IOT ENABLED SMART FARMING APPLICATION

SPRINT DELIVERY-1

TEAM ID: **PTN2022TMID03145**

1.INTRODUCTION:

The main objective of this project is to provide an effective way to automate the farmer's farm land by giving them access to web application so that they can monitor the field without any personal intervention. They can monitor temperature, humidity, soil moisture etc.

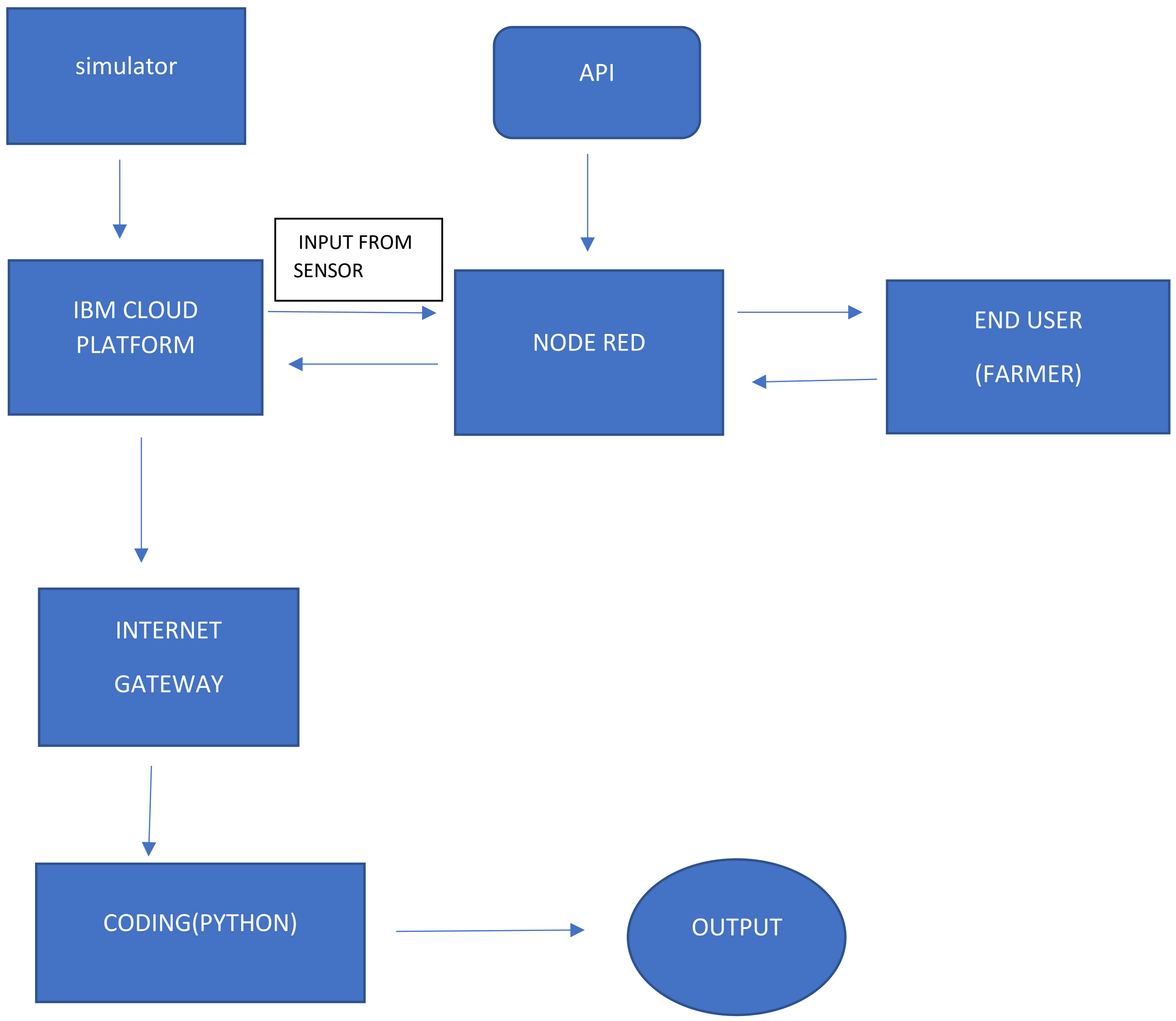
2.PROBLEM STATEMENT:

Farmers are to be present at farm for its maintenance irrespective of the weather conditions. They must ensure that the crops are well watered and the farm status is monitored by them physically. Farmer must stay most of the time in field in order to get a good yield. In difficult times like in the presence of pandemic also they must work hard in their fields risking their lives to provide food for the country.

3.PROPOSED SOLUTION:

In order to improve the farmer's working conditions and make them easier, we introduce IoT services, in which we use cloud services and internet to enable farmer to continue his work remotely via internet. He can monitor the field parameters and control the devices in farm.

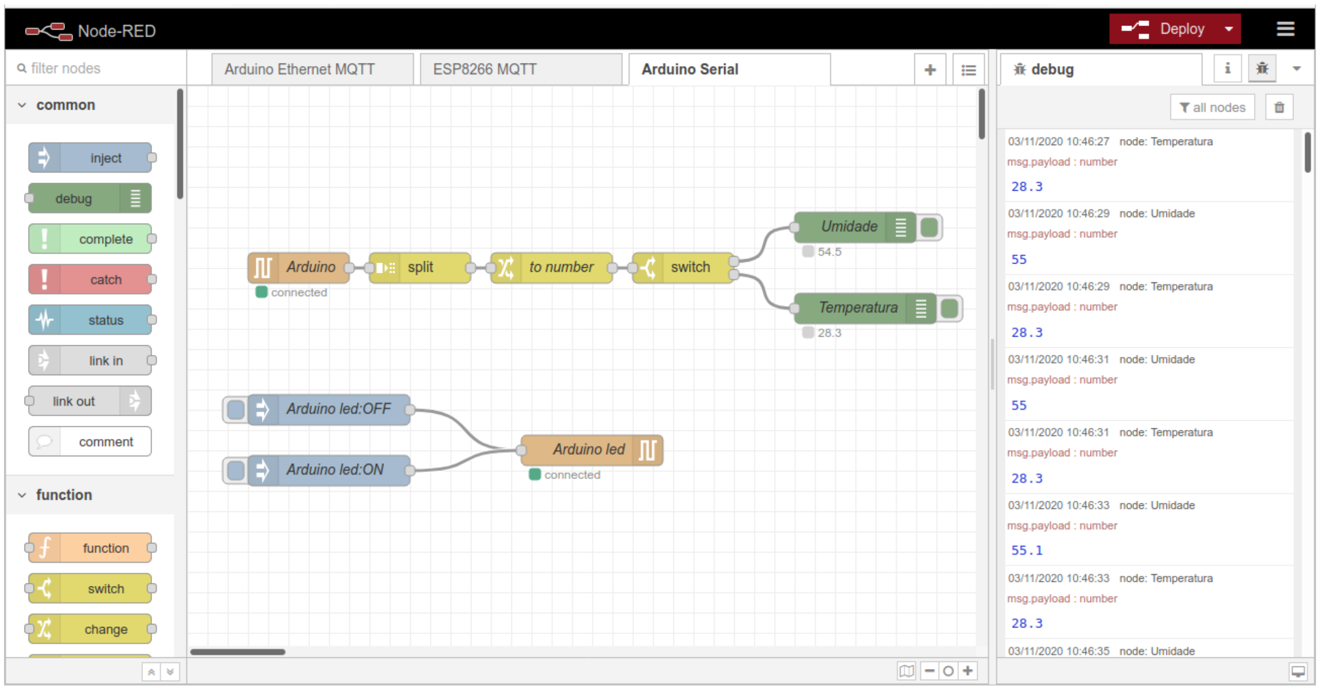
4.BLOCK DIAGRAM:



5. NODE RED SOFTWARE:

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.



```
03/11/2020 10:46:27   node: Temperatura
msg.payload : number
28.3
03/11/2020 10:46:29   node: Umidade
msg.payload : number
55
03/11/2020 10:46:29   node: Temperatura
msg.payload : number
28.3
03/11/2020 10:46:31   node: Umidade
msg.payload : number
55
03/11/2020 10:46:31   node: Temperatura
msg.payload : number
28.3
03/11/2020 10:46:33   node: Umidade
msg.payload : number
55.1
03/11/2020 10:46:33   node: Temperatura
msg.payload : number
28.3
03/11/2020 10:46:35   node: Umidade
```

6.IBM Watson IOT platform

Watson IoT Platform features Analytics and Watson APIs Completely manage your IoT landscape and make better business decisions. Using a secure, smart, and scalable platform as the hub of your IoT, get real-time analysis of user, machine and system-generated data, including speech, text video and social sentiment.

7.Phyton IDE

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Code:

```
import time

import sys import

ibmiotf.application import

ibmiotf.device import

random

#details of  IBM Watson Device Credentials

organization = "157uf3" deviceType = "abcd"

deviceId = "7654321" authMethod = "token"

authToken = "87654321"

def myCommandCallback(cmd): print("Command

received: %s" % cmd.data['command'])

status=cmd.data['command'] if status=="motoron":

print ("motor is on") elif status == "motoroff": print

("motor is off") else :

 print ("please send proper command")


try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId,

"auth-method": authMethod, "auth-token": authToken}

deviceCli = ibmiotf.device.Client(deviceOptions)

except Exception as e:

print("Caught exception connecting device: %s" % str(e))

sys.exit()
```

```
event of type "greeting" 10 times deviceCli.connect()
while True:
 #Get Sensor Data from DHT11

 temp=random.randint(90,110)
 Humid=random.randint(60,100)
Mois=random.randint(20,120)

 data = { 'temp' : temp, 'Humid': Humid, 'Mois' :Mois}
myOnPublishCallback():
print ("Published Temperature
= %s C" % temp, "Humidity = %s
%%" % Humid, "Moisture =%s
deg c" %Mois, "to IBM
Watson")
 success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback) if not success: print("Not connected
to IoTF") time.sleep(10)

 deviceCli.commandCallback = myCommandCallback
```

Aurdino code for C :

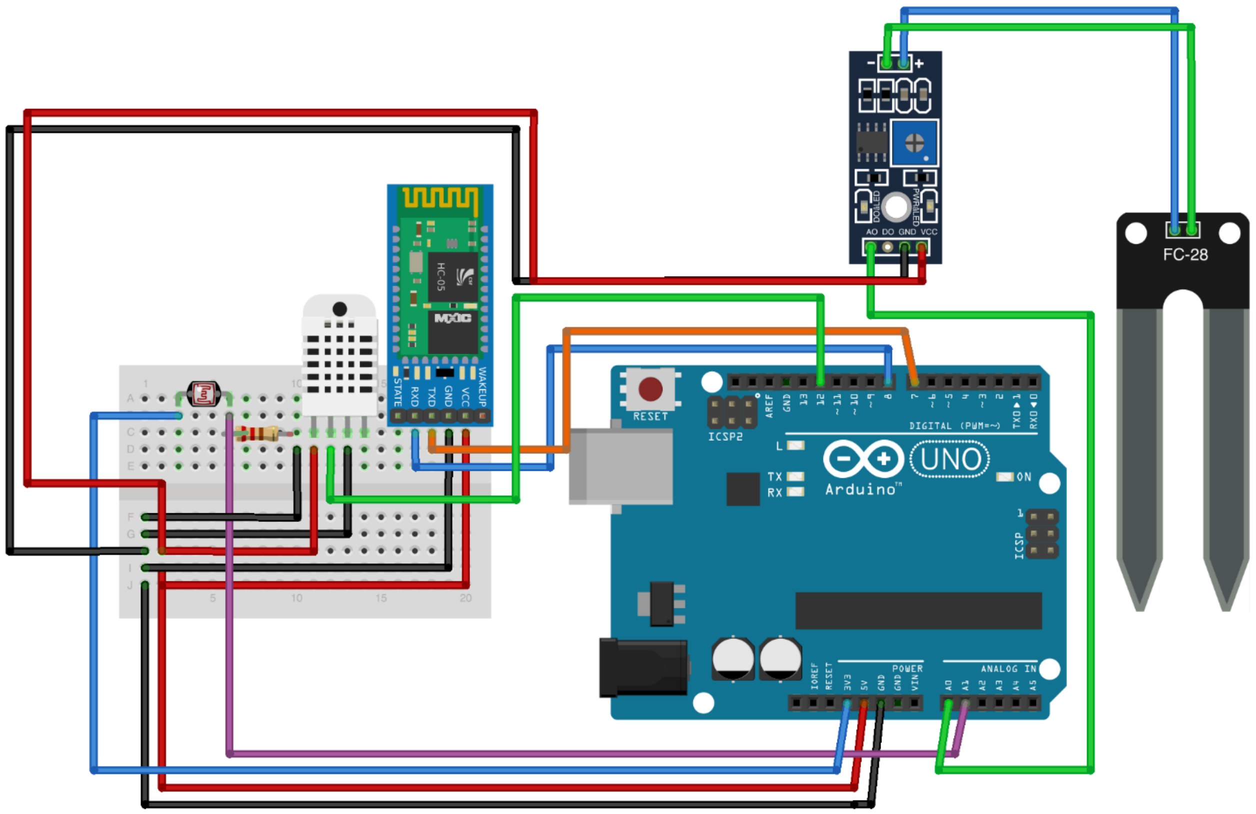
```
#include <dht.h>
#include <SoftwareSerial.h>

#define dht_apin A0
SoftwareSerial mySerial(7,8);
const int sensor_pin = A1;
int pin_out = 9;
dht DHT;
int c=0;

void setup()
{
pinMode(2, INPUT);
pinMode(3, OUTPUT);
pinMode(9, OUTPUT);
}
void loop()
{
if (digitalRead(2) == HIGH)
{
digitalWrite(3, HIGH);
delay(10000);
digitalWrite(3, LOW);
```

```
delay(100);
}
Serial.begin(9600);
delay(1000);
DHT.read11(dht_apin);
float h=DHT.humidity;
float t=DHT.temperature;
delay(5000);
Serial.begin(9600);
float moisture_percentage;
int sensor_analog;
sensor_analog = analogRead(sensor_pin);
moisture_percentage = ( 100 - ( (sensor_analog/1023.00) * 100 ) );
float m=moisture_percentage;
delay(1000);
if(m<40)
{
while(m<40)
{
digitalWrite(pin_out,HIGH);
sensor_analog = analogRead(sensor_pin);
moisture_percentage = ( 100 - ( (sensor_analog/1023.00) * 100 ) );
m=moisture_percentage;
delay(1000);
}
digitalWrite(pin_out,LOW);
}
if(c>=0)
{
mySerial.begin(9600);
delay(15000);
Serial.begin(9600);
delay(1000);
Serial.print("\r");
delay(1000);
Serial.print("AT+CMGF=1\r");
delay(1000);
Serial.print("AT+CMGS=\"+XXXXXXXXXX\"\r");
delay(1000);
Serial.print((String)"update>"+(String)"Temprature="+t+(String)"Humidity="+h+(S
tring)"Moisture="+m);
delay(1000);
Serial.write(0x1A);
delay(1000);
mySerial.println("AT+CMGF=1");
delay(1000);
mySerial.println("AT+CMGS=\"+XXXXXXXXXX\"\r");
mobile number
delay(1000);
```

```
mySerial.println((String)"update-
>"+(String)"Temprature="+t+(String)"Humidity="+h+(String)"Moisture="+m);
mySerial.println();
delay(100);
Serial.write(0x1A);
delay(1000);
c++;
}
}
```


HC-05
STATE
RXD
TXD
GND
VCC
WAKEUP
AO DO GND VCC
FC-28
RESET
ICSP2
DIGITAL (PWM=~)
UNO
Arduino
TX
RX
ON
ICSP
POWER
ANALOG IN