## IOT BASED SMART AGRICULTURE MONITORING SYSTEM

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **TABLE OF CONTENTS** | **PageNo** |
| Abstract | 1 |
| Introduction | 2 |
| Components required | 3 |
| Procedure | 4 |
| Circuit diagram | 5 |
| Code | 6 |
| Output | 7 |
| Result | 8 |
| Conclusion | 9 |

## ABSTRACT

Agriculture area unit controlled space surroundings to grow plants. so as to realize most plant growth, the continual watching and controlling of environmental parameters like temperature, humidity, soil moisture, intensity level, soil pH, etc. area unit necessary for a greenhouse system. The most aim of this project is to style an easy, low cost, Arduino based mostly system to observe the values of environmental parameters which area unit endlessly updated and controlled so as to realize optimum plant growth and yield. The DHT11 sensing element, Soil wet sensing element, LDR sensing element, and pH scale sensing element is that the main sensors utilized in this project offer the precise price of temperature, humidity, water content, intensity level, and soil pH scale severally. All environmental parameters area unit sent to android mobile via offline and on-line. GSM (Global System for Mobile communication) electronic equipment is used to send SMS (Short Message Service) that displays the current standing of the environmental parameters. The SMS is distributed to the user once the sensing element value exceeds an outlined level. All farmers will control their agriculture from anywhere by knowing the standing of their greenhouse parameters at any time and that they will management actuators (cooling fan, fan, and motor pump) to regulate environmental parameters by causation SMS. WI-FI is additionally wont to send the information parameters to a mobile that eliminates the SMS charges. All environmental parameters area units sent to the server through WI-FI and hold on within the info. Therefore the user will monitor and management parameters through the android mobile application.

**What is Smart Agriculture Monitoring System?**

Smart agriculture monitoring system or simply smart farming is an emerging technology concept where data from several agricultural fields ranging from small to large scale and its surrounding are collected using smart electronic sensors. The collected data are analyzed by experts and local farmers to draw short term and long term conclusion on weather pattern, soil fertility, current quality of crops, amount of water that will be required for next week to a month etc.

We can take smart farming a step further by automating several parts of farming, for example smart irrigation and water management. We can apply predictive algorithms on microcontrollers or SoC to calculate the amount of water that will be required today for a particular agriculture field.

Say, if there was rain yesterday and the quantity of water required today is going to be less. Similarly if humidity was high the evaporation of water at upper ground level is going to be less, so water required will be less than normal, thus reducing water usage.

# INTRODUCTION

In today’s agriculture, observation and controlling of the many parameters square measure necessary for the great quality and productivity of plants. However to induce the specified result some parameters like temperature, humidity, soil wet, intensity level, and soil PH square measure necessary for higher plant growth. An IoT-based smart agriculture monitoring system employs a network of interconnected devices and sensors to collect and analyze real-time data from the farm environment. These devices can include soil moisture sensors, weather stations, crop health monitors, livestock trackers, and automated irrigation systems, among others. The collected data is transmitted to a central server or cloud-based platform, where it is processed and transformed into actionable insights for farmers and agricultural stakeholders. Therefore associate degree Arduino based mostly on greenhouse environment observation and controlling system mistreatment sensors are designed. The key objective of an IoT-based smart agriculture monitoring system is to enable farmers to make data-driven decisions and manage their farms more efficiently. By continuously monitoring various environmental parameters, such as soil moisture levels, temperature, humidity, and light intensity, farmers can gain valuable insights into the condition of their crops and make informed decisions about irrigation, fertilization, and pest control. Furthermore, by utilizing livestock trackers and health monitors, farmers can closely monitor the well-being of their animals and promptly detect any signs of illness or distress. For this project, the Arduino microcontroller is employed. Arduino will receive input from a range of sensors and it wills management motors, lights, and different actuators. Four sensors, DHT11 sensor, LDR sensor, Soil moisture sensor, and pH sensor square measure used. The DHT11 sensor is employed to live temperature and wetness. Soil wet device measures the water content within the soil. PH sensor measures the PH of the soil. LDR sensor is employed to live intensity levels. A cooling fan, exhaust fan, and motor pump are connected to the Arduino. All environmental parameters square measure sent to automaton transportable via offline and on-line. A GSM modem and WI-FI square measure won’t send environmental parameters to android mobile.

This project consists of Arduino as brain and we are utilizing 5 sensors which measures six different environmental factors that crop’s growth and nourishment depend on:

### Temperature and humidity sensor.

1. **Air Quality / gas sensor.**

### Light sensor.

1. **Soil moisture sensor.**
2. **Barometric pressure sensor.**

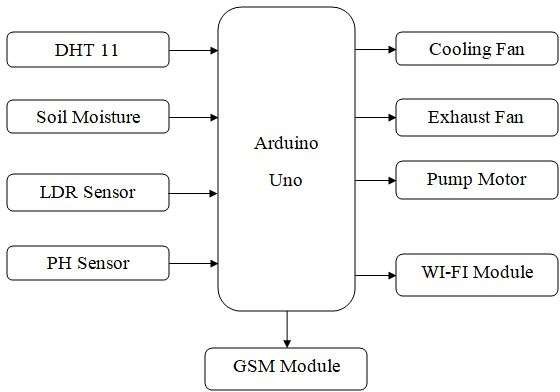
**Hardware Requirements**

* Arduino Uno
* DHT11
* LDR Sensor
* PH Sensor
* Soil Moisture Sensor
* Cooling Fan
* Exhaust Fan
* Pump Motor
* Wi-Fi Module
* GSM Module

## Software Requirements

* Arduino IDE
* Embedded-C

**Block Diagram**



## PROCEDURE

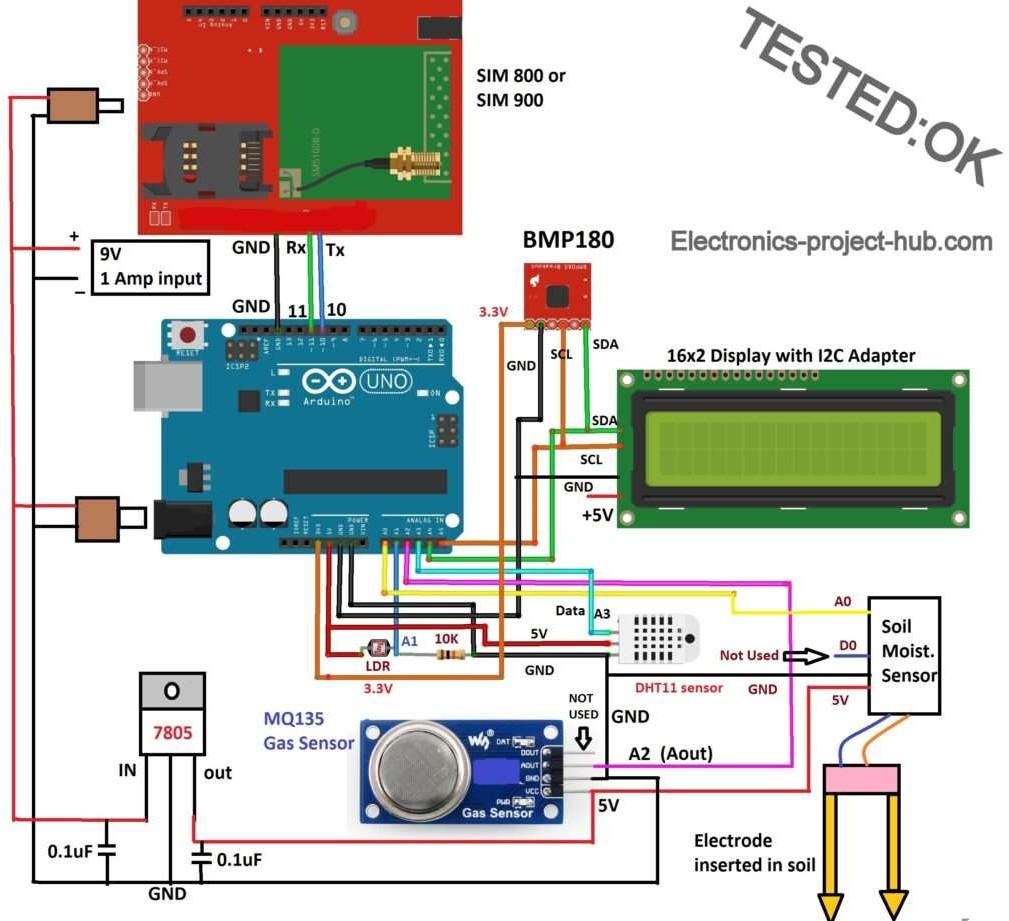
To implement an IOT-based agriculture monitoring and controlling system, you can follow this procedure:

### Define System Requirements:

Identify the specific objectives and requirements of your agriculture monitoring and controlling system, considering factors such as crop type, environmental conditions, and desired functionality.

### Select Hardware Components:

Choose an appropriate Arduino board based on the system requirements. Arduino Uno or Arduino Mega are commonly used options for IOT-based agriculture systems. Select sensors relevant to your agricultural monitoring needs, such as soil moisture sensors, temperature and humidity sensors, light sensors, or weather sensors. Determine the actuators required for controlling irrigation, ventilation, or other processes in your agricultural setup. Consider the need for communication modules like Wi-Fi, GSM, or LoRa to establish connectivity.



### BuildHardwareSetup:

Connect the sensors and actuators to the Arduino board following the manufacturer's instructions and pin mappings. Use breadboards, jumper wires, and other prototyping components to build and test the circuit connections.

### Install Arduino IDE:

Download and install the Arduino Integrated Development Environment (IDE) on your computer.

Installing necessary library files to Arduino IDE:

1. LiquidCrystal\_I2C.h
2. Adafruit\_BMP085.h
3. DHT11 sensor

## Write Arduino Code:

//-----Electronics-project-hub>com //

#include <SoftwareSerial.h> #include <Wire.h>

#include <LiquidCrystal\_I2C.h> #include <dht.h>

#include <Adafruit\_BMP085.h> Adafruit\_BMP085 bmp;

dht DHT;

LiquidCrystal\_I2C lcd(0x27, 16, 2); SoftwareSerial gsm(10, 11); // RX, TX #define DHT11\_PIN A3

int chk;

int humi = 0; int temp = 0; int soil = 0; int light = 0; int BMP = 0; int gas = 0; boolean HT; void setup(){

gsm.begin(9600); pinMode(A0, INPUT); pinMode(A1, INPUT); pinMode(A2, INPUT); pinMode(A3, INPUT); lcd.init(); lcd.backlight(); lcd.setCursor(0, 0);

lcd.print("Please wait for"); lcd.setCursor(0, 1); lcd.print("60 seconds."); delay(20000); delay(20000); delay(20000); modem\_init();

data\_init();

internet\_init(); lcd.clear();} void loop(){

chk = DHT.read11(DHT11\_PIN); temp = DHT.temperature;

humi = DHT.humidity; soil = analogRead(A0); light = analogRead(A1); gas = analogRead(A2);

BMP = bmp.readPressure(); lcd.clear();

lcd.setCursor(0, 0); lcd.print("Soil:");

soil = map(soil, 0, 1023, 100, 0); lcd.print(soil);

lcd.print("%"); lcd.setCursor(0, 1); lcd.print("Light:");

light = map(light, 0, 1023, 0, 100); lcd.print(light);

lcd.print("%"); delay(3000); lcd.clear(); lcd.setCursor(0, 0);

switch (chk) {

case DHTLIB\_OK: HT = true;

break; default:

HT = false; break;

}

if (HT == true) { lcd.print("Temp:"); lcd.print(temp); lcd.print(" \*C"); lcd.setCursor(0, 1); lcd.print("Humidity:"); lcd.print(humi); lcd.print("%");

}

else { temp = 0;

humi = 0; lcd.print("Temp:"); lcd.print("No Data"); lcd.setCursor(0, 1); lcd.print("Humidity:");

lcd.print("No Data");

}

delay(3000); lcd.clear(); lcd.setCursor(0, 0); lcd.print("Air Qlt: ");

gas = map(gas, 0, 1023, 0, 100); lcd.print(gas);

lcd.print("%"); lcd.setCursor(0, 1); lcd.print("Pressure:");

if (!bmp.begin()) { lcd.print("No Data"); BMP = 0;

}

else { lcd.print(BMP);

}

lcd.print("Pa"); delay(3000); Send\_data();

}

void modem\_init(){ Serial.println("Please wait ");

gsm.println("AT"); delay(1000); gsm.println("AT+CMGF=1"); delay(1000);

gsm.println("AT+CNMI=2,2,0,0,0"); delay(1000);

}

void data\_init(){ Serial.println("Please wait ");

gsm.println("AT"); delay(1000); delay(1000);

gsm.println("AT+CPIN?"); delay(1000);

delay(1000); gsm.print("AT+SAPBR=3,1"); gsm.write(',');

gsm.write('"'); gsm.print("contype"); gsm.write('"');

gsm.write(',');

gsm.write('"');

gsm.print("GPRS");

gsm.write('"');

gsm.write(0x0d); gsm.write(0x0a); delay(1000); ;

gsm.print("AT+SAPBR=3,1"); gsm.write(',');

gsm.write('"');

gsm.print("APN");

gsm.write('"');

gsm.write(',');

gsm.write('"');

// APN // gsm.print("bsnlnet"); //APN Here gsm.write('"');

gsm.write(0x0d); gsm.write(0x0a); delay(1000);

gsm.print("AT+SAPBR=3,1"); gsm.write(',');

gsm.write('"');

gsm.print("USER");

gsm.write('"');

gsm.write(',');

gsm.write('"');

gsm.print(" ");

gsm.write('"'); gsm.write(0x0d); gsm.write(0x0a); delay(1000);

gsm.print("AT+SAPBR=3,1"); gsm.write(',');

gsm.write('"');

gsm.print("PWD");

gsm.write('"');

gsm.write(',');

gsm.write('"');

gsm.print(" ");

gsm.write('"'); gsm.write(0x0d); gsm.write(0x0a); delay(2000);

gsm.print("AT+SAPBR=1,1"); gsm.write(0x0d); gsm.write(0x0a);

delay(3000);

}

void internet\_init(){ Serial.println("Please wait ");

delay(1000);

gsm.println("AT+HTTPINIT"); delay(1000); delay(1000); gsm.print("AT+HTTPPARA="); gsm.print('"');

gsm.print("CID");

gsm.print('"');

gsm.print(',');

gsm.println('1'); delay(1000);

}

void Send\_data(){ lcd.clear();

lcd.print("Sending the data"); lcd.setCursor(0, 1); lcd.print("to Thingspeak..."); delay(1500);

gsm.print("AT+HTTPPARA="); gsm.print('"');

gsm.print("URL");

gsm.print('"');

gsm.print(',');

gsm.print('"');

gsm.print("http:");

gsm.print('/');

gsm.print('/');

// Your API Key Here //

//Replace xxxxxxxxxxx with your write API key. gsm.print("api.thingspeak.com/update?api\_key=xxxxxxxxxxxxx&field1=");

// // gsm.print(soil);// >>>>>> variable 1 gsm.print("&field2=");

gsm.print(light); // >>>>>> variable 2 gsm.print("&field3="); gsm.print(gas); // >>>>>> variable 3 gsm.print("&field4="); gsm.print(temp); // >>>>>> variable 4 gsm.print("&field5="); gsm.print(humi); // >>>>>> variable 5 gsm.print("&field6="); gsm.print(BMP); // >>>>>> variable 6 gsm.write(0x0d);

gsm.write(0x0a); delay(1000);

gsm.println("AT+HTTPACTION=0"); delay(1000);

}

### Changes you need to make in the code:

1. You need to insert your write API key in the given code. gsm.print("api.thingspeak.com/update?api\_key=xxxxxxxxxxxxx&field1=");
2. You need to insert correct APN in the code for your SIM card (network provider). gsm.print("bsnlnet"); //APN Here

### Power Supply and Enclosure:

Ensure a reliable power supply for the Arduino board and connected components. Consider using batteries, solar panels, or electrical connections based on the system requirements. Consider an enclosure to protect the Arduino board and associated components from environmental factors.

### Test and Deployment:

Test the complete system to ensure proper functionality, including sensor data collection, actuator control, and cloud integration. Deploy the IOT-based agriculture monitoring and controlling system in the desired location, such as a field or greenhouse. Monitor the system's performance and make necessary adjustments based on the collected data and user feedback. Remember to consider security aspects, such as data encryption, access control, and secure communication protocols, to protect the system from unauthorized access and data breaches. Please note that this procedure provides a general overview, and the specific implementation details may vary depending on the requirements and components chosen for your IoT-based agriculture monitoring and controlling system.

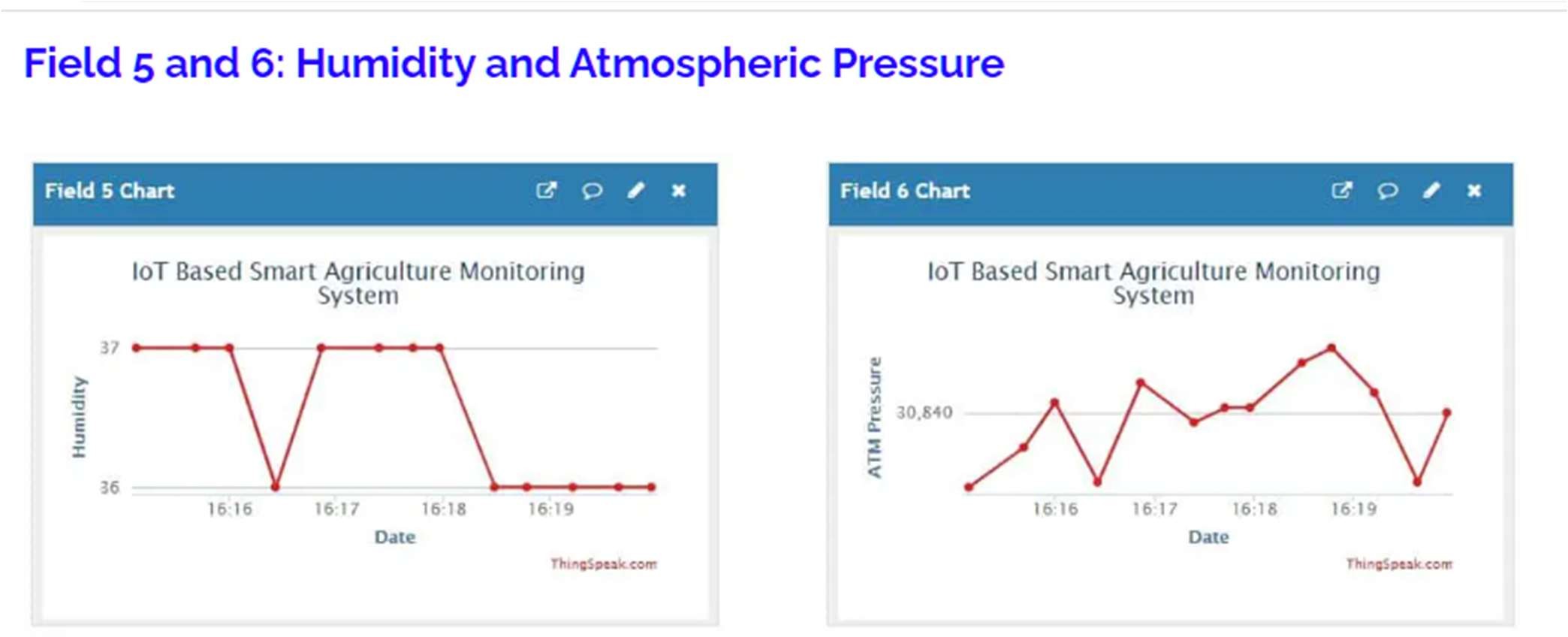
### How to operate this smart agriculture monitoring system properly:

1. With completed hardware setup insert a valid SIM card that has a working mobile data plan.
2. Upload the program code using USB cable with correct APN and write API key (without powering the circuit from wall adapter).
3. Once the program is uploaded remove the USB, now turn on the circuit using 9V to 12V / 1 amp (or more) wall adapter.
4. The circuit will boot and it tells you to wait for 1 minute. Meanwhile the GSM module is getting ready and latching to the mobile network. The MQ 135 is also getting ready for operation by heating up the metal body.
5. After one minute the LCD will display all the sensor data (two at a time).
6. Now you can see network LED of GSM module is blinking fast which means your project connected to internet and started to send data to Thing speak.
7. Open your Thing speak account and click on private view tab; you will see this (output of this project):

## OUTPUT:

Field 1 and 2: Soil Moisture and Light Sensor





### RESULT:

As technology continues to advance, we can expect further advancements and refinements in IoT-based smart agriculture, ultimately contributing to a more efficient and resilient agricultural sector. IOT based agriculture monitoring system starts it checks the water level, humidity and moisture level. Sensors sense the level of water if it goes down, it automatically starts the water pump. If the temperature goes above the level, fan starts.The set up uses the temperature sensor, moisture sensor and humidity sensor which measure the approximate temperature, moisture and humidity in the soil. This value enables the system to use appropriate quantity of water which avoids over/under irrigation.

## CONCLUSION:

In conclusion, IoT-based smart agriculture monitoring systems have the potential to transform traditional farming practices by leveraging the power of data and connectivity. By enabling real-time monitoring, precise resource management, early detection of issues, and data-driven decision-making, these systems can significantly improve agricultural productivity, sustainability, and profitability. As technology continues to advance, we can expect further advancements and refinements in IoT-based smart agriculture, ultimately contributing to a more efficient and resilient agricultural sector.s An Arduino based mostly agriculture observance and controlling system is designed. DHT11 sensor, Soil moisture sensor, LDR sensor, and pH sensor is that the main sensors utilized in this project that provide the exact value of temperature, humidity, wetness content, intensity level, and soil pH severally. This technique is intended for controlling and observance environmental parameters in a very greenhouse by an easy SMS from anywhere via the GSM network. WI-FI is additionally wont to send the information parameters to the mobile that eliminates the SMS charges. This technique reduces power consumption, maintenance, and complexness. This project is often employed in the agricultural field, in a very nursery, and within the installation.