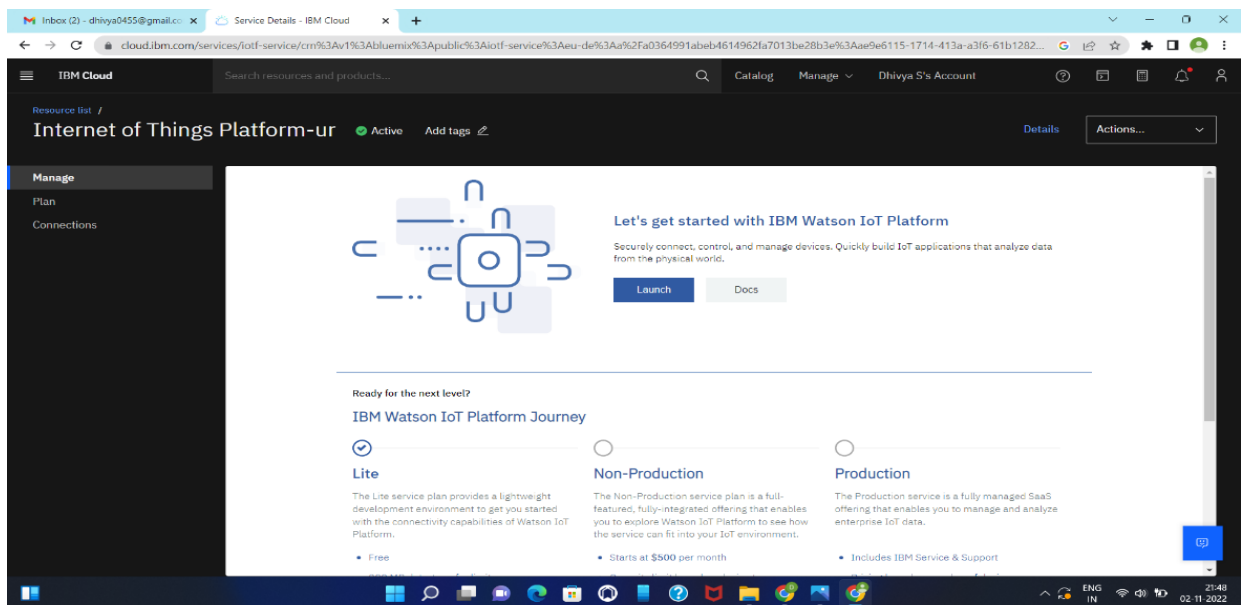


Sprint 2

Software- Create device in the IoT Watson Platform,
workflow for IoT Scenarios using Local Node

Date	8 October 2022
Team ID	PNT2022TMID41338
Project Name	Project – Smart Farmer-IoT Enabled smart Farming Application
Maximum Marks	4 Marks

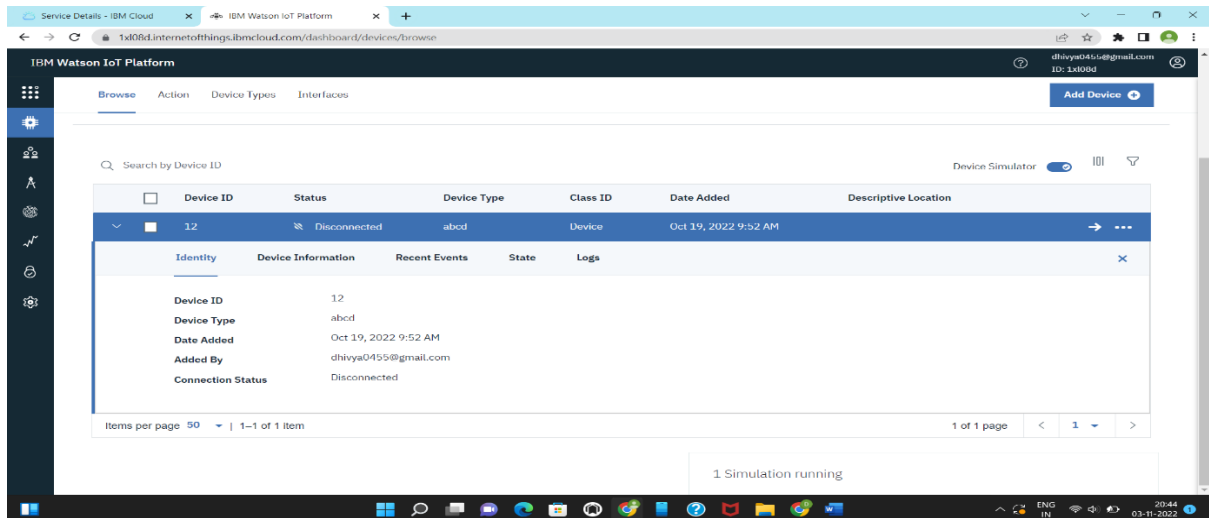
Launch IBM Watson IoT Platform:



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.

Create a new device:



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/>

Connecting IoT Simulator to IBM Watson IoT Platform:

My credentials given to simulator are:

Org: 1xl08d

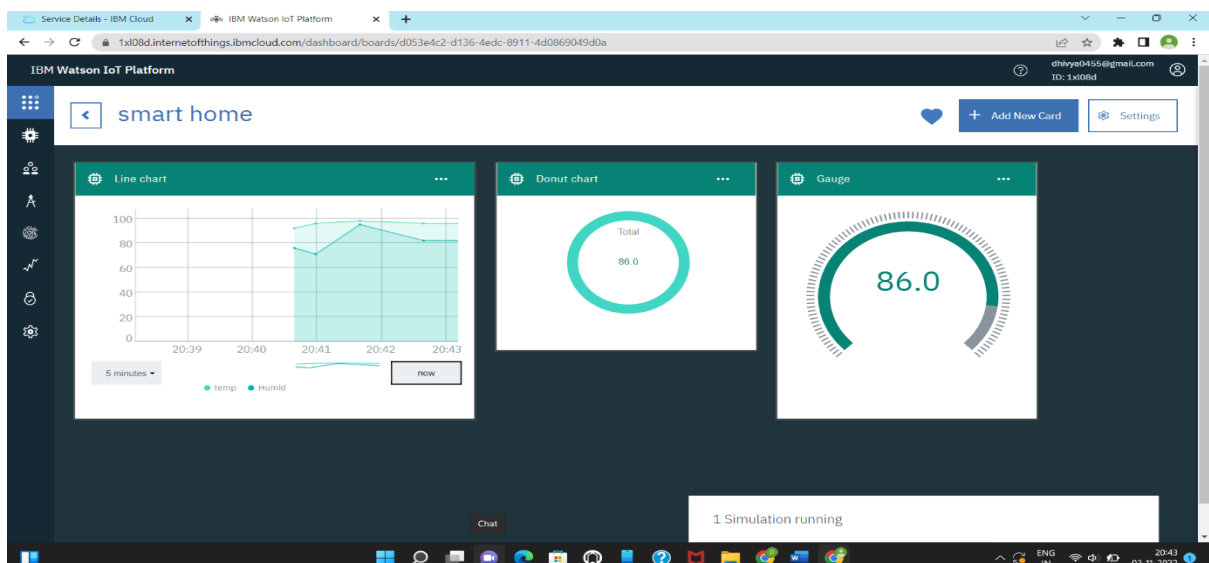
API: a-1xl08d-p5eywn2eu

Auth Token: GpIJ5spsrx0ZB*RLmJ

Device Type: abcd

Device ID: 12

Device Token: 12345678



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

- 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)

The screenshot shows the IBM Watson IoT Platform interface. The main dashboard displays a table of devices, with one device selected. A modal window titled 'Device Type: abcd' is open, showing the 'Events' configuration. The modal includes a 'Send' button, a 'Schedule' dropdown set to 'Every Minute', and a 'Payload' editor. The payload is a JSON object with random values for temperature, humidity, and a sensor ID.

Event	Value	Format	Last Received
eventflow	{"randomNumber":98,"temp":102,"Humid":90}	json	a few seconds ago
eventflow	{"randomNumber":11,"temp":98,"Humid":78}	json	a few seconds ago
eventflow	{"randomNumber":2,"temp":99,"Humid":82}	json	a few seconds ago
eventflow	{"randomNumber":15,"temp":102,"Humid":93}	json	a few seconds ago
eventflow	{"randomNumber":34,"temp":92,"Humid":79}	json	a few seconds ago

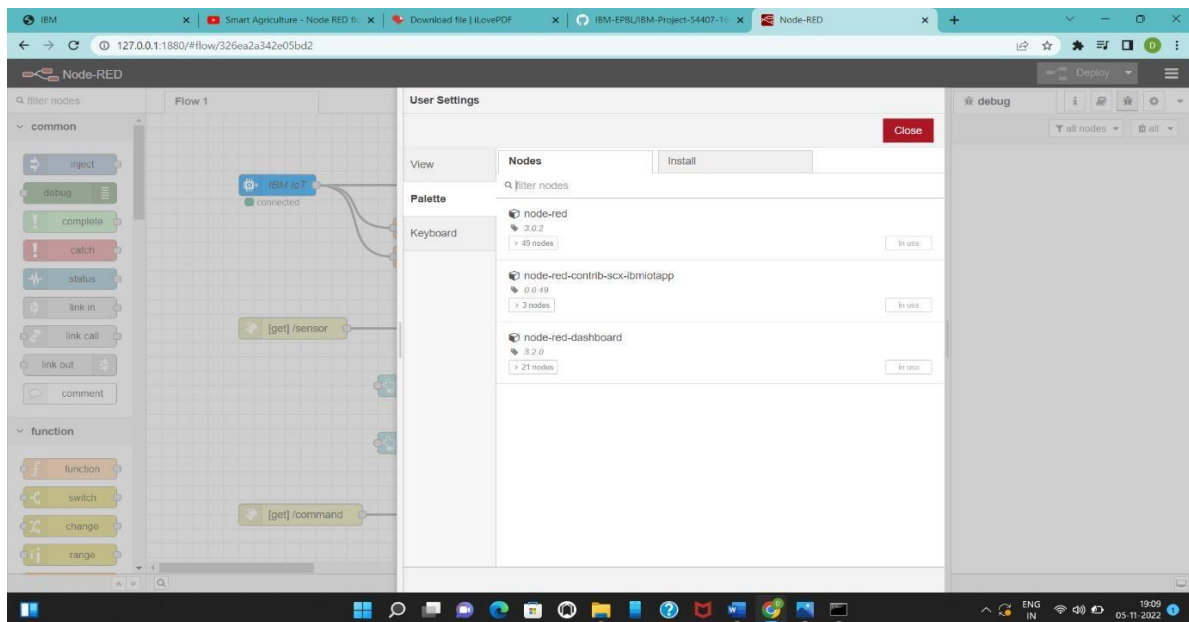
Configuring IBM-IoT to Node-RED connection

The screenshot shows the Node-RED interface. The 'Edit IBM IoT node' dialog is open, displaying the configuration for the 'API' node. The configuration includes the API Key, API Token, and Server Name. The 'Keep Alive' setting is set to 60 seconds, and the 'Use Clean Session' checkbox is checked.

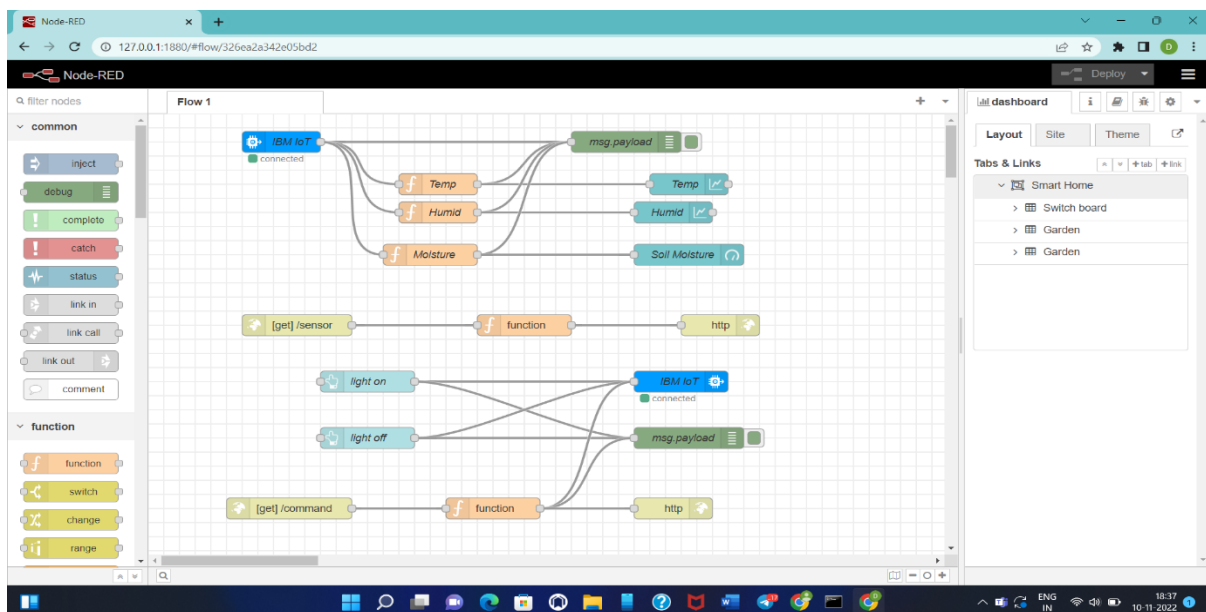
Properties:

- Name: API
- API Key: a-1x108d-p5eyyn2eu
- API Token: *****
- Server Name: orgid.messaging.internetofthings.ibmcloud.com
- Scalable: ☐
- Application ID:
- Keep Alive: 60 Seconds
- Use Clean Session: ☒

Installing a node-red-contrib-scx-ibmiotapp and node-red dashboard

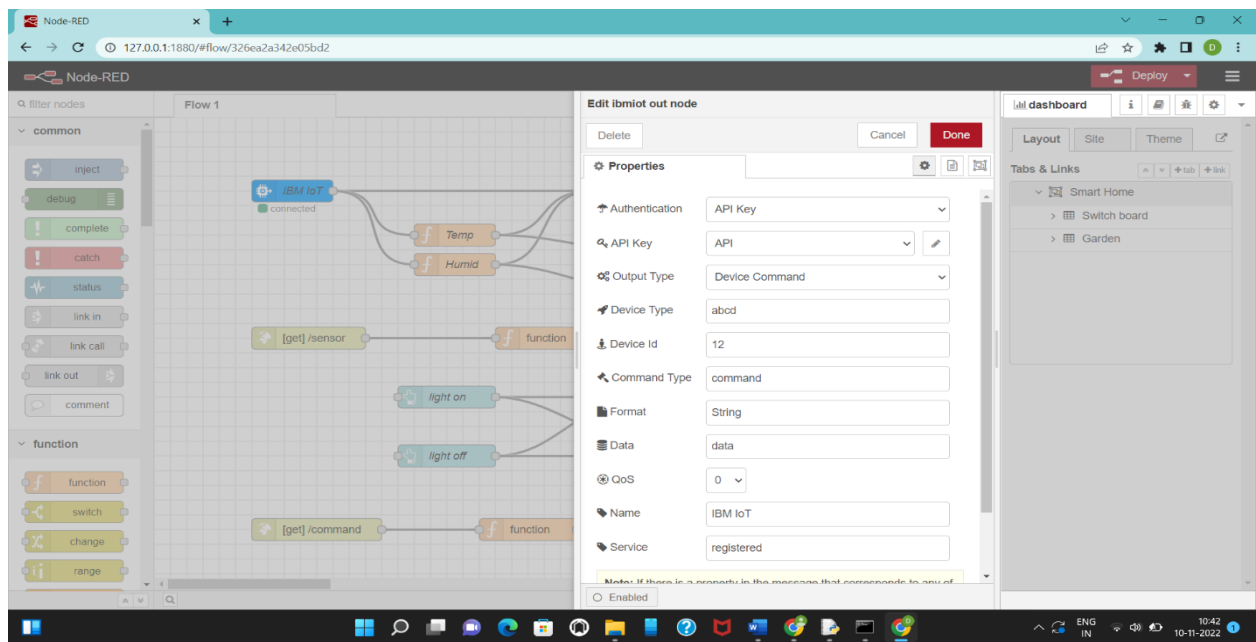


Complete Program Flow:



Configuration of Node-Red to collect IBM cloud data

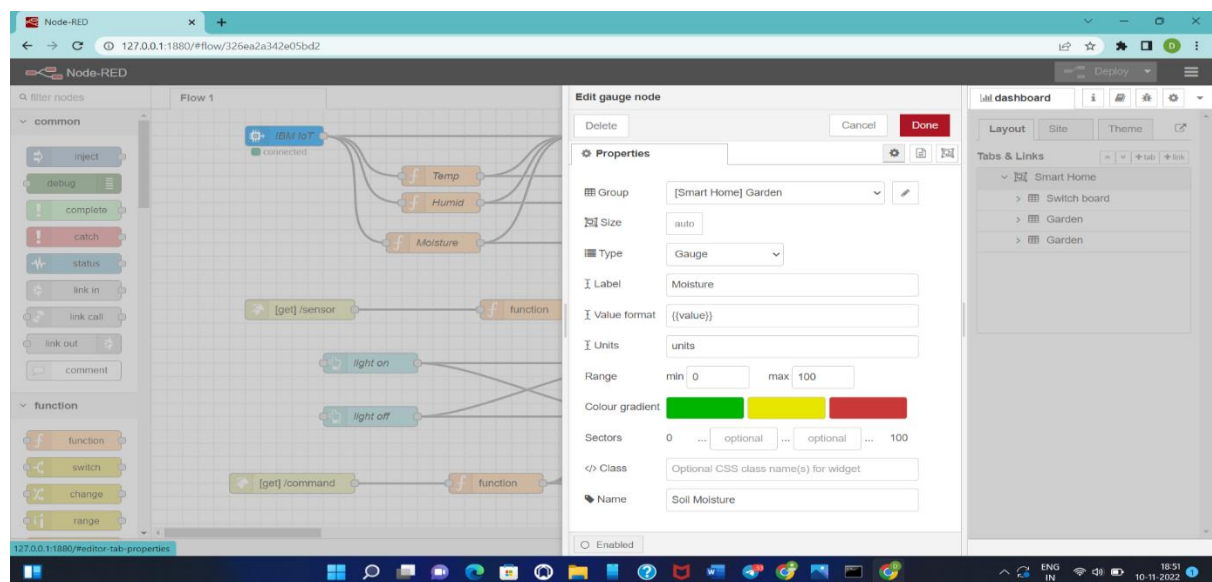
The node IBM IoT App In is added to Node-Red workflow. Then the appropriate device credentials obtained earlier are entered into the node to connect and fetch device telemetry to Node-Red.



Connect function node and The Java Script code for the function node is:

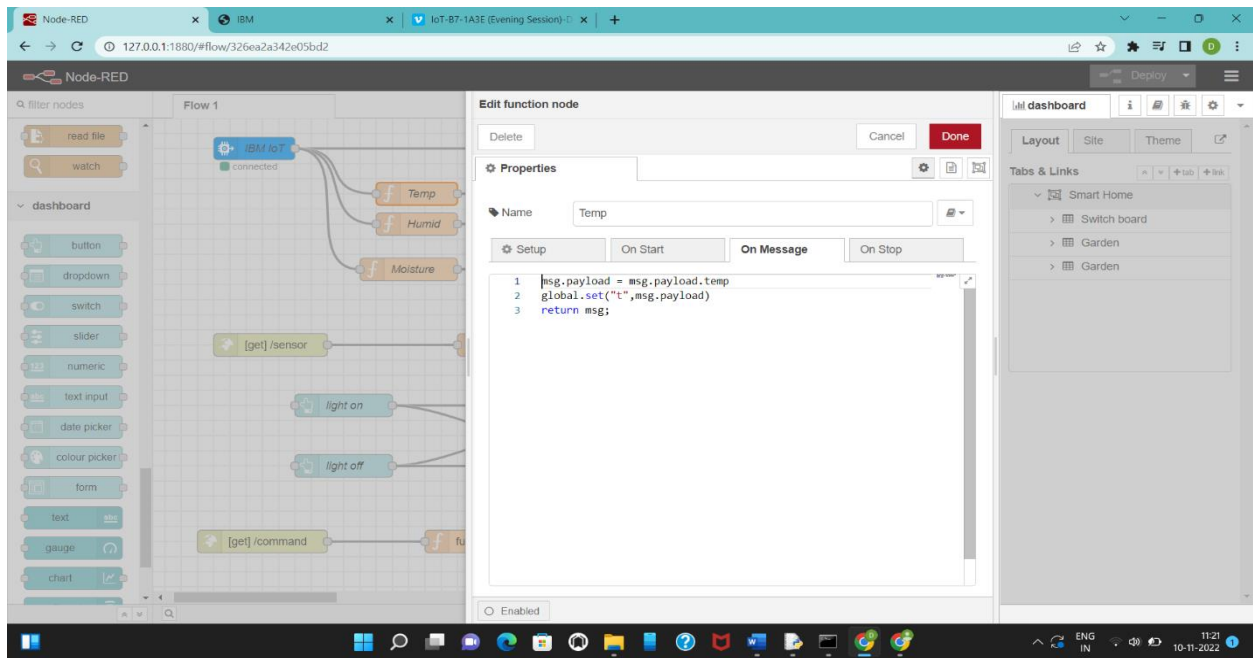
```
msg.payload=msg.payload.temp
return msg;
```

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

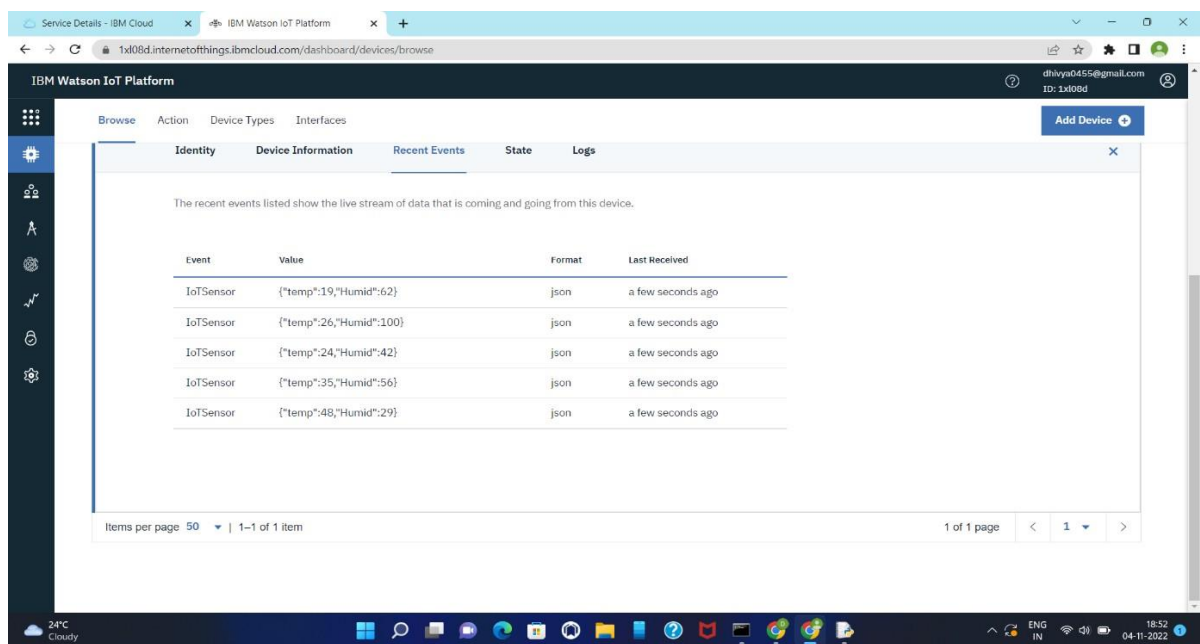


Configuration of Node-Red to collect data from OpenWeather

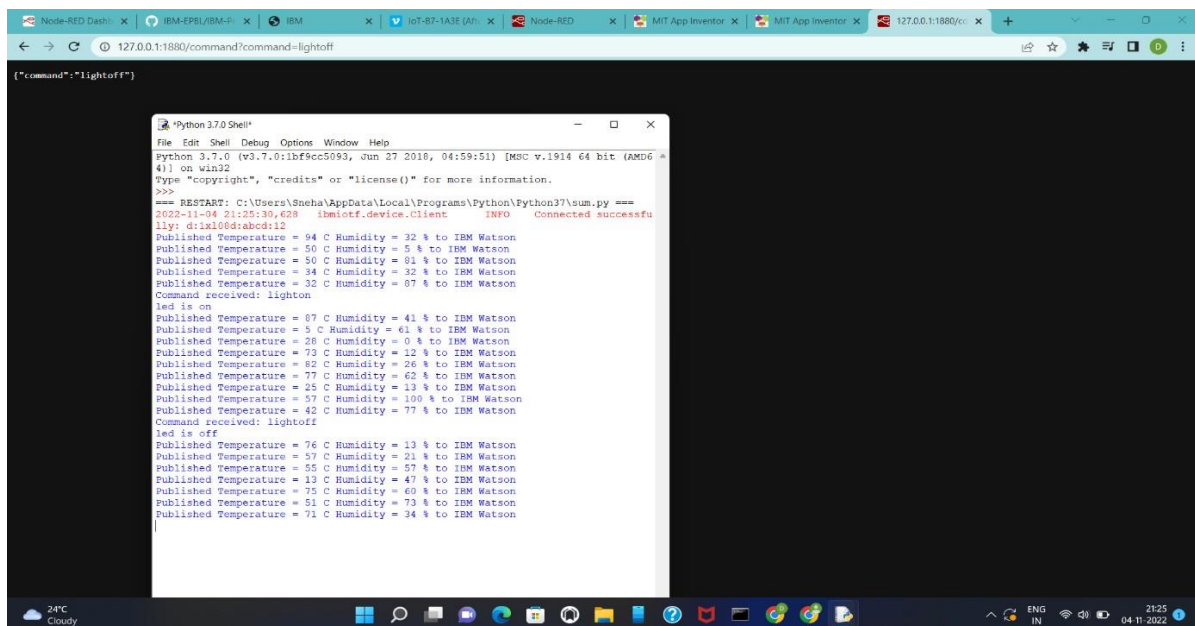
The Node-Red also receive data from the OpenWeather API by HTTP GET request. An inject trigger is added to perform HTTP request for every certain interval.



Checking IoT sensor Output in IBM Watson



Checking IoT sensor using command in Node-RED



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\Sneha\AppData\Local\Programs\Python\Python37\sum.py ====
2022-11-04 21:25:30.628 ibmiotf.device.Client INFO Connected successfully: d1x100d1abcd12
Published Temperature = 94 C Humidity = 32 % to IBM Watson
Published Temperature = 50 C Humidity = 5 % to IBM Watson
Published Temperature = 50 C Humidity = 81 % to IBM Watson
Published Temperature = 34 C Humidity = 32 % to IBM Watson
Published Temperature = 32 C Humidity = 87 % to IBM Watson
Command received: lighton
led is on
Published Temperature = 87 C Humidity = 41 % to IBM Watson
Published Temperature = 5 C Humidity = 61 % to IBM Watson
Published Temperature = 28 C Humidity = 0 % to IBM Watson
Published Temperature = 73 C Humidity = 12 % to IBM Watson
Published Temperature = 82 C Humidity = 26 % to IBM Watson
Published Temperature = 77 C Humidity = 62 % to IBM Watson
Published Temperature = 25 C Humidity = 13 % to IBM Watson
Published Temperature = 57 C Humidity = 100 % to IBM Watson
Published Temperature = 42 C Humidity = 77 % to IBM Watson
Command received: lightoff
led is off
Published Temperature = 76 C Humidity = 13 % to IBM Watson
Published Temperature = 57 C Humidity = 21 % to IBM Watson
Published Temperature = 55 C Humidity = 57 % to IBM Watson
Published Temperature = 13 C Humidity = 47 % to IBM Watson
Published Temperature = 75 C Humidity = 69 % to IBM Watson
Published Temperature = 51 C Humidity = 73 % to IBM Watson
Published Temperature = 71 C Humidity = 34 % to IBM Watson
```

Output in Node-RED Dashboard:

