

SPRINT 2 - Receiving datas in IBM Watson and developing web

Application using Node RED

Date:	15 th November 2022
Team ID	PNT2022TMID27964
Project Name	Project – Smart Farmer- IoT basedSmartFarmingApplication

AIM:

To create a device in IBM Watson to receive datas from where Node RED is used to develop a web application

SOFTWARES USED:

- IBM Cloud
- IBM Watson for IoT
- Node RED
- Python

PROCEDURE :

- The python code is first run to receive the Sensor values.
- A new device is created in Ibm Watson and the API and authentication token is generated.
- The output of code is linked to Ibm Cloud from where the datas are sent to Ibm Watson platofrm for Iot.
- From here the datas are send to Node Red , which is used to create the web application.
- The received data are graphically represented in IBM Watson and Node RED respectively.

PYTHON CODE :

```
import time
```

```
import sys
```

```
importibmiotf.application
```

```
importibmiotf.device
```

```
import random
```

```
#Provide your IBM Watson Device Credentials
```

```
organization = "asgkbm"
```

```
deviceType = "smart_farming"
```

```
deviceId = "69696969"
```

```
authMethod = "token"
```

```
authToken = "12345678"
```

Initialize GPIO

```
def myCommandCallback(cmd):
```

```
    print("Command received: %s" % cmd.data['command'])
```

```
    status=cmd.data['command']
```

```
    if status=="lighton":
```

```
        print ("led is on")
```

```
    elif status == "lightoff":
```

```
        print ("led 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

```
temp=random.randint(90,110)
```

```
Humid=random.randint(60,100)
```

```
data = { 'temp' : temp, 'Humid': Humid }
```

#print data

```
defmyOnPublishCallback():
```

```
print ("Published Temperature = %s C" % temp, "Humidity = %s %" % Humid, "to IBM Watson")
```

```
success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
```

```
if not success:
```

```
print("Not connected to IoT")
```

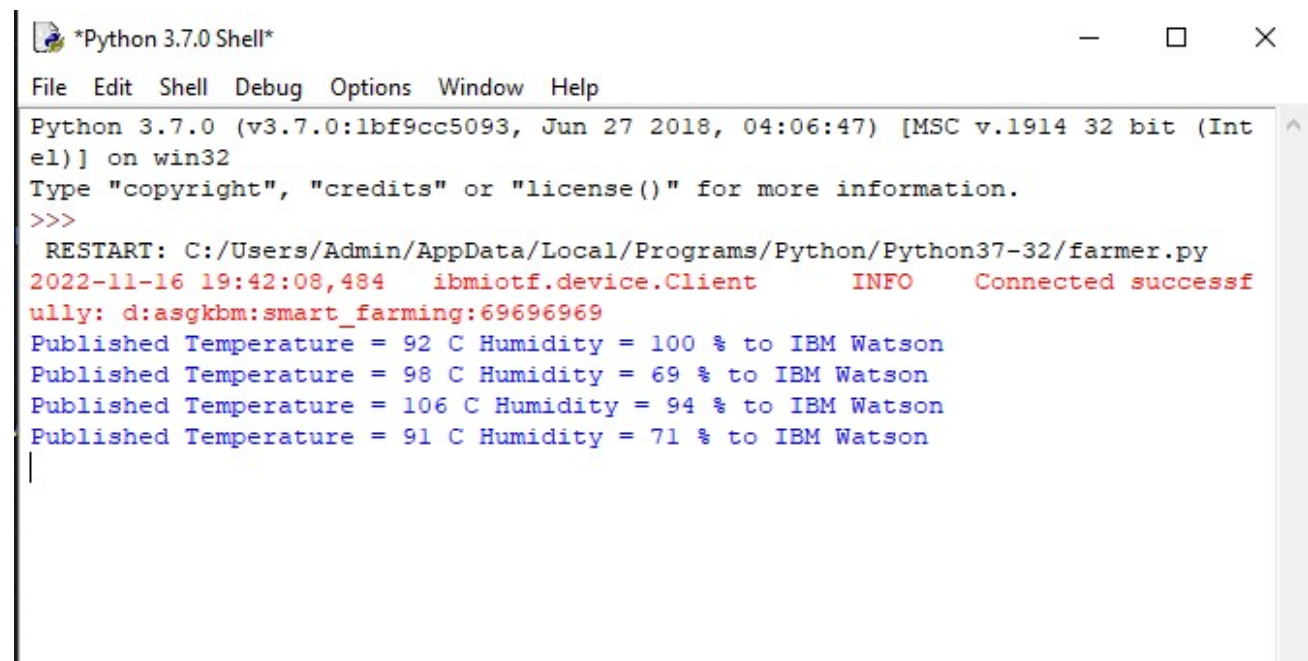
```
time.sleep(10)
```

```
deviceCli.commandCallback = myCommandCallback
```

Disconnect the device and application from the cloud

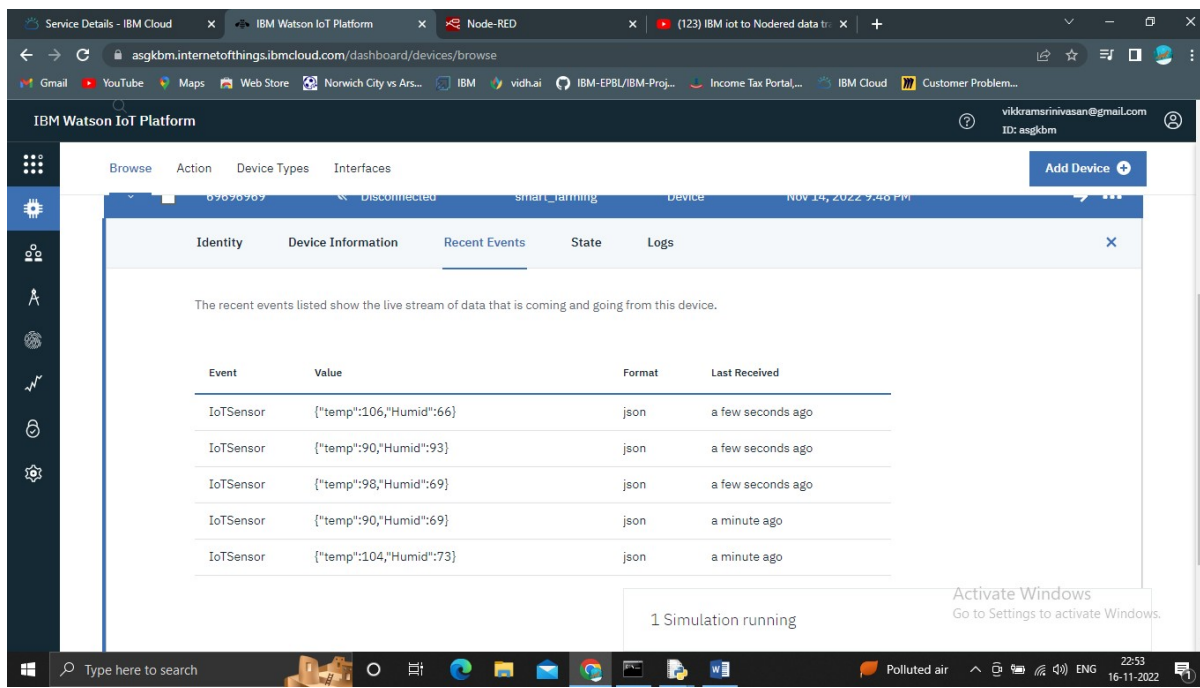
```
deviceCli.disconnect()
```

PYTHON OUTPUT :



```
*Python 3.7.0 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Int
el)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python37-32/farmer.py
2022-11-16 19:42:08,484 ibmiotf.device.Client INFO Connected successf
ully: d:asgkbm:smart_farming:69696969
Published Temperature = 92 C Humidity = 100 % to IBM Watson
Published Temperature = 98 C Humidity = 69 % to IBM Watson
Published Temperature = 106 C Humidity = 94 % to IBM Watson
Published Temperature = 91 C Humidity = 71 % to IBM Watson
|
```

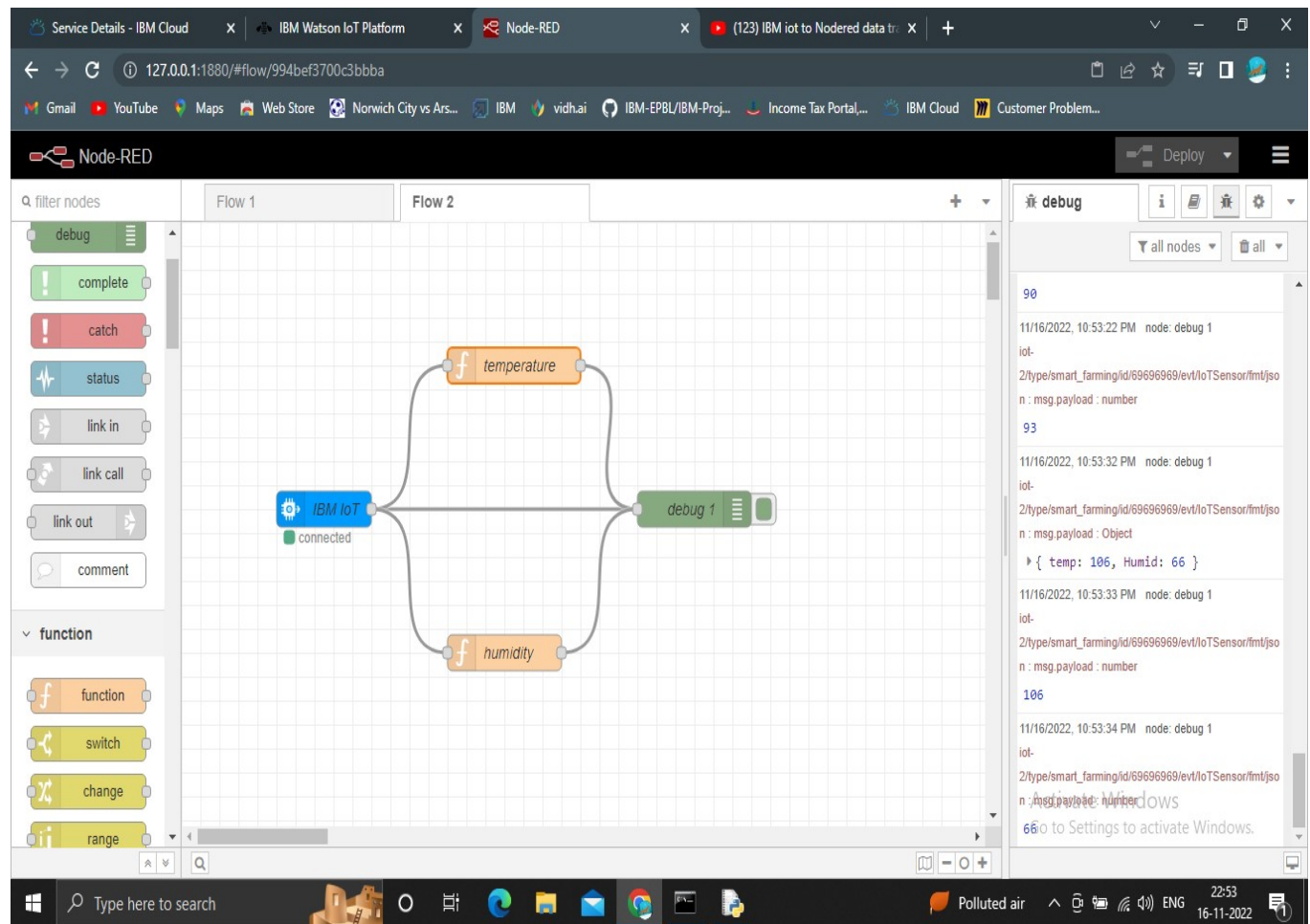
COLLECTING OUTPUT DATA IN IBM WATSON :



The screenshot shows the IBM Watson IoT Platform dashboard. The 'Recent Events' tab is selected, displaying a table of events. The table has four columns: Event, Value, Format, and Last Received. The events are from an IoT Sensor, with values like {"temp":106,"Humid":66} and {"temp":90,"Humid":93}. The format is json, and the last received time is a few seconds ago. A status bar at the bottom indicates '1 Simulation running'.

Event	Value	Format	Last Received
IoT Sensor	{"temp":106,"Humid":66}	json	a few seconds ago
IoT Sensor	{"temp":90,"Humid":93}	json	a few seconds ago
IoT Sensor	{"temp":98,"Humid":69}	json	a few seconds ago
IoT Sensor	{"temp":90,"Humid":69}	json	a minute ago
IoT Sensor	{"temp":104,"Humid":73}	json	a minute ago

CREATING WEB APPLICATION USING NODE RED



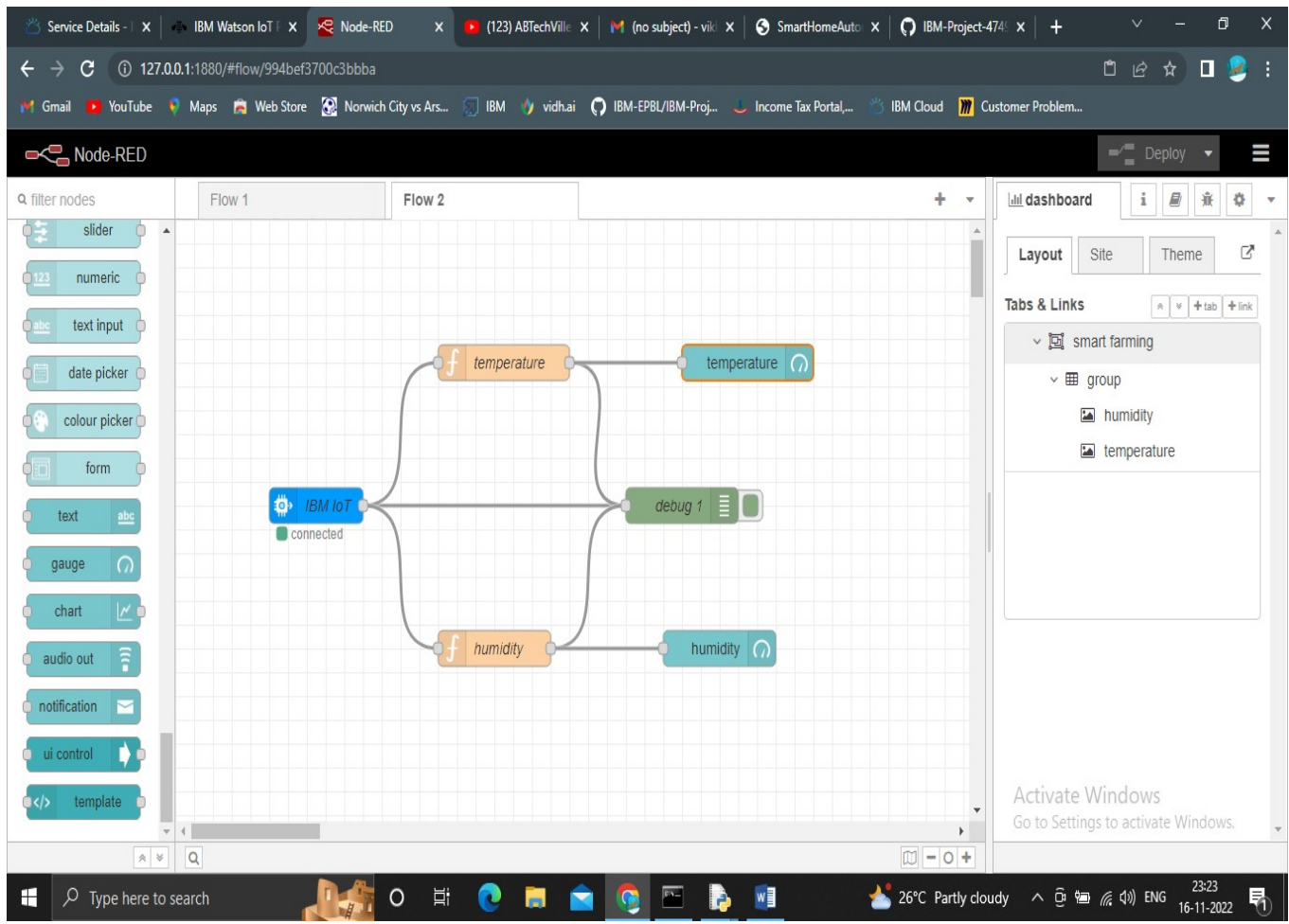
The screenshot shows the Node-RED web application interface. The flow diagram consists of an 'IBM IoT' node connected to a 'debug 1' node. The flow is split into two paths: one through a 'temperature' function node and another through a 'humidity' function node. The debug console on the right shows the output of the flow, displaying the temperature and humidity values as JSON objects.

```
graph LR
    IoT[IBM IoT] --> Temp[temperature]
    IoT --> Humid[humidity]
    Temp --> Debug[debug 1]
    Humid --> Debug
```

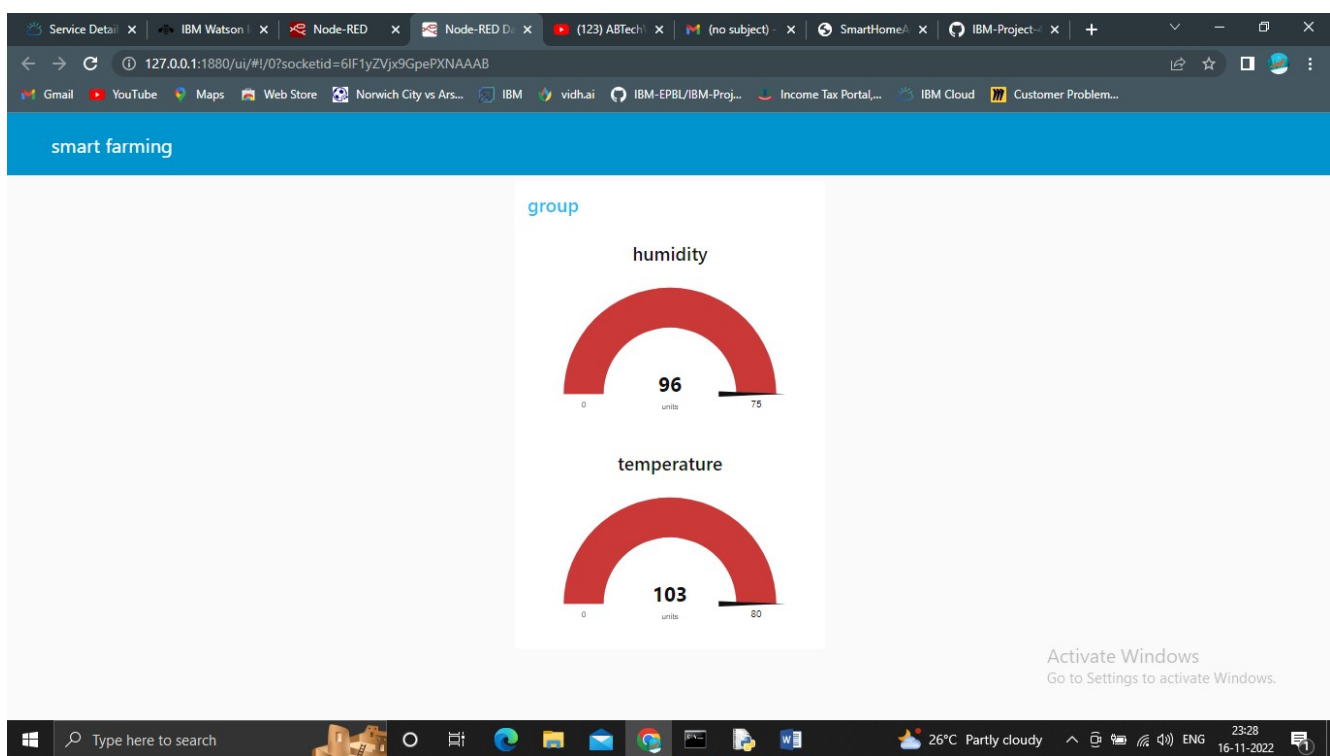
Debug Console Output:

```
11/16/2022, 10:53:22 PM node: debug 1
iot-2/type/smart_farming/id/69696969/evt/IoTSensor/rmtl/json: msg.payload: number
93
11/16/2022, 10:53:32 PM node: debug 1
iot-2/type/smart_farming/id/69696969/evt/IoTSensor/rmtl/json: msg.payload: Object
{ temp: 106, Humid: 66 }
11/16/2022, 10:53:33 PM node: debug 1
iot-2/type/smart_farming/id/69696969/evt/IoTSensor/rmtl/json: msg.payload: number
106
11/16/2022, 10:53:34 PM node: debug 1
iot-2/type/smart_farming/id/69696969/evt/IoTSensor/rmtl/json: msg.payload: number
66
```

EDITING THE FUNCTIONS AND COMPONENTS



NODE RED OUTPUT



IBM WATSON OUTPUT

