**ABSTRACT**

Smart agriculture systems leverage advanced versions such as the Internet of Things, sensor networks, and data visualisation methods to optimize farming practices, improve crop yield, and reduce resource consumption. These systems integrate various sensors to monitor environmental parameters such as soil moisture, temperature, humidity, and light intensity. The data which is collected data is analyzed in real-time to provide actionable insights for farmers, enabling precision agriculture. Automated irrigation systems can adjust watering schedules based on soil moisture levels, ensuring optimal water usage. Additionally, smart agriculture systems can include pest detection and weather forecasting capabilities, allowing for timely interventions and better risk management. By utilizing these technologies, farmers can enhance productivity, minimize waste, and promote sustainable farming practices, ultimately contributing to food security and environmental conservation. The paper's feature involves creating a system that can monitor temperature, humidity, moisture, and animal movement in agricultural fields using Arduino sensors. It will send SMS and app notifications to the farmer's smartphone in case of any issues, using Wi-Fi/3G/4G.

Keywords: Temperature , Humidity ,Soil moisture,Light Intensity

**1. INTRODUCTION**

Traditional farming is facing unknown challenges in meeting the growing global demand for food, water by factors such as climate change, water scarcity, and population growth. In response, there is a pressing need for agricultural systems that are not only productive but also sustainable. The emergence of IoT technologies offers a solution by enabling the creation of smart agriculture systems capable of monitoring, analyzing, and optimizing various farm parameters in real-time. This paper presents an overview of such a system, designed to address the complexities of modern agriculture while making environmental atmosphere and economic viability.

With the growth of world population, according to the UN Food and Agriculture Organization, the world will need to produce 70% more food in 2050, shrinking agricultural lands, and depletion of finite natural resources, We need to enhance farm yield has become critical. Limited availability of natural resources such as fresh water and fertile land along with slowing yield trends in several staple crops, have further aggravated the problem. Another concern over the farming industry is the shifting structure of agricultural workforce. Moreover, agricultural labor in most of the countries has declined. As a result of the declining agricultural workforce, adoption of internet connectivity solutions in farming practices has been triggered, to reduce the need for manual labor. IoT solutions are focused on helping farmers close the supply demand gap, by ensuring high yields, profitability, and protection of the environment.

Precision agriculture, leveraging IoT technology, optimizes resource usage for enhanced crop yields and reduced operational expenses. This approach integrates special equipment, wireless connections,software used.

Based on a survey by BI Intelligence, it was anticipated that the adoption of IoT devices in agriculture would reach 75 million by 2020, with a 20% annual growth rate. Concurrently, the global smart agriculture market is expected to triple by 2025, reaching $15.3 billion, compared to just over $5 billion in 2016.



To identify plant monitoring to obtain the best method of smart agriculture system and to monitor few parameters for the growth of plants.

• To develop an algorithm in-plant monitoring system such as temperature, humidity, automatic irrigation system, detect the motion activity and rain status.

• To develop Smart Monitoring Plant Using IoT System.

The project focuses on utilizing the ESP32 as an environmental monitoring system, specifically for plant care. It incorporates a soil moisture detector to regulate plant watering based on moisture levels. Acting as the central control server, the ESP32 orchestrates the entire system, connecting to essential sensors for maintaining an ideal plant environment. Plant health status is monitored through Blynk apps, with an optimal pH range of 5.5 to 6.5 for plant thriving. The ESP32 gathers data from sensors, stores it, and facilitates integration of hardware and software components for real-time plant status monitoring via the Blynk server, ensuring prompt assessment of plant health.



Despite considerable government efforts, small-scale farmers have not reaped the benefits, resulting in a significant gap between them and large-scale farmers, leading to an imbalance. To address this disparity, systematic planning is essential, requiring detailed regional information. In many countries like India, where farming sustains a majority of the population and contributes significantly to the national income, outdated agricultural practices persist despite the widespread availability of modern technology. Small farmers continue to rely on traditional methods such as manual seed distribution, limited cropping patterns, and outdated cultivation systems. The irregularity of monsoons and uneven water availability throughout the year further compound these challenges

**2.REQUIRED SPECIFICATION**

**Hardware components:**

1.Node MCU

2.DHT 11

3.Relay module

4.Rain Sensor

5.Soil moisture Sensor

6.PIR Sensor

7.pump

8.Bread board

9.Battery

10.UV Light

11.A Glass Box

**Software components:**

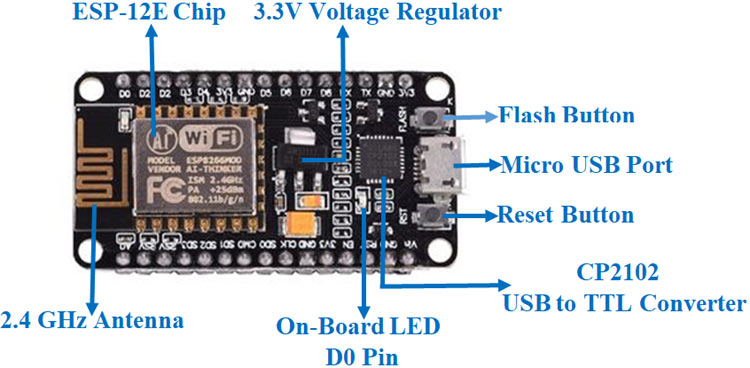
1.Windows 11

2.Arduino ide

3.Blink App

**2.1 NODE MCU ESP8266**

The NodeMCU development board showcases the ESP-12E module housing the ESP8266 chip, featuring a Tensilica Xtensa 32-bit LX106 RISC microprocessor. This versatile chip operates at a clock frequency range of 80MHz to 160MHz, supporting real-time operating systems, and offers 128 KB of RAM and 4MB of Flash memory for ample storage. With built-in Wi-Fi and Bluetooth capabilities, it's an ideal choice for Internet of Things (IoT) projects and can be powered via its Micro USB jack or the VIN pin.offering versatility in power sources. Additionally, it supports multiple communication interfaces including UART, SPI, and I2C, further enhancing its adaptability for various project requirements.

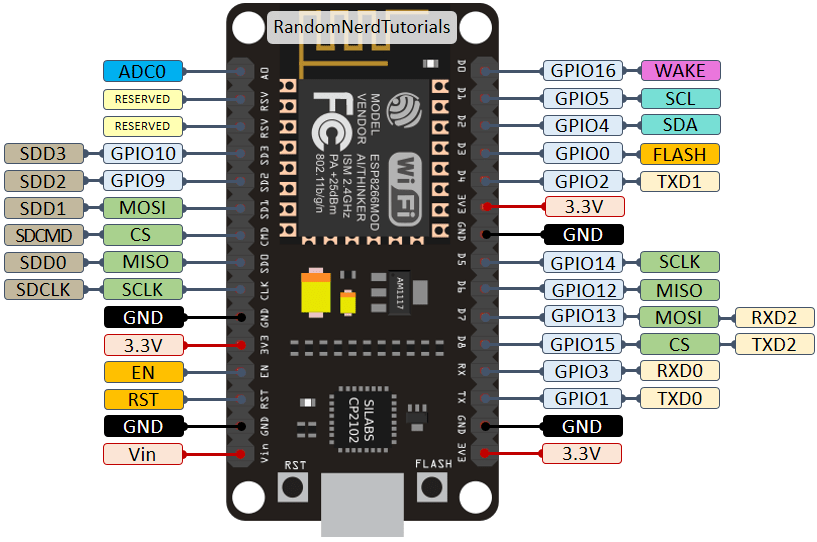


### **2.2 NodeMCU ESP8266 Specifications & Features**

* Microcontroller used : Tensilica 32-bit RISC CPU Xtensa LX106
* Range of operating voltage : 3.3V
* Input Voltage Range: 7-12V
* Digital Input/OutputPins (DIO): 16
* Analog Input Pins in : 1
* UART: 1
* SPI: 1
* I2C: 1
* Flash Memory: 4 MB
* SRAM: 64 KB
* Clock Speed: 80 MHz
* USB-TTL based on CP2102 onboard, enabling plug and play functionality.

|  |  |  |
| --- | --- | --- |
| **Pin Category** | **Name** | **Description** |
| Power | Micro-USB, 3.3V, GND, Vin | **Micro-USB:** NodeMCU can be powered through the USB port  **3.3V:** Regulated 3.3V can be supplied to this pin to power the board  **GND:** Ground pins  **Vin:**External Power Supply |
| Control Pins | **EN, RST** | The pin and the button resets the microcontroller |
| Analog Pin | A0 | Used to measure analog voltage in the range of 0-3.3V |
| GPIO Pins | GPIO1 to GPIO16 | NodeMCU has 16 general purpose input-output pins on its board |
| SPI Pins | SD1, CMD, SD0, CLK | NodeMCU has four pins available for SPI communication. |
| UART Pins | TXD0,RXD0, TXD2, RXD2 | NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program. |
| I2C Pins |  | NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C. |

This NodeMCU consists of total 30 pins and each pin has its own specification.



*Fig4: ESP8266 NodeMCU pinout*

Vin can be used to directly supply the NodeMCU/ESP8266 and its peripherals. Power delivered on VIN is regulated through the onboard regulator on the NodeMCU module – you can also supply 5V regulated to the VIN pin. 3.3V pins are the output of the onboard voltage regulator and can be used to supply power to external components.

**Applications**

* Prototyping of IoT device.
* Low power battery operated applications
* Network projects. Projects requiring multiple I/O interfaces with Wi-Fi and Bluetooth functionalities.

**2.3 DHT11**

**T**he DHT11 sensor measures temperature and humidity, two critical parameters in agriculture. By deploying these sensors throughout a farm or greenhouse, farmers can monitor the microclimate in different areas. This data can help optimize growing conditions for various crops, ensure proper humidity levels for seed germination, and prevent issues like mold or fungal growth. Different areas within a farm or greenhouse may have distinct microclimates due to factors like shading, topography, or proximity to water sources. DHT11 sensors can help identify these variations, allowing farmers to adjust planting schedules, crop selection, and cultivation practices accordingly. By continuously monitoring humidity with DHT11 sensors, farmers can implement preventive measures such as adjusting ventilation, spacing plants, or applying fungicides when necessary

This sensor does have 4 pins:

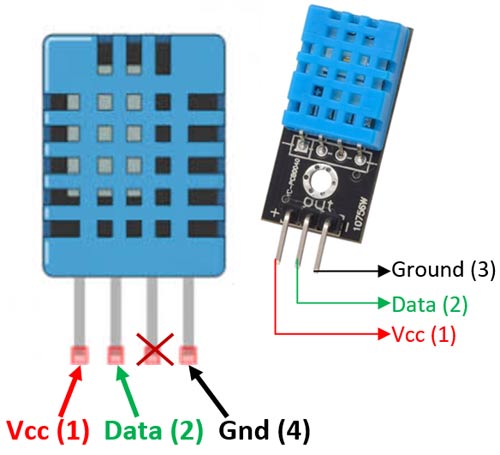
1) Vcc : Power supply 3.5V to 5.5V.

2) Data : Outputs both Temperature and Humidity

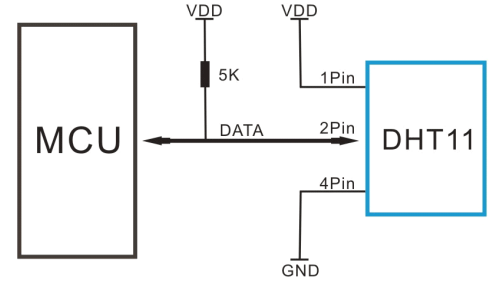
through serial Data.

3) NC : No Connection and hence not used

4) Ground: Connected to the ground of the circuit



As we can see the data pin is connected to an I/O pin of the MCU and a 5K pull-up resistor is used. This data pin outputs the value of both temperature and humidity as serial data. Through this connection DHT11 values get read by the NodeMCU.



As we can see the data pin is connected to an I/O pin of the MCU and a 5K pull-up resistor is used. This data pin outputs the value of both temperature and humidity as serial data. Through this connection DHT11 values get read by the NodeMCU.

DHT11 Specifications:

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

DHT11 Applications:

* Measure temperature and humidity
* Local Weather station
* Automatic climate control
* Environment monitoring

**2.4 Relay Module:**

A relay is an electromechanical switch that is controlled by an electrical signal. It consists of a coil (electromagnet) and one or more sets of contacts. When current flows through the coil, it generates a magnetic field that activates the switch, allowing current to flow through the contacts. Relays are commonly used to control high-power devices with low-power signals from microcontrollers or other electronic circuits.



Pin Configuration:

Relay modules typically have several pins for connecting to external circuits and power sources. Here's a typical pin configuration:

**VCC:** This pin is used to supply power to the relay module. It's usually connected to a 5V or 3.3V power source.

**GND:** This pin is the ground connection for the relay module. It's connected to the ground (0V) of the power supply.

**Signal Input:** This pin is used to control the relay. Applying a voltage or logic signal to this pin activates the relay.

**Normally Open (NO) and Normally Closed (NC) Contacts:** These pins are the switching contacts of the relay. They are usually labelled as NO and NC and are connected to the load (device) that you want to control.

**COM (Common):** This pin is common to both the NO and NC contacts. It's typically connected to one side of the power supply or load.

**Optional Pins**: Some relay modules may have additional pins for features like indicator LEDs, protection diodes, or additional control options.

Applications of Relay module:

Home automation

Industrial automation

Safety system

Security system

**2.5 Soil Moisture sensor**

The Soil Moisture sensor is used to sense moisture content in the soil. It checks the volume of water content or moisture present in the soil. The calculations are done in the soil moisture sensor through coefficients. It estimates the volume of water content in the soil. It detects the water content in the soil and gets and sends the analog signals which is shown digitally.

It transmits the signals containing information or data or values of the condition of soil to Arduino to further process it and display. Soil moisture sensors help farmers avoid waterlogging and soil salinity issues by monitoring moisture levels and drainage conditions in the soil. Soil moisture sensors provide real-time data on soil moisture levels at various depths. Farmers can use this information to determine when and how much to irrigate, ensuring that crops receive the right amount of water at the right time.



Pin configuration:

**VCC (or VDD):** This is the power supply pin. It is typically connected to a 3.3V or 5V pin on the microcontroller or power source.

**GND (or Ground):** This is the ground pin. It is connected to the ground or GND pin of the microcontroller or power source.

**Analog Output**: This pin provides an analog voltage output that corresponds to the moisture level detected by the sensor.

It is usually connected to one of the analog input pins (e.g., A0, A1, etc.) on a microcontroller.

**Digital Output** (Optional): Some soil moisture sensors also have a digital output pin that provides a binary signal (usually HIGH or LOW) based on a preset moisture threshold.

This pin can be connected to a digital input pin on a microcontroller for digital readings.

Application of soil moisture sensor:

Precision irrigation

Crop Management

Environment monitoring

Water resource management

**2.6 PIR SENSOR**

PIR sensors, which stands for Passive Infrared sensors, are devices that detect motion by measuring changes in infrared radiation levels in their field of view. They are commonly used in security systems, automatic lighting systems, and other applications where motion detection is needed.



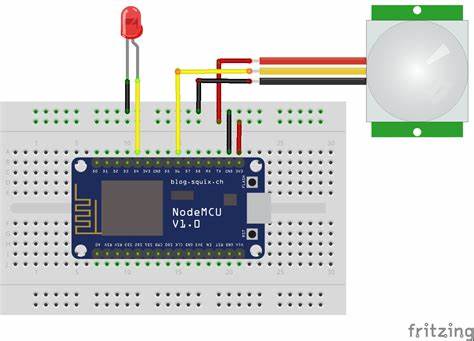
**Pin Configuration:**

**VCC or VDD:** This pin is for the power supply. It's where you connect the positive terminal of your power source, usually 3.3V or 5V, depending on the specifications of your PIR sensor.

**GND**: This pin is for the ground connection. Connect the negative terminal of your power source (ground) to this pin.

**OUT or SIGNAL:** This pin outputs a digital signal when motion is detected. It usually sends a HIGH (or 1) signal when motion is detected and LOW (or 0) when there's no motion.

CIRCUIT :

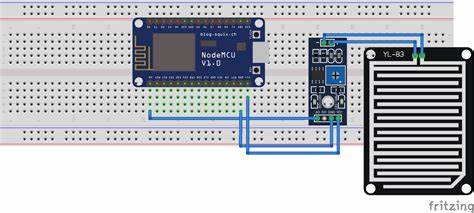


**2.7 RAIN SENSOR:**

A rain sensor, also known as a rain switch or rain detector, is a device used to detect the presence of rain or moisture. It's commonly used in automated weather systems, irrigation systems, and automotive applications such as automatic windshield wipers. The basic principle behind most raindrop sensors involves detecting changes in conductivity or capacitance caused by the presence of water. When raindrops fall on the sensor surface, they alter its electrical properties, which can be measured by the sensor circuitry.



There are different types of raindrop sensors available, ranging from simple ones that trigger when they detect moisture to more sophisticated models that can measure rainfall intensity or accumulate rainfall data over time.



**Pin configuration:**

**VCC (Power Supply):** This pin is used to provide power to the rain sensor. It is usually connected to a positive voltage source, such as +5V or +3.3V, depending on the sensor's specifications.

**GND (Ground):** This pin is the ground connection for the rain sensor. It should be connected to the ground terminal of your power supply or microcontroller.

**Analog Output**: Many rain sensors feature an analog output pin. This pin provides an analog voltage or current signal that varies based on the intensity of rainfall. The output voltage may increase as the amount of rain detected increases.

**Digital Output**: Some rain sensors also offer a digital output pin. This pin typically provides a digital signal, such as a logic high or logic low to indicate the presence or absence of rain. The digital output may be triggered when a certain threshold of rainfall is detected.

**Heater Control (Optional):** In some advanced rain sensor models, there may be a pin for controlling an internal heating element. This heating element helps prevent the accumulation of dew or frost on the sensor's surface, ensuring accurate rain detection in various weather conditions.

**Adjustment/Control (Optional):** Certain rain sensors may feature pins for adjusting sensitivity or threshold levels. These pins allow you to customize the sensor's response to different rainfall intensities or environmental conditions.

**2.8 BATTERY :**

In this project may involve mobile applications for monitoring and management tasks. Tablets, smartphones, or handheld devices used by farmers or agricultural technicians require battery power to operate in the field. batteries play a crucial role in providing power to various components and devices used for monitoring, automation, and data collection. Here's how batteries are typically used in smart agriculture projects:. 

Batteries enable the deployment of remote monitoring and control systems in agricultural settings. Farmers can use smartphones, tablets, or computers to access real-time data and control irrigation systems, fertilization equipment, and other machinery from anywhere, thanks to battery-powered communication

**2.9 BREAD BOARD**

A breadboard is a fundamental tool used in electronics for prototyping and experimenting with circuits. It's essentially a rectangular plastic board with a grid of holes into which you can insert electronic components and connect them without soldering. The holes are typically arranged in rows and columns, and the rows are often connected internally. This allows you to quickly build and modify circuits by inserting components and connecting them with jumper wires



**2.10 JUMPER WIRES**

Jumper wires are essential components in electronics prototyping and circuit building. They are flexible wires with connectors at each end, typically used to connect components on a breadboard or between various points on a circuit. These wires come in various lengths, colors, and types, such as male-to-male, male-to-female, and female-to-female



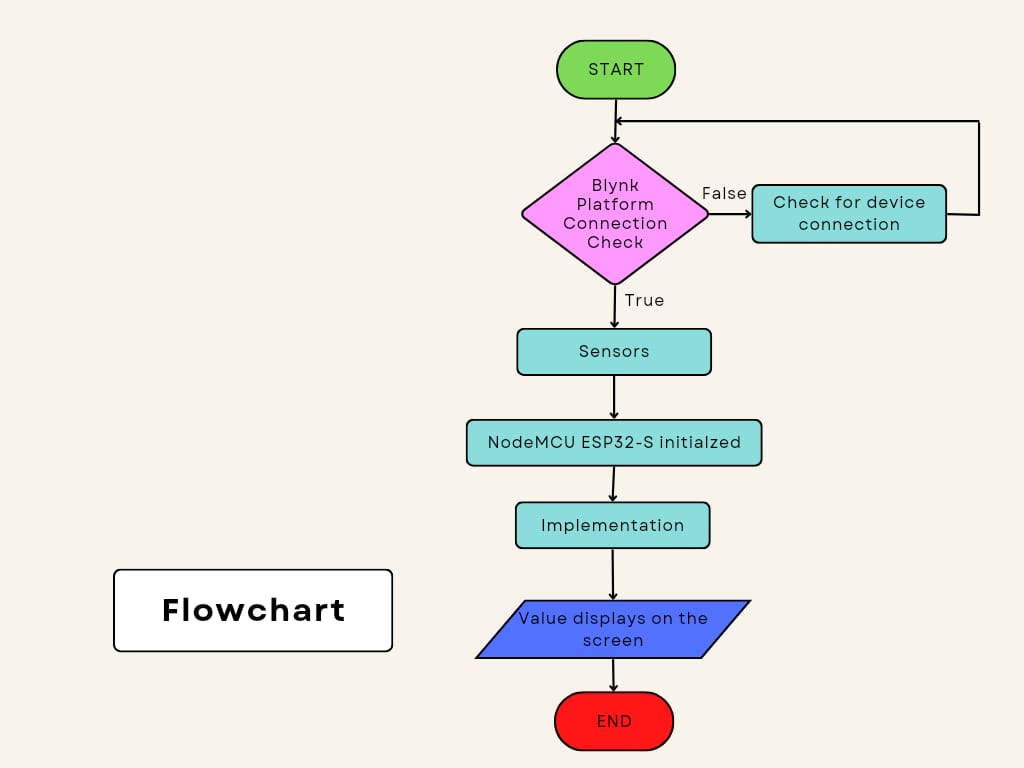
**3. METHODOLOGY**

This document outlines the research methodology employed in the study, detailing the data collection equipment, procedures, and data analysis processes. It also serves as a reference for addressing any future concerns and evaluating the project's overall strategy.

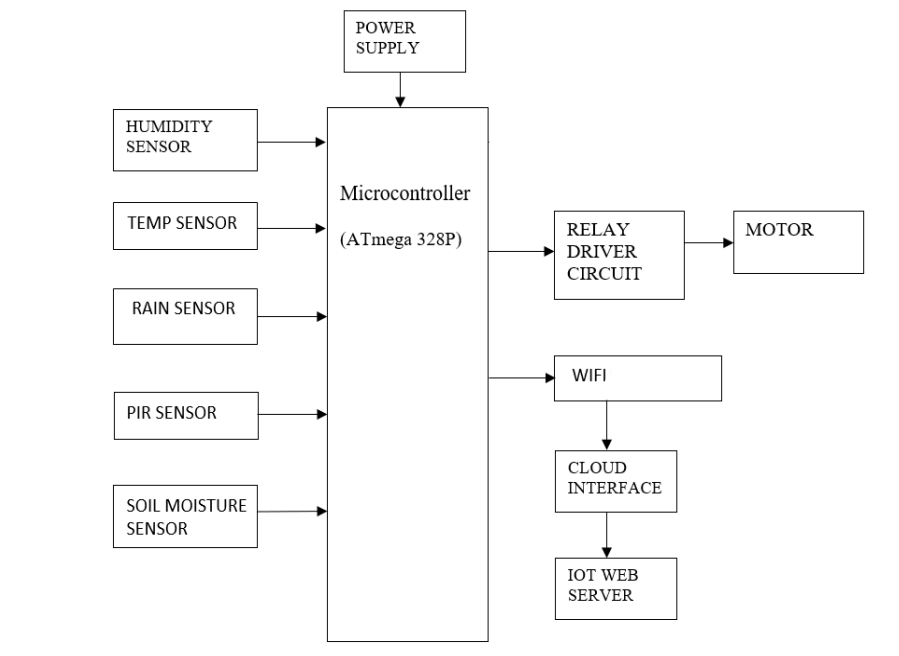
Key hardware components utilized in the study include a soil moisture sensor connected to the NodeMCU ESP32 microcontroller, along with additional sensors such as a rain sensor, PIR sensor, and DHT11 sensor.

The Arduino IDE software is employed to program the microcontroller, enabling it to collect and process data from the sensors. Additionally, Blynk apps are utilized to facilitate the transfer of information and data obtained by the system, ensuring seamless communication with farmers or relevant organizations.

This comprehensive approach ensures the efficient collection, analysis, and dissemination of valuable agricultural data, ultimately contributing to improved decision-making and resource management in agricultural practices.



**3.1 BLOCK DIAGRAM**

****

Here's a structured outline of the methodology:

Needs Assessment and Planning

System Design

Hardware Implementation

Software Development

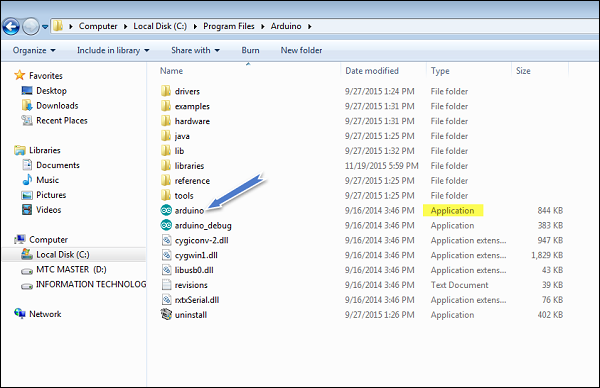
Field Testing and validation

Data Analysis

Evaluation and verification

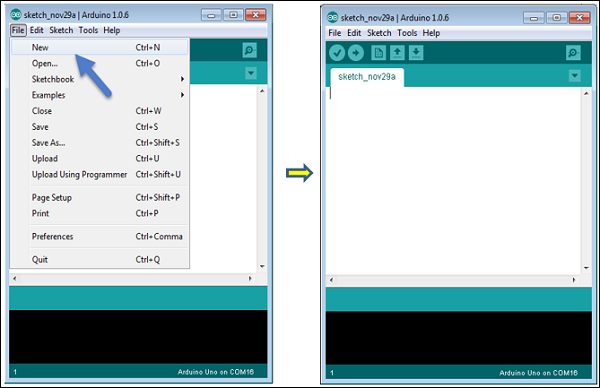
**4.IMPLEMENTATION**

**4.1.1 LAUNCH ARDUINO IDE**

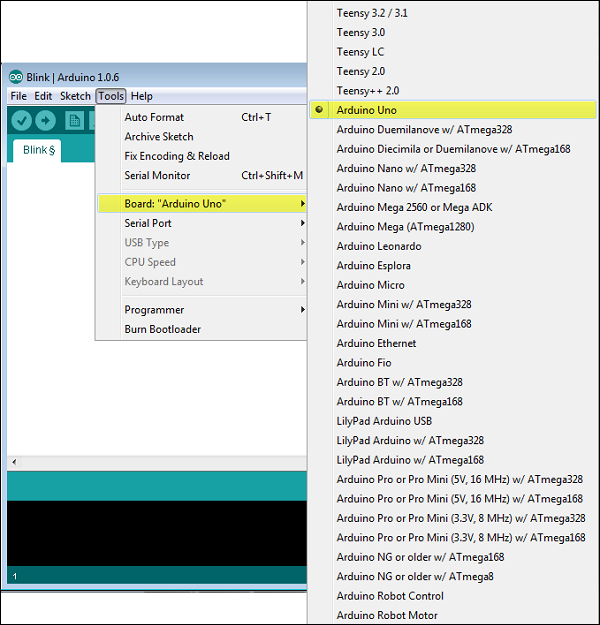


4.1.2 Create a new project.

To create a new project, select File → **New**



4.1.3 Select your Arduino Board



We must avoid any error while uploading a program to the board, we must select the correct Arduino board name, which holds same with the board connected to your pc.

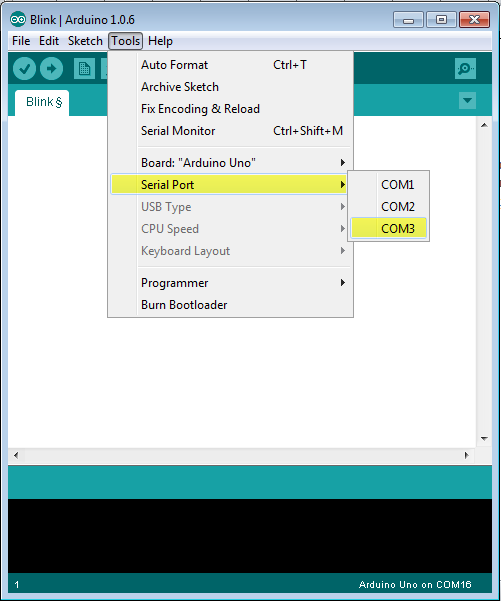
Go to Tools → Board and select your board.

**Select your serial port.**

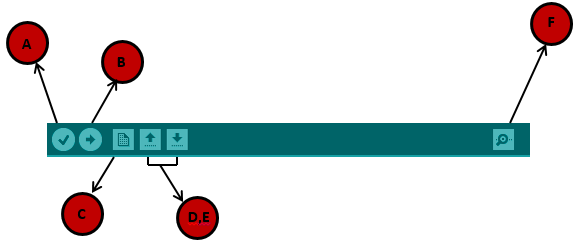
Select the serial device of the Arduino board.

Go to **Tools → Serial Port** menu.

This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.



**Upload the program**



**A** − Used to check if there is any compilation error.

**B** − Used to upload a program to the Arduino board.

**C** − Shortcut used to create a new sketch.

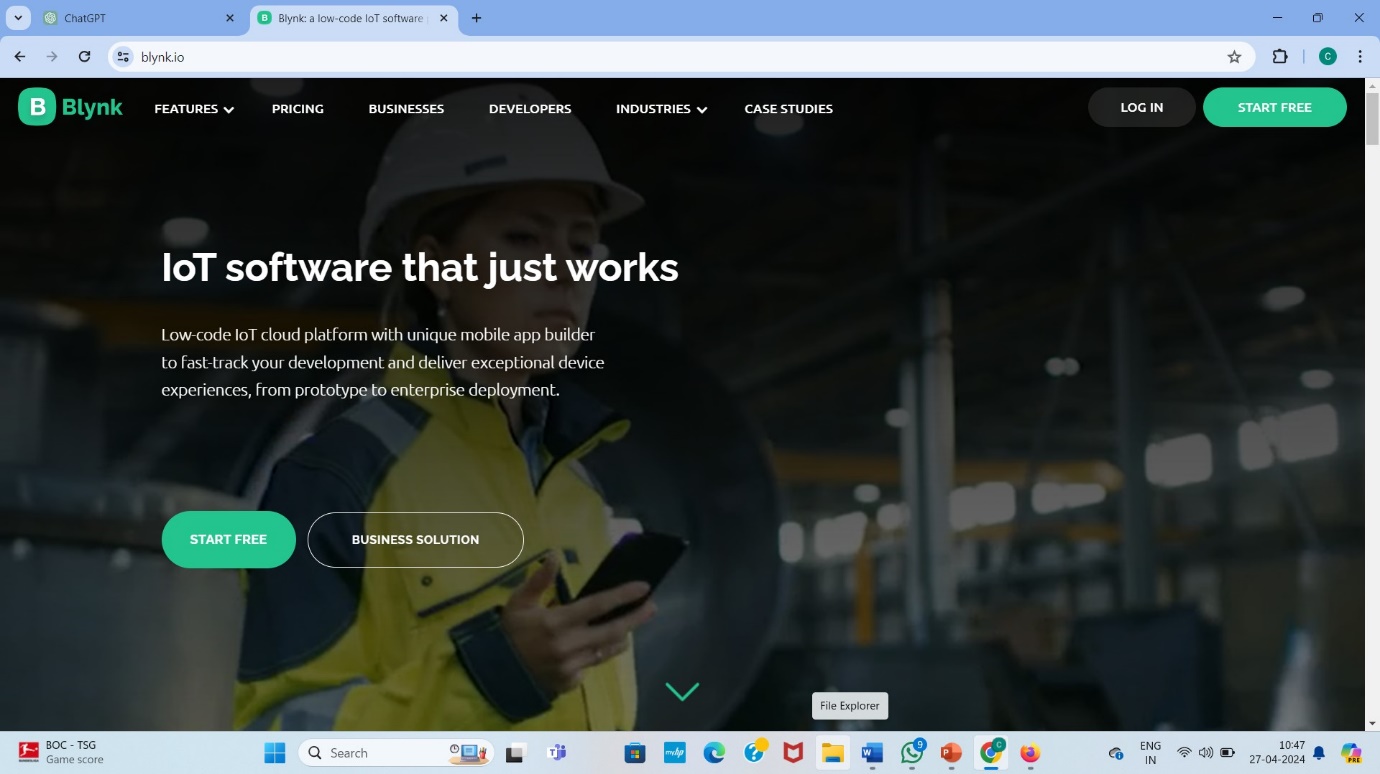
**D** − Used to directly open one of the example sketch.

**E** − Used to save your sketch.

**F** − Serial monitor used to receive serial data from the board and send the serial data to the board.

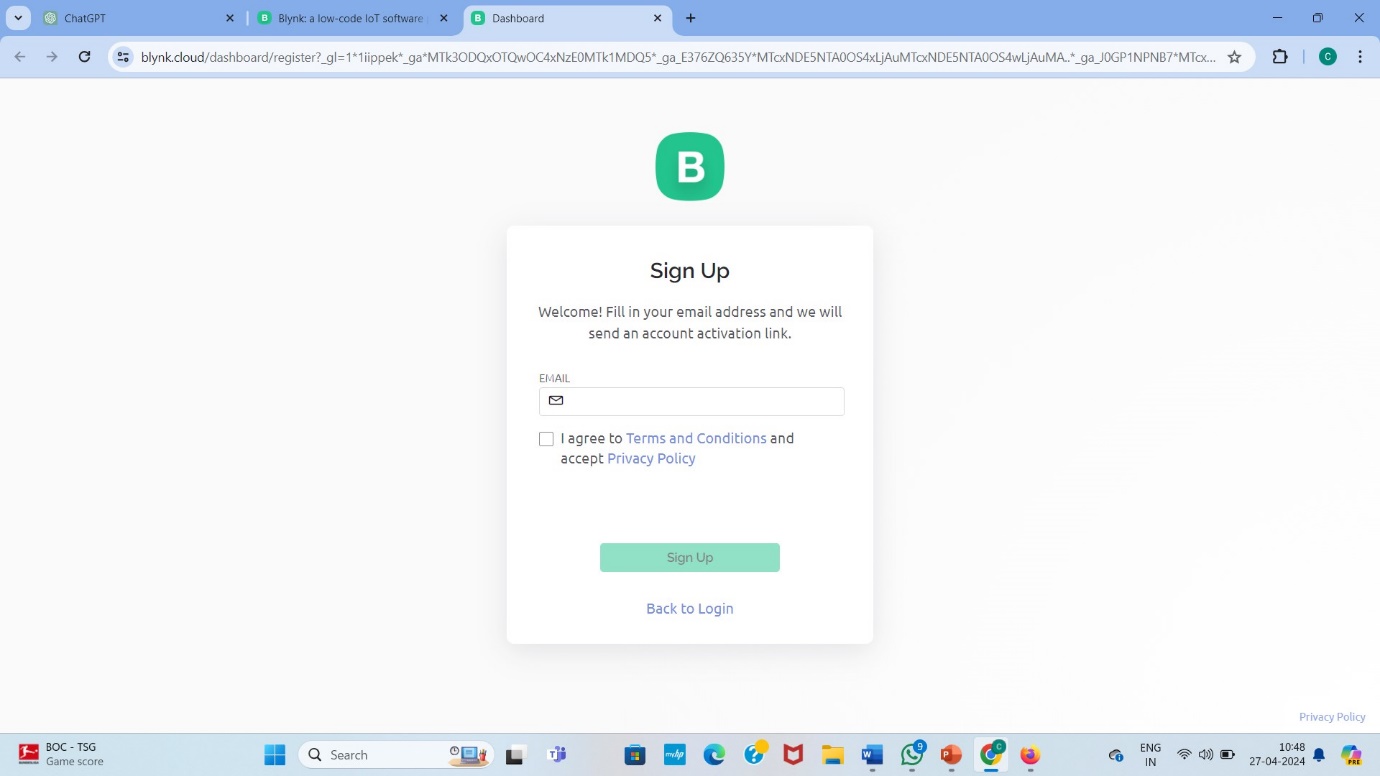
**4.2 BLYNK APP SETUP**

**4.2.1**



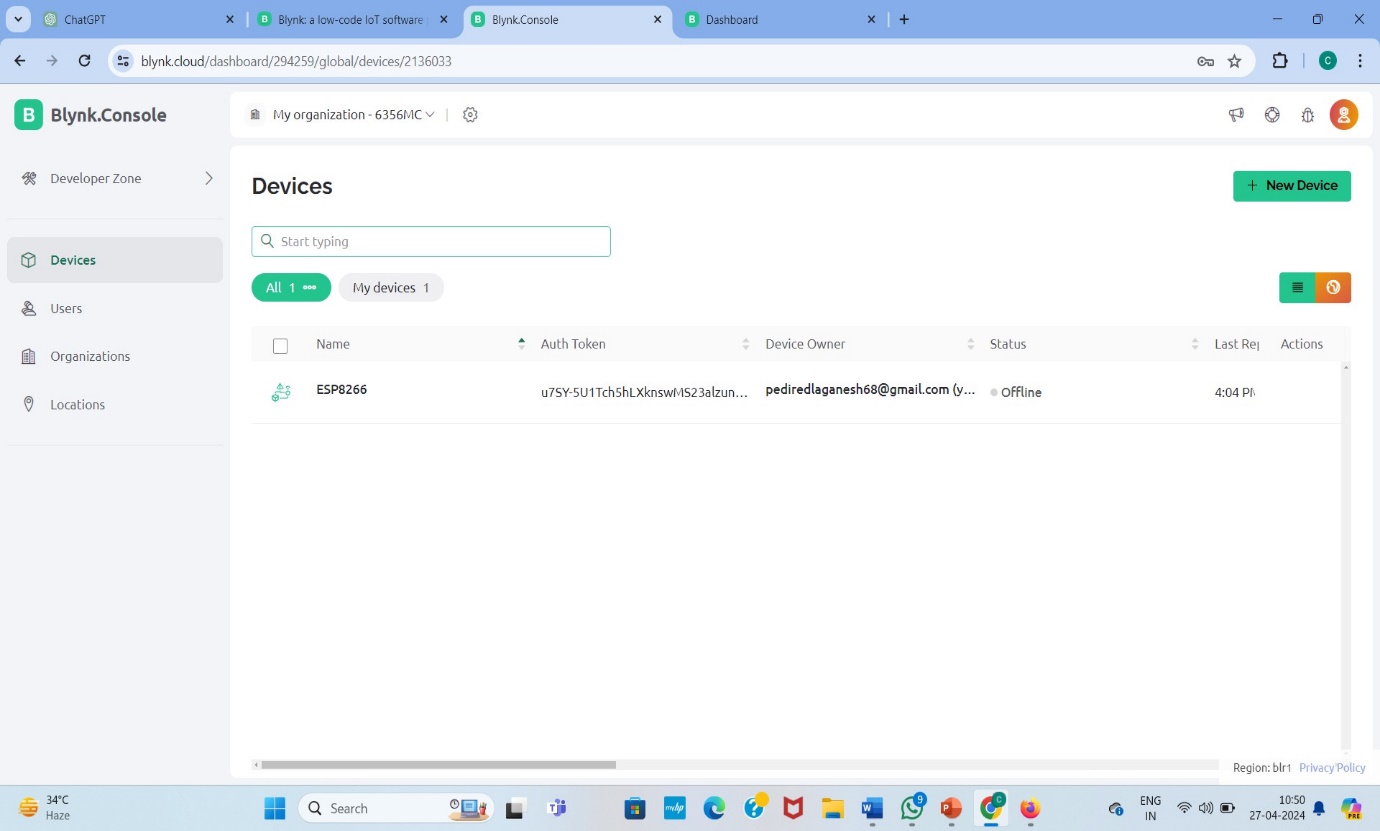
4.2.2

Trying to sign up or if already have a account try to login the page .



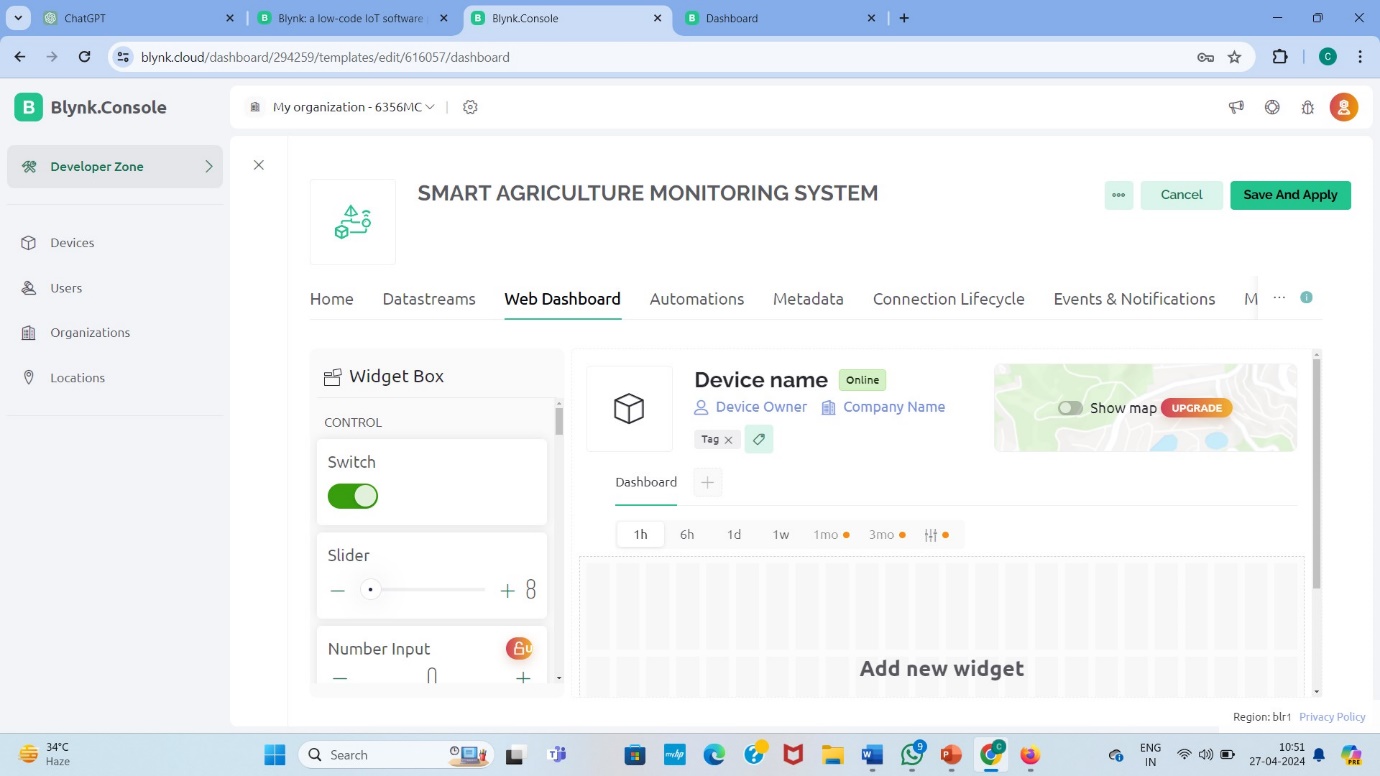
**4.2.3**

**BLYNK CONSOLE ALONG WITH THE DEVICES**



4.2.4

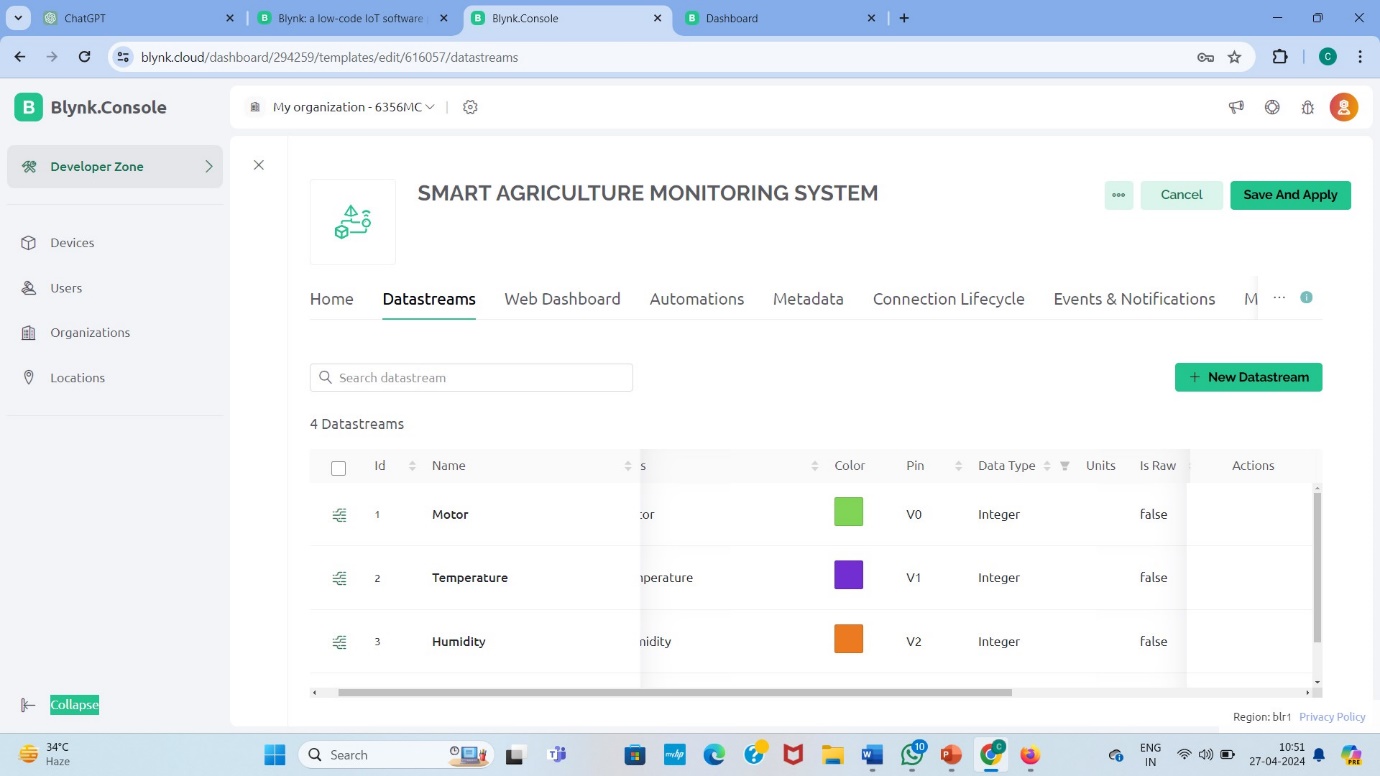
WEB DASH BOARD

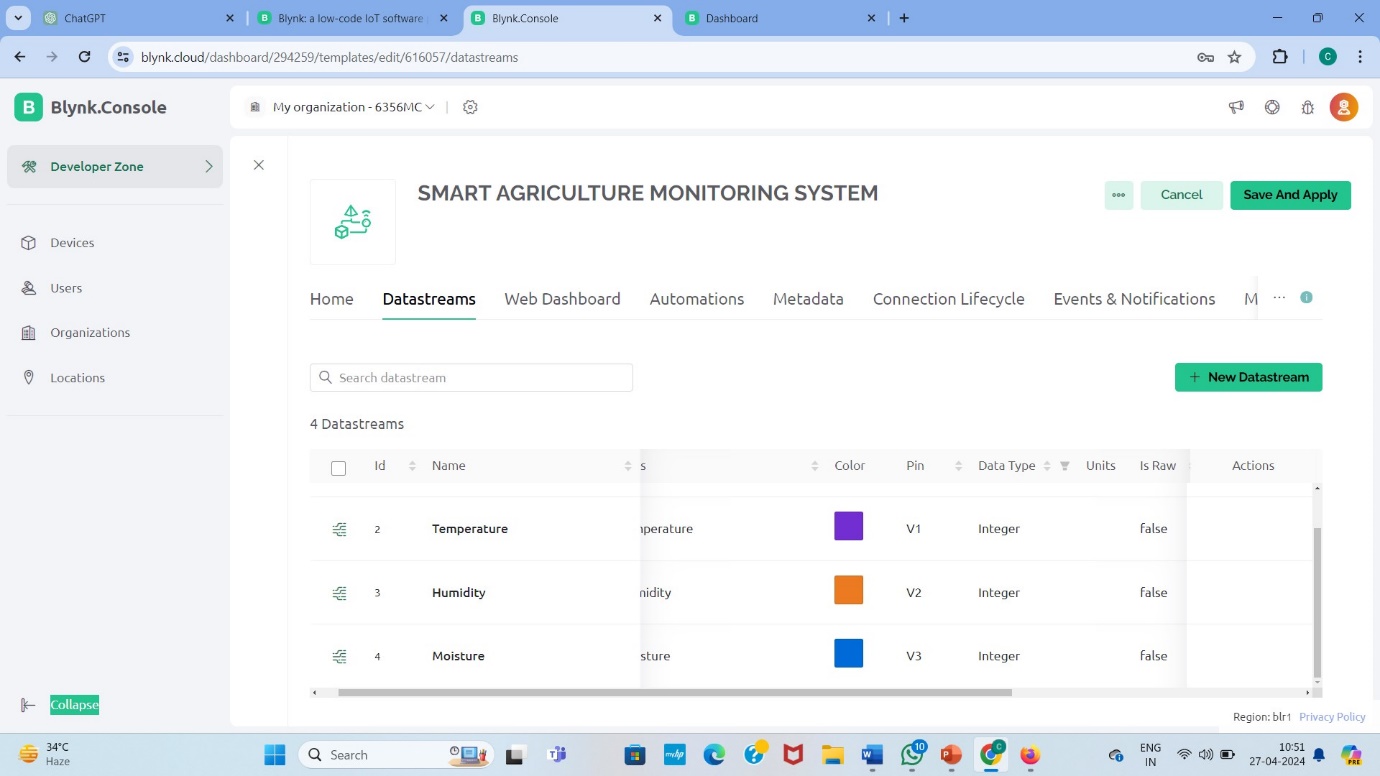


We can get the device name and control the project with a particular switch

4.2.5

DATASTREAMS

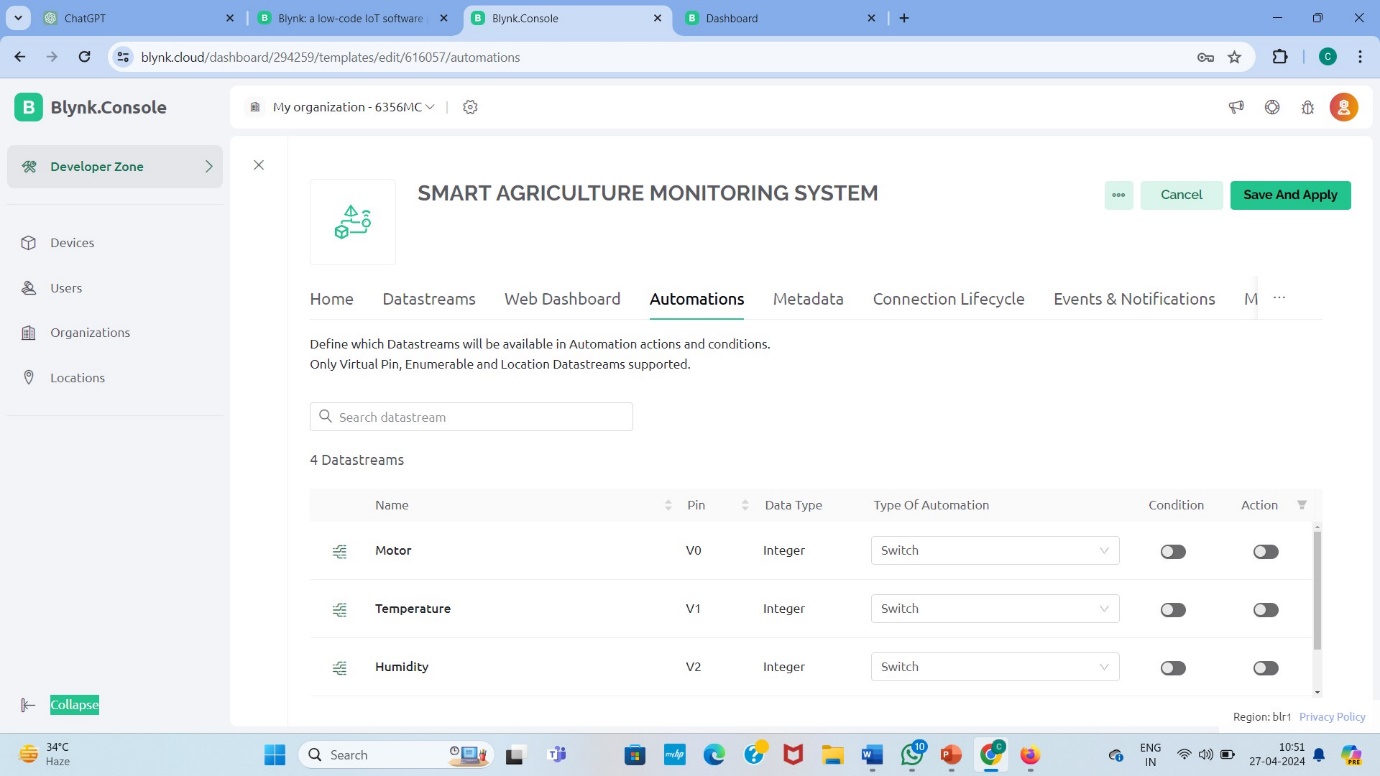




Here we get the components we use and data type used and the action taken can be found.

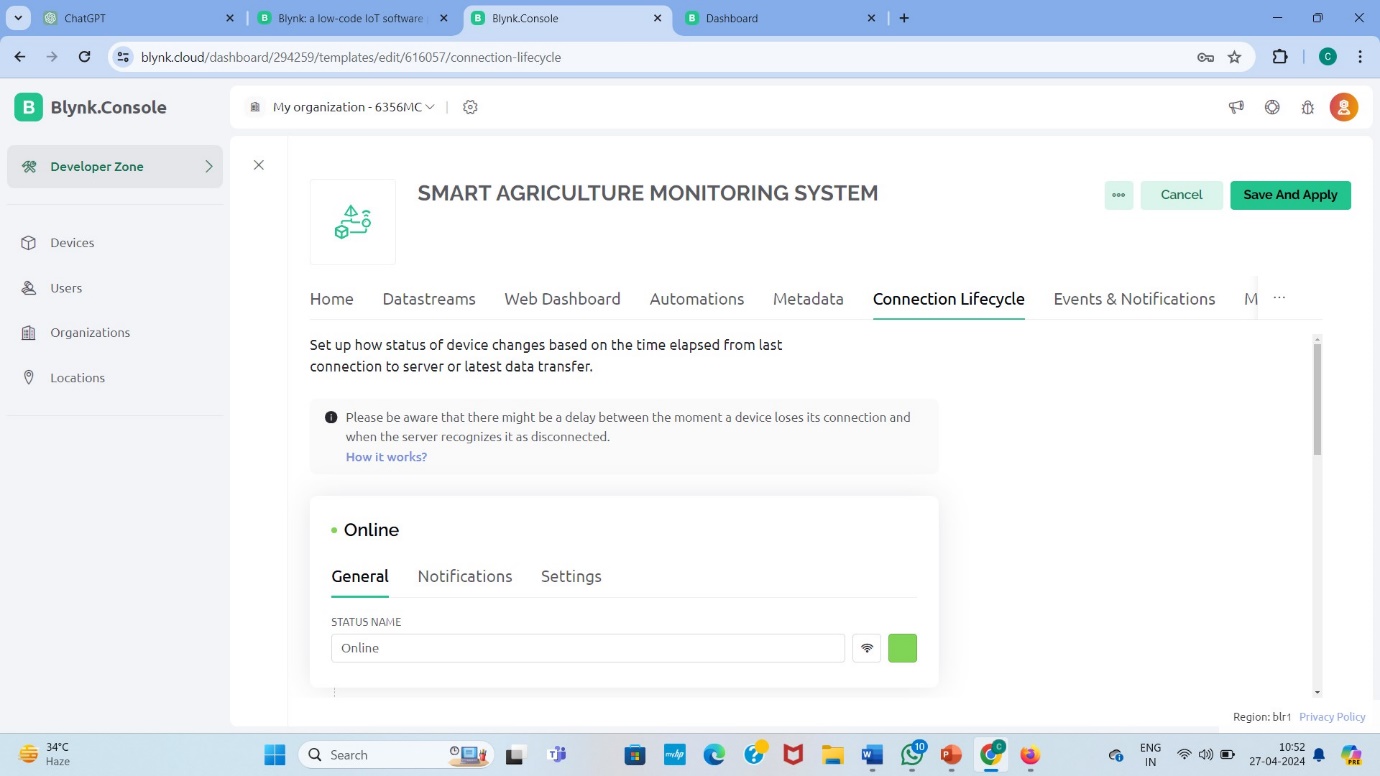
4.2.6

AUTOMATIONS



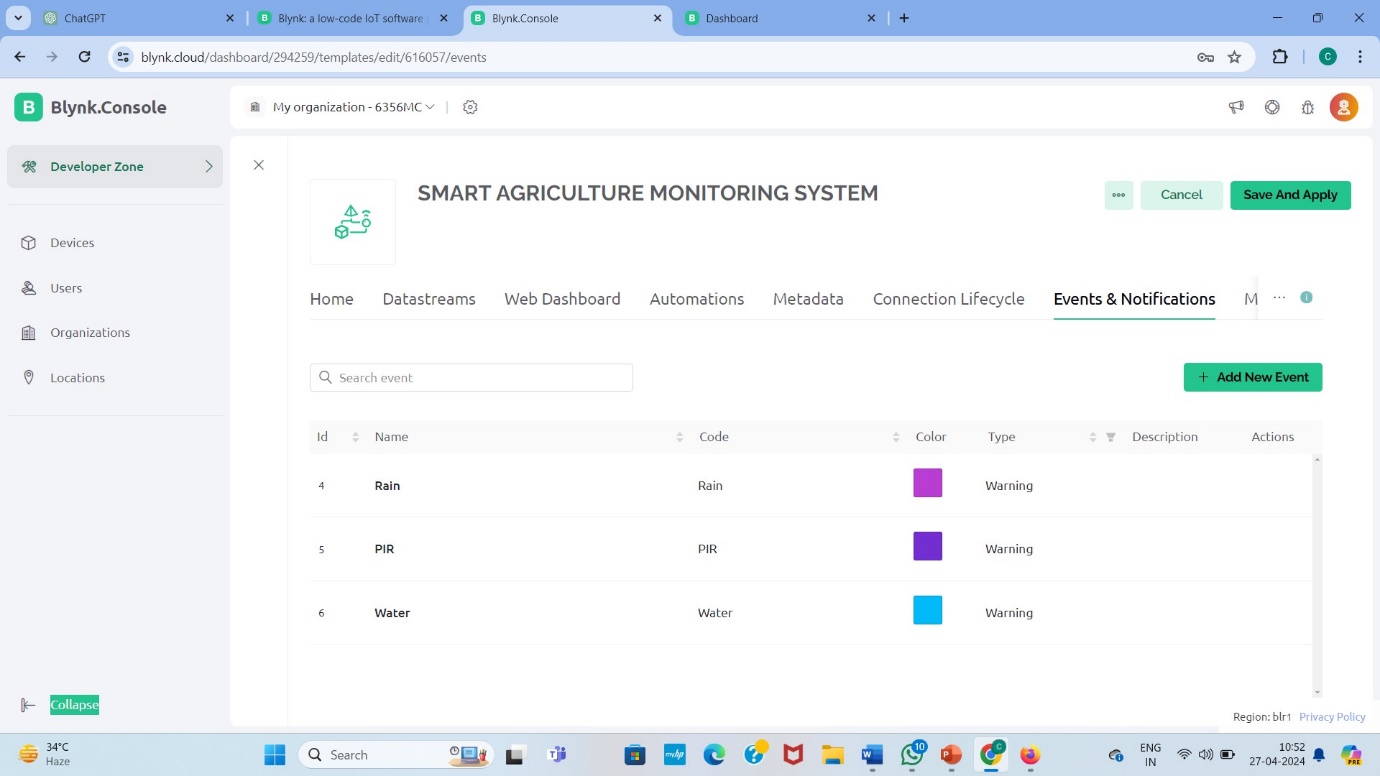
4.2.7

CONNECTION LIFE CYCLE



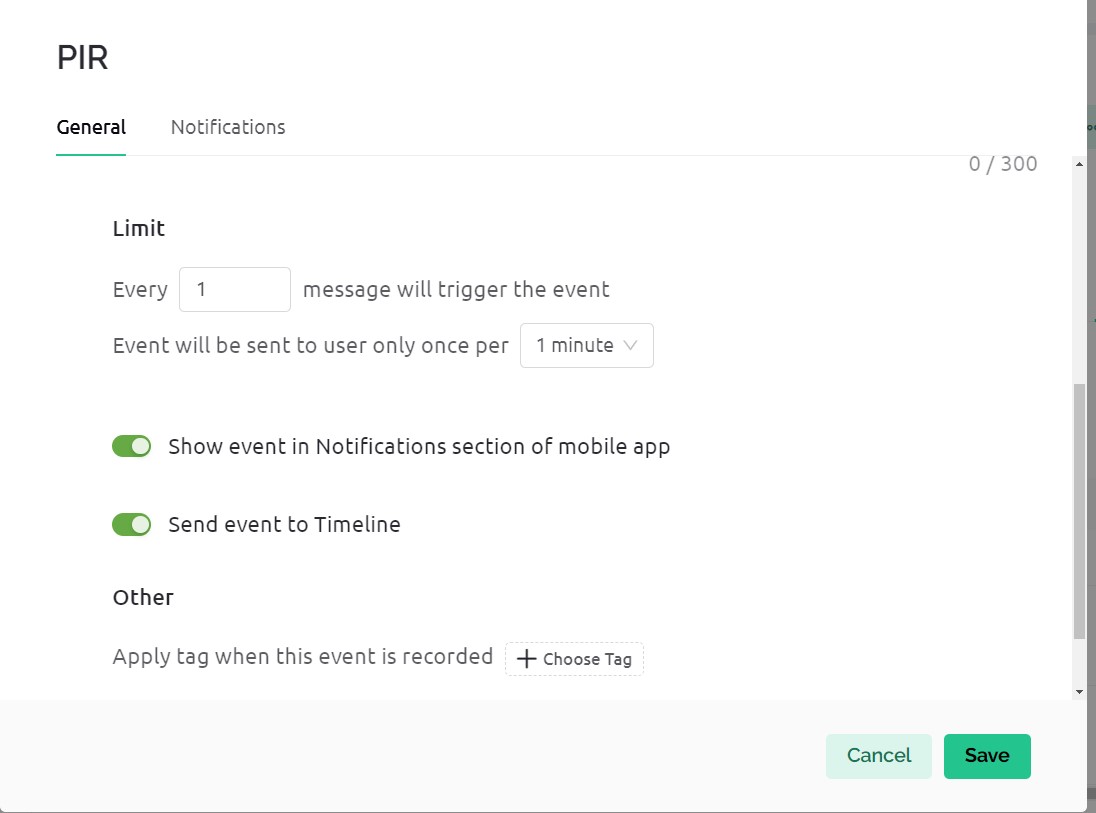
4.2.8

EVENTS AND NOTIFICATIONS

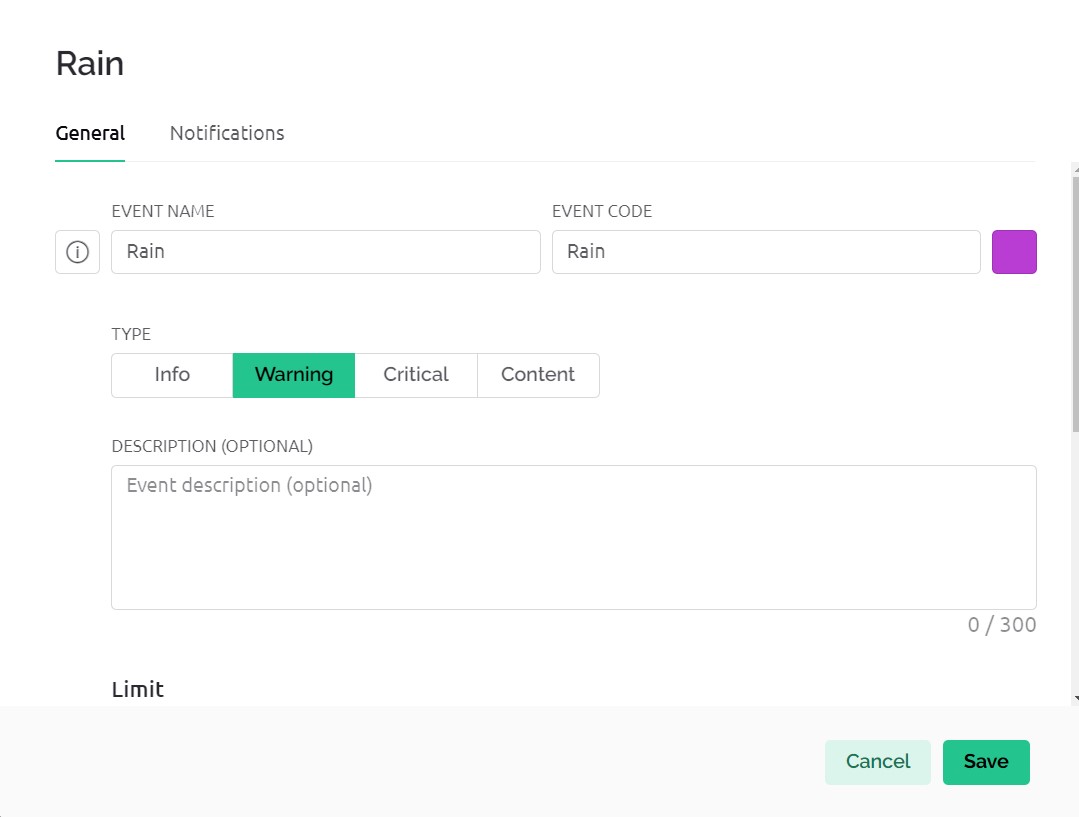


4.2.9

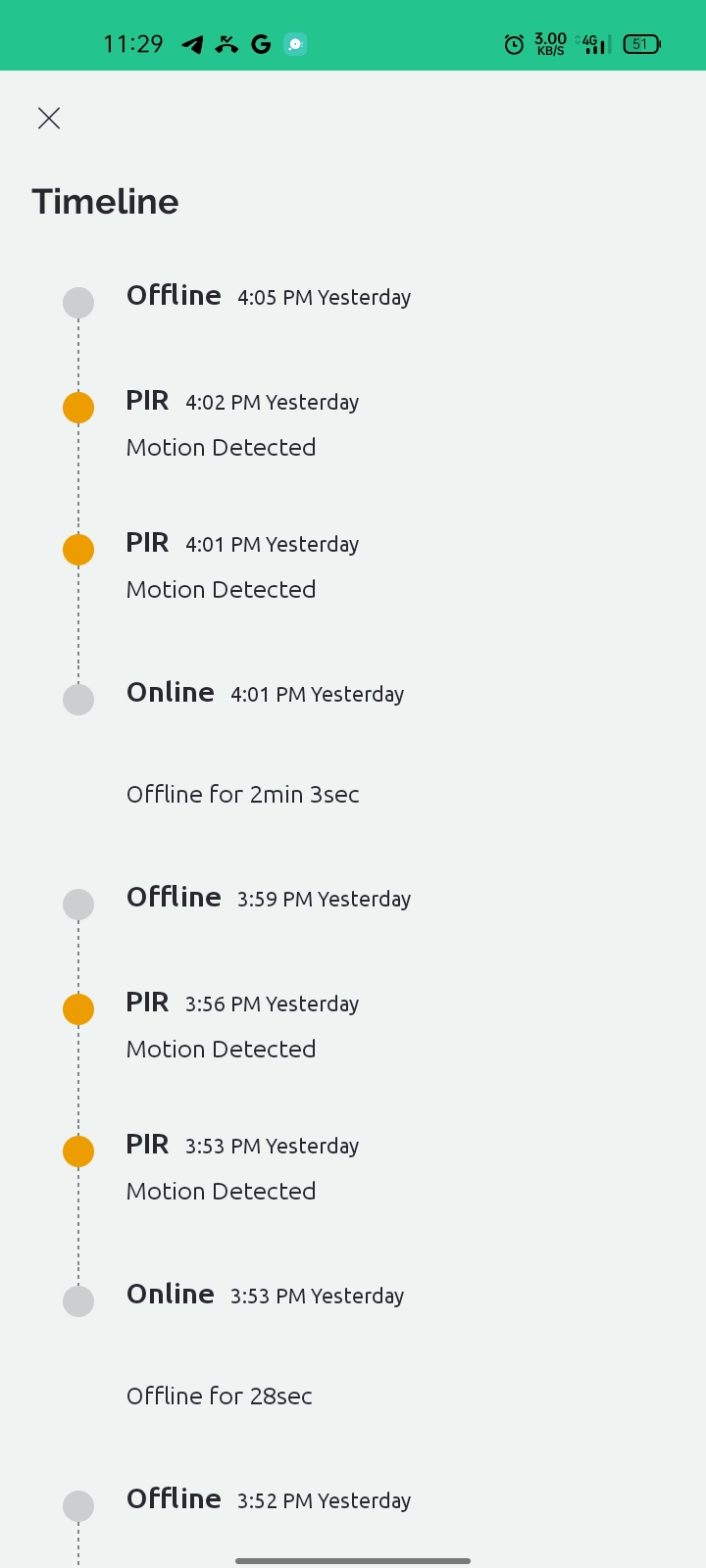
NOTIFICATIONS LIMITS





s

NOTIFICATIONS



4.3 SOURCE CODE

// Project Name : Smart Agriculture Monitoring System Using UV Light

// Platform : Blynk Mobile App

#define BLYNK\_TEMPLATE\_ID "TMPL39IOQY6aX"

#define BLYNK\_TEMPLATE\_NAME "SMART AGRICULTURE MONITORING SYSTEM"

#define BLYNK\_AUTH\_TOKEN "u7SY-5U1Tch5hLXknswMS23alzun6cva"

#define BLYNK\_PRINT Serial

#define BLYNK\_PRINT Serial

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

#include "DHT.h"

char auth[] = BLYNK\_AUTH\_TOKEN;

// Set password to "" for open networks.

char ssid[] = "GS"; // type your wifi name

char pass[] = "12341234"; // type your wifi password

#define DHTPIN D4

#define rainSensor D1

#define moistureSensor D3

#define analogreading A0

#define PIR D2

#define motor D0

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

BlynkTimer timer;

int moistureState =0;

int lastmoisState=0;

int rainState = 0;

int lastRainState = 0;

int MotorState=0;

int lastPIRState = 0;

int PIRState=0;

unsigned long old =0;

unsigned long current =0;

int interval=0;

BLYNK\_CONNECTED() {

Blynk.syncVirtual(V0);

}

BLYNK\_WRITE(V0)

{

MotorState = param.asInt();

if (MotorState==1)

{

digitalWrite(motor,HIGH);

Serial.println("Motor ON");

}

else

{

digitalWrite(motor,LOW);

Serial.println("Motor OFF");

}

}

void Reading()

{

float h = dht.readHumidity();

float t = dht.readTemperature();

moistureState = digitalRead(moistureSensor);

PIRState = digitalRead(PIR);

delay(10);

Serial.println("PIRSTATTE: "+String(PIRState));

rainState = digitalRead(rainSensor);

if (isnan(h) || isnan(t)) {

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

return;

}

Blynk.virtualWrite(V2, h); //V2 is for Humidity

Blynk.virtualWrite(V1, t); //V1 is for Temperature

int i = analogRead(analogreading);

i=4095-i;

i= map(i, 0, 4095, 0, 100);

// Serial.println ("AnalogReading: " + String (i) );

Blynk.virtualWrite(V3, i);

if (moistureState == 1 && lastmoisState == 0) {

Serial.println("Blynk Notification, needs water, send notification");

Blynk.logEvent("WATER","Water your plants");

lastmoisState = 1;

delay(1000);

}

else if (moistureState == 0 && lastmoisState == 1) {

Serial.println("Water is sufficient");

lastmoisState=0;

}

else if (rainState == 1 && lastRainState == 0) {

Serial.println("Blynk notification: Its Raining!");

Blynk.logEvent("RAIN","Raining in Village");

lastRainState = 1;

delay(1000);

}

else if (rainState == 1 && lastRainState == 1) {

Serial.println("has not been watered yet");delay(1000);

Blynk.virtualWrite(V0, LOW);

}

else if (PIRState == 1 && lastPIRState == 0) {

Serial.println("Blynk Notification,PIR Object detected");

Blynk.logEvent("PIR","Motion Detected");

lastPIRState = 1;

delay(1000);

}

if (rainState == 0) {

Serial.println("Its Raining Continuously!");

lastRainState = 0;

}

else if (moistureState == 1 && lastmoisState == 1) {

Serial.println("NO RAIN");

}

current=millis();

interval=(current-old)/1000;

if (interval > 60 && PIRState==0)

{

interval=0;

old=current;

lastPIRState = 0;

Serial.println("PIR Reactivating...");

delay(3000);

}

}

void setup()

{

Serial.begin(115200);

Serial.println();

delay(2000);

Serial.println("Pleae wait.... DHT Sensor activation");

dht.begin();

delay(3000);

Serial.println("Please wait for Blynk Server connection");

pinMode(moistureSensor, INPUT);

pinMode(rainSensor, INPUT);

pinMode(PIR, INPUT);

pinMode(analogreading,INPUT);

pinMode(motor,OUTPUT);

Blynk.begin(auth, ssid, pass);

Blynk.virtualWrite(V0, MotorState);

timer.setInterval(1000L, Reading);

}

void loop()

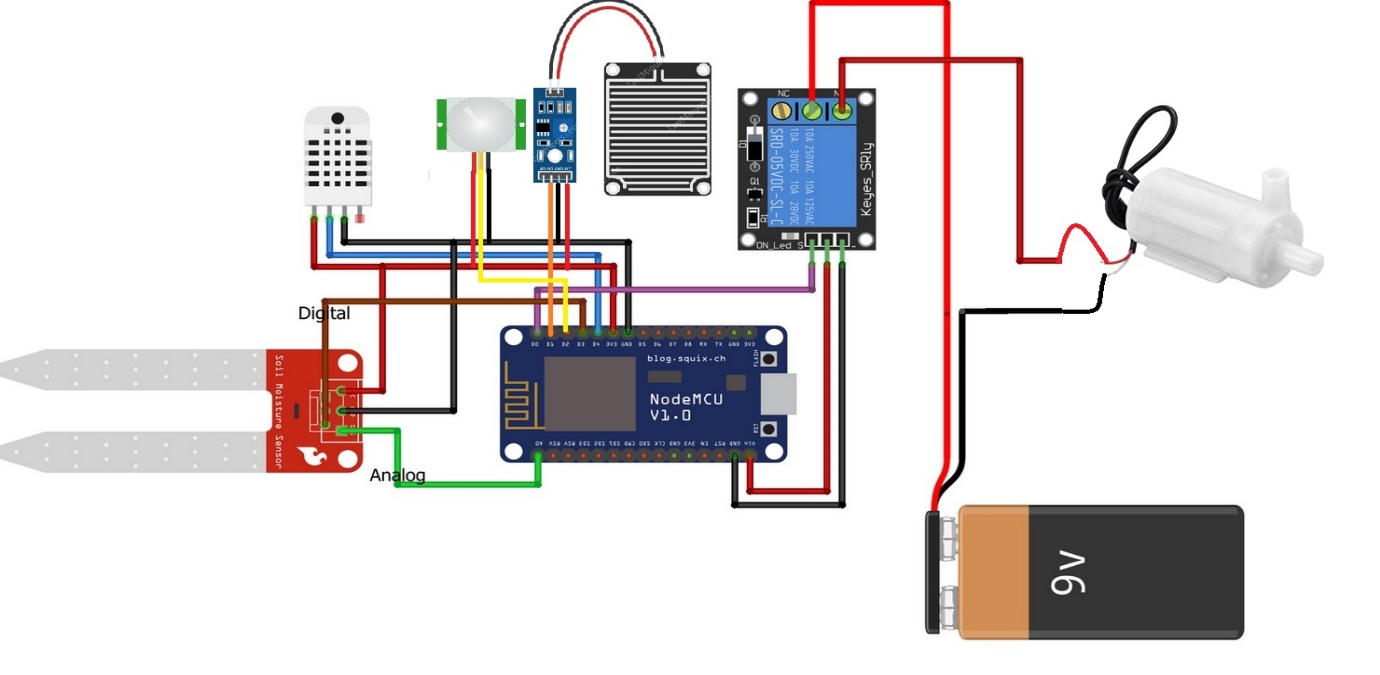
{

Blynk.run();

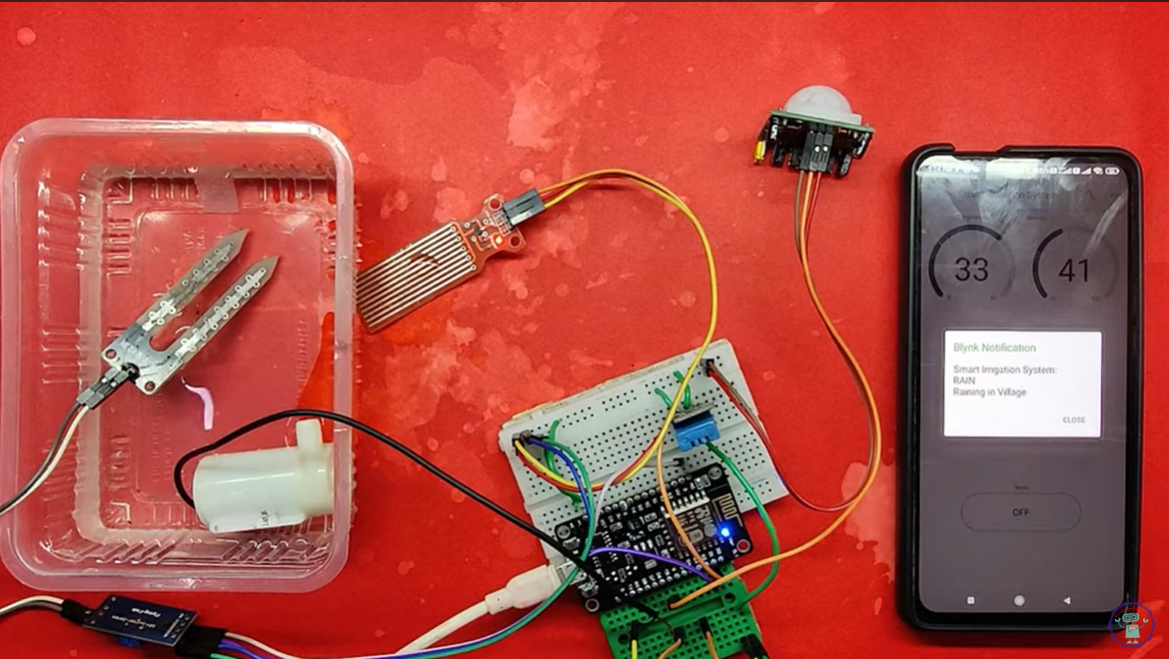
 timer.run(); }

**5.CIRCUIT DIAGRAM AND LIVE PICTURE**

**5.1 CIRCUIT DIAGRAM**



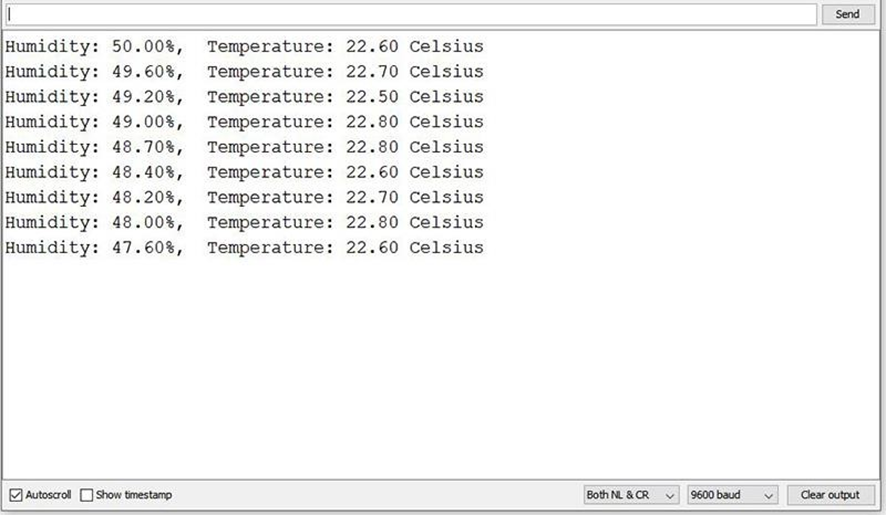
**5.2 LIVE PICTURE**



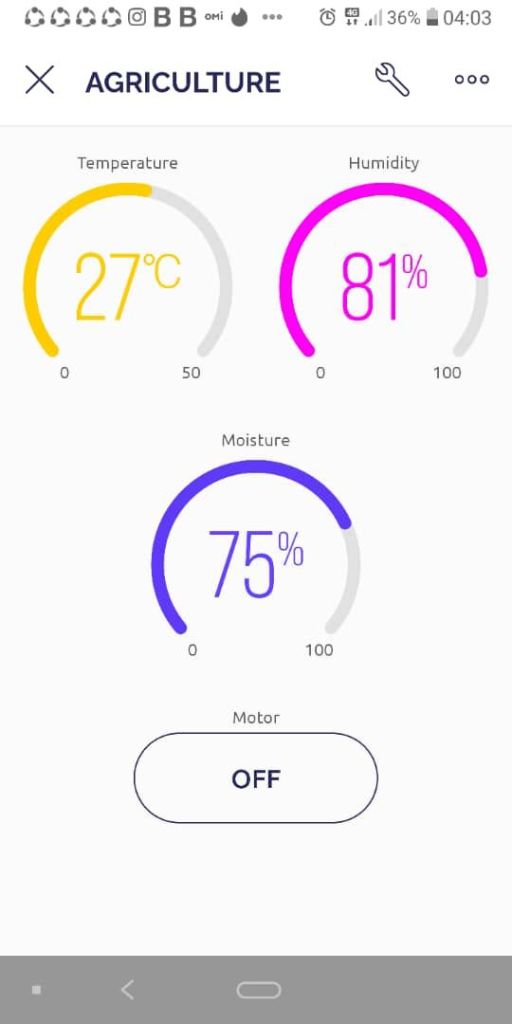
**6.RESULTS**

**6.1**





**6.2.IN BLINK APP RESULT**



**7.FUTURE SCOPE**

In other words, for successful IoT software development, it's essential to prioritize testing to ensure a high-quality end product and mitigate potential faults arising from complexity or hardware-software interactions. Continuous monitoring throughout development is crucial to catch and address issues promptly, minimizing errors and failures. Management should take accountability and utilize analytics to assess risks and maintain quality standards, especially regarding security. Lastly, choosing a competent IoT solution provider is paramount for achieving desired outcomes and ensuring the success of IoT applications.

For the purpose of avoiding errors and failures, continuous monitoring is essential during the development of Internet of Things software.

Management must accept accountability. Analytics must guarantee quality and evaluate software risks because any carelessness in the creation of IoT applications can result in security issues that can be very expensive to fix

**8.CONCLUSION**

In conclusion, the development of a smart agriculture system signifies a transformative shift in the agricultural sector, leveraging advanced technologies to enhance productivity, efficiency, and sustainability. By integrating Internet of Things (IoT) devices, sensors, and automated systems, farmers can monitor and manage their crops and livestock with unprecedented precision. This leads to optimized resource usage, reduced environmental impact, and improved yield quality. The real-time data analytics and predictive insights provided by smart agriculture systems empower farmers to make informed decisions, adapt to changing conditions, and ultimately increase their profitability. As technology continues to evolve, the potential for smart agriculture to revolutionize traditional farming practices and contribute to global food security becomes increasingly evident. Embracing these innovations will be crucial for meeting the demands of a growing population and ensuring a sustainable future for agriculture.