

# Smart Farmer - IoT Enabled Smart Farming Application

## SPRINT-3

<b>Team ID</b>	PNT2022TMID35924
<b>Project name</b>	Smart farmer- IoT enabled smart farming

In this session, we have designed added extra features, which is novelty of our project i.e. detection of flame level (Flame sensor), Nitrogen, Phosphorus and Potassium level (NPK sensor) of the soil along with moisture, humidity & temperature of the field.

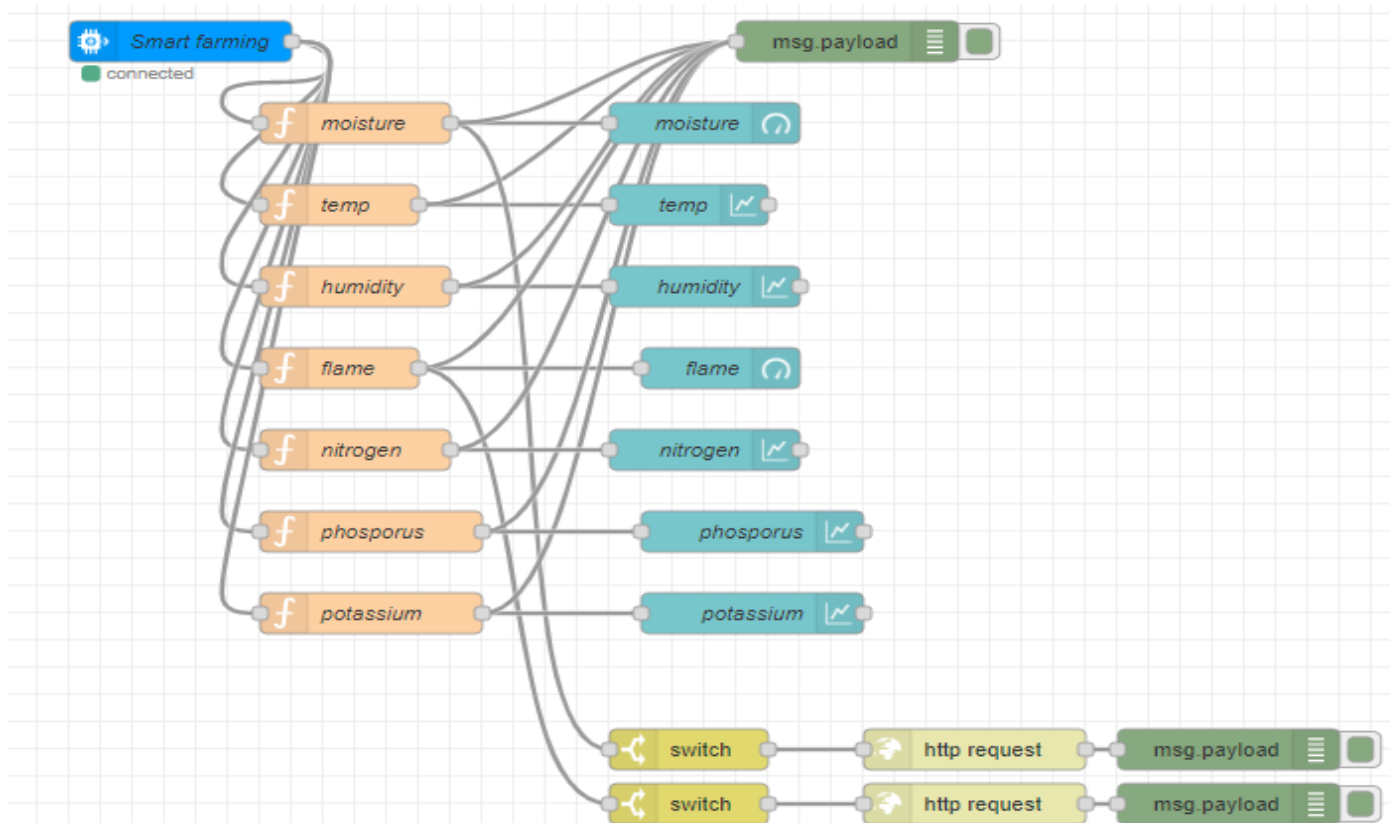
Here is the python code:

IBM (1).py - C:\Users\User\Downloads\IBM (1).py (3.7.0)

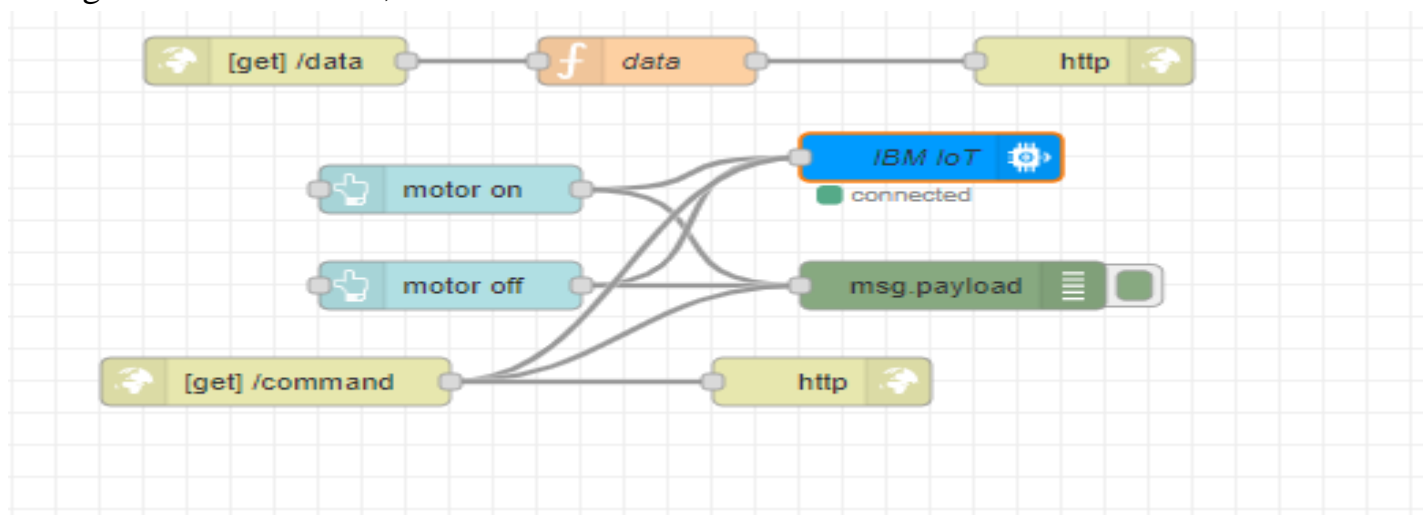
File Edit Format Run Options Window Help

```
import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig = {"identity":{"orgId": "jx4tsh",
                        "typeId": "ESP32",
                        "deviceId": "vatsaaa"},
            "auth":{"token": "12345678"}}
client=wiotp.sdk.device.DeviceClient (config=myConfig, logHandlers=None)
client.connect()
def myCommandCallback(cmd):
    print ("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="motoron"):
        print ("Motor is switched ON")
    elif(m=="motoroff"):
        print ("Motor is switched OFF")
    print(" ")
while True:
    moisture = random.randint (0, 100)
    temp = random.randint (0,100)
    humidity = random.randint (0,100)
    flame= random.randint(0,100)
    nitrogen=random.randint(0,100)
    phosphorus=random.randint(0,100)
    potassium=random.randint(0,100)
    myData={'moisture': moisture, 'temp': temp, 'humidity': humidity,
            'flame': flame, 'nitrogen': nitrogen, 'phosphorus': phosphorus, 'potassium': potassium}
    client.publishEvent (eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
    print("Published data Successfully: %s", myData)
    time.sleep (2)
    client.commandCallback = myCommandCallback
client.disconnect()
```

Next, we designed Node- RED flow along with web UI for our project. The designed Node- RED window is shown below.



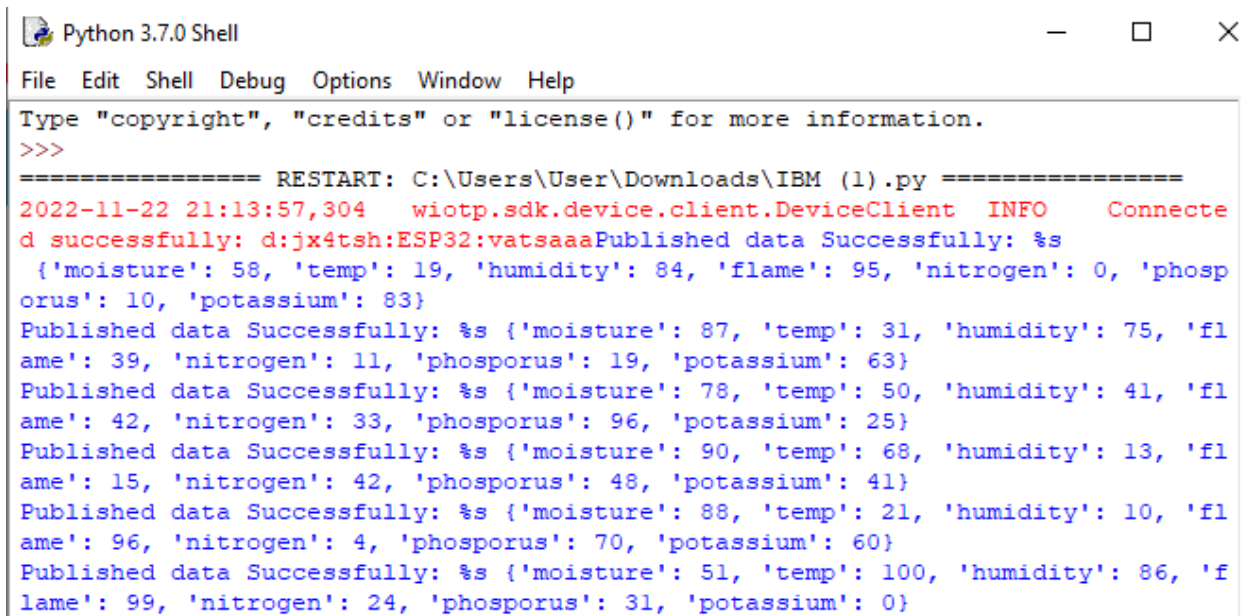
In this flow, we have integrated to cloud in Node- RED to transfer the information to cloud. In IBM Iot Input node, we have given the details of the device which was created in IBM cloud using API key. Then, we have included all the required field parameters and have used graphical representation for these through Gauges & Charts. Then, finally, we have connected Switch button to moisture & flame. When the values of both these parameters are below some threshold, the famer will get an alert message through SMS and for this, we have used Fast2sms service.



Then, here, we have used http in & http request function for transfer of data from Node- RED to the app. Also, we have used IBM IoT output node & 2 buttons to switch ON & OFF the motor present in the field remotely according to the requirement. The command for switching ON & OFF the motor shall be given by farmer either through the app or through the web UI. Then, we have used another http in & http request function along with button to transfer the command from app/UI to the Node- RED & ultimately to Python.

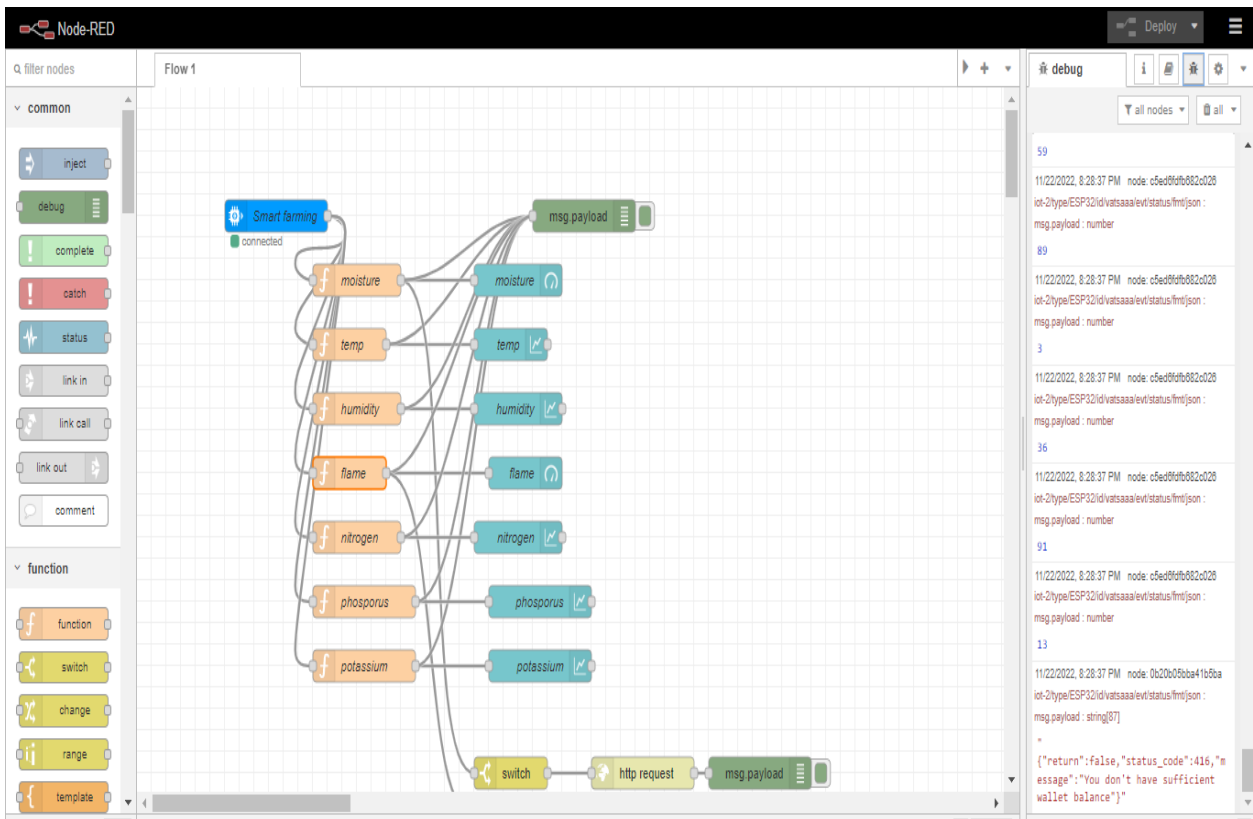
## OUTPUT:

### Python IDLE:

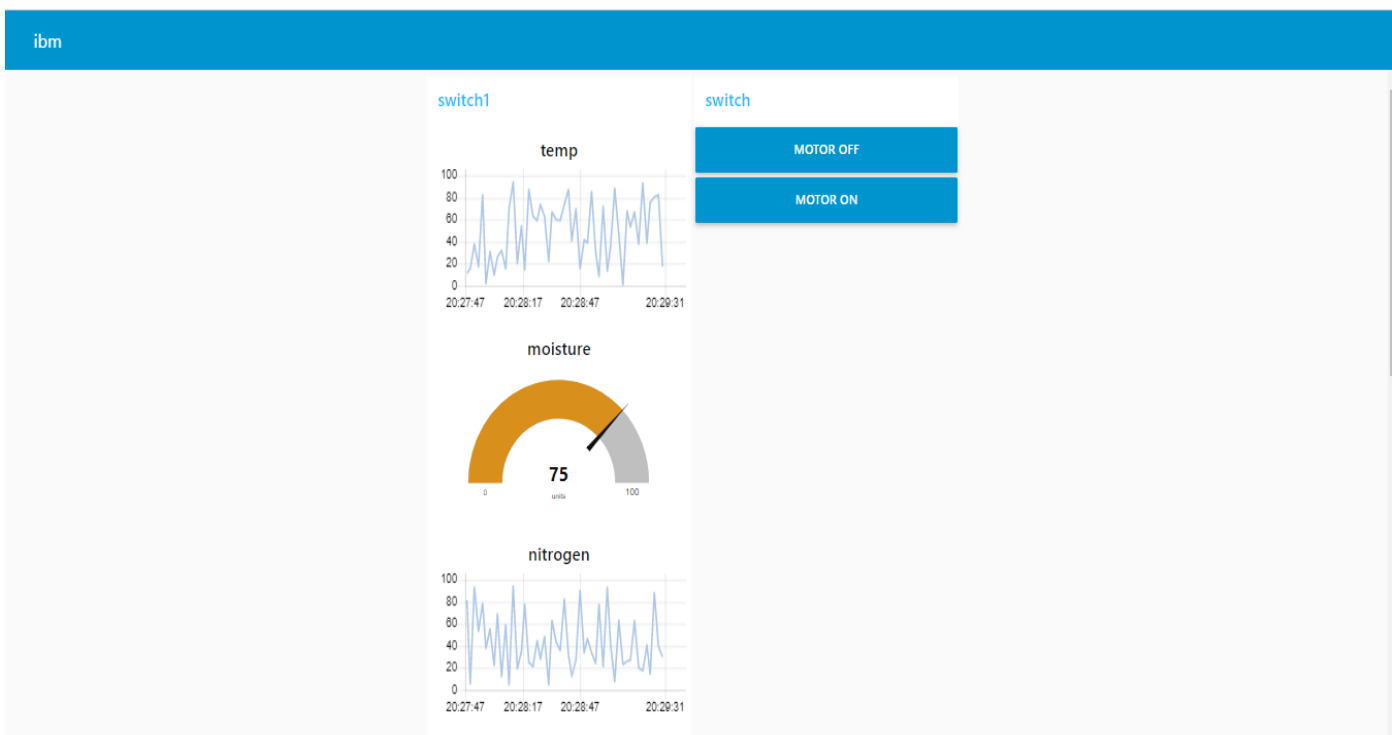


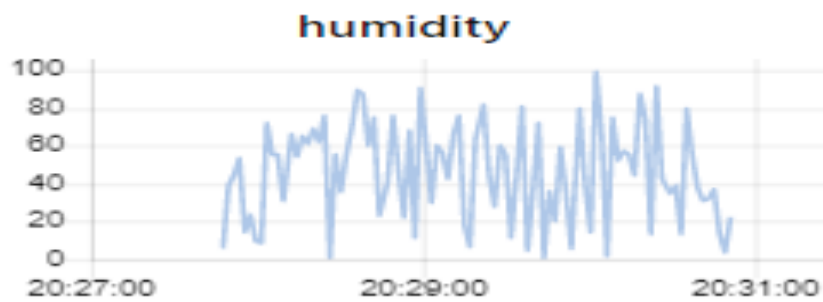
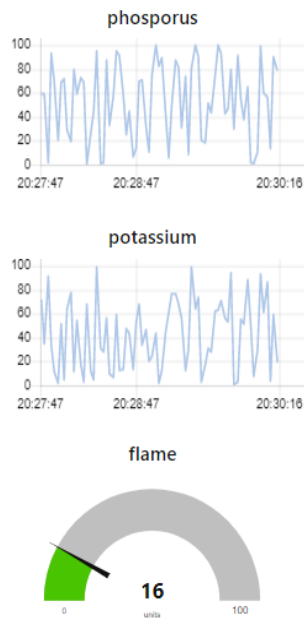
```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\User\Downloads\IBM (1).py =====
2022-11-22 21:13:57,304 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:jx4tsh:ESP32:vatsaaaPublished data Successfully: %s
{'moisture': 58, 'temp': 19, 'humidity': 84, 'flame': 95, 'nitrogen': 0, 'phosphorus': 10, 'potassium': 83}
Published data Successfully: %s {'moisture': 87, 'temp': 31, 'humidity': 75, 'flame': 39, 'nitrogen': 11, 'phosphorus': 19, 'potassium': 63}
Published data Successfully: %s {'moisture': 78, 'temp': 50, 'humidity': 41, 'flame': 42, 'nitrogen': 33, 'phosphorus': 96, 'potassium': 25}
Published data Successfully: %s {'moisture': 90, 'temp': 68, 'humidity': 13, 'flame': 15, 'nitrogen': 42, 'phosphorus': 48, 'potassium': 41}
Published data Successfully: %s {'moisture': 88, 'temp': 21, 'humidity': 10, 'flame': 96, 'nitrogen': 4, 'phosphorus': 70, 'potassium': 60}
Published data Successfully: %s {'moisture': 51, 'temp': 100, 'humidity': 86, 'flame': 99, 'nitrogen': 24, 'phosphorus': 31, 'potassium': 0}
```

Node- RED flow:



Web UI:





## IBM CLOUD:

IBM Watson IoT Platform

srivatsanmohan2001@gmail.com  
ID: px4tsh

Browse Action Device Types Interfaces

Add Device +

▼ vatsaaa Connected ESP32 Device Oct 27, 2022 3:14 PM → ...

Identity Device Information **Recent Events** State Logs X

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
status	{"moisture":48,"temp":30,"humidity":2,"flame":8...	json	a few seconds ago
status	{"moisture":44,"temp":28,"humidity":25,"flame":...	json	a few seconds ago

Items per page 50 | 1-2 of 2 items

1 of 1 page < 1 >

**When “Motor ON” button is pressed in web UI, the output in Python is observed below:**

```
Message received from IBM IoT Platform: motoron  
Motor is switched ON
```

**When “Motor OFF” button is pressed in web UI, the output in Python is observed below:**

```
Message received from IBM IoT Platform: motoroff  
Motor is switched OFF
```