

# **IOT BASED SMART CROP PROTECTION SYSTEMFOR AGRICULTURE**

**TEAM ID: PNT2022TMID54090**

**TEAM LEADER: DHARUN.E**

**TEAM MEMBER 1: GOKULAKRISHNAN.B**

**TEAM MEMBER 2: MOHAN RAJ.M**

**TEAM MEMBER 3: MULANGI CHENCHU KRISHNA TEJA**

## **1. INTRODUCTION:**

### **a. Overview :**

- i. This project is based onInternet Of Things (IoT), that can measure soil moisture,Humidityand temperature conditions for

agriculture and crop protection using Watson IoT services. IoT is a network that connects physical objects or things embedded with electronics, software and sensors through network connectivity that collects and transfers data using cloud for communication. Data is transferred through internet without human to human or human to computer interaction.

- ii. In this project we have not used any hardware. Instead of real soil moisture, Humidity and Temperature data obtained from sensors we make use of IBM IoT Simulator which can transmit these parameters as required.

b. **Project requirements:** Node-RED, IBM Cloud, IBM Watson IoT, Node.js, IBM Device, IBM IoT Simulator, Python 3.7, Open Weather API platform.

c. **Project Deliverables:** Application for IoT based Smart Agriculture System

d. **Purpose:**

- i. An intelligent crop protection system helps the farmers in protecting the crop from the animals and birds which destroy the crop.
- ii. This system also helps farmers to monitor the soil moisture levels in the field and also the temperature and humidity values near the field.

## 1.1 SCOPE OF WORK

1. Create a device in IBM Cloud Account.

2. Install Node-RED and configure the nodes that we want to use in the project.
3. Create the openweathermap account and get the API key and the weatherconditions using API key in the Node-RED.
4. Create a web application for user interaction for observation and controlactions.

## **2. LITERATURESURVEY**

### **a. EXISTING PROBLEM**

- i. Agriculture is a field which forms the basis of our economy.Yet it faces alot of problemsin terms of availability of resources, Irrigation, increasing rate of Pesticides, Climatic disasters, Insects which ruin the crops and makes a huge lossthis sector.
- ii. In agriculture water isneeded for the crops for their growth. If the Soil getsdry it is necessary to supply water. But sometimeif the farmerdoesn't visit the field it isnot possibleto knowthe condition of soil.
- iii. Sometimes over supply of water or less supply of wateraffects the growthof crops.
- iv. Sometimes if the weather/temperature changessuddenly it is necessaryto takecertainactions.

### **b. PROPOSED SOLUTION**

1. Soil Moisture can be checked by using the sensors that can sense the soil conditionand send the moisture contentin the soil over the cloud serviceto the web application.
2. The supplyof water can be controlled from anywhere by controlling themotor

state (ON/OFF), using web application.

3. Surrounding temperature can also be sensed by the sensors and displayed on the application.
4. Real time weather conditions can also be known by using different

### 3 THEORITICAL ANALYSIS

#### a. Block Diagram

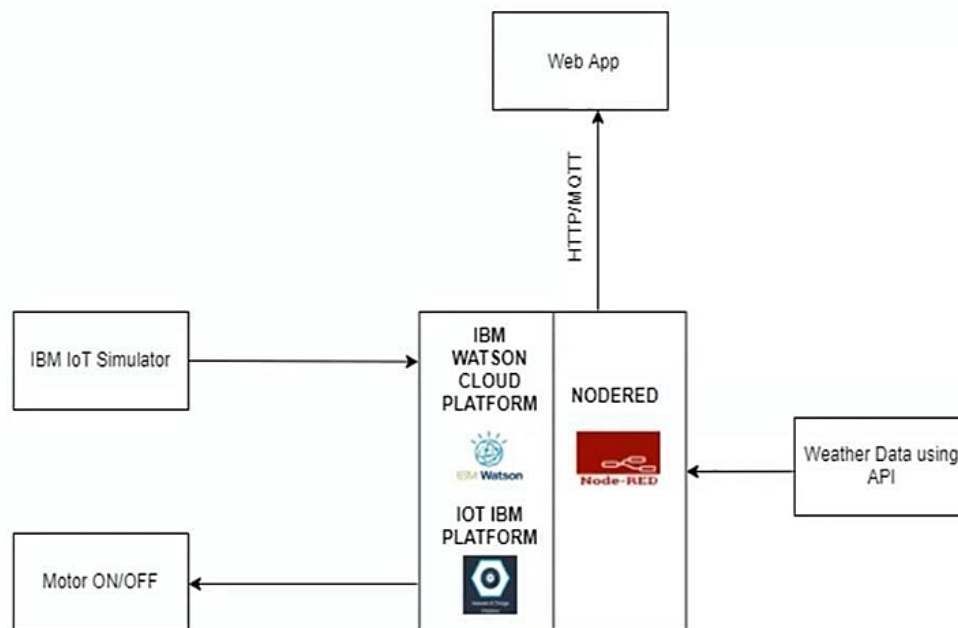


figure 3.1

#### b. Hardware / Software Designing

1. Create a device in IBM Cloud.
2. Connect the device to IBM Simulator to get the weather conditions.
3. Build Node-RED flow to build a web application to display the weather conditions and control the devices.
4. Get the real time weather condition data from open weather map

and integrate it in the Node-RED.

5. Control the working of the web application to the devices by python coding.

### c. Solution & Technical Architecture

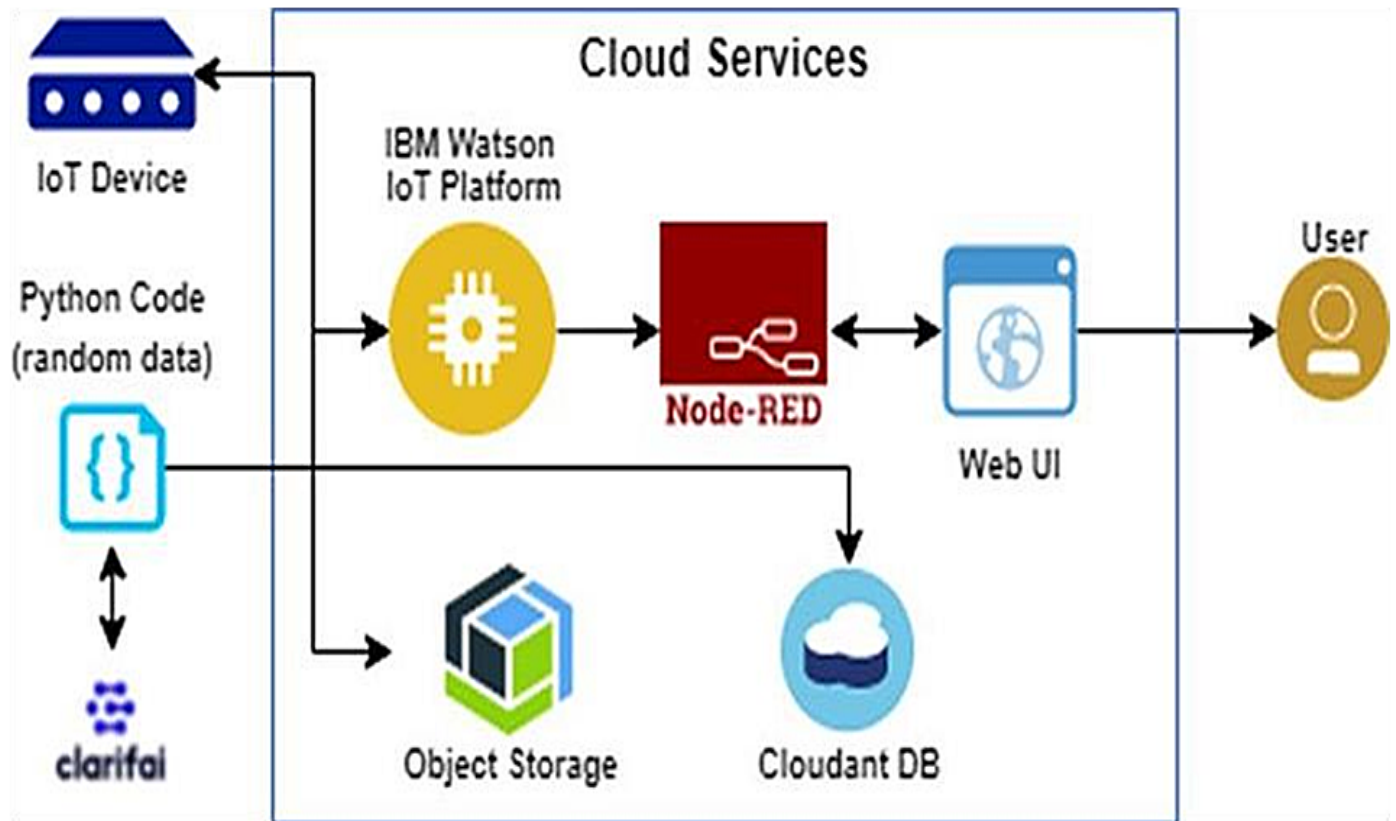


figure 3.2

### d. User stories

User Type	Functional requirement(Epic)	User Story number	User Story/Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	User can enter into the web application	I can access my account /dashboard	High	Sprint 1
		USN-2	User can register their credentials like email id and password	I can receive confirmation email & click confirm	High	Sprint 1
	Login	USN-3	User can log into the application by entering email & password	I can login to my account	High	Sprint 1
	Dashboard	USN-4	User can view the temperature	I can view the data given by the device	High	Sprint 2
		USN-5	User can view the level of sensor monitoring value	I can view the data given by the device	High	Sprint 2
Customer (Web user)	Usage	USN-1	User can view the web page and get the information	I can view the data given by the device	High	Sprint 3
Customer	Working	USN-1	User act according to the alert given by the device	I can get the data work according to it	High	Sprint 3

## 4 PROJECT PLANNING & SCHEDULING

### Sprint planning & estimation

TITLE	DESCRIPTION	DATE
Literature Survey on The Selected Project and Information Gathering	A Literature Survey is a compilation summary of research done previously in the given topic. Literature survey can be taken from books, research paper online or from any source.	20 September 2022
Prepare Empathy Map	Empathy Map is a visualization tool which can be used to get a better insight of the customer	22 September 2022
Ideation-Brainstorming	Brainstorming is a group problem solving session where ideas are shared, discussed and organized among the team members.	28 September 2022
Define Problem Statement	A Problem Statement is a concise description of the problem or issues a project seeks to address. The problem statement identifies the current state, the desired future state and any gaps between the two.	20 September 2022
Problem Solution Fit	This helps us to understand the thoughts of the customer their likes, <u>behaviour</u> , emotions etc.	01 October 2022
Proposed Solution	Proposed solution shows the current solution and it helps is going towards the desired result until it is achieved.	18 October 2022
Solution Architecture	Solution Architecture is a very complex process <u>i.e</u> it has a lot of sub-processes and branches. It helps in understanding the components and features to complete our project.	18 October 2022
Customer Journey	It helps us to <u>analyse</u> from the perspective of a customer, who uses our project.	01 November 2022
Functional Requirement	Here functional and nonfunctional requirements are briefed. It has	01 November 2022
	specific features like usability, security, reliability, performance, availability and scalability.	
Data Flow Diagrams	Data Flow Diagram is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement.	03 November 2022
Technology Architecture	Technology Architecture is a <u>more well</u> defined version of solution architecture. It helps us analyze and understand various technologies that needs to be implemented in the project.	03 November 2022
Prepare Milestone & Activity List	It helps us to understand and evaluate our own progress and accuracy so far.	06 November 2022
Spring Delivery Plan	Sprint planning is an event in scrum that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved.	06 November 2022

## 5 Coding & solution

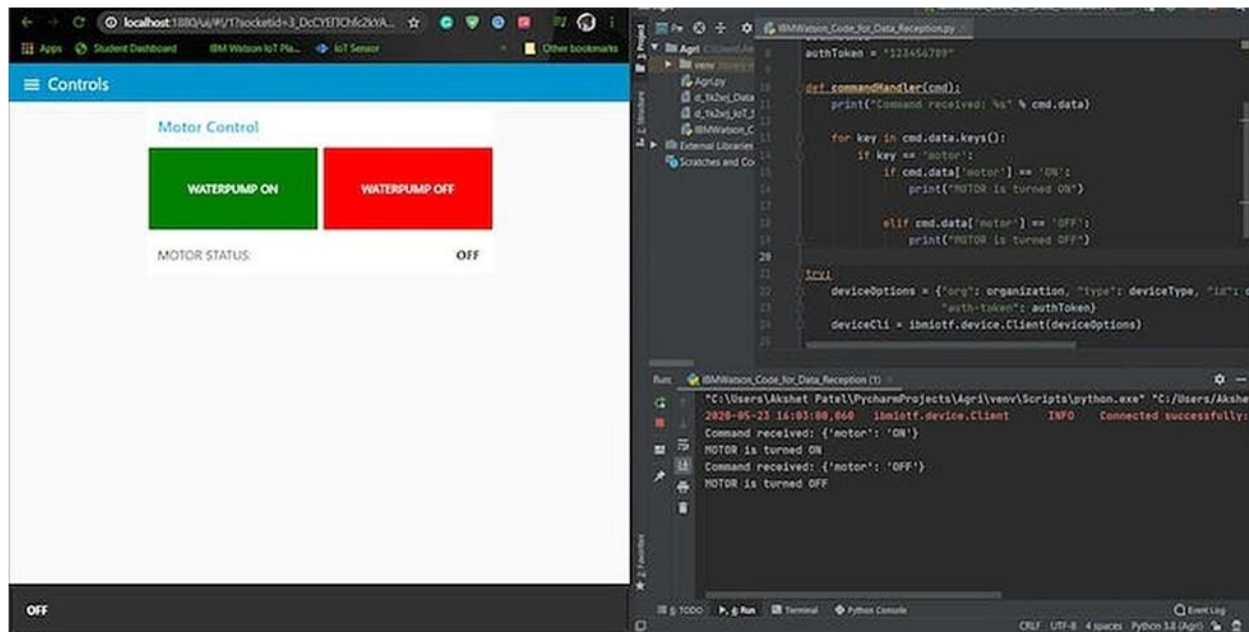


figure 5.1

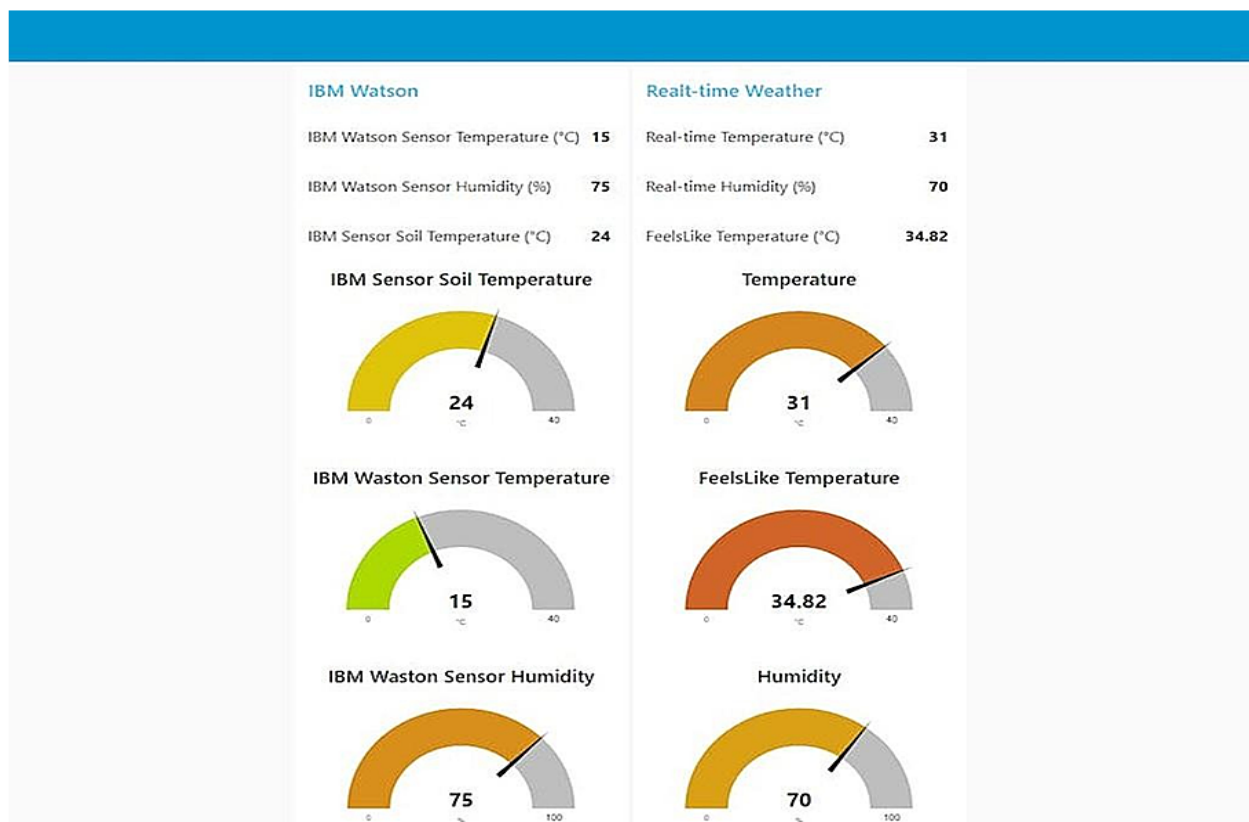


figure 5.2

## DEVICE CONTROLACTION

In this project we send the weather data through IoT Simulator shown



in fig(a) instead of real soil and temperature conditions. Simulator passes the data through IBM Cloud to the web application. The data is displayed on the Dashboard shown in fig (b1 & b2). Web Application is built using Node-RED. We have created 2 tabs:

1. IoT Smart Agriculture.
2. Graphical Representation.

Web Application is also used to control the devices further like motor, pumps, lights, or any other devices in the agricultural field. In this project the output is passed using python code and the control action is displayed in python

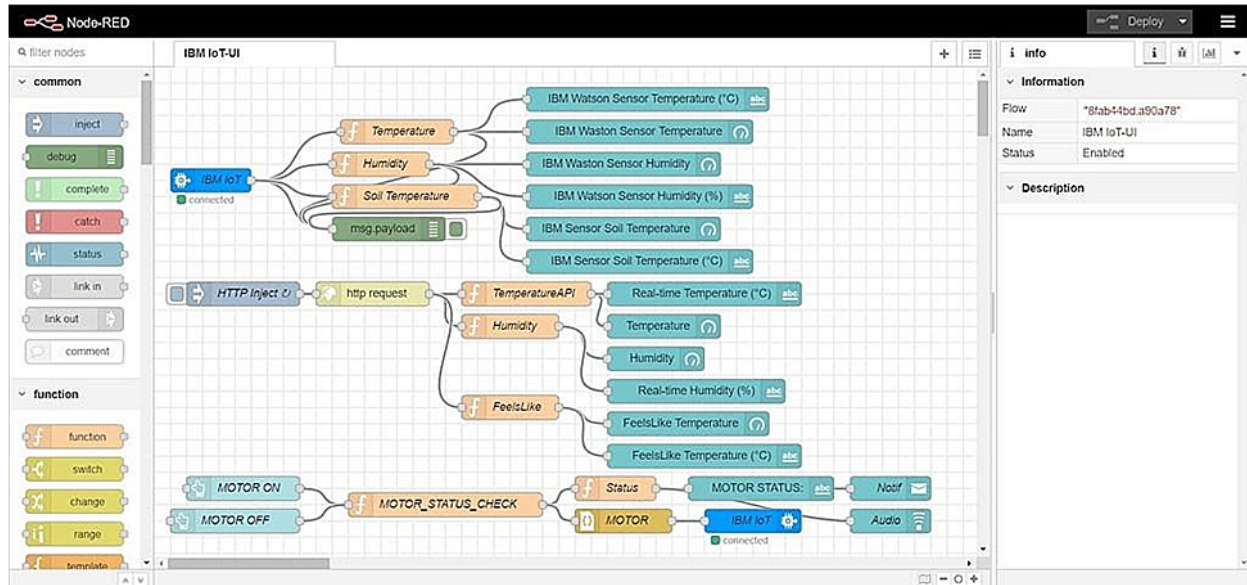
### **Following are the nodes used in the project in the Web Application:**

1. IBM IoT: IN and OUT Nodes.
2. function Nodes.
3. Gauge Nodes.
4. Chart Nodes.
5. Debug Node.
6. Button Nodes

### **Following are the nodes used for the weather condition from openweathermap:**

1. Timestamp Node.
2. http request Node
3. Function Nodes.
4. Text Nodes.

## **6 Node Red:**



## 7 Result:

We have successfully build a web based UI and integrated all the services using Node-RED.

Web Application : <https://node-red-aab.eu-gb.mybluemix.net/ui/>

## 8 ADVANTAGES & DISADVANTAGES:

### a. ADVANTAGES

3.All the data like climaticconditions and changesin them, soil or

crop conditions everything can be easilymonitored.

Risk of crop damage can be loweredto a greater extent.

Many difficultchallenges can be avoided makingthe process automatedand the quality of crops can be maintained.

The processincluded in farmingcan be controlled using the web

### b.DISADVANTAGES:

1. Smart Agriculture requires internet connectivity continuously, but rural parts cannot fulfill this requirement.
2. Any faults in the sensors can cause great loss in the agriculture, due to wrong records and the actions of automated processes.
3. IoT devices need much money to implement.

## **9 APPLICATIONS:**

1. Precision Farming that is farming processes can be made more controlled and accurate.
2. Live monitoring can be done of all the processes and the conditions on the agricultural field.
3. All the controls can be made just on the click.
4. Quality can be maintained.

## **10 CONCLUSION:**

IoT based smart Crop Monitoring System for Agriculture for Live Monitoring of Temperature and Soil Moisture and to control motor and light remotely has been proposed using Node Red and IBM Cloud Platform. The System has high efficiency and accuracy in fetching the live data of temperature and soil moisture. The IoT based smart farming System being proposed via this project will assist farmers in increasing the agriculture yield and take efficient care of food production as the System will always provide a helping hand to farmers for getting accurate live feed of environmental temperature and soil moisture with more than 99% accurate results. Therefore, the project proposes a thought of consolidating the most recent innovation into the agrarian field to turn the customary techniques for water system to current strategies in this way making simple profitable and temperature trimming.

## 11 FUTURE SCOPE:

In future due to more demand of good and more farming in less time, for betterment of the crops and reducing the usage of extravagant resources like electricity and water IoT can be implemented in most of the places.

## 12 APPENDIX:

**GithubLink:** <https://github.com/IBM-EPBL/IBM-Project-35934-1660290587>

### **source code**

```
import
time
import
sys
import ibmiotf.application # to installpip install ibmiotfimport
ibmiotf.device

# Provide your IBM Watson DeviceCredentials
organization = "8gyz7t" # replace the ORG ID
deviceType = "weather_monitor" # replace the Device type
deviceId = "b827ebd607b5" # replace Device ID authMethod =
"token"
authToken = "LWVpQPpVQ166HWN48f" # Replace the authtoken

def myCommandCallback(cmd): # function for Callbackif

    cmd.data['command'] == 'motoron':

    print("MOTOR ON IS RECEIVED")

elif cmd.data['command'] ==
'motoroff':print("MOTOR OFF IS
RECEIVED")
```

```

if cmd.command == "setInterval":if

    'interval' not in cmd.data:

    print("Error - command is missing required information: 'interval'")
    else:
        interval = cmd.data['interval']
    elif cmd.command == "print":
        if 'message' not in cmd.data:
            print("Error - command is missing required information: 'message'")

        else:
            output = cmd.data['message']
            print(output)

    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 cloudas an event oftype
    "greeting" 10 times
    deviceCli.connect()

    while True:
        deviceCli.commandCallback = myCommandCallback

    # Disconnect the device and application from the clouddeviceCli.disconnect()

```

## SENSOR.PY

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

# Provide your IBM Watson Device Credentials
organization = "8gyz7t" # replace the ORG ID
deviceType = "weather_monitor" # replace the Device type
deviceId = "b827ebd607b5" # replace Device ID authMethod =
"token"
authToken = "LWVpQPpVQ166HWN48f" # Replace the authtoken
```

```
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    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:
    temp=random.randint(0,100)
    pulse=random.randint(0,100)
    soil=random.randint(0,100)

    data = { 'temp': temp, 'pulse': pulse , 'soil':soil}

    #print data
    def myOnPublishCallback():
        print ("Published Temperature = %s C"% temp, "Humidity = %s %%"%pulse,"Soil
        Moisture = %s %%" % soil,"to IBM Watson")

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

    deviceCli.commandCallback = myCommandCallback

    # Disconnect the device and application from the clouddeviceCli.disconnect()

```

#### A. Node-RED FLOW :

```

[
{
  "id":"625574ead9839b
34",
  "type":"ibmiotout",
  "z":"630c8601c5ac3295",
  "authentication":"apiKey",
  "apiKey":"ef745d48e395ccc0",
  "outputType":"cmd",
  "deviceId":"b827ebd607b5",
  "deviceType":"weather_monitor",

```

```
"eventCommandType":"data",
"format":"json",
"data":"data",
"qos":0,
"name":"IBM IoT",
"service":"registere
d",
```

```
"x":680,
```

```
"y":220,
```

```
"wires":[]
```

```
},
```

```
{
```

```
"id":"4cff18c3274cccc4","type":"ui_button",
```

```
"z":"630c8601c5ac3295",
```

```
"name": "",
```

```
"group":"716e956.00eed6c",
```

```
"order":2,
```

```
"width":0,
```

```
"height":0,
```

```
"passthru":false,
```

```
"label":"MotorO
```

```
N",
```

```
"tooltip": "",
```

```
"color": "",
```

```
"bgcolor": "",
```

```
"className": "",
```

```
"icon": "", "payload": {"command": "motoron"},
```

```
"payloadType": "str",
```

```
"topic": "motoron",
```

```
"topicType": "str",
```

```
"x":360,
```

```
"y":160,
```

```
"wires": [{"625574ead9839b34"}]},
```

```
{
```

```
"id":"659589baceb4e0b
```

```
0","type":"ui_button",
```



```
"z":"630c8601c5ac329
5",
"name":"",
"group":"716e956.00eed6c",
"order":3,

"width":"0",
"height":"0",
"passthru":true,
"label":"MotorOFF",
"tooltip":"",
"color":"",
"bgcolor":"",
"className":"",
"icon":"","payload":{"command\\":"motoroff\\"},
"payloadType":"str",
"topic":"motoroff",
"topicType":"s
tr","x":350,
"y":220,
"wires":[["625574ead9839b34"]]},
{"id":"ef745d48e395ccc0
","type":"ibmiot",
"name":"weather_monitor
","keepalive":"60",
"serverName":"
",
"cleansession":t
rue,"appId":"",
"shared":false},
{"id":"716e956.00eed6c",
"type":"ui_group",
"name":"Form",
"tab":"7e62365e.b7e6b
8","order":1,
"disp":true,
"width":"6",
"collapse":fal
```

```
se},
{"id":"7e62365e.b7e6b8",
"type":"ui_tab",
```

```
"name":"contorl",
"icon":"dashboa
rd","order":1,
"disabled":false,
"hidden":false}
]
```

```
[
{
"id":"b42b5519fee73ee2",
"type":"ibmiotin",
"z":"03acb6ae05a0c712",
"authentication":"apiKey",
"apiKey":"ef745d48e395ccc0",
"inputType":"evt",
"logicalInterface":"","ruleId":"",
"deviceId":"b827ebd607b5",
"applicationId":"",
"deviceType":"weather_monitor", "eventType":"+",
"commandType":"",
"format":"json",
"name":"IBMIoT",
"service":"registered",
"allDevices":"",
"allApplications":"",
"allDeviceTypes":"",
"allLogicalInterfaces":"",
"allEvents":true,
"allCommands":"",
"allFormats
":"", "qos":0,

"x":270,
"y":180,
```

```

"wires":[["50b13e02170d73fc","d7da6c2f5302ffaf","a949797028158f3f","a71f164bc378bcf1"]]
},
{
  "id":"50b13e02170d73fc",
  "type":"function",
  "z":"03acb6ae05a0c712",
  "name":"Soil Moisture",
  "func":"msg.payload = msg.payload.soil;\nglobal.set('s',msg.payload);\nreturn msg;",
  "outputs":1,
  "noerr":0,
  "initialize": "",
  "finalize": "",
  "lib": "s":[
  ],
  "x": 490,
  "y": 120,
  "wires":[["a949797028158f3f","ba98e701f55f04fe"]]
},
{
  "id":"d7da6c2f5302ffaf",
  "type":"function",
  "z":"03acb6ae05a0c712",
  "name":"Humidity",
  "func":"msg.payload = msg.payload.pulse;\nglobal.set('p',msg.payload)\nreturn msg;",
  "outputs":1,
  "noerr":0,
  "initialize": "",
  "finalize": "",
  "lib": "s":[
  ],
  "x": 490

```

```
80,  
"y":260,"wires":[["a949797028158f3f","70a5b076eeb80b70"]]
```

```
},
```

```
{
```

```
"id":"a949797028158f
```

```
3f",
```

```
"type":"debug",
```

```
"z":"03acb6ae05a0c71
```

```
2","name":"IBMo/p",
```

```
"active":true,
```

```
"tosidebar":true,
```

```
"console":false,
```

```
"tostatus":false,
```

```
"complete":"payload",
```

```
"targetType":"msg",
```

```
"statusVal": "",
```

```
"statusType":"auto",
```

```
"x":780,
```

```
"y":180,
```

```
"wires":[]
```

```
},
```

```
{
```

```
"id":"70a5b076eeb80b70",
```

```
"type":"ui_gauge",
```

```
"z":"03acb6ae05a0c712",
```

```
"name": "",
```

```
"group":"f4cb8513b95c98a4",
```

```
"order":6,
```

```
"width":"0",
```

```
"height":"0",
```

```
"gtype":"gage",
```

```
"title":"Humidity",
```

```
"label":"Percentage(%)",
```

```
"format":"{{value}}
```

```
", "min":0,
```

```
"max":"100",
```

```
"colors":["#00b500","#e6e600","#ca3838"],
```

```

"seg1":"","
"seg2":"","
"classNam
e":"","x":860,
"y":260,
"wires":[]
},
{
"id":"a71f164bc378bcf
1","type":"function",
"z":"03acb6ae05a0c71
2",
"name":"Temperature",
"func":"msg.payload=msg.payload.temp;\nglobal.set('t',msg.payload);\nreturn msg;","outputs":1,
"noerr":0,
"initialize
":"","
"finalize":"","
"lib
s":[
],
"x"
:4
90,
"y":360,
"wires":[["8e8b63b110c5ec2d","a949797028158f3f"]]
},
{
"id":"8e8b63b110c5ec2d",
"type":"ui_gauge",
"z":"03acb6ae05a0c712",
"name":"",
"group":"f4cb8513b95c98a4",
"order":11,
"width":"0",
"height":"0",
"ctype":"gage",

```

```
"title": "Temperature",  
"label": "DegreeCelcius",  
s",
```

```
"format": "{{value}}",  
", "min": 0,  
"max": "100",  
"colors": ["#00b500", "#e6e600", "#ca3838"],  
"seg1": "",  
"seg2": "",  
"className":  
e": "", "x": 790,  
"y": 360,  
"wires": []  
},  
{
```

```
"id": "ba98e701f55f04fe",  
"type": "ui_gauge",  
"z": "03acb6ae05a0c712",  
"name": "",  
"group": "f4cb8513b95c98a4",  
"order": 1,  
"width": "0",  
"height": "0",  
"gtype": "gage",  
"title": "Soil Moisture",  
"label": "Percentage(%  
)",  
"format": "{{value}}",  
", "min": 0,  
"max": "100",  
"colors": ["#00b500", "#e6e600", "#ca3838"],  
"seg1": "",  
"seg2": "",  
"className":  
e": "", "x": 790,  
"y": 120,  
"wires": []  
},
```

```

{
  "id":"a259673baf5f0f9
8", "type":"httpin",
  "z":"03acb6ae05a0c71
2", "name": "",
  "url":"/sensor",
  "method":"get
",
  "upload":fals
e,
  "swaggerDoc
": "", "x":370,
  "y":500,
  "wires":[["18a8cdbf7943d27a"]]
},
{
  "id":"18a8cdbf7943d27a",
  "type":"function",
  "z":"03acb6ae05a0c712",
  "name":"httpfunction",
  "func":"msg.payload{\"pulse\":global.get('p'),\"temp\":global.get('t'),\"soil\":global.get('s')};\nreturn msg;",
  "outputs":1,
  "noerr":0,
  "initialize
": "",
  "finalize": "",
  "lib
s":[
],
  "x"
:6
30,
  "y":500,
  "wires":[["5c7996d53a445412"]]
},
{

```

```
"id": "5c7996d53a445412",
"type": "httpresponse",
"z": "03acb6ae05a0c712", "name": "",

"statusCode": "",
"header
s": {},
"x": 870,
"y": 500,
"wires": []
},
{
  "id": "ef745d48e395ccc0",
  "type": "ibmiot",
  "name": "weather_monitor",
  "keepalive": "60",
  "serverName": "",
  "cleansession": true,
  "appId": "",
  "shared": false,
  {
    "id": "f4cb8513b95c98a4", "type": "ui_group",
    "name": "monitor",
    "tab": "1f4cb829.2fdee8",
    "order": 2,
    "disp": true,
    "width": "6",
    "collapse": false,
    "className": ""
  },
  {
    "id": "1f4cb829.2fdee8",
    "type": "ui_tab",
```



```
"name":"Home",  
"icon":"dashboard","order":3, "disabled":false, "hidden":false }
```