

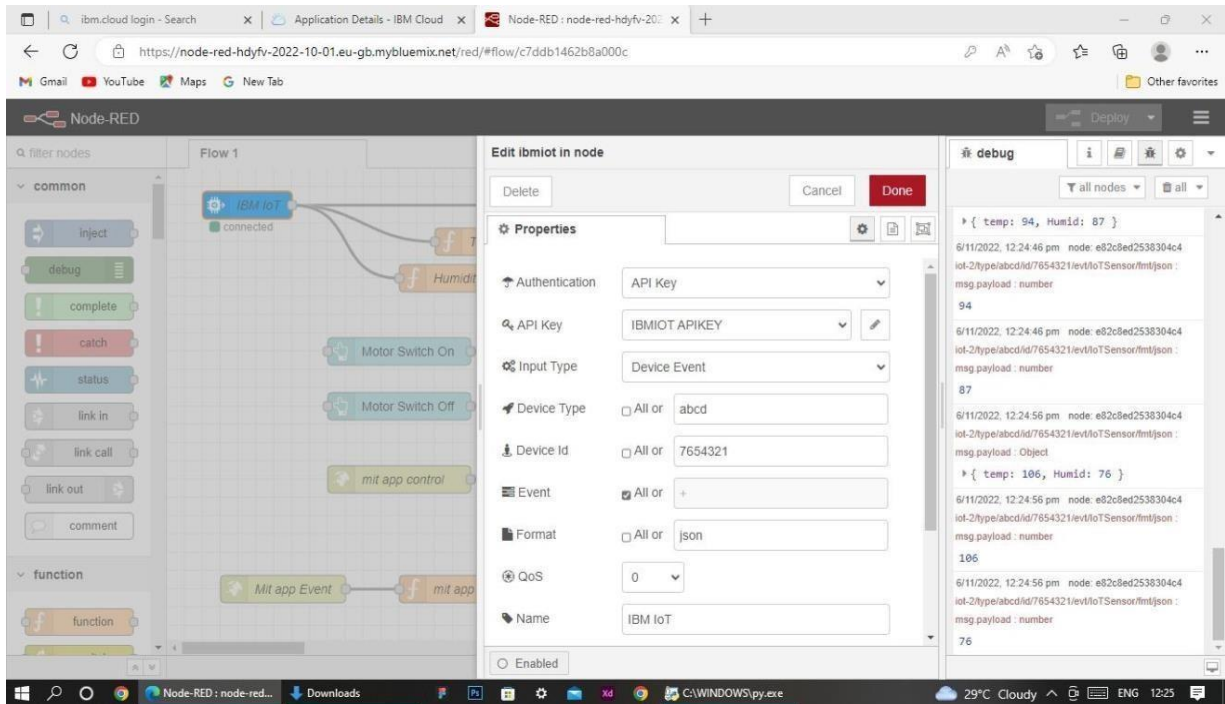
# **IOT    ENABLED    SMART FARMING APPLICATION.**

**Build A Web Application Using Node-RED**

TEAM ID : [PNT2022TMID22782](#)

## Configuration of Node-Red to collect IBM cloud data

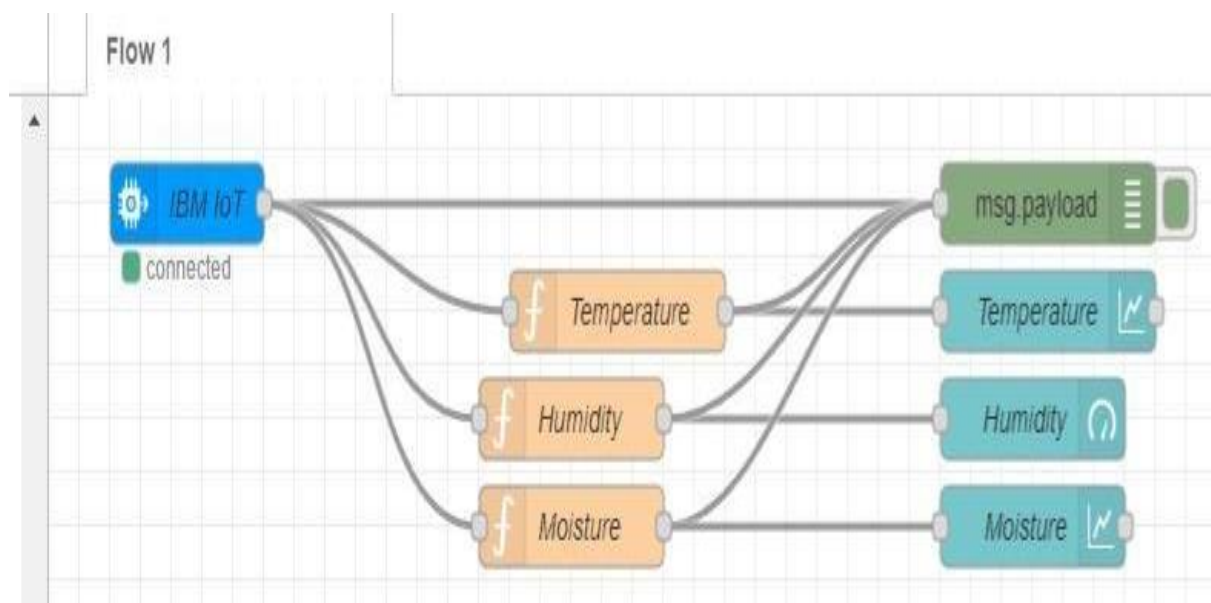
The node IBM IOT App In is added to Node-Red workflow. Then the appropriate device credentials obtained earlier are entered into the node to connect and fetch device telemetry to Node-Red



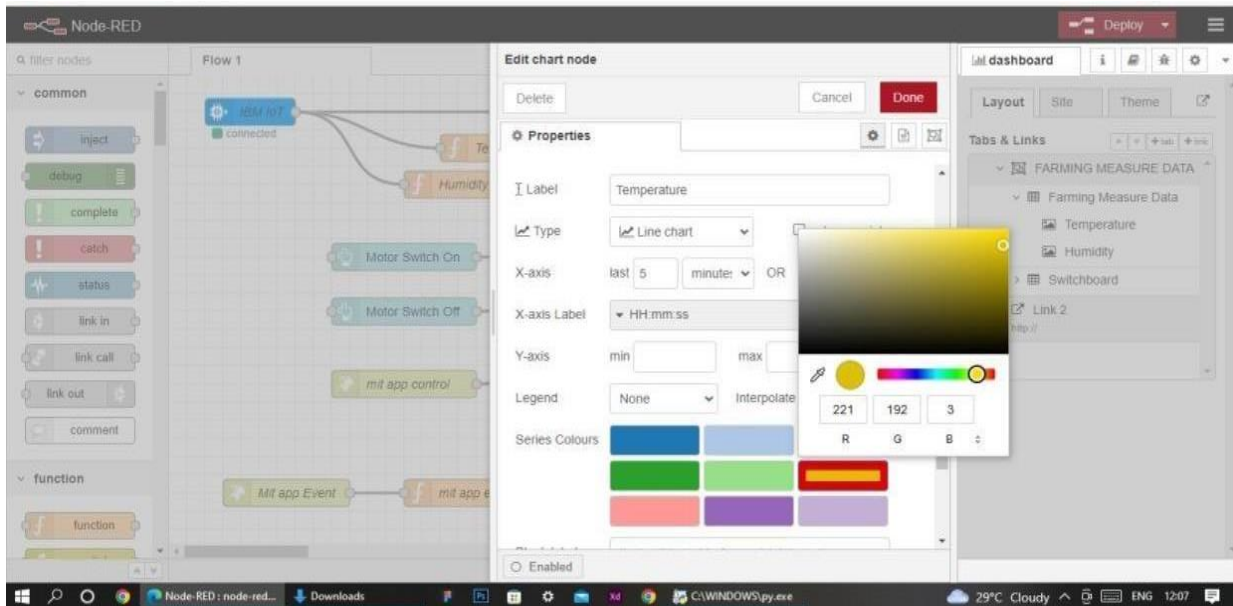
- Once it is connected Node-Red receives data from the device
- Display the data using debug node for verification
- Connect function node and write the Java script code to get each reading separately.
- The Java script code for the function node is:  
`msg.payload=msg.payload.d.temperature return msg;`
- Finally connect Gauge nodes from dashboard to see the data in UI

```
C:\WINDOWS\py.exe
Published Temperature = 109 C Humidity = 64 % to IBM Watson
Published Temperature = 105 C Humidity = 86 % to IBM Watson
Published Temperature = 105 C Humidity = 83 % to IBM Watson
Published Temperature = 102 C Humidity = 86 % to IBM Watson
Published Temperature = 103 C Humidity = 60 % to IBM Watson
Published Temperature = 106 C Humidity = 83 % to IBM Watson
Published Temperature = 101 C Humidity = 85 % to IBM Watson
Published Temperature = 106 C Humidity = 84 % to IBM Watson
Published Temperature = 95 C Humidity = 74 % to IBM Watson
Published Temperature = 107 C Humidity = 73 % to IBM Watson
Published Temperature = 92 C Humidity = 96 % to IBM Watson
Published Temperature = 93 C Humidity = 82 % to IBM Watson
Published Temperature = 98 C Humidity = 80 % to IBM Watson
Published Temperature = 107 C Humidity = 71 % to IBM Watson
Published Temperature = 94 C Humidity = 87 % to IBM Watson
Published Temperature = 106 C Humidity = 76 % to IBM Watson
Published Temperature = 98 C Humidity = 81 % to IBM Watson
Published Temperature = 103 C Humidity = 95 % to IBM Watson
Published Temperature = 92 C Humidity = 66 % to IBM Watson
Published Temperature = 99 C Humidity = 76 % to IBM Watson
Published Temperature = 93 C Humidity = 68 % to IBM Watson
```

**Data received from the cloud in Node-Red console**



**Nodes connected in following manner to get each reading separately**



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

## Configuration of Node-Red to collect data from Open Weather

- The Node-Red also receive data from the Open Weather 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
- The data we receive from Open Weather after request is in below JSON

**format:** { "coord": { "lon": 79.85, "lat": 14.13 }, "weather": [ { "id": 803, "main": "Clouds", "description": "broken clouds", "icon": "04n" } ], "base": "stations", "main": { "temp": 307.59, "feels\_like": 305.5, "temp\_min": 307.59, "temp\_max": 307.59, "pressure": 1002, "humidity": 35, "sea\_level": 1002, "ground\_level": 1000 }, "wind": { "speed": 6.23, "deg": 170 }, "clouds": { "all": 68 }, "dt": 1589991979, "sys": { "country": "IN", "sunrise": 1589933553, "sun

```
set":1589979720},"timezone":19800,"id":1270791,"name":"Gūdūr",  
cod":20 0}
```

In order to parse the JSON string we use Java script functions and get each parameters

```
var temperature = msg.payload.main.temp;
```

```
temperature = temperature-273.15;
```

```
return {payload : temperature.toFixed(2)};
```

In the above Java script code we take temperature parameter into a new variable and convert it from kelvin to Celsius

Then we add Gauge and text nodes to represent data visually in UI.

