IOT ENABLED SMART

FARMING

APPLICATION.

Sprint Delivery – 1 TEAMID : PNT2022TMID44727

- ## Introduction

The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature, soil moisture, humidity and etc and control the equipment like water motor and other devices remotely via internet without their actual presence in the field.

- ## Problem Statement

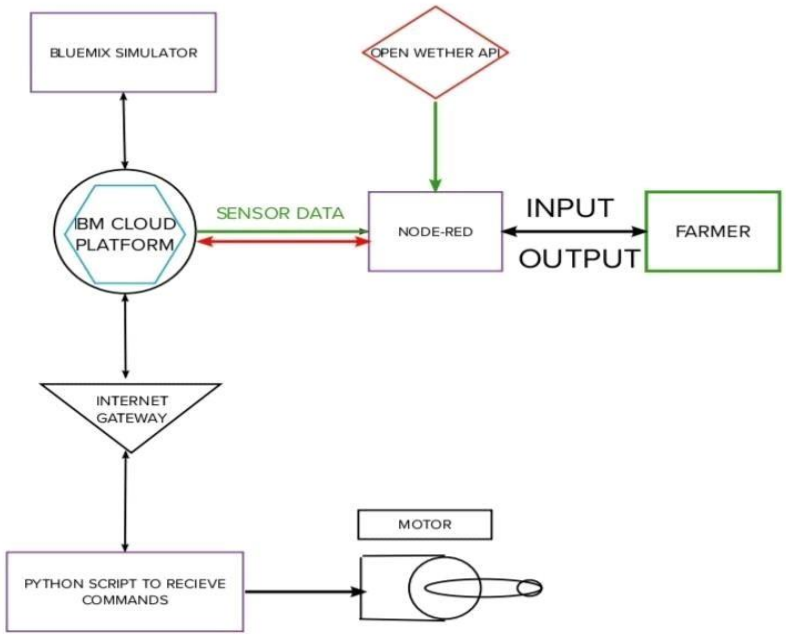
Farmers are to be present at farm for its maintenance irrespective of the weather conditions. They have to ensure that the crops are well watered and the farm status is monitored by them physically. Farmer have to stay most of the time in field in order to get a good yield. In difficult times like in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

- ## Proposed Solution

In order to improve the farmer's working conditions and make them easier, we introduce IoT services to him 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.

- ## Theoretical Analysis
- ## Block Diagram

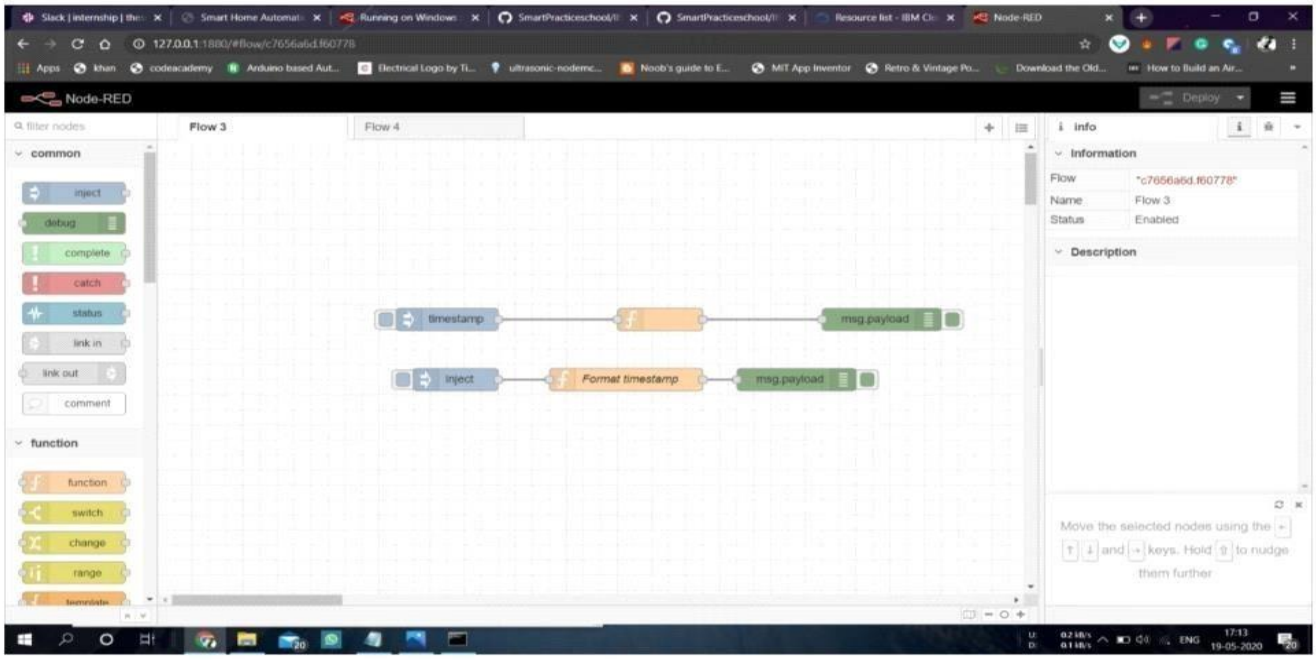
In order to implement the solution , the following approach as shown in the blockdiagram is used



- Required Software Installation
- Node-Red

Node-RED is a flow-based development tool for visual programming developedoriginally by IBM for wiring together hardware devices, APIs and online services as

part of the Internet of Things. Node-RED provides a web browser-based flow editor,which can be used to create JavaScript functions.



Installation :

- First install npm/node.js
- Open cmd prompt
- Type => npm install node-red

To run the application :

- Open cmd prompt
- Type=>node-red
- Then open http://localhost:1880/ in browser

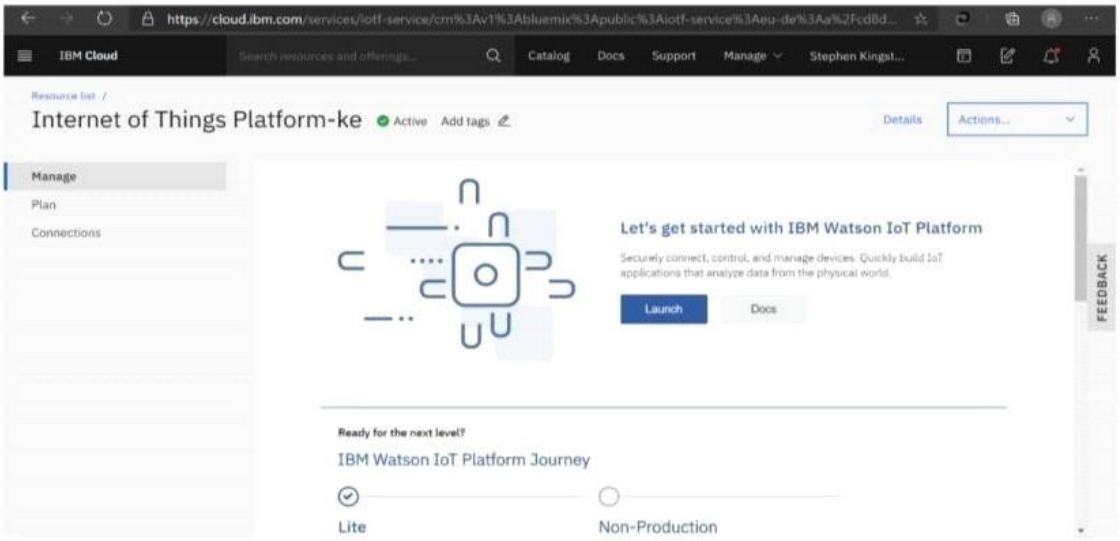
Installation of IBM IoT and Dashboard nodes for Node-Red

In order to connect to IBM Watson IoT platform and create the Web App UI these nodes are required 1. IBM IoT node

2. Dashboard node

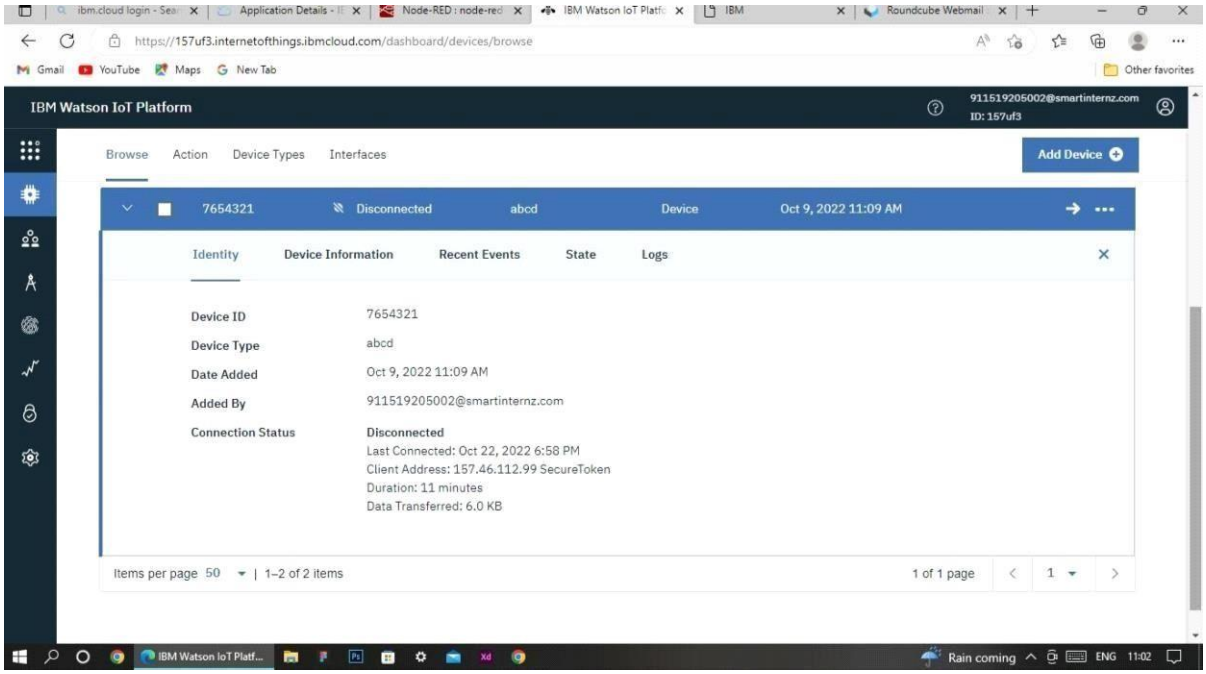
- IBM Watson IoT Platform

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization and data storage. IBM Watson IoT Platformis a managed, cloud-hosted service designed to make it simple to derive value fromyour IoT devices.



Steps to configure:

- Create an account in IBM cloud using your email ID
- Create IBM Watson Platform in services in your IBM cloud account
- Launch the IBM Watson IoT Platform
- Create a new device
- Give credentials like device type, device ID, Auth. Token
- Create API key and store API key and token elsewhere.



- Python IDE

Install Python3 compiler

Install any python IDE to execute python scripts, in my case I used Spyder to executethe code.

Code:

```
import time
import sys
import
ibmiotf.appl
ication
import
ibmiotf.devi
ce import
random



#Provide your IBM Watson
Device Credentialsorganization
= "157uf3" deviceType = "abcd"
```

```
deviceId = "7654321"
authMethod = "token"
authToken = "87654321"

# Initialize GPIO

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("
      print ("please send proper command")


try:
            deviceOptions = {"org": organization, "type": deviceType, "id":
            deviceId,
"auth-method":     authMethod,     "auth-token":     authToken}deviceCli = ibmiotf.dev
        #.....


except Exception as e:
        print("Caught   exception   connecting   device:
                        %s"          %              str(e))
sys.exit()


# Connect and send a datapoint "hello" with value "world" into the
cloud as anevent of type "greeting" 10 times deviceCli.connect()
```

```
while True:
        #Get Sensor Data from DHT11

        temp=random.ra
        ndint(90,110)
        Humid=random.r
        andint(60,100)

Mois=random.randint(20,120)

        data = { 'temp' : temp, 'Humid':
        Humid, 'Mois' :Mois}#print data  def
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 connectedto IoTF")  time.sleep(10)
```

```
    deviceCli.commandCallback = myCommandCallback


# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

Aurdino code for C :

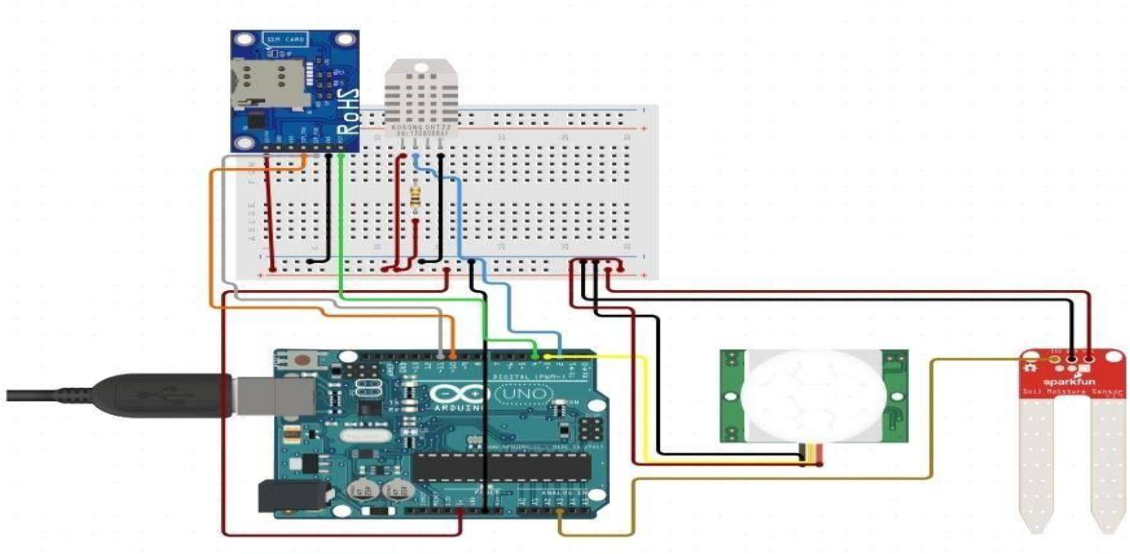
```
//in
 clu
 de
 libr
 ari
 es
 #in
 clu
 de
 <d
 ht.
 h>
 #include <SoftwareSerial.h>


 //define pins
 #define dht_apin A0 // Analog Pin sensor is
 connectedSoftwareSerial mySerial(7,8);//serial port
 of gsm
 const int sensor_pin = A1; // Soil moisture
 sensor O/P pinint pin_out = 9;
 //allo
 cate
 varia
```

```
bles
dht
DHT;
int c=0;


void setup()
{
pinMode(2, INPUT); //Pin 2
as INPUT pinMode(3,
OUTPUT); //PIN 3 as
OUTPUTpinMode(9,
OUTPUT);//output for
pump
}
void loop()
{
 if (digitalRead(2) == HIGH)
 {
 digitalWrite(3, HIGH); // turn the
 LED/Buzz ONdelay(10000); // wait for
 100 msecond digitalWrite(3, LOW);  //
 turn the LED/Buzz OFFdelay(100);
 }
  Seri
    al.b
    egi
    n(9
    600
    );
    del
    ay(
    100
    0);
```

```
 DHT.read11(dht_apin);
//tempraturefloat
h=DHT.humidity;
float
 t=DHT.te
 mperatur
 e;
 delay(500
 0);
 Serial.begi
 n(9600);
  float
moisture_percentage;//
moistureint
sensor_analog;
sensor_analog = analogRead(sensor_pin);
moisture_percentage = ( 100 - ( (sensor_analog/1023.00) * 100 ) );

float
m=moisture_pe
rcentage;
delay(1000);
if(m<40)//pump
{
while(m<40)
{
digitalWrite(pin_out,HIGH);
//open pump
sensor_analog =
analogRead(sensor_pin);
moisture_percentage = ( 100 - (
(sensor_analog/1023.00) * 100 ) );
m=moisture_percentage;
delay(1000);
```

```
  }
  digitalWrite(pin_out,LOW);//closepump
  }
  if(c>=0)
  {
  mySerial.begin(
  9600);
  delay(15000);
  Serial.begin(96
  00);
  delay(1000);
  Serial.print("\r
  "); delay(1000);
  Serial.print("AT
  +CMGF=1\r");
  delay(1000);
  Serial.print("AT+CMGS=\"+XXXXXXXXXX\"\r"); //replace X with
10 digit mobile number
  delay(1000);
  Serial.print((S
  tring)"update
  -
>"+(String)"Temprature="+t+(String)"Humidity="+h+(String)"Moistu
  re="+m); delay(1000);
  Seri
  al.w
  rite(
  0x1
  A);
  dela
  y(10
  00);
  mySerial.println("AT+CMGF=1");//Sets the GSM Module
  in Text Modedelay(1000);
```

```
  mySerial.println("AT+CMGS=\"+XXXXXXXXXX\"\r");    //replace
X with 10 digitmobile number
  delay(1000);
  mySerial.println((S
  tring)"update-
>"+(String)"Temprature="+t+(String)"Humidity="+h+(String)"Moistu
re="+m);// message format
  myS
  erial
  .prin
  tln()
  ;
  dela
  y(10
  0);
  Seri
  al.w
  rite(
  0x1
  A);
  dela
  y(10
  00);
  c++;

   }


}
```



- IoT Simulator

In our project in the place of sensors we are going to use IoT sensor simulator whichgive random readings to the connected cloud.

The link to simulator:

https://watson-iot-sensor-simulator.mybluemix.net/

We need to give the credentials of the created device in IBM Watson IoT Platformto connect cloud to simulator.

- OpenWeather API

OpenWeatherMap is an online service that provides weather data. It provides current weather data, forecasts and historical data to more than 2 million customer.

Website link:

https://openweathermap.org/guide

Stepsto configure:

- Create account in OpenWeather o

Find the name ofyour city by searching o

Create API key to your account

- Replace “city name” and “your api key” with your city and API key in below red text

api.openweathermap.org/data/2.5/weather?q={city name}&appid={your api key}