Sprint 2

Date	16 November 2022
Team ID	PNT2022TMID45187
Project Name	Smart Farmer-IoT Enabled smart Farming Application
Maximum Marks	4 Marks

INTRODUCTION:

The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature, soil moisture, humidity and etc and control the equipment like water motor and other devices remotely via internet without their actual presencein the field.

Sprint-2

Software (Create device in the IoT Watson platform, workflow for IoT scenarios using Node-Red)

Steps to configure:

- Create a IBM Cloud account using the instructions given by tutorial instructors / watching the manual in the Drive Documents.
- Then Login to the IBM Cloud and it open to the Dashboard.

- Create a IoT Service and save the individual credential shown in the display.
- Launch the IBM IoT Watson Platform and Create a new device and save the credential for future references.
- Add new device and code the program for the module / project and save it.
- Finally Simulate using the device simulator and create the board to show the data’s in output of the eventflow.
- Then create a Node-Red flow using the required credential like shown in the tutorial as a reference.
- Connect the Node-Red with IBM IoT Watson with the required API Key information and deploy the Node-Red for the Data to show.

Connecting IoT Simulator to IBM Watson IoT Platform:

My credentials given to simulator are:

Organization ID:bj6xkv

API: a-bj6xkv-w2fdwqb9ku

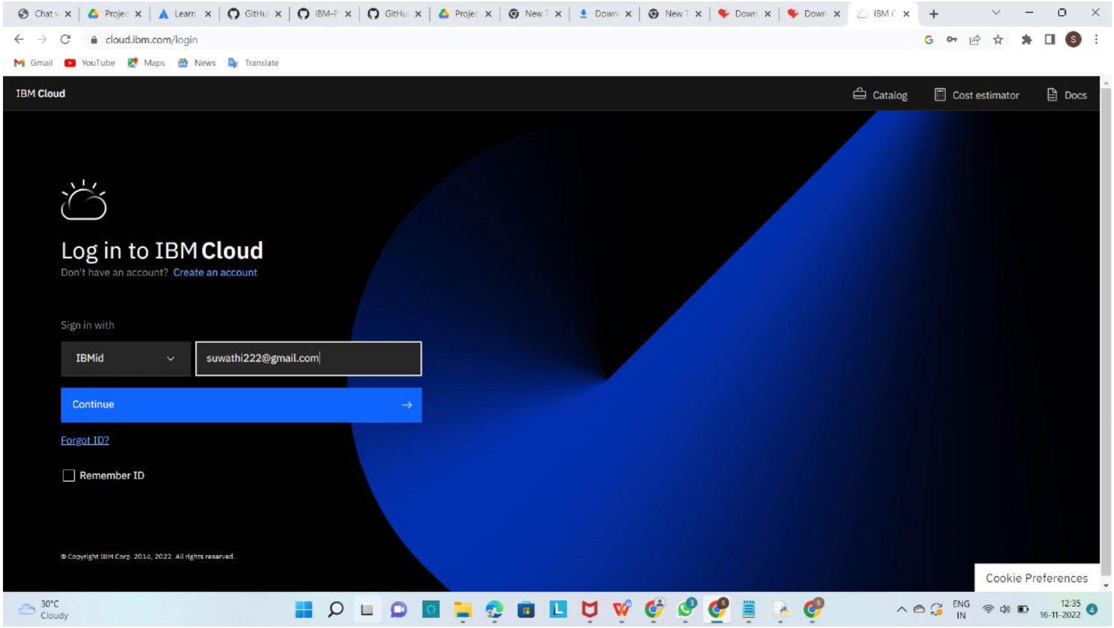
AuthenticationMethod:5+d3_THFhyI*Voaw9T

Device Type:nodemcu

Device ID:12345

Device Token:12345678

THE PROCESS:



Log in to IBM Cloud

Don't have an account? Create an account

Sign in with

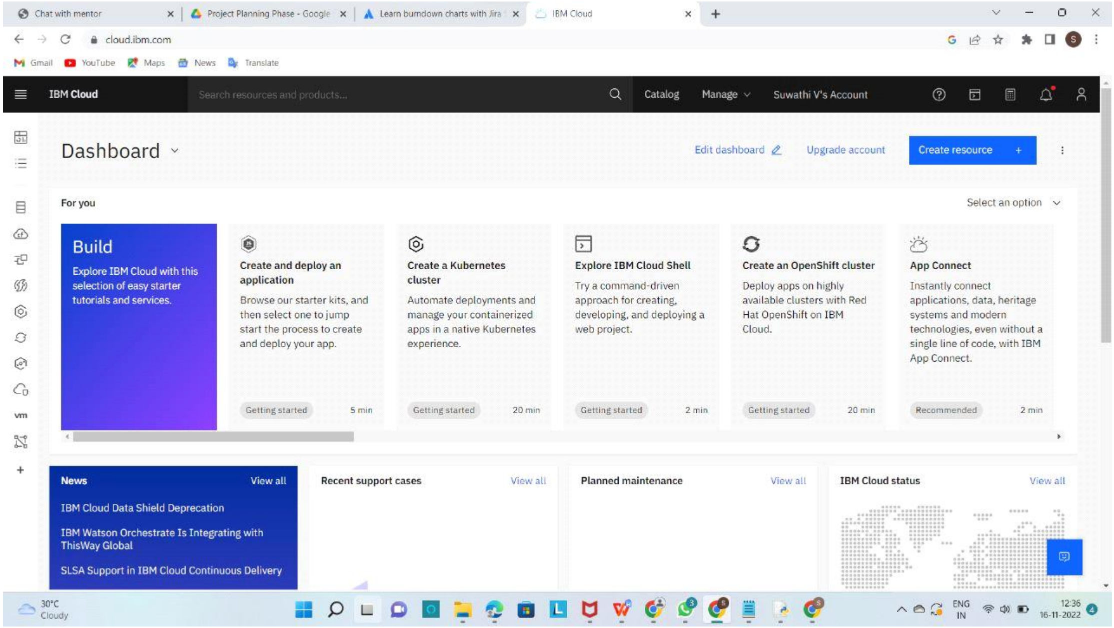
IBMid: suwathi222@gmail.com

Continue

Forgot ID?

□ Remember ID

© Copyright IBM Corp. 2014, 2022. All rights reserved.



Dashboard

For you

Build
Explore IBM Cloud with this selection of easy starter tutorials and services.

Create and deploy an application
Browse our starter kits, and then select one to jump start the process to create and deploy your app.
Getting started 5 min

Create a Kubernetes cluster
Automate deployments and manage your containerized apps in a native Kubernetes experience.
Getting started 20 min

Explore IBM Cloud Shell
Try a command-driven approach for creating, developing, and deploying a web project.
Getting started 2 min

Create an OpenShift cluster
Deploy apps on highly available clusters with Red Hat OpenShift on IBM Cloud.
Getting started 20 min

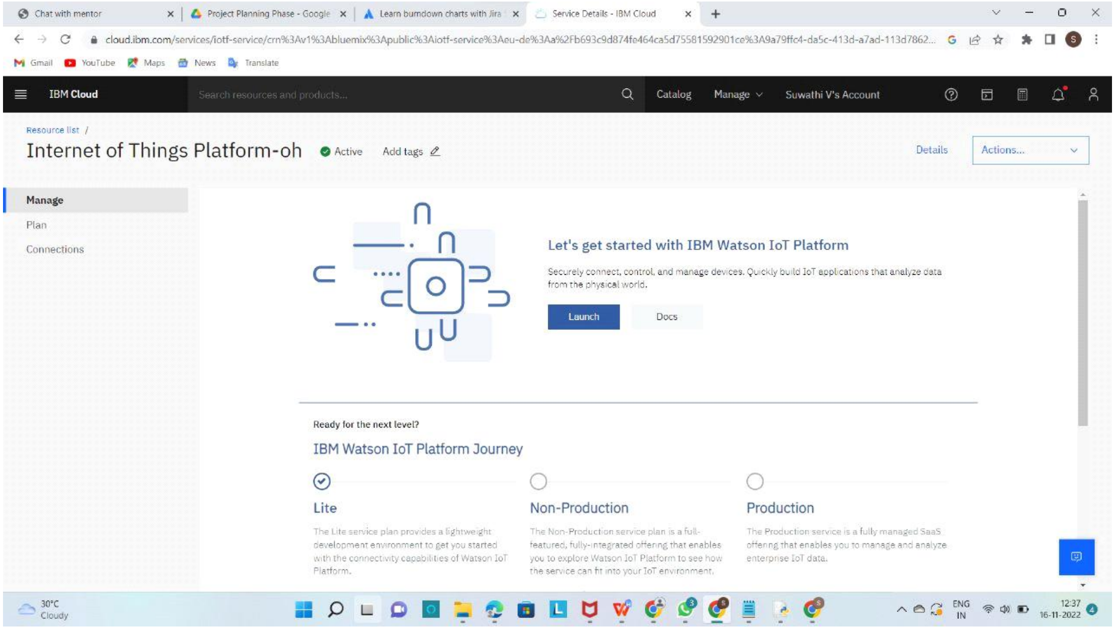
App Connect
Instantly connect applications, data, heritage systems and modern technologies, even without a single line of code, with IBM App Connect.
Recommended 2 min

News
IBM Cloud Data Shield Deprecation
IBM Watson Orchestrate Is Integrating with ThisWay Global
SLSA Support in IBM Cloud Continuous Delivery

Recent support cases

Planned maintenance

IBM Cloud status


Chat with mentor
Project Planning Phase - Google
Learn burndown charts with Jira
Service Details - IBM Cloud
cloud.ibm.com/services/iotf-service/crn%3Av1%3Abluemix%3Apublic%3Aiotf-service%3Aeu-de%3Aa%2Fb693c9d874fe464ca5d7558159290 1ce%3A9a79ffc4-da5c-413d-a7ad-113d7862...
Gmail
YouTube
Maps
News
Translate
IBM Cloud
Search resources and products...
Catalog
Manage
Suwathi V's Account
Resource list /
Active
Add tags
Details
Actions...
Manage
Plan
Connections
Launch
Docs
30°C
Cloudy
ENG
IN
12:37
16-11-2022

Internet of Things Platform-oh

Let's get started with IBM Watson IoT Platform

Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.

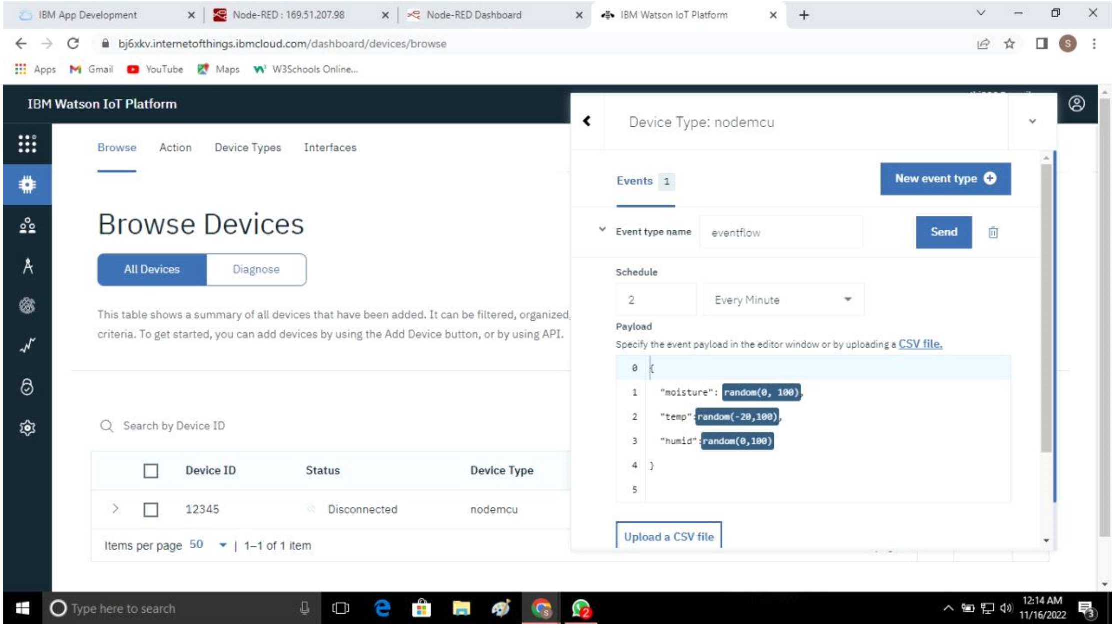
Ready for the next level?

IBM Watson IoT Platform Journey

Lite
The Lite service plan provides a lightweight development environment to get you started with the connectivity capabilities of Watson IoT Platform.

Non-Production
The Non-Production service plan is a full-featured, fully-integrated offering that enables you to explore Watson IoT Platform to see how the service can fit into your IoT environment.

Production
The Production service is a fully managed SaaS offering that enables you to manage and analyze enterprise IoT data.



IBM Watson IoT Platform

Browse | Action | Device Types | Interfaces

Browse Devices

All Devices | Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device ID	Status	Device Type
12345	Disconnected	nodemcu

Items per page 50 | 1–1 of 1 item

Device Type: nodemcu

Events 1 | New event type

Event type name: eventflow | Send

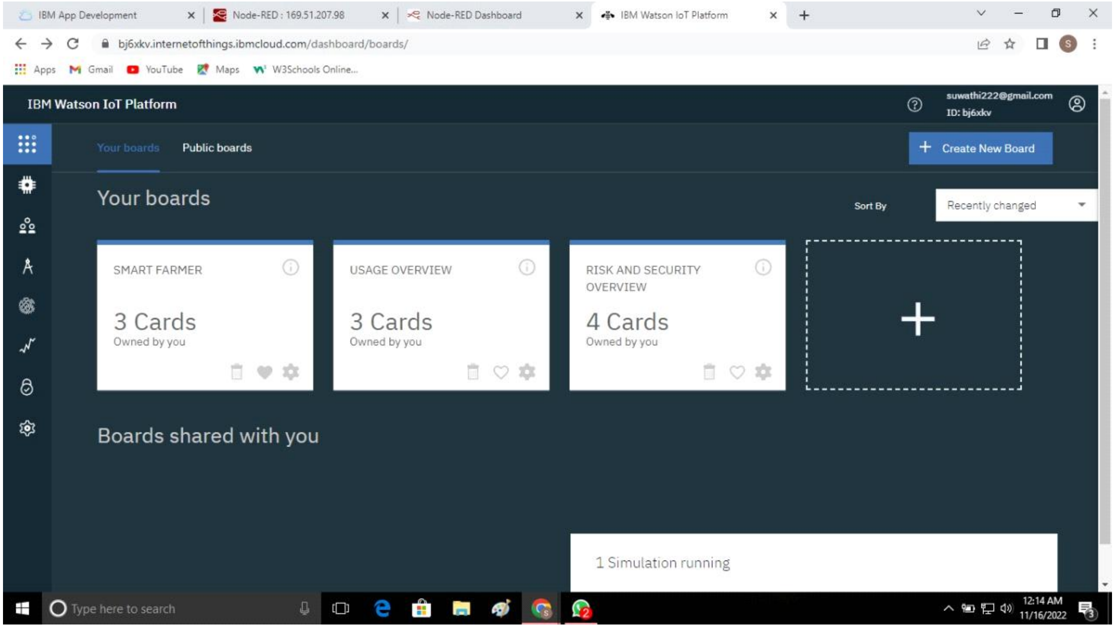
Schedule: 2 | Every Minute

Payload

Specify the event payload in the editor window or by uploading a CSV file.

```
{
  "moisture": random(0, 100),
  "temp":random(-20,100),
  "humid":random(0,100)
}
```

Upload a CSV file



IBM Watson IoT Platform

suwathi222@gmail.com
ID: bj6xkv

Your boards | Public boards | Create New Board

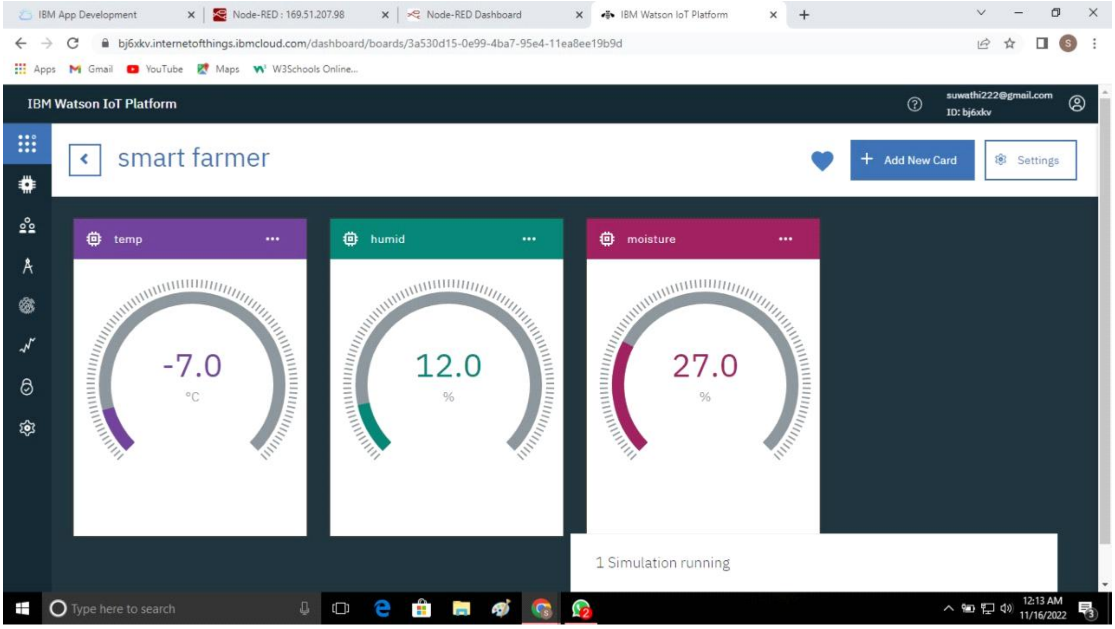
Your boards

Sort By: Recently changed

Board	Cards	Owner
SMART FARMER	3 Cards	Owned by you
USAGE OVERVIEW	3 Cards	Owned by you
RISK AND SECURITY OVERVIEW	4 Cards	Owned by you

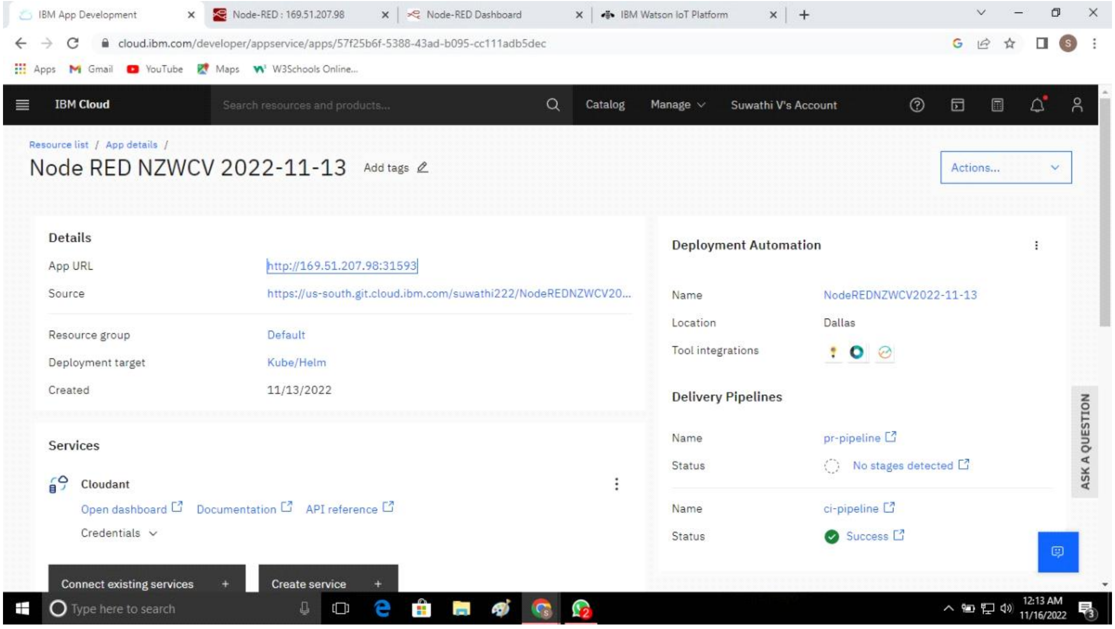
Boards shared with you

1 Simulation running



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 yourdevice
- Data received in this format(json)



Node RED NZWCV 2022-11-13

Details

App URL	http://169.51.207.98:31593
Source	https://us-south.git.cloud.ibm.com/suwathi222/NodeREDNZWCV20...
Resource group	Default
Deployment target	Kube/Helm
Created	11/13/2022

Services

Cloudant

Open dashboard · Documentation · API reference

Credentials

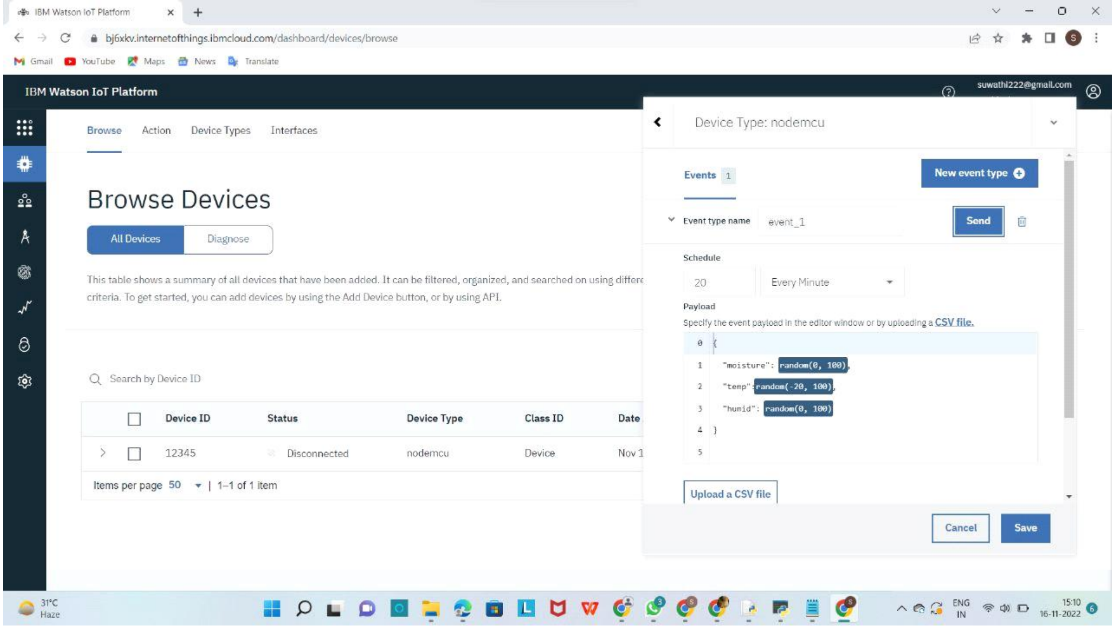
Deployment Automation

Name	NodeREDNZWCV2022-11-13
Location	Dallas
Tool integrations	

Delivery Pipelines

Name	pr-pipeline
Status	No stages detected
Name	ci-pipeline
Status	Success

Configuring IBM-IoT to Node-RED connection:



Browse Devices

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using differe criteria. To get started, you can add devices by using the Add Device button, or by using API.

Device ID	Status	Device Type	Class ID	Date
12345	Disconnected	nodemcu	Device	Nov 1

Device Type: nodemcu

Events 1

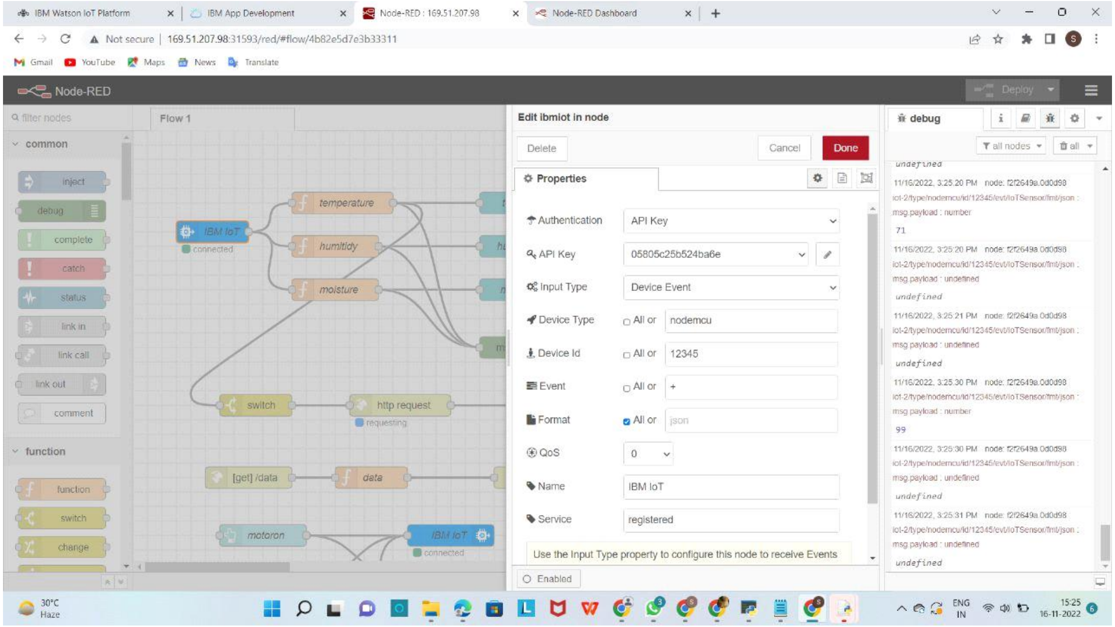
Event type name: event_1

Schedule: 20 Every Minute

Payload

Specify the event payload in the editor window or by uploading a CSV file.

```
{
  "moisture": random(0, 100),
  "temp":random(-20, 100),
  "humid": random(0, 100)
}
```



Edit ibmiot in node

Delete | Cancel | Done

Properties

Authentication: API Key

API Key: 05805c25b524ba6e

Input Type: Device Event

Device Type: □ All or nodemcu

Device Id: □ All or 12345

Event: □ All or +

Format: ☑ All or json

QoS: 0

Name: IBM IoT

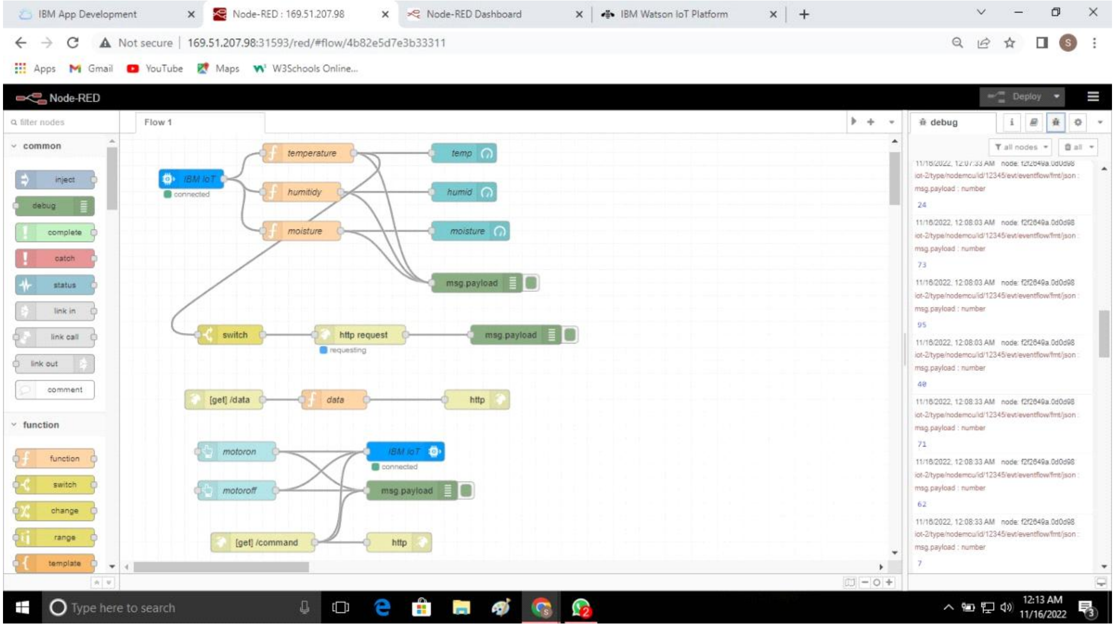
Service: registered

Use the Input Type property to configure this node to receive Events

○ Enabled

```
undefined
11/16/2022, 3:25:20 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
71
11/16/2022, 3:25:20 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:25:21 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:25:30 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
99
11/16/2022, 3:25:30 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:25:31 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
```

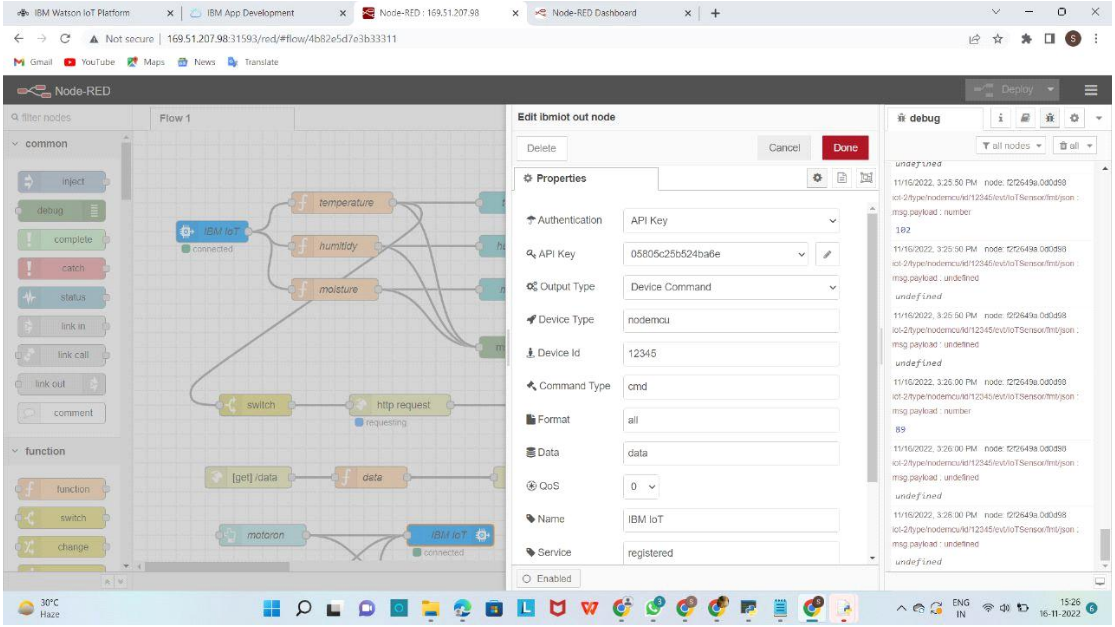
Complete Program Flow:



```
11/16/2022, 12:07:33 AM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/eventflow/fmt/json :
msg.payload : number
24
11/16/2022, 12:08:03 AM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/eventflow/fmt/json :
msg.payload : number
73
11/16/2022, 12:08:03 AM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/eventflow/fmt/json :
msg.payload : number
95
11/16/2022, 12:08:03 AM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/eventflow/fmt/json :
msg.payload : number
40
11/16/2022, 12:08:33 AM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/eventflow/fmt/json :
msg.payload : number
71
11/16/2022, 12:08:33 AM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/eventflow/fmt/json :
msg.payload : number
62
11/16/2022, 12:08:33 AM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/eventflow/fmt/json :
msg.payload : number
7
```

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.



Edit ibmiot out node

Authentication: API Key
API Key: 05805c25b524ba6e
Output Type: Device Command
Device Type: nodemcu
Device Id: 12345
Command Type: cmd
Format: all
Data: data
QoS: 0
Name: IBM IoT
Service: registered

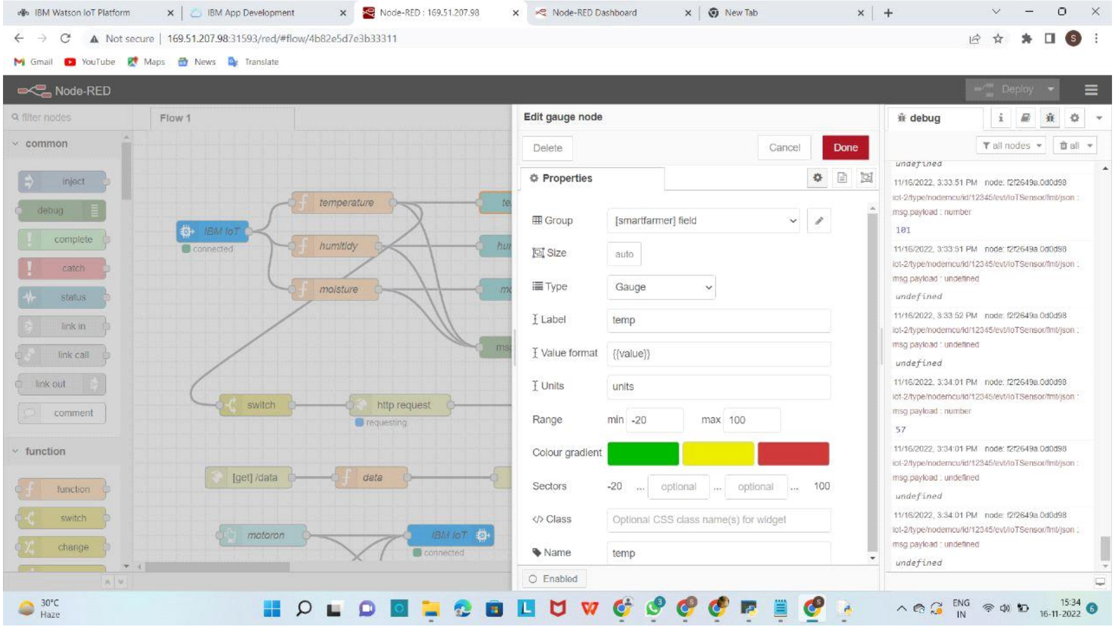
```
undefined
11/16/2022, 3:25:50 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
102
11/16/2022, 3:25:50 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:25:50 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:26:00 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
89
11/16/2022, 3:26:00 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:26:00 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
```

Connect function node and The Java Script code

for the functionnode is:

```
msg.payload=msg.payload.temp
  return msg;
```

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



Edit gauge node

Properties

Group: [smartfarmer] field

Size: auto

Type: Gauge

Label: temp

Value format: {{value}}

Units: units

Range: min -20 max 100

Colour gradient: 🟩 🟨 🟥

Sectors: -20 ... optional ... optional ... 100

Class: Optional CSS class name(s) for widget

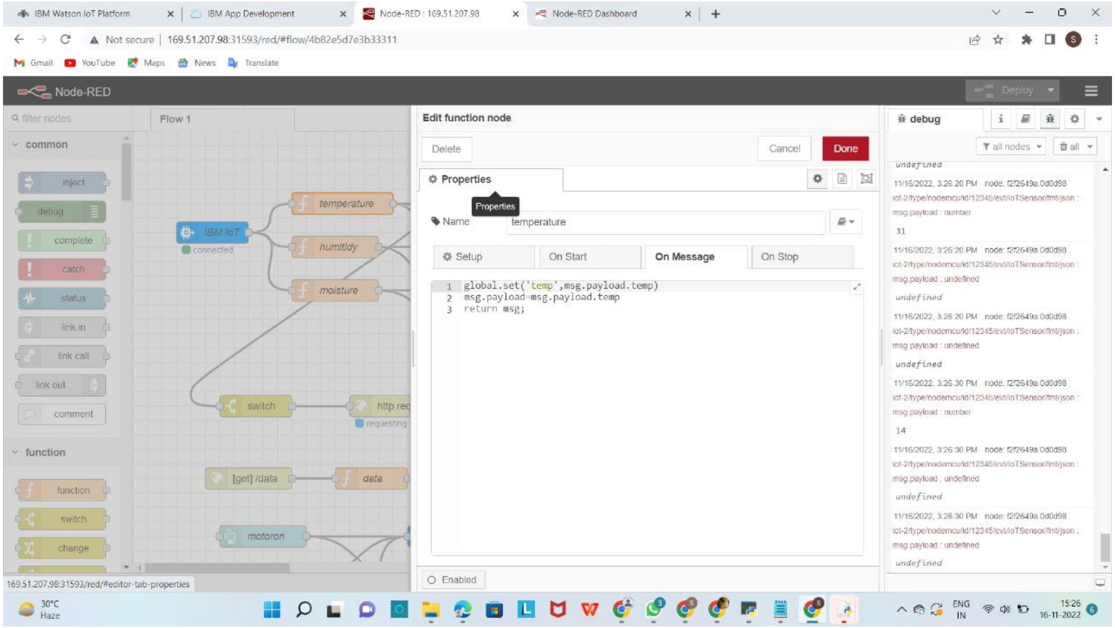
Name: temp

○ Enabled

```
undefined
11/16/2022, 3:33:51 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
101
11/16/2022, 3:33:51 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:33:52 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:34:01 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
57
11/16/2022, 3:34:01 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:34:01 PM   node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
```

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.

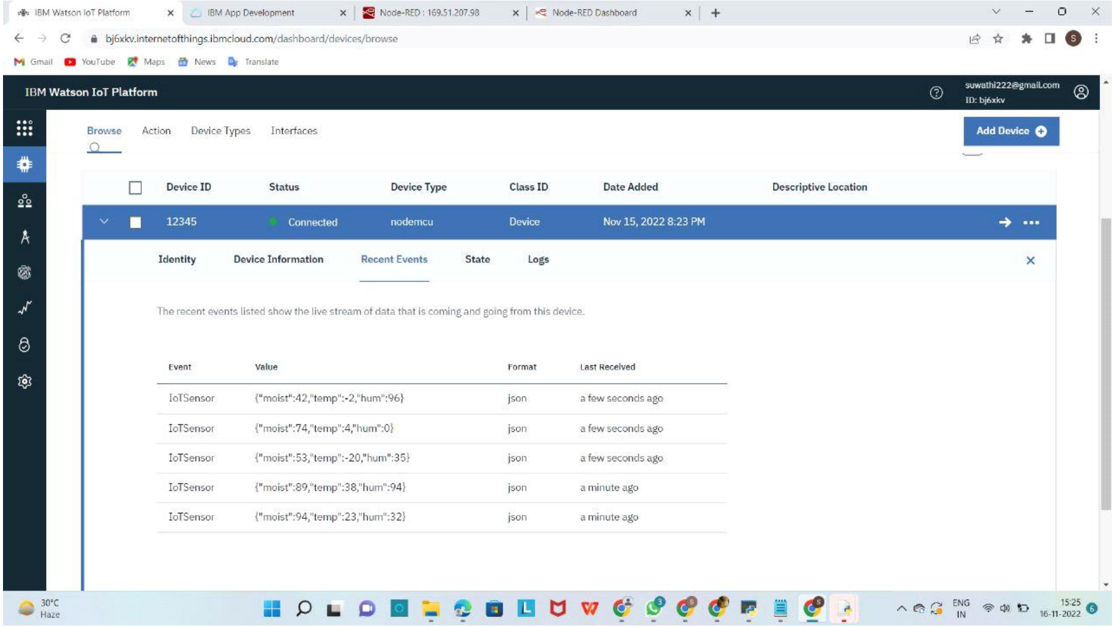


Name: temperature

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

```
undefined
11/16/2022, 3:26:20 PM  node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
31
11/16/2022, 3:26:20 PM  node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:26:20 PM  node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:26:30 PM  node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : number
14
11/16/2022, 3:26:30 PM  node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
11/16/2022, 3:26:30 PM  node: f2f2649a.0d0d98
iot-2/type/nodemcu/id/12345/evt/IoTSensor/fmt/json :
msg.payload : undefined
undefined
```

Checking IoT sensor Output in IBM Watson:



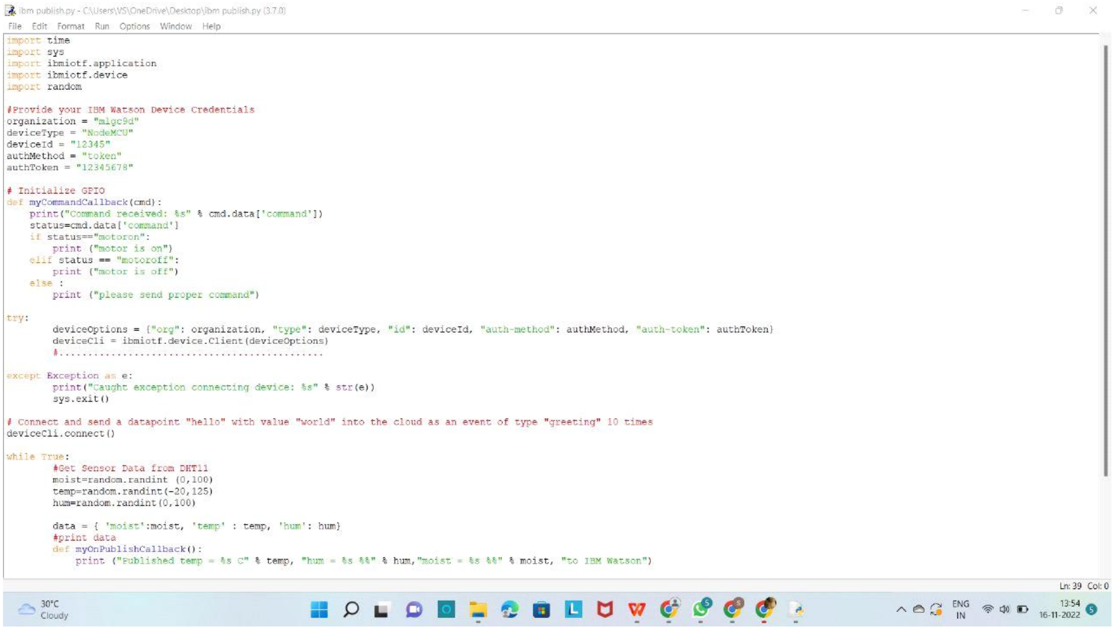
Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
12345	Connected	nodemcu	Device	Nov 15, 2022 8:23 PM	

Identity | Device Information | Recent Events | State | Logs

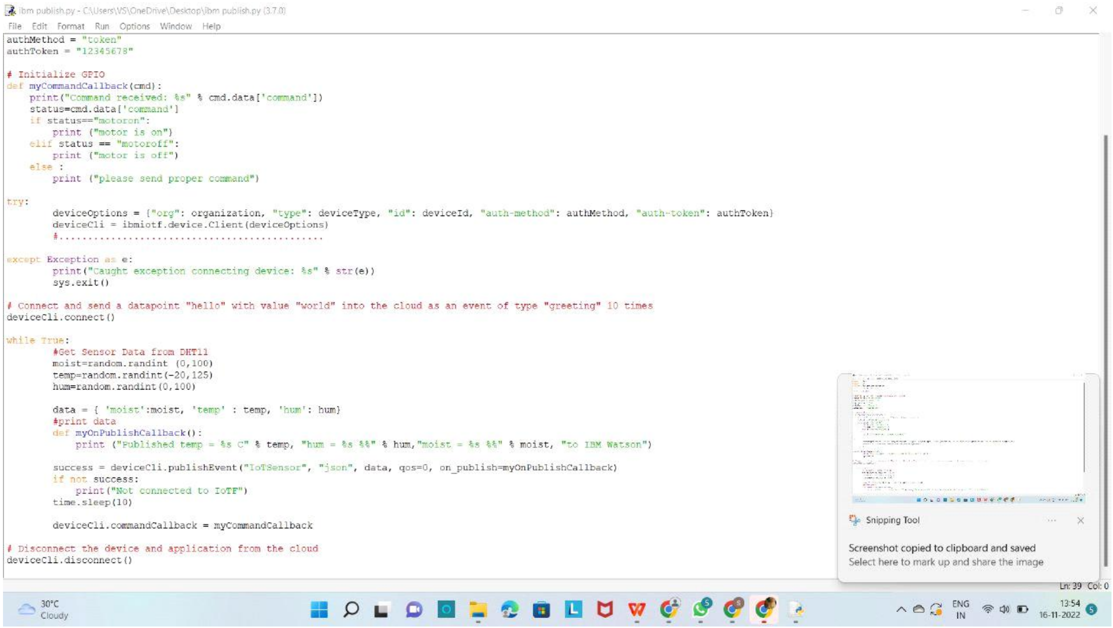
The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
IoTSensor	{"moist":42,"temp":-2,"hum":96}	json	a few seconds ago
IoTSensor	{"moist":74,"temp":4,"hum":0}	json	a few seconds ago
IoTSensor	{"moist":53,"temp":-20,"hum":35}	json	a few seconds ago
IoTSensor	{"moist":89,"temp":38,"hum":94}	json	a minute ago
IoTSensor	{"moist":94,"temp":23,"hum":32}	json	a minute ago

Checking IoT sensor using command in Node-RED:

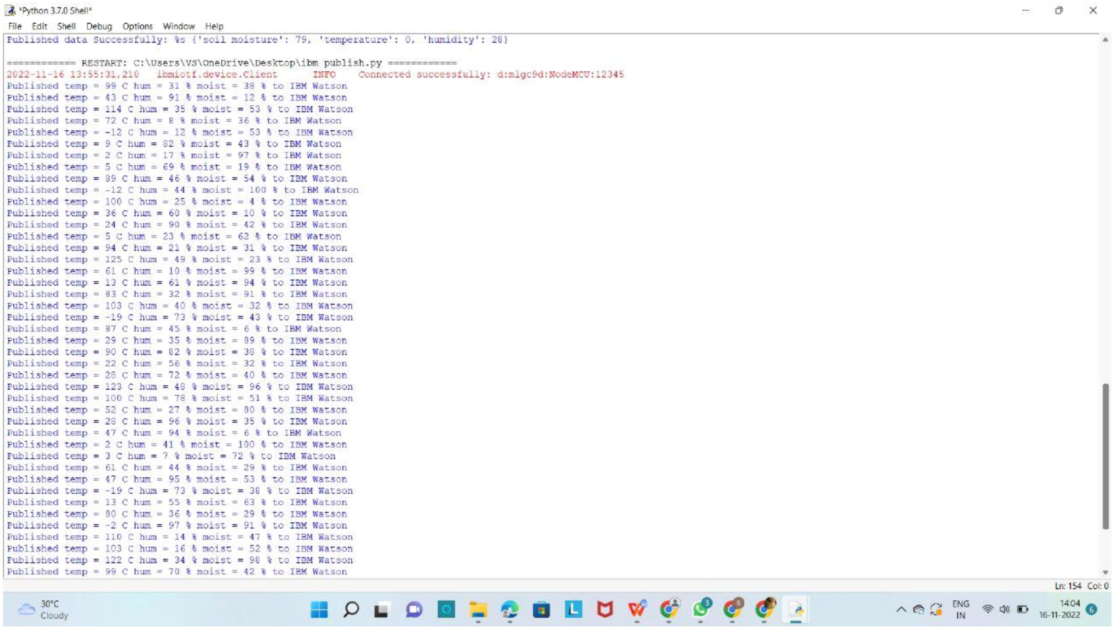

ibm publish.py - C:\Users\VS\OneDrive\Desktop\ibm publish.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 39 Col: 0
30°C
Cloudy
ENG
IN
13:54
16-11-2022

```python
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "mlgc9d"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
    elif status == "motoroff":
        print ("motor is off")
    else :
        print ("please send proper command")

try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        #..............................................

except Exception as e:
        print("Caught exception connecting device: %s" % str(e))
        sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:
        #Get Sensor Data from DHT11
        moist=random.randint (0,100)
        temp=random.randint(-20,125)
        hum=random.randint(0,100)

        data = { 'moist':moist, 'temp' : temp, 'hum': hum}
        #print data
        def myOnPublishCallback():
            print ("Published temp = %s C" % temp, "hum = %s %%" % hum,"moist = %s %%" % moist, "to IBM Watson")
```


ibm publish.py - C:\Users\VS\OneDrive\Desktop\ibm publish.py (3.7.0)
File Edit Format Run Options Window Help
Snipping Tool
Ln: 39 Col: 0
30°C
Cloudy
ENG
IN
13:54
16-11-2022

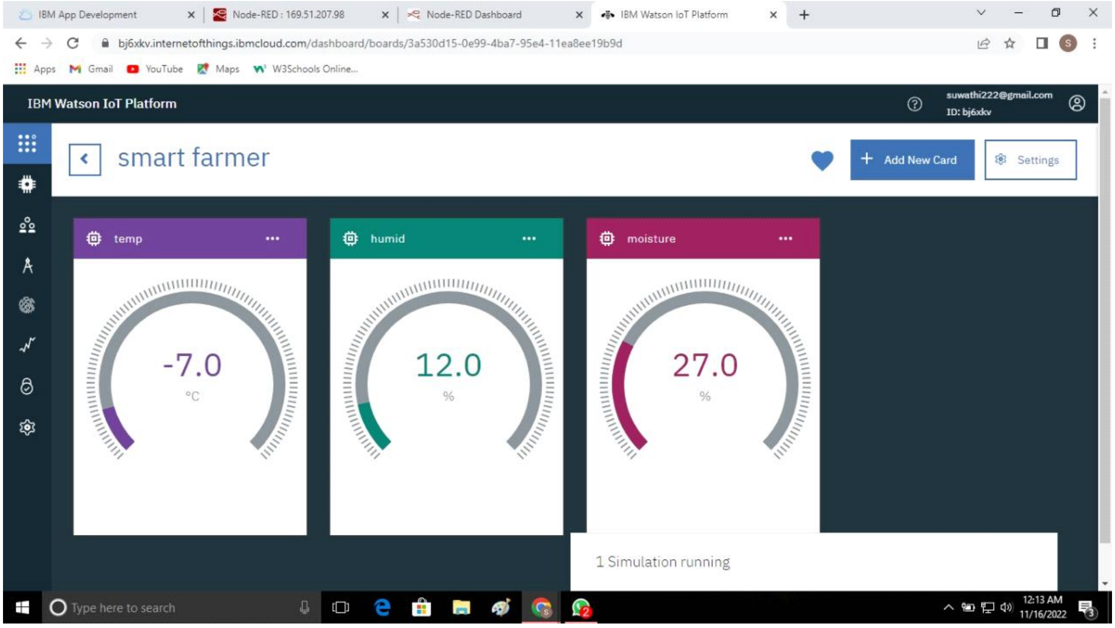
```python
authMethod = "token"
authToken = "12345678"

# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
    elif status == "motoroff":
        print ("motor is off")
    else :
        print ("please send proper command")

try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        #..............................................

except Exception as e:
        print("Caught exception connecting device: %s" % str(e))
        sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:
        #Get Sensor Data from DHT11
        moist=random.randint (0,100)
        temp=random.randint(-20,125)
        hum=random.randint(0,100)

        data = { 'moist':moist, 'temp' : temp, 'hum': hum}
        #print data
        def myOnPublishCallback():
            print ("Published temp = %s C" % temp, "hum = %s %%" % hum,"moist = %s %%" % moist, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(10)

        deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

Screenshot copied to clipboard and saved
Select here to mark up and share the image



```
Published data Successfully: %s {'soil moisture': 79, 'temperature': 0, 'humidity': 28}

============ RESTART: C:\Users\VS\OneDrive\Desktop\ibm publish.py ============
2022-11-16 13:55:31,210   ibmiotf.device.Client      INFO    Connected successfully: d:mlgc9d:NodeMCU:12345
Published temp = 99 C hum = 31 % moist = 38 % to IBM Watson
Published temp = 43 C hum = 91 % moist = 12 % to IBM Watson
Published temp = 114 C hum = 35 % moist = 53 % to IBM Watson
Published temp = 72 C hum = 8 % moist = 36 % to IBM Watson
Published temp = -12 C hum = 12 % moist = 53 % to IBM Watson
Published temp = 9 C hum = 82 % moist = 43 % to IBM Watson
Published temp = 2 C hum = 17 % moist = 97 % to IBM Watson
Published temp = 5 C hum = 69 % moist = 19 % to IBM Watson
Published temp = 89 C hum = 46 % moist = 54 % to IBM Watson
Published temp = -12 C hum = 44 % moist = 100 % to IBM Watson
Published temp = 100 C hum = 25 % moist = 4 % to IBM Watson
Published temp = 36 C hum = 60 % moist = 10 % to IBM Watson
Published temp = 24 C hum = 90 % moist = 42 % to IBM Watson
Published temp = 5 C hum = 23 % moist = 62 % to IBM Watson
Published temp = 94 C hum = 21 % moist = 31 % to IBM Watson
Published temp = 125 C hum = 49 % moist = 23 % to IBM Watson
Published temp = 61 C hum = 10 % moist = 99 % to IBM Watson
Published temp = 13 C hum = 61 % moist = 94 % to IBM Watson
Published temp = 83 C hum = 32 % moist = 91 % to IBM Watson
Published temp = 103 C hum = 40 % moist = 32 % to IBM Watson
Published temp = -19 C hum = 73 % moist = 43 % to IBM Watson
Published temp = 87 C hum = 45 % moist = 6 % to IBM Watson
Published temp = 29 C hum = 35 % moist = 89 % to IBM Watson
Published temp = 90 C hum = 82 % moist = 38 % to IBM Watson
Published temp = 22 C hum = 56 % moist = 32 % to IBM Watson
Published temp = 28 C hum = 72 % moist = 40 % to IBM Watson
Published temp = 123 C hum = 48 % moist = 96 % to IBM Watson
Published temp = 100 C hum = 78 % moist = 51 % to IBM Watson
Published temp = 52 C hum = 27 % moist = 80 % to IBM Watson
Published temp = 28 C hum = 96 % moist = 35 % to IBM Watson
Published temp = 47 C hum = 94 % moist = 6 % to IBM Watson
Published temp = 2 C hum = 41 % moist = 100 % to IBM Watson
Published temp = 3 C hum = 7 % moist = 72 % to IBM Watson
Published temp = 61 C hum = 44 % moist = 29 % to IBM Watson
Published temp = 47 C hum = 95 % moist = 53 % to IBM Watson
Published temp = -19 C hum = 73 % moist = 38 % to IBM Watson
Published temp = 13 C hum = 55 % moist = 63 % to IBM Watson
Published temp = 80 C hum = 36 % moist = 29 % to IBM Watson
Published temp = -2 C hum = 97 % moist = 91 % to IBM Watson
Published temp = 110 C hum = 14 % moist = 47 % to IBM Watson
Published temp = 103 C hum = 16 % moist = 52 % to IBM Watson
Published temp = 122 C hum = 34 % moist = 98 % to IBM Watson
Published temp = 99 C hum = 70 % moist = 42 % to IBM Watson
```

Output in Node-RED Dashboard:



IBM App Development
Node-RED : 169.51.207.98
Node-RED Dashboard
IBM Watson IoT Platform
bj6xkv.internetofthings.ibmcloud.com/dashboard/boards/3a530d15-0e99-4ba7-95e4-11ea8ee19b9d
Apps
Gmail
YouTube
Maps
W3Schools Online...
IBM Watson IoT Platform
suwathi222@gmail.com
ID: bj6xkv
smart farmer
Add New Card
Settings
temp
-7.0
°C
humid
12.0
%
moisture
27.0
%
1 Simulation running
Type here to search
12:13 AM
11/16/2022