Sprint- 1

Team ID	**PNT2022TMID11481**
Project Title	**Smart Farmer -Iot Enabled Smart Farming Application**
Date	**12.11.2022**

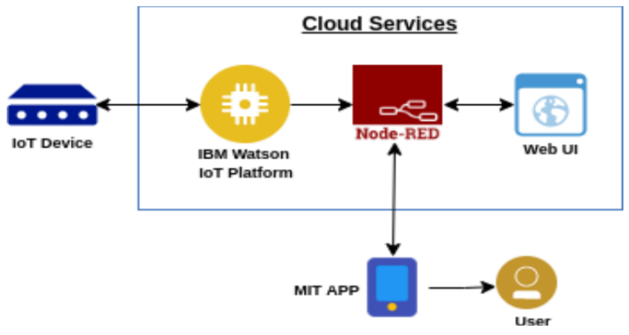
1. Introduction
The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature, soil moisture, humidity and etc and control the equipment like water motor and other devices remotely via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

2. Problem Statement
Farmers are to be present at the 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. Farmers have to stay in the field most of the time in order to get a good yield. In difficult times like in the presence of a pandemic also they have to 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 to him in which we use cloud services and the internet to enable farmers to continue his work remotely via the internet. He can monitor the field parameters and control the devices in farm

Theoretical Analysis : **Block Diagram**

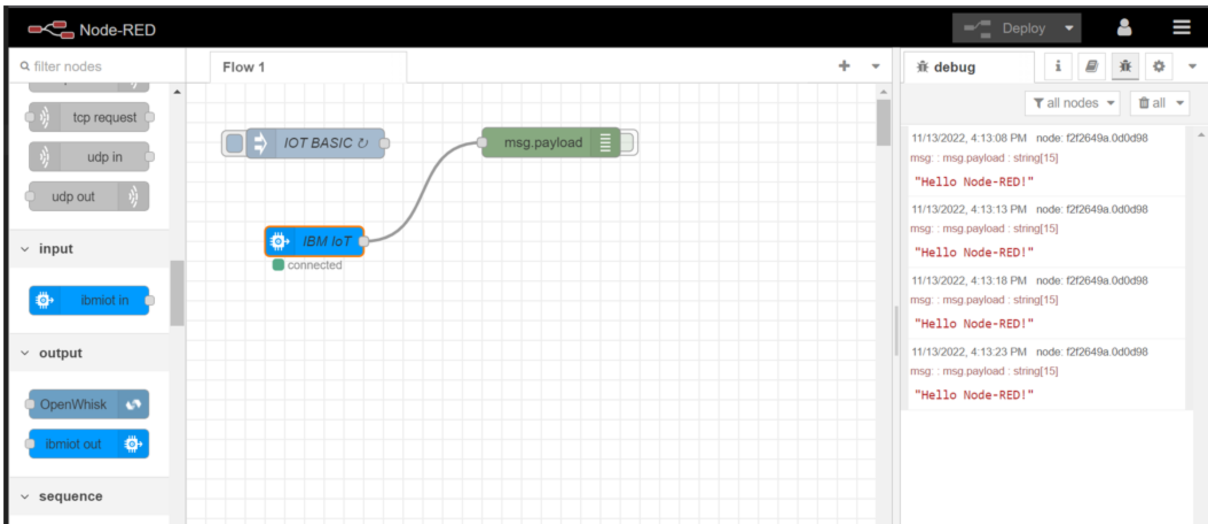
In order to implement the solution , the following approach as shown in the block diagram is used:



Required Software Installation :

Node-Red

It is a flow-based development tool for visual programming developed originally 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.



```
11/13/2022, 4:13:08 PM   node: f2f2649a.0d0d98
msg: : msg.payload : string[15]
"Hello Node-RED!"
11/13/2022, 4:13:13 PM   node: f2f2649a.0d0d98
msg: : msg.payload : string[15]
"Hello Node-RED!"
11/13/2022, 4:13:18 PM   node: f2f2649a.0d0d98
msg: : msg.payload : string[15]
"Hello Node-RED!"
11/13/2022, 4:13:23 PM   node: f2f2649a.0d0d98
msg: : msg.payload : string[15]
"Hello Node-RED!"
```

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

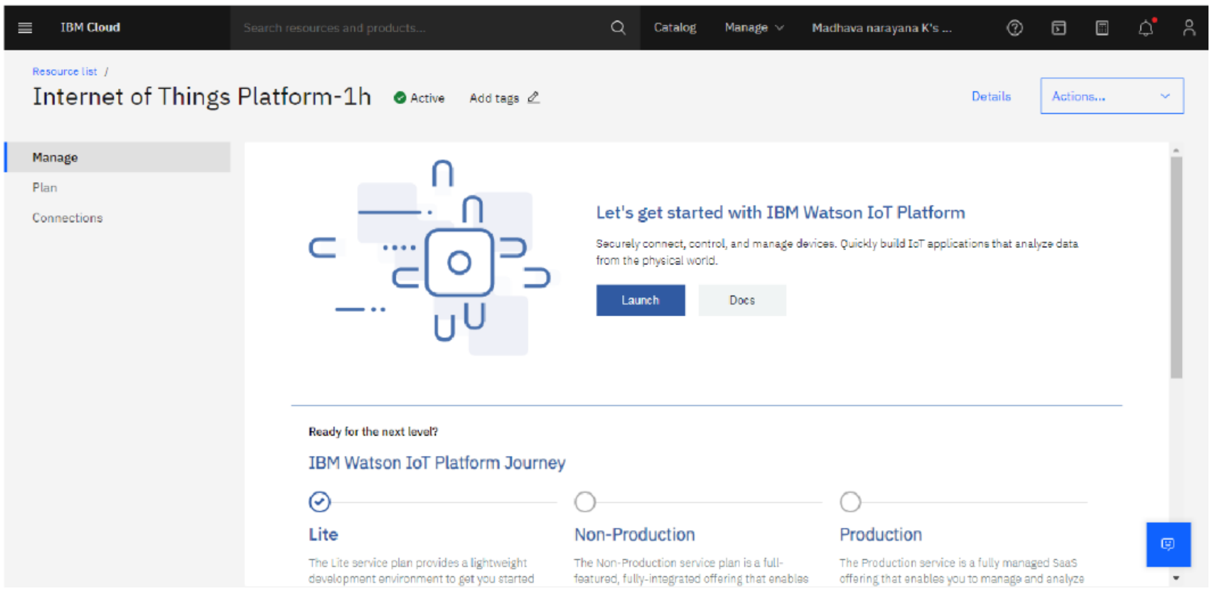
1. IBM IoT node
2. Dashboard node
2. Dashboard node

IBM Watson IoT Platform

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualisation and data storage. IBM Watson IoT Platform Is a managed, cloud hosted service designed to make it simple to derive value from your 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.



Let's get started with IBM Watson IoT Platform

Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.

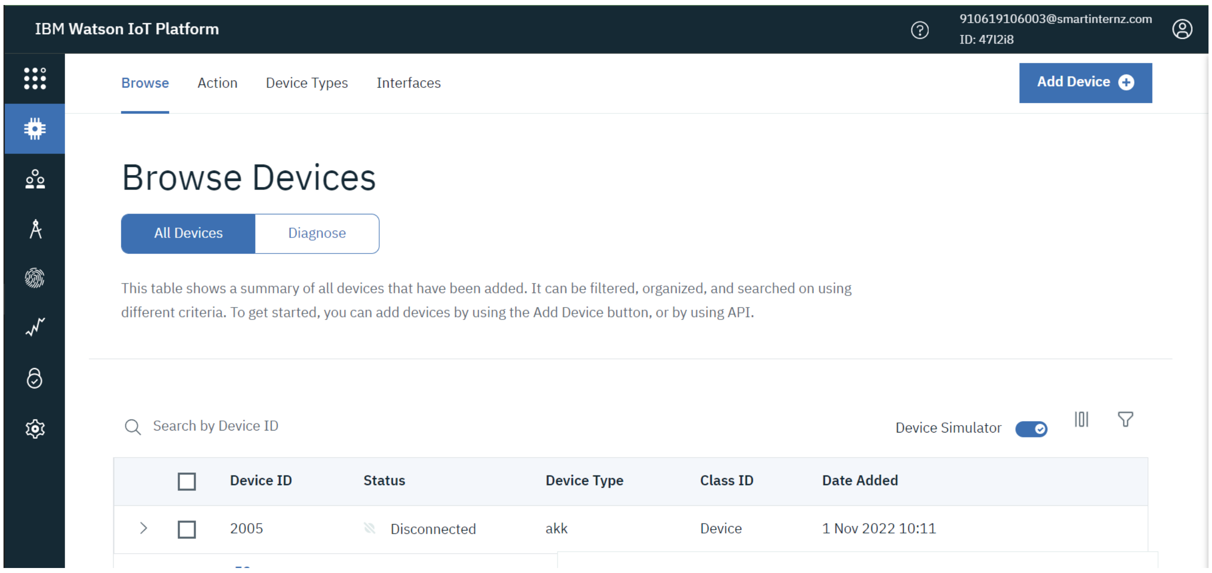
Ready for the next level?

IBM Watson IoT Platform Journey

Lite — The Lite service plan provides a lightweight development environment to get you started

Non-Production — The Non-Production service plan is a full-featured, fully-integrated offering that enables

Production — The Production service is a fully managed SaaS offering that enables you to manage and analyze

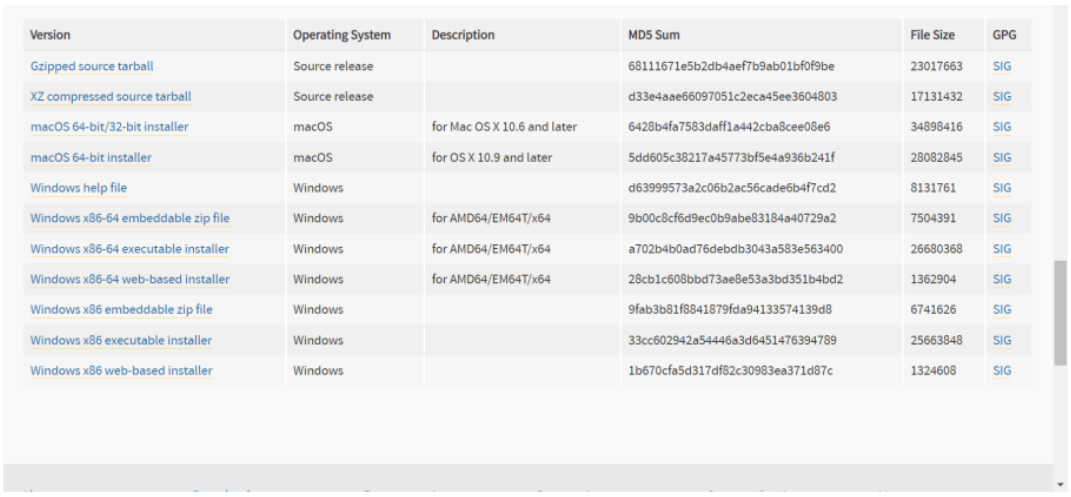


Browse Devices

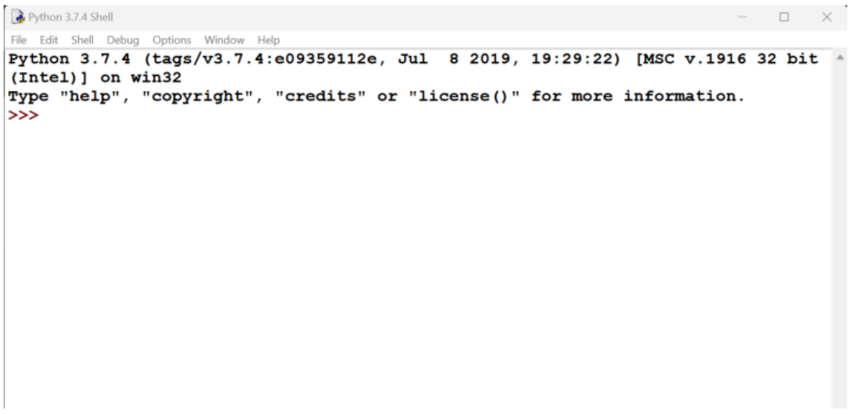
This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Device ID	Status	Device Type	Class ID	Date Added
2005	Disconnected	akk	Device	1 Nov 2022 10:11

Python IDE: 3.7.4



Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	macOS	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	macOS	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999573a2c06b2ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0b9abe83184a40729a2	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76debdb3043a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608bbd73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b81f8841879fda94133574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc602942a54446a3d6451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

Python Code:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "47l2i8"
deviceType = "akk"
deviceId = "2005"
authMethod = "token"
authToken = "akk12345"

# Initialize GPIO

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="switchon":
        print ("Switch is on")
    else :
        print ("Switch is off")

    #print(cmd)

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()

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

while True:
        #Get Sensor Data from DHT11

        temp=random.randint(0,100)
        Humid=random.randint(0,100)
        SoilMoisture=random.randint(0,100)


        data = { 'temp' : temp, 'Humid': Humid, "SoilMoisture": SoilMoisture}

        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid,
"SoilMoisture = %s %%" % SoilMoisture, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(1)

        deviceCli.commandCallback = myCommandCallback

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

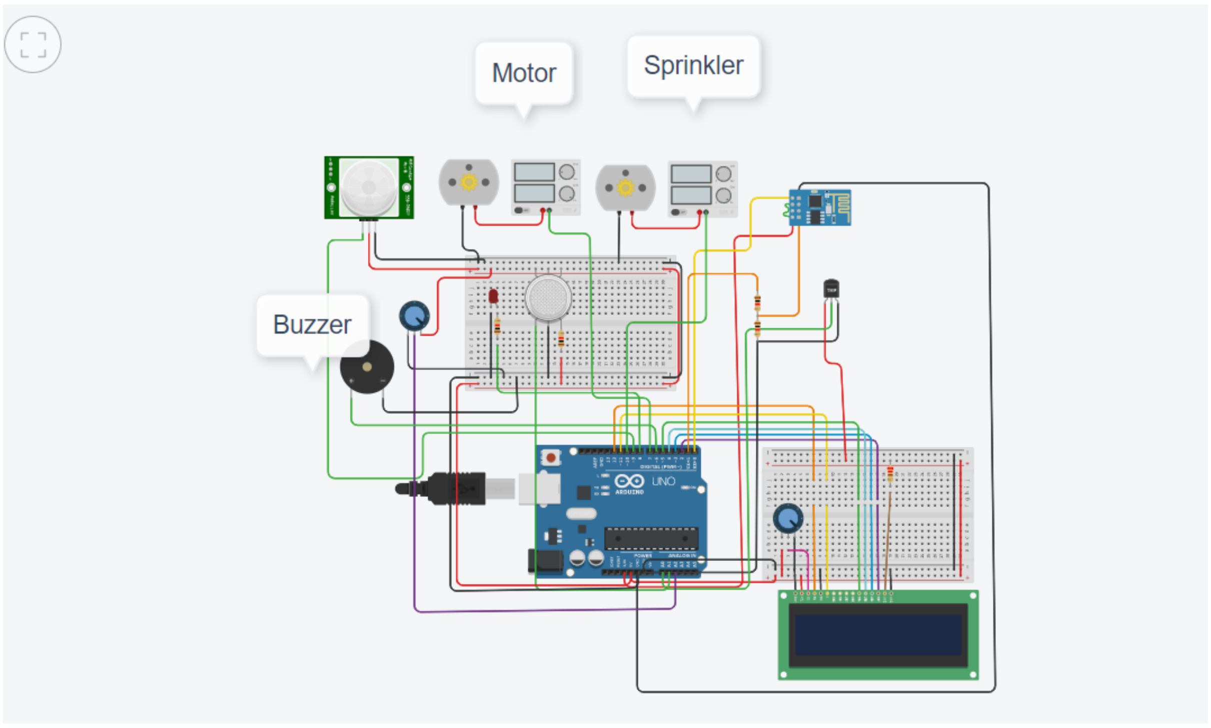
Tinkercad Simulation:

Arduino code for C :

```
String ssid     = "Simulator Wifi";  // SSID to connect to
String password = "";
String host     = "api.thingspeak.com";
const int httpPort   = 80;
String url     = "/update?api_key=6YDIQZLVKXPQN7GL&field1=";

int setupESP8266(void) {
  // Start our ESP8266 Serial Communication
  Serial.begin(115200);   // Serial connection over USB to computer
  Serial.println("AT");   // Serial connection on Tx / Rx port to ESP8266
  delay(10);        // Wait a little for the ESP to respond
  if (!Serial.find("OK")) return 1;

  // Connect to 123D Circuits Simulator Wifi
  Serial.println("AT+CWJAP=\"" + ssid + "\",\"" + password + "\"");
  delay(10);        // Wait a little for the ESP to respond
  if (!Serial.find("OK")) return 2;

  // Open TCP connection to the host:
  Serial.println("AT+CIPSTART=\"TCP\",\"" + host + "\"," + httpPort);
  delay(50);        // Wait a little for the ESP to respond
  if (!Serial.find("OK")) return 3;
  return 0;
}

#include<LiquidCrystal.h>
LiquidCrystal lcd(12,11,5,4,3,2);

#include<Servo.h>;
Servo servo;

int air;
int motor=7;
int buzz=6;
int sprinkler=10;
int led=8;
int sensor=9;
int temp;
int pir;
float mois;
byte degree[8]={
  B00110,
  B01001,
  B01001,
  B00110,
  B00000,
  B00000
};

void setup(){
  lcd.begin(16,2);
  setupESP8266();
  Serial.begin(9600);
  pinMode(sensor,INPUT);
  pinMode(A0,INPUT);
  pinMode(A1,INPUT);
  pinMode(A2,INPUT);
  pinMode(buzz,OUTPUT);
  pinMode(sprinkler,OUTPUT);
  pinMode(motor,OUTPUT);
  pinMode(led,OUTPUT);

}

void senddata(void) {
  int temp = map(analogRead(A0),20,358,-40,125);
  // Construct our HTTP call
  String httpPacket = "GET " + url + String(temp) + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n";
  int length = httpPacket.length();

  // Send our message length
  Serial.print("AT+CIPSEND=");
  Serial.println(length);
  delay(10); // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

  // Send our http request
  Serial.print(httpPacket);
  delay(10); // Wait a little for the ESP to respond
  if (!Serial.find("SEND OK\r\n")) return;
}


void loop() {
  senddata();
  delay(20);
  air=map(analogRead(A1),0,358,0,125);
  temp=map(analogRead(A0),20,358,-40,125);
  mois=map(analogRead(A2),0,5,0,1);

  if(mois<0.5)
  {
    digitalWrite(motor,HIGH);
    lcd.setCursor(0,1);
    lcd.print("low moisture, Motor on");
    delay(10);
  }
  else if(temp>=75)
  {
    digitalWrite(sprinkler,HIGH);
    digitalWrite(led,HIGH);
    delay(10);
  }
  else
  {
    digitalWrite(sprinkler,LOW);
    digitalWrite(led,LOW);
    digitalWrite(motor,LOW);
  }
  pir=digitalRead(sensor);
  if(pir==1)
  {
    digitalWrite(buzz,HIGH);
  }
  else if(pir==0)
  {
    digitalWrite(buzz,LOW);
  }
  //Temperature:
  lcd.createChar(0,degree);
  lcd.clear();
  lcd.print("Temp:");
  lcd.print(temp);
  lcd.write(byte(0));
  lcd.print("C");
  if(mois<0.5)
  {
    lcd.setCursor(0,1);
    lcd.print("low moisture, Motor on");
  }
  else if(temp>=75)
```

```
    {
      lcd.setCursor(0,1);
      lcd.print("FIRE! EVACUATE!!");
    }
    delay(1000);
    //Air Quality:
    lcd.clear();
    lcd.print("AirQ:");
    lcd.print(air);
    lcd.print("ppm");
    lcd.setCursor(0,1);
    //Door
    if(pir==1)
    {
    lcd.print("Intruder");
    }
  }
```

Link: https://www.tinkercad.com/things/9YLynkNdia5


Motor
Sprinkler
Buzzer
UNO