Smart Farmer-IOT Enabled Smart Farming Application

SPRINT – 2

TITLE	**Smart Farmer-IOT Enabled Smart Farming Application**
DOMAIN NAME	INTERNET OF THINGS
TEAM ID	PNT2022TMID20449
LEADER NAME	KARTHIKA A
TEAM MEMBER NAME	KAVIYA SATHYA HARI KARA SANKAR SURYA
MENTOR NAME	VIVEK ANAND I

Building Project

Connecting IoT Simulator to IBM Watson IoT Platform

Give the credentials of your device in IBM Watson IoT PlatformClick on connect

My credentials given to simulator are:

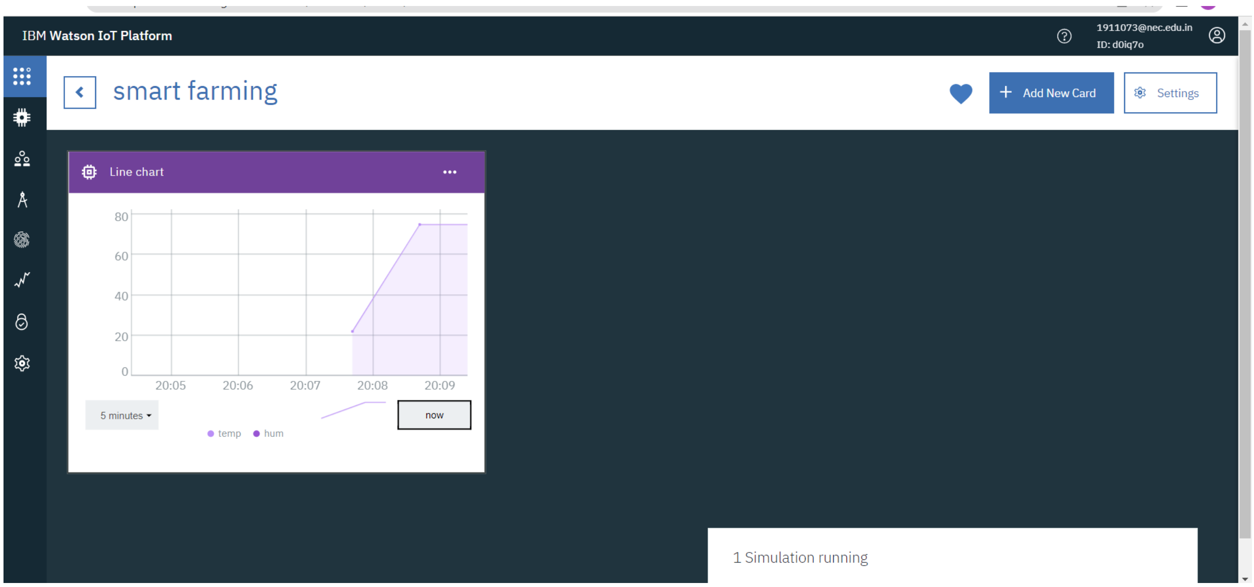
Organization ID: d0iq7o

api: a-d0iq7o-zwfxngqogr

Device type: IOT_DEV

Token: 1911073abcdefgh

Device ID : 1911073



- ➢ 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)

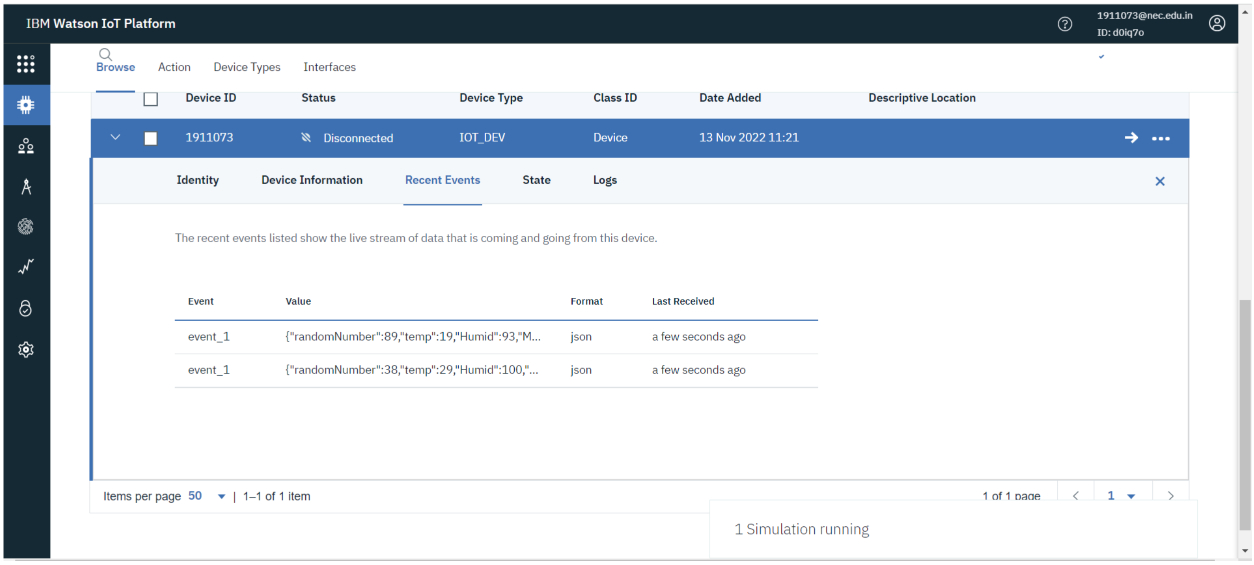
{

"d": {

- "name": "1911073",
- "temperature": 25,
- "humidity": 80,
- "Moisture ": 35

}

}



□	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
□	1911073	Disconnected	IOT_DEV	Device	13 Nov 2022 11:21	

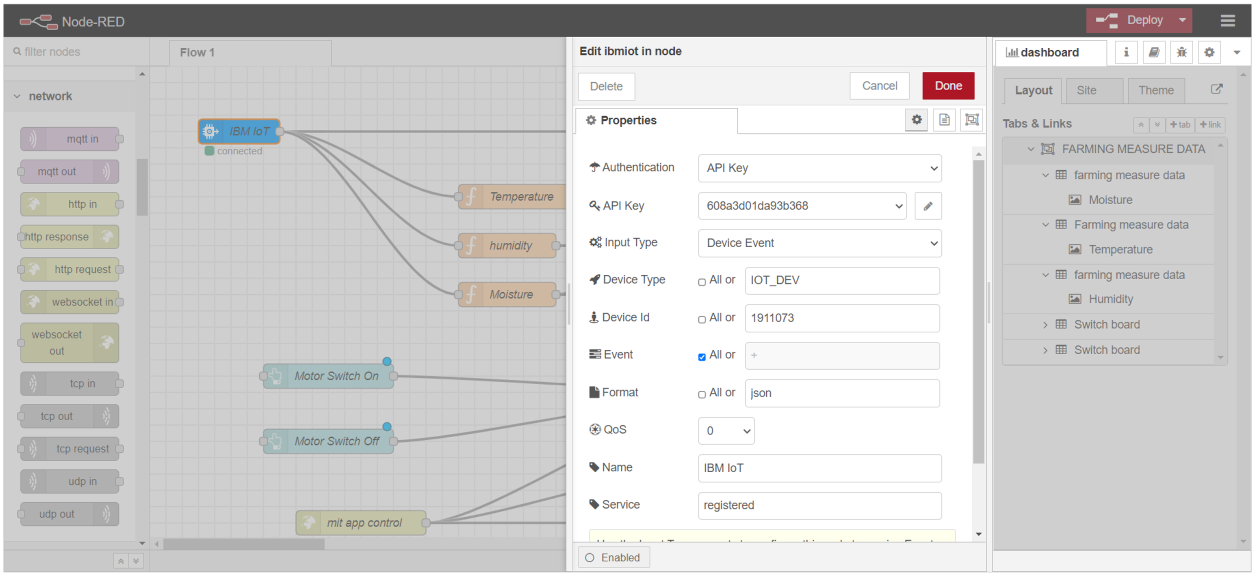
The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
event_1	{"randomNumber":89,"temp":19,"Humid":93,"M...	json	a few seconds ago
event_1	{"randomNumber":38,"temp":29,"Humid":100,"...	json	a few seconds ago

Items per page 50 | 1–1 of 1 item 1 of 1 page 1

Configuration of Node-Red to collect IBM cloud data

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



Edit ibmiot in node

Properties

Authentication: API Key

API Key: 608a3d01da93b368

Input Type: Device Event

Device Type: □ All or IOT_DEV

Device Id: □ All or 1911073

Event: ☑ All or +

Format: □ All or json

QoS: 0

Name: IBM IoT

Service: registered

○ Enabled

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.

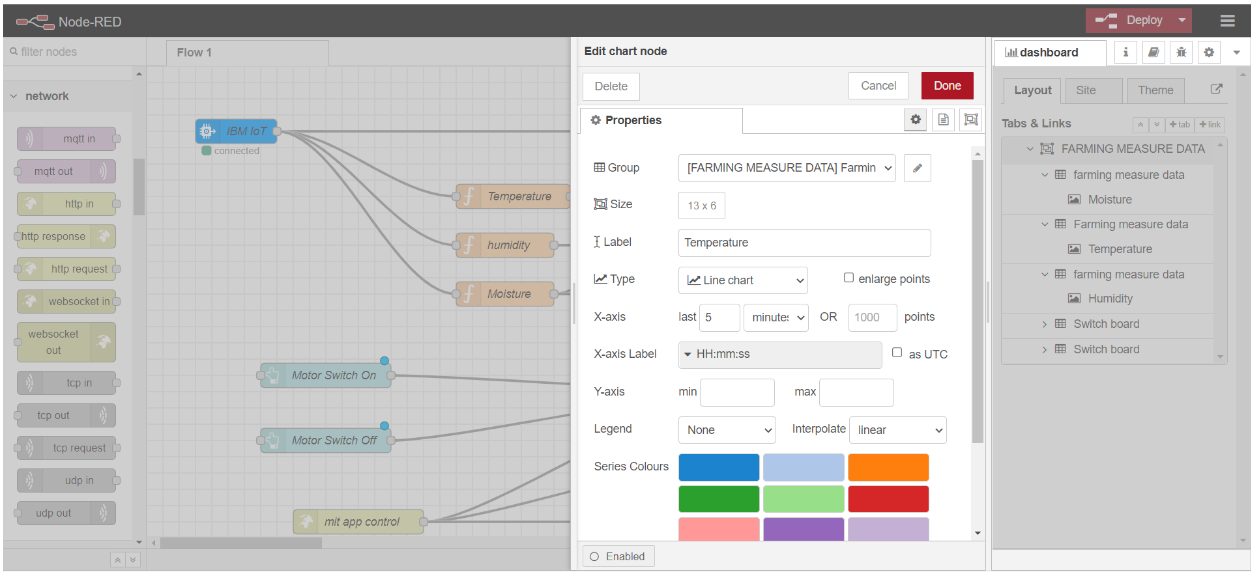
The Java script code for the

function node is:

msg.payload=msg.payload.d.te

mperature returnmsg;

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



Group [FARMING MEASURE DATA] Farmin

Size 13 x 6

Label Temperature

Type Line chart □ enlarge points

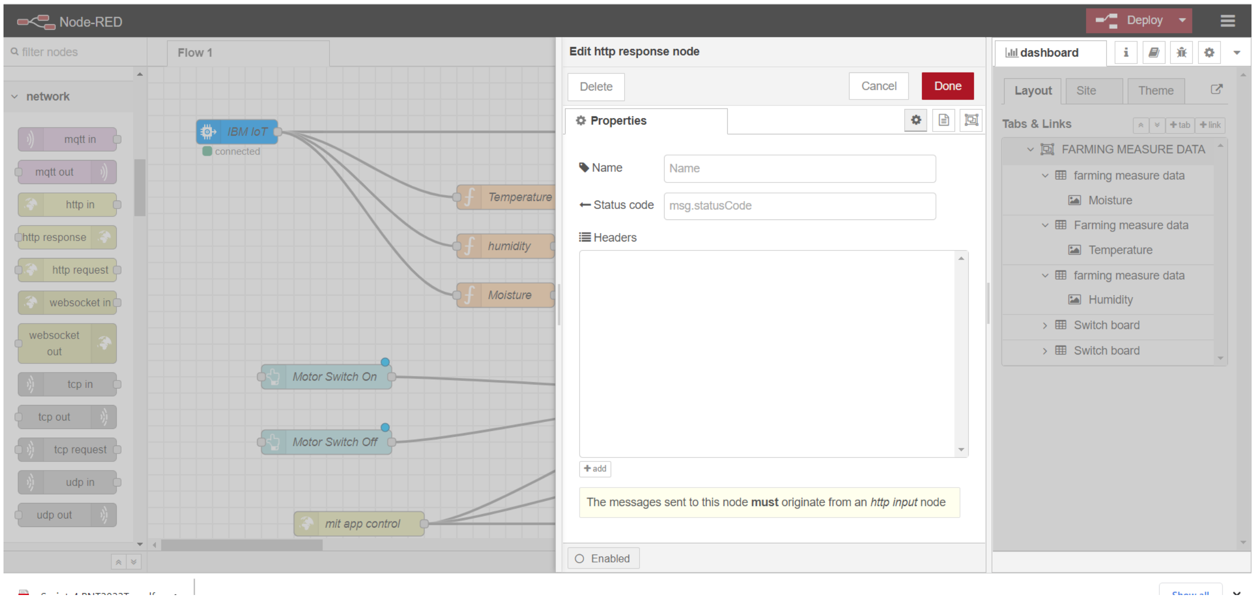
X-axis last 5 minutes OR 1000 points

X-axis Label HH:mm:ss □ as UTC

Y-axis min max

Legend None Interpolate linear

Series Colours 🟦 🟦 🟧 🟩 🟩 🟥 🟥 🟪 🟪



Name Name

Status code msg.statusCode

Headers

The messages sent to this node **must** originate from an *http input* node

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":"Cl
ouds","

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":{"s
peed":6.23,"deg":170

}

,"clouds":{"all":68},"dt":1589991979,"sys":{"country":"IN","sunrise":158
9933553
,
"sunset":1589979720},"timezone":19800,"id":1270791,"na
me":"Gūdūr","cod":20 0}
```

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

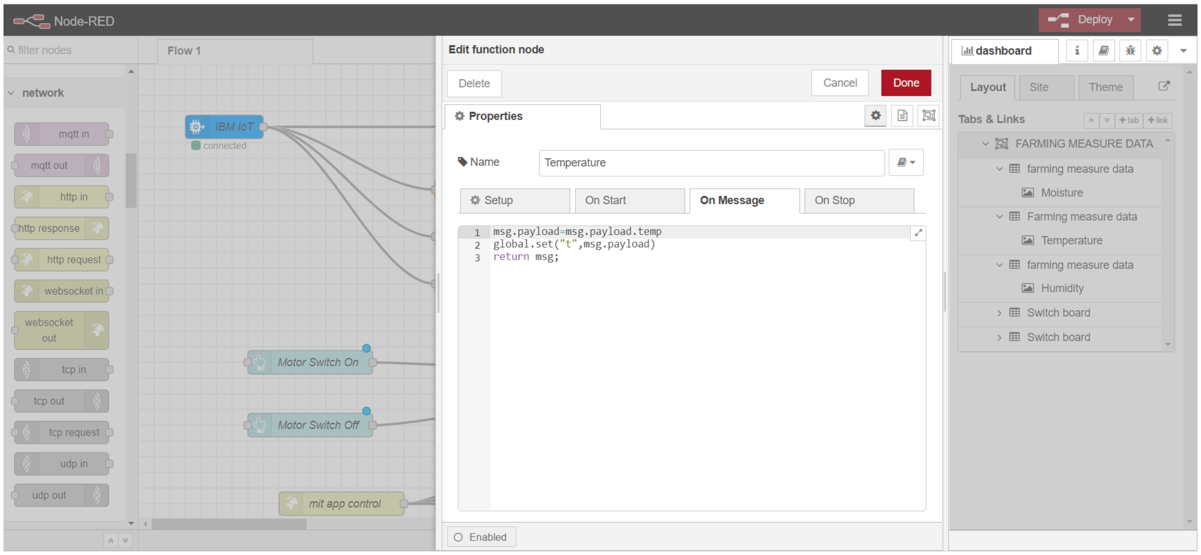
```
        var temperature = msg.payload.main.temp;temperature

        = temperature-273.15;

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

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

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



Edit function node

Delete | Cancel | Done

Properties

Name: Temperature

Setup | On Start | On Message | On Stop

```
msg.payload=msg.payload.temp
global.set("t",msg.payload)
return msg;
```

○ Enabled

Device Token : **87654321**