

SPRINT - 2

Team ID	PNT2022TMID00072
Project Name	Smart Farmer-IoT Enabled Smart Farming Application

Connecting IOT Simulator to IBM Watson IOT Platform

Give the credentials of your device in IBM Watson

My credentials given to simulator are:

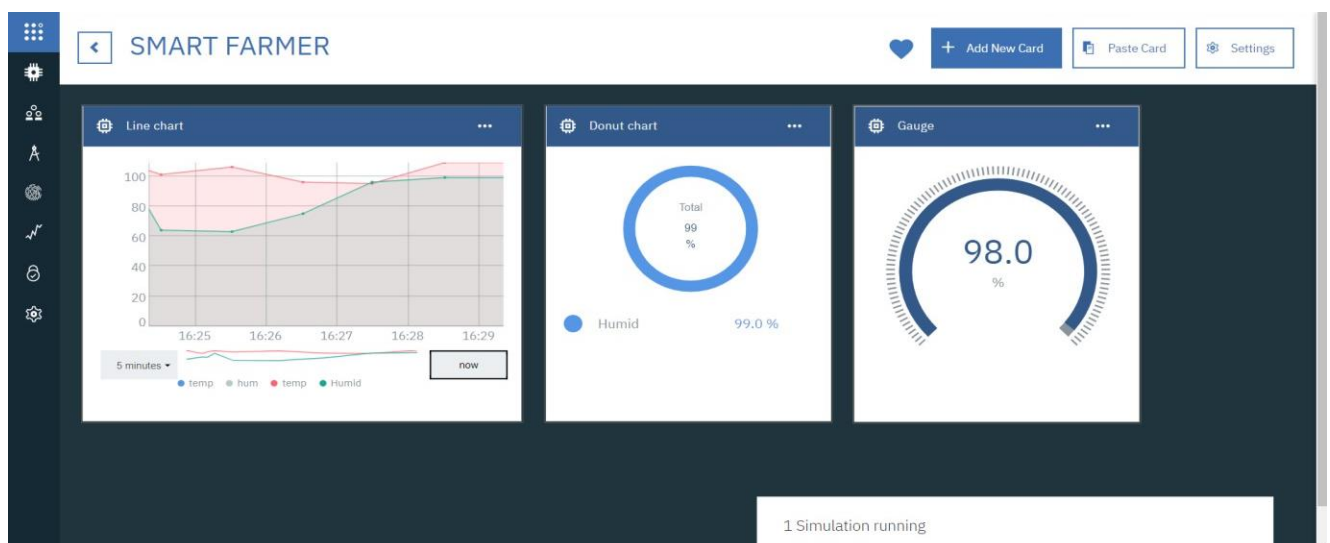
OrgID: **p2cfk6**

Device type: **SMART**

Device ID : **15**

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)

```
{
  "Moisture":98,
  "temp":109,
  "Humid":98
}
```

The screenshot shows a web interface for managing IoT devices. At the top, there are tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A search bar labeled 'Search by Device ID' is present. On the right, there's a 'Device Simulator' toggle and an 'Add Device' button. The main table lists devices with columns: Device ID, Status, Device Type, Class ID, Date Added, and Descriptive Location. Device 15 is selected, and its details are shown in a sidebar. The 'Recent Events' tab is active, displaying a table of events with columns: Event, Value, Format, and Last Received. The events show a stream of JSON data for moisture, temperature, and humidity. A status indicator at the bottom right says '1 Simulation running'.

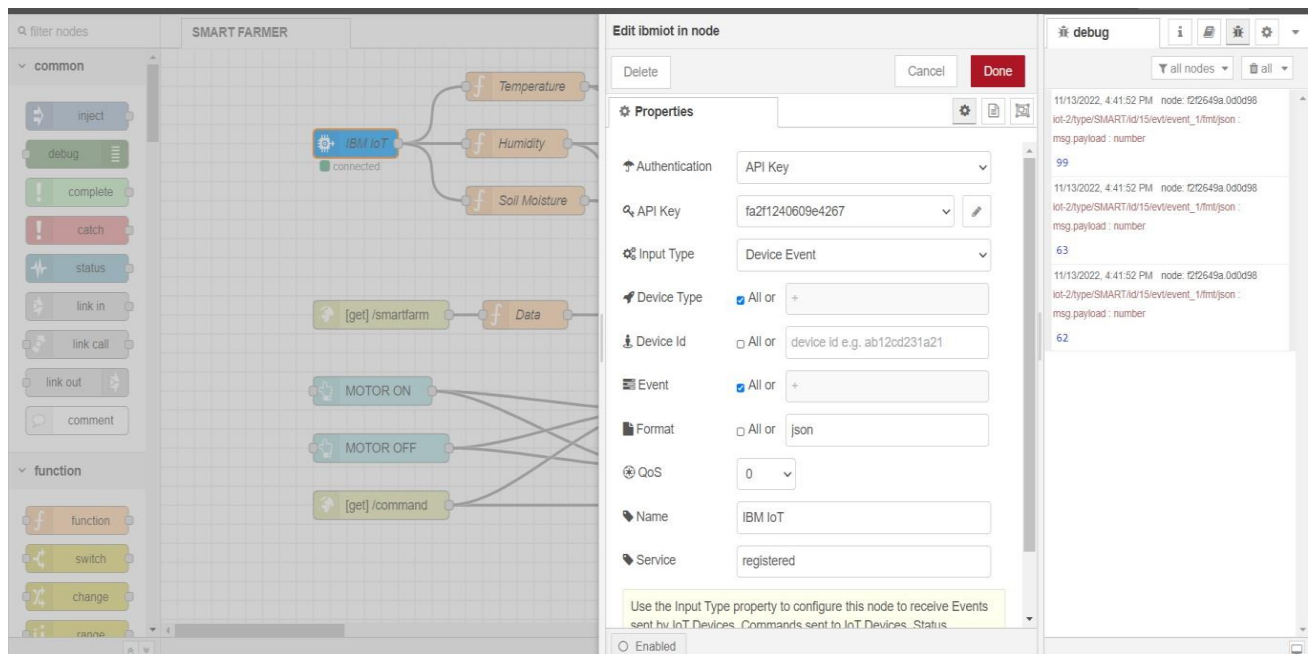
Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
12	Disconnected	ABCD	Device	Oct 13, 2022 11:13 AM	
15	Disconnected	SMART	Device	Nov 7, 2022 2:52 PM	

Event	Value	Format	Last Received
event_1	{"Moisture":69,"temp":98,"Humid":81}	json	a few seconds ago
event_1	{"Moisture":58,"temp":107,"Humid":68}	json	a few seconds ago
event_1	{"Moisture":97,"temp":100,"Humid":61}	json	a few seconds ago
event_1	{"Moisture":83,"temp":105,"Humid":85}	json	a few seconds ago
event_1	{"Moisture":68,"temp":104,"Humid":87}	json	19 minutes

1 Simulation running

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.temp
return msg;
- Finally connect Gauge nodes from dashboard to see the data in UI.

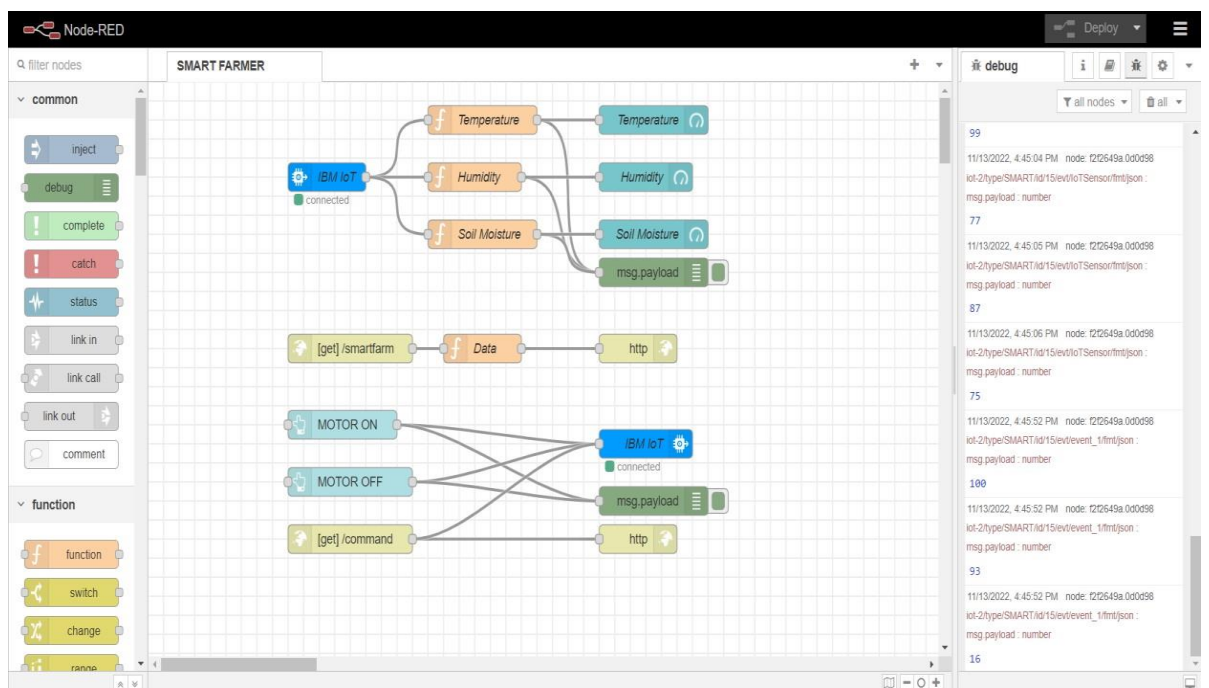
- Data send by the python code

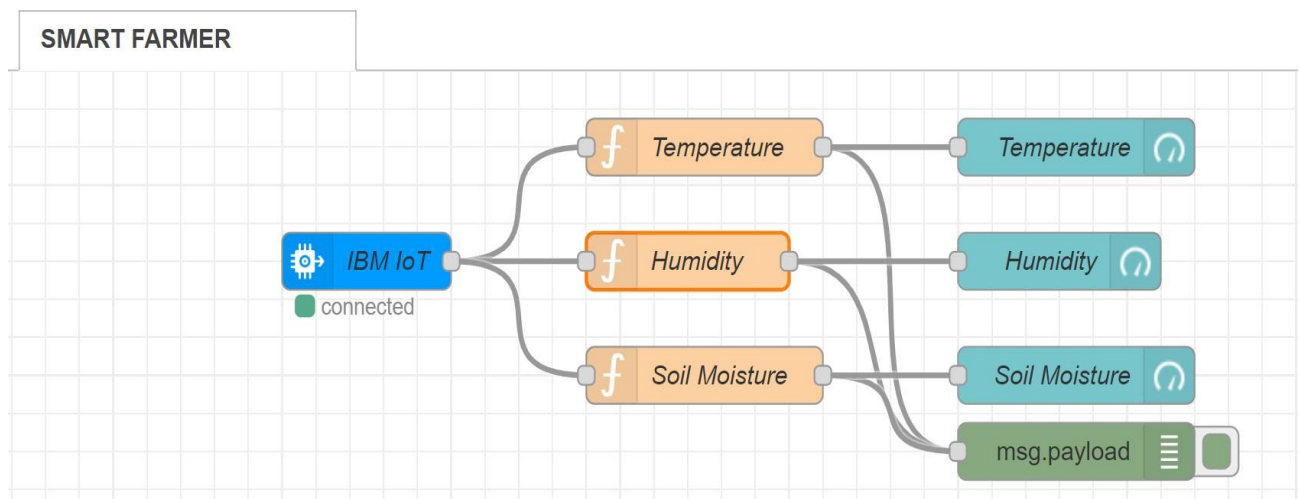
```

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: D:\IBM PROJECT\python 3.7\ibmiotpython.py =====
2022-11-13 16:44:34,200 ibmiotf.device.Client INFO Connected successfully: d:p2cfk6:SMART:15
Published Temperature = 0 C Humidity = 44 % Soil Moisture = 84 % to IBM Watson
Published Temperature = 34 C Humidity = 49 % Soil Moisture = 99 % to IBM Watson
Published Temperature = 77 C Humidity = 90 % Soil Moisture = 99 % to IBM Watson

```

- Data received from the cloud in Node-Red console





- Nodes connected in following manner to get each reading 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.
- The link to get open weather API :
<https://api.openweathermap.org/data/2.5/weather?lat=11.4383197&lon=77.5402674&appid=124d808d2039542453a0b1b05f37e900>
- The data we receive from Open Weather after request is in below JSON format.
- {"coord":{"lon":77.5403,"lat":11.4383},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04d"}],"base":"stations","main":{"temp":300.33,"feels_like":303.19,"temp_min":300.33,"temp_max":300.33,"pressure":1009,"humidity":79,"sea_level":1009,"grnd_level":986},"visibility":10000,"wind":

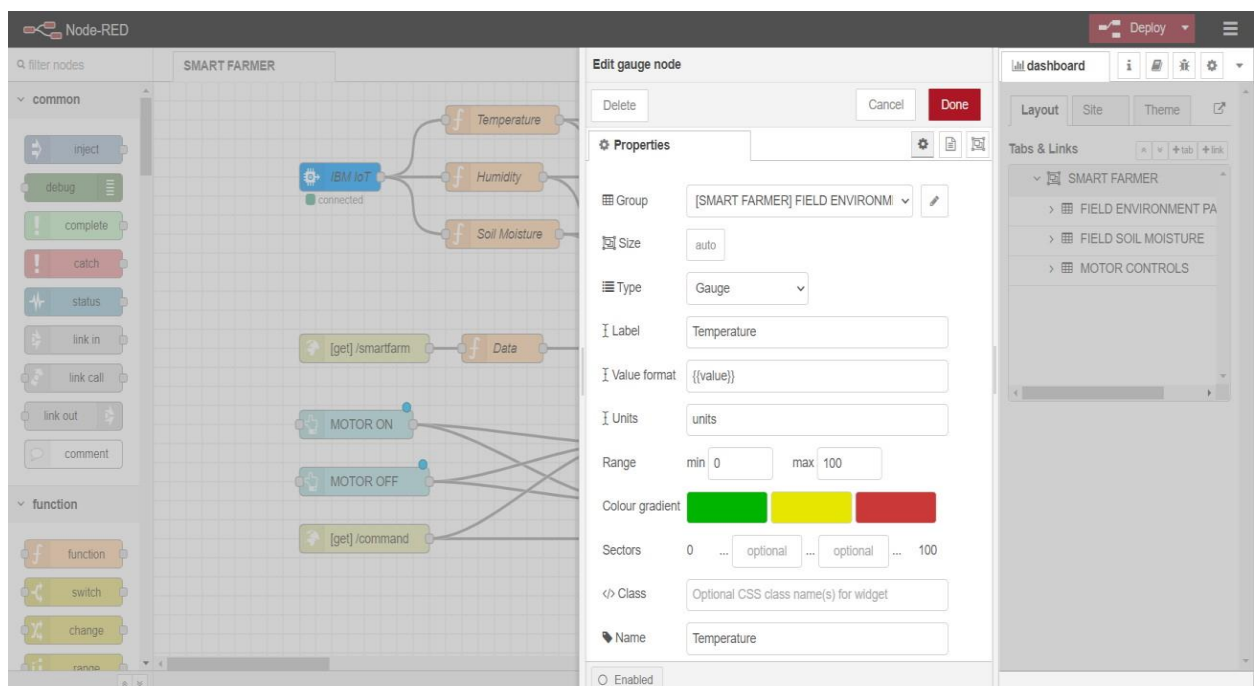
```
{ "speed":2.3,"deg":113,"gust":3.05},"clouds":{"all":97},"dt":1668332957
,"sys":{"country":"IN","sunrise":1668300334,"sunset":1668342165},"tim
ezone":19800,"id":1270947,"name":"Gobichettipalayam","cod":200}
```

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

```
msg.payload = { "temp" : global.get("t") ,
                "Humid" : global.get("h") ,
                "Moisture" : global.get("m")
              }

return msg;
```

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



- You can the data in the node-red dashboard.

