

Introduction

1.1 Overview

- Smart Agriculture is an IoT based project which involves various software that enables us to make the process of Agriculture easy. By using this "Smart Agriculture" farmer can get all the information such as temperature ,cloud, wind speed etc ; on his mobile. He can monitor and control his watering system by using buttons displayed on his mobile.

By using ibm cloud platform, ibm sensor ,node red platform along with its dashboard ,and a python compiler , a dashboard was created which will display **temperature, humidity, object temperature, windspeed, clouds, and two buttons to switch the motor on or off.**

1.2 Purpose

Countries like India depends majorly on agriculture sector. But still we are facing a lot of problems in agriculture. There are lack of resources, funds and even price for goods in markets. As an engineer, we may simplify the process of agriculture by making it a bit faster by providing them with limited resources and efficient outcome. This is possible with Internet of Things. By using these things we will create wonders . We all know that internet is flying all over the world and we connect devices to it we make our work easy.

Literature survey

2.1 Existing problem

Farmers need to deal with many problems including how to cope up with climate change, Irrigation facilities, soil erosion, biodiversity loss Lack of investment in agriculture, negligence of natural resources sluggish fertilizer industry and dependency on monsoon.

farmer to know the temperature, humidity , soil erosion , drone status, animal gazing etc;

SMART AGRICULTURE

Climatic change has a great impact on Indian agriculture .Sudden change in humidity sometimes may result unseasonal rains which may cause damage to crop. If the farmer is able to know climatic changes a little bit early ,the farmer may protect his crop according to his need. However, there are some IoT based applications such as cropx's soil monitoring system which measures moisture, temperature , and electrical conductivity in the soil. There are other applications also which monitors animal gazing, detecting obstacles for drones, etc;

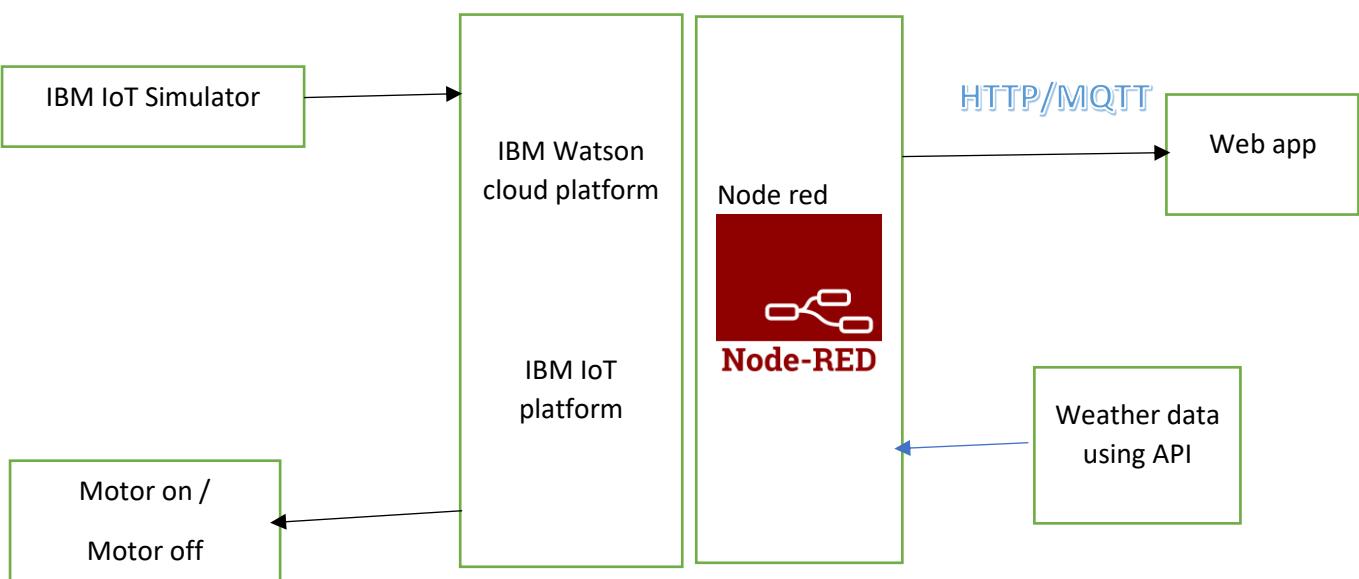
2.2 Proposed solution

This application displays current temperature and windspeed of a city so that farmer can plan his day accordingly. It also contains two buttons as 'motoron ' and 'motoroff' by which farmer can control his motor on farm.

So by using this farmer can cope up better with climatic changes. As irrigation is also serving as one of the major problem in agriculture sector , if we supply water as much as needed to farm , water can be conserved and may supplied to another farms. As operation of motor is in farmer's hand ,he can irrigate as much as he required and can save more.

Theoretical analysis

3.1 Block diagram



3.2 Hardware/ Software designing

Node Red

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of **nodes** in the palette that can be deployed to its runtime in a single-click. We can see output, connect it to various platforms, (IBM Cloud, other websites using API keys etc)

The data from connected platforms will be displayed on node red, and by using Node red dashboard we can display data in a colourful way.

IBM Cloud

IBM Watson IoT platform

IBM is a cloud platform which provides various services for its users to work on various projects. One of its service ‘Internet of Things’ provides us with sensor and we can connect it to various other platforms such as node red by giving device details and API Keys. It will display how many services connected, what was the data flowing in each service in json format . also we can use same configurations of devices connected whenever we use them.

Open weather API

Open weather map is one of the most popular source that provides global weather data via API, including current weather data , forecasts , nowcasts an historical weather data for any geographical location. We can access this data through other apps with API keys by creating an account in this website and API keys are personal once it is connected to device it will give current weather of city .

Pycharm

pycharm is a python compiler available in two versions , community edition and profession edition. Community edition is a free and open source . it includes all the libraries available in python 3.6 to 3.10 versions. we can use it wherever we want like in other websites for web development and others and we can modify source code. We can connect it into various websites and can operate things through it.

Experimental Investigations

1. Creating an account in IBM cloud

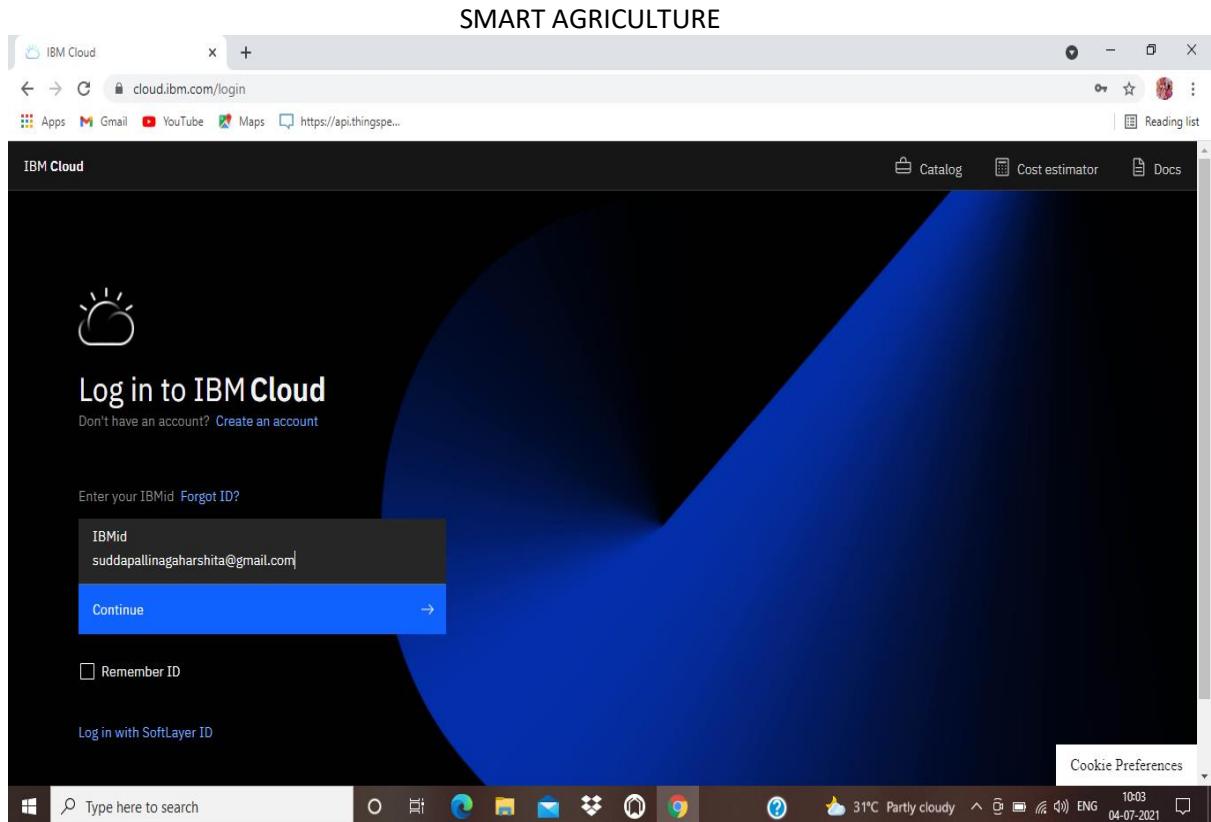


Figure-1

Create an account in IBM cloud using a mail ID and provide required details .we need to sign in with this account every time we use this platform.

SMART AGRICULTURE

2. Selecting a service

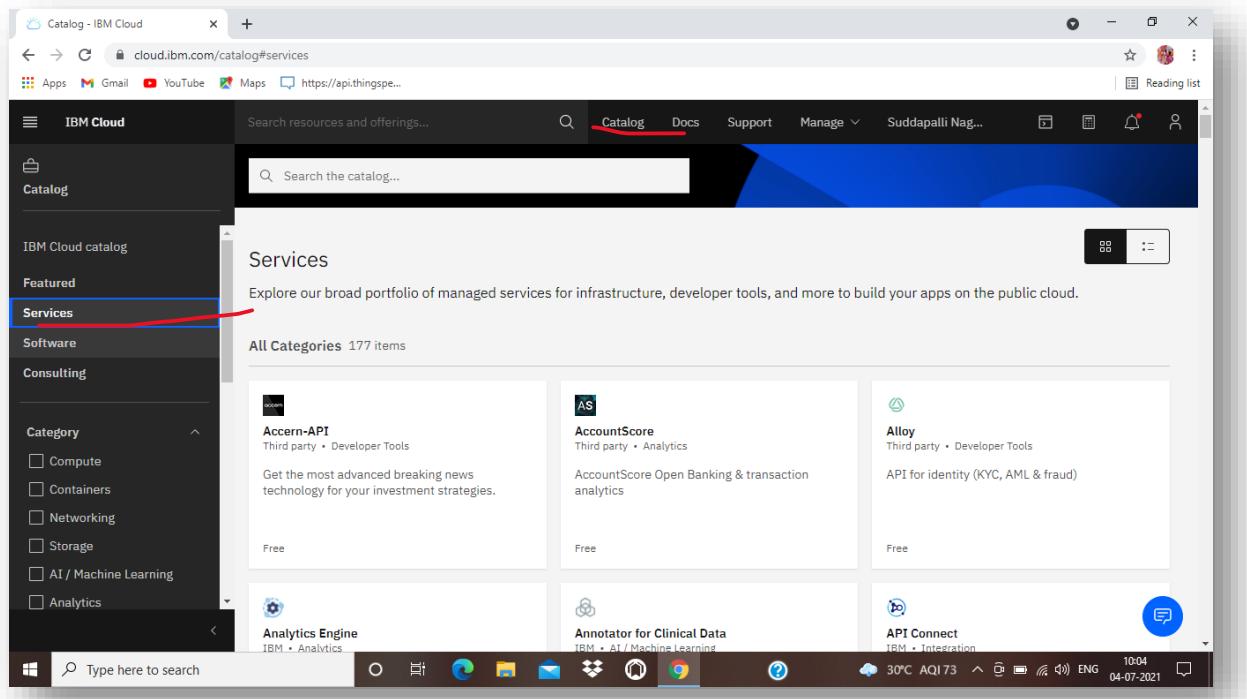
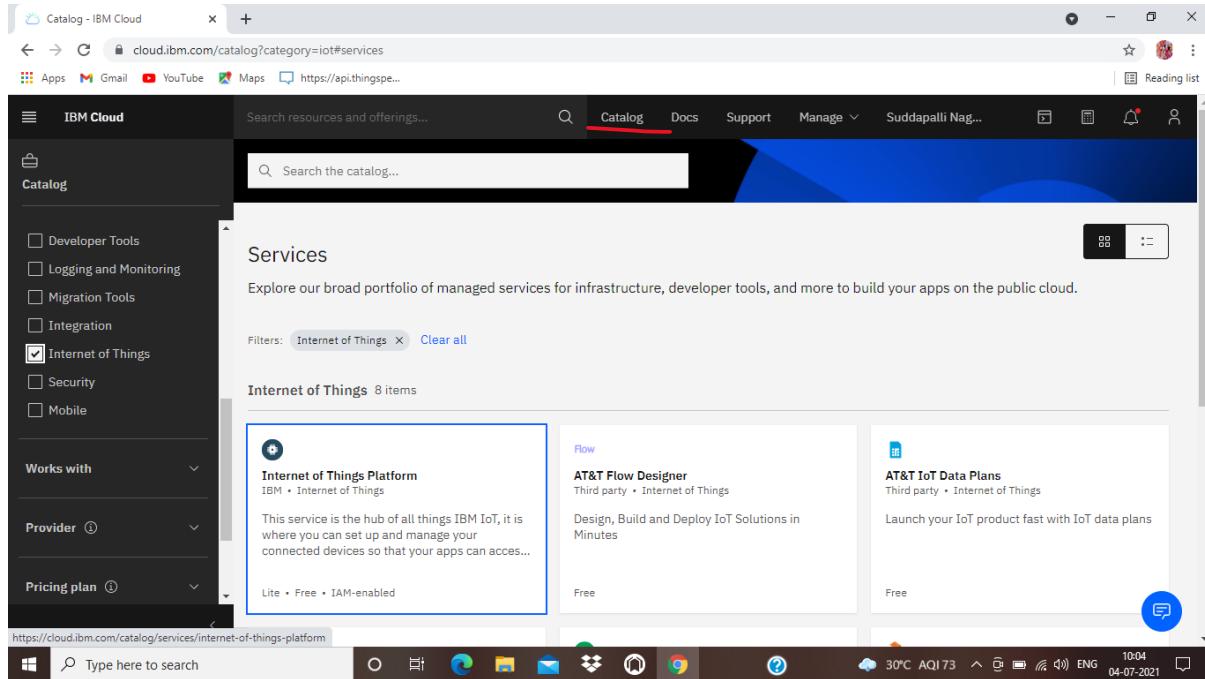


Figure-2

After creating account go to catalog--> services→and then choose Internet of Things as shown in figure-2.

3. Choosing Internet of Things



SMART AGRICULTURE

Figure-3

After selecting Internet of Things click on the first box which will appear in figure-3.

4.creating a IoT service

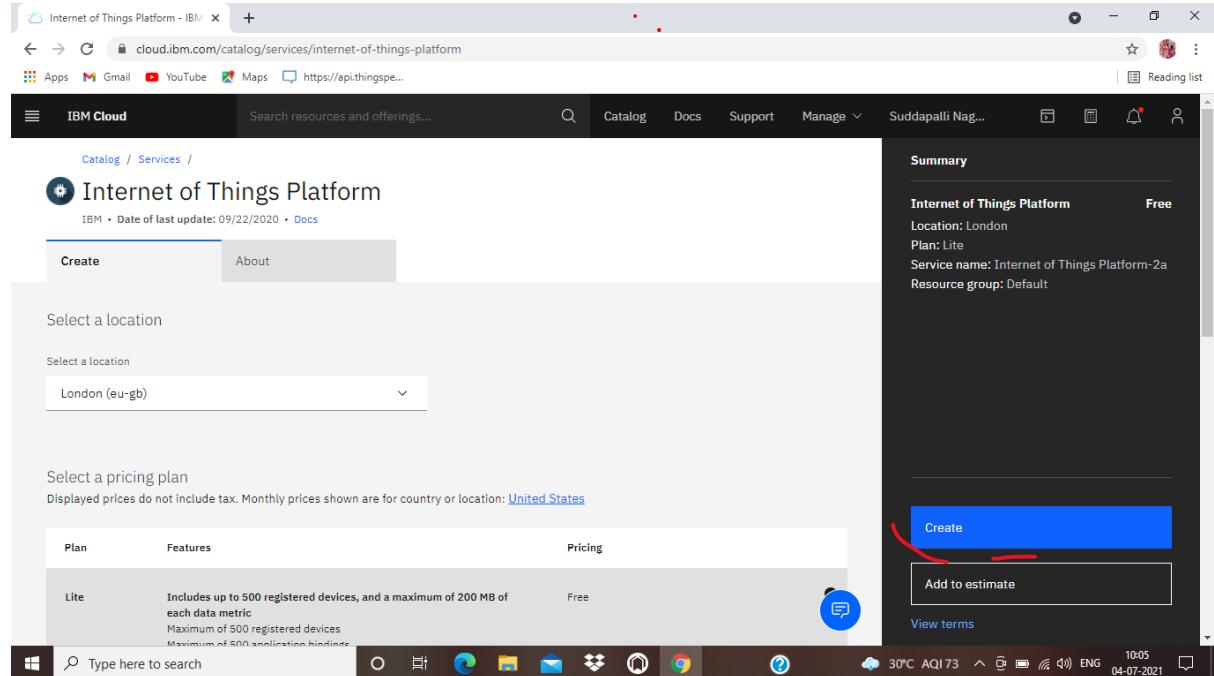


Figure-4

After choosing Internet of Things create a service by choosing create option shown in figure - 4. Choose cloud platform location if want. At present we have used cloud platform which is located at London.

5.starting Node-red

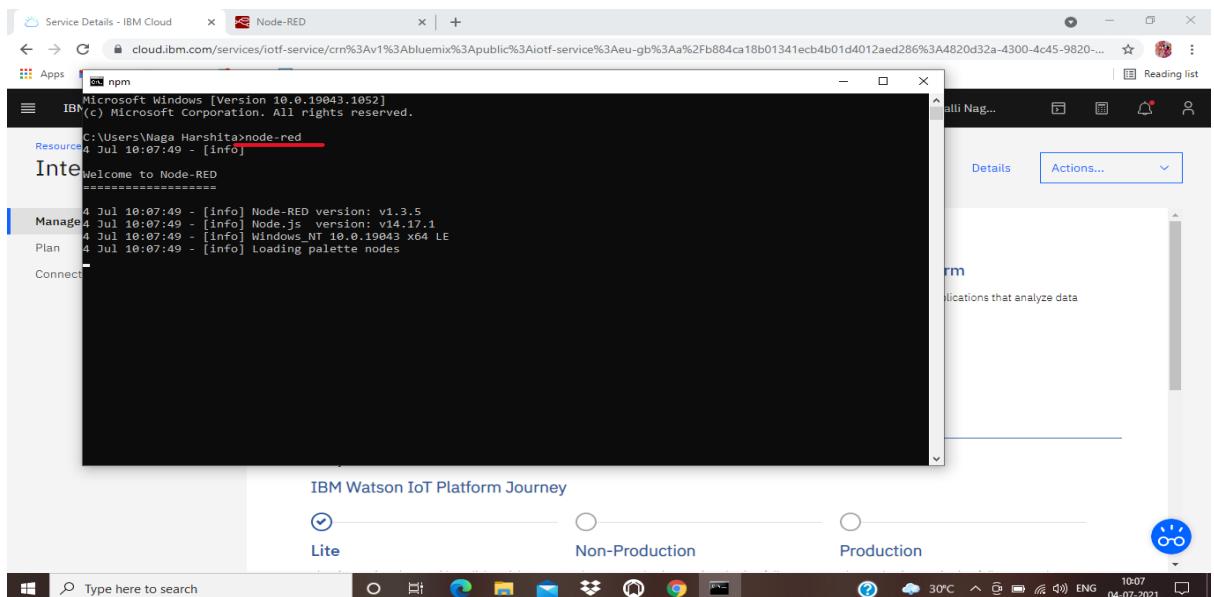


Figure-5

SMART AGRICULTURE

Note: To use node-red we have to install node-red software in our system.

After creating a service in IBM cloud start your Node -red from command prompt by giving a command “node-red” and press enter. The node red will get started as shown in figure-5

6.getting into Node-red server

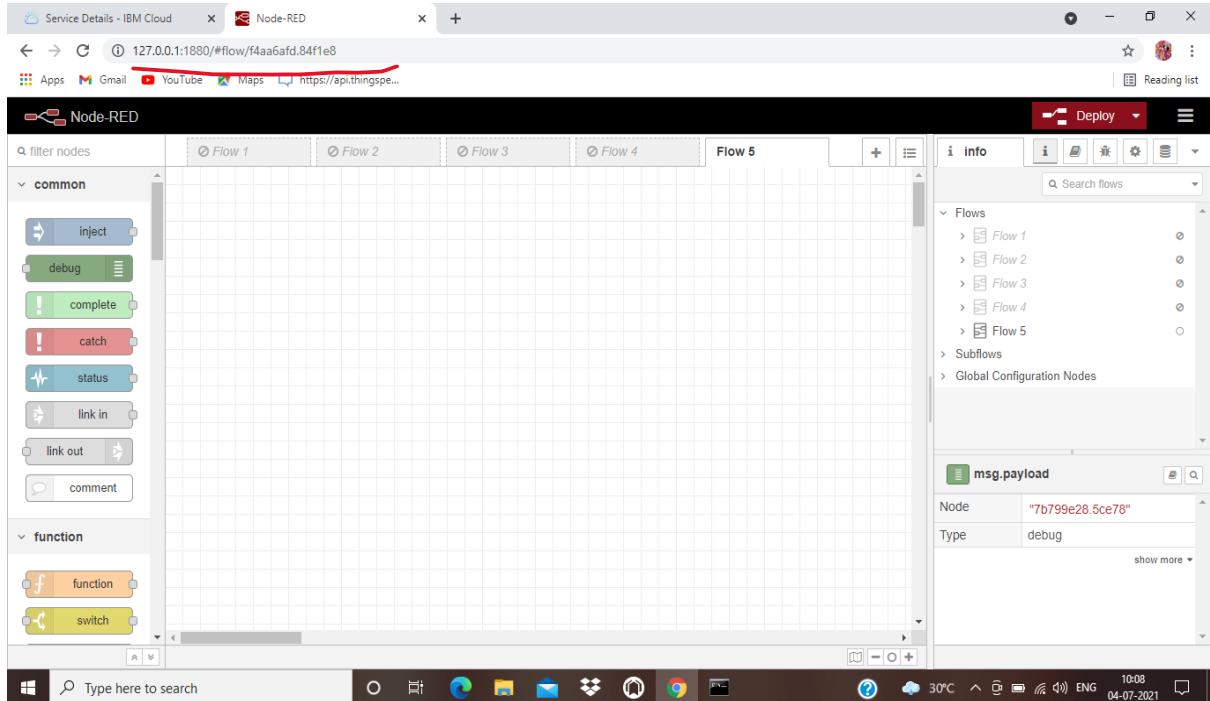


Figure-6

When you will start node-red in your system address of node-red will be appear in command prompt(<http://127.0.0.1:1880/>) . We have to copy and paste it properly or else we can type same address on google it will redirect into the page which is shown in figure-6.

SMART AGRICULTURE

7.launching our IoT service

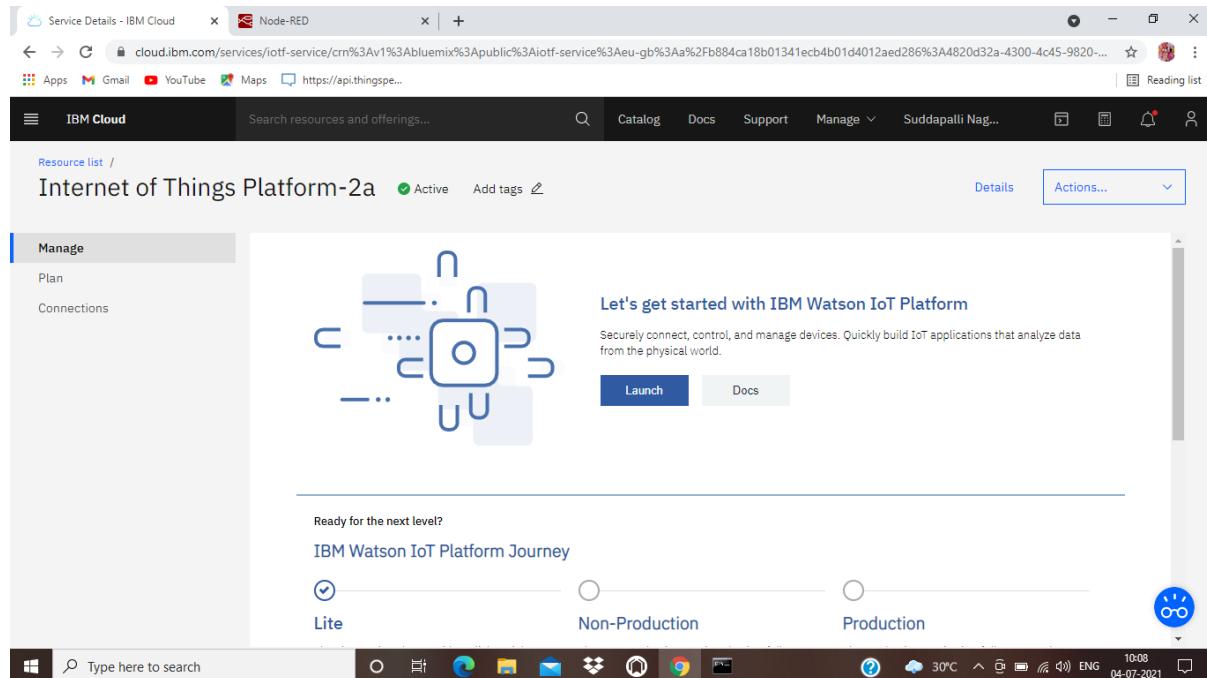
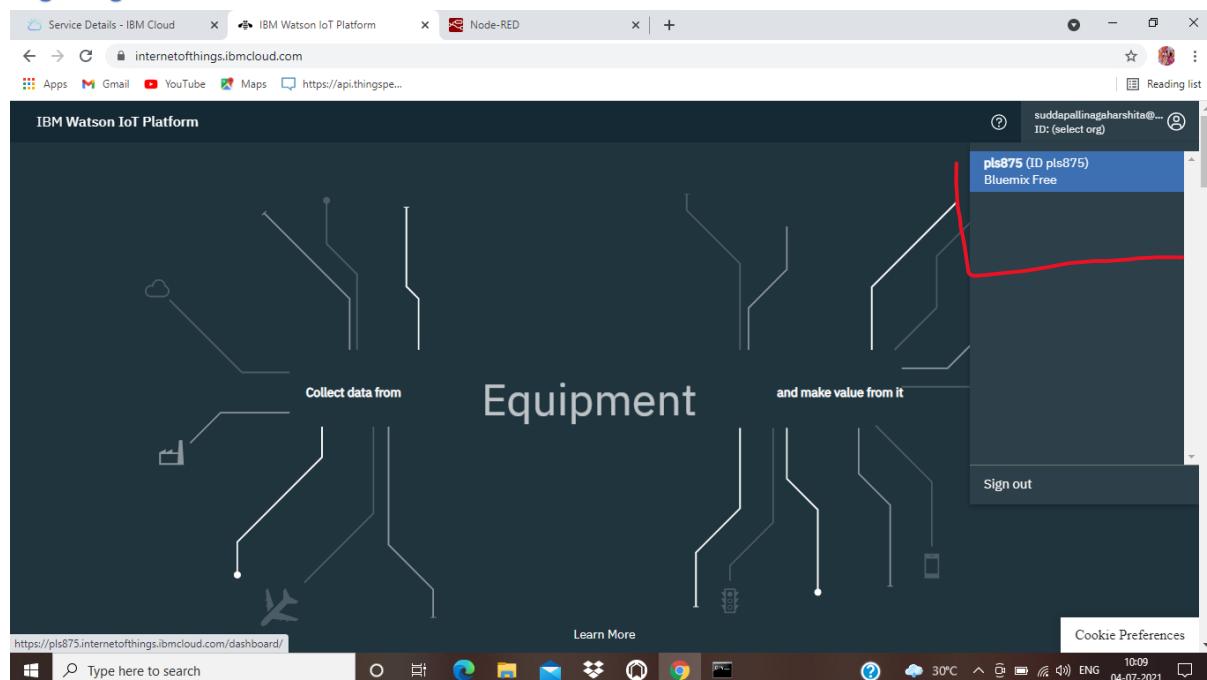


Figure-7

After starting node-red again come to IBM Cloud homepage , there we will find our created services. By choosing services we will get our service Internet of Things and by choosing that option the above page in figure-7 will appear. Then click on launch button.Then we will be redirected into an IBM Watson IoT platform in figure-8.

8. getting into IBM watson



SMART AGRICULTURE

Figure-8

Then we have to sign in again into this with our IBM account if necessary and have to select an ID(unique for every one) which will appear on choosing our account at the top right corner as shown in figure-8.

9. Adding device type

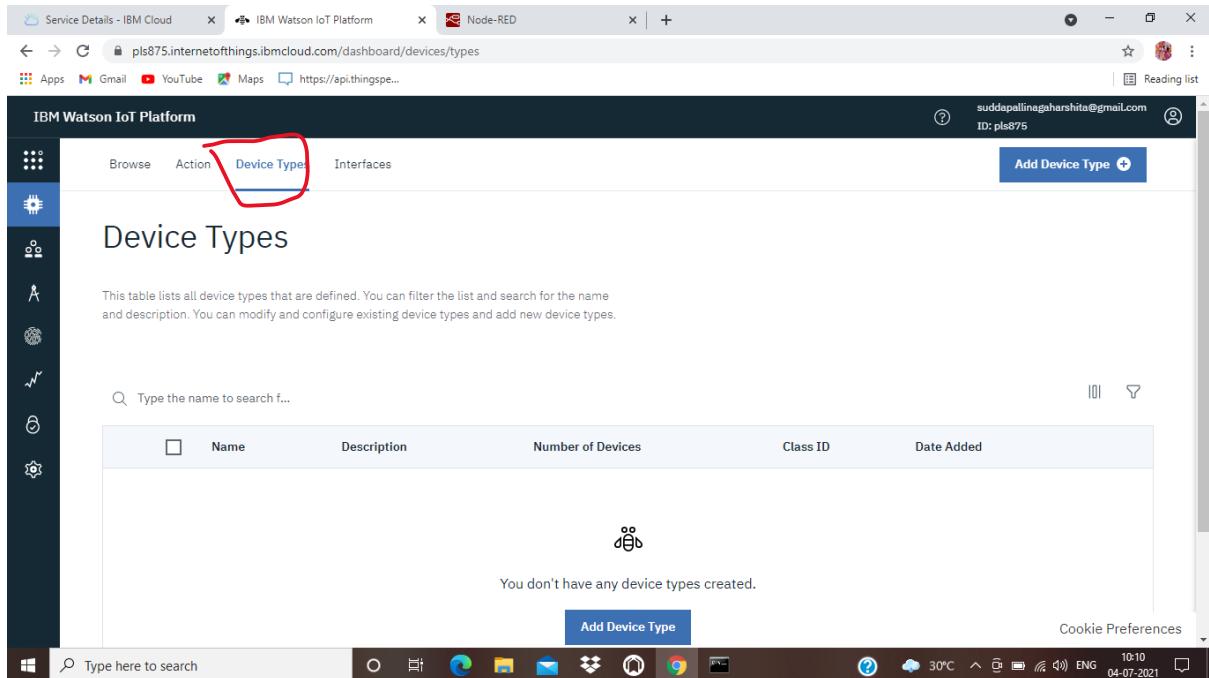


Figure-9

After the page shown in figure-9 will appear. In that we have to choose device types on the top left. And click on add device type which is in top right corner of the page.

SMART AGRICULTURE

10.Device name

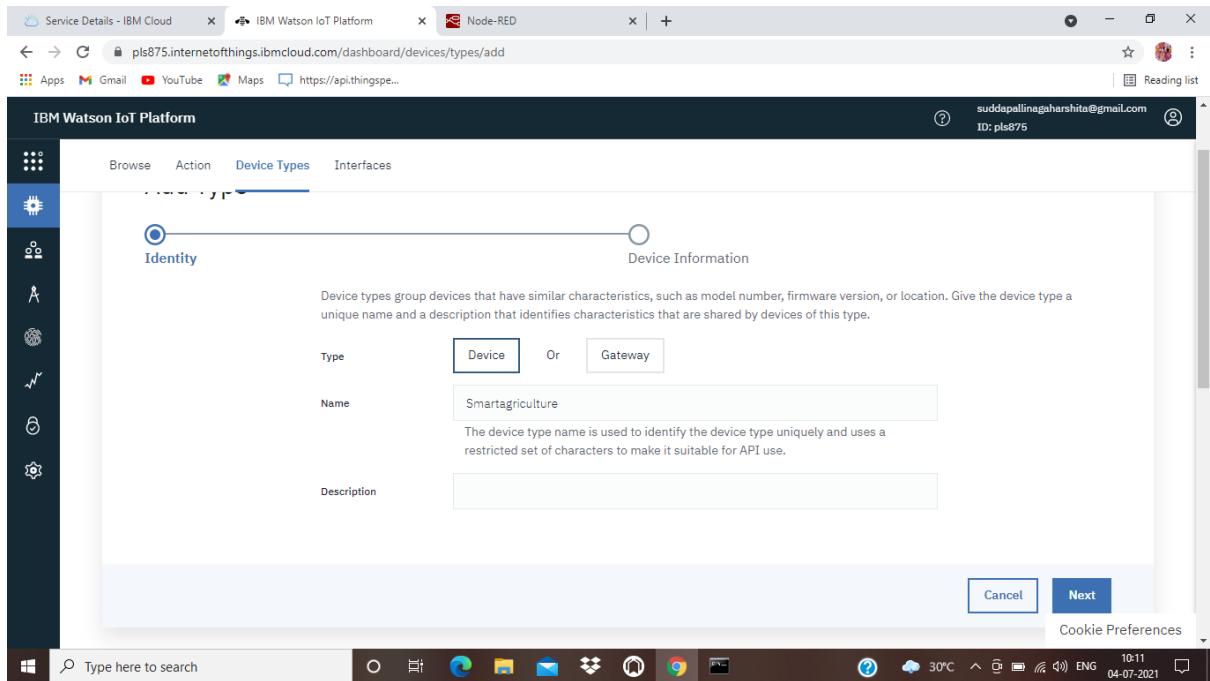


Figure-10

After choosing device type we have to give device name and while working with hardware we may specify pin numbers those connected to hardware objects etc; otherwise we may finish the process. Then it will ask to register device type and after choosing this option the page in figure-11 will appear.

SMART AGRICULTURE

11. Device details

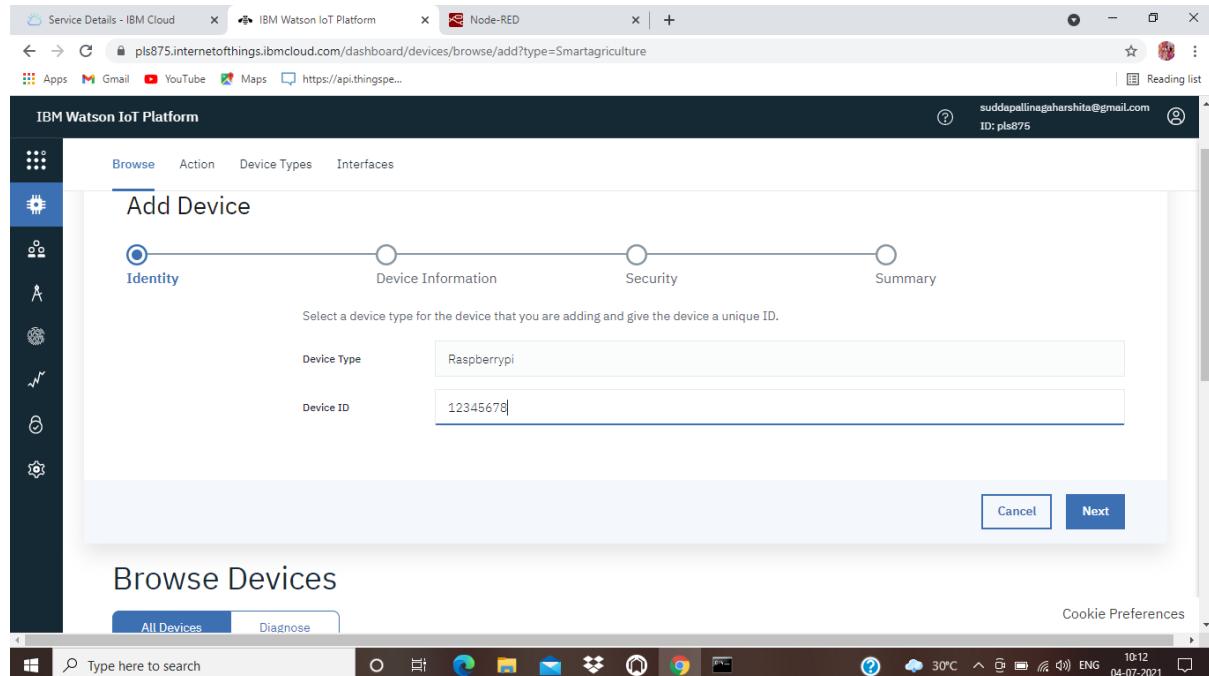


Figure-11

Here we have to give specifications of our device type such as name, ID ,authentication token for security .

12.summary of our device

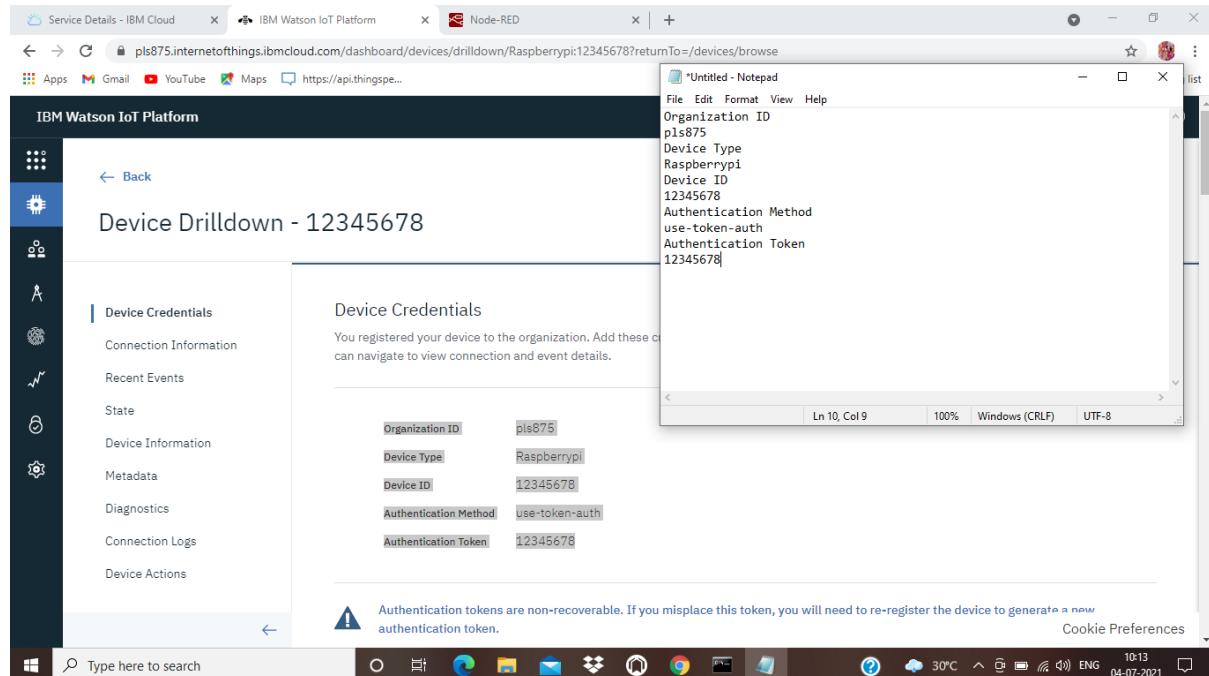


Figure-12

SMART AGRICULTURE

After providing all these details finally summary page will appear displaying all details that we have provided before as shown in figure-12. We need all these details to connect with our sensor and node-red. For that purpose usually we will copy and store it in somewhere in our system(notepad).

13.sensor

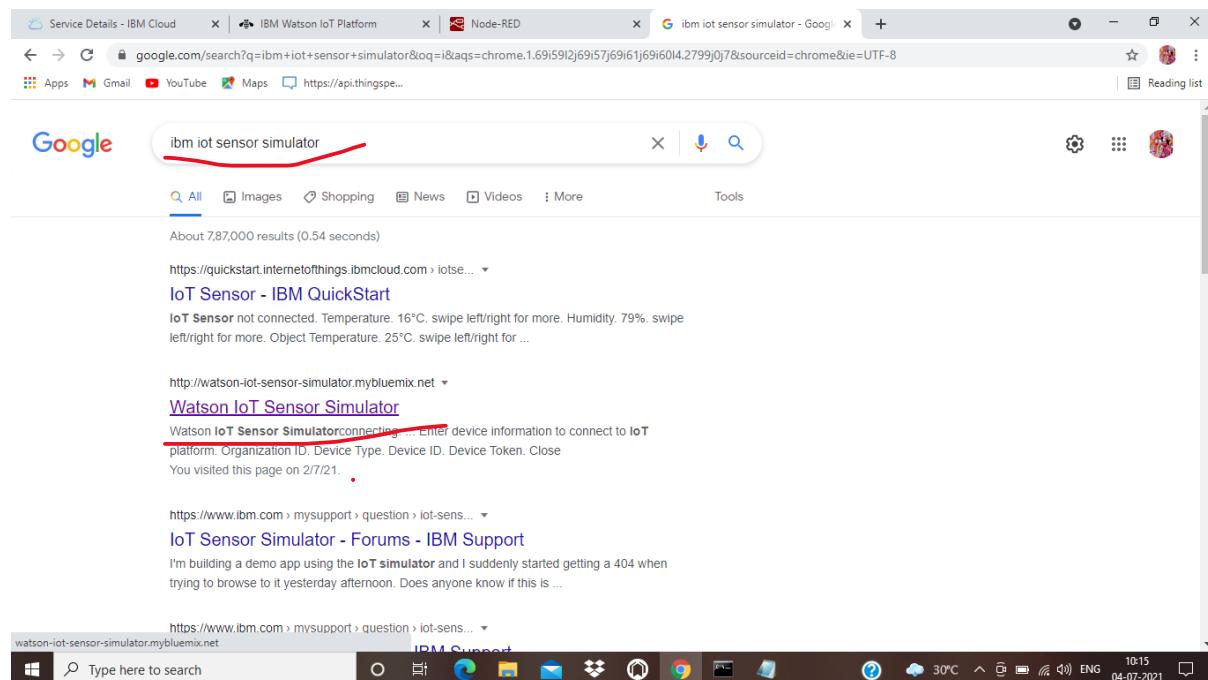


Figure-13

Now, again we need to go to new tab and search for “IBM IoT sensor simulator”. And in that we will choose second one named “Watson IoT sensor simulator”.

SMART AGRICULTURE

14. configuring the sensor

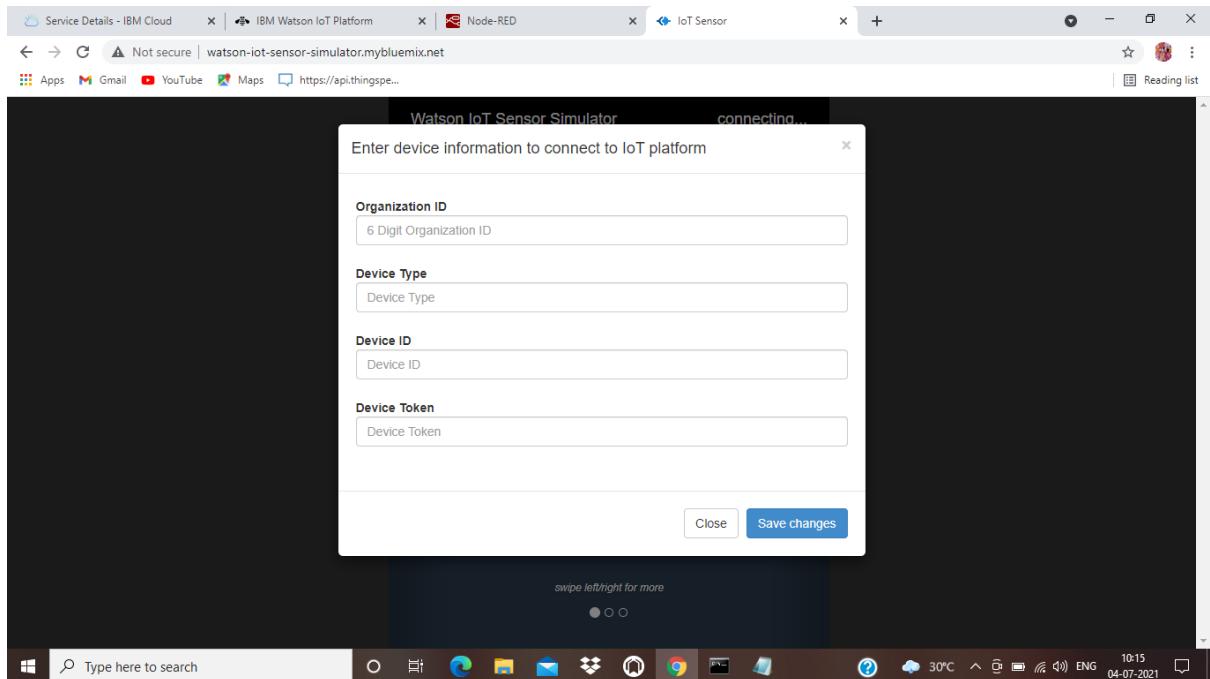
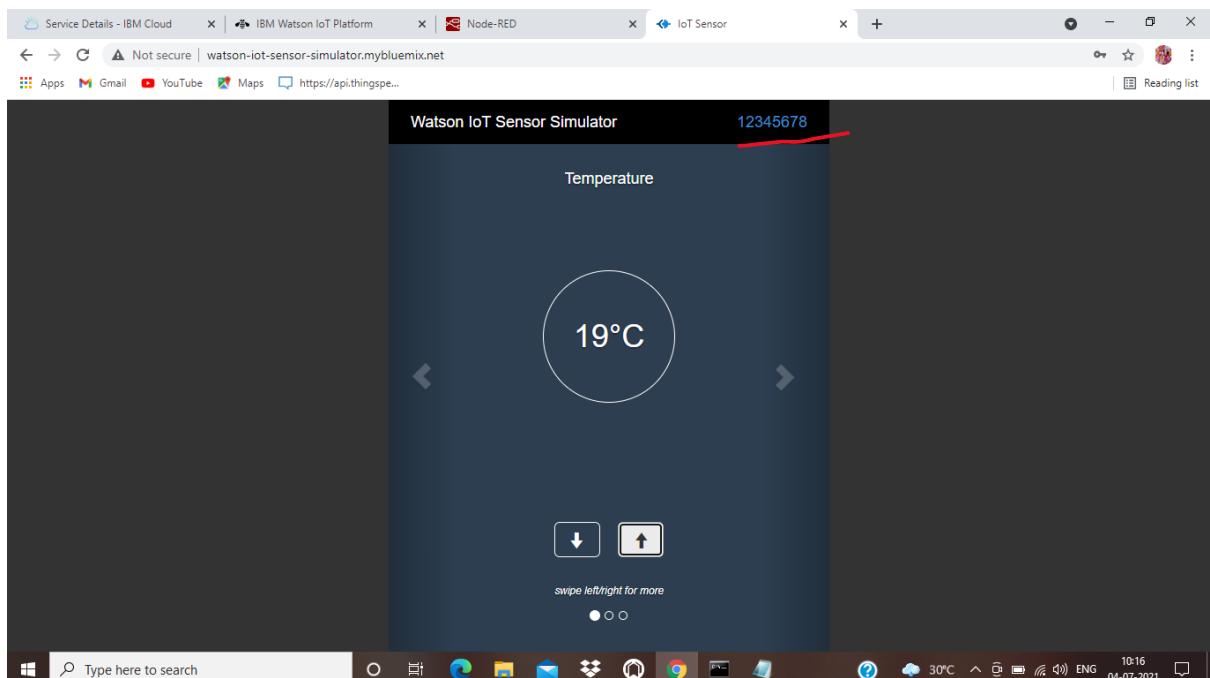


Figure-14

Now we are asked to enter details of our device as shown in figure-14. We have to enter those details that previously we have saved. And later click on “save changes”.

15. sensor data



SMART AGRICULTURE
Figure-15

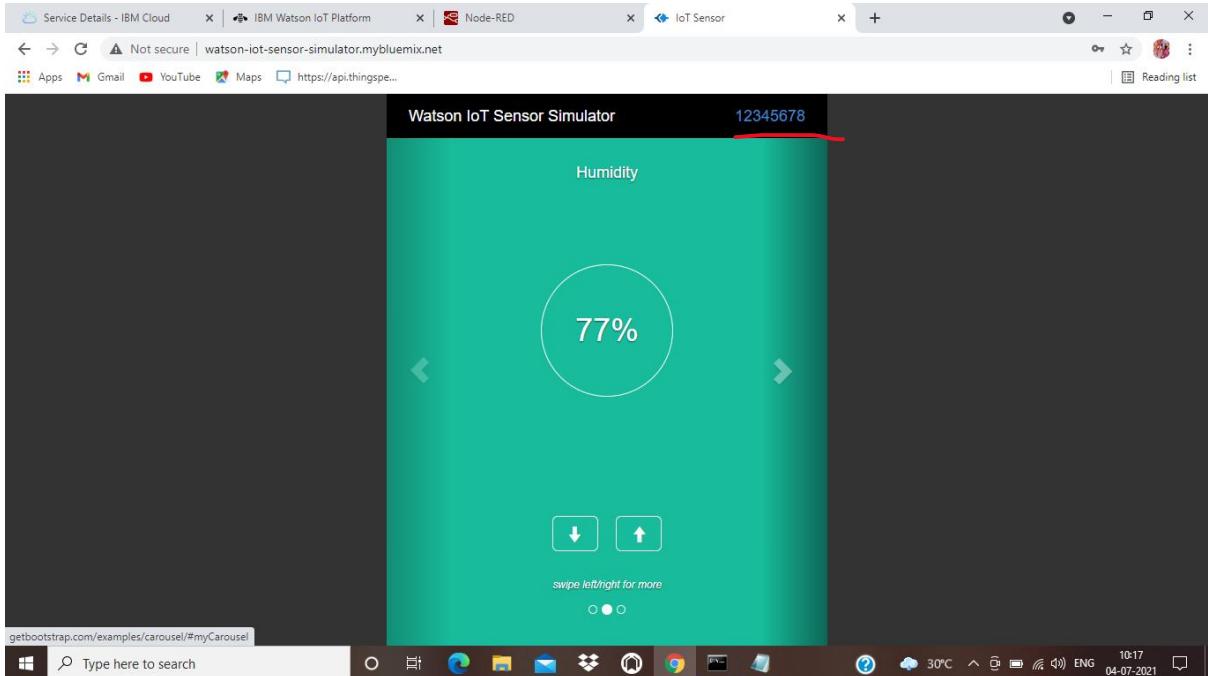


Figure-16

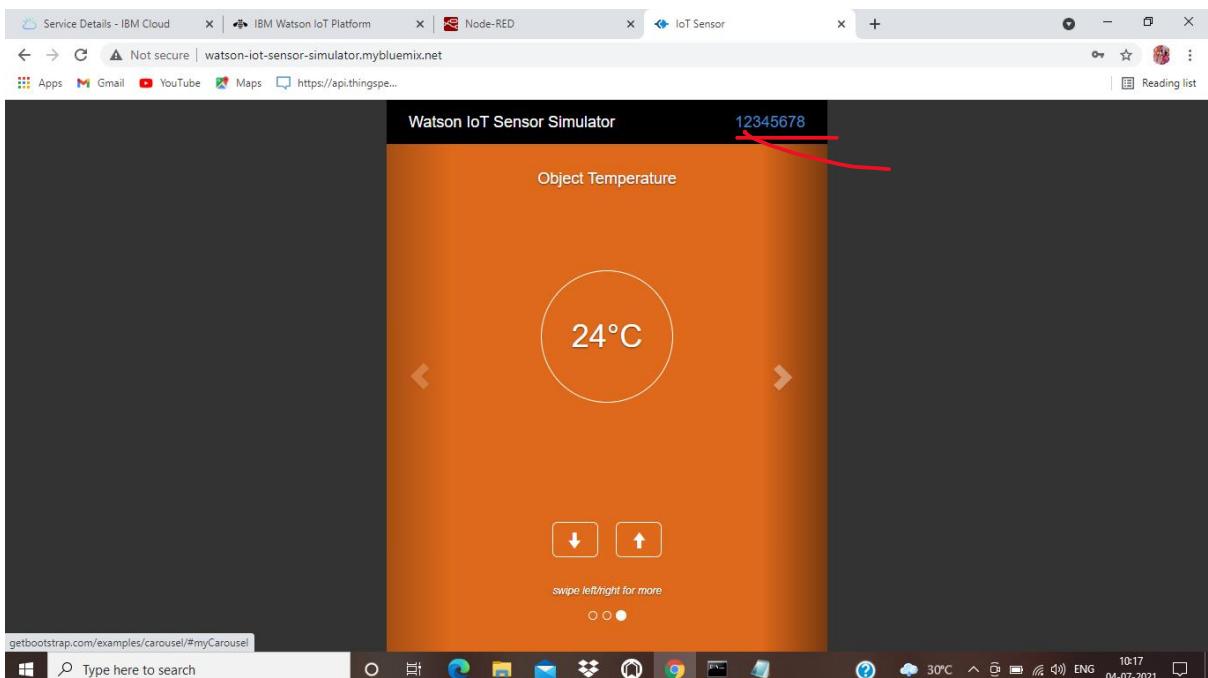


Figure-17

we can see the sensor displaying our device ID at the top most corner and three different quantities **temperature, humidity and object temperature** as shown in figures-15,16 and17. We can vary these three quantities by the arrow marks displayed below if we click on down arrowmark

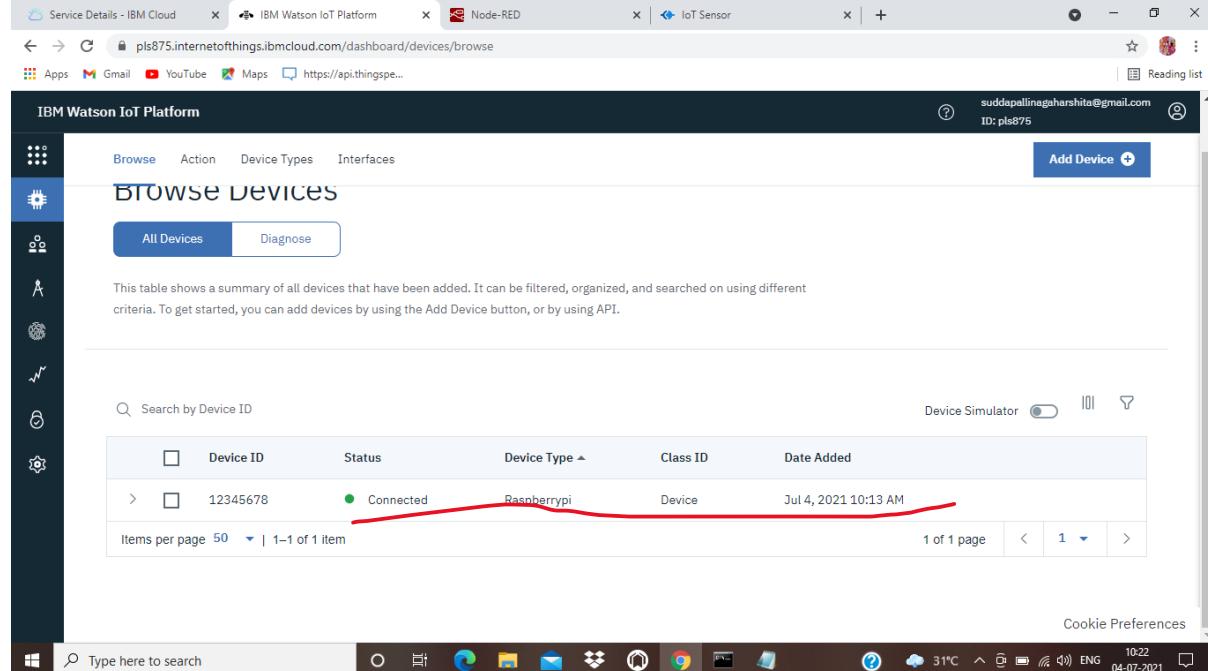


SMART AGRICULTURE

The quantity will be decreased and if we click on upside arrowmark 

The quantity will get increased.

16. status of the sensor

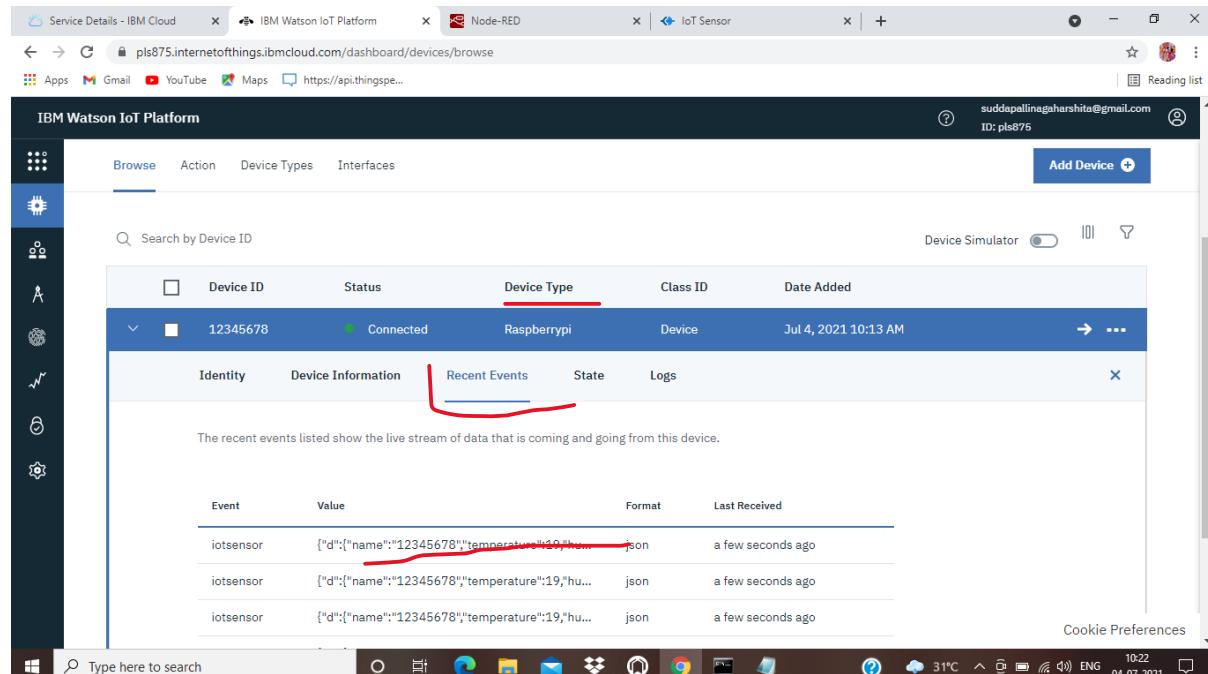


This screenshot shows the 'Browse Devices' page in the IBM Watson IoT Platform. The device list table includes columns for Device ID, Status, Device Type, Class ID, and Date Added. One row is visible, showing a Device ID of 12345678, a Connected status, a Device Type of RaspberryPi, a Class ID of Device, and a Date Added of Jul 4, 2021 10:13 AM. A red arrow points to the 'Connected' status indicator.

Figure-18

And later if we again go to our IBM page we can see the status of our sensor whether it is connected or not. In figure-18 it is showing that our device is connected.

17. Data from the sensor



This screenshot shows the detailed view for the device with ID 12345678. The 'Recent Events' tab is selected, showing a table of recent sensor data. The table has columns for Event, Value, Format, and Last Received. Three entries are listed, all showing the same JSON value for 'iotSensor'. A red arrow points to the 'Recent Events' tab header.

Event	Value	Format	Last Received
iotSensor	{"d": {"name": "12345678", "temperature": 18, "humidity": 65}}	json	a few seconds ago
iotSensor	{"d": {"name": "12345678", "temperature": 19, "humidity": 66}}	json	a few seconds ago
iotSensor	{"d": {"name": "12345678", "temperature": 19, "humidity": 67}}	json	a few seconds ago

SMART AGRICULTURE

Figure-19

And later if we go to device type and then recent events(figure-19) we can see the data that is sending by our sensor in json format.

18.json format

The screenshot shows a browser window with several tabs open: Service Details - IBM Cloud, IBM Watson IoT Platform, Node-RED, and IoT Sensor. The main content area is titled "Event Payload" for a device named "iotsensor". The event was received on July 4, 2021, at 10:22 AM. The JSON payload is displayed in a code editor:

```
1  {  
2   "d": {  
3     "name": "12345678",  
4     "temperature": 19,  
5     "humidity": 77,  
6     "objectTemp": 24  
7   }  
8 }
```

Below the payload, there is a table showing recent events:

Event	Value	Type	Time
iotsensor	{"d": {	json	a few seconds ago
iotsensor	{"d": {	json	
iotsensor	{"d": {"name": "12345678", "temperature": 19, "hu...}}	json	

figure-20

in figure-20 we can see the json format displaying Temperature as 19, Humidity as 77% and object Temperature as 24.

SMART AGRICULTURE

19. selecting apps to connect with the sensor

The screenshot shows the IBM Watson IoT Platform dashboard. The left sidebar has icons for Boards, Devices, Members, Apps, Access Management, Usage, Security, and Settings. The main area is titled 'IBM Watson IoT Platform' and shows a table for 'Devices'. One device, 'IoT Sensor', is listed with the following details:

Device ID	Status	Device Type	Class ID	Date Added
12345678	Connected	RaspberryPi	Device	Jul 4, 2021 10:13 AM

Below the device table, there are tabs for Identity, Device Information, Recent Events, State, and Logs. The 'Recent Events' tab is active, displaying three entries:

Event	Value	Format	Last Received
iotsensor	{"d": {"name": "12345678", "temperature": 19, "humidity": 45}}	json	a few seconds ago
iotsensor	{"d": {"name": "12345678", "temperature": 19, "humidity": 45}}	json	a few seconds ago
iotsensor	{"d": {"name": "12345678", "temperature": 19, "humidity": 45}}	json	a few seconds ago

Figure-21

Now we have to connect it into Node-red, for that on the top right corner we may find different options , we choose apps as shown in figure-21.

20. Generating API key

The screenshot shows the 'API Keys' section of the IBM Watson IoT Platform. The left sidebar has icons for Boards, Devices, Members, Apps, Access Management, Usage, Security, and Settings. The main area is titled 'IBM Watson IoT Platform' and shows a table for 'API Keys'. The table header includes columns for Key, Description, Role, and Expires. A red arrow points to the 'Generate API Key' button at the top right of the table area. Below the table, a message states 'There are no API Keys'.

Figure-22

Then the page shown in figure-22 will appear . To connect our device into apps we have to generate an API KEY. To generate API key click on generate API Key on the top right of the page.

SMART AGRICULTURE

21.specifying application type

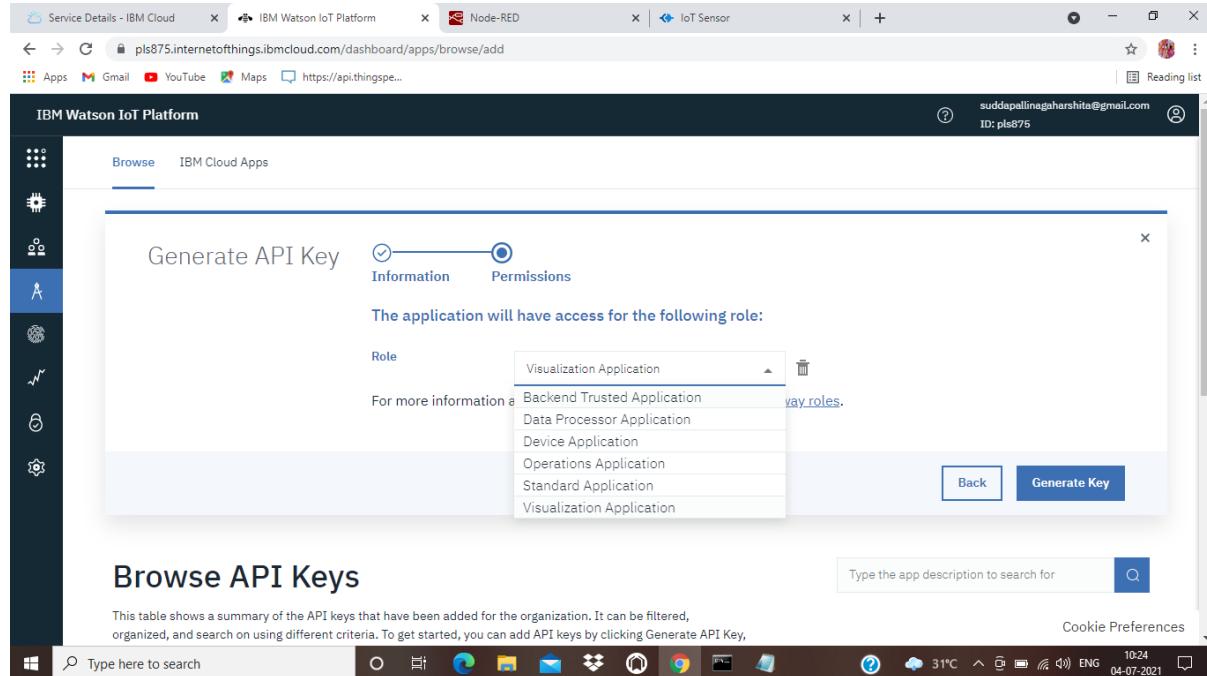


Figure-23

After choosing “generate API Key” we will see a page in figure-23. Here we have to specify our application type , we will choose ‘standard application’.

22.API key

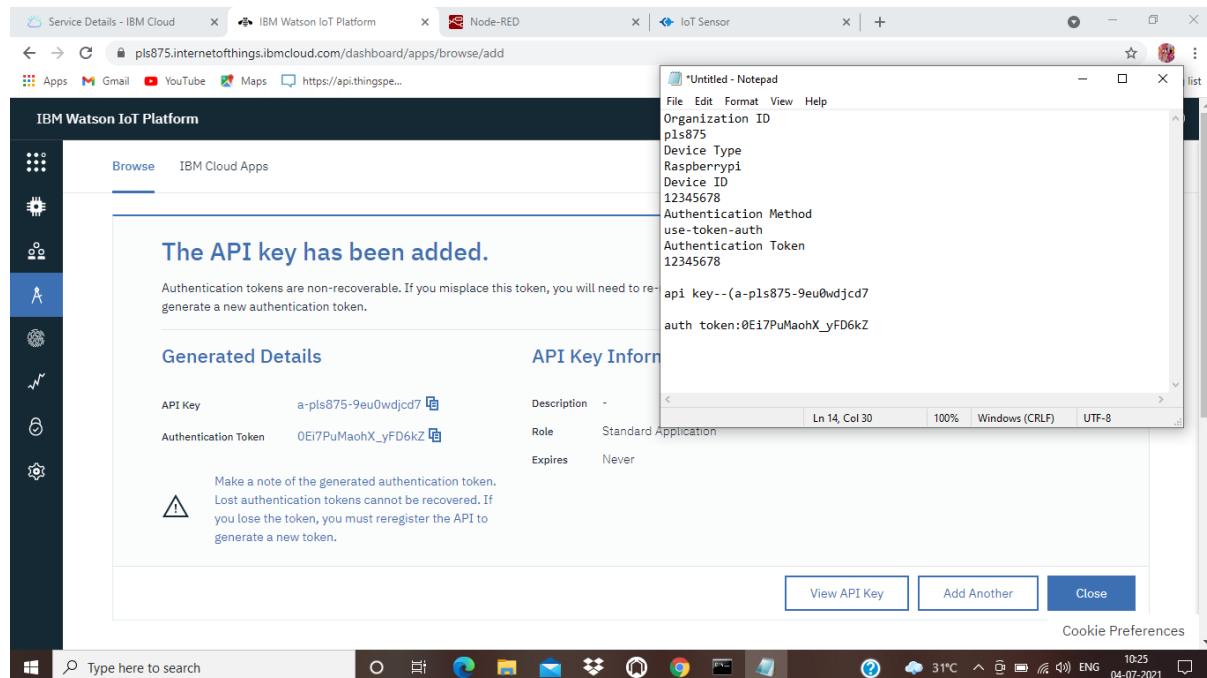


Figure-24

Then , the system will generate an API key and an authentication token as shown in figure -24 , just as before, we need to save these details also to connect our device into node-red.

SMART AGRICULTURE

23.starting Node-red flow with IBM IoT in

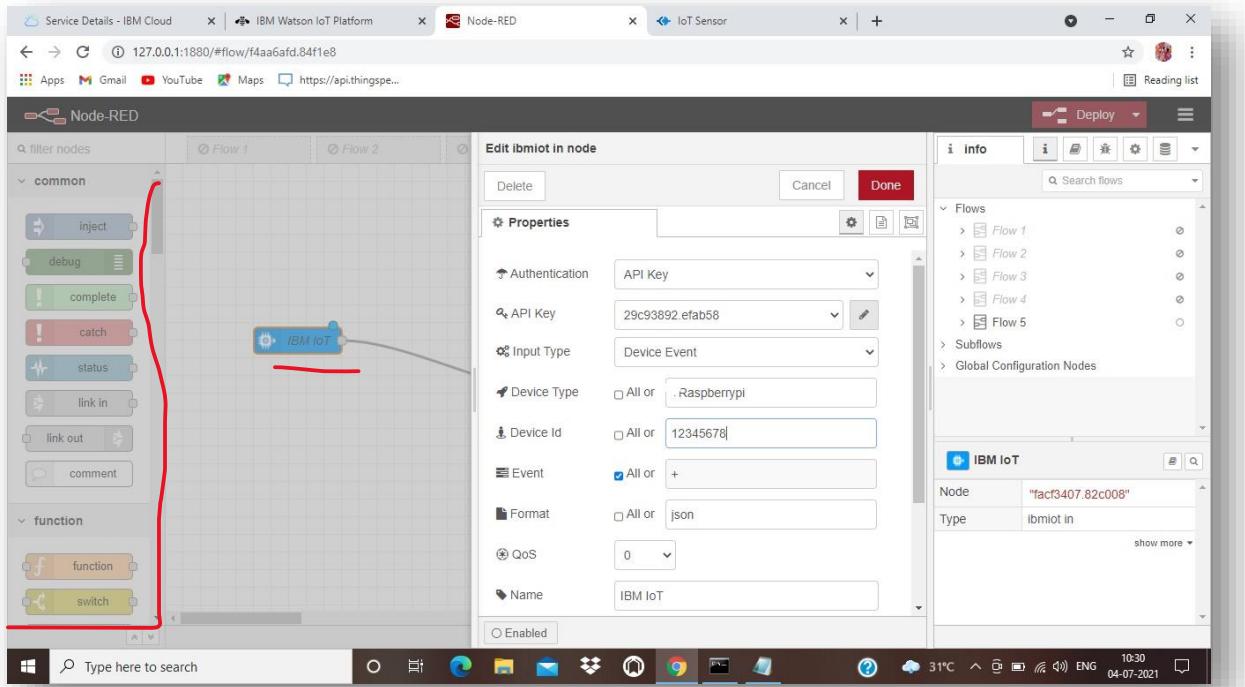


Figure-25

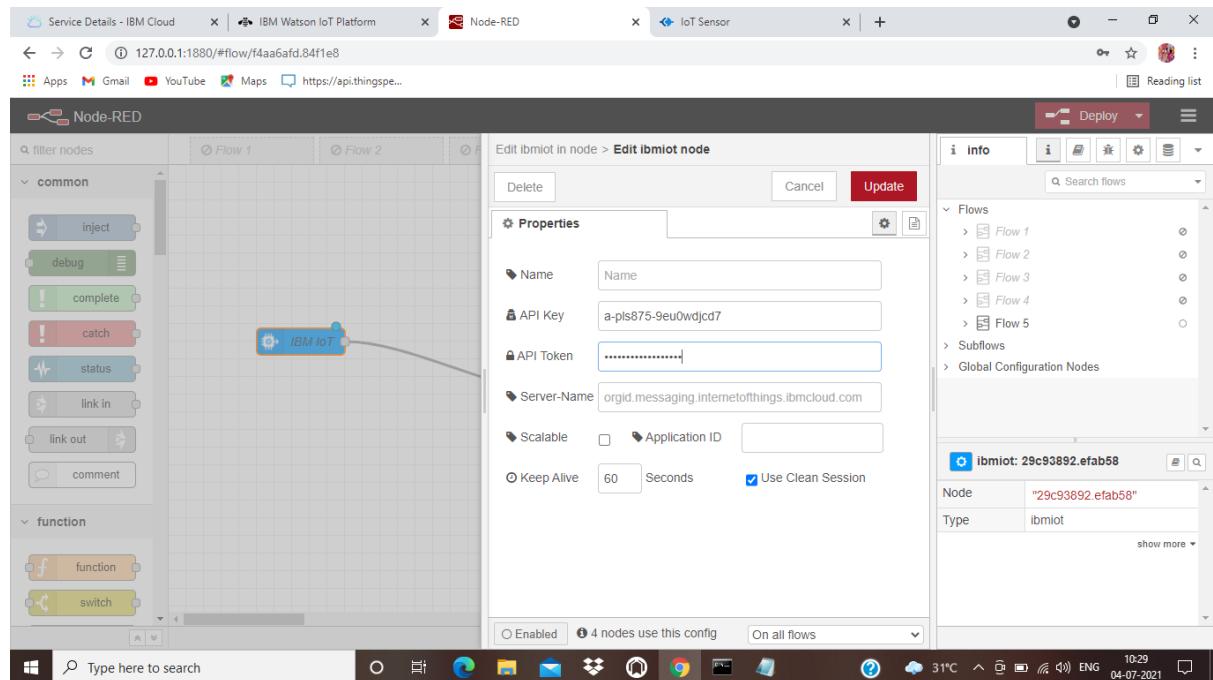


figure-26

Then after coming back to Node-red again we have to select “IBM IoT in” from palette available on left side . to bring this ibmiot in we have to install this into our node-red before. We can install this from node-red’s settings. After double-click on this button and it will show properties , we have to

SMART AGRICULTURE

give details of our device such as authentication, device type ,input type as shown in figure-25.
Give the API Key and token and click on update(figure-26) and then done(figure-25).

24.connecting IBM IoT in with debug node

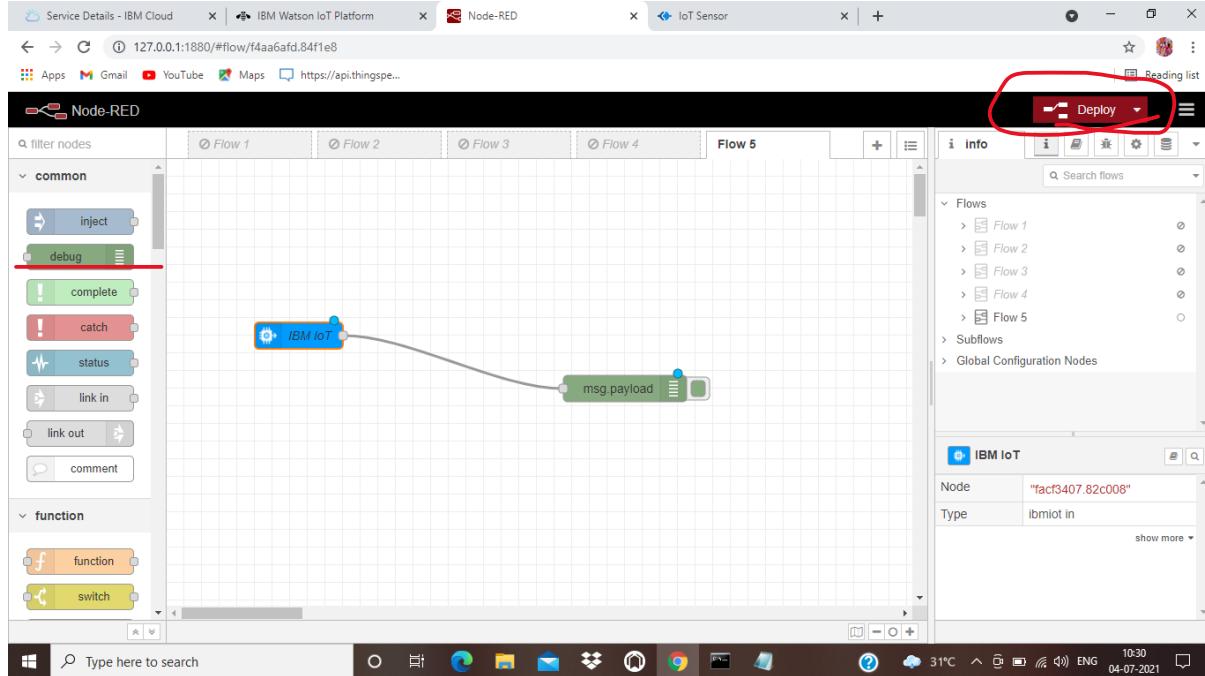


Figure-27

And then bring debug node from nodes and connect it to IBM IoT node as shown In figure -27 and then click deploy on the top right of the page.

25.output from IBM sensor

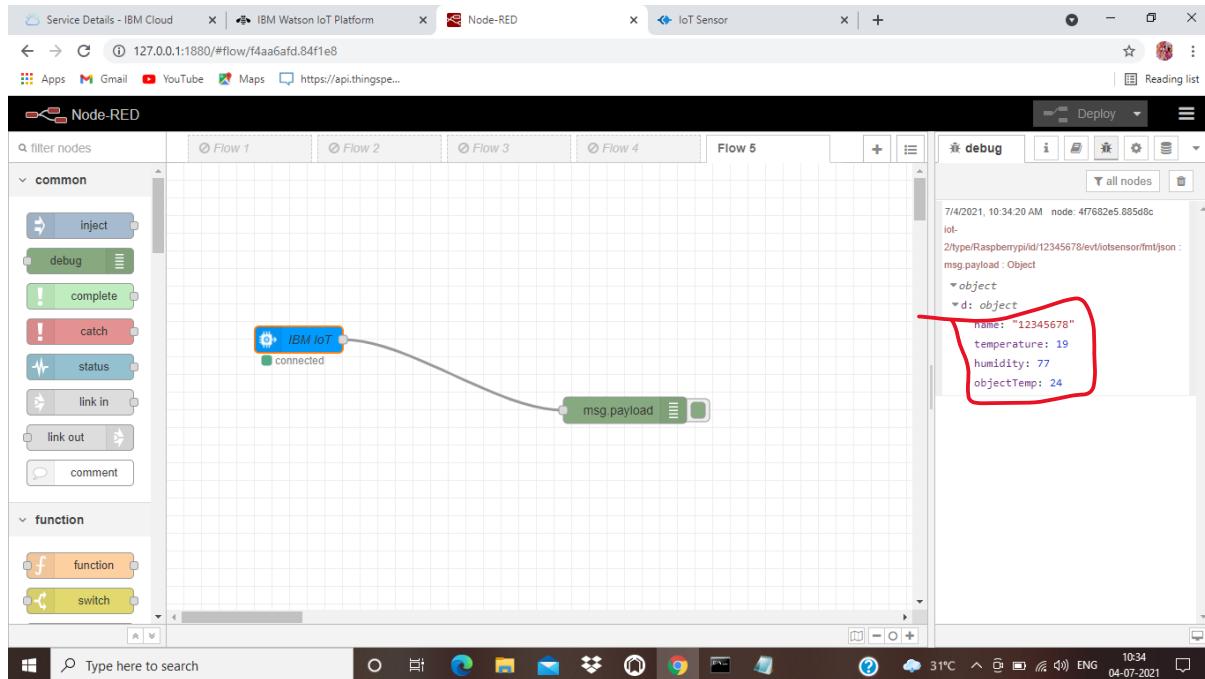


Figure-28

SMART AGRICULTURE

Here we can see output displaying the quantities of our IBM sensor along with device ID (figure-28).

26.getting data using a function

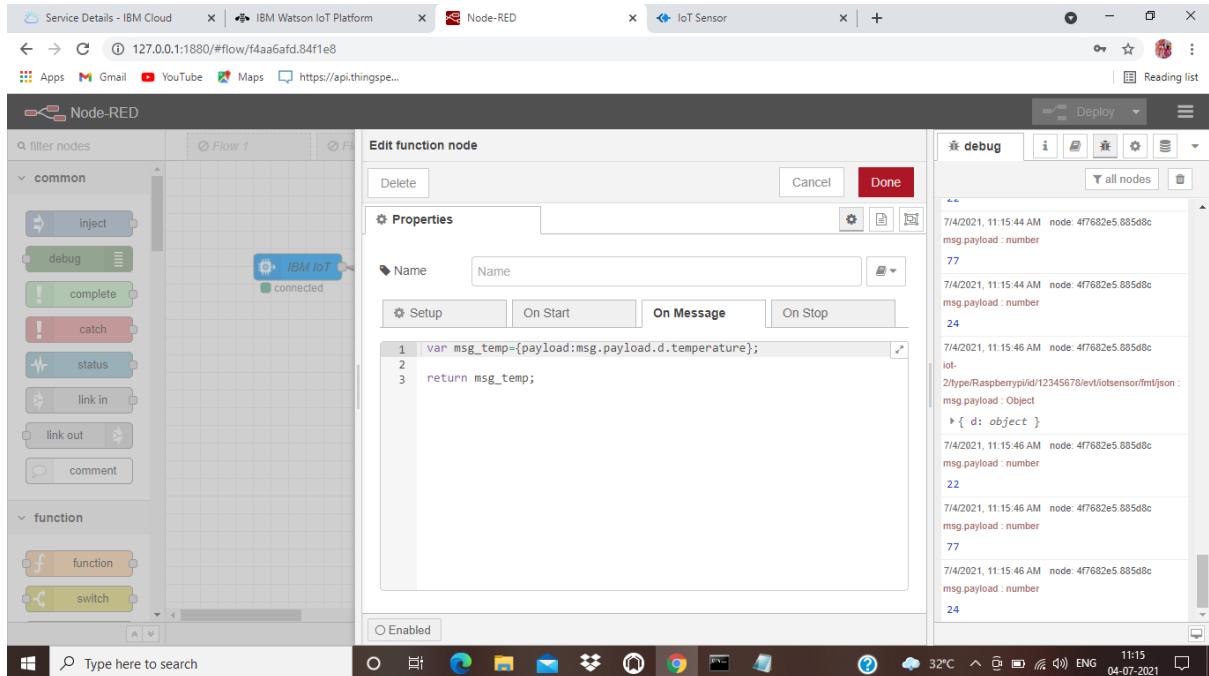


Figure-29

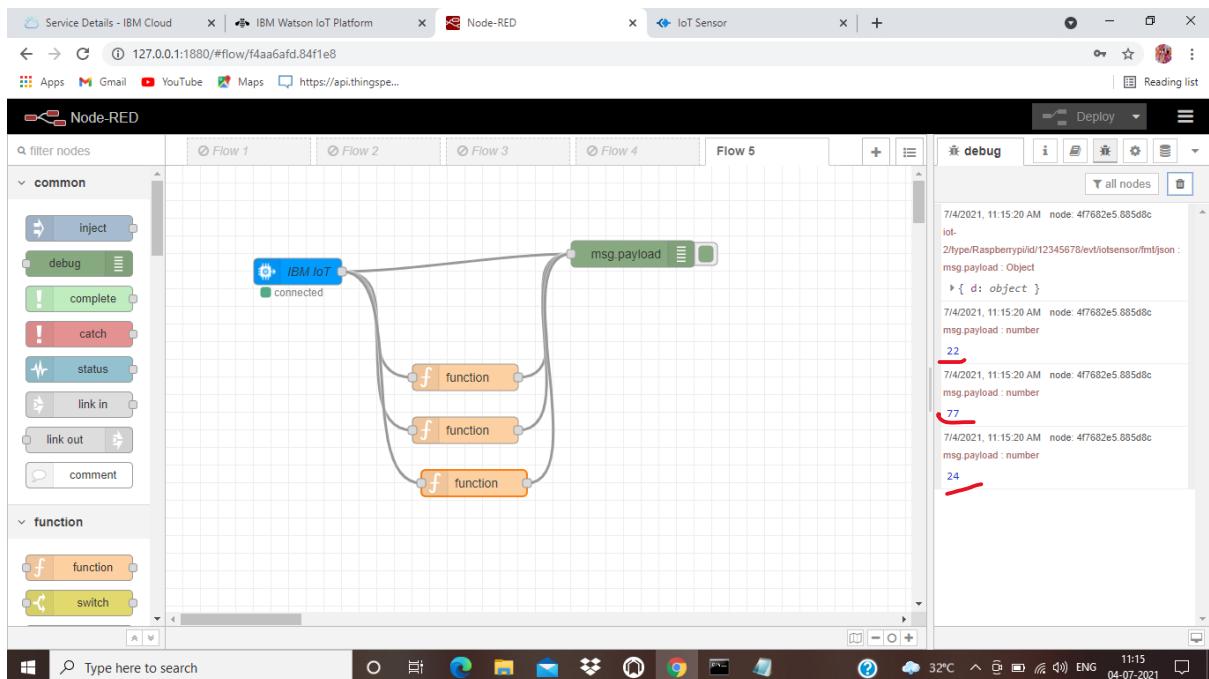


Figure-30

To display the data from sensor separately that is temperature, humidity and object temperature as single entities , we will take three different functions each for three quantities. A function displays specified quantity in the code separately. And we have to configure these functions as shown in figure-29.

SMART AGRICULTURE

for humidity and object temperature we will take two different variables and in place pf temperature we will change it as "humidity and object Temperature".)
in figure -30 we can see the data from sensor separately.

27.Dashboard node

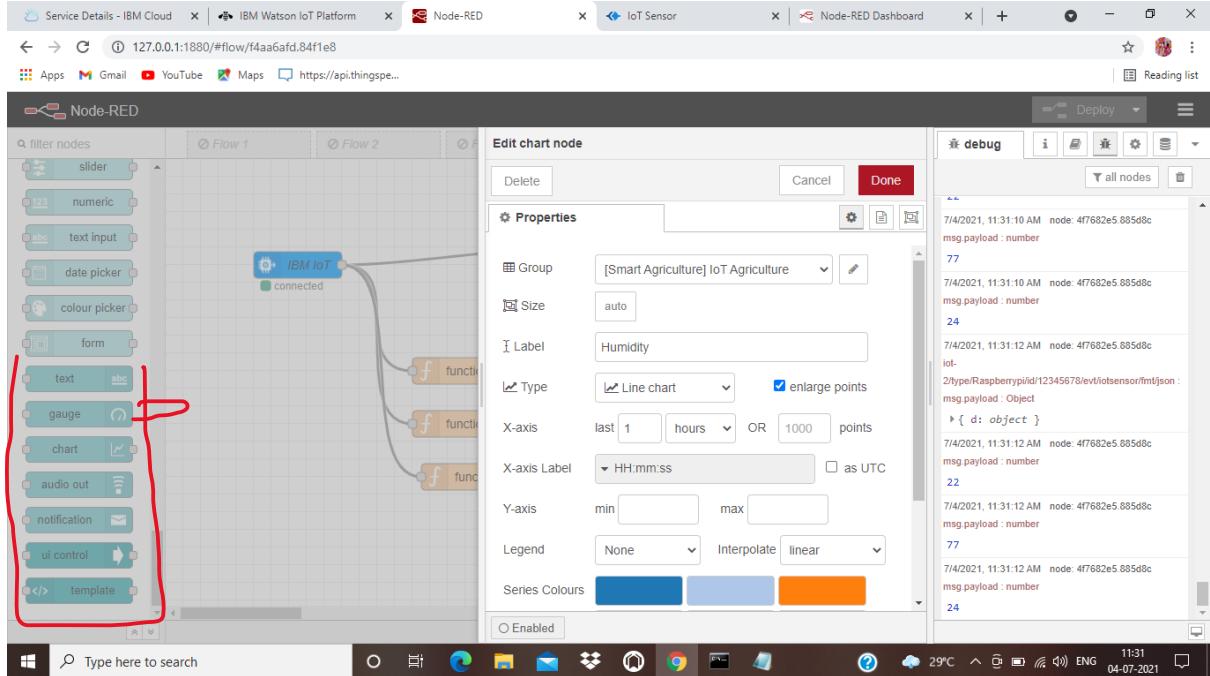


Figure-31

To display this data in the dashboard we need to bring some dashboard nodes such as gauge and chart. After bringing the node into our flow double click on the gauge for temperature and object temperature and chart for humidity. Give details name, tab name, label, units etc, as shown in figure-31. Later click on done and then deploy.

SMART AGRICULTURE

28.Node-red dashboard

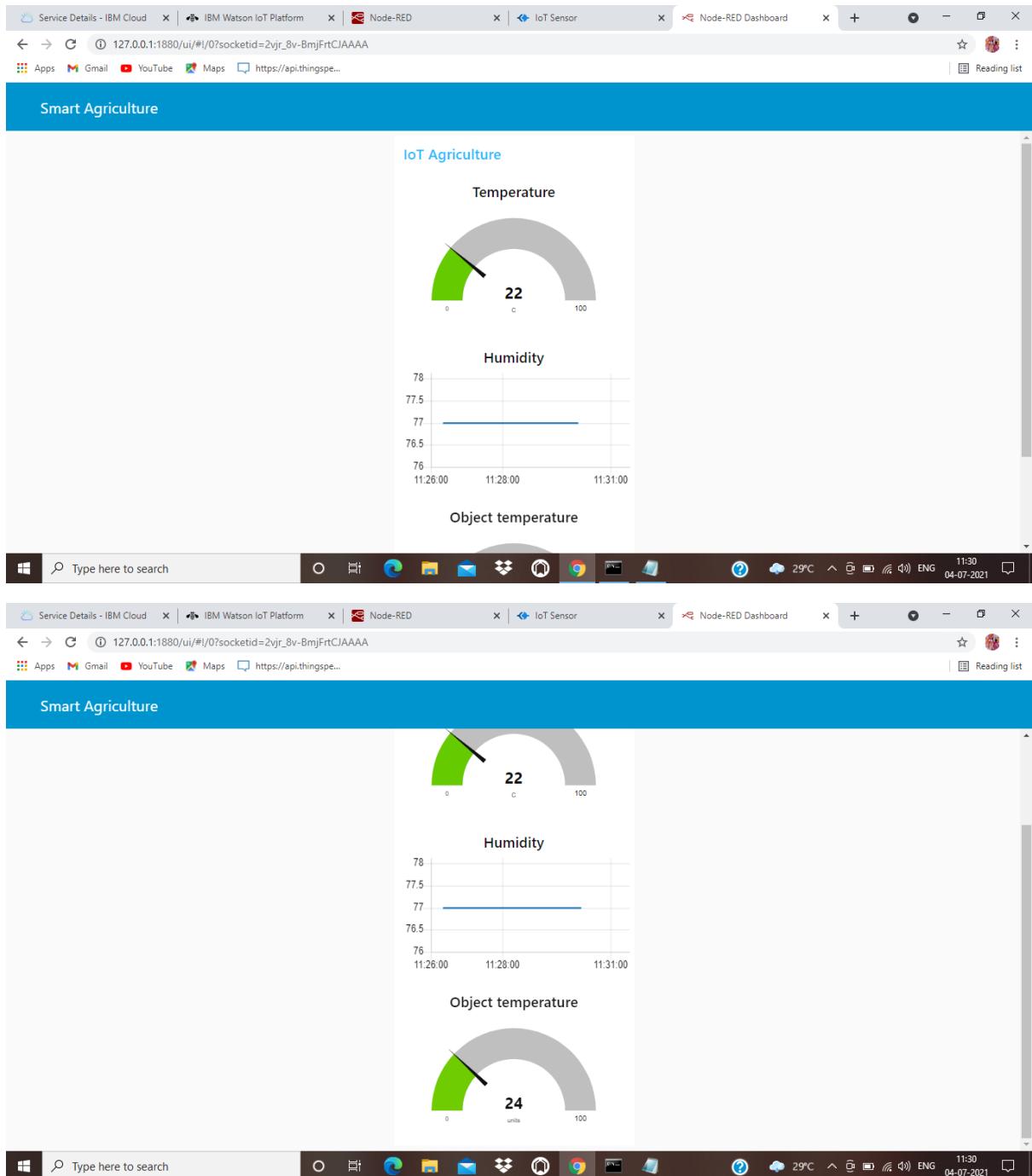


Figure-32

To view data on Node-red dashboard, we have to go to new tab and search for (<http://127.0.0.1:1880/ui>) ,then our data is displayed as shown in figure-32.

Up to now we have seen sensor's temperature, humidity and object temperature which can be controlled by ourselves also.(**changing data whenever we changed data in sensor is shown in figures-53&54**) But we need to access the real time temperature in order to improve our agriculture system.

SMART AGRICULTURE

29.open weather

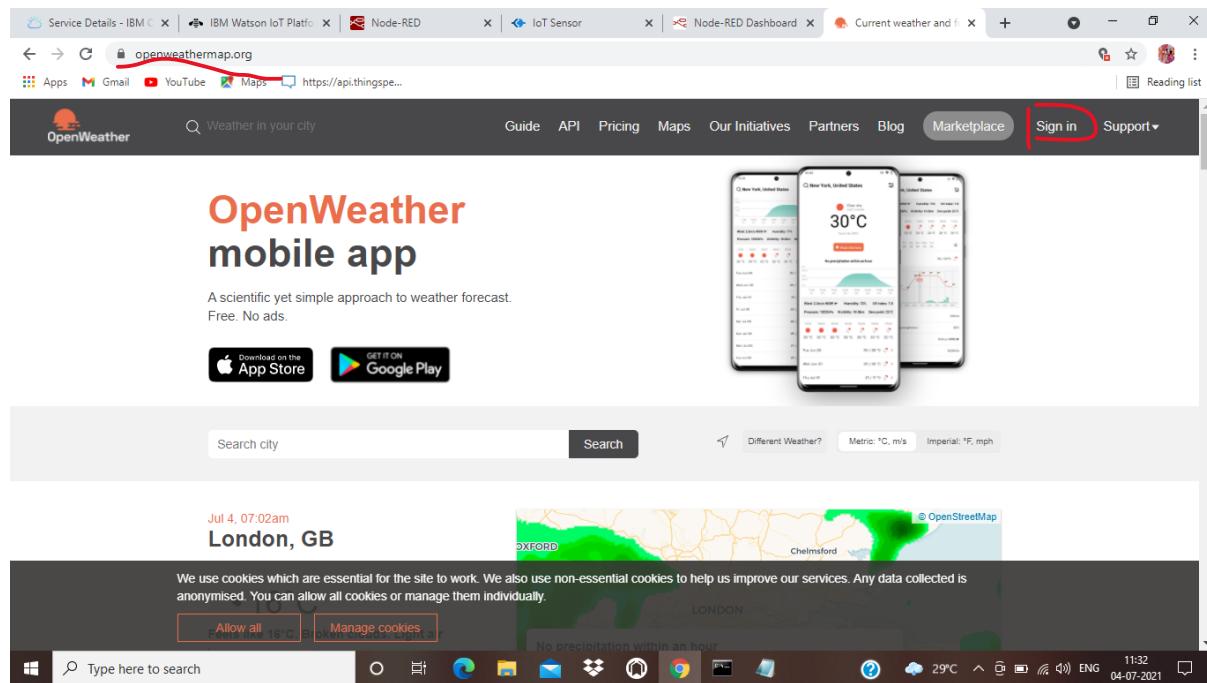


Figure-33

We can access current weather from internet.

For that we will search for openweatherapi in google → and then go to the first link that will appear (open weather map: current weather and forecast) . then the page shown in figure-33 will appear.

After we have to create an account in this website to access the data ,so click on the sign in button that will appear on the top right corner of this page. Then it will ask details and provide them so that your account will be created . them it will ask the company and purpose , give the required details and save them.

SMART AGRICULTURE

30.Open weather API

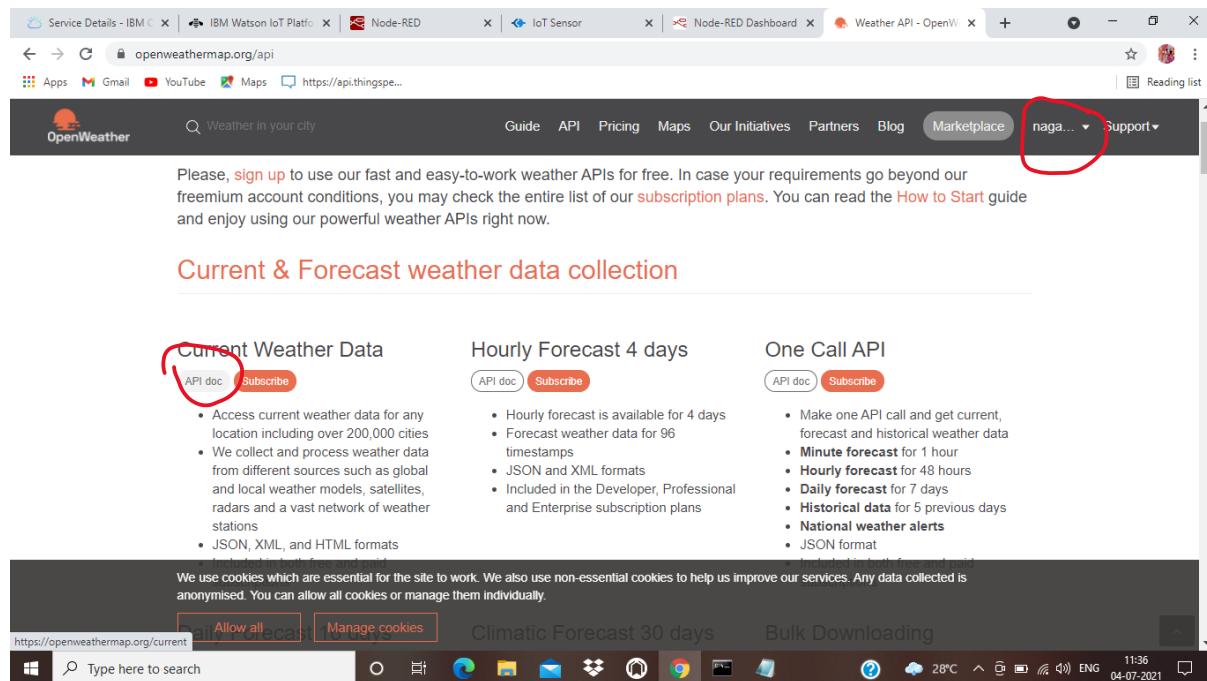


Figure-34

After that the page showing marketplace details will appear , click on API on the left side to it. Then we can see figure-34 there. Click on API doc under current weather data as marked in above figure.

31.API link

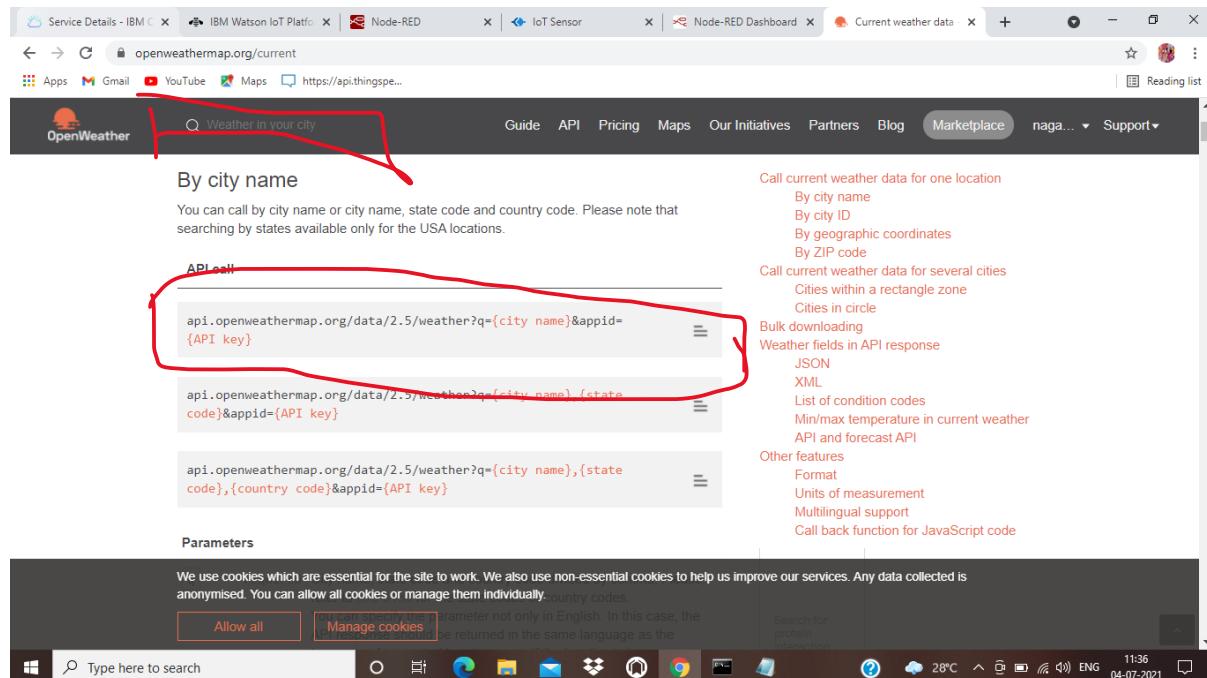


Figure-35

Then we will get page shown in figure-35. We have given a link (marked in figure-35) to get weather data in which we need to enter city name and API key. We need to note this link to use in Node-red.

SMART AGRICULTURE

Then we have to search for the city we want and have to replace as it is in place of city name in the link .

32.API Key

The screenshot shows a web browser window with multiple tabs open. The active tab is 'home.openweathermap.org/api_keys'. The page displays a table of API keys. One key is highlighted with a red underline, showing the value '0f3dfe54aecbb4eab518345a4bd448e9'. To the right of the table is a 'Create key' section with a 'Generate' button. Below the table, there's a sidebar with links for 'Product Collections', 'Subscription', and 'About us'. The status bar at the bottom shows system information like weather (28°C), date (04-07-2021), and time (11:44).

Figure-36

We need our API key also , for that we need to click on our username and select “my API Key” ,then the page shown in figure-36 will appear. We need to note the API Key provided there. And enter the same URL in the link in place of API Key.

SMART AGRICULTURE

33.final API link for a city

iot - Notepad

File Edit Format View Help

Organization ID
p1s875

Device Type
RaspberryPi

Device ID
12345678

Authentication Method
use-token-auth

Authentication Token
12345678

api key--(a-p1s875-9eu0wdjcd

auth token:0Ei7PuMaohX_yFD6kZ

api.openweathermap.org/data/2.5/weather?q=Hyderabad,%20IN&appid=0f3dfe54aecbb4eab518345a4bd448e9

Windows taskbar: Type here to search, File Explorer, Edge, Mail, File History, Task View, Word, Excel, 28°C, 18:29, 13-07-2021

Figure-37

Our final link is like this as shown in figure-37.

After that we need to enter the link on new tab ,then we can see the weather data in json format as shown in figure-38.

34.data in json format

Service Details - | IBM Watson IoT | Node-RED | IoT Sensor | Node-RED Dashboard | Members | https://api.openweathermap.org/data/2.5/weather?q=Hyderabad,%20IN&appid=0f3dfe54aecbb4eab518345a4bd448e9 | + | - | X | Reading list

Apps Gmail YouTube Maps https://api.thingspe...

{"coord":{"lon":78.4744,"lat":17.3753}, "weather":[{"id":802,"main":"Clouds","description":"scattered clouds","icon":"03d"}], "base":"stations", "main": {"temp":301.44,"feels_like":305,"temp_min":301.38,"temp_max":304.88,"pressure":1011,"humidity":74}, "visibility":6000, "wind": {"speed":4.12,"deg":300}, "clouds": {"all":40}, "dt":1625379230, "sys": {"type":1,"id":9214,"country":"IN","sunrise":1625357769,"sunset":1625405069}, "timezone":19800, "id":1269843, "name":"Hyderabad", "cod":200}

Windows taskbar: Type here to search, File Explorer, Edge, Mail, File History, Task View, Word, Excel, 28°C, 11:49, 04-07-2021

Figure-38

SMART AGRICULTURE

35. Accessing data from API link

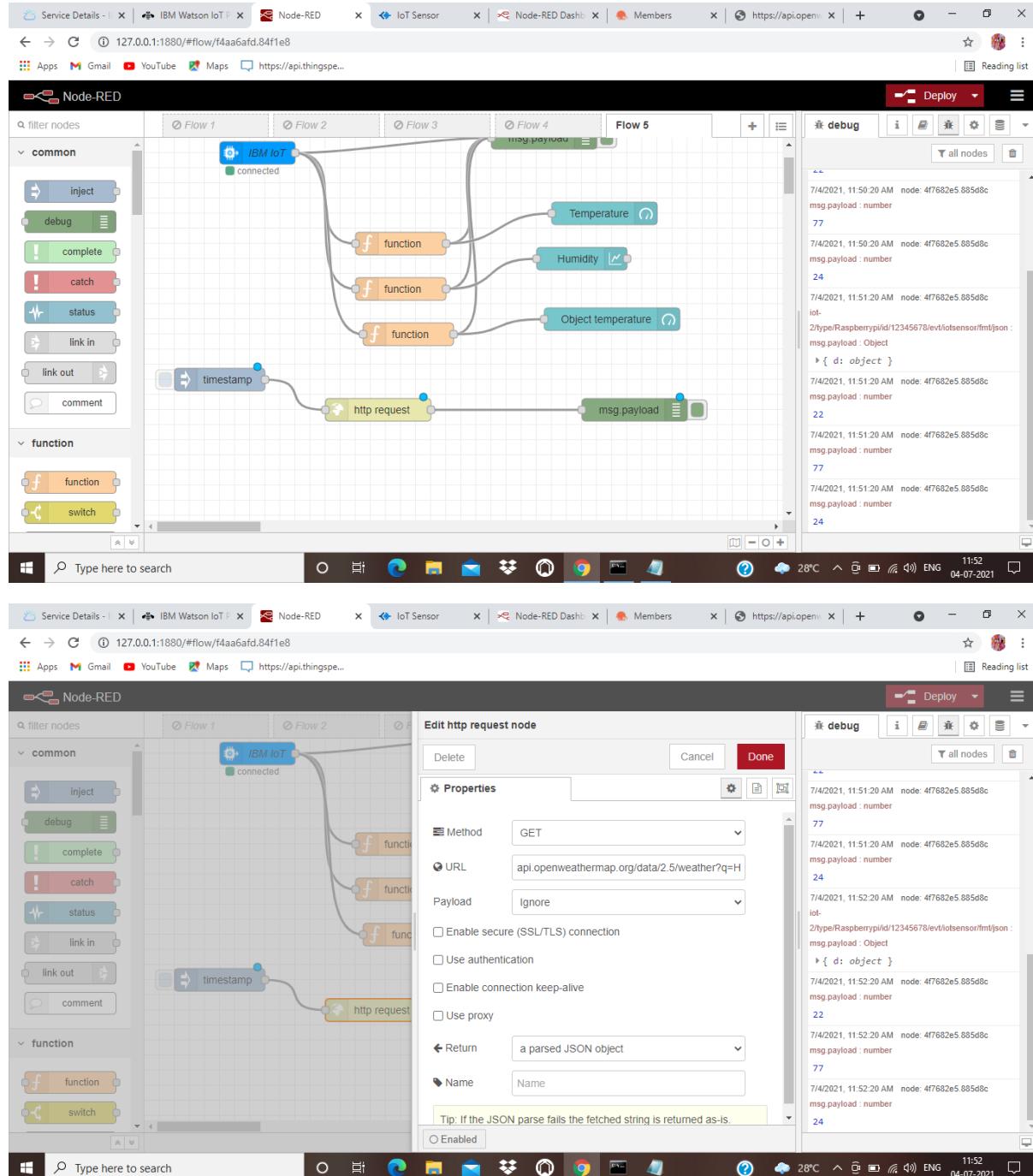


Figure-39

Now getting back to our Node-red we need to bring inject node and http request to access the data from open weather, and a debug node to display output. After that double click on the http request and give the URL that we have shown in figure-38. If we deploy flow , we can see the data in same json format (marked with red in figure-40) . It includes a lots of things such as humidity, clouds , windspeed etc; (. we need only clouds and windspeed for that we need to include a function, which will display the data that is only required.

SMART AGRICULTURE

36. including function

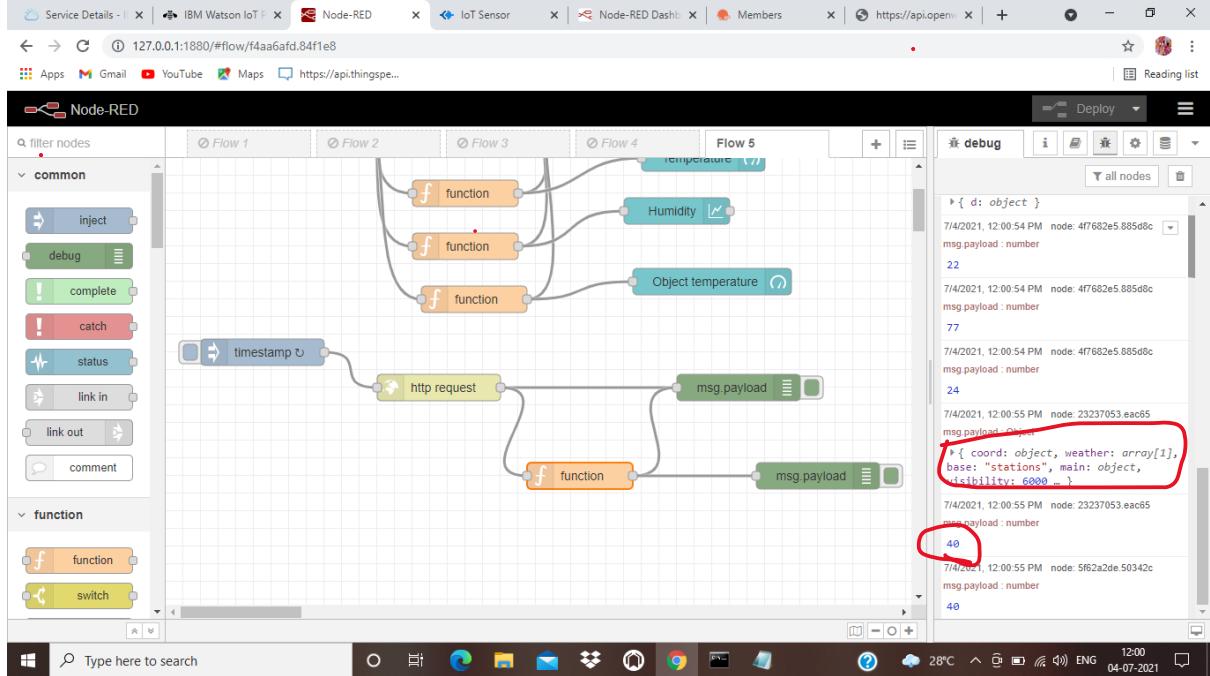


Figure-40

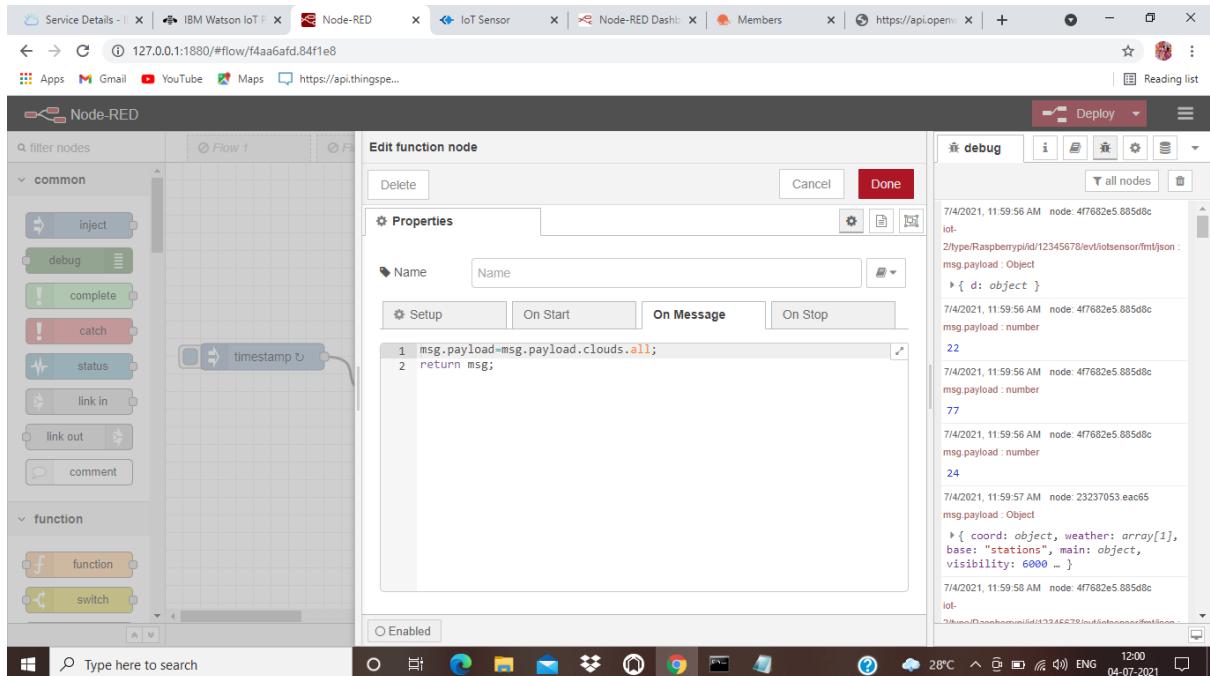


Figure-41

In side the function we need to write a code as shown in figure -41, And to display the data we will use a debug node again. After deploying the flow we may see cloud data separately (marked with blank in figure-40).

SMART AGRICULTURE

37. displaying cloud data on dashboard

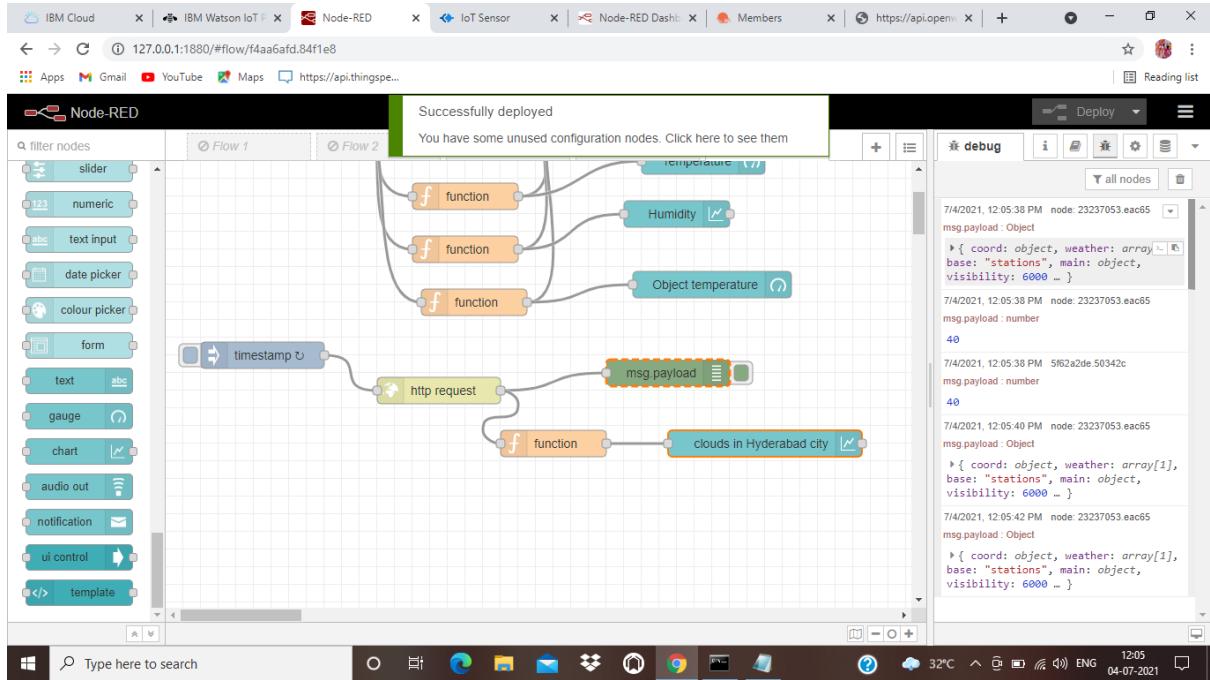


Figure-42

Now we have to display the same data in dashboard , for that we include a chart(we may include anything) from nodes list. Double click on the node to change tab name , label etc. and then deploy the flow.(same as temperature node configuration)

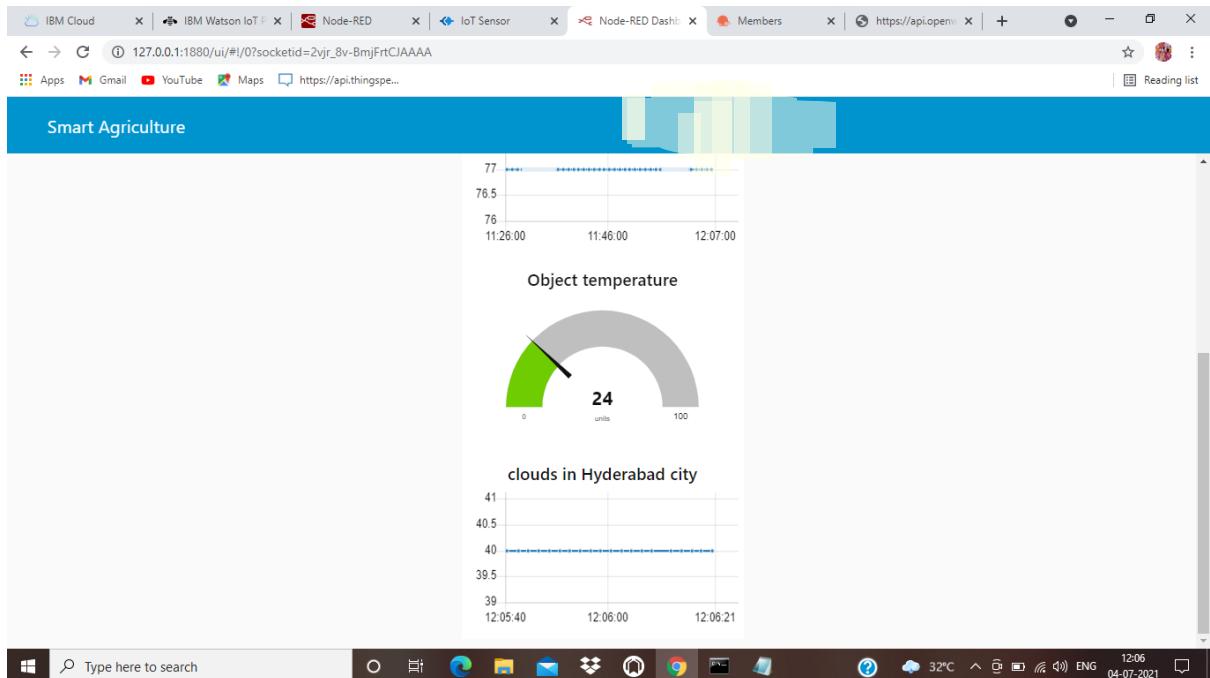


Figure -43

SMART AGRICULTURE

After deploying flow , in our node -red dashboard we may see the cloud data in the city we want as shown figure-43.

38.windspeed data on dashboard

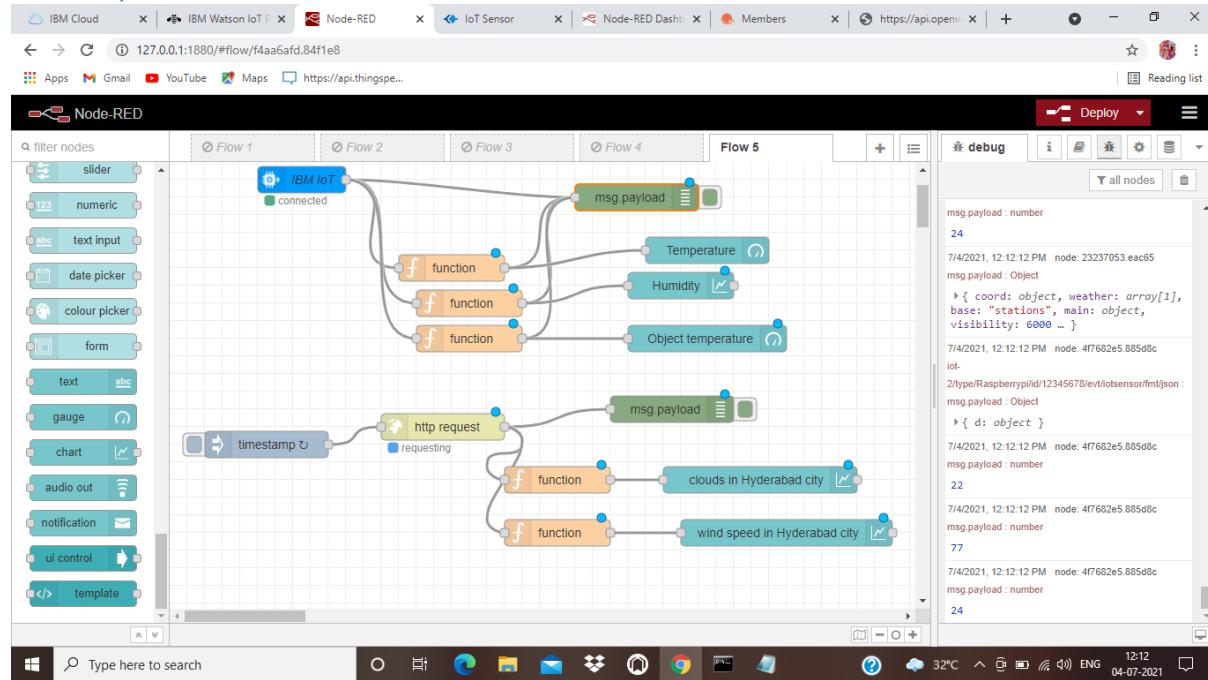


Figure-44

Similarly, for windspeed also we will bring another function(**change “windspeed” in place of “clouds.all”**) and a dashboard node and the same process will be followed for windspeed as clouds. later , we will deploy the flow.

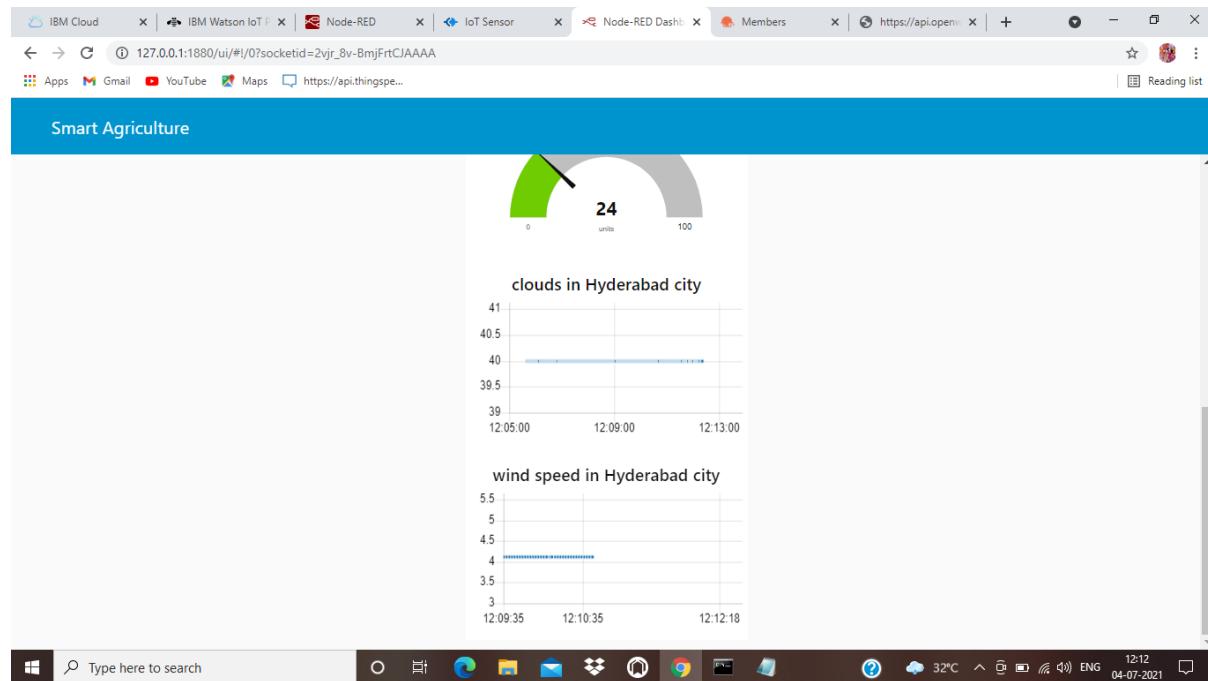


Figure-45

We can see windspeed data in the dashboard as shown in figure-45.

SMART AGRICULTURE

39. python code for motor button configurations and as well as to display data from IoT sensor

The screenshot shows the PyCharm IDE interface with the file `IBM.py` open. The code is a Python script using the `ibmiotf` library to configure a device. It defines variables for organization, device type, device ID, authentication method, and token. It includes a command callback function to handle 'motoron' and 'motoroff' commands. It then attempts to connect to the device using the defined options. If an exception occurs, it prints the error message and exits.

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

organization = "pls875"
deviceType = "RaspberryPi"
deviceId = "12345678"
authMethod = "token"
authToken = "12345678"

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)
    if cmd.data['command'] == 'motoron':
        print("MOTOR ON")
    elif cmd.data['command'] == 'motoroff':
        print("MOTOR OFF")

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))
```

Figure-46

The screenshot shows the PyCharm IDE interface with the file `IBM.py` open. The code is similar to Figure-46 but includes additional logic. After connecting to the device, it enters a loop where it publishes temperature and humidity data to IBM Watson every 2 seconds. It uses a command callback function to handle 'motoron' and 'motoroff' commands. Finally, it disconnects from the device.

```
deviceCli = ibmiotf.device.Client(deviceOptions)

except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()

deviceCli.connect()

while True:
    T = 22
    H = 77
    data = {'Temperature': T, 'Humidity': H}

    def myOnPublishCallback():
        print("Published Temperature = %s C" % T, "Humidity = %s %%" % H, "to IBM Watson")
        success = deviceCli.publishEvent("event", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(1)

    deviceCli.commandCallback = myCommandCallback

deviceCli.disconnect()
```

Figure-47

SMART AGRICULTURE

40. connecting buttons to IBM out

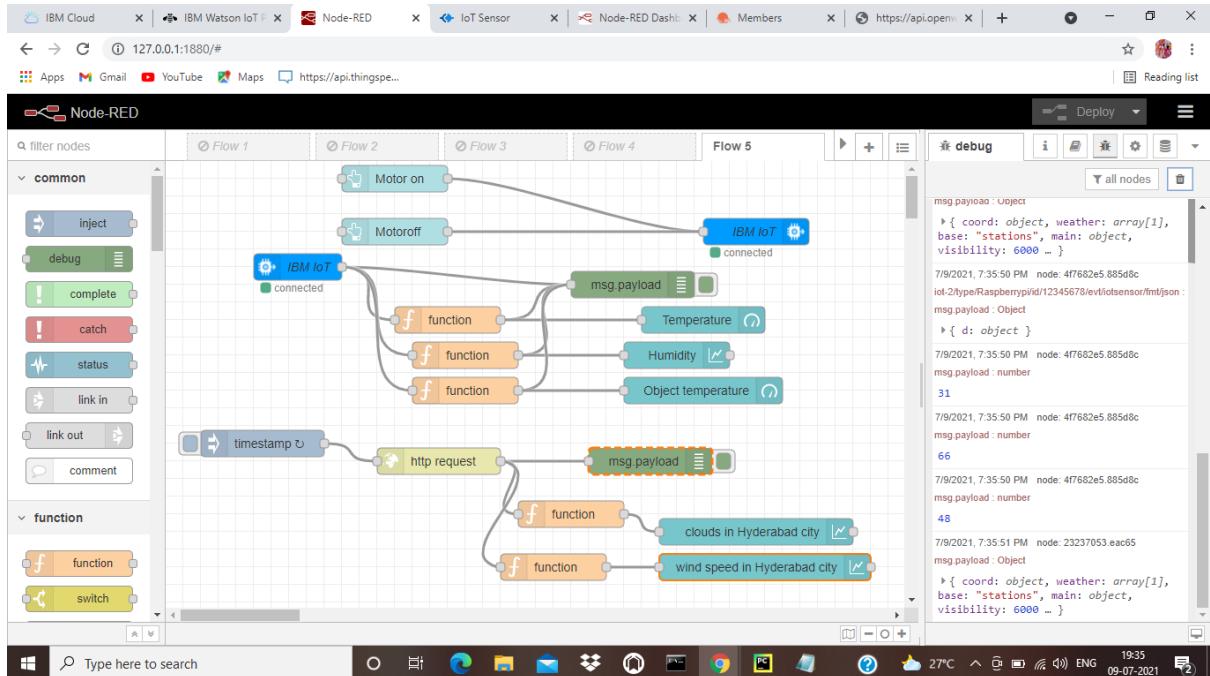


Figure-48

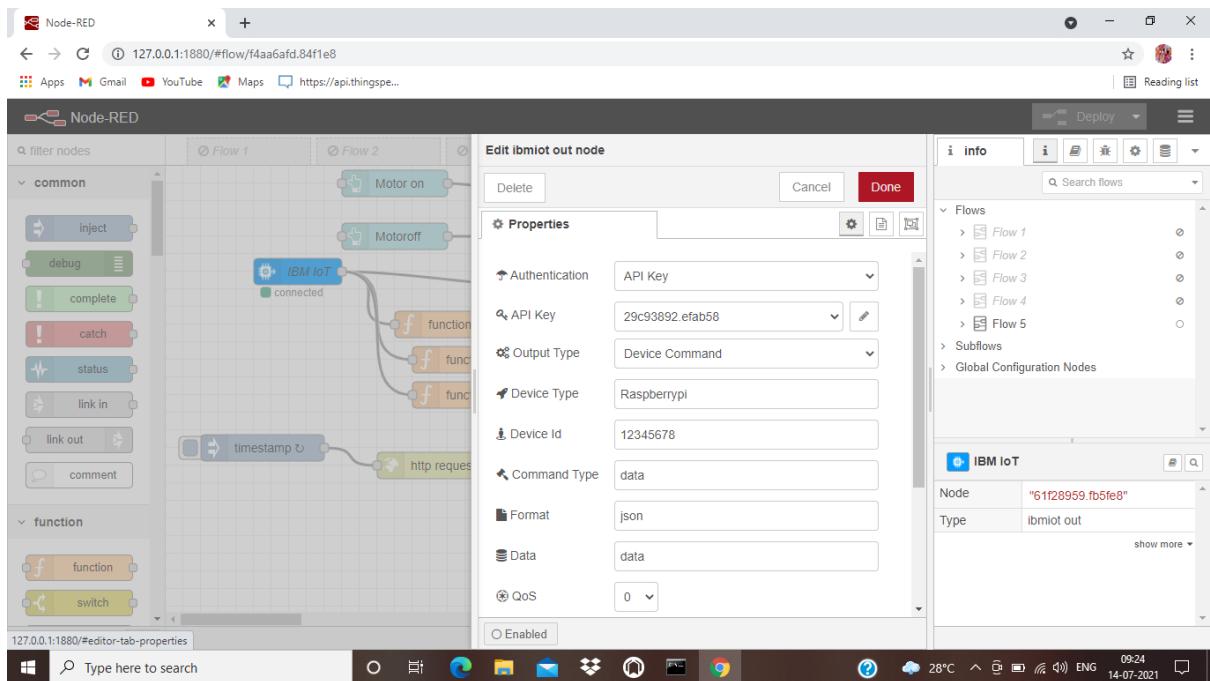


Figure-49

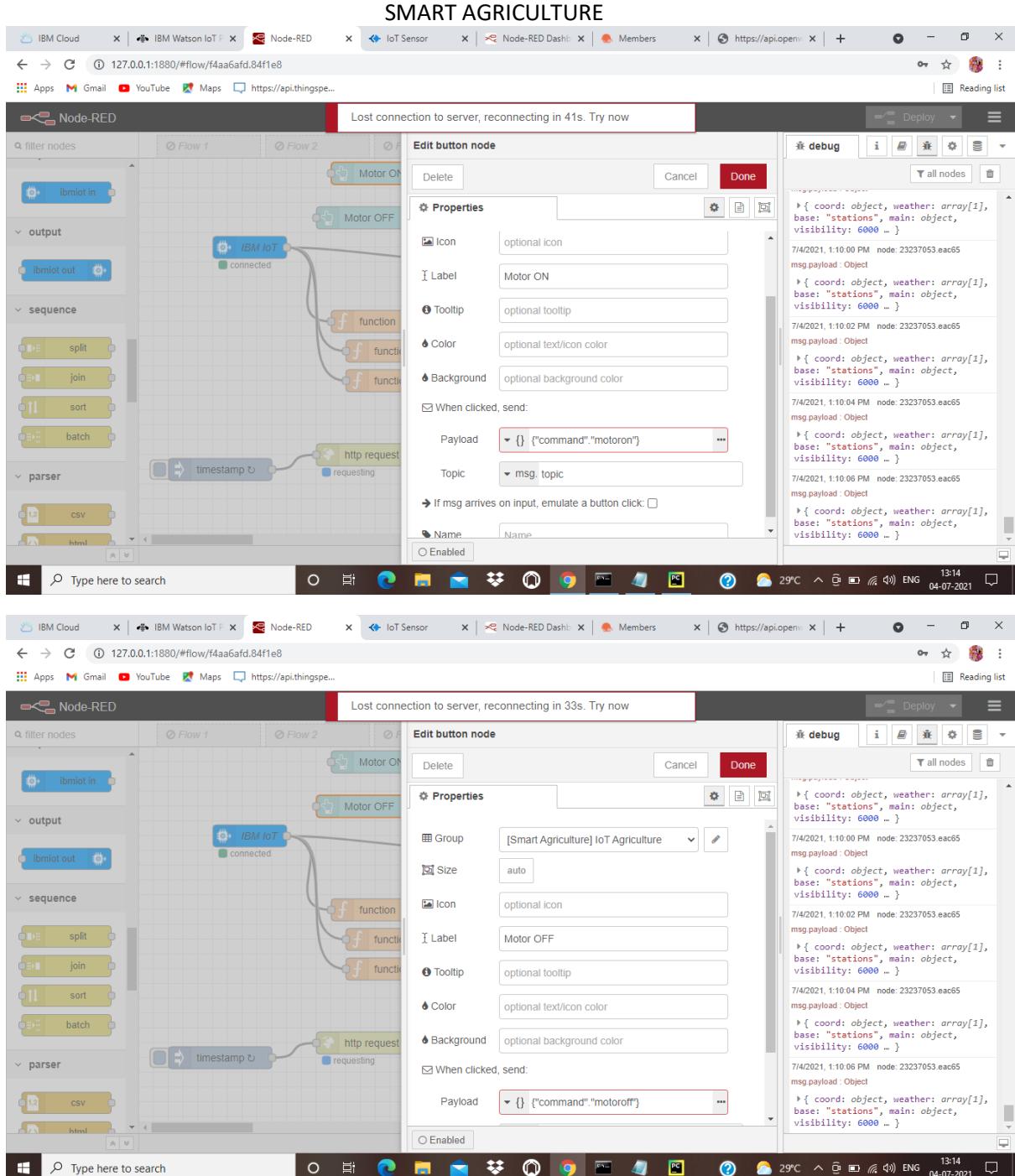


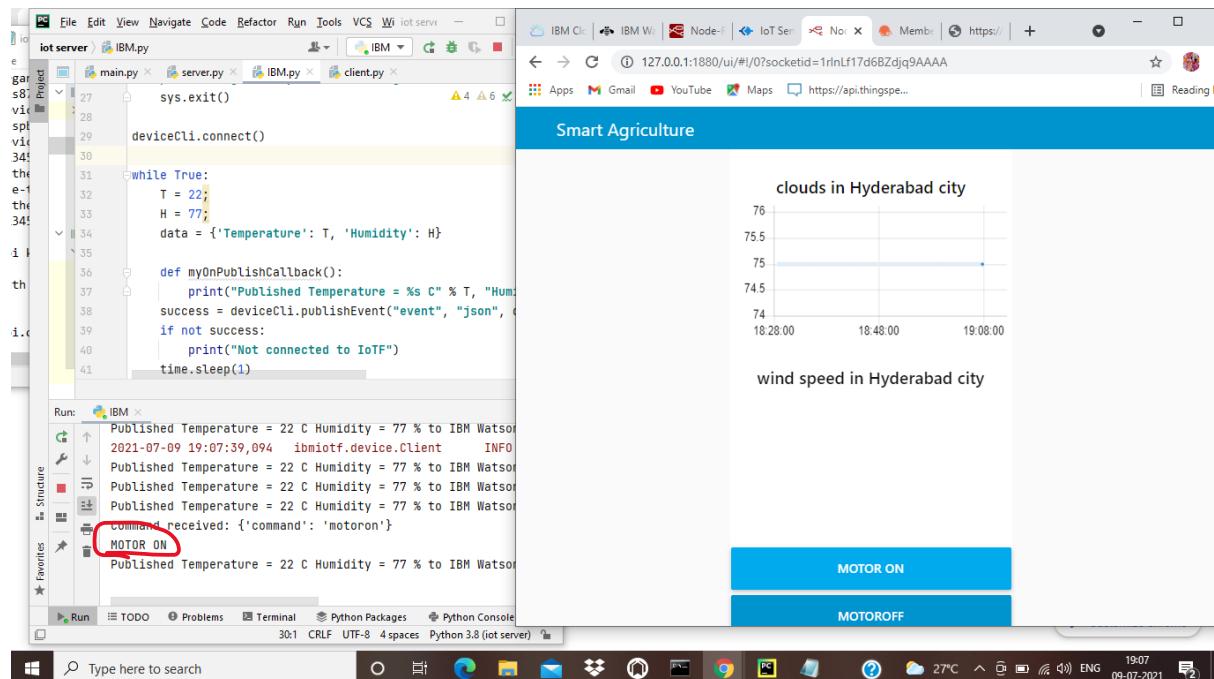
Figure-50

To operate buttons from dashboard we need to include two buttons and an ibm out node . double click on IBM IoT out node and configure it as shown in figure -49. Bring two buttons from nodes list one for switching on the motor and another for switching off the motor and label them as motor ON and motor OFF and give type of data as json format and give the commands as shown in figure-50 .

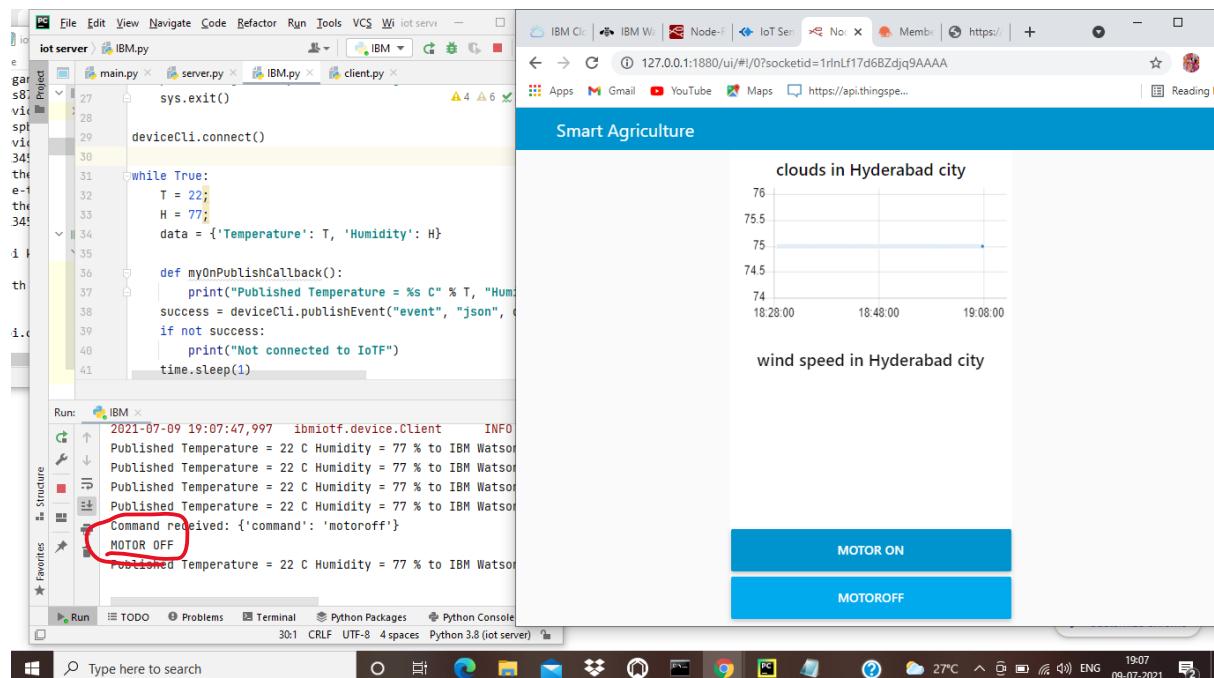
Deploy this and go to node-red dashboard and the code that we have .when we run the code , it has to display “MOTOR ON” when we click on motor on button and “MOTOR OFF” when we click on motor off button along with temperature, humidity. **We can see it is displaying the same thing that we have discussed in figure-51.**

SMART AGRICULTURE

41.output for python code



The screenshot shows a development environment with two main panes. The left pane is a code editor for an 'iot server' project containing four files: main.py, server.py, IBM.py, and client.py. The main.py file contains code for publishing sensor data and handling commands. The IBM.py file shows command-line output from running the script, including a log message and a command received: 'MOTOR ON'. The right pane is a web browser displaying a 'Smart Agriculture' dashboard with two charts: 'clouds in Hyderabad city' and 'wind speed in Hyderabad city'. Below the charts is a control panel with two buttons: 'MOTOR ON' and 'MOTOR OFF', with 'MOTOR ON' currently highlighted.



This screenshot is similar to the one above, but it shows a different command being received: 'MOTOR OFF'. The command is circled in red in the terminal output. The rest of the interface and data displays are identical to the first screenshot.

Figure-51

SMART AGRICULTURE

42.final flow and dashboard

The image shows two screenshots illustrating a smart agriculture system setup.

Node-RED Flow:

- The flow consists of five parallel streams (Flow 1 to Flow 5) triggered by an IBM IoT node.
- Flow 1:** Triggers a "Motor on" button.
- Flow 2:** Triggers a "Motoroff" button.
- Flow 3:** Triggers an "IBM IoT" node connected to an "IBM IoT" input node.
- Flow 4:** Triggers three "function" nodes which output "Temperature", "Humidity", and "Object temperature".
- Flow 5:** Triggers an "http request" node, followed by a "function" node that outputs "clouds in Hyderabad city" and another "function" node that outputs "wind speed in Hyderabad city".
- A "timestamp" node is used to timestamp the data before the "http request".
- The "msg payload" from the "http request" is shown in the debug tab, containing weather data for Hyderabad city.

Smart Agriculture Dashboard:

- The dashboard title is "Smart Agriculture".
- The "IoT Agriculture" section displays real-time data for Temperature, Humidity, Object temperature, clouds in Hyderabad city, and wind speed in Hyderabad city.
- Below the dashboard are two large blue buttons labeled "MOTOR ON" and "MOTOROFF".
- The bottom of the screen shows a Windows taskbar with various icons and system status.

Figure-52

Figure -52 displays our complete flow and corresponding dashboard that displays data of temperature, humidity, object temperature, clouds and windspeed in Hyderabad city and motor on and motor off buttons.

As we have discussed above the data from our IBM sensor is monitored by ourselves , let us see the changes in our dashboard.

SMART AGRICULTURE

43.changes in data in dashboard corresponding to sensor

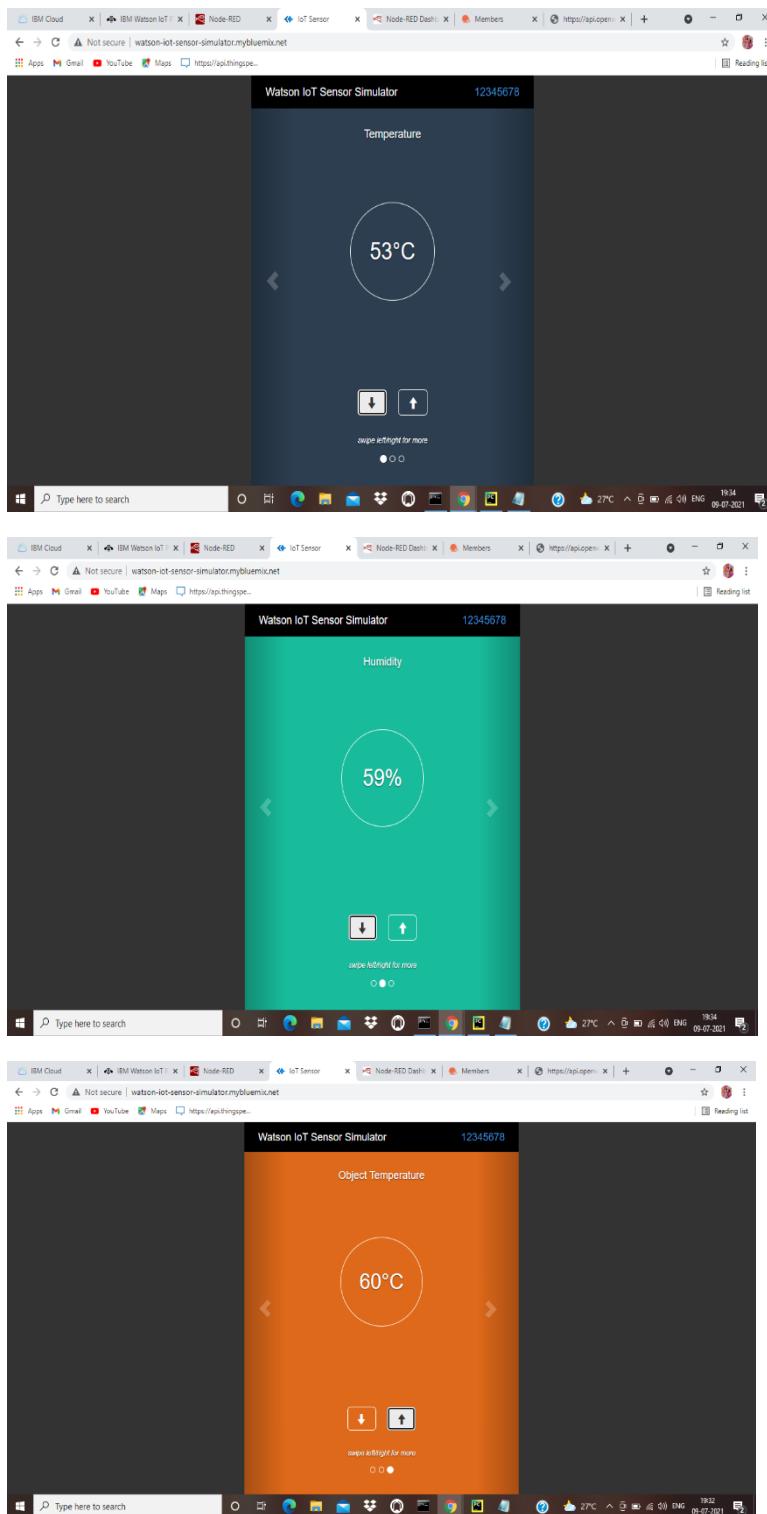


Figure-53

We have changed the values of quantities in our sensor as shown in figure-53

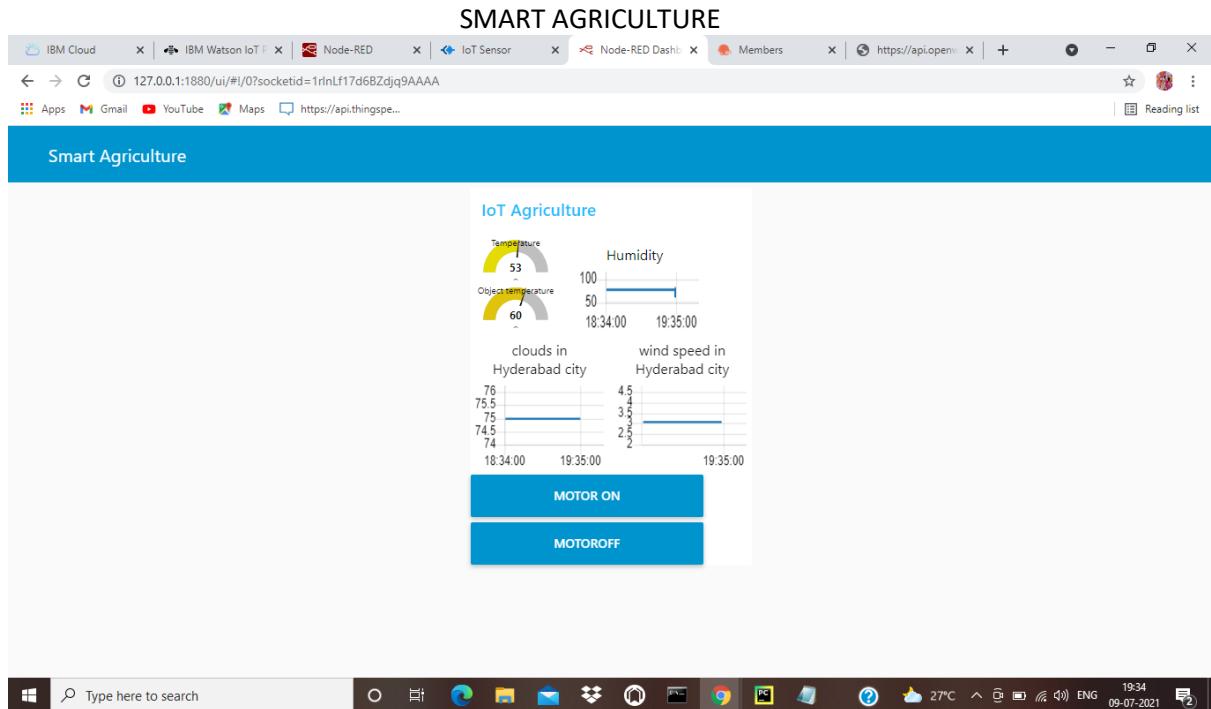
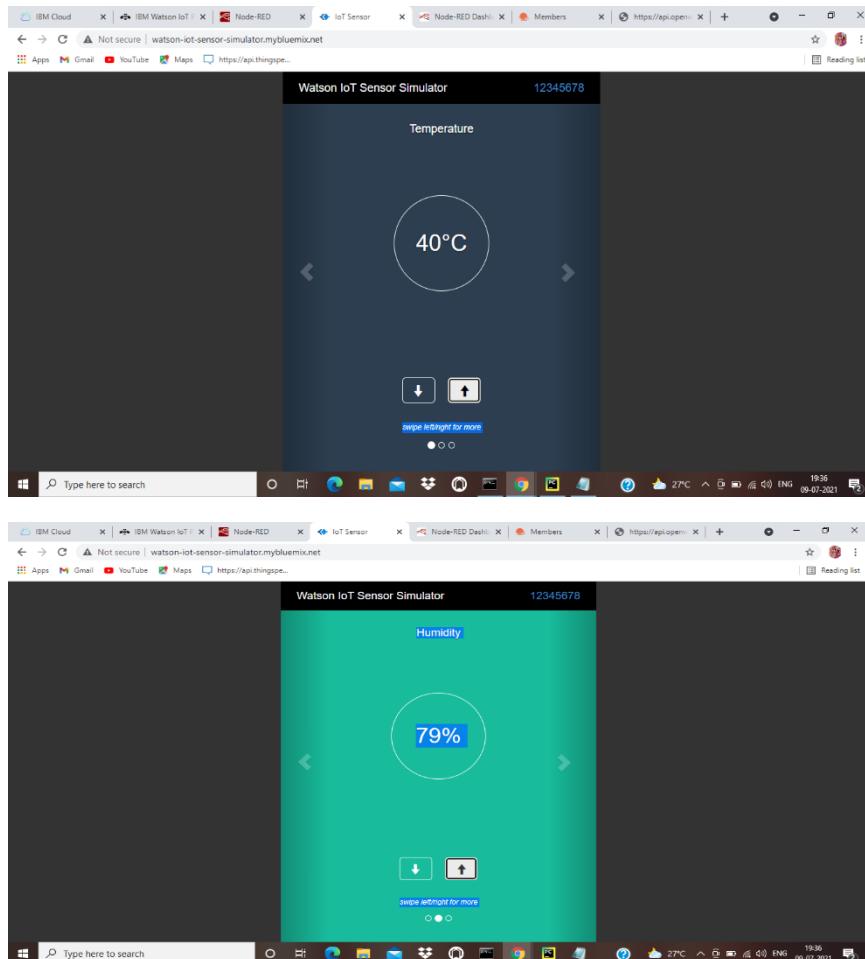


Figure-54

As we can see the change of data as well as colour accordingly in figure-54.



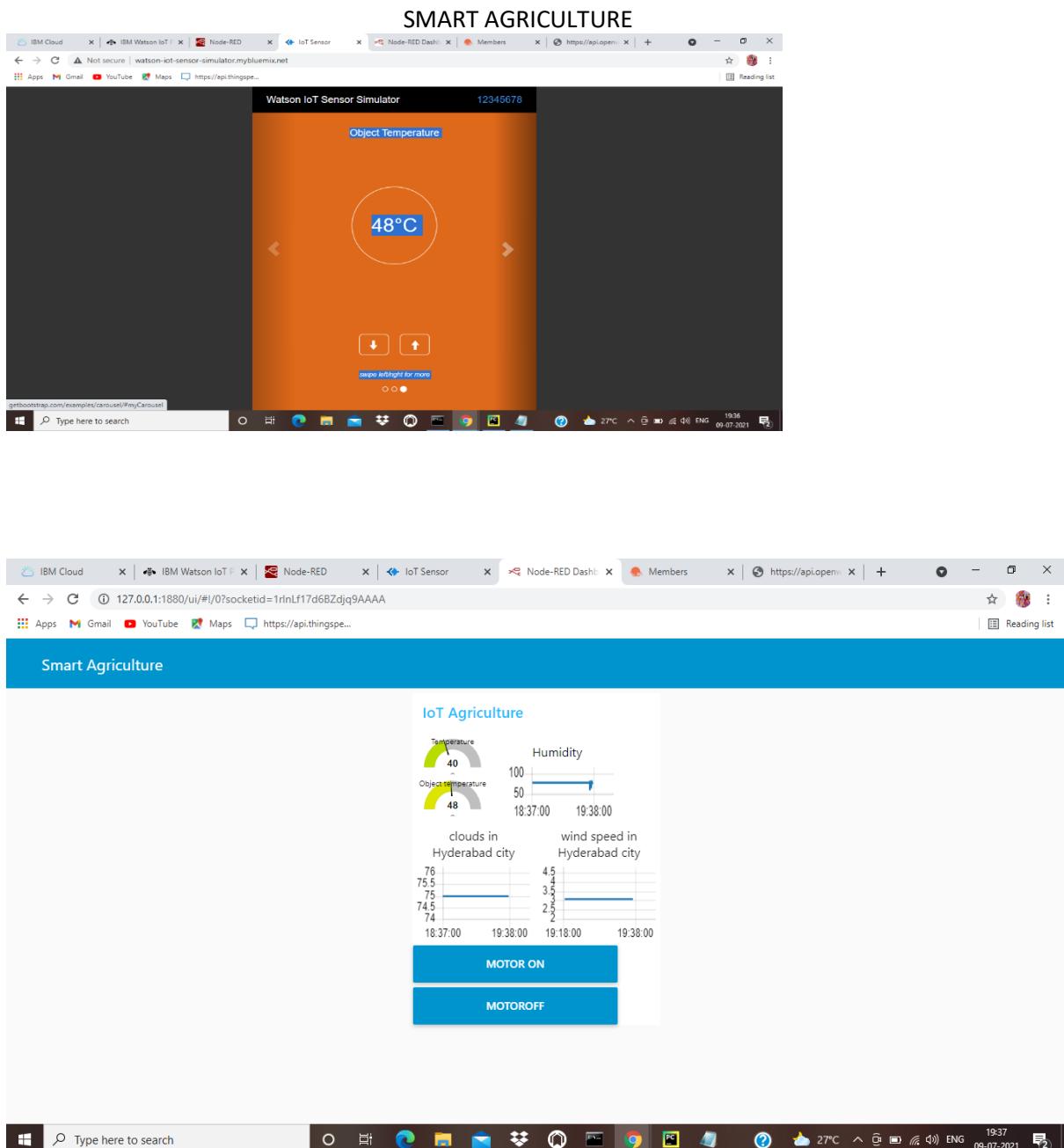
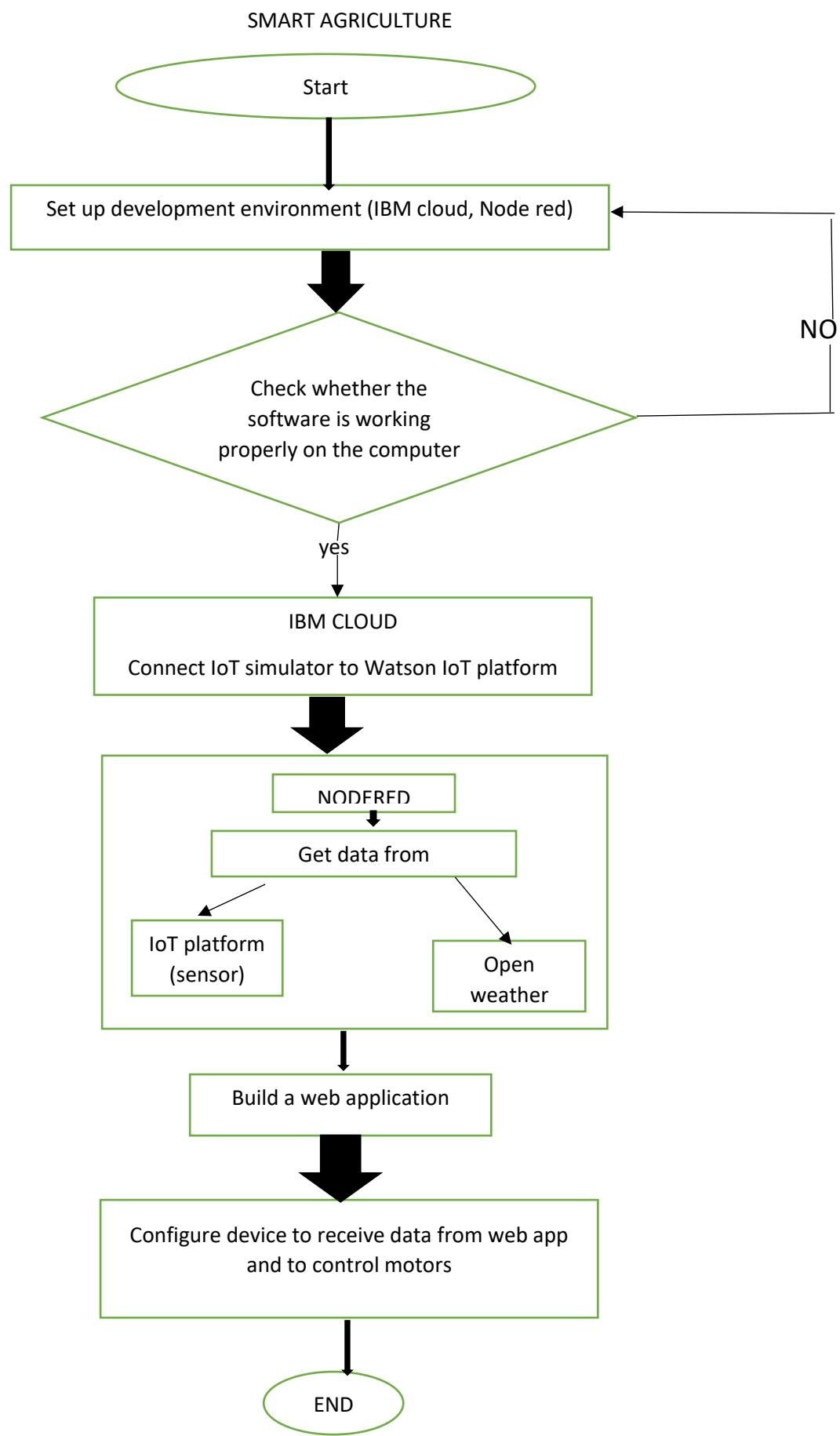


Figure-55

Figure-55 shows change in data in dashboard according to sensor.

Flow chart



SMART AGRICULTURE

Result

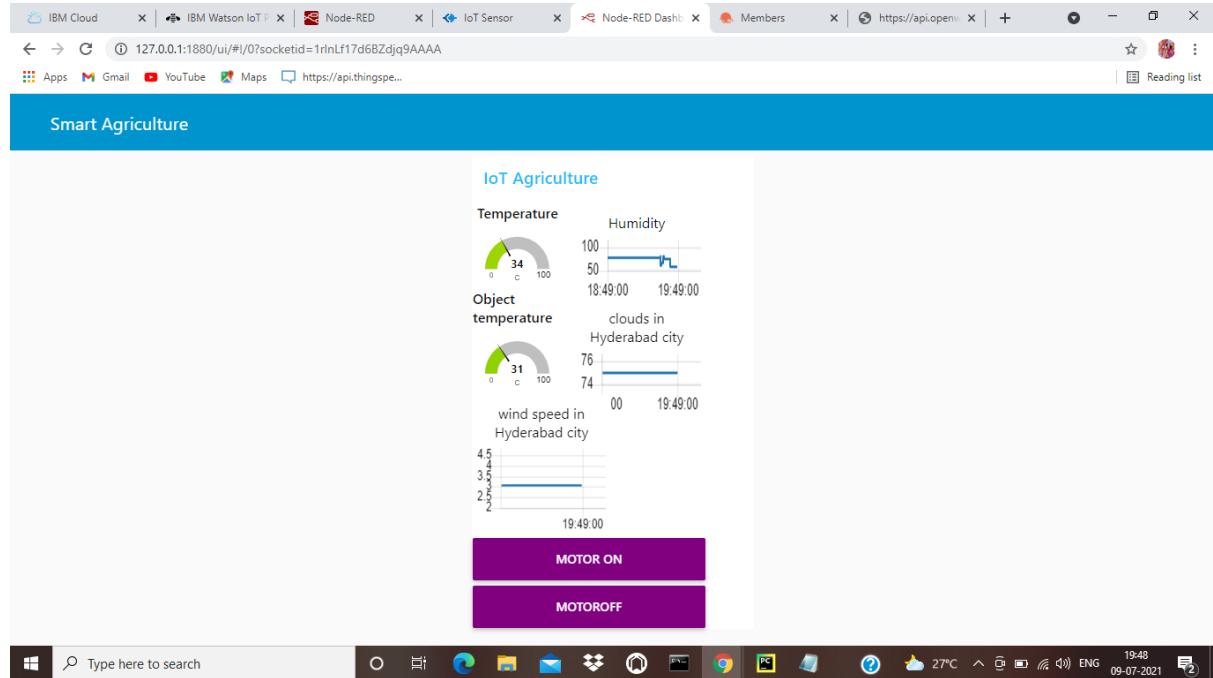


Figure-56

Figure -54 shows final dashboard (application) that displays temperature, humidity, and the object temperature from IBM sensor and clouds and windspeed in Hyderabad city. Two buttons for switching the motor on and motor off. We can control our motor from here.

Advantages & Disadvantages

Advantages

- *We can control motor by one click from anywhere .
- *We can easily know the level of the weather by colour displayed in dashboard.
- * Time will be saved if the farm is water lined properly ,and while irrigation of farm, the farmer need not to go to farm and check .
- *By this application farmer is able to know cloud data and wind speed , so that the farmer may make arrangements to protect crop from heavy rain.

Disadvantages

- *This application requires proper internet connection.
- *If the farmer does not have proper internet facility he may not get the right data from the server. Moreover we need to setup sensors each of these devices properly.
- *All the hardware or software whatever is included we need to moniter It thoroughly, if there is a small mistake at anywhere ,then the application may not work properly.

Applications

We can do some modifications and use the same application in home-automation. We can use the same application for operating lights, washing machine and refrigerator etc.

Washing machine:

We may use same motor on and motor off to allow water into the machine . similarly, we include two more buttons for switching on and off the machine. Temperature will display the water temperature to wash clothes in the machine .

Refrigerator :

We use two buttons to switch on and off the fridge. We can use temperature to alter freezer to refrigerator and refrigerator to freezer. We may use some other buttons to change the fridge into power saving mode etc.

Conclusion

We all know that devices connected to IoT are increasing day-by -day as it makes human work very easy and efficient. This a small IoT application for agriculture which serves the basic needs of a farmer(climate changes and irrigation control). By satisfying these needs of a farmer in real -life will help to boost our economy in agriculture sector.

Future scope

This smart agriculture is a web application. In future , if we make it as an app that is accessible for every farmer in their mobile phone , it will be more beneficial. And as discussed above , by doing little modifications on this application we may use it for home automation also. But, the need of IoT is needed in agriculture sector like countries in India. As we know Indian economy is mostly dependent on agricultural sector , by using IoT devices we may boost up our economy.

Reference

Literature survey: [An In-Depth Look at IoT Agriculture Projects & Use Cases \(link-labs.com\)](#)

Node- red functions code: <https://nodered.org/docs/user-guide/writing-functions>

a. Source code

```

import time
import sys
import ibmiotf.application
import ibmiotf.device

organization = "pls875"
deviceType = "Raspberrypi"
deviceId = "12345678"
authMethod = "token"
authToken = "12345678"

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)
    if cmd.data['command'] == 'motoron':
        print("MOTOR ON")
    elif cmd.data['command'] == 'motoroff':
        print("MOTOR OFF")

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()

deviceCli.connect()

while True:
    T = 22;
    H = 77;
    data = {'Temperature': T, 'Humidity': H}

    def myOnPublishCallback():
        print("Published Temperature = %s C" % T, "Humidity = %s %%" % H,
              "to IBM Watson")
        success = deviceCli.publishEvent("event", "json", data, qos=0,
                                         on_publish=myOnPublishCallback)
        if not success:

```

SMART AGRICULTURE

```
print("Not connected to IoTF")
time.sleep(1)

deviceCli.commandCallback = myCommandCallback
```
