

SMART
AGRICULTURE
SYSTEM BASED ON
IOT

~ *D . NIKITHA*

CONTENTS

1. Introduction
 - 1.1 Overview
 - 1.2 Purpose
2. Literature Survey
 - 2.1 Existing problem
 - 2.2 Proposed solution
3. Theoretical Analysis
 - 3.1 Block diagram
 - 3.2 Required software installation
 - i.) Node-Red
 - ii.) IBM Watson IoT Platform
 - iii.) Python IDLE(3.7 or more)
 - 3.3 IoT Simulator
 - 3.4 Open weather API
4. Experimental Investigations
 - 4.1 Connecting IoT Simulator to IBM Watson IoT Platform

4.2 Configuration of Node-Red to collect IBM cloud data .
4.3 Configuration of Node-Red to collect data from
OpenWeather.
4.4 Configuration of Node-Red to send commands to IBM
cloud
4.5 Building User Interface
4.6 Web App UI
 i.) Smart Home Tab
 ii.) Motor Controls Tab
4.7 Receiving commands from IBM cloud using Python
program

5. Flow chart
6. Result
7. Advantages & Disadvantages
8. Applications
9. Conclusion
10. Bibliography

PROJECT REPORT

1. Introduction

1.1 Overview :

- ❖ This is all about creating a application for the farmers. To save their time to go and see all weather conditions are perfect to yield the crops, unless farmer can't yield. For the farmers we are creating a flexible application based on IOT. Smart Agriculture System based on IoT can monitor soil moisture and climatic conditions to grow and yield a good crop.

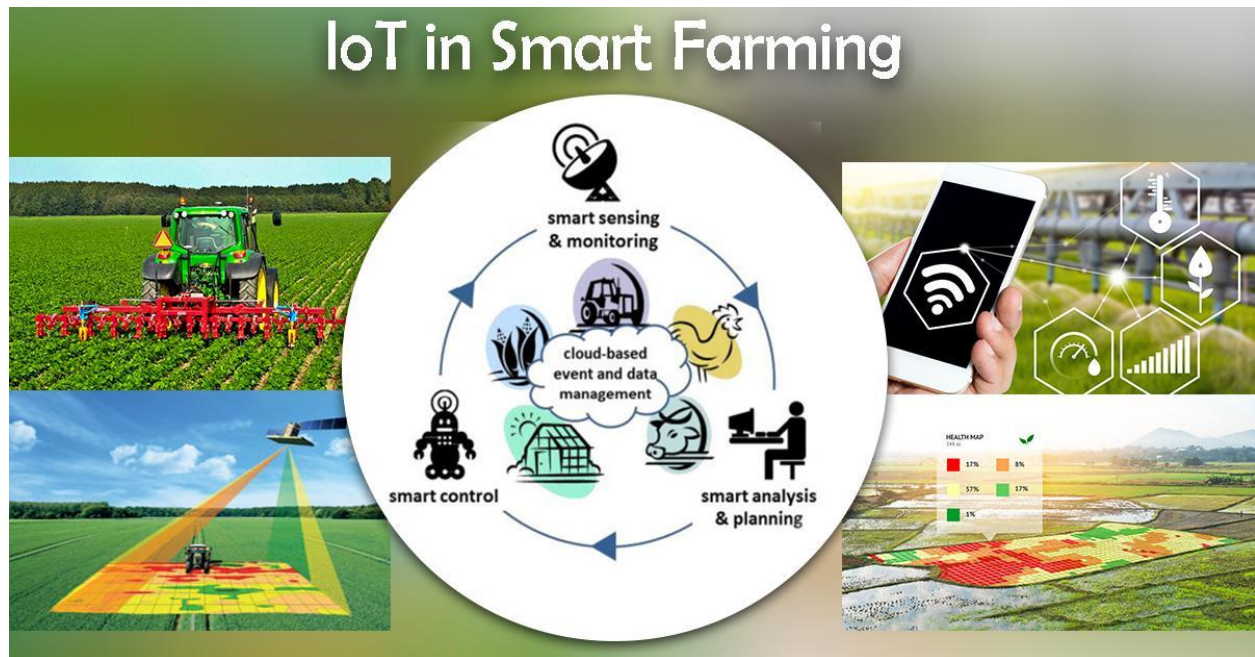


er

- ❖ The farmer can also get the realtime weather forecasting data by using external platforms like Open Weather API. By using IBM cloud platform we will perform all the required activities. This platform will provide us Simulators that are used to enable the operator or user to represent under test phenomena likely to occur in actual performance.
- ❖ From the IBM cloud platform we will connect the IOT simulator to the IBM watson platform. This enables us to create a new device and we can develop this by using Node-Red framework based on javascript. Now we will build a web application such that it shows all neccessary functions which a farmer needed.

1.2 Purpose :

- ❖ The main purpose of this smart agriculture application is to enhance the solutions of a farmer in which they are facing problems to yield a better crop. Farmer will provided with a mobile application using which he can monitor the temperature, humidity and soil moisture parameters along with weather forecasting.



- ❖ Based on all the parameters he can water his crop by controlling the motors using the mobile application. Even if the farmer is not present near his crop he can water his crop by controlling the motors using the mobile application from anywhere.
- ❖ Here we are using the Online IoT simulator for getting the Temperature, Humidity and Soil Moisture values.
- ❖ Farmer need not go and check all the conditions for yeilding , So by this application he can monitor all the things he needed.
- ❖ The sample picture is shown below are the ways that this application is useful for the farmer.

2. Literature Survey

2.1 Existing problem :

- ❖ Farmers are to be present at farm for their 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.

- ❖ To have a control on their work virtually by monitoring weather. For this they need minute to minute updates on weather conditions. This will save the time to go farm daily to check weather conditions.

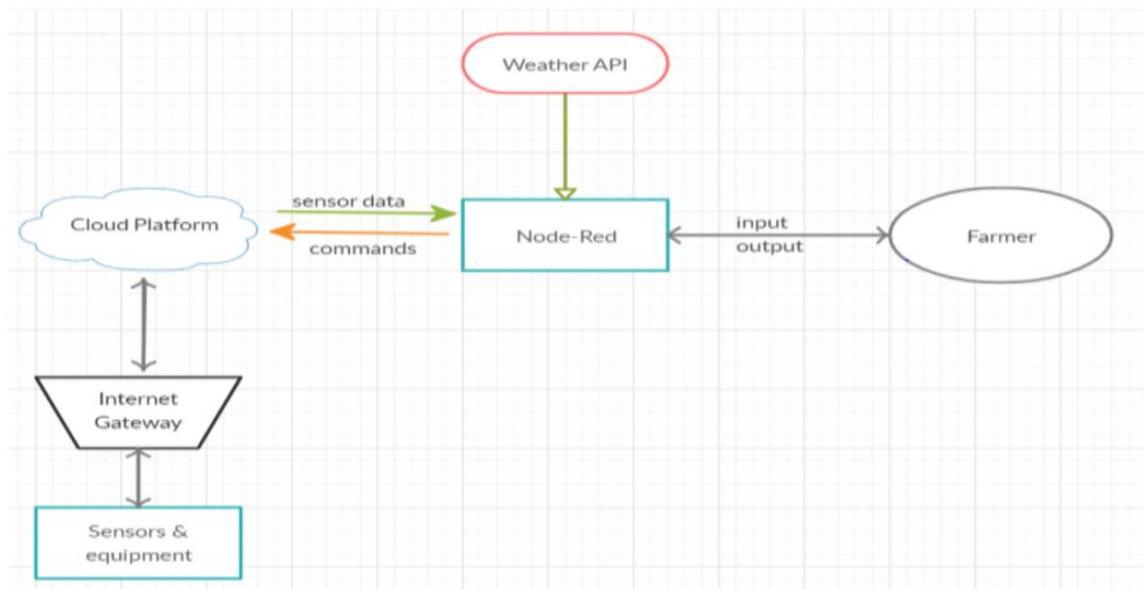
2.2 Proposed Solution :

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

3.Theoretical Analysis

3.1 Block Diagram :

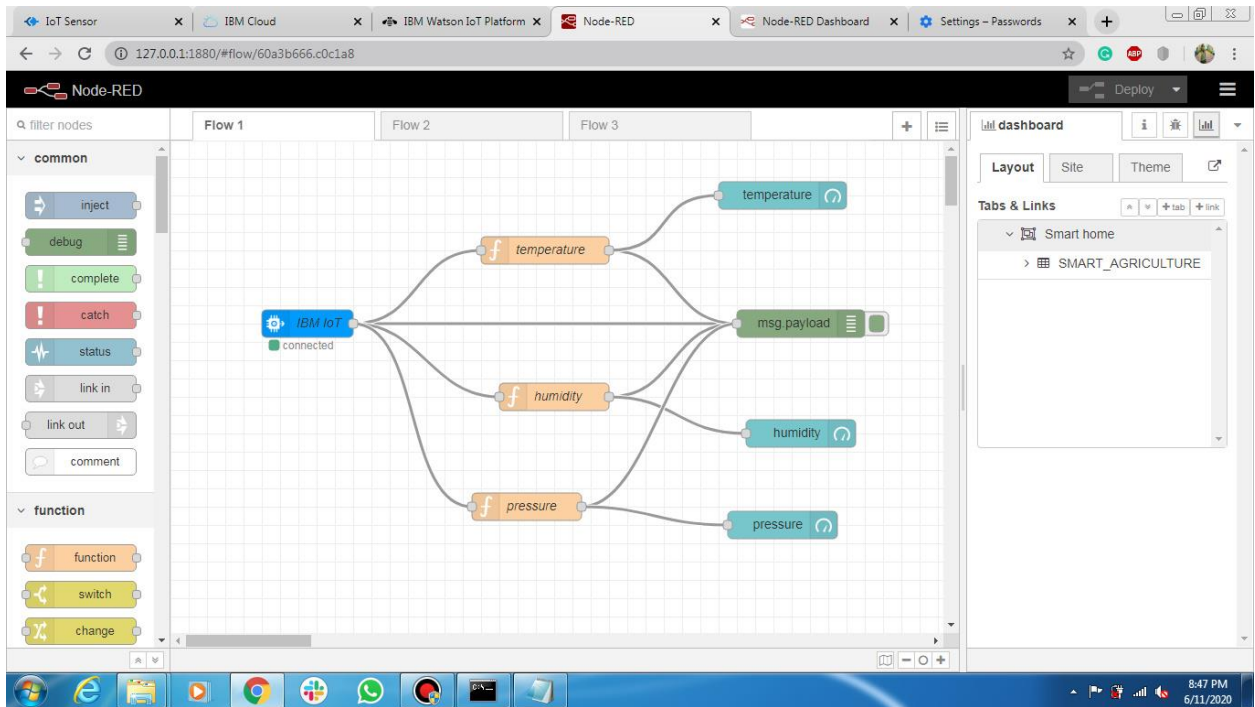
- ❖ In order to implement the solution to the farmer , the following approach as shown in the block diagram is used .



3.2 Required software installation :

i.) 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 :

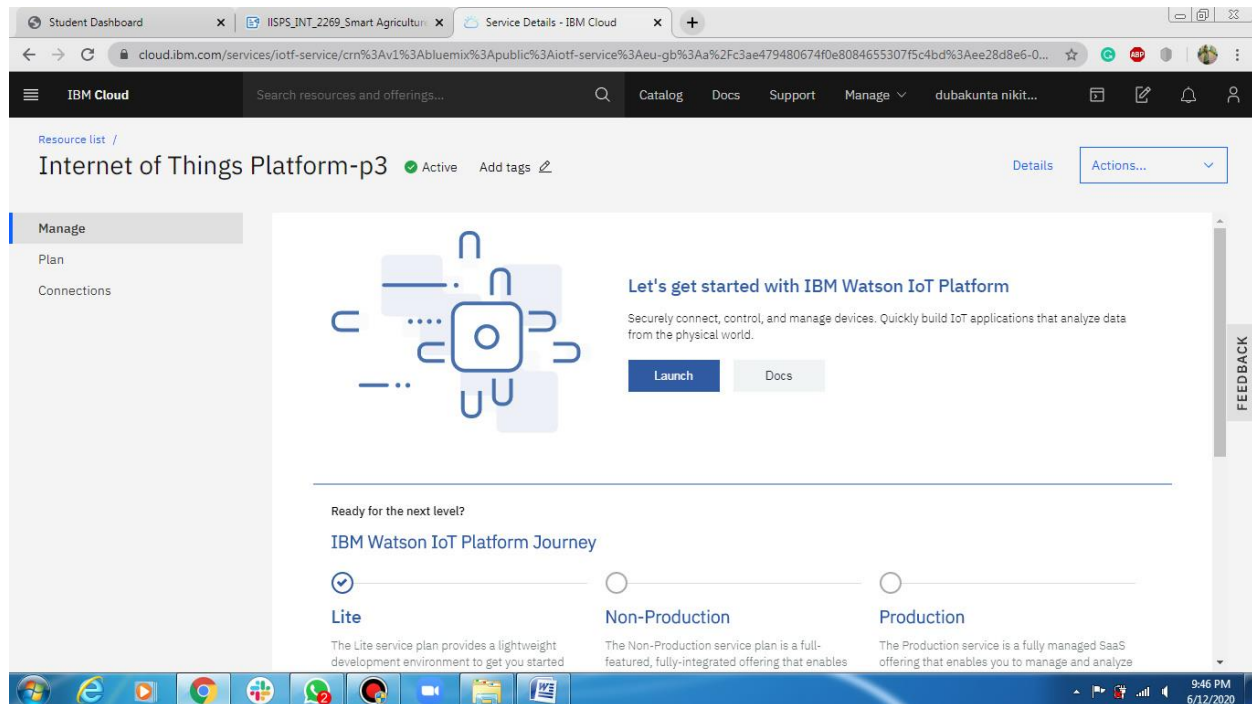
- ❖ First install node.js(npm will be installed by default when node.js is installed).
- ❖ Open cmd prompt .
- ❖ Type => npm install node-red
- ❖ To run the application : Open cmd prompt and type => Node-red.
- ❖ By that we will see one http link " <http://localhost:1880/> "
- ❖ Then open above link in desired 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

ii.) IBM Watson IoT Platform

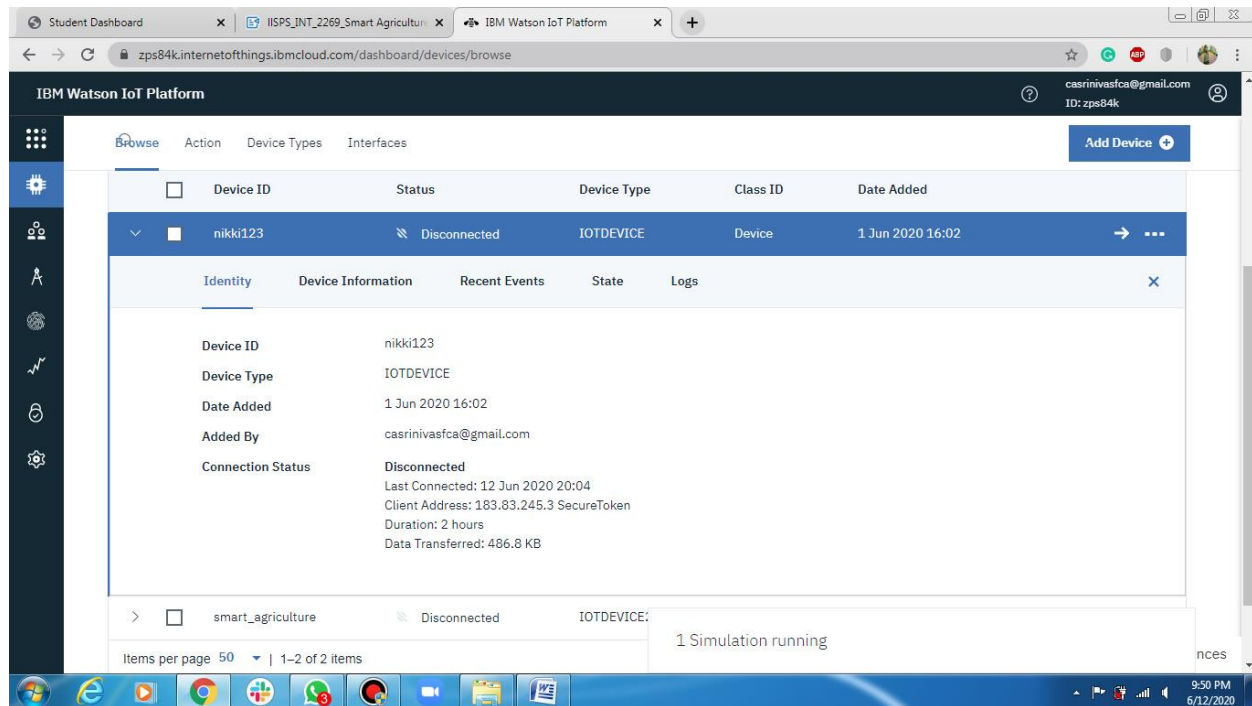
- ❖ A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization and data storage. IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices.
- ❖ We should go to this through ibm cloud by launching the service we created as shown below.



Steps to configure :

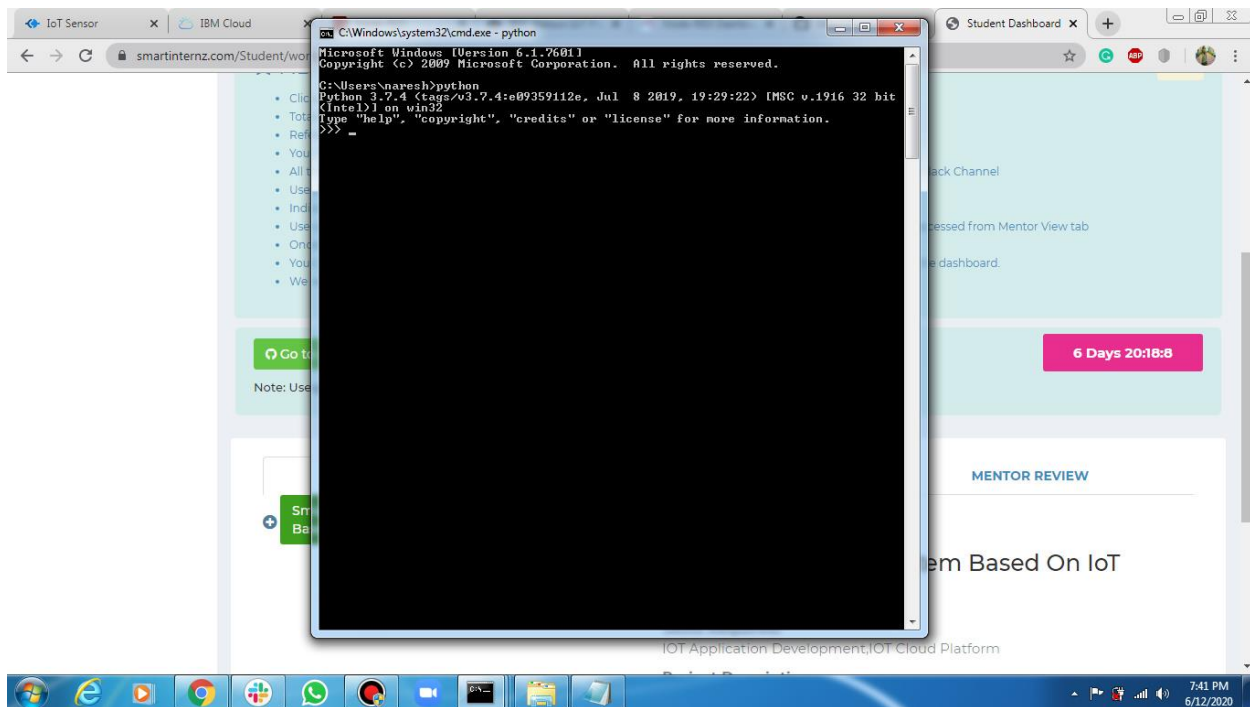
- ❖ Create an account in IBM cloud using your email ID.
- ❖ Create IBM Watson Platform in services in your IBM cloud account.
- ❖ Launch the IBM Watson IoT Platform.
- ❖ Create a new device
- ❖ Give credentials like device type, device ID, Authentication Token.
- ❖ Create API key and store API key and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

❖ *Below is the devices we should create .*



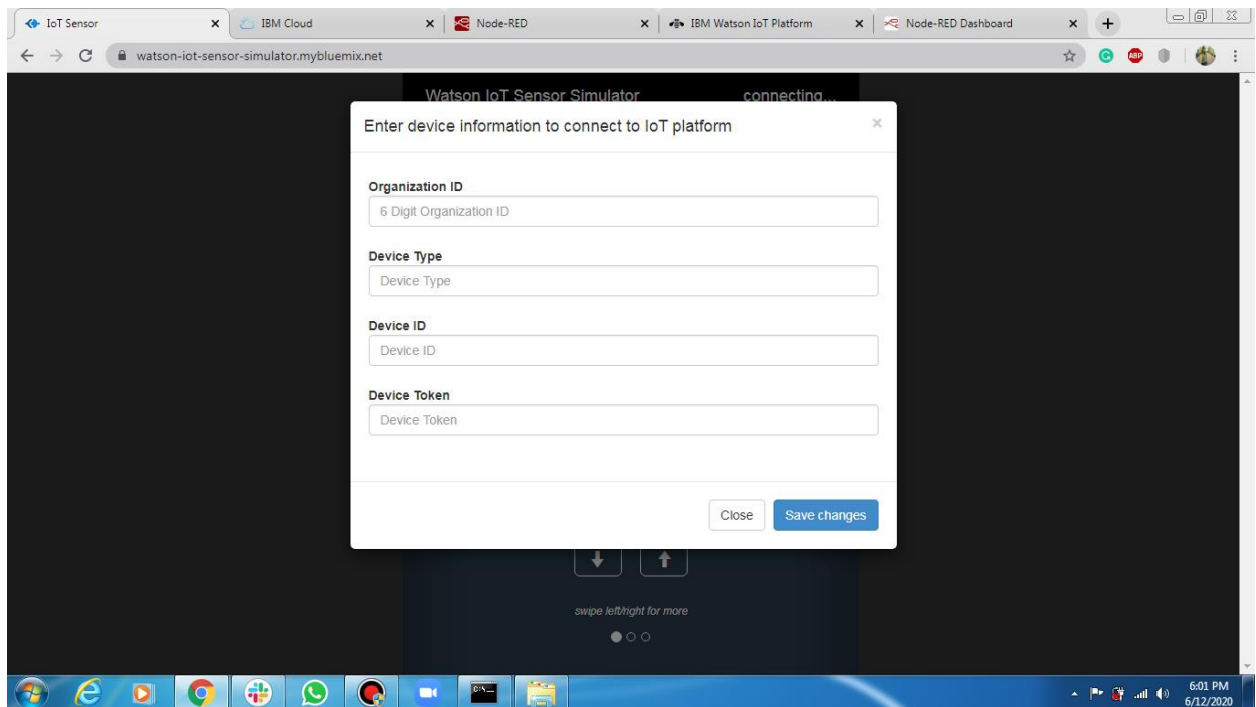
iii.) Python IDE

- ❖ Install Python3 IDE compiler
- ❖ Install any python IDE to execute python scripts that will receive motor commands.



3.3 Iot Simulator

- ❖ In our project in the place of sensors we are going to use IoT sensor simulator which will give random readings to the connected cloud.
- ❖ The link to simulator is given below
- ❖ <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.



3.4 Open weather 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 is : <https://home.openweathermap.org/>

Steps to configure :

1. Create account in OpenWeather
2. Find the name of your city by searching
3. Create API key to your account
4. 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}

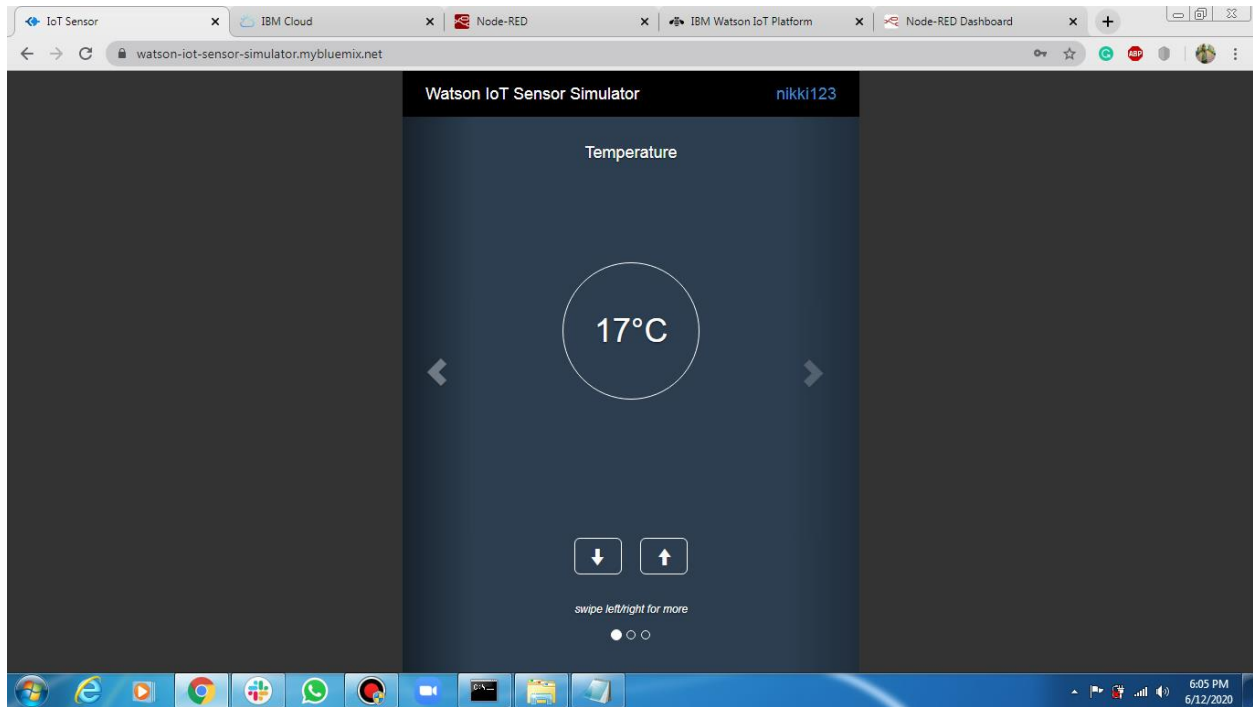
In the above link I have replaced by cityname and APi key. My link which i was used given below

<http://api.openweathermap.org/data/2.5/weather?q=Hyderabad,IN&appid=10499d92d087490bfd3b8c096db6f2bf>

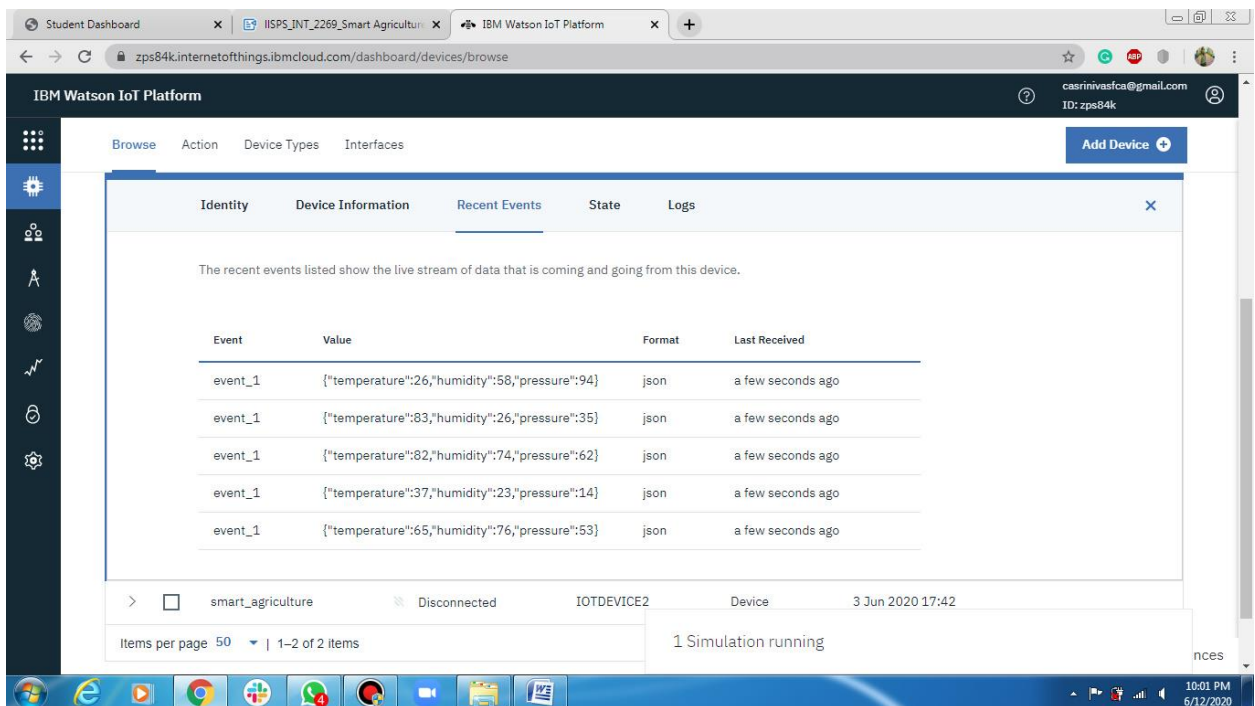
4. Experimental Investigations

4.1 Connecting IoT Suimulator To IBM Watson IoT Platform

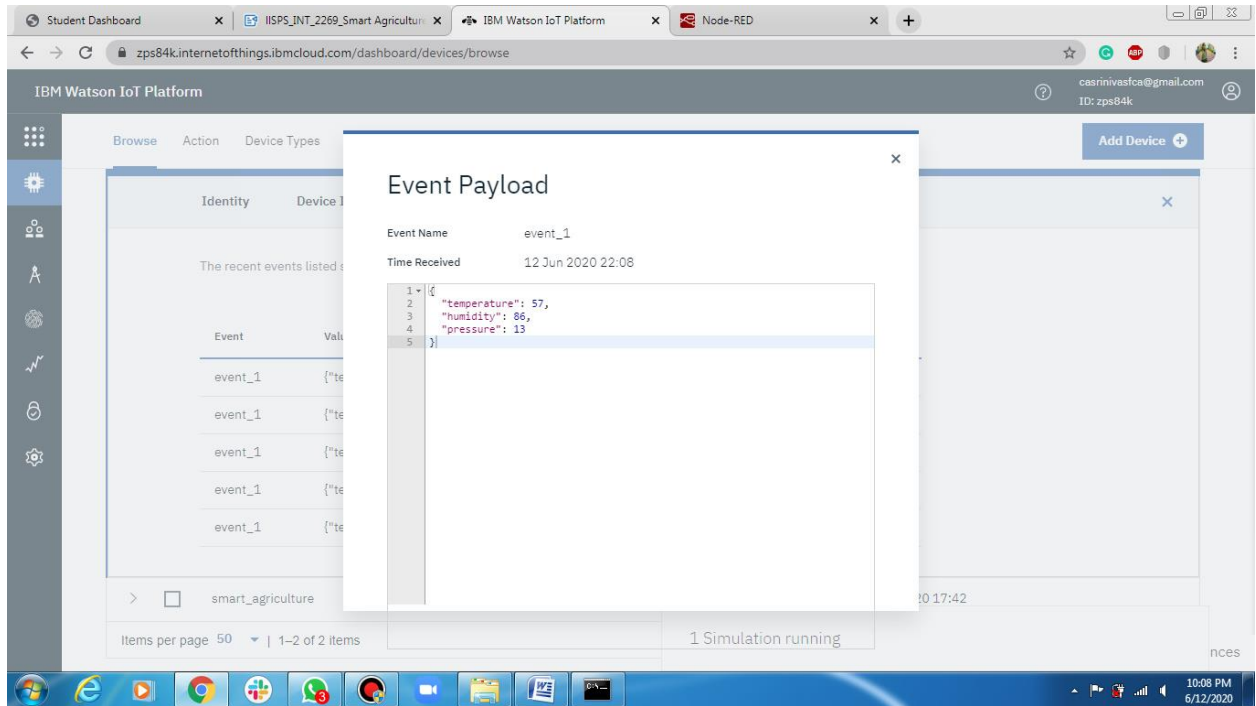
- ❖ We have to connect the lot Simulator in such way that by giving essential credentials.
- ❖ The link is provided in the section itself, so by opening that we will displayed one prompt to give us credentials in order to connect it.
- ❖ My first device credentials are given below
 - **Organization ID : zps84k**
 - **Device Type : IOTDEVICE**
 - **Device ID : nikki123**
 - **Authentication Method : use-token-auth**
 - **Authentication Token : 123456789**



❖ So by that we see some json format data in the recent events of device . This is shown in IBM Watson lot platform . Below is the recent events picture where we are getting random readings .

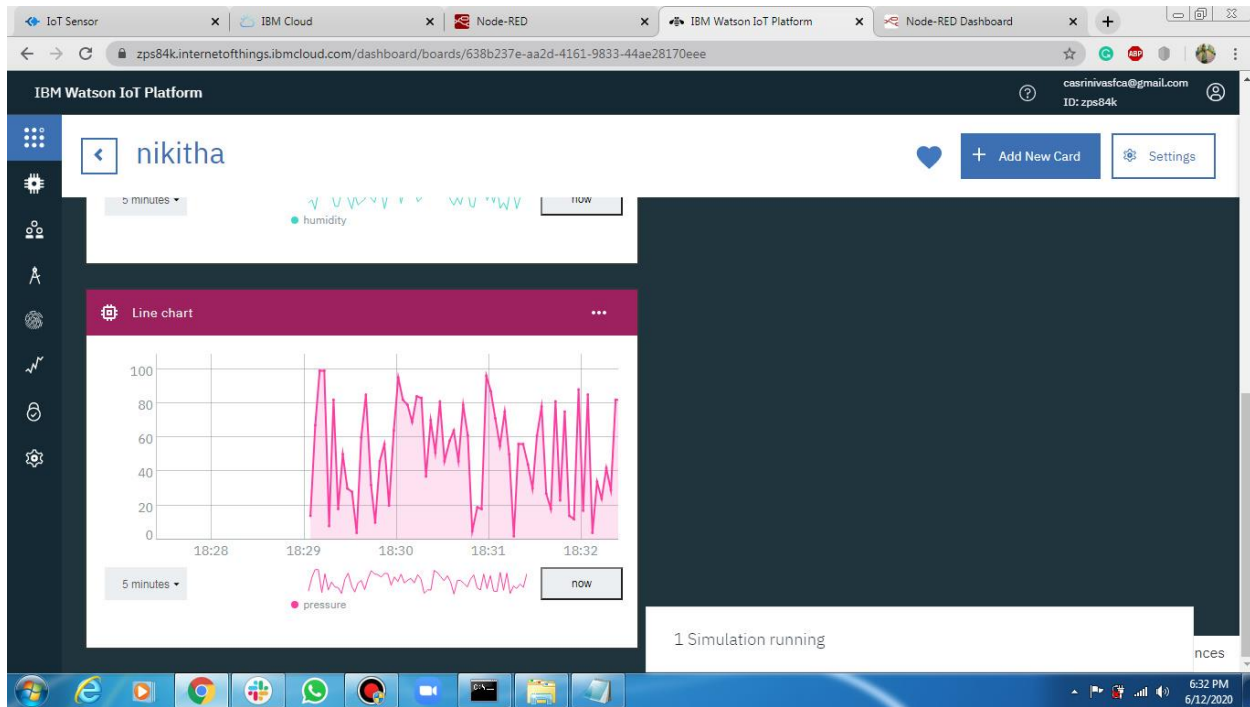


❖ Below picture is showing the format of json data.



❖ We can also see the data in the form of line chart in IBM boards.

❖ Below is the picture of one sample line chart graph where we can see the fluctuation.



- ❖ In the IBM Watson IoT platform I have created two devices , One device is for IoT Simulator and another second device is for ibm out node ,also for python code.
- ❖ Second device should be given in only ibm out node and python code. Rest all other should be used first device .
- ❖ My second device credentials are

Organization ID : zps84k

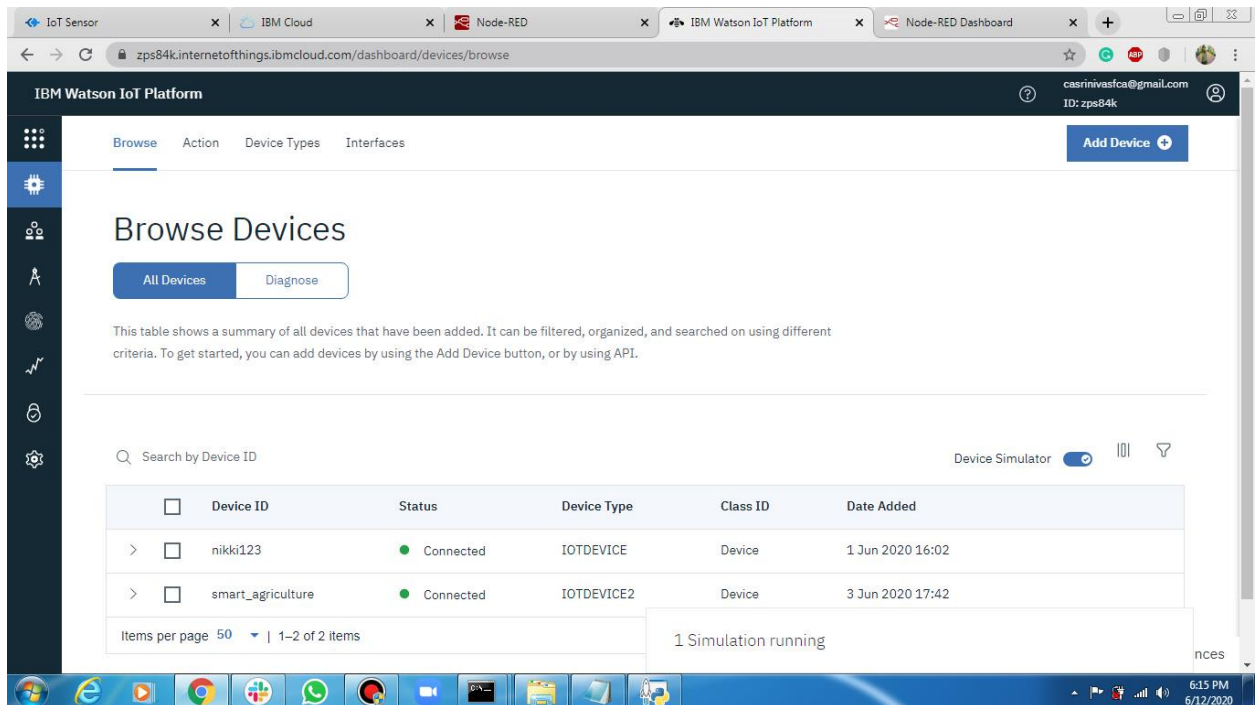
Device Type : IOTDEVICE2

Device ID : smart_agriculture

Authentication Method : use-token-auth

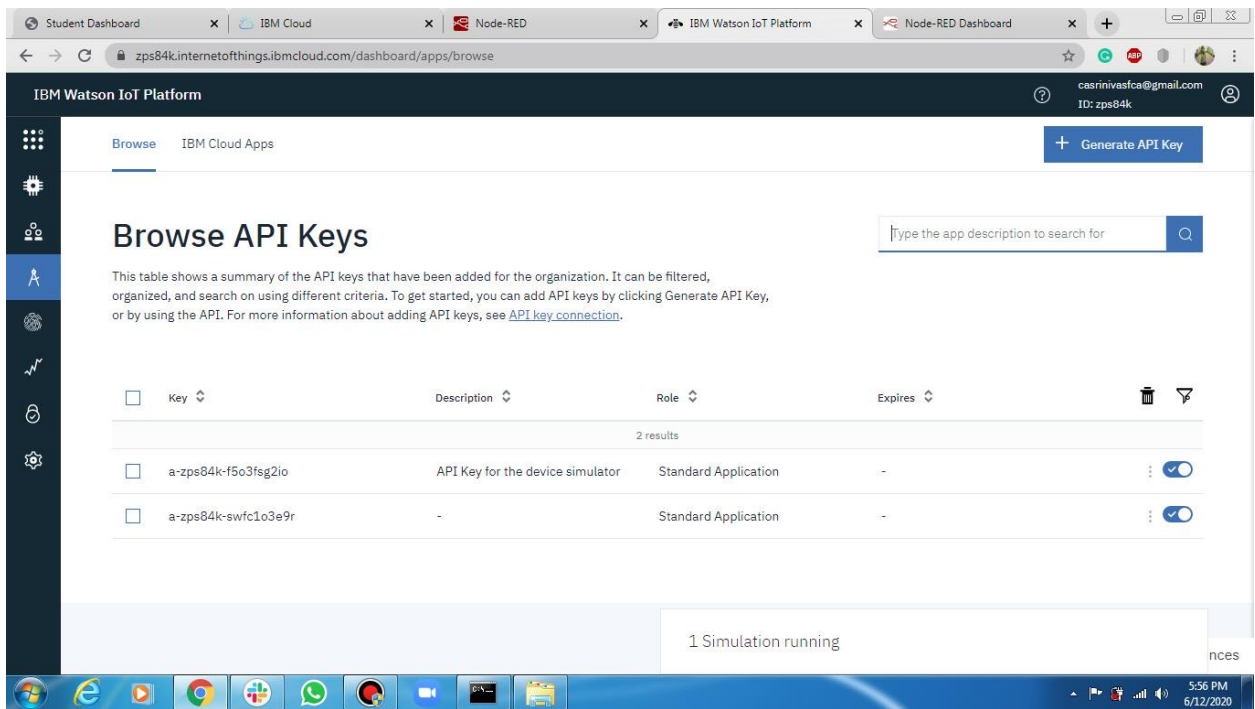
Authentication Token : 987654321

- ❖ Below is the image of devices I have created in IBM watson platform.

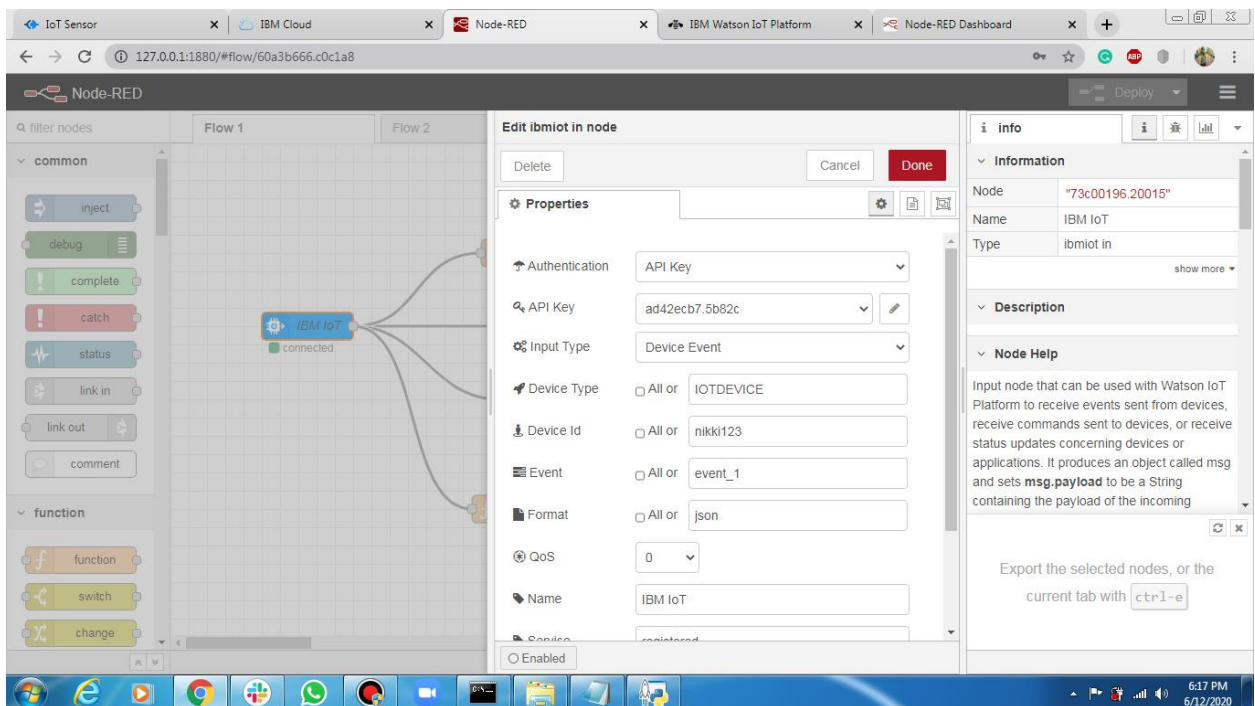


4.2 Configuration of Node-Red to collect IBM cloud data

- ❖ 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.
- ❖ In this we will have to generate one API key to configure. We can generate this key in Apps management section of IBM Watson IOT platform
- ❖ My API key and authentication token which I have generated is given below :
 - **API Key : a-zps84k-swfc1o3e9r**
 - Authentication token : 0b!uD@a*6297ITaLW0**
- ❖ Below is the picture of my API key .

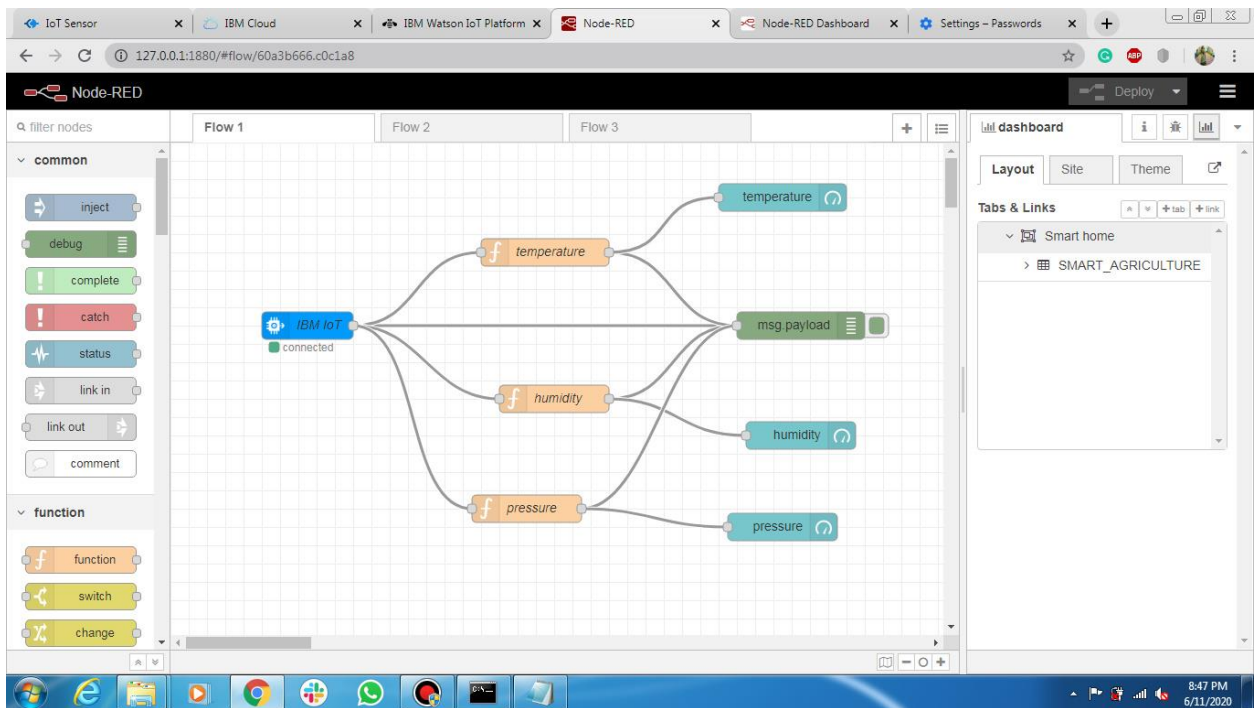


❖ So , give all the essential details and connect it to node-red.

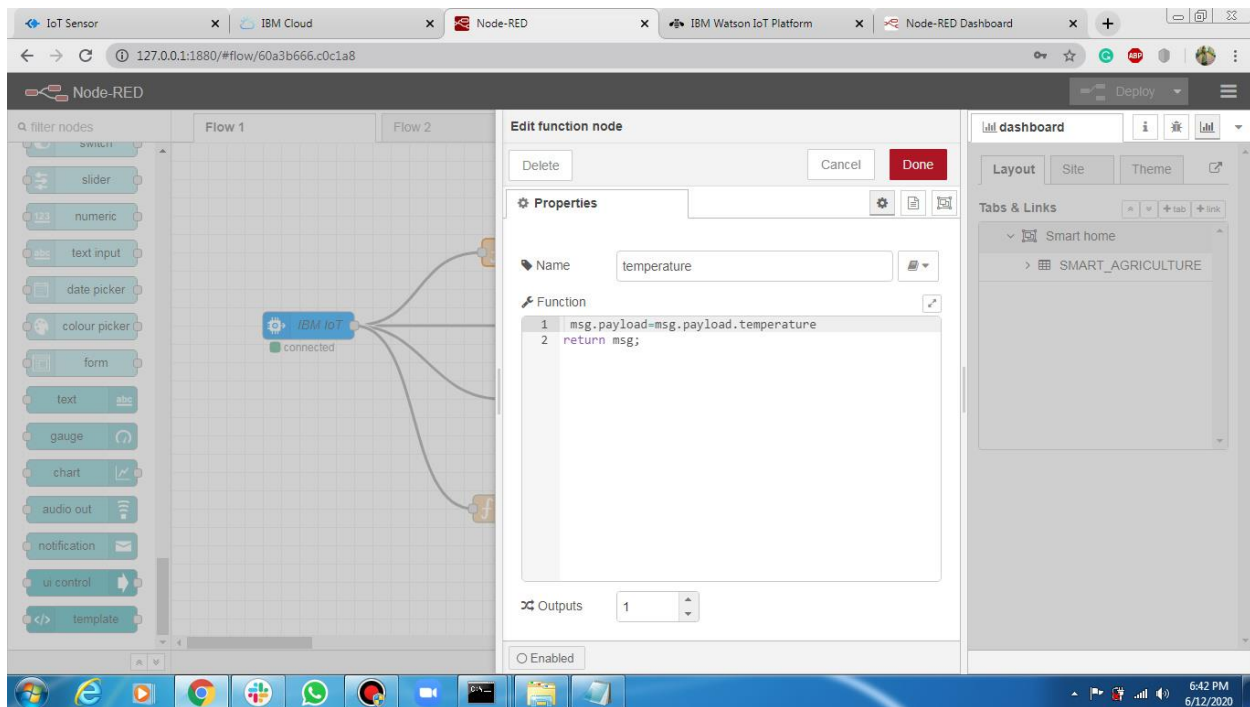


❖ Once it is connected Node-Red receives data from the device.

- ❖ 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.temperature
return msg;
- ❖ we will be able to see the debug messages on the debug tab as shown below.
- ❖ Finally connect Gauge nodes from dashboard to see the data in UI.



- ❖ Below is the javascript code I have given in function node of temperature, similarly remaining function nodes to be filled with the same kind of code.



4.3 Configuration of Node-Red to collect data from OpenWeather

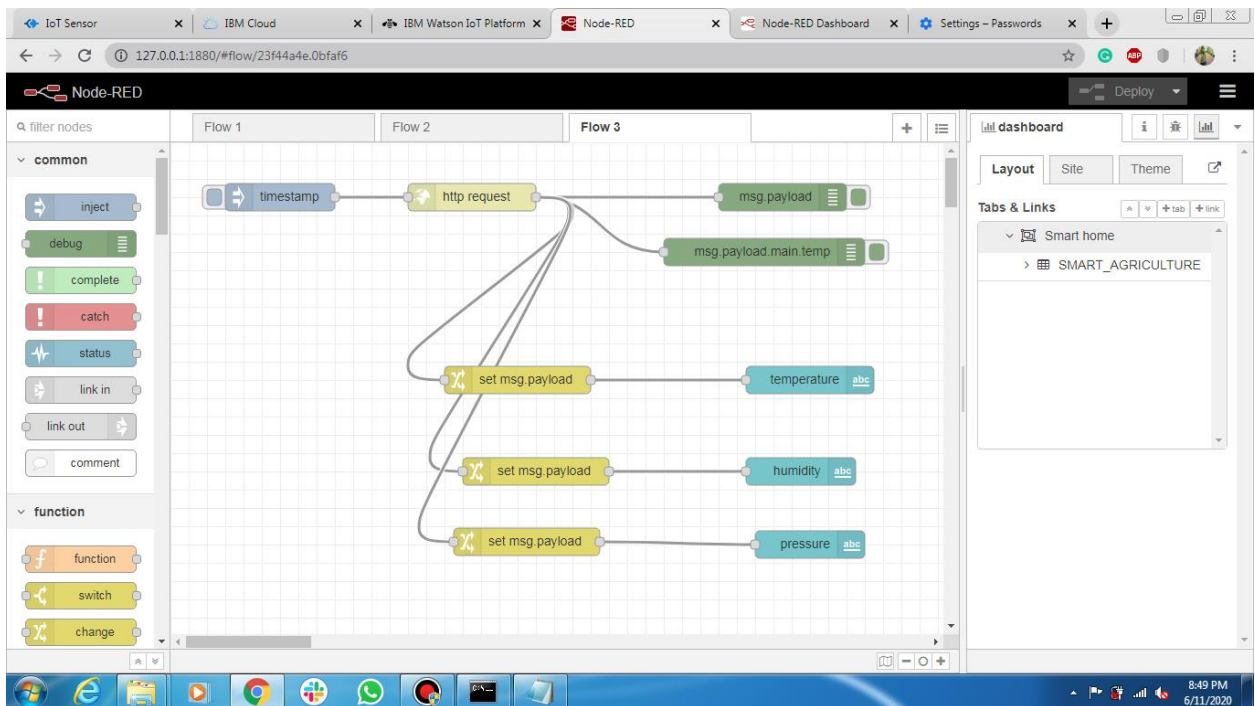
- ❖ The Node-Red also receive data from the OpenWeather API by HTTP GET request.
- ❖ An inject trigger is added to perform HTTP request for every certain interval. HTTP request node is configured with URL we saved before in section 4.4
- ❖ The URL which i have configured in API weather is the
- ❖ On clicking above Link the data we receive from OpenWeather after request is in below JSON format:

```
{
  "coord": {
    "lon": 78.47,
    "lat": 17.38
  },
  "weather": [
    {
      "id": 802,
      "main": "Clouds",
      "description": "scattered clouds",
      "icon": "03d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 307.04,
    "feels_like": 309.22,
    "temp_min": 306.15,
    "temp_max": 308.15,
    "pressure": 1008,
    "humidity": 46,
    "visibility": 6000,
    "wind": {
      "speed": 2.6,
      "deg": 340
    },
    "clouds": {
      "all": 40
    },
    "dt": 1591687945,
    "sys": {

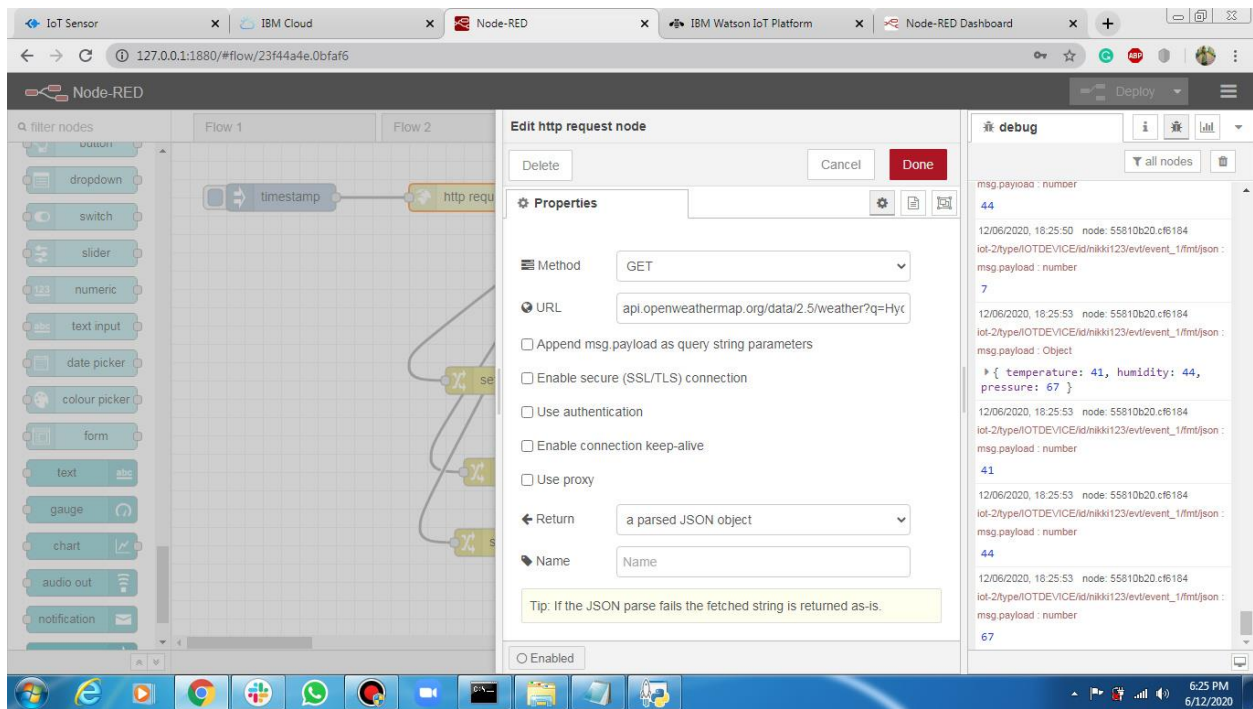
```

```
{"type":1,"id":9214,"country":"IN","sunrise":1591661474,"sunset":1591708788},"timezone":19800,"id":1269843,"name":"Hyderabad","cod":200}
```

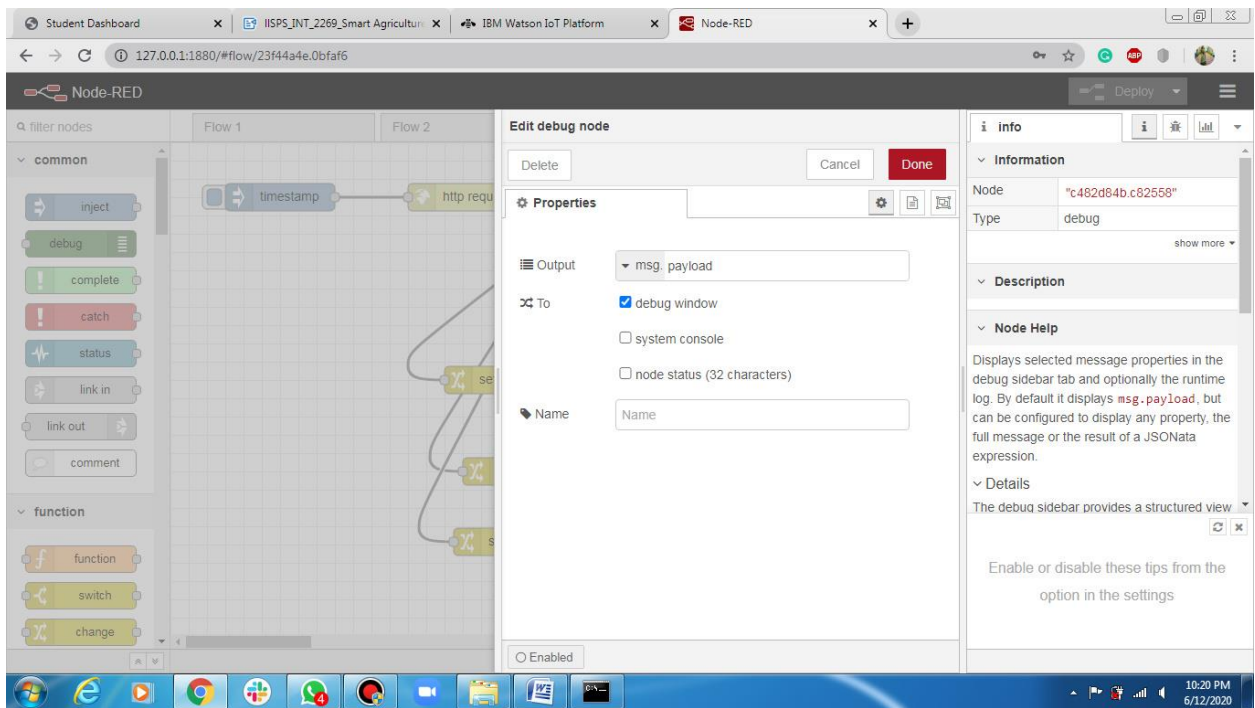
- ❖ The below image has the program flow for receiving data from OpenWeather.
- ❖ For this we will use one http request node and debug nodes. I have taken 2 debug nodes to separate temp from the api data. It is optional.



- ❖ Below is the picture of http request node properties.

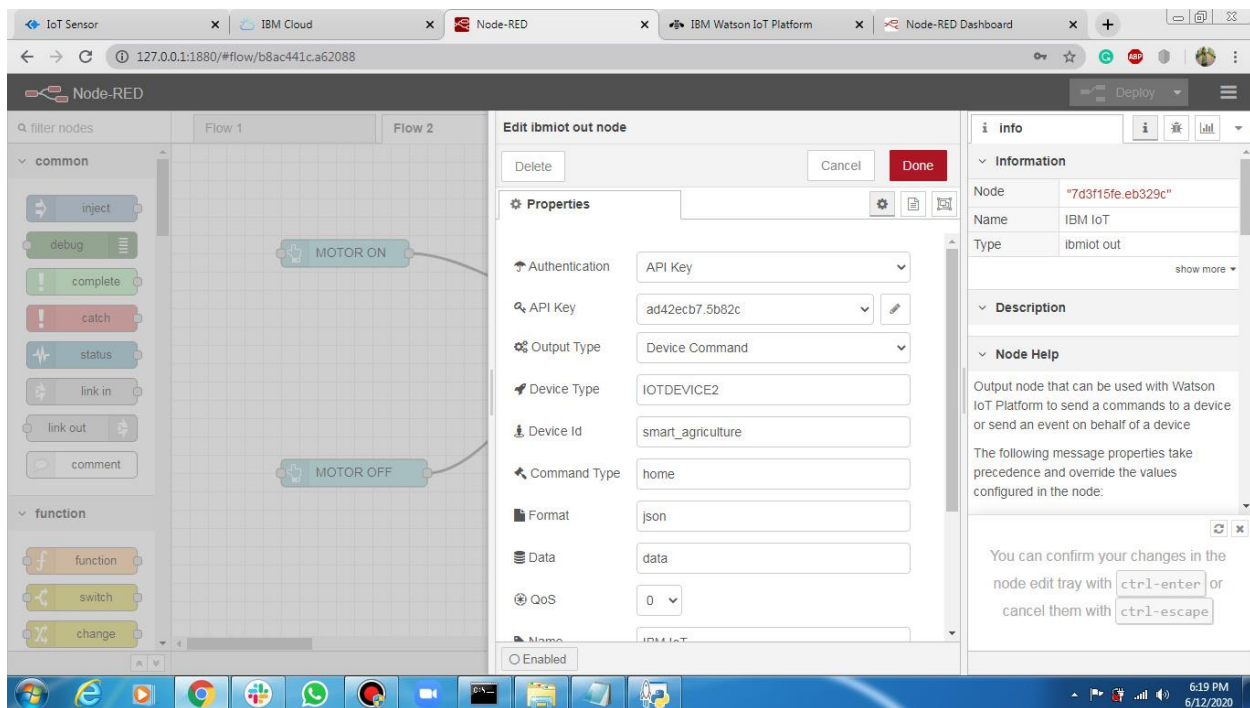


❖ Below is the picture of debug node properties which I have used to separate temperature.

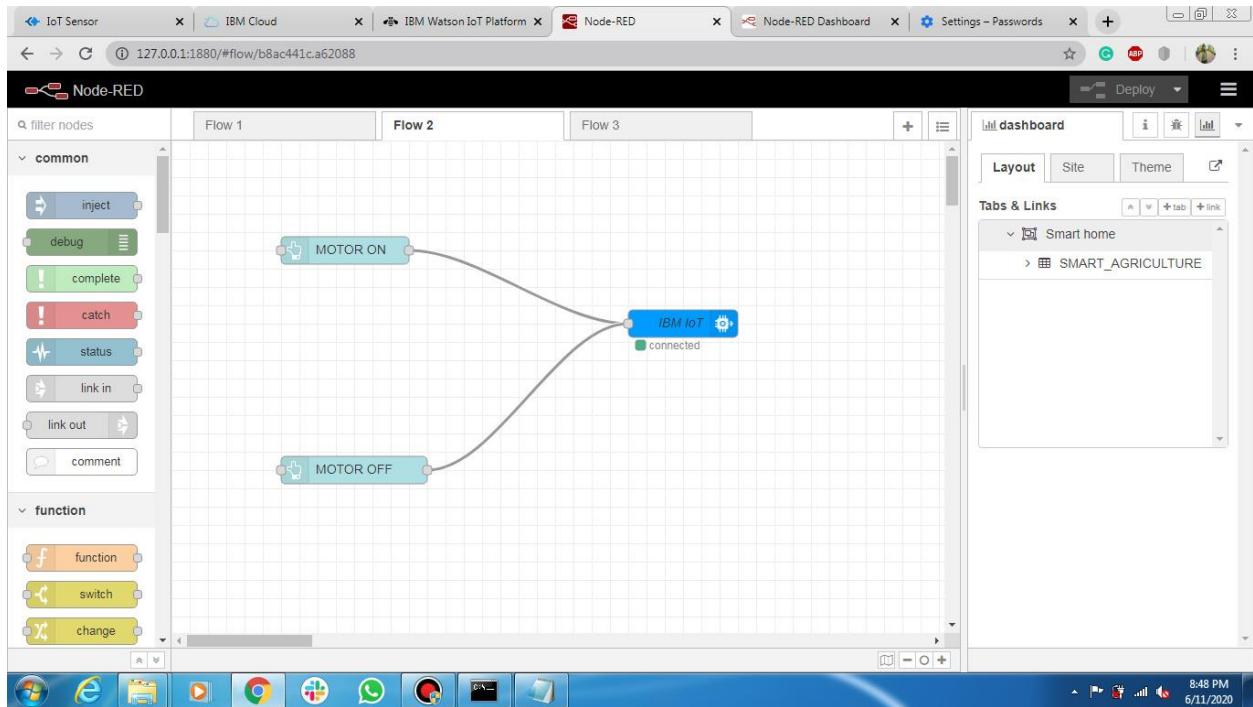


4.4 Configuration of Node-Red to send commands to IBM cloud

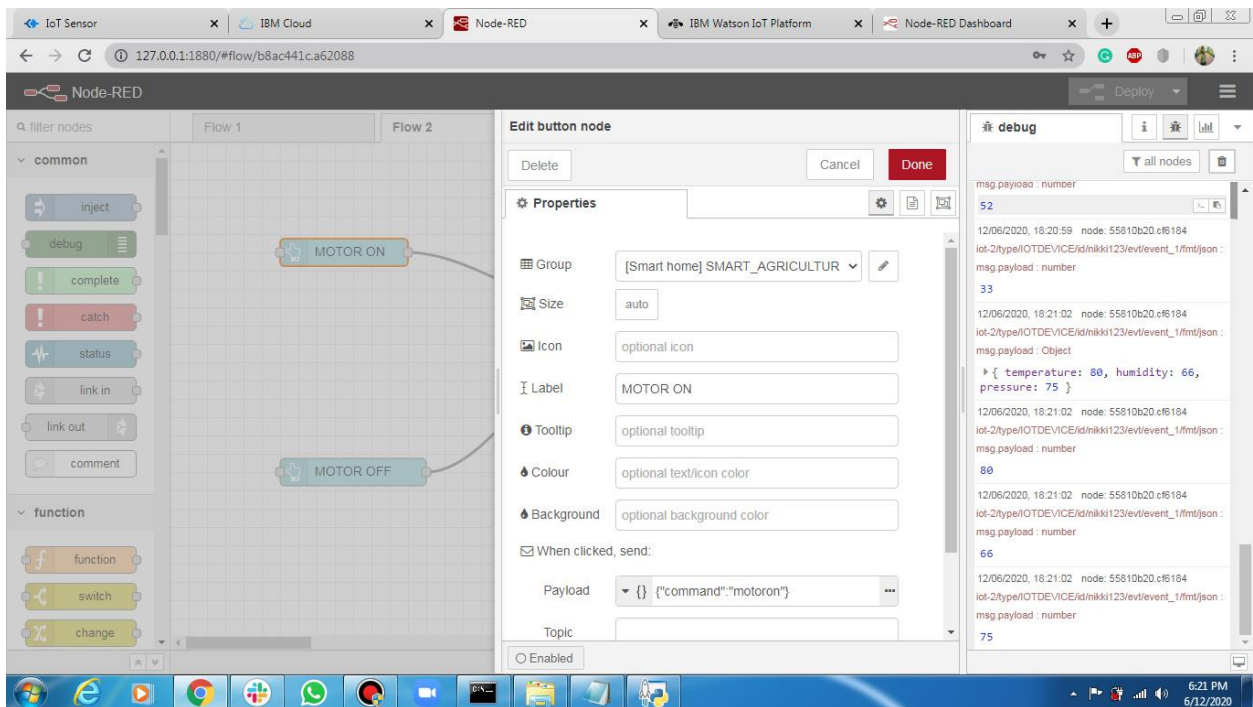
- ❖ We have to use IbmIoT out node to send data from Node-Red to IBM Watson device.
- ❖ So, after adding it to the flow we need to configure it with 2nd device credentials of our Watson device.
- ❖ Below is the picture of IbmIoT out node properties.



- ❖ Here in this flow I have given 2 buttons for MOTOR ON and MOTOR OFF .



- ❖ In UI dashboard when we click the motor on/off button we will receive these commands in the python code that we will see in the lower sections.
- ❖ If we receive the commands thus our buttons are working successfully.
- ❖ We used a button node to analyse the data received and assign a command to each number.
- ❖ The javascript code I used in button nodes to get commands is :
 - `{ "command" : "motoron" }` for MOTOR ON button
 - `{ "command" : "motoroff" }` for MOTOR OFF button
- ❖ *This code should be placed in button node as shown in the below picture.*

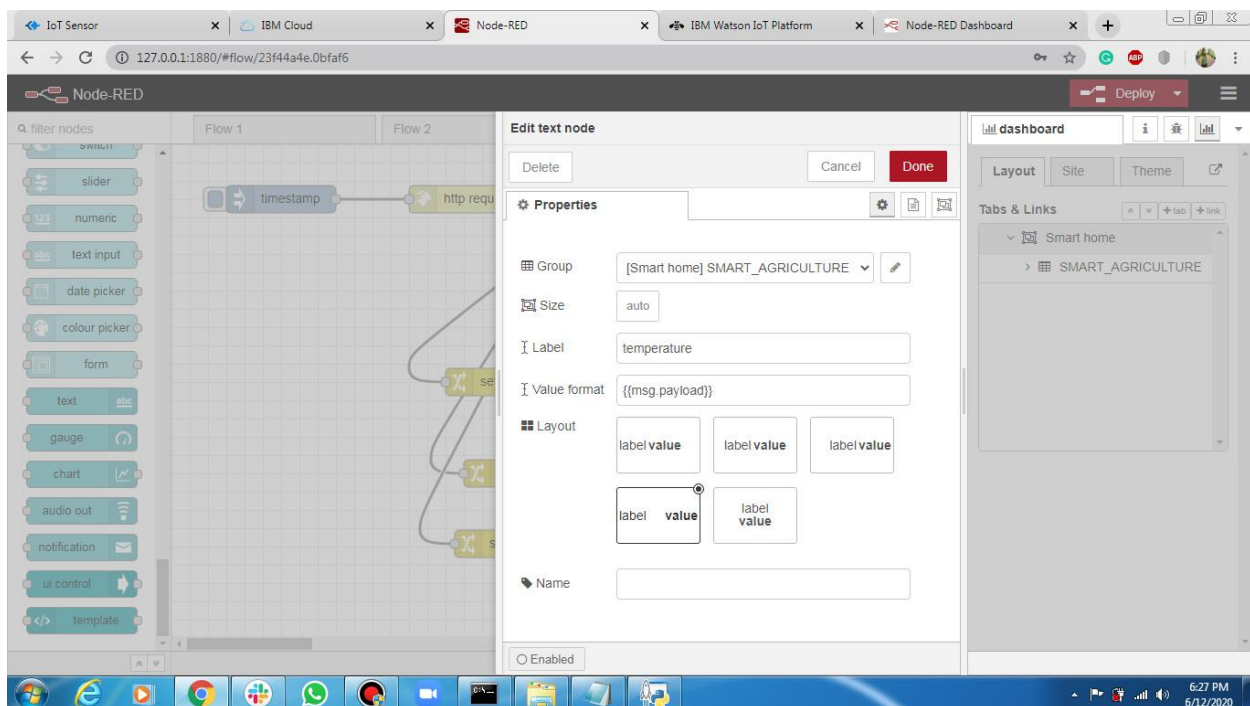


- ❖ Then we use another change node to parse the data and represent it visually with text node. This will be seen in below sections.

4.5 Building User Interface

- ❖ In order to display the JSON data we have created a Node-Red dashboard .
- ❖ Here we are using Gauges, text and button nodes to display in the UI and helps to monitor the parameters and control the farm equipment.
- ❖ I have created 3 gauge nodes to display humidity, temperature and pressure in flow 1.
- ❖ I have created 2 buttons for motor on/off in flow 2
- ❖ I have also created three text nodes with change node to convert them into numbers in flow 3 using http request from weather API to display values of humidity, temperature and pressure in UI dashboard.

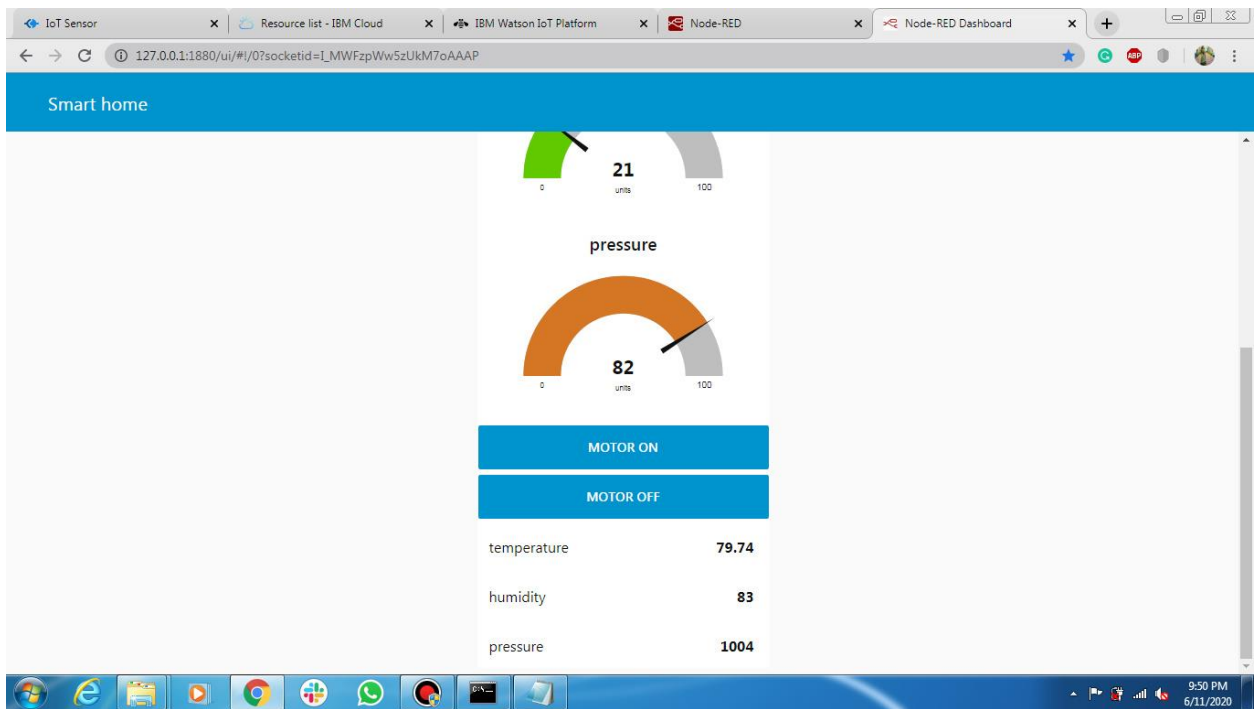
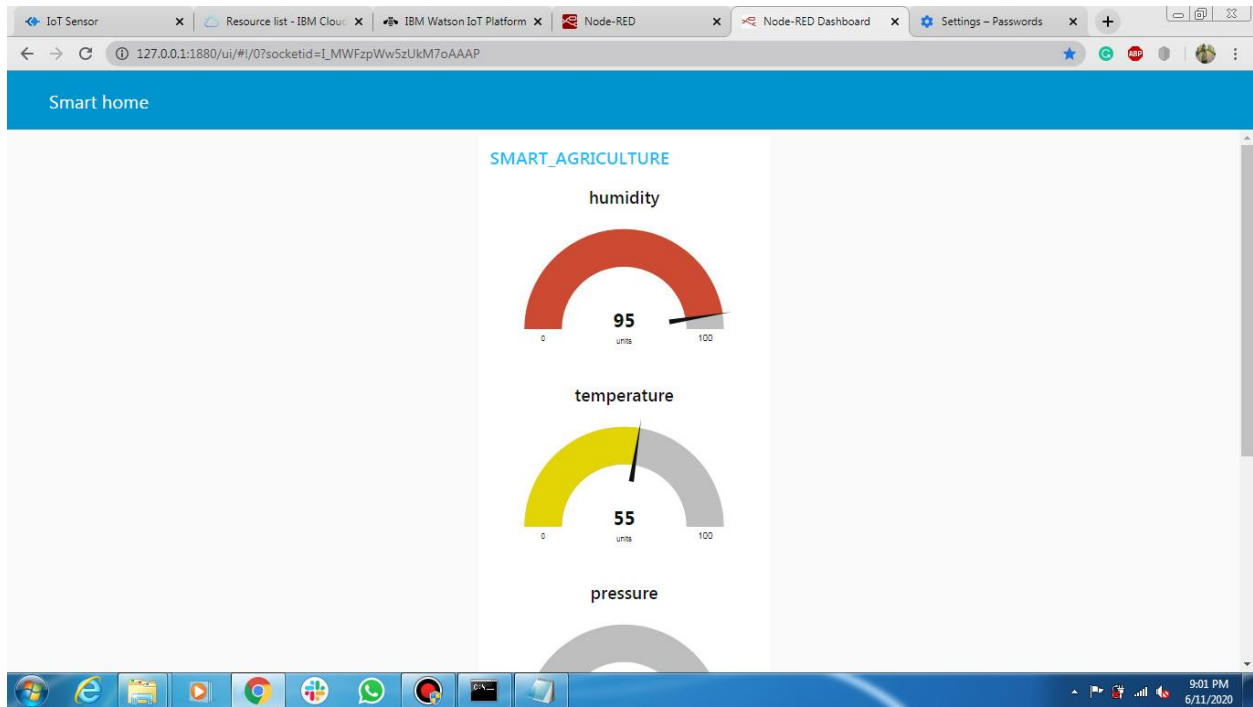
- ❖ As this all will be displayed in the UI that we can see the output.
- ❖ Below images are the all flow programs where I have created Gauge, text and button node configurations.
- ❖ Below is the picture of text-node properties. By using this node we can display temperature etc fields in UI Dashboard. similarly we use same properties for text node of humidity and pressure.



4.6 WEB APP UI

Smart Home Tab :

- ❖ This is the tab of UI dashboard where we have all nodes in node-red flow , Here is the our desired output.
- ❖ This is the tab where we have created motor buttons also in nodered flow. On clicking these button commands we can control the motor status to on/off .



4.7 Receiving commands from IBM cloud using Python program

- ❖ I have given here second device credentials because we should not use same device for IoT simulator and Python code this will disconnect the device and throws an error, so use different device credentials only for both .
- ❖ Below is the python code program we should use to receive commands .

```
import time
import sys
import ibmiotf.application # to install pip install ibmiotf
import ibmiotf.device

#Provide your IBM Watson Device Credentials
organization = "zps84k" #replace the ORG ID
deviceType = "IOTDEVICE2"#replace the Device type wi
deviceId = "smart_agriculture"#replace Device ID
authMethod = "token"
authToken = "987654321" #Replace the authtoken

def myCommandCallback(cmd): # function for Callback
    print("Command received: %s" % cmd.data)
    if 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 cloud as an event  
of type "greeting" 10 times
```

```
deviceCli.connect()
```

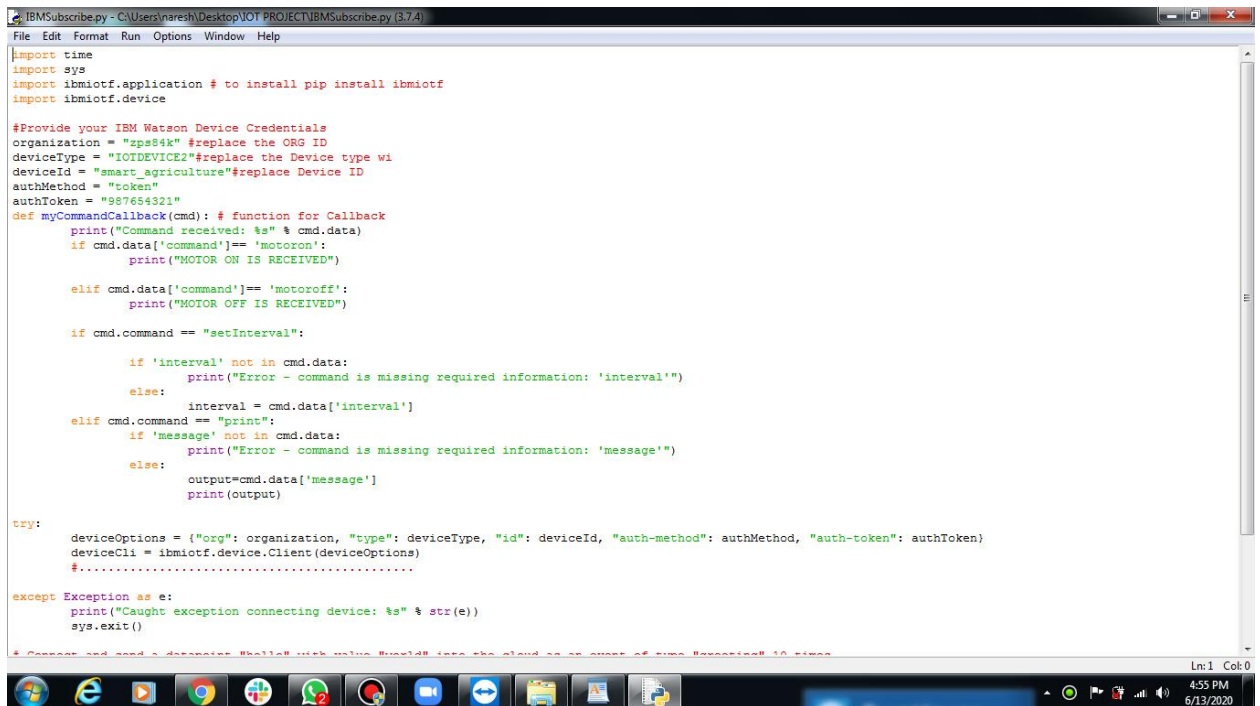
```
while True:
```

```
    deviceCli.commandCallback = myCommandCallback
```

```
# Disconnect the device and application from the cloud
```

```
deviceCli.disconnect()
```

❖ Below picture is the python code :



```
IBMSubscribe.py - C:\Users\naresh\Desktop\IOT PROJECT\IBMSubscribe.py [3.7.4]
File Edit Format Run Options Window Help

import time
import sys
import ibmiotf.application # to install pip install ibmiotf
import ibmiotf.device

#Provide your IBM Watson Device Credentials
organization = "zps84k" #replace the ORG ID
deviceType = "IOTDEVICE2" #replace the Device type w1
deviceId = "smart_agriculture" #replace Device ID
authMethod = "token"
authToken = "987654321"

def myCommandCallback(cmd): # function for Callback
    print("Command received: %s" % cmd.data)
    if 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)

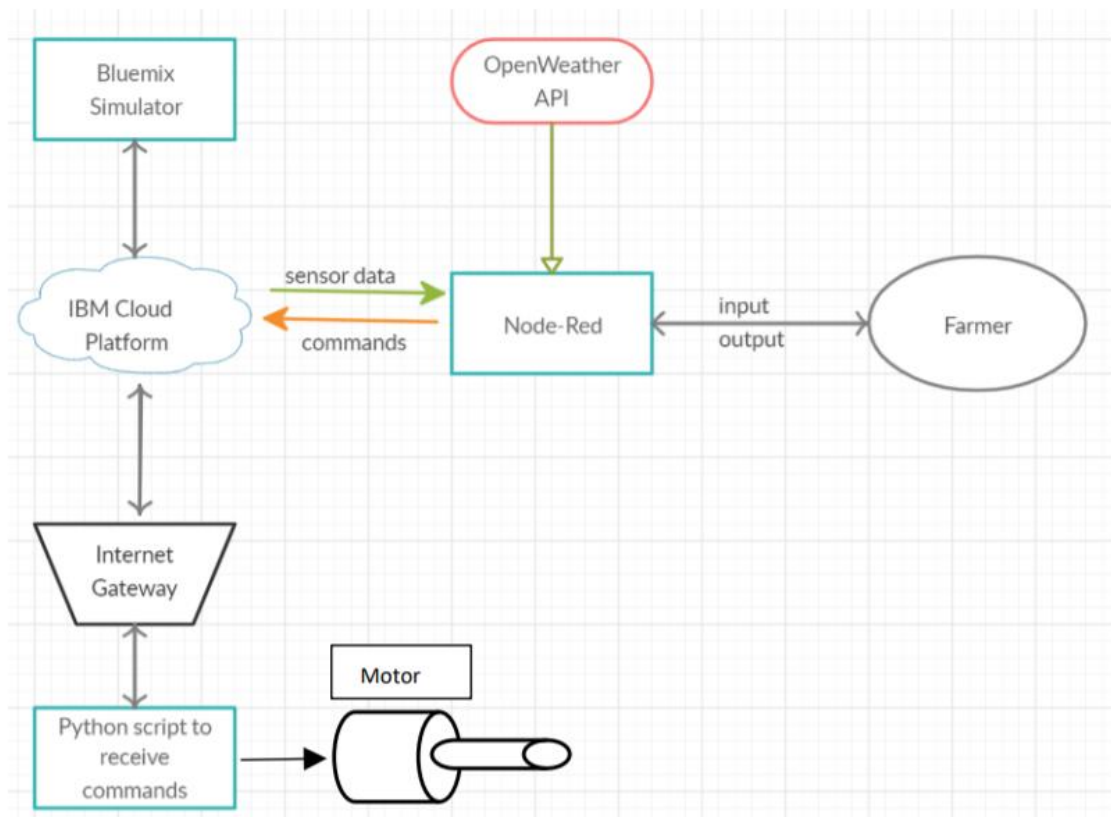
try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #.....

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 cloud as an event of type "greeting" 10 times
```

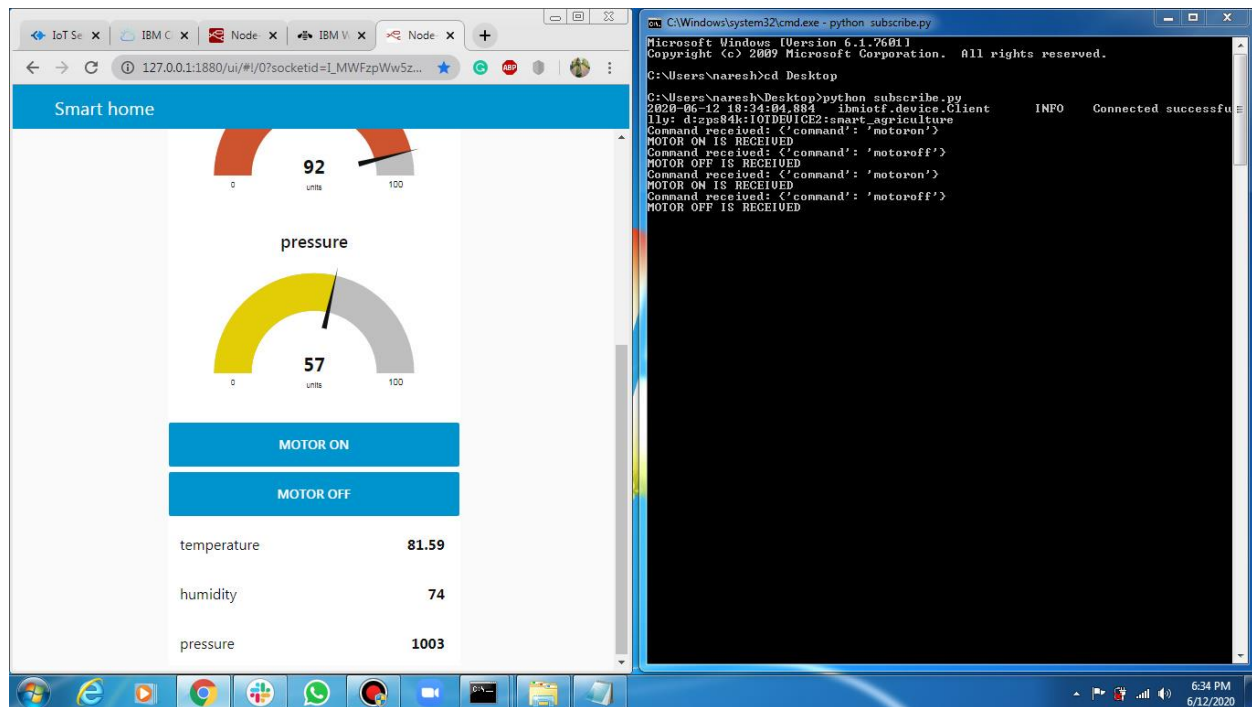
5.Flow Chart

❖ Below is the flow chart diagram of whole process .



6. Result

- ❖ As a result we will be able to receive the motor on/off commands from the UI dashboard that I have created.
- ❖ We can control the motor by motor commands that are given in UI.
- ❖ Below is the picture of our desired result.



7. Advantages & Disadvantages

Advantages :

- ✓ Farms can be monitored and controlled remotely.
- ✓ Improves livestock monitoring and management.
- ✓ Less labour cost .
- ✓ Better standards of living.
- ✓ Very helpful in knowing weather conditions.
- ✓ Increases the quality of production and water conservation.

Disadvantages :

- ✓ Lack of internet/connectivity issues.
- ✓ Added cost of internet and internet gateway infrastructure.
- ✓ Loss of privacy and security.

- ✓ Less compatibility and more complexity.
- ✓ Farmers are not used to these type of high-end technologies.

8.Applications

- ❖ In this project we have created one web application called "Smart_agriculture System".
- ❖ By using the IBM cloud resources, IBM Watson IoT platform we have configured to node-red.
- ❖ Weather API is applied to get current weather conditions.

9.Conclusion

- ❖ Thus the objective of the project to implement an IoT system in order to help farmers to control and monitor their farms has been implemented successfully.
- ❖ This will enhance the productivity in achieving better yield of crops.

10.Bibliography

- **IBM cloud reference** : <https://cloud.ibm.com/>
- **IoT simulator** : <https://watson-iot-sensor-simulator.mybluemix.net/>

- **OpenWeather :** <https://openweathermap.org/>
- **Python code reference:**
<https://github.com/rachuriharish23/ibmsubscribe>
- **IBM Watson IoT Platform :**
<https://internetofthings.ibmcloud.com/>