

**Project Development Phase
Delivery of Sprint 1**

Date	14 November 2022
Team ID	PNT2022TMID14459
Project Name	SmartFarmer – IoT Enabled Smart Farming Application

**IOT BASED SMART FARMING
SOIL SENSOR WITHOUT WIFI**

/*

Plant Watering Sytem

The circuit:

- Water pump

Power supply: 4.5~12V DC

Interface: Brown +; Blue - Temperature/moisture sensor

Power supply: 3.3-5v

- Moisture sensor

Power supply: 3.3-5v */

```
#include "DHT.h"

#define DHTPIN 2 // what digital pin we're connected to
#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321 DHT dht(DHTPIN, DHTTYPE);

const int SOIL_MOISTURE_SENSOR_PIN = A0; const int WATER_PUMP_PIN = 4;

const int dry = 520; const int wet = 270; const int moistureLevels = (dry - wet) / 3; // TODO: Should we
have a counter so if it waters for X times, then take a break?

// OPTIMIZE: how dry to start watering and for how long.
const int soilMoistureSartWatering = 400; const
int soilMoistureStopWatering = 300;

// 60 seconds
const long waterDuration = 1000L * 60L;

// 60 seconds
const long sensorReadIntervals = 1000L * 60L;

// 2 hr
const long waterIntervals = 1000L * 60L * 60L * 2;
```

```
long lastWaterTime = -waterIntervals - 1; boolean isWatering = false;
```

```
void setup() {
```

```
    Serial.begin(9600);
```

```
    pinMode(WATER_PUMP_PIN, OUTPUT); waterPumpOff(); dht.begin();
```

```
}
```

```
void loop()
```

```
    { mainLoop
```

```
      (); }
```

```
void mainLoop() {
```

```
    float temperature = getTemperature(); float
```

```
    humidity = getHumidity(); long soilMoisture = analogRead(SOIL_MOISTURE_SENSOR_PIN);
```

```
    Serial.println("Soil Moisture: " + readableSoilMoisture(soilMoisture) + ", " + soilMoisture);
```

```
    Serial.println("Temperature: " + String(temperature) + " *F");Serial.println("Humidity: " + String(humidity) + " %");
```

```
    if (millis() - lastWaterTime > waterIntervals)
```

```
{  
  waterPlants(soilMoisture); lastWaterTime = millis();  
}
```

```
  delay(sensorReadIntervals);
```

```
}
```

```
void waterPlants(int soilMoisture) {  
  // Should this take a moving avg of the soilMoisture?  
  // Can get outliers on the right after watering. if  
    (soilMoisture > soilMoistureSartWatering)  
    { isWatering = true;  
    } else if (soilMoisture < soilMoistureStopWatering)  
    { isWatering = false;  
    }  
  Serial.println(isWatering ? "Starting to water" : "Skipping water");
```

```
    if (isWatering) { waterPumpOn();  
        delay(waterDuration);  
        waterPumpOff();  
        Serial.println("Done watering");  
    }  
}
```

```
String readableSoilMoisture(int soilMoisture){  
    if (soilMoisture <= wet) {  
        return "Water"; }  
    else if (soilMoisture > wet && soilMoisture < (wet + moistureLevels)) {  
        return "Very Wet"; }  
    else if (soilMoisture > (wet + moistureLevels) && soilMoisture < (dry moistureLevels)) {  
        return "Wet"; }  
    else if (soilMoisture < dry && soilMoisture > (dry - moistureLevels)) {return "Dry"; }  
    else{  
        return "Air"; }  
    }  
float getTemperature() {
```

```

// Read temperature as Fahrenheit
(isFahrenheit = true)
float temperature dht.readTemperature(true);
if (isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
}    return temperature;
}

float getHumidity()
{ float humidity = dht.readHumidity();
if
    (isnan(humidity)) {
        Serial.println("Failed to read from DHT sensor!");
    } return humidity; }

void waterPumpOn() {
    Serial.println("Water    pump on");
    digitalWrite(WATER_PUMP_PIN, LOW);
}

```

```
void waterPumpOff() {
    Serial.println("Water pump off");
    digitalWrite(WATER_PUMP_PIN, HIGH); }
```

OUTPUT :

