SMART FARMER- IOT BASED SMART FARMER APPLICATION

DHABASVINI T

PETHANATCHI C

POOJA K

PRIYADHARSHINI K P

SHANGEETHA G

CONTENTS

1. INTRODUCTION

The main objective 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

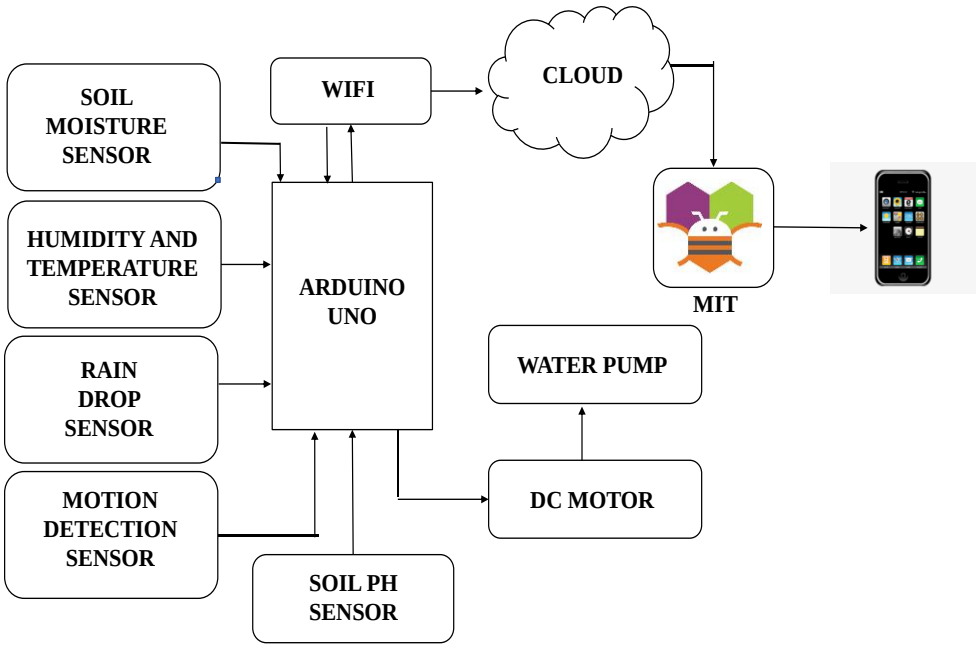
Farmers are to be present at farm for its maintenance irrespective of the weather conditions. They must ensure that the crops are well watered and the farm status is monitored by them physically. Farmer must stay most of the time in field in order to get a good yield. In difficult times like in the presence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

3. PROPOSED SOLUTION

The environmental condition using sensors with Wi-fi technology to connect the sensors and transfer the data to the central database. The system to be designed will collect the data from soil moisture sensor, Humidity and Temperature sensor. Arduino microcontroller board to connect all IoT devices and use the cloud platform to transfer and store the information/ data collected from the sensors.

4. THEORETICAL ANALYSIS

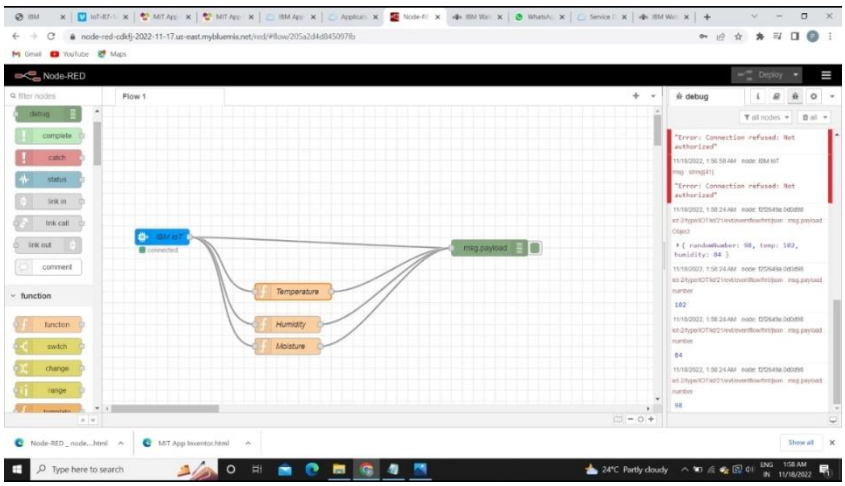
4.1 BLOCK DIAGRAM



4.2 REQUIRED SOFTWARE INSTALLATION

4.2.A NODE- RED

Node-RED 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.



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

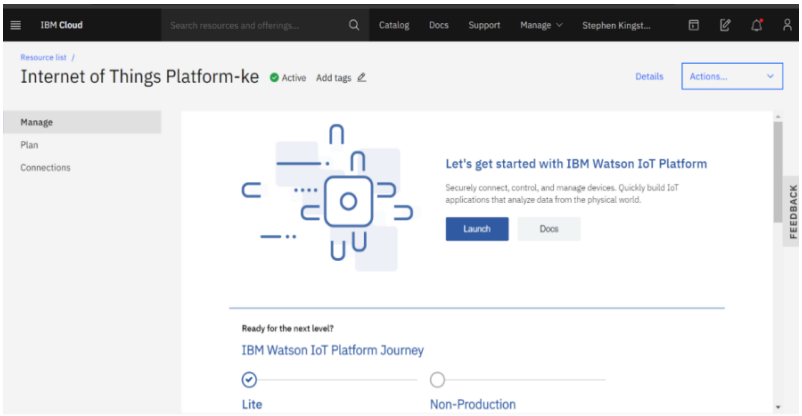
Installation of IBM IoT and Dashboard nodes for Node-Red

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

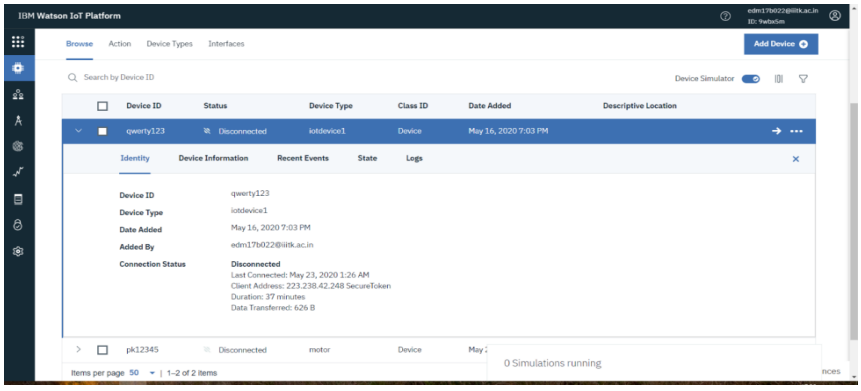
4.2.B IBM WATSON IOT PLATFORM

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization, and data storage. IBM Watson IoT 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



4.2.C PYTHON IDE

Install Python3 compiler

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

4.3 IOT SIMULATOR

In our project in the place of sensors we are going to use IoT sensor simulator which give 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 Platform to connect cloud to simulator

5. BUILDING PROJECT

5.1 CONNECTING IOT SIMULATOR TO IBM WATSON IOT PLATFORM

Open link provided in above section 4.3

Give the credentials of your device in IBM Watson IoT Platform

Click on connect

My credentials given to simulator are:

OrgID: 9wbx5

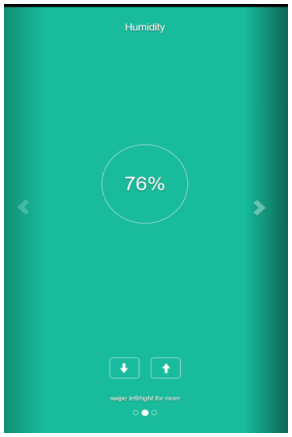
api: a-9wbx5m-1qfklrf7jl

Device type: iotdevice1

token: JcU(4(9Z37PdL!Rmz(

Device ID : qwerty123

Device Token : johnyjohnyyespapa



➢ You will receive the simulator data in cloud

➢ You can see the received data in Recent Events under your device

➢ Data received in this format(json)

{

"d": {

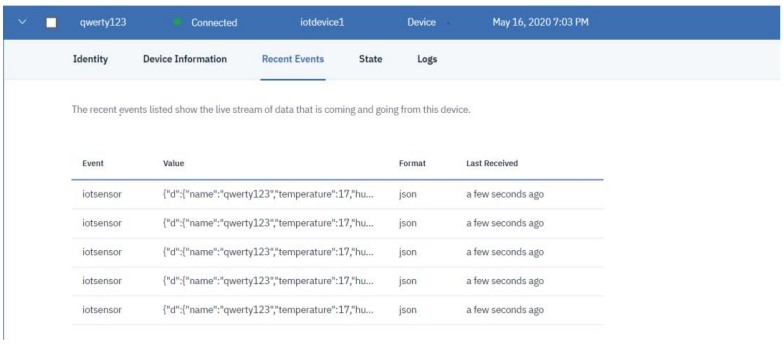
"name": "qwerty123",

"temperature": 17,

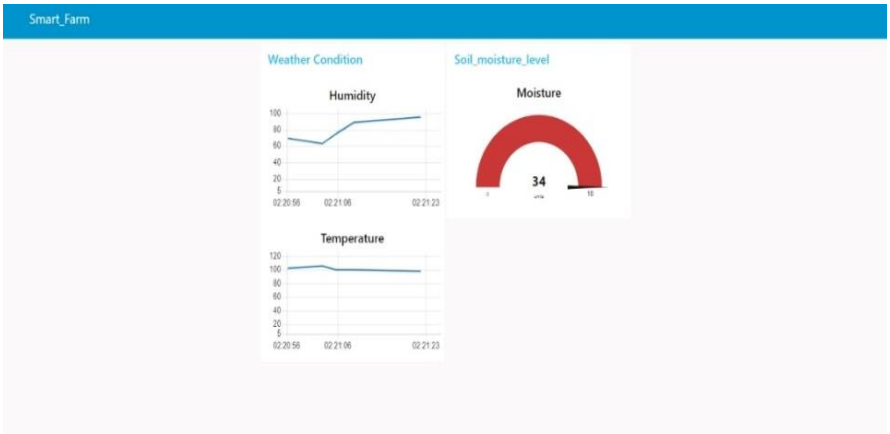
"humidity": 76,

"objectTemp": 25

} }

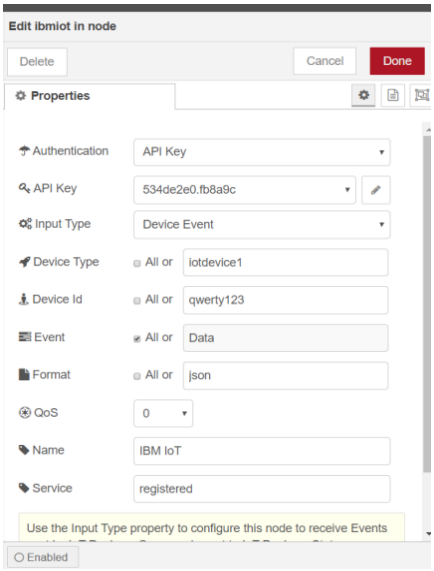


You can see the received data in graphs by creating cards in Boards tab



5.2 CONFIGURATION OF NODE- RED TO COLLECT IBM CLOUD DATA

The node IBM IoT App In is added to Node-Red workflow. Then the appropriate device credentials obtained earlier are entered into the node to connect and fetch device telemetry to Node-Red



Once it is connected Node-Red receives data from the device

Display the data using debug node for verification

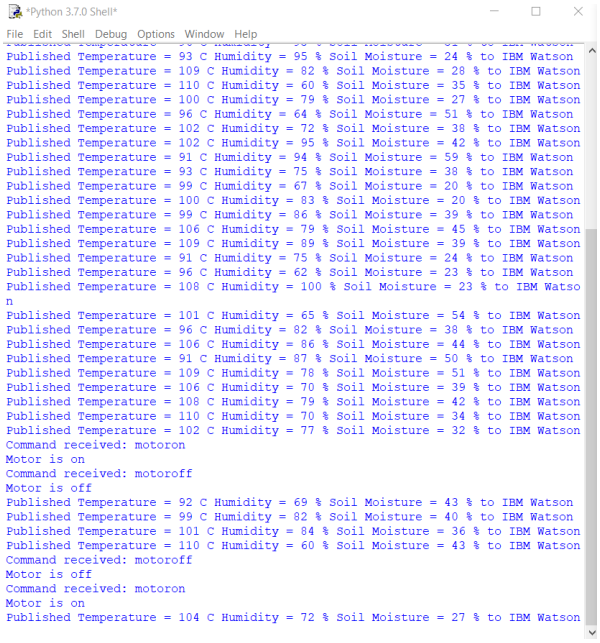
Connect function node and write the Java script code to get each reading separately.

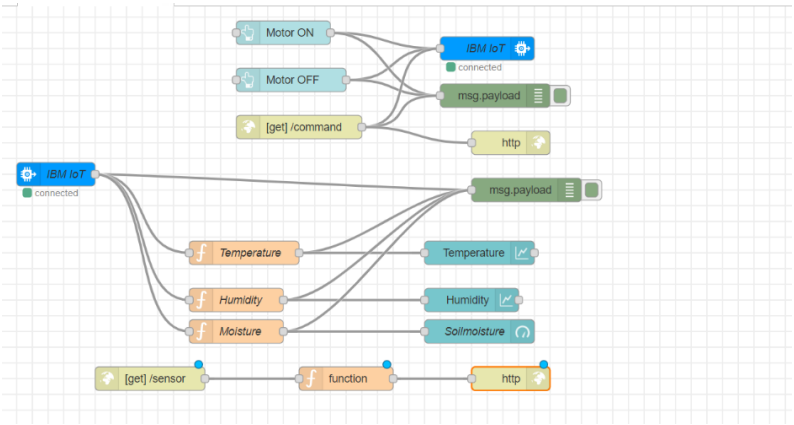
The Java script code for the function node is:

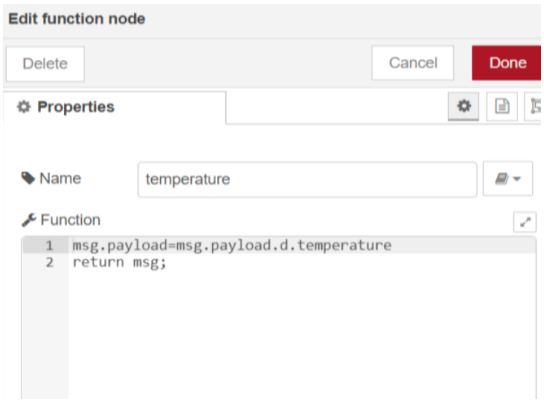
```
msg.payload=msg.payload.d.temperature

return msg;
```

Finally connect Gauge nodes from dashboard to see the data in UI



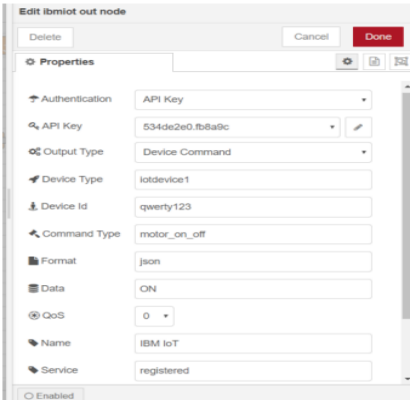




This is the Java script code I written for the function node to get Temperature separately

5.3 CONFIGURATION OF NODE- RED TO SEND COMMANDS TO IBM CLOUD

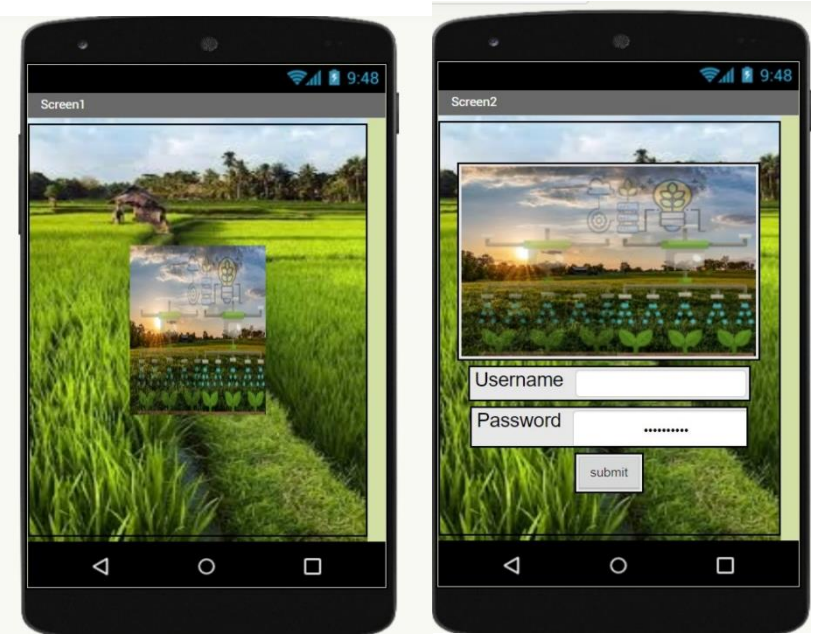
ibmiot out node I used to send data from Node-Red to IBM Watson device. So, after adding it to the flow we need to configure it with credentials of our Watson device.



Here we add two buttons in UI for motor ON and OFF.

We used a function node to analyse the data received and assign command to each number.

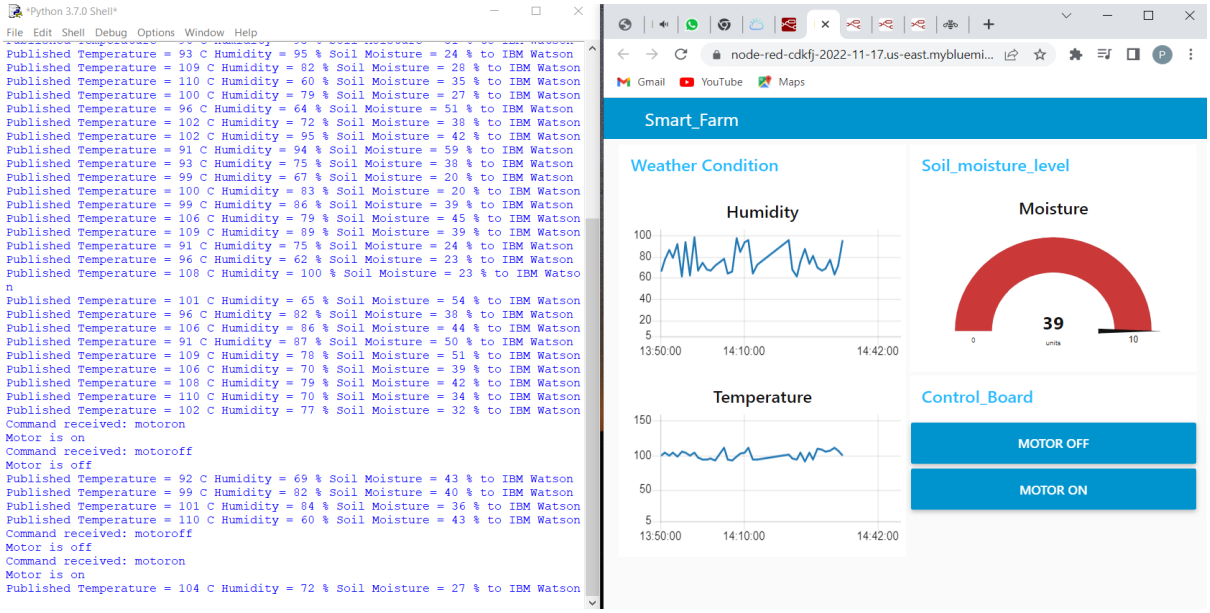
Web APP UI



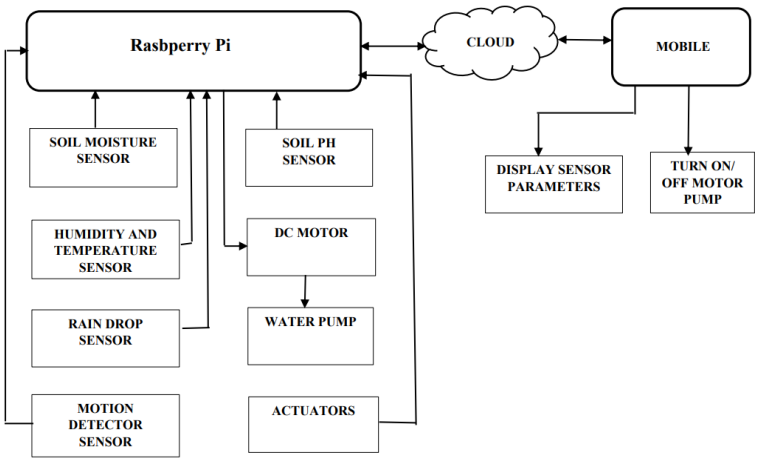
5.4 RECEIVING COMMANDS FROM IBM CLOUD USING PYTHON

```
Import time
import sys
import ibmiotf.application
import ibmiotf.device
import paho.mqtt.client as mqtt
import random
organization="6eut6z"
deviceType="IOT"
deviceId="21"
authMethod="token"
authToken="12345678"
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 connection device:%s" %str(e))
    sys.exit()
while True:
    temp=random.randint(90,110)
    humidity=random.randint(60,100)
    data={'Temperature':temp,'Humidity':humidity}
    def myonPublishCallback():
```

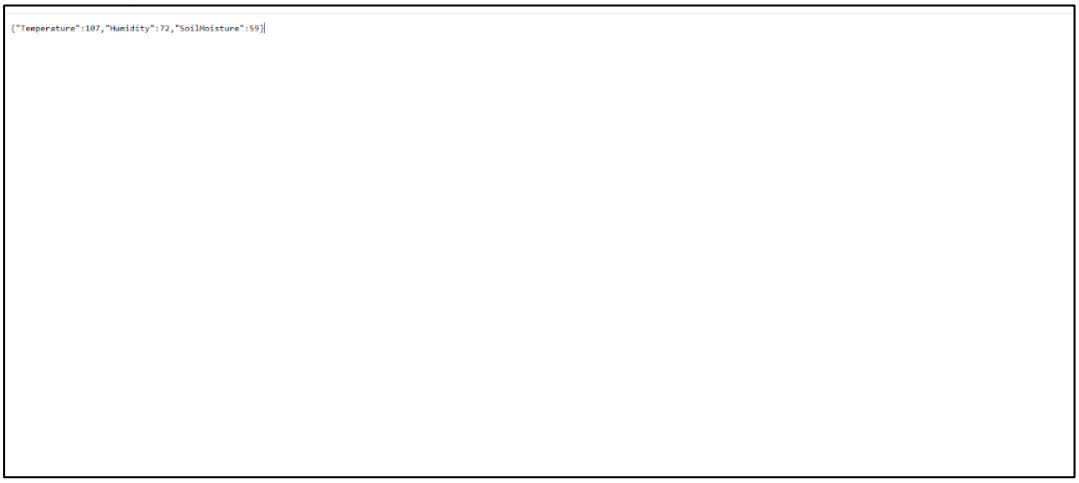
```
print("published Temperature=%s C"%temp,"Humidity=%s%%" %humidity,"to IBM Watson")
    success=deviceCli.publishEvent("IoTSensor","json",data,qos=0,on_publish=myonPublishCallback)
    if not success:
        print("Not connected to IoTF")
    time.sleep(10)
    deviceCli.commandCallback=myCommandCallback
deviceCli.disconnect()
```



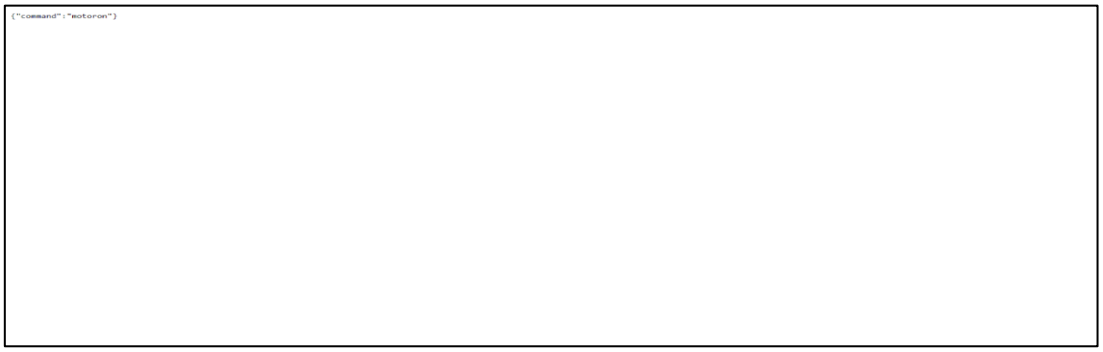
6. **FLOW DIAGRAM**



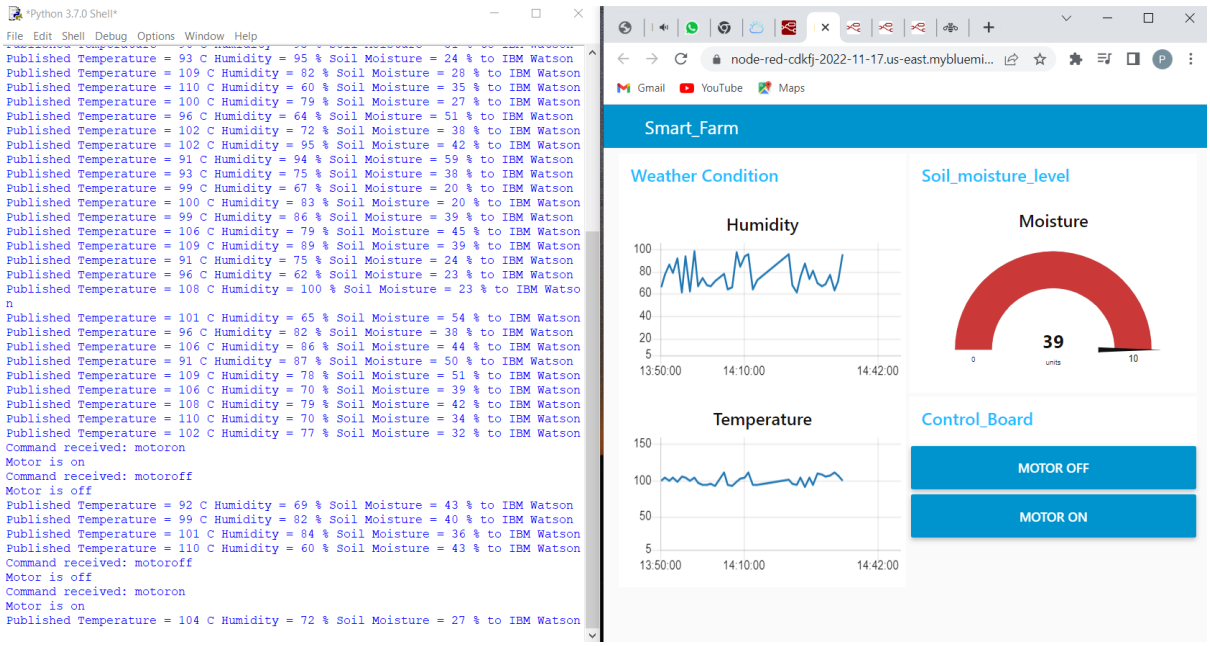
7. OBSERVATIONS AND RESULTS



Node- red output



Node- red output



Python IDE output

MIT app inventor output

8. ADVANTAGES AND DISADVANTAGES

Advantages:

- Farmers can take care of the land remotely
- Reduction in the labour cost
- Increase in productivity

Disadvantages:

- Lack of internet
- Farmers must have a good knowledge about web applications

9. CONCLUSION

The significance of this project is that manual work is reduced, and watering is automated. Healthy plants can be grown with limited use of water and electricity. Even elderly people can easily do farming. IOT plays a major role in agricultural field. This paper is mainly applied to agricultural field. It helps us to achieve a healthy farming. Increase in agriculture also helps us to increase the economical state of the country.

10. BIBILIOGRAPHY

IBM cloud reference: https://cloud.ibm.com/

Python code reference: https://github.com/rachuriharish23/ibmsubscribe

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