

Sprint - 2

Project Title: SmartFarmer - IoT Enabled Smart Farming Application

Team ID: PNT2022TMID00236

Connecting IoT Simulator to IBM Watson IoT Platform

Open link provided in above section

Give the credentials of your device in IBM Watson IoT Platform

My credentials given to simulator are:

api: s-xanmsr-dqrgvealm

Device type: iotSensor

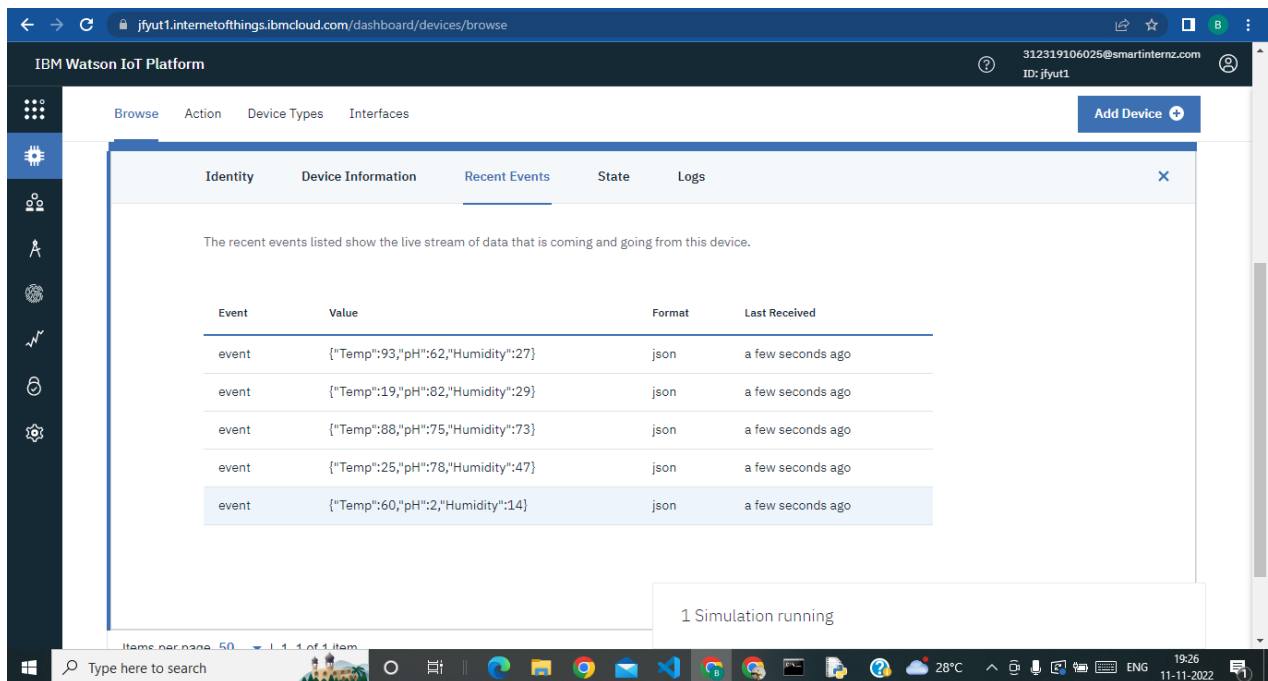
Token: 12345678

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)

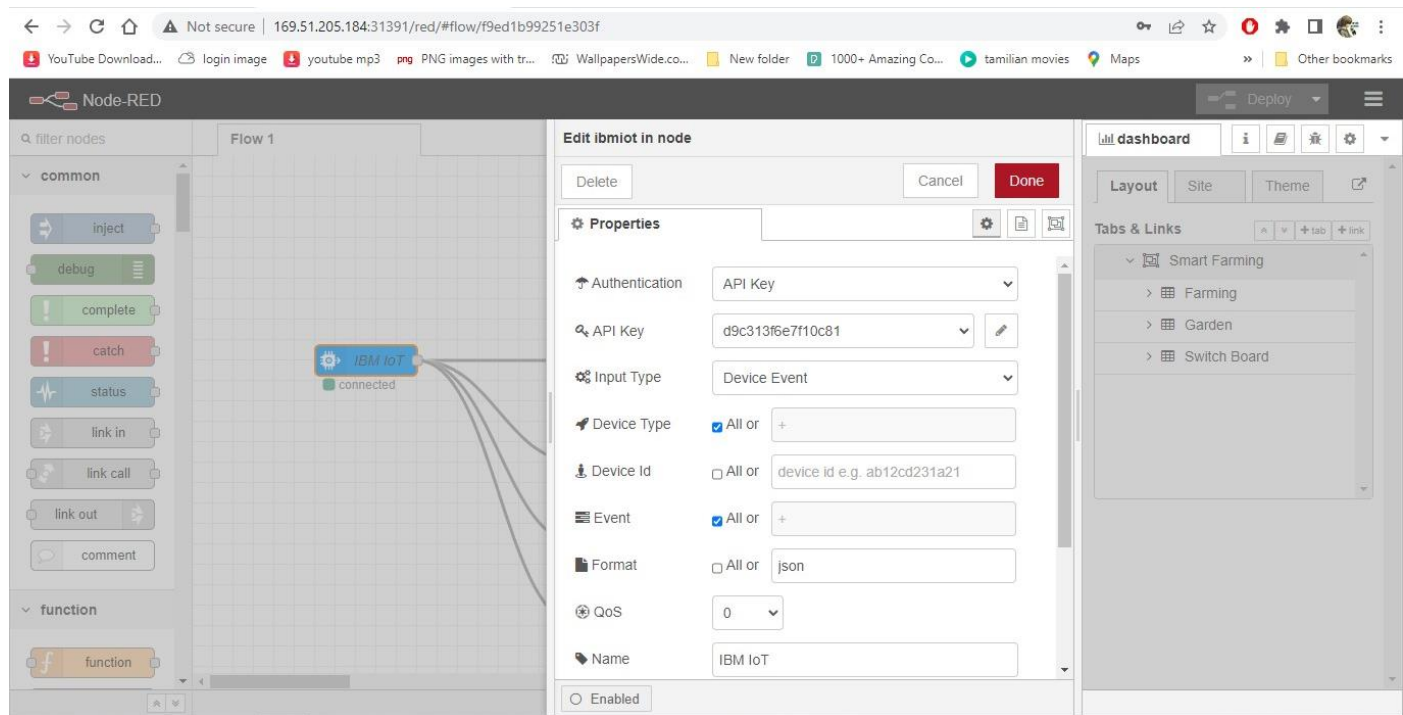
```
{  
  "d": {  
    ▪ "name": "abcd",  
    ▪ "temperature": 17,  
    ▪ "humidity": 76,  
    ▪ "Moisture ": 25  
  }  
}
```



The screenshot shows the IBM Watson IoT Platform dashboard. The 'Recent Events' tab is selected, displaying a table of live data events. The table has four columns: Event, Value, Format, and Last Received. There are five rows of data, each representing an event received 'a few seconds ago'. The values are JSON objects containing temperature, pH, and humidity data. A notification at the bottom right indicates '1 Simulation running'.

Event	Value	Format	Last Received
event	{"Temp":93,"pH":62,"Humidity":27}	json	a few seconds ago
event	{"Temp":19,"pH":82,"Humidity":29}	json	a few seconds ago
event	{"Temp":88,"pH":75,"Humidity":73}	json	a few seconds ago
event	{"Temp":25,"pH":78,"Humidity":47}	json	a few seconds ago
event	{"Temp":60,"pH":2,"Humidity":14}	json	a few seconds ago

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

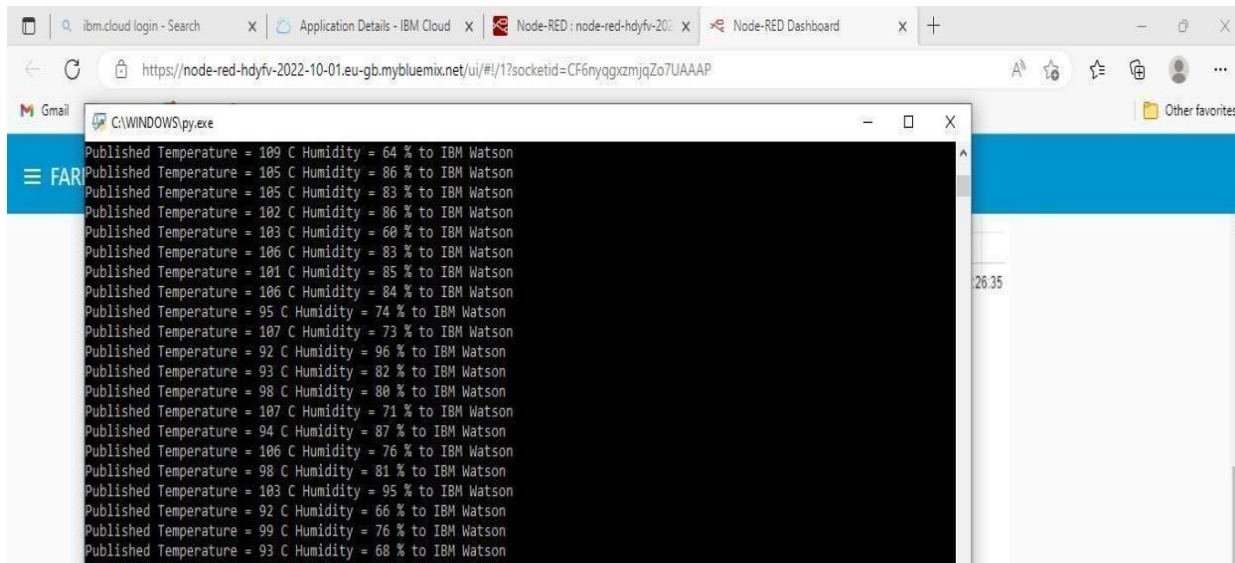
Display the data using debug node for verification

Connect function node and write the Java script code to get each reading separately.

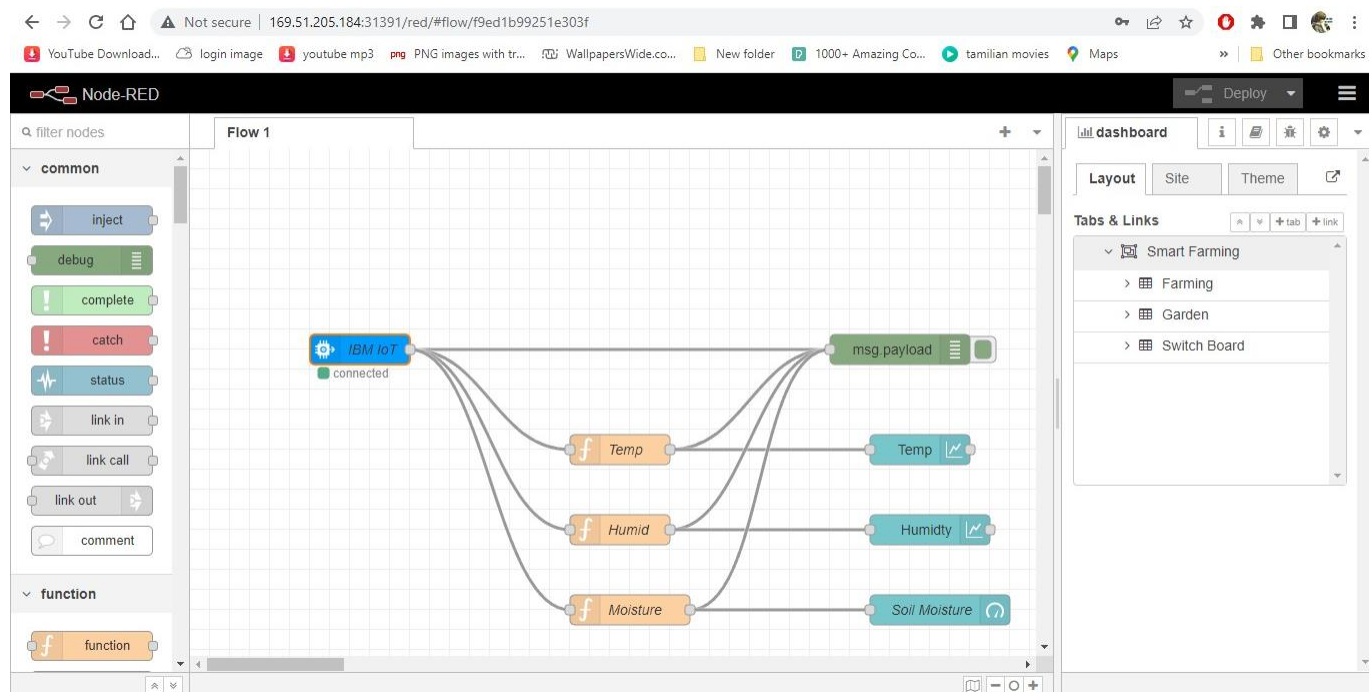
The Java script code for the function node is:

```
msg.payload=msg.payload.d.temperature return msg;
```

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



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 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 data we receive from OpenWeather after request is in below JSON

```
format:{"coord":{"lon":79.85,"lat":14.13},"weather":[{"id":803,"main":"Clouds","description":"brokenclouds","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,"grnd_level":1000},"wind":{"speed":6.23,"deg":170}
,"clouds":{"all":68},"dt":1589991979,"sys":{"country":"IN","sunrise":1589933553,
```

```
"sunset":1589979720},"timezone":19800,"id":1270791,"name":"Gūdūr","cod":200}
```

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

