

# **IOT ENABLED SMART FARMING APPLICATION**

**SPRINT DELIVERY – 2**

## 5. Building Project

### 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: **x0cl0i**

api: **a-157uf3- f5rg4qxp3**

Device type: **nodemcu**

token: **6ogMaaQHNWFEgOD8R?**

Device ID : **sensor**

Device Token : **6GsCaVQ3-PfYy+J3ts**

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": 67,
    "humidity": 20,
    "Moisture ": 28
  }
}
```

The screenshot displays the IBM Watson IoT Platform interface. At the top, the header shows 'IBM Watson IoT Platform' and a user profile with email '412519106110@smartinternz.com' and ID 'x0cl0i'. Below the header, there's a navigation bar with tabs: 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area shows a table of recent events with the following columns: 'Event', 'Value', 'Format', and 'Last Received'. The table contains five rows of event data. Below the table, there's a status bar indicating '1 Simulation running' and a pagination control showing 'Items per page 50' and '1-1 of 1 item'.

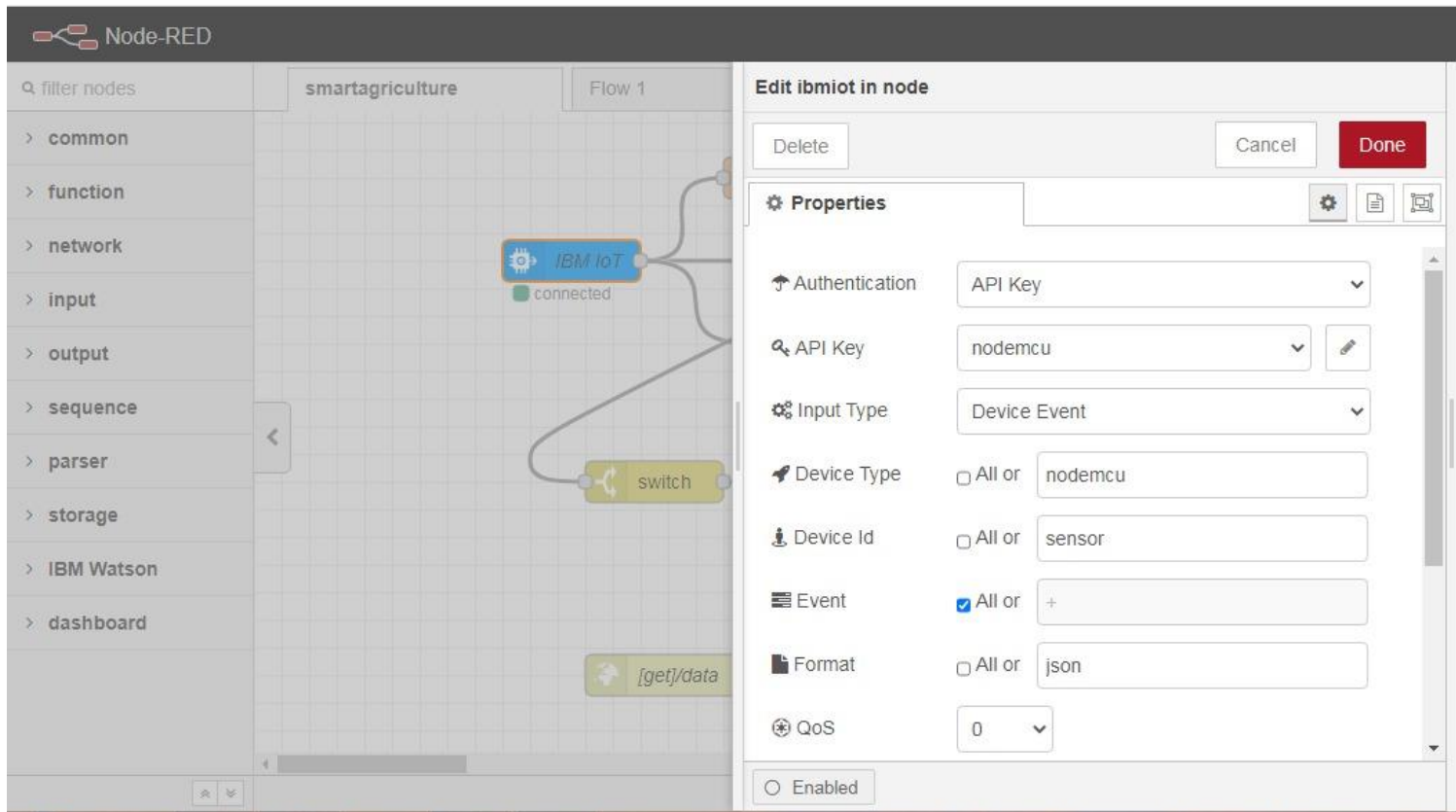
Event	Value	Format	Last Received
event_1	{"tempetrature":54,"humidity":17,"soil moisture"...	json	a few seconds ago
event_1	{"tempetrature":12,"humidity":96,"soil moisture"...	json	a few seconds ago
event_1	{"tempetrature":74,"humidity":85,"soil moisture"...	json	a few seconds ago
event_1	{"tempetrature":72,"humidity":33,"soil moisture"...	json	a few seconds ago
event_1	{"tempetrature":8,"humidity":72,"soil moisture"...	json	a few seconds ago

1 Simulation running

Items per page 50 | 1-1 of 1 item

## 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

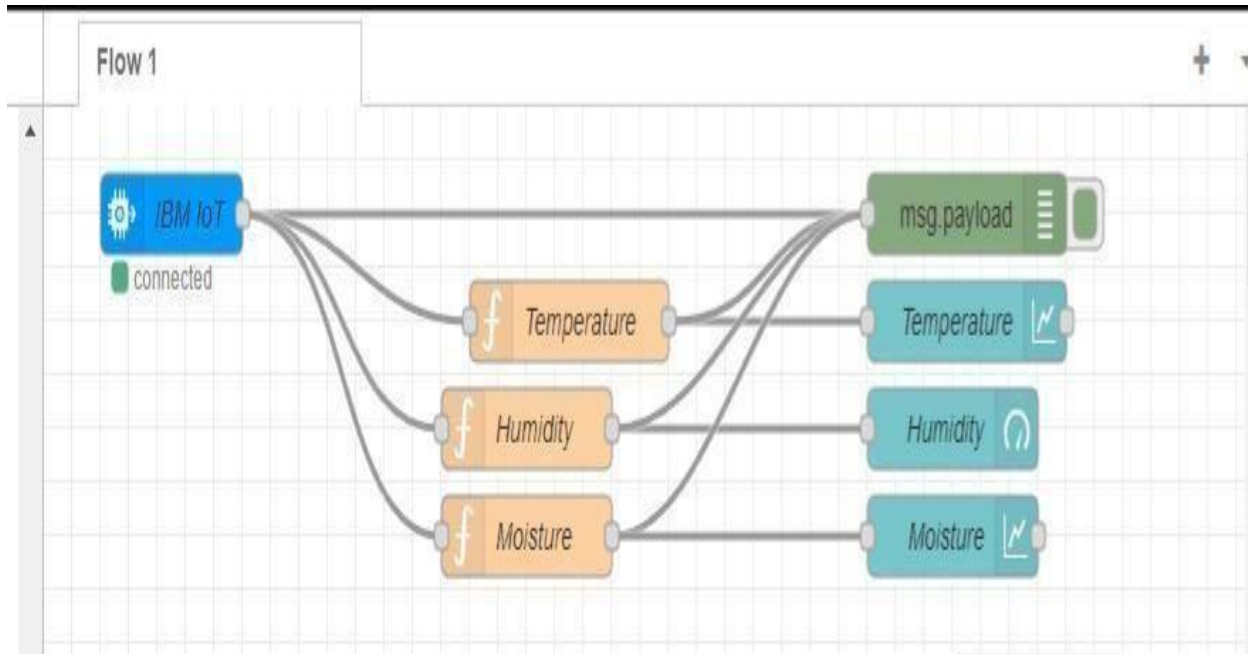
Connect function node and write the Java script code to get each reading separately.

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 separate.

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

The screenshot shows the Node-Red interface. On the left, the 'network' tab is selected, showing various nodes like 'mqtt in', 'http in', 'websocket in', etc. In the center, the 'Edit function node' dialog is open. The 'Name' field is set to 'temperature'. The 'On Message' tab is selected, and the following JavaScript code is entered:

```
1 global.set('temp',msg.payload.temperature)
2 msg.payload=msg.payload.temperature
3 return msg;
```

On the right, a help panel is visible, showing the 'function' node description: 'A JavaScript function to run against the messages being received by the node. The messages are passed in as a JavaScript object called msg. By convention it will have a msg.payload property containing the body of the message. The function is expected to return a message object (or multiple message objects), but can choose to return nothing in order to halt a flow.'