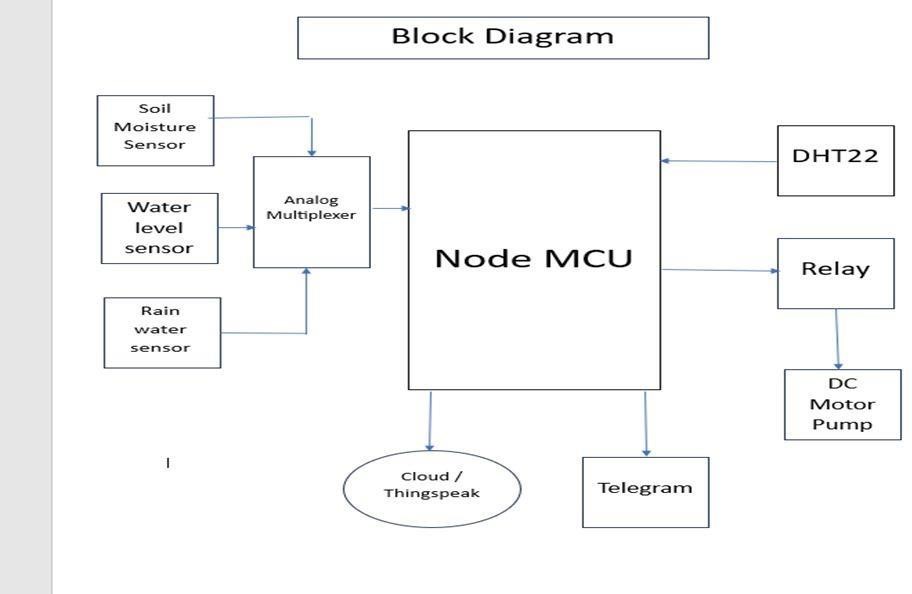
### 3. SYSTEM DESIGN AND IMPLEMENTATION DETAIL

**3.1. System Design**

Smart Agriculture Monitoring System include different sensor the to solve different problems about agriculture and use of these sensor effectively according to their need and utilize the ability of sensors and use it effectively for our purposes. [10] Smart agriculture for more yields of crops convert the agriculture towards new technologies by using new trends. There are different sensors are connected with micro controller and every single sensor performs their own task effectively. The use of a soil moisture sensor in smart agriculture can solve the issue of water scarcity and optimize the use of environmental conditions. It uses smart Soil moisture sensors to actually detect the physical condition of earth that whether it has a need of water or not. Smart Agriculture include gas sensor it will detect different toxic gases those are dangerous and harmful for crops health and growth.

Many other sensors are used and will discuss in detail in Portion of Hardware. [11]

**3.2. Block Diagram**



**Figure 3.1**. System Block Diagram

**3.3. Requirements Analysis**

The major requirements of this system are

Determining actual soil moisture level with aid of soil moisture sensor.

Converting this analogue data collected in digital form through Micro-controller with aid of coding.

Make this data to be transferred between sensors unit and to main controller.

Making main controller (Arduino Uno) capable to receive actual sensor data.

* Setting the Threshold voltage of Different sensors.

Fetch the data of different sensor in IoT platform (Thingspeak) through API in the system to make system to generate reading. Monitored these all values on Thingspeak in form on Graph.

* 1. **Methodology**

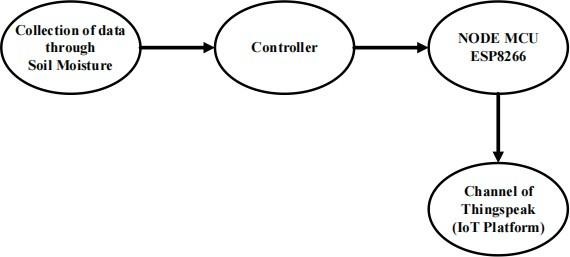
In this section, we outline the methodological and implementation details for the IoT-based agriculture control system leveraging a Telegram bot for smart farming. This process involves several steps, including setting up the hardware, programming the microcontroller, integrating with a cloud platform, and developing the Telegram bot for real-time interaction.

* 1. **Hardware Setup**

Setting up the hardware and development environment is a crucial step for the implementation of an IoT-based agriculture control system leveraging a Telegram bot. This section details the components, their connections, and the development setup required for the project.

* + 1. **Physical earth Condition**

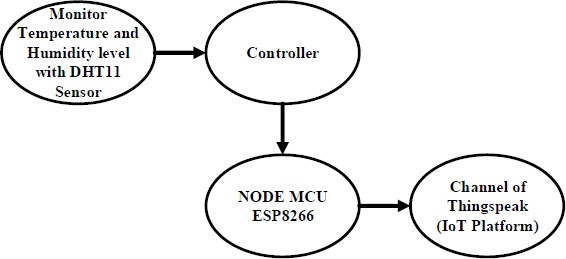
Soil moisture sensor that collects data (moisture content) from earth/field and sends it to Micro-controller ATMEGA 328 UNO where this analogue data is converted to digital form. Because digital form is easily understandable by micro-controller and micro- controller have able to read the data in digital form.Also controller send all data toward NODE MCU ESP8266 and then NODE MCU ESP8266 is connected with internet with the help of internet NODE MCU ESP8266 send data to Thingspeak channel.



***Figure 3.2.*** Soil Moisture Data Collection Process

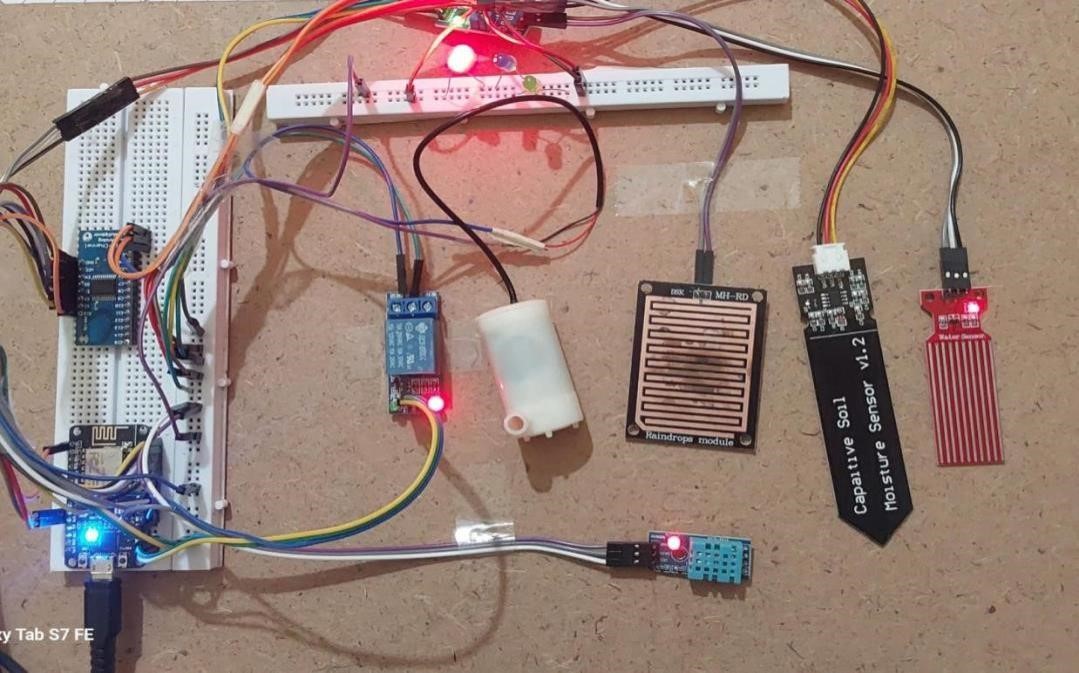
* + 1. **Temperature and Humidity Level**

DHT11 sensor collect Temperature and Humidity level and then send to controller. Then convert this analogue data into digital. Then controller send this data to NODE MCU ESP8266 and with the help of Internet NODE MCU ESP8266 able to send data on channel of Thingspeak.



**Figure 3.3.** Process to Measure Temperature and Humidity Level with DHT-11

**3.6. Hardware Component Detail**



**Figure 3.4.** Final Look of Project with NODE MCU ESP8266

**3.6.1. Hardware Component to be Used**

* Temperature and Humidity Sensor

* Soil Moisture Sensor

* Relay

* Water Pump

* Arduino USB Cable

* NODEMCU ESP8266

* Male and Female Wires

* Breadboard Power Supply

* water level sensor

* Multiplexer

* Rain water sensor

**3.6.2. Temperature and Humidity Sensor**

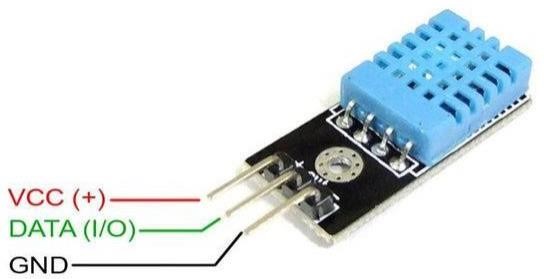
This device is use to take data about Temperature and Humidity. Temperature is Important factor when crops are beginning to ready. When temperature exceed the maximum level, it will be dangerous for crops growth, health and Quality.

Accordingly, we use DHT-11 sensor to measure the temperature and Humidity. This sensor take data from its surrounding. All data in its surrounding in analog form and this analog data send to micro-controller and change into digital form in sequence. Digital form is easily understandable and transferable.

Why we are using Temperature and Humidity Sensor?

Just like humans and other living things, plants have evolved to be sensitive to changes in temperature and humidity. In order to keep comfortable, we get ready for the impending winter, summer, or rainy seasons. Similar to how people prepare for the next seasons, plants do the same, either to adjust for the worse or to blossom with fruits and blooms.

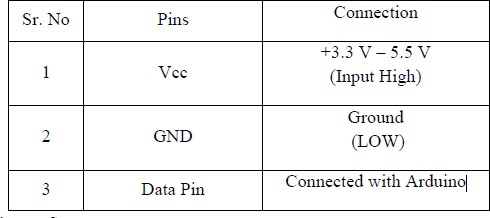
The DHT11 digital sensor is used to measure temperature and humidity in order to determine the start of crop growth or fruit production, as these variables play a significant role in the process.



**Figure 3.5.** Temperature and Humidity Sensor DHT-11

#### 3.6.2. Pin Configuration

The pin arrangement of the DHT-11 sensor, which is available from several manufacturers, can change from the one seen in the circuit diagram. Determining the appropriate Vcc, GND, and output/Data Pins for your particular DHT-11 sensor is so crucial.



**Table 3.1.** Pin Configuration Table of DHT Sensor

**3.6.3. Soil Moisture Sensor**

A tool that gauges the quantity of water in the soil is called a soil moisture sensor. A soil moisture sensor may be used in an IoT-based smart agriculture monitoring system to give 17 farmers and growers vital information on the water requirements of their crops. Here are some reasons why soil moisture sensor is useful in this context:

1. **Automation of Irrigation System:** By measuring the soil moisture levels, the sensor can be used to control the irrigation system to automatically provide water to the crops only when it is needed, improving the efficiency of water usage and reducing waste.
2. **Real-time monitoring:** The sensor can be connected to the internet through a microcontroller or gateway and the data can be monitored in real-time using a mobile application or web portal, enabling farmers to make informed decisions about irrigation.
3. **Early warning:** By monitoring the soil moisture levels, the sensor can provide early warning of potential issues, such as dry soil or waterlogging, allowing farmers to take action before the problem becomes severe.
4. **Adaptability:** Soil moisture sensors can be used in a variety of crops and soil types, making them a versatile tool for farmers and growers.
5. **Cost-effective:** Soil moisture sensors are relatively inexpensive, making them a cost- effective solution for automating irrigation systems.

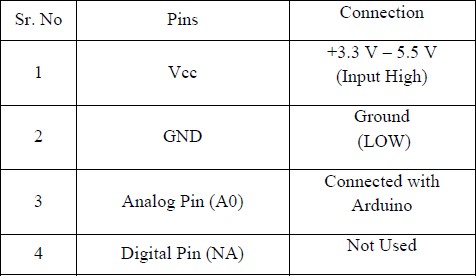


#### Figure 3.6. Soil Moisture Sensor

**6. Water saving:** By providing accurate data about the soil moisture, the sensor can help farmers to save water and reduce the water consumption.

Overall, Soil moisture sensor is a key component in the IoT-based smart agriculture monitoring system as it plays an important role in providing accurate information about the water needs of the crops, allowing farmers to make informed decisions about irrigation. **Pin Configuration**

The pin configuration of a soil moisture sensor can vary depending on the specific sensor being used. Some soil moisture sensors have additional pin like digital output pin and power supply pin like 3.3V or 5V. Please note that the pin configuration may vary depending on the specific soil moisture sensor used, it's always a good idea to check the datasheet before using it. And also make sure that the power supply voltage is matched with the sensor voltage rating. However, here is an example of a common pin configuration for a soil moisture sensor in following below table.



#### Table 3.2. Pin Configuration Table of Soil moisture Sensor

**3.6.4. Water level sensor**

The level transmitter working principle is based on the relationship between liquid static pressure and the liquid level (P=ρgH), and the static pressure is converted into an electrical signal to measure the liquid level, which is an important application of the pressure sensor.

1. **Simple structure**: There are no movable or elastic elements, so the reliability is extremely high, and there is no need for regular maintenance during use. The operation is simple and convenient.
2. Convenient **installation**: When using, first connect one end of the wire correctly, and then put the other end of the water level probe into the solution to be measured.
3. Ranges **are optional**: you can measure the water level in the range of 1-200 meters, and other measurement ranges can also be customized.
4. Wide **range of applications**: suitable for liquid level measurement of high temperature and high pressure, strong corrosion, high pollution, and other media. [Building an electronic water level gauge o](https://youtu.be/ZVkI80UYaVc)n the river bank can be used for tide monitoring.
5. Wide **range of measuring medium**: High-precision measurement can be carried out from the water, oil to paste with high viscosity, and wide-range temperature compensation is not affected by the foaming, deposition, and electrical characteristics of the measured medium.
6. **Long service life**: Generally, the liquid level sensor can be used for 4-5 years in a normal environment, and it can also be used for 2-3 years in a harsh environment.
7. **Strong function**: It can be directly connected to the digital display meter to display the value in real-time, or it can be connected to a variety of controllers and set the upper and lower limits to control the water volume in the container.
8. **Accurate measurement**: The built-in high-quality sensor has high

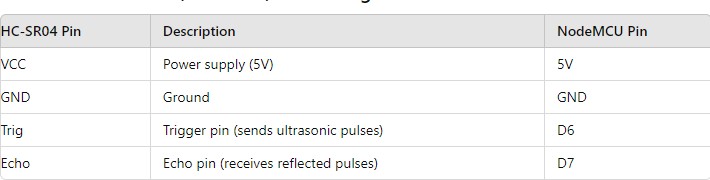


#### Figure 3.7. Water level Sensor

sensitivity, fast response, and accurately reflects the subtle changes of the flowing or static liquid level, and the measurement accuracy

1. **Variety of types:** [liquid level sensors h](https://www.renkeer.com/level-and-pressure-sensors/)ave various structural designs such as input type, straight rod type, flange type, thread type, inductive type, screw-in type, and float type. It can meet the measurement needs of all different place.

**Pin Configuration**



**Table 3.3.**Table of water level sensor

#### 3.6.5. Relay

Relays are electrically powered switches that may be used to control circuits with high voltage or current using signals with lower voltage or current. In an IoT-based smart agriculture monitoring system, relays can be used to control various devices such as irrigation systems, lighting systems, and ventilation systems based on the information collected by the sensors.

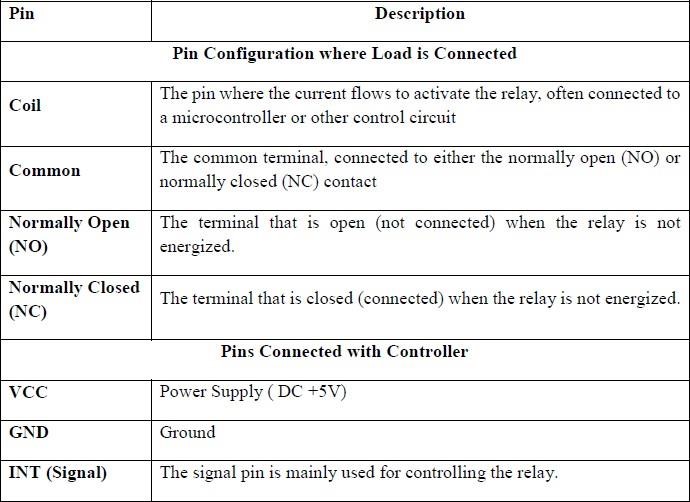
1. **Isolation**: Relays provide electrical isolation between the control circuit and the load circuit, which can protect the microcontroller and other low-voltage components from damage caused by high-voltage or high-current circuits.
2. **Control High Voltage/Current**: Relays can control high voltage and high current loads with a low voltage control signal, which is compatible with microcontrollers and other low- voltage electronic devices.
3. **Remote Control**: Relays can be controlled remotely, for example, when the microcontroller receives a signal from the cloud, it can use the relay to turn on or off the light or irrigation system
4. **Automation**: Relays can be used to automate various systems and processes, such as irrigation, lighting, and ventilation, based on the information collected by the sensors.
5. **Safety**: Relays can be used to control potentially dangerous equipment or systems, such as electrical heating elements, in a safe manner.

Relays are widely used in IoT projects to control devices, automate processes, and ensure safety.



**Figure 3.8.** Double Channel Relay

##### Pin Configuration



**Table 3.4.** Table of Relay

### 3.6.5. Water Pump

In an IoT-based smart agriculture monitoring system, a water pump can be used to automatically provide water to crops based on the information collected by the sensors. Some reasons for using a water pump in this context are:

1. **Irrigation Automation:** By using a water pump in conjunction with sensors that measure soil moisture, temperature, and humidity, the irrigation system can be automated to provide water to the crops only when it is needed, improving the efficiency of water usage and reducing waste.
2. **Remote Control:** A water pump can be controlled remotely through the use of a microcontroller or other control circuit, allowing farmers to start or stop the pump remotely.
3. **Cost-effective:** Water pumps are relatively inexpensive and efficient, making them an affordable and practical solution for automating irrigation systems.
4. **Efficiency:** It can pump water from a water source to the field in a faster and more efficient way than manual watering, allowing the farmer to focus on other tasks.
5. **Adaptability:** Water pumps can be used with different types of irrigation systems, such as drip irrigation, sprinkler systems, and flood irrigation.
6. **Watering in Dry area:** Water pumps can be used in areas where water is scarce or difficult to access, making it possible to bring water to the crops even in dry conditions. Overall, Water pump is an important component in the smart agriculture monitoring system as it plays a vital role in providing water to the crops at the right time, in the right quantity and in the right place. Special program fed into controller. Controller automatically turn on the water pump when moisture percentage is less than 50% and turn off automatically when moisture level is greater than 70% percentage. We can vary these level of turn on and turn off by do changing in programming of controller.



**Figure 3.9.** Hydroponic Water Pump

**3.6.6. Arduino USB Cable**

Arduino USB cables are used to connect the Arduino microcontroller board to a computer or other USB-enabled device. The USB cable provides a communication interface between the microcontroller and the computer, allowing the computer to send data to the microcontroller, receive data from the microcontroller, or program the microcontroller's flash memory with new firmware. Additionally, the USB cable provides a power source for the microcontroller, allowing it to operate without the need for a separate power supply. By using a USB cable, the Arduino can be easily integrated into a variety of projects and systems, making it a versatile and accessible platform for makers and hobbyists.

Arduino USB cable is use to communicate between Arduino and computer. It helps us to upload the c program (code) into Arduino. We can easily upload the code in Arduino through this USB cable. This USB cable make communication very easy between Arduino and computer.



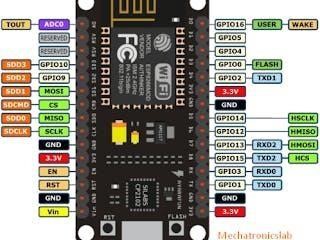
**Figure 3.10.** Arduino USB Cable

**3.6.7. NODEMCU ESP8266**

An open-source development board called the NodeMCU ESP8266 combines a microcontroller unit (MCU) with the ESP8266 Wi-Fi module. Its goal is to offer an intuitive environment for quick prototyping of IoT (Internet of Things) solutions. A low-cost, Wi- Fi-24 enabled microcontroller with a built-in TCP/IP protocol stack, the ESP8266 module is perfect for Internet of Things applications. It utilizes the 2.4GHz Wi-Fi band and is compatible with TCP/IP, UDP, HTTP, and MQTT, among other common Wi-Fi protocols. The ESP8266 module is integrated with extra functionality by the NodeMCU board to make it more usable and accessible. It has a chip that converts USB to serial, making it simple to programmed and communicate with the board via a USB connection.

Additionally, the board has GPIO (General Purpose Input/Output) pins that may be used to connect to and operate extraneous hardware like sensors. The NodeMCU ESP8266's interoperability with the Arduino IDE (Integrated Development Environment) is one of its main benefits. This implies that you may utilize the NodeMCU rich Arduino libraries and community support while programming it using the well-known Arduino programming language. The NodeMCU ESP8266 has a number of properties, including as its tiny form size, low power consumption, and onboard Wi-Fi connection, that make it appropriate for IoT applications. It may be used for a variety of IoT projects, including data logging, sensor monitoring, home automation, and remote-control systems.

Generally speaking, the NodeMCU ESP8266 offers a user-friendly and adaptable platform for creating IoT applications. It is a popular option for developers and amateurs alike because to its mix of Wi-Fi capabilities, Arduino compatibility, and ease of usage.



**Figure 3.11**. NODE MCU ESP8266

**3.6.8. Male and Female Wires**

Male and Female wires are used to communicating micro controller and sensors. We also know as jumper wires. Male and female wires are used to connect different electrical components and devices together. The terms "male" and "female" refer to the physical connectors on the wires, which can be plugged together to create a connection. Male connectors have projecting pins or prongs that can be inserted into the corresponding female connectors, which have holes or sockets to receive the pins. This type of connection is commonly used in various electronics applications, such as in cables, sensors, actuators, and other devices. By using male and female wires, different devices can be easily connected and disconnected as needed, providing a flexible and modular approach to system design. This makes it easier to build and modify systems, as well as to troubleshoot and repair them. The choice of male and female connectors depends on the specific requirements of the devices being connected, including the current and voltage requirements, the type of signals being transmitted, and other factors such as environmental conditions and mechanical stability.



**Figure 3.12.** Male and Female Wires

**3.6.9. Bread board Power Supply**

Breadboard power supplies are used in electronics projects to provide a regulated and isolated power source for the components and devices being used on a breadboard. A breadboard is a type of prototyping board that allows the user to quickly and easily test and prototype electronics circuits. Breadboard power supplies are designed to be used with breadboards, and they typically provide both positive and negative voltage rails that can be connected to the breadboard's power and ground buses. This allows the components and devices on the breadboard to be powered and connected to the power source in a simple and straightforward manner.

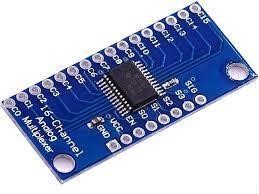
We give DC input 5V-12V to Breadboard and it provide us different junctions of 5V and 3V. On this breadboard GND is also available with positive voltages.



**Figure 3.13.** Breadboard Power Supply

**3.6.9. Analog Multiplexer**

A 16:1 multiplexer is a digital switch that allows one of sixteen inputs to be selected and routed to a single output line. It is an essential component in digital systems, particularly for applications requiring multiple inputs to be handled by a single output line efficiently.



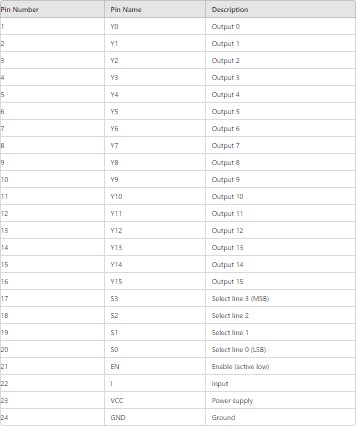
**Figure 3.14.** Analog multiplexer

* **Selection Lines**: The 16:1 multiplexer has 4 selection lines (S3, S2, S1, S0) because24=162^4 = 1624=16, allowing one of 16 inputs to be selected.

**Inputs and Output**: There are 16 input lines (I0 to I15) and one output line (Y).

* **Enable Signal**: Many multiplexers include an enable signal (EN) which, when low, disables the output, making it high impedance (tri-stated)

**Pin Configuration:**



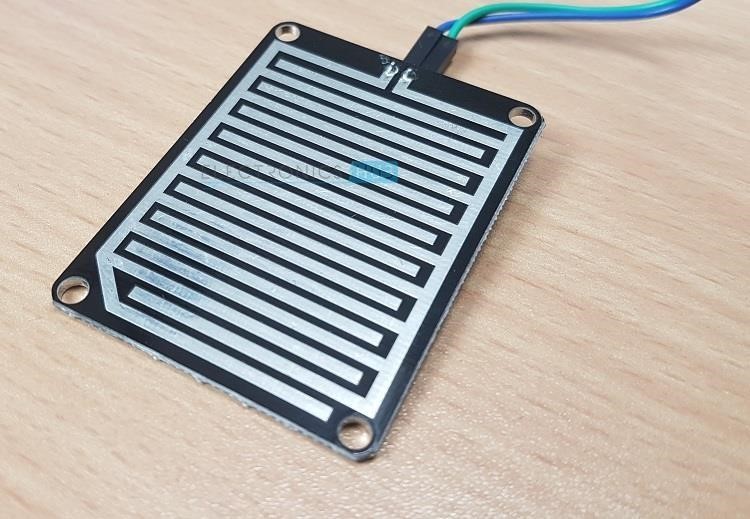
**Table 3.5**. Table of analog multiplexer

**3.6.10. Rainwater sensor**

A rainwater sensor, often referred to as a rain sensor, is a device used to detect the presence and intensity of rain. It is commonly used in applications such as automatic wipers, irrigation systems, and weather monitoring. In the context of your IoT-based agriculture control system, a rainwater sensor can be used to monitor rainfall and make real-time decisions for irrigation.

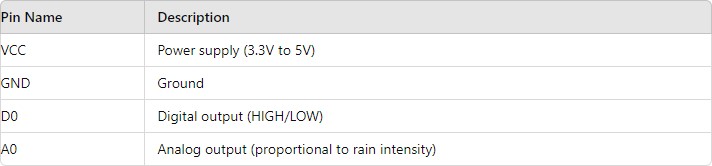
A typical rainwater sensor consists of two main components:

* 1. **Rainwater Sensing Plate**: A plate with exposed conductive traces that can detect rainwater.
  2. **Control Module**: Processes the signal from the sensing plate and outputs a digital or analog signal indicating the presence of rain.



**Figure 3.15.** Rainwater sensor

**Pin Configuration**

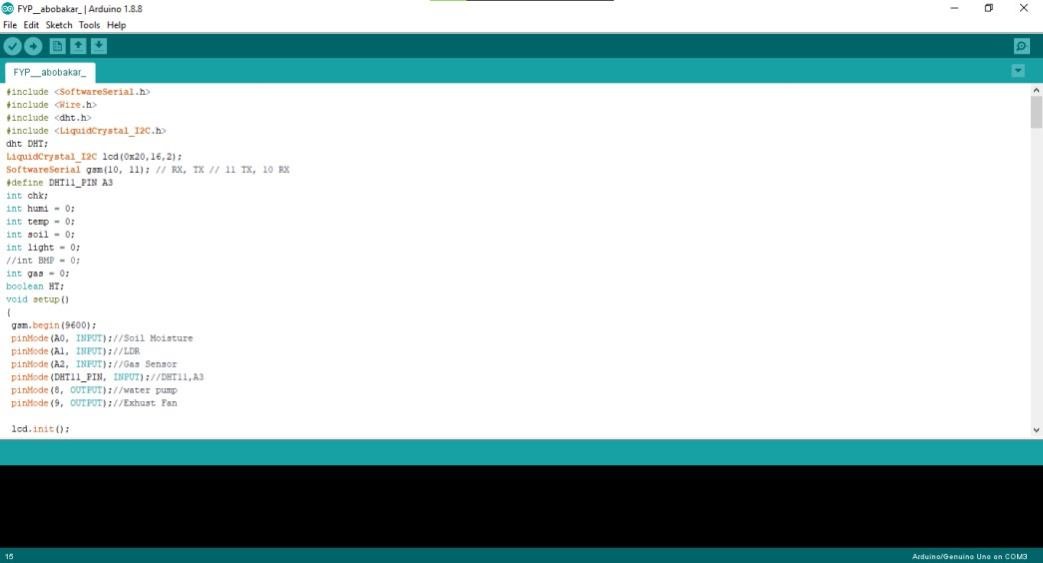


**Table 3.6.**Table of Rainwater sensor

**3.7 Software Detail**

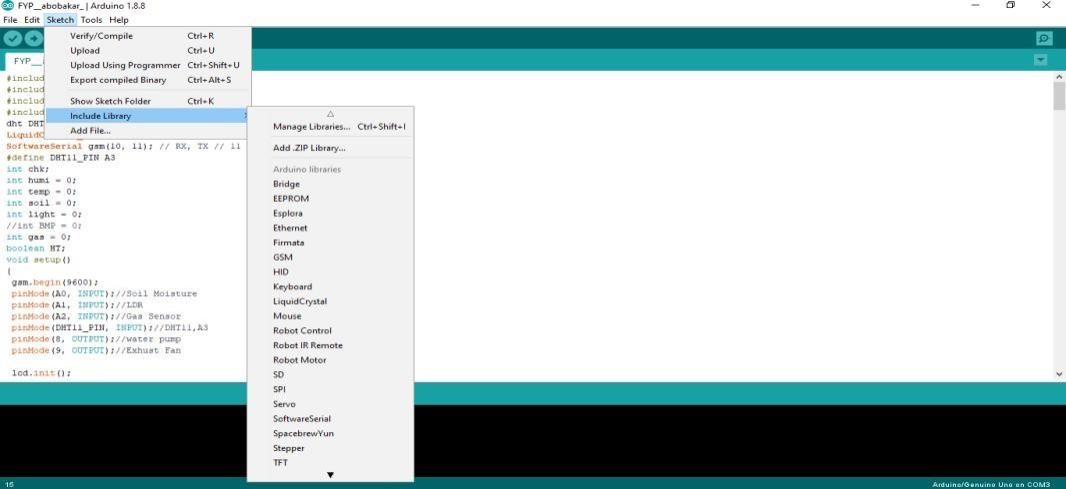
#### 3.7.1. Arduino IDE

Software for Arduino boards is programmed and developed using the software environment known as the Arduino IDE (Integrated Development Environment). It’s simple and user- friendly interface for programming Arduino boards includes a code editor with syntax highlighting and auto-completion, a serial monitor for communicating with the board, and a built-in library of functions and examples. The Arduino IDE is used to create the code for the IoT-based smart agricultural monitoring system that reads data from the sensors, analyses the data, and regulates the actuators depending on the data. Using a USB cable, the code is sent to the Arduino board, which then executes it to manage the sensors and actuators that are attached to it. The serial monitor in the Arduino IDE can be used to debug and test the code, and to monitor the data being read from the sensors and sent to the NODE MCU ESP8266 module for transmission to Thingspeak. The Arduino IDE is an essential tool for developing and programming the code that controls the system.



##### Figure 3.16. Arduino IDE Code

How to add all required Libraries in Arduino IDE? Step to add libraries in Arduino is following below: 1. OpenArduinoIDE.

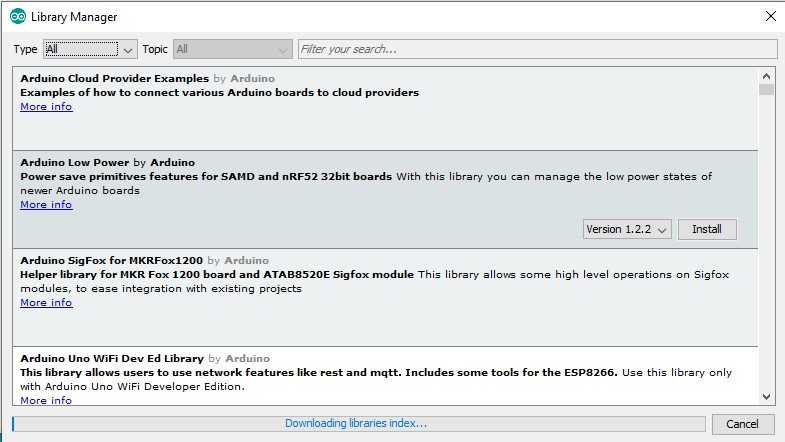


##### Figure 3.17. Add Library Menu

1. In the Left Top Corner Click on Sketch >> Include Library. Then menu will display.

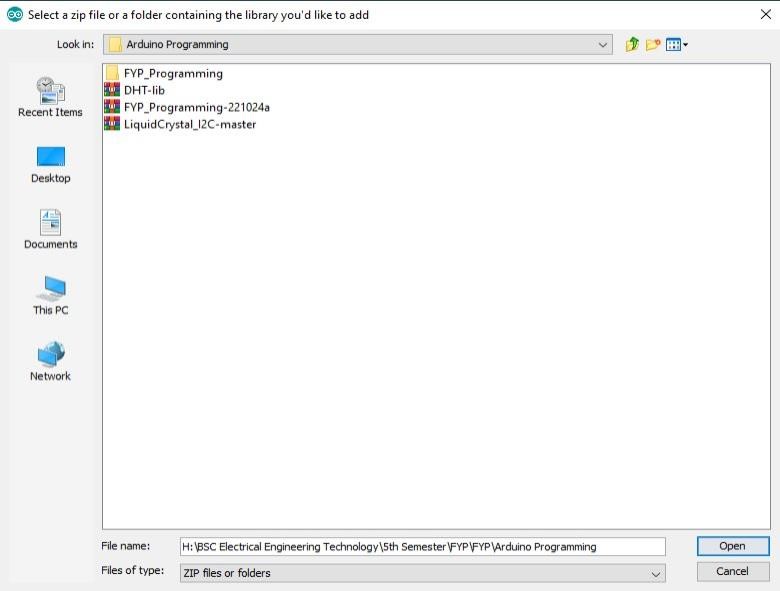
There are two main menus will be display.

1. First download direct library from Arduino IDE from Manage libraries menu. These libraries provided by the platform of Arduino and GitHub. The menu will display when we click on Library Manager. Add Libraries according to requirement



##### Figure 3.18. Library Manger Menu

3. Second add library if you have any library in Zip file. Following menu will display, thenweclickonAdd.ZipLibrary



##### Figure 3.19. Zip Library Menu

1. LiquidCrystal\_I2C.h
2. b. DHT11 sensor

Note:

When whole programming is done need to add correct APN Number and Channel link at required place of Arduino code. Otherwise data cannot be send.

**Other details:**

Thingspeak

Proteus

**3.7.2. Thingspeak**

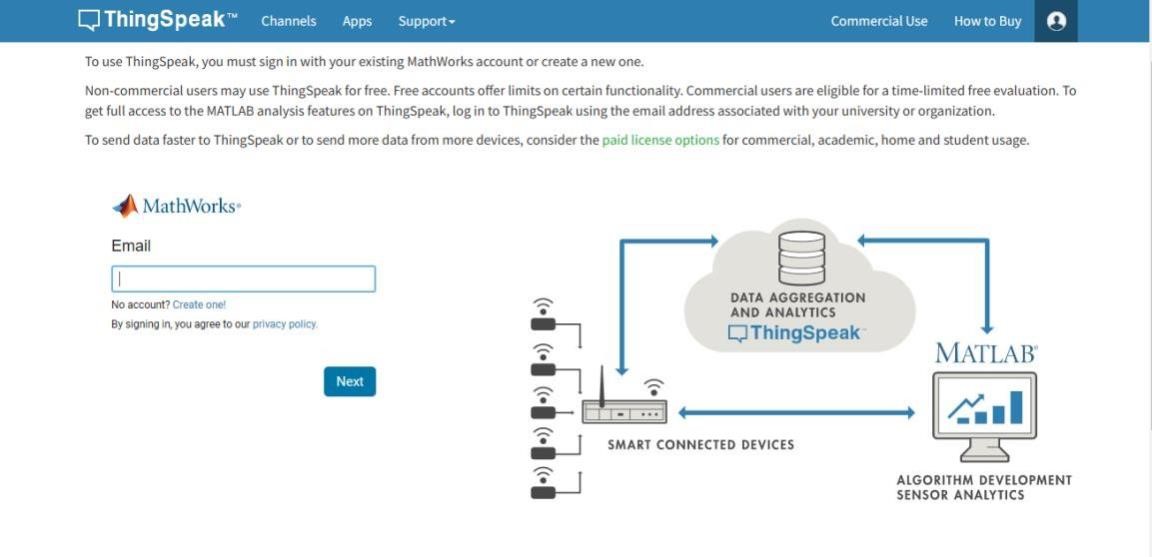
An open-source IoT platform called ThingSpeak may be used to gather, examine, and display data from IoT devices. ThingSpeak may be used to gather and store data from the many sensors and actuators utilized in the IoT-based smart agricultural monitoring system. The data collected by ThingSpeak can be analysed to gain insights into the environmental conditions and plant growth, and this can be used to make informed decisions about the use of resources such as water and fertilizers. Additionally, the platform can be used to set up alerts and notifications when certain conditions are met, such as when the soil moisture levels are too low, which can help farmers take immediate action to address the issue. The data collected by ThingSpeak can also be visualized using charts and graphs, which can provide farmers with a better understanding of the trends and patterns in the data, and help them to make more informed decisions about their farming practices.

Overall, ThingSpeak is a useful tool for collecting, analyzing, and visualizing data in the IoT-based smart agriculture monitoring system, and can help farmers to make more informed decisions about their farming practices, leading to more efficient and sustainable agriculture.

**Create Channel in Thingspeak**

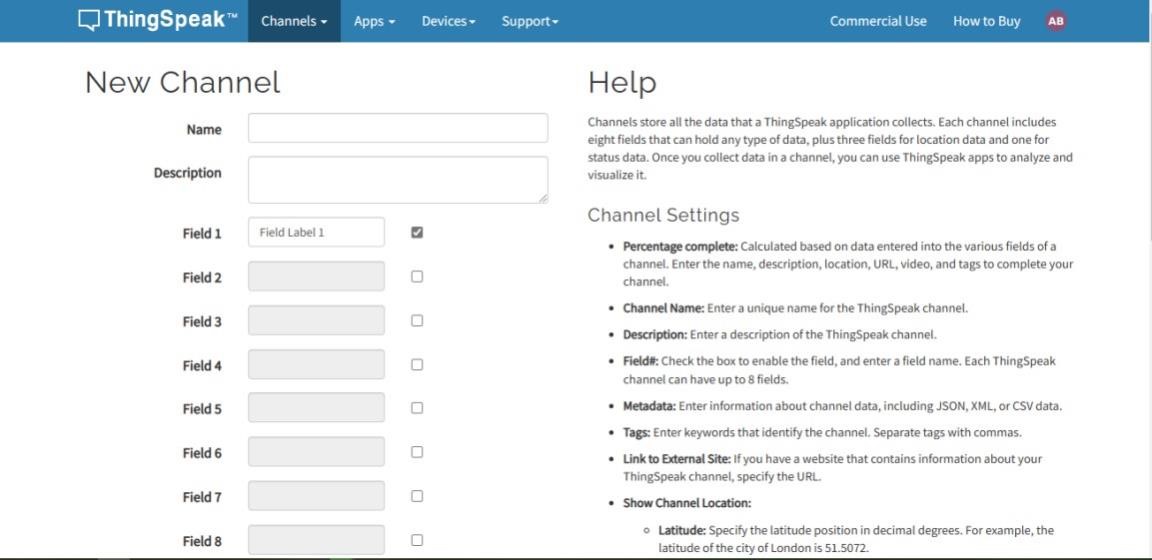
To create a channel on ThingSpeak, follow these steps:

1. Go to the ThingSpeak website (thingspeak.com) and sign in or create a new account.



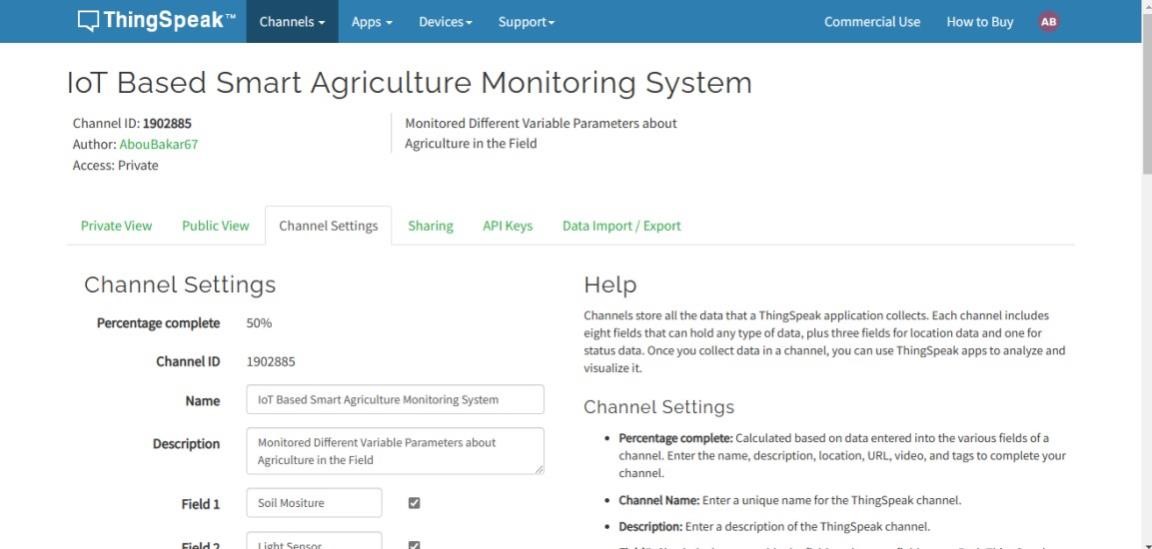
#### *Figure 3.20.* Thingspeak Website

2. Click on the "Channels" tab in the top menu and then click "My Channels".



#### Figure 3.21. Unfilled Field on My Channel

1. Click the "New Channel" button.
2. Fill out the form with the required information such as the name of the channel, field names, and other details.

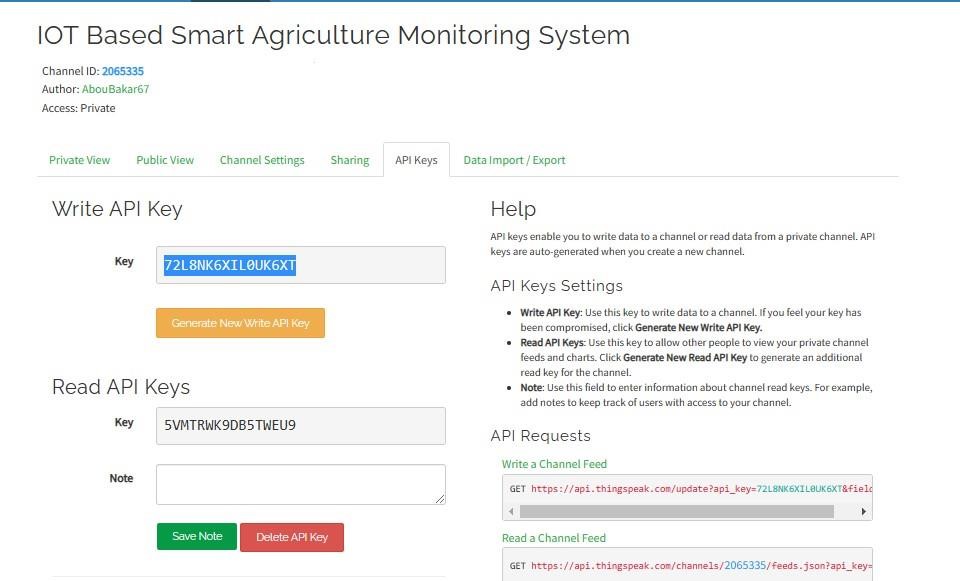


#### Figure 3.22. Filled Field on Thingspeak Channel

1. Click the "Save Channel" button to create the channel.
2. Once the channel is created, you can find the channel ID and API key in the "API Keys" tab of the channel dashboard.

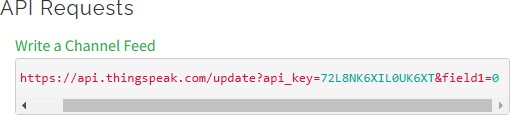


#### Figure 3.23:API Key



#### Figure 3.24. API Key of Channel

Every channel has unique API Key and Every unique API Key have unique Channel link. With the help of this link we can easily access the channel. This link is added in arduino programming then data automatically send on this channel.



#### Figure 3.25. API Requested link

**What is API?**

An API is a collection of protocols, procedures, and development resources for software applications. It outlines how components should work together and enables communication between various applications. APIs are a set of protocols, routines, and tools for building software applications. They are used to simplify and standardize the way software components interact, making it easier to build and integrate software applications. APIs are often used to allow a mobile app to communicate with a backend server, or to enable different software applications to exchange data. An API defines the methods that are available for accessing a particular software component, and specifies the input and output parameters for each method.

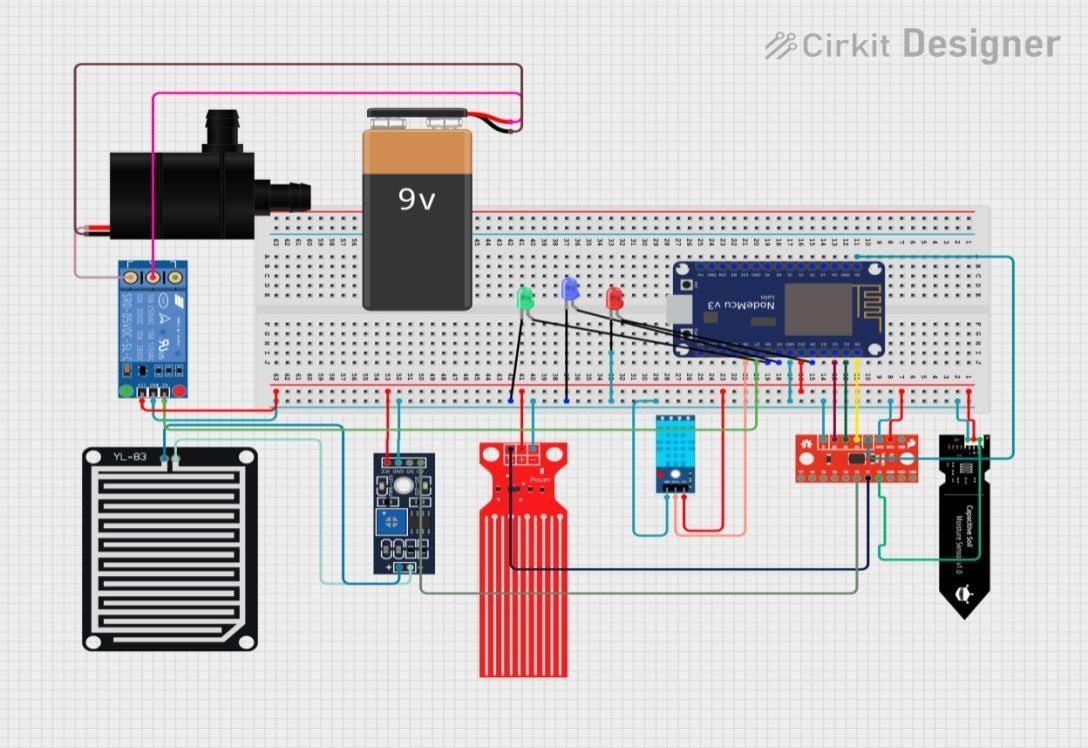
By following the rules and protocols specified in the API, software developers can build applications that can communicate with the software component and use its functionality. After creating the channel, you can begin sending data to ThingSpeak from your IoT device or application. The platform provides various options for data visualization, analysis, and alerts that you can use to monitor and manage your data.

**3.8 Simulation Details**

#### 3.8.1Simulation of Physical Data

Proteus software is a popular simulation software that can be used to simulate and test the IoT-based smart agriculture monitoring system before deploying it in the field. This software allows you to simulate the behavior of the system, test the code, and verify the overall functionality of the system. It can simulate the behavior of the Arduino board, sensors, and actuators, and test the code that controls the actuators.

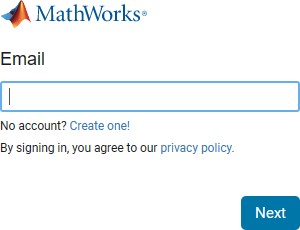
Proteus software can also verify the accuracy of the data obtained from the sensors and ensure that the actuators are being controlled as intended based on the sensor data. It can also be used to test the system under different environmental conditions, such as varying light intensity and soil moisture levels, to save time and resources.



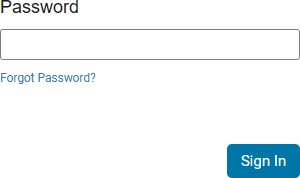
**Figure 3.26.** Proteus Simulation of Sensors

##### Remotely Access IoT Channel from Desktop over the Internet from the worldwide

We can easily access main controller from our desktop. We need to just install any web browser. Like, Chrome, Mozilla Firefox and Microsoft Edge or etc. After Installing anyone of browser. Just go on thingspeak official link. Link is following below: https://thingspeak.com/ Just visit this website and login your account by adding required Email and Password. Then we can easily Access and no need an extra knowledge to access IoT Channel.



**Figure 3.27.** Email Interface



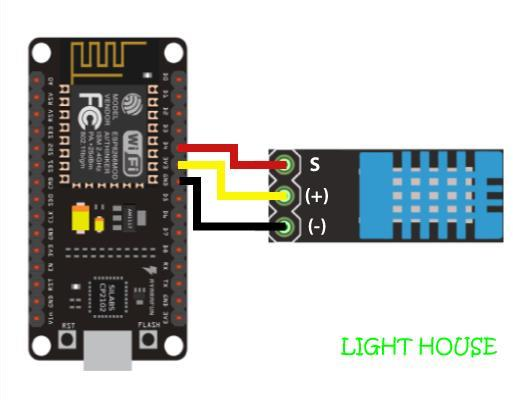
**Figure 3.28.** Password Interface

**4. CODE AND RESULTS**

**4.1 Prototype**

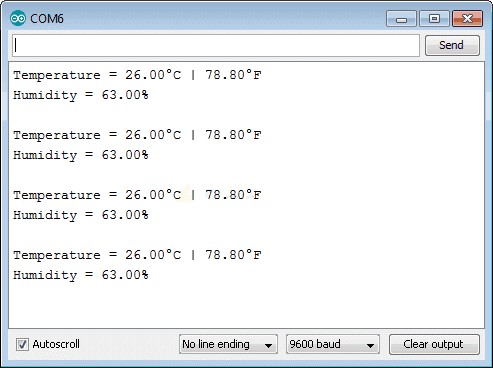
#### 4.1.1 Measure Temperature and Humidity Level through Sensor DHT11

Firstly, Sensor is energized through Arduino or an external power source. In ordered like above sensors connected sensor pins with arduino. Like: Vcc, Data and Ground. With the combination of +5V and Ground sensor is now powered on and start working. But with Data Pin of sensor give some inputs to arduino as a voltage of gathered data or values. In this way sensor working. After this, now sensor is able to measure the Temperature and Humidity level.



##### Figure 4.1. DH11 Sensor Integrated with NodeMCU ES8266

**Serial Monitor**

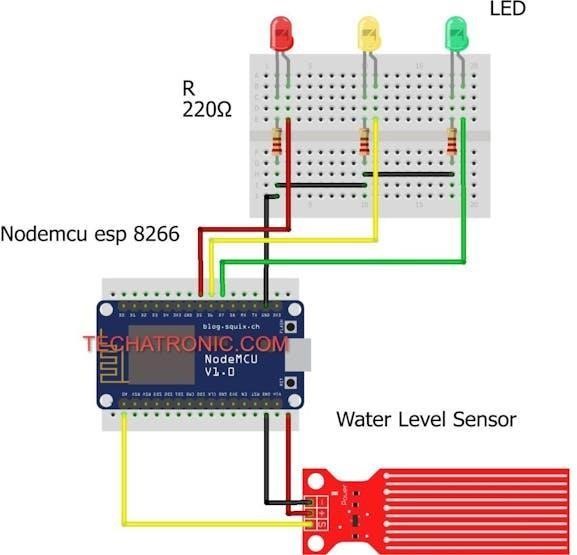


**Figure 4.2.** DHT11 Reading on Serial Monitor

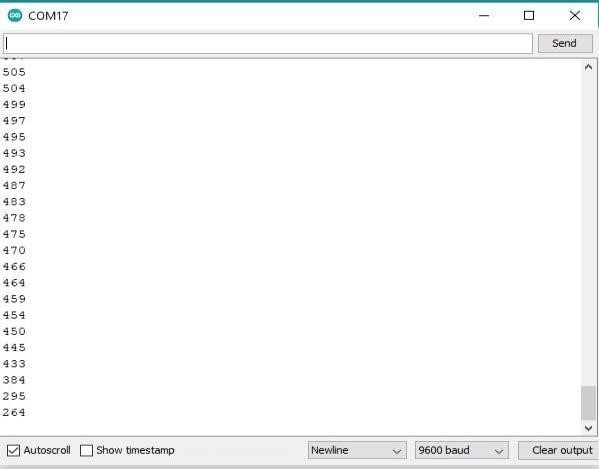
**4.1.2. Measured water level through sensor water level**

Water level is one of the most commonly measured parameters, as accurate level data are essential for many applications. While climate change, pollution monitoring, and industrial water usage are broad reasons for monitoring water levels, more specific applications are discussed throughout this page.

Level is perceived as one of the most straightforward water parameters. In general, it is the level of water in a body of water, in groundwater, in a tank, etc. However, there’s a lot to unpack with this parameter. Not only are there very different water level applications and technologies used to measure it, but there are also a variety of terms used when describing water level, some of which have only subtle differences.



##### Figure 4.3. Water level sensor on NodeMCU ES8266

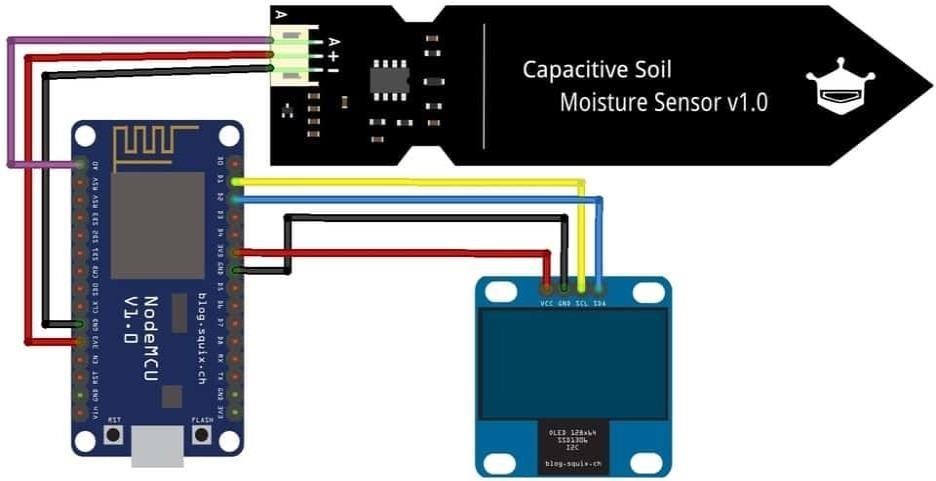


**Figure 4.4.** Water level sensor on Serial Monitor

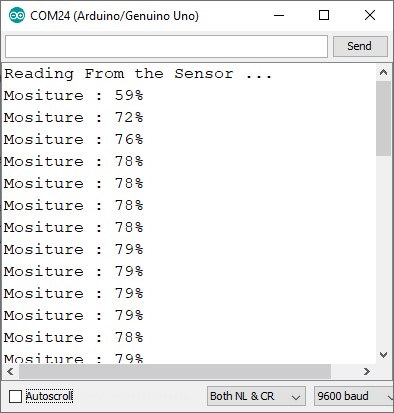
**4.1.3.Physical Data Gathering through Soil Moisture Sensor and Wireless**

##### Sensor Network

Firstly, the sensor will get power. In order we need to connect other connections of sensor with arduino. Like ground GND and Analog Pin A0. Without these Pins Sensor cannot work properly. If any mistake during connection, we need check whole wiring connection before turn it on. Vcc Pin connect with any Digital Pin. For this we need to declared Digital Pin as an output. Otherwise we can provide them some other external power supply. We need to Provide full voltage to sensor until sensor’s Power LED ON.



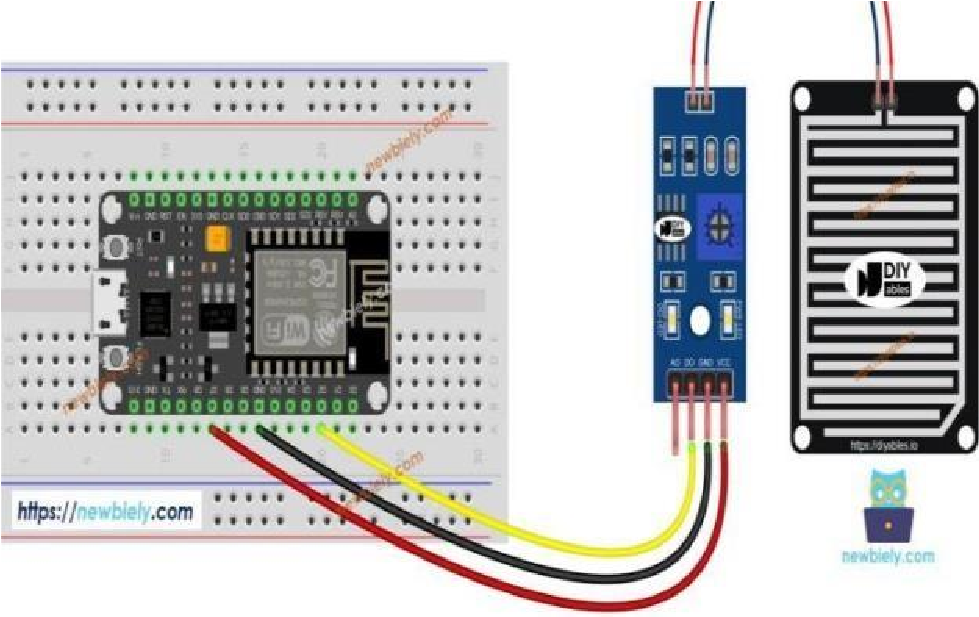
###### Figure 4.5. Soil Moisture Sensor Integrated with Nodemcu es8266



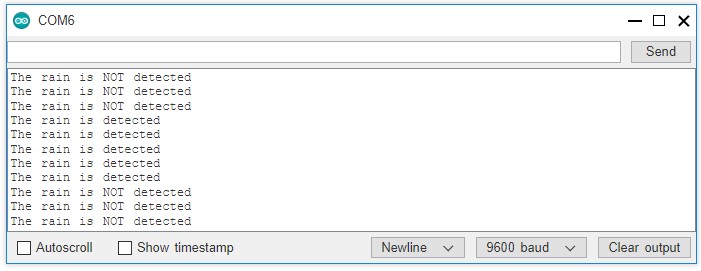
###### Figure 4.6. Serial Monitor Showing the values of Soil Moisture

###### 4.1.5. Rain water sensor

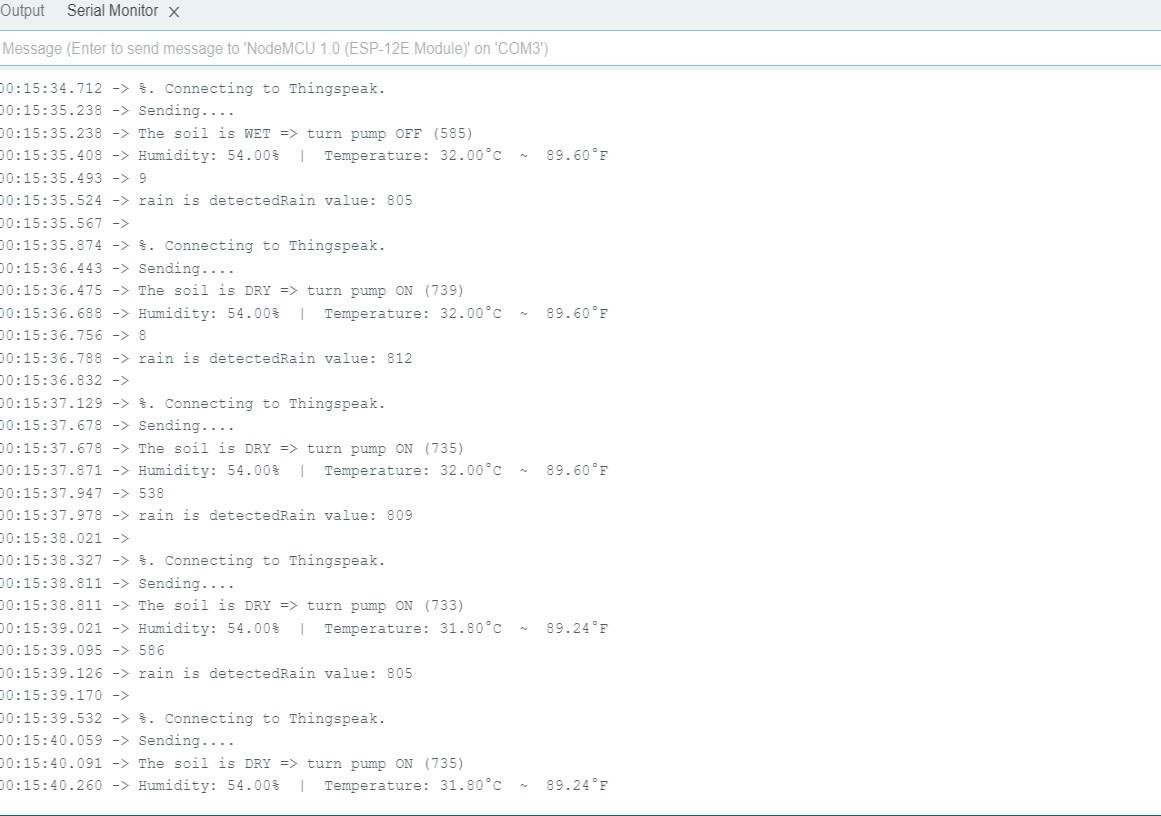
Rainwater detection is an essential task in various fields such as agriculture, water conservation, and weather forecasting. In this project, we have used an ESP8266 microcontroller to detect rainwater and send notifications to the user telegram msgs.



###### Figure 4.7. Rainwater Integrated with NodeMCU ES8266

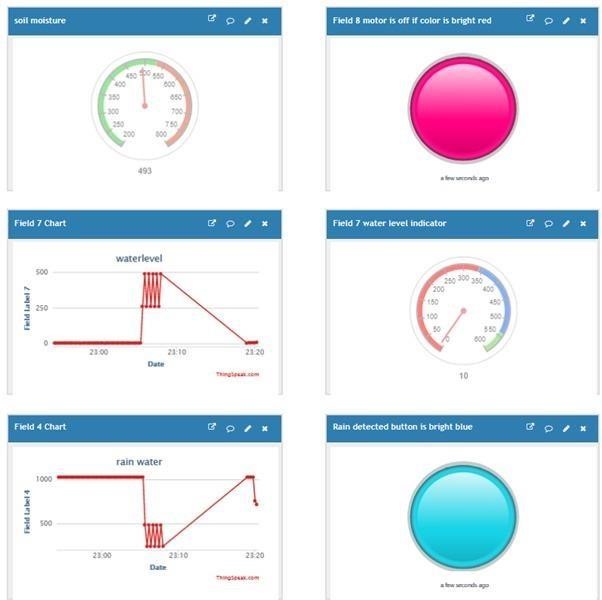
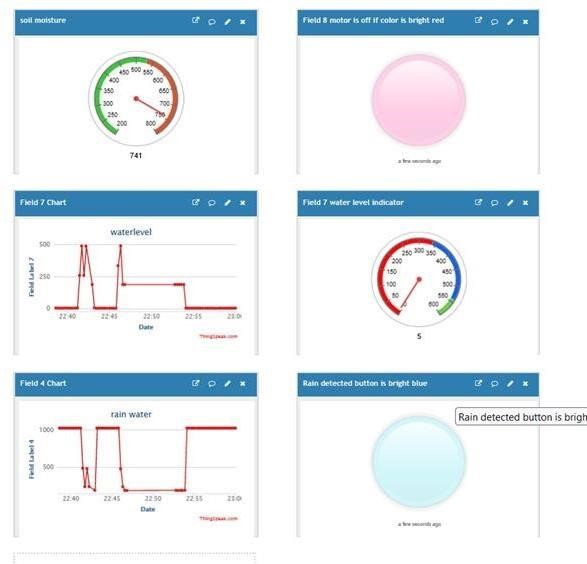


###### Figure 4.8. Results of Rainwater in serial monitor



**4.2 Results**

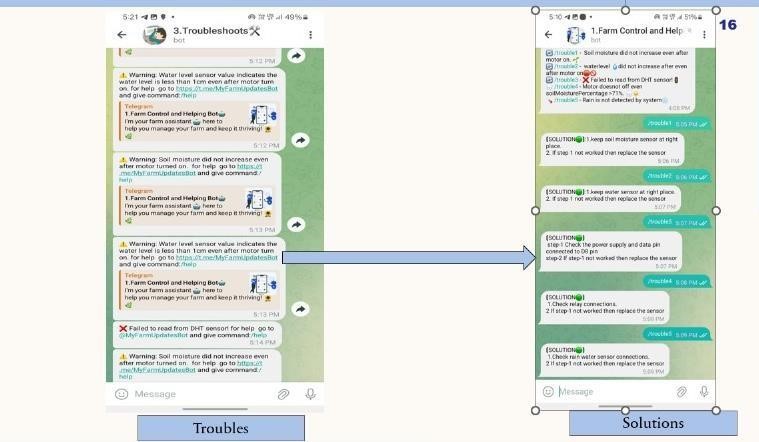
The Output of Different Sensors is Shown Below. These reading shown on Laptop. But We Can Monitor easily these parameters on Cell Phone. All internet GPRS supported cell phones can show these graphical readings on thingspeak.



**Figure 4.9**. Soil Moisture, Water level and Rain Water Real Time Reading on Thingspeak

**Telegram Results**

Telegram notifications and controlling



### TELEGRAM NOTIFICATIONS



**Steps:**

1.Open the Telegram app on your Android phone.

2.Tap the three-line button at the top left and choose 'Settings. '

3.Click on 'Notifications and Sounds. '

3.Make sure you've enabled options under 'Notifications for chats' and 'In-app notifications. ' If not, enable them.

**TELEGRAM NOTIFICATIONS:**



**Figure 4.10.** All Sensors Readings Graphical Representation on Telegram

### 4.3 CODE

// MUX

#define MUX\_A D3 // S3

#define MUX\_B D2 // S2

#define MUX\_C D1 // S1

#define MUX\_D D0 // S0

// LED

#define red D4

#define yellow D5

#define green D6

// Analog for soil moisture, water level, rain

#define ANALOG\_INPUT A0 // SIGNAL PIN ON MUX

// Relay

#define RELAY\_PIN D7 // The ESP8266 pin D7 that connects to relay

#define MOISTURE\_THRESHOLD 50 // Threshold for soil moisture percentage // DHT Sensor

#include <DHT.h>

#include <DHT\_U.h>

#define DHT\_SENSOR\_PIN D8 // The ESP8266 pin D8 connected to DHT11 sensor

#define DHT\_SENSOR\_TYPE DHT11

DHT dht\_sensor(DHT\_SENSOR\_PIN, DHT\_SENSOR\_TYPE);

// WiFi

#include <ESP8266WiFi.h>

const char\* ssid = "Me";

const char\* password = "MMMMMMMM";

WiFiClientSecure client; // Telegram

#include <UniversalTelegramBot.h> // Universal Telegram Bot Library written by Brian Lough: https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot

#include <ArduinoJson.h>

// Initialize Telegram BOT

#define BOTtoken "7405061680:AAFCEhi7ceBIfTCXAEJE9Hqz\_GTD-GeysI0" // your Bot Token

(Get from Botfather)

#define CHAT\_ID "5989476137"

//#define CHAT\_ID "1002116062"

#ifdef ESP8266

X509List cert(TELEGRAM\_CERTIFICATE\_ROOT);

#endif

UniversalTelegramBot bot(BOTtoken, client);

#define BOTtoken1 "7292123197:AAFyZIF0kEieQptQOK\_wcK3Bo6pE8NYu8jA" // your Bot Token (Get from Botfather)

UniversalTelegramBot bot2(BOTtoken1, client);

#define BOTtoken2 "7008038969:AAENg9LKylLleOoLn6jMZIOoqSD5m69Bu4A" // your Bot Token (Get from Botfather)

UniversalTelegramBot bot3(BOTtoken2, client); // Checks for new messages every 1 second. int botRequestDelay = 1000; unsigned long lastTimeBotRan;

bool status=digitalRead(RELAY\_PIN); // Ensure motor is off // Define moisture levels

#define MOISTURE\_HIGH 350

#define MOISTURE\_LOW 800

// Add these variables to track time and previous values unsigned long motorOnTime = 0; // Time when the motor was turned on int previousSoilMoistureValue = 0; // Previous soil moisture value

void setup() { // WiFi checking Serial.begin(9600); delay(10);

Serial.println("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password); while (WiFi.status() != WL\_CONNECTED) { delay(500); Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

// Telegram #ifdef ESP8266

configTime(0, 0, "pool.ntp.org"); // get UTC time via NTP

client.setTrustAnchors(&cert); // Add root certificate for api.telegram.org #endif

// Initialize relay to OFF state

pinMode(RELAY\_PIN, OUTPUT);

// Pin modes pinMode(red, OUTPUT); // Red LED Pin Connected To D4 Pin pinMode(yellow, OUTPUT); // Yellow LED Pin Connected To D5 Pin pinMode(green, OUTPUT); // Green LED Pin Connected To D6 Pin

// Initialize MUX pins

pinMode(MUX\_A, OUTPUT); // D3 pinMode(MUX\_B, OUTPUT); // D2 pinMode(MUX\_C, OUTPUT); // D1

pinMode(MUX\_D, OUTPUT); // D0

// Initialize DHT sensor

dht\_sensor.begin(); // Initialize the DHT sensor

}

void changeMux(int d, int c, int b, int a) {

digitalWrite(MUX\_A, a); digitalWrite(MUX\_B, b); digitalWrite(MUX\_C, c); digitalWrite(MUX\_D, d);

}

void handleNewMessages(int numNewMessages) { for (int i = 0; i < numNewMessages; i++) { String chat\_id = String(bot.messages[i].chat\_id); if (chat\_id != CHAT\_ID) { bot.sendMessage(chat\_id, "Unauthorized user", ""); continue;

}

String text = bot.messages[i].text;

String from\_name = bot.messages[i].from\_name;

Serial.print("Received message: "); Serial.println(text); if (text == "/start") {

String welcome = " \*\*Welcome, " + from\_name + "!\*\* \n\n"; welcome += "I’m your farm assistant here to help you manage your farm and keep it thriving!

\n\n";

welcome += "Here’s what I can do for you:\n\n"; welcome += " \*\*/on\*\* - Turn ON the motor to water your crops \n"; welcome += " \*\*/off\*\* - Turn OFF the motor to stop watering \n"; welcome += " \*\*/state\*\* - Check if the motor is currently ON or OFF \n"; welcome += " \*\*/rain\*\* - Check the current rain status \n"; welcome += " \*\*/temp\*\* - Get the current temperature of your farm \n"; welcome += " \*\*/humidity\*\* - Get the current humidity level \n"; welcome += " \*\*/level\*\* - Check the water level in the tank \n"; welcome += " \*\*/moisture\*\* - Check the soil moisture percentage \n"; welcome += " \*\*/status\*\* - Get the overall status of your farm \n\n"; welcome += " \*\*/help\*\* - Facing Troubles\n\n"; welcome += "Feel free to use these commands to manage and monitor your farm. Happy Farming!

";

String keyboard = "[[{\"text\":\"/on\"},{\"text\":\"/off\"}],"

"[{\"text\":\"/state\"},{\"text\":\"/rain\"}],"

"[{\"text\":\"/temp\"},{\"text\":\"/humidity\"}],"

"[{\"text\":\"/level\"},{\"text\":\"/moisture\"}],"

"[{\"text\":\"/status\"}]]";

String replyMarkup = "{\"keyboard\":" + keyboard + ",\"resize\_keyboard\":true}"; bot.sendMessage(chat\_id, welcome, "Markdown"); bot.sendMessage(chat\_id, replyMarkup, "");

Serial.println("Command /start received. Welcome message sent.");

}

if (text == "/on") { if (status == LOW) {

digitalWrite(RELAY\_PIN, HIGH);

bot.sendMessage(chat\_id, " Motor is now \*\*ON\*\*. Ready to water your crops! ", "Markdown");

Serial.println("Motor turned ON."); delay(60000);

} else {

bot.sendMessage(chat\_id, " Motor is already \*\*ON\*\*.", ""); Serial.println("Motor is already ON.");

}

}

if (text == "/off") {

if (status == HIGH) {

digitalWrite(RELAY\_PIN, LOW);

bot.sendMessage(chat\_id, " Motor is now \*\*OFF\*\*. No more watering for now. ", "Markdown");

Serial.println("Motor turned OFF."); delay(60000);

} else {

bot.sendMessage(chat\_id, " Motor is already \*\*OFF\*\*.", ""); Serial.println("Motor is already OFF.");

} }

if (text == "/state") { if (status == HIGH) {

bot.sendMessage(chat\_id, " Motor is currently \*\*OFF\*\*.", ""); Serial.println("Motor state: OFF.");

} else {

bot.sendMessage(chat\_id, " Motor is currently \*\*ON\*\*.", ""); Serial.println("Motor state: ON.");

} }

if (text == "/rain") {

changeMux(LOW, LOW, HIGH, LOW); int sensorValue3 = analogRead(A0);

String rain = " Rain Sensor Value: " + String(sensorValue3); if (sensorValue3 >= 990) {

bot.sendMessage(chat\_id, rain + "\n No rain detected. ", ""); Serial.println("Rain status: No rain.");

} else {

bot.sendMessage(chat\_id, rain + "\n Rain detected! Take precautions to protect your crops.

", "");

Serial.println("Rain status: Rain detected.");

}

}

if (text == "/temp") {

float temperature\_C = dht\_sensor.readTemperature();

String temp = " Current Temperature: " + String(temperature\_C) + "°C"; bot.sendMessage(chat\_id, temp, "");

Serial.print("Temperature: ");

Serial.print(temperature\_C);

Serial.println("°C");

}

if (text == "/humidity") {

float humidity = dht\_sensor.readHumidity();

String hum = " Current Humidity: " + String(humidity) + "%"; bot.sendMessage(chat\_id, hum, "");

Serial.print("Humidity: ");

Serial.print(humidity);

Serial.println("%");

} if (text == "/level") {

changeMux(LOW, LOW, LOW, HIGH); int sensorValue2 = analogRead(A0);

float waterLevel = map(sensorValue2, 350, 520, 1, 4);

String level = " Water Level: " + String(waterLevel) + " cm (Raw Value: " +

String(sensorValue2) + ")"; if (sensorValue2 < 350) {

bot.sendMessage(chat\_id, level + "\n Low water level - Immediate action required! ", ""); Serial.println("Water level: Low."); } else if (sensorValue2 > 540) {

bot.sendMessage(chat\_id, level + "\n HIGH water level -. ", ""); Serial.println("Water level: Full.");

} else {

bot.sendMessage(chat\_id, level + "\n Medium water level - . ", ""); Serial.println("Water level: Medium.");

}

}

if (text == "/moisture") {

changeMux(LOW, LOW, LOW, LOW);

int soilMoistureValue = analogRead(A0);

int soilMoisturePercentage = map(soilMoistureValue, MOISTURE\_LOW, MOISTURE\_HIGH,

0, 100);

String moisture = " Soil Moisture Percentage: " + String(soilMoisturePercentage) + "%"; bot.sendMessage(chat\_id, moisture, ""); Serial.print("Soil moisture percentage: ");

Serial.print(soilMoisturePercentage);

Serial.println("%");

}

if (text == "/status") { // Read soil moisture

changeMux(LOW, LOW, LOW, LOW);

int soilMoistureValue = analogRead(A0);

int soilMoisturePercentage = map(soilMoistureValue, MOISTURE\_LOW, MOISTURE\_HIGH,

0, 100);

// Read temperature and humidity float humidity = dht\_sensor.readHumidity(); float temperature\_C = dht\_sensor.readTemperature();

// Read water level

changeMux(LOW, LOW, LOW, HIGH); int waterLevelValue = analogRead(A0);

float waterLevel = map(waterLevelValue, 350, 520, 1, 4);

// Read rain sensor

changeMux(LOW, LOW, HIGH, LOW); int rainSensorValue = analogRead(A0);

String rainStatus = (rainSensorValue < 1010) ? " Rain detected. Take precautions!" : " No rain detected."; // Send the status

String status = " \*\*Farm Status\*\*:\n\n"; status += " \*\*Temperature\*\*: " + String(temperature\_C) + "°C\n"; status += " \*\*Humidity\*\*: " + String(humidity) + "%\n";

status += " \*\*Soil Moisture\*\*: " + String(soilMoisturePercentage) + "%\n";

status += " \*\*Water Level\*\*: " + String(waterLevel) + " cm\n"; status += rainStatus;

bot.sendMessage(chat\_id, status, "Markdown");

Serial.println("Farm status sent.");

}

if(text == "/help"){

String welcome2 = " \*\*Hello, " + from\_name + "!\*\* \n\n"; welcome2 += "Dont worry.I’m your troubleshoot assistant here to help you! \n\n"; welcome2 += "Here’s what I can do for you:\n\n"; welcome2 += " \*\*/trouble1\*\* - Soil moisture did not increase even after motor on. \n"; welcome2 += " \*\*/trouble2\*\* - waterlevel did not increase after even after motor on \n";

welcome2 += " \*\*/trouble3\*\* - Failed to read from DHT sensor! \n"; welcome2 += " \*\*/trouble4\*\* - Motor doesnot off even soilMoisturePercentage >71%.

\n";

welcome2 += " \*\*/trouble5\*\* - Rain is not detected by system \n"; bot.sendMessage(chat\_id, welcome2, "Markdown");

}

if (text == "/trouble1") {

bot.sendMessage(chat\_id, "⟬SOLUTION ⟭:\n1. Ensure the soil moisture sensor is placed correctly.\n2. If step 1 does not resolve the issue, try replacing the sensor.", "Markdown");

}

if (text == "/trouble2") {

bot.sendMessage(chat\_id, "⟬SOLUTION ⟭:\n1. Ensure the water level sensor is placed correctly.\n2. If step 1 does not resolve the issue, try replacing the sensor.", "Markdown");

}

if (text == "/trouble3") {

bot.sendMessage(chat\_id, "⟬SOLUTION ⟭:\n1. Check the power supply and data pin connected to the D8 pin.\n2. If step 1 does not resolve the issue, try replacing the sensor.", "Markdown");

}

if (text == "/trouble4") {

bot.sendMessage(chat\_id, "⟬SOLUTION ⟭:\n1. Check the relay connections.\n2. If step 1 does not resolve the issue, try replacing the relay.", "Markdown");

} if (text == "/trouble5") { bot.sendMessage(chat\_id, "⟬SOLUTION ⟭:\n1. Check the connections of the rainwater sensor.\n2. If step 1 does not resolve the issue, try replacing the sensor.", "Markdown");

}

} }

void loop() { // Read rain sensor

changeMux(LOW, LOW, HIGH, LOW); int rainSensorValue = analogRead(A0); if (rainSensorValue < 900) {

Serial.println(" Rain detected. Take precautions!");

Serial.println(rainSensorValue); bot2.sendMessage(CHAT\_ID," Rain detected! Take necessary precautions.", ""); //delay(10000);

} else {

Serial.println(" No rain detected.");

Serial.println(rainSensorValue);

}

// Read soil moisture

changeMux(LOW, LOW, LOW, LOW);

int soilMoistureValue = analogRead(A0);

int soilMoisturePercentage = map(soilMoistureValue, MOISTURE\_LOW, MOISTURE\_HIGH, 0, 100);

Serial.print(" Soil Moisture Percentage: ");

Serial.print(soilMoisturePercentage);

Serial.println("%");

// Check if motor is ON if (status == HIGH) {

// Track the time when the motor is turned on if (motorOnTime == 0) {

motorOnTime = millis(); // Set the time when the motor was turned on

previousSoilMoistureValue = soilMoistureValue; // Record the initial soil moisture value

}

// Check if soil moisture has not increased after 30 seconds if (millis() - motorOnTime >= 30000) { if (soilMoisturePercentage < 50) {

Serial.println(" Warning: Soil moisture did not increase even after motor turned on."); bot3.sendMessage(CHAT\_ID," Warning: Soil moisture did not increase even after water supply. for help go to https://t.me/MyFarmUpdatesBot and give command:/help", "");

}

}

// Check if water level is less than 1 after 1 minute

if (millis() - motorOnTime >= 30000) { changeMux(LOW, LOW, LOW, HIGH); int waterLevelValue = analogRead(A0);

float waterLevel = map(waterLevelValue, 350, 520, 1, 4); if (waterLevel < 1) {

Serial.println(" Warning: Water level sensor value indicates the water level is less than 1cm even after motor turn on."); bot3.sendMessage(CHAT\_ID," Warning: Water level sensor value indicates the water level is less than 1cm even after motor turn on. for help go to https://t.me/MyFarmUpdatesBot and give command:/help", "");

}

}

} else {

// Reset the motorOnTime variable when the motor is turned off

motorOnTime = 0;

}

// Control the motor based on soil moisture

if (soilMoisturePercentage < MOISTURE\_THRESHOLD&&rainSensorValue>900) { Serial.println(" Soil is DRY(No rain Detected). Motor is ON to water the crops."); digitalWrite(RELAY\_PIN, HIGH);

} else {

digitalWrite(RELAY\_PIN, LOW);

}

if (soilMoisturePercentage < MOISTURE\_THRESHOLD&&rainSensorValue<900) { Serial.println(" Soil is DRY and Rain Detected (MOTOR OFF).");

bot2.sendMessage(CHAT\_ID," Soil is DRY and Rain Detected (MOTOR OFF)", "");

}

if (soilMoisturePercentage > MOISTURE\_THRESHOLD) { Serial.println(" Soil is WET. Motor is OFF .");

digitalWrite(RELAY\_PIN, LOW);

}

// Read temperature and humidity float humidity = dht\_sensor.readHumidity(); float temperature\_C = dht\_sensor.readTemperature();

// Check if readings are successful

if (isnan(temperature\_C) || isnan(humidity)) {

Serial.println(" Failed to read from DHT sensor!");

bot3.sendMessage(CHAT\_ID," Failed to read from DHT sensor! for help go to @MyFarmUpdatesBot and give command:/help","");

} else {

Serial.print(" Temperature: ");

Serial.print(temperature\_C);

Serial.println("°C");

Serial.print(" Humidity: ");

Serial.print(humidity);

Serial.println("%");

}

if(humidity>80){

bot2.sendMessage(CHAT\_ID,"humidity>75% high chances of rain! ", ""); // delay(100000);

}

// Read water level

changeMux(LOW, LOW, LOW, HIGH); int waterLevelValue = analogRead(A0);

float waterLevel = map(waterLevelValue, 350, 520, 1, 4);

Serial.print(" Water Level: ");

Serial.print(waterLevel);

Serial.println(" cm"); if (waterLevelValue < 350) {

waterLevel = 0; // Set water level to 0 if sensor value is less than 350

Serial.print(" Water Level: ");

Serial.print(waterLevel);

Serial.println(" cm");

}

// Control the LEDs based on water level if (waterLevelValue < 350) { digitalWrite(red, HIGH); digitalWrite(yellow, LOW); digitalWrite(green, LOW);

bot2.sendMessage(CHAT\_ID," Low water level!", "");

//delay(100000);

Serial.println(" Low water level!"); } else if (waterLevelValue > 500) { digitalWrite(red, LOW); digitalWrite(yellow, LOW); digitalWrite(green, HIGH);

Serial.println(" Water level is full.");

} else {

digitalWrite(red, LOW); digitalWrite(yellow, HIGH); digitalWrite(green, LOW);

Serial.println(" Medium water level.");

}

delay(1000); // Delay to avoid hitting API rate limits

// Handle Telegram bot requests

if (millis() > lastTimeBotRan + botRequestDelay) {

int numNewMessages = bot.getUpdates(bot.last\_message\_received + 1); while (numNewMessages) { handleNewMessages(numNewMessages);

numNewMessages = bot.getUpdates(bot.last\_message\_received + 1);

}

lastTimeBotRan = millis();

}

}

### 5. CONCLUSION AND FUTURE RECOMMENDATIONS

**5.1 Conclusion**

In conclusion, the IoT-based smart agriculture monitoring system offers a practical and effective solution for farmers to monitor and optimize the growth of their crops**Sensor Integration:** Successful integration of soil moisture, water evel, rain detection, temperature, and humidity sensors with NodeMCU for real-time monitoring.

* **Data Management:** Efficient data storage and analysis using ThingSpeak, facilitating remote monitoring.
* **Remote Control:** Implementation of a Telegram bot for seamless remote monitoring and control.
* **Notification System:** Effective alerts for rain detection and low soil moisture, ensuring timely interventions.
* **Impact on Farming:** Improved water usage efficiency, increased crop yield, and reduced labor costs.

Looking to the future, there is great potential for further advancements in smart agriculture technology. For example, integrating artificial intelligence and machine learning algorithms could improve the accuracy of data analysis and prediction of crop growth and yield. Additionally, incorporating drone technology for aerial monitoring of crops could provide even more detailed insights into crop health and growth.

**5.2 Future Recommendation**

In terms of recommendations, it is important for further research and development to focus on making smart agriculture technology more accessible and affordable for small-scale farmers who may not have the resources to invest in expensive equipment. Additionally, efforts should be made to address any potential privacy and security concerns related to the collection and storage of sensitive agricultural data.

Overall, the IoT-based smart agriculture monitoring system has the potential to revolutionize the way we approach farming and food production, offering a more sustainable, efficient, and profitable approach to agriculture.

### References

1. Rawal, S. (2017). IOT based smart irrigation system. International Journal of Computer Applications, 159(8), 7-11.
2. García, L., Parra, L., Jimenez, J. M., Lloret, J., & Lorenz, P. (2020). IoT-based smart irrigation systems: An overview on the recent trends on sensors and IoT systems for irrigation in precision agriculture. Sensors, 20(4), 1042.
3. Nawandar, N. K., & Satpute, V. R. (2019). IoT based low cost and intelligent module for smart irrigation system. Computers and electronics in agriculture, 162, 979-990.
4. Saraf, S. B., & Gawali, D. H. (2017, May). IoT based smart irrigation monitoring and controlling system. In 2017 2nd IEEE International Conference on Recent Trends in

Electronics, Information & Communication Technology (RTEICT) (pp. 815-819). IEEE [5] Shekhar ,Y. Dagur, E. Mishra, S. & Sankaranarayanan, S. (2017). Intelligent IoT based automated irrigation system. International Journal of Applied Engineering Research,

12(18), 7306-7320

1. Pernapati, K. (2018, April). IoT based low cost smart irrigation system. In 2018 Second

International Conference on Inventive Communication and Computational Technologies (ICICCT) (pp. 1312-1315). IEEE.

1. Kansara, K., Zaveri, V., Shah, S., Delwadkar, S., & Jani, K. (2015). Sensor based automated irrigation system with IOT: A technical review. International Journal of Computer Science and Information Technologies, 6(6), 5331-5333.
2. Rajkumar, M. N., Abinaya, S., & Kumar, (2017, March). Intelligent irrigation system An IOT based approach. In 2017 International Conference on Innovations in Green Energy and Healthcare Technologies (IGEHT) (pp. 1-5). IEEE.
3. Obaideen, K., Yousef, B. A., AlMallahi, M. N., Tan, Y. C., Mahmoud, M., Jaber, H., & Ramadan, M. (2022). An overview of smart irrigation systems using IoT. Energy Nexus, 100124.
4. Rau, A. J., Sankar, J., Mohan, A. R., Krishna, D. D., & Mathew, J. (2017, July). IoT based smart irrigation system and nutrient detection with disease analysis. In 2017 IEEE

Region 10 Symposium (TENSYMP) (pp. 1-4). IEEE. 66

1. Goap, A., Sharma, D., Shukla, A. K., & Krishna, C. R. (2018). An IoT based smart irrigation management system using Machine learning and open source technologies.

Computers and electronics in agriculture, 155, 41-49.

1. Krishnan, R. S., Julie, E. G., Robinson, Y. H., Raja, S., Kumar, R., & Thong, P. H.

(2020). Fuzzy logic based smart irrigation system using internet of things. Journal of Cleaner Production, 252, 119902.

1. Mishra, D., Khan, A., Tiwari, R., & Upadhay, S. (2018, February). Automated irrigation system-IoT based approach. In 2018 3rd International conference on internet of things: Smart Innovation and Usages (IoT-SIU) (pp. 1-4). IEEE. 56 [14] Benyezza, H., Bouhedda, M., Djellout, K., & Saidi, A. (2018, November). Smart irrigation system based

ThingSpeak and Arduino. In 2018 International conference on applied smart systems (ICASS) (pp. 1-4). IEEE.

1. VinothKumar, V., Ramasamy, R., Janarthanan, S., & Vasim Babu, M. (2017).

Implementation of IOT in smart irrigation system using arduino processor. International Journal of Civil Engineering and Technology (IJCIET).

1. Nandhini, R., Poovizhi, S., Jose, P., Ranjitha, R., & Anila, S. (2017, March). Arduino based smart irrigation system using IoT. In 3rd National Conference on Intelligent Information and Computing Technologies (IICT ‘17) (pp. 1-5).
2. Ragab, M. A. (2022). IoT based smart irrigation system. International Journal of Industry and Sustainable Development, 3(1), 76-86.