TEAM ID: PNT2022TMID51214

Smart Farmer -IOT Enabled Smart

FarmingApplication

# 5. Building Project

**SPRINT DELIVERY–2**

Connecting IOT Simulator to IBM Watson IOT Platform

Open link provided in above section 4.3 Give the credentials of your device in IBM Watson IOT Platform Click on connect

My credentials given to simulator are:

OrgID**: bnkhnl**

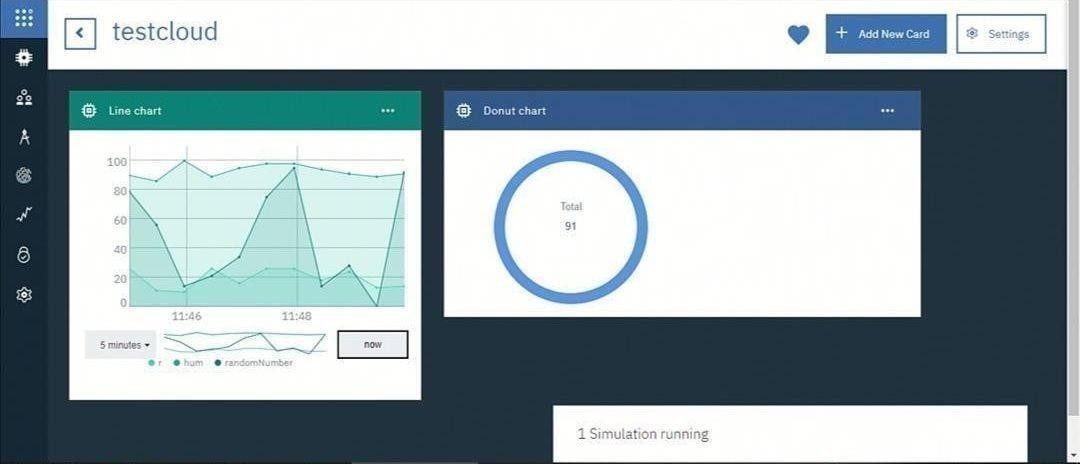
api: **a-bnkhnl-veyruwypcb**

Device type:**NodeMCU**

token: **KOueb8NWLNUKN422ZM**

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

* You will receive the simulator data in cloud



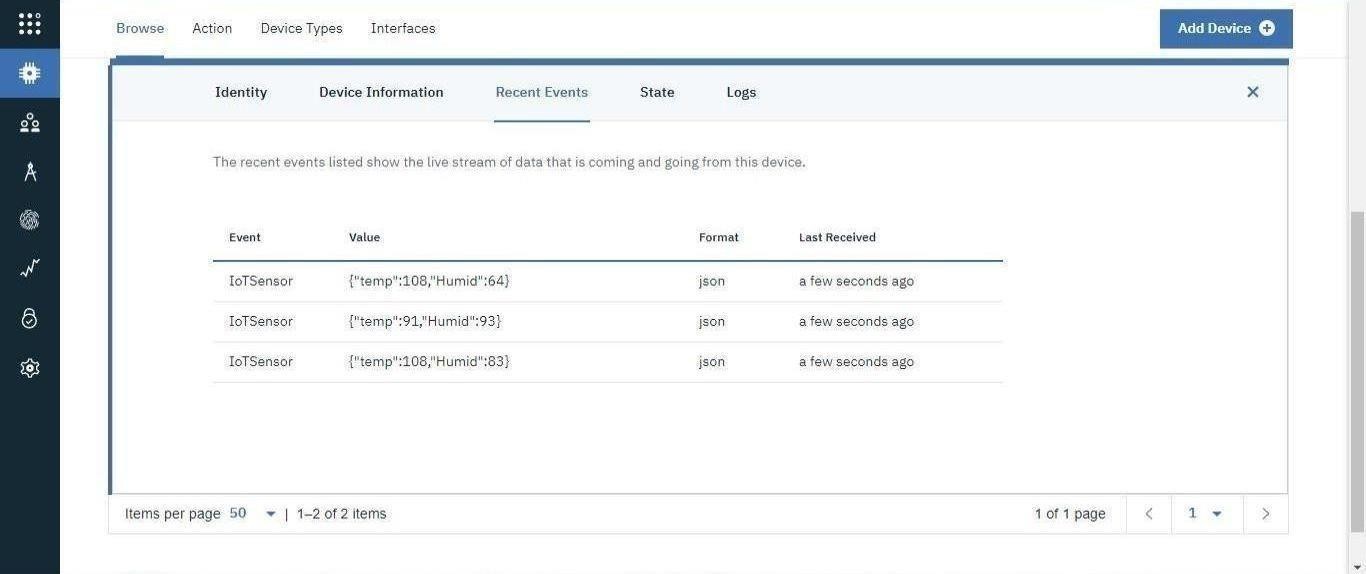
* You can see the received data in Recent Events under your device
* Data received in this format(json)

{

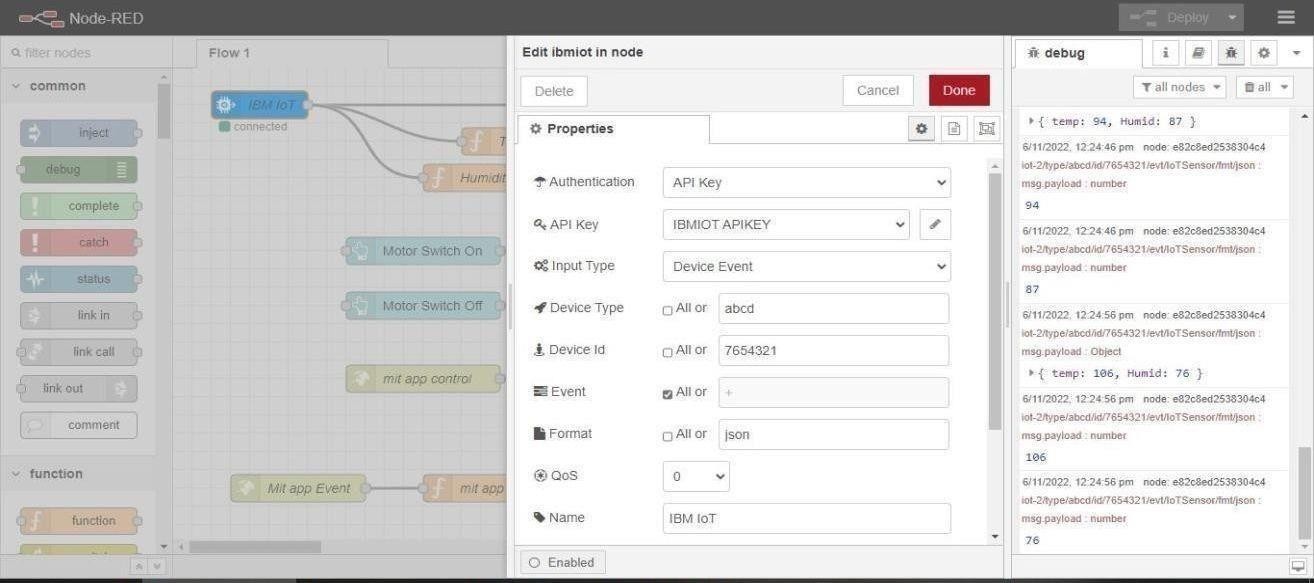
"d": {

* "name": "NodeMCU",
* "temperature": 17,
* "humidity": 76,
* "Moisture ": 25

}

}

# Configuration of Node-Red to collect IBM cloud data

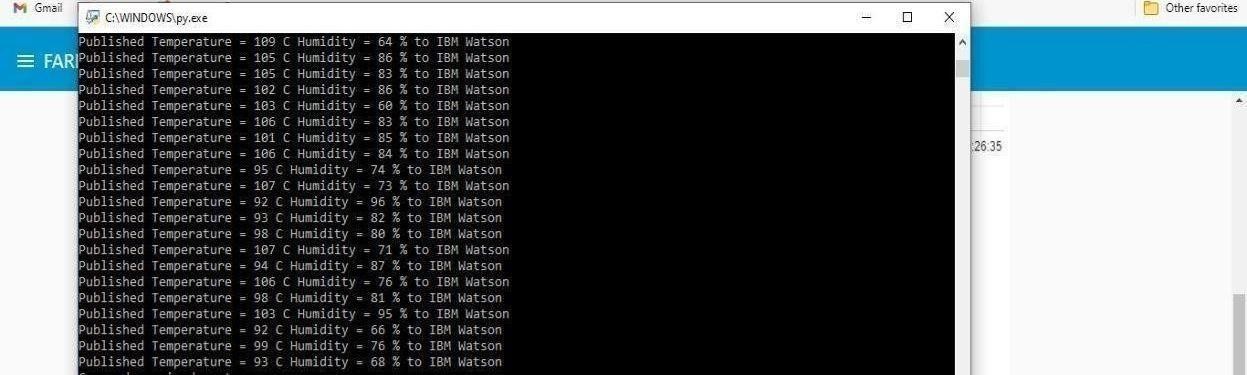


The node IBM IOT App In is added to Node-Red workflow. Then the appropriate device credentials obtained earlier are entered into the node toconnect and fetch device telemetry to Node-Red.

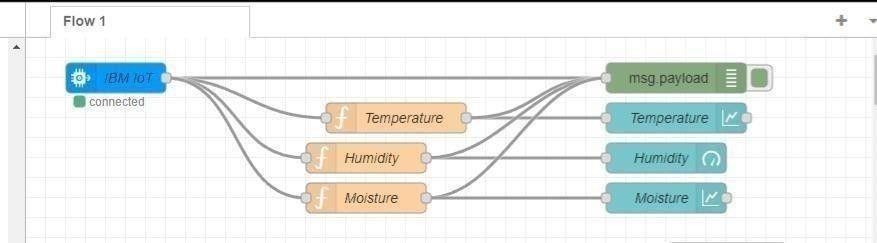
Once it is connected Node-Red receives data from the device Display the data using debug node for verification

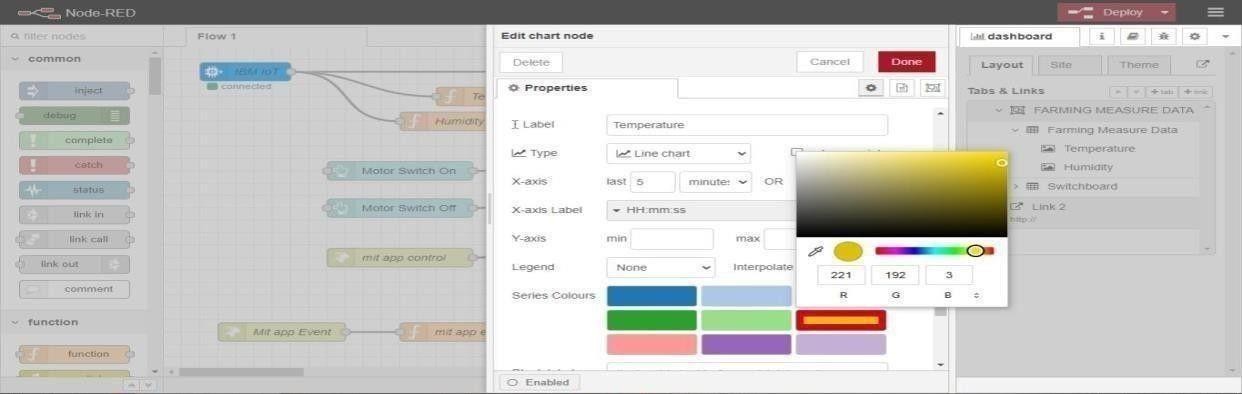
Connect function node and write the Java script code to get each reading separately. The Java script code for the function node is:

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

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 we saved before in section 4.4 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":"brokenclouds","icon":"04n"}],"base":"stations","main":{"temp":307 59,"feels\_like":305.5,"temp\_min":307.59,"temp\_max":307.59,"pressure":1002,"h

umidity":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":20 0}

In order to parse the JSON string we use Java script functions and get eachparameters 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 variableand convert it from kelvin to Celsius

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

