***Smart Sprinkler System For Golf Course***

***Abstract:***

*As water supply is becoming scarce in today’s world there is an urgency of adopting smart ways of irrigation. The project describes how the golf course can be irrigated smartly using IOT. This project aims at conserving water and saving time to avoiding problems like continuous vigilance of the water sprinklers. It also helps in conserving water by automatically providing water to the field depending on the water requirements, which is essential for playing the sport. The objective of this system is to detect the moisture content of the soil and depending on it sprinkle water .This entire information will be sent to the user’s mobile phone­­­­­­­­­­­­­­­.*

***Problem Statement:***

*In the case of traditional irrigation system of the sprinklers, water saving is not considered. Since, the water is irrigated directly in the land, the field would undergo high stress from variation in soil moisture, and therefore the field of the golf course appears wet. At present there is emerging global water crisis where managing scarcity of water has become a serious job. So this is the serious problem hence, we want to design an Smart Sprinkler for Golf Course which is based on Arduino IDE software using soil moisture sensor that operates by sensing the moisture content of the soil and turns ON/OFF the pump.*

***Working of the Project:***

*It is implemented by using a NodeMCU-ESP2866 which is programmed in C programming language. In working process we use the components namely NodeMCU-ESP2866, mini DC motor and soil moisture sensor. Also an app is used for displaying the moisture content of the soil and water pump status. The soil moisture sensor senses the amount of moisture in the soil .If the moisture measured of soil is less than the particular level i.e. dry, then the signal is sent which turns ON the motor and make motor to pump water to the field. When the soil is wet, the moisture is greater than the reference moisture level, then signal is sent which will turn OFF the motor and stop pumping water to the field. The condition of soil and motor are displayed on the mobile app.*

*Components:*

*Hardware:*

* *NodeMCU*
* *Moisture Sensor*

*Softwares:*

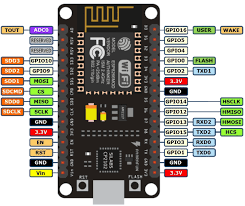
* *Arduino IDE*

*NodeMCU: NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The firmware uses the Lua scripting language.*

*Specification:*

* *Wi-Fi Module – ESP-12E module similar to*[*ESP-12*](https://www.aliexpress.com/item/new-esp8266-ESP-12-wifi-module-ESP8266-serial-WIFI-coexistence-module-AP-STA-AP-STA-WIFI/32239125397.html)*module but with 6 extra GPIOs.*
* *USB – micro USB port for power, programming and debugging*
* *Headers – 2x 2.54mm 15-pin header with access to GPIOs, SPI, UART, ADC, and power pins*
* *Misc – Reset and Flash buttons*
* *Power – 5V via micro USB port*
* *Dimensions – 49 x 24.5 x 13mm*

*Working principle: The ESP8266 can be controlled from your local Wi-Fi network or from the internet (after port forwarding). The ESP-01 module has GPIO pins that can be programmed to turn an LED or a relay ON/OFF through the internet. The module can be programmed using an Arduino/USB-to-TTL converter through the serial pins (RX, TX).*

**

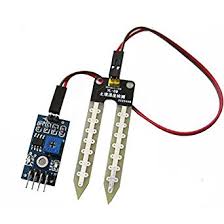
*Soil moisture sensor: The soil moisture sensor measures the volumetric water content in the soil. Soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant soil moisture may vary depending on environmental factors such as*[*soil type*](https://en.wikipedia.org/wiki/Soil_type)*. Golf courses are using soil moisture sensors to increase the efficiency of their irrigation systems to prevent over-watering and leaching of fertilizers and other chemicals into the ground.*

*Specification:*

* *Working Voltage:****5V***
* *Working Current:****<20mA***
* *Interface type:* ***Analog***
* *Working Temperature:****10°C~30°C***

*Working principle:*

*The Soil Moisture Sensor uses capacitance to measure dielectric permittivity of the surrounding medium. In soil, dielectric permittivity is a function of the water content. The sensor creates a voltage proportional to the dielectric permittivity, and therefore the water content of the soil. The sensor averages the water content over the entire length of the sensor. There is a 2 cm zone of influence with respect to the flat surface of the sensor, but it has little or no sensitivity at the extreme edges. The Soil Moisture Sensor is used to measure the loss of moisture over time due to evaporation and plant uptake , evaluate optimum soil moisture contents for various species of plants, monitor soil moisture content to control irrigation in greenhouses and enhance bottle biology experiments.*

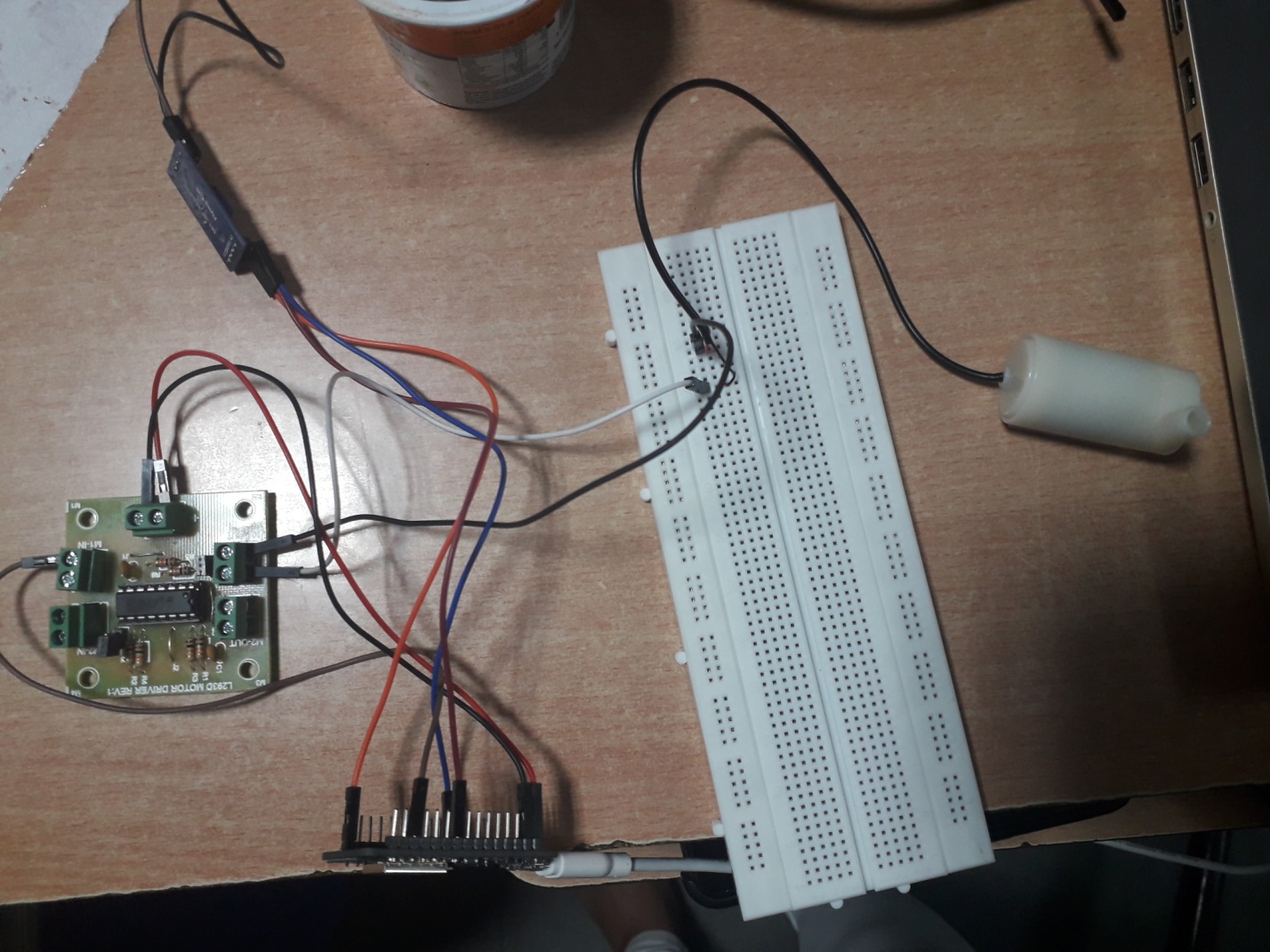
**

*Arduino IDE: The*[*Arduino*](https://en.wikipedia.org/wiki/Arduino) *integrated development environment (IDE) is a cross- platform application. The arduino IDE supports the languages C and C++ using special rules of code structuring.*

***Code****:*

*#include <ESP8266WiFi.h>  
#include <PubSubClient.h>  
const int sensor\_pin=A0;  
void callback(char\* topic, byte\* payload, unsigned int payloadLength);*

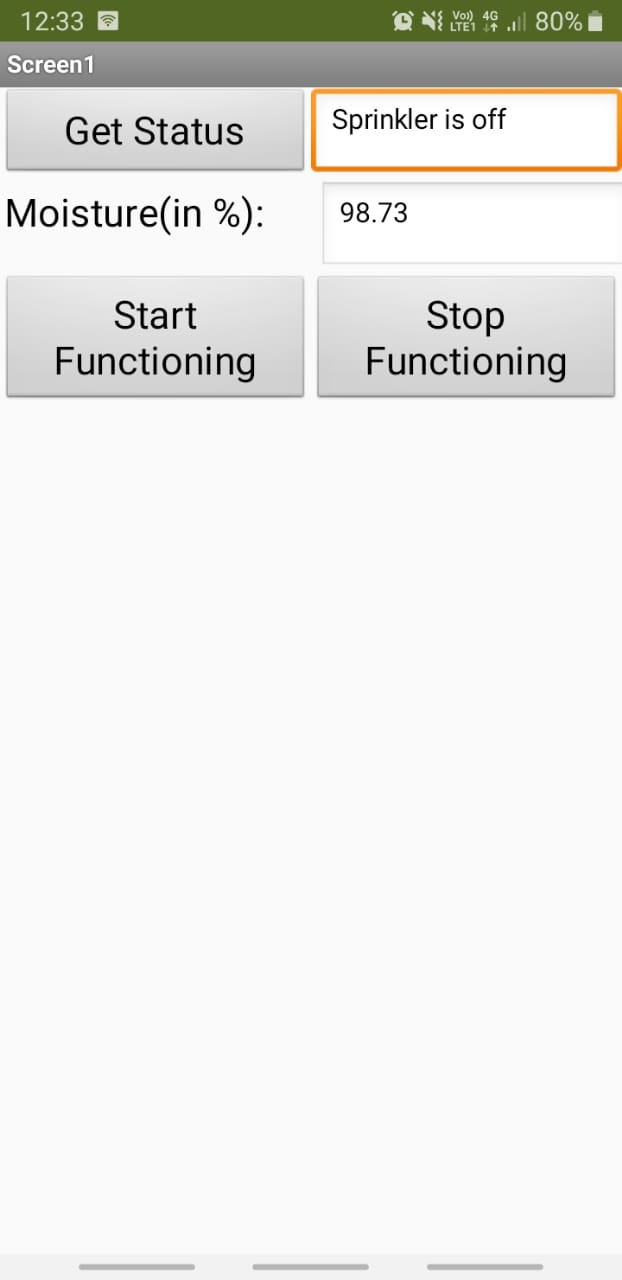
*const char\* ssid = "Shreya";  
const char\* password = "rgnk5422";  
  
#define ORG "0ui3ae"  
#define DEVICE\_TYPE "shreya"  
#define DEVICE\_ID "1234"  
#define TOKEN "1234567890"  
  
 float moist;  
String data3;  
String data="";  
  
const char publishTopic[] = "iot-2/evt/robo/fmt/json";  
char server[] = ORG ".*[*messaging.internetofthings.ibmcloud.com*](http://messaging.internetofthings.ibmcloud.com/)*";  
char topic[] = "iot-2/cmd/home/fmt/String";  
char authMethod[] = "use-token-auth";  
char token[] = TOKEN;  
char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;  
//Serial.println(clientID);  
void callback(char\* topic, byte\* payload, unsigned int payloadLength);  
  
WiFiClient wifiClient;  
PubSubClient client(server, 1883, callback, wifiClient);  
  
int publishInterval = 5000; // 30 seconds  
long lastPublishMillis;  
void publishData();  
  
void setup()  
{  
  Serial.begin(9600);  
   pinMode(D2,OUTPUT);  
   digitalWrite(D2,LOW);  
  wifiConnect();  
  mqttConnect();  
}  
  
void loop()   
{  
   
  moist=(100.00-((analogRead(sensor\_pin)/1023.00)\*100));  
  Serial.print("Soil Moisture(in percentage)=");  
  Serial.print(moist);  
  Serial.println("%");  
  delay(1000);  
  if (millis() - lastPublishMillis > publishInterval)  
  {  
    publishData();  
    lastPublishMillis = millis();  
  }  
 if (!client.loop()) {  
    mqttConnect();  
  }  
}  
  
void wifiConnect() {  
  Serial.print("Connecting to ");   
  Serial.print(ssid);  
  WiFi.begin(ssid, password);  
  while (WiFi.status() != WL\_CONNECTED) {  
  delay(500);  
    Serial.print(".");  
  }  
  Serial.print("nWiFi connected, IP address: "); Serial.println(WiFi.localIP());  
}  
  
void mqttConnect() {  
  if (!client.connected()) {  
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);  
    while (!client.connect(clientId, authMethod, token)) {  
      Serial.print(".");  
      delay(500);  
    }  
    initManagedDevice();  
    Serial.println();  
  }  
}  
void initManagedDevice() {  
  if (client.subscribe(topic)) {  
    Serial.println("subscribe to cmd OK");  
  } else {  
    Serial.println("subscribe to cmd FAILED");  
  }  
}  
void publishData()   
{  
    
  String payload = "{\"d\":{\"moist\":";  
  payload += moist;  
  payload += "}}";  
  
  
  Serial.print("\n");  
  Serial.print("Sending payload: "); Serial.println(payload);  
  
  if (client.publish(publishTopic, (char\*) payload.c\_str())) {  
    Serial.println("Publish OK");  
  } else {  
    Serial.println("Publish FAILED");  
  }  
}  
  
void callback(char\* topic, byte\* payload, unsigned int payloadLength) {  
    
  Serial.print("callback invoked for topic: ");  
  Serial.println(topic);  
  
  for (int i = 0; i < payloadLength; i++) {  
    //Serial.print((char)payload[i]);  
    data3 += (char)payload[i];  
  }  
    
  Serial.print("data: "+ data3);  
  control\_func();  
  data3 = "";  
}  
  
void control\_func()  
{  
  if(data3 == "spon")  
 {  
   if(moist<50)  
   {   
    digitalWrite(D2,HIGH);   
    delay(1000);  
    Serial.println(".......SPRINKLER ON..........");  
  
   }  
 }  
  else if(data3 == "spoff")  
  {  
    digitalWrite(D2,LOW);  
  
    delay(1000);  
    Serial.println(".......SPRINKLER OFF..........");  
  
  }  
  
  else  
  {  
    Serial.println(".....NO COMMAND INPUT..........");  
    }  
  
}*

*The Project: *

*Connections:*

* *+Ve terminal of Power supply to 3V*
* *-Ve terminal of the power supply to ground(GND)*
* *M1-IN to D2*
* *M1-OUT to breadboard*
* *A0 to A0*
* *VCC to 3V*
* *GND to GND*

***App display:***

**

*Documentation by:TYRANTS*

*D.SHREYA*

*E.HARIKA*

*P.SNIGDHA*