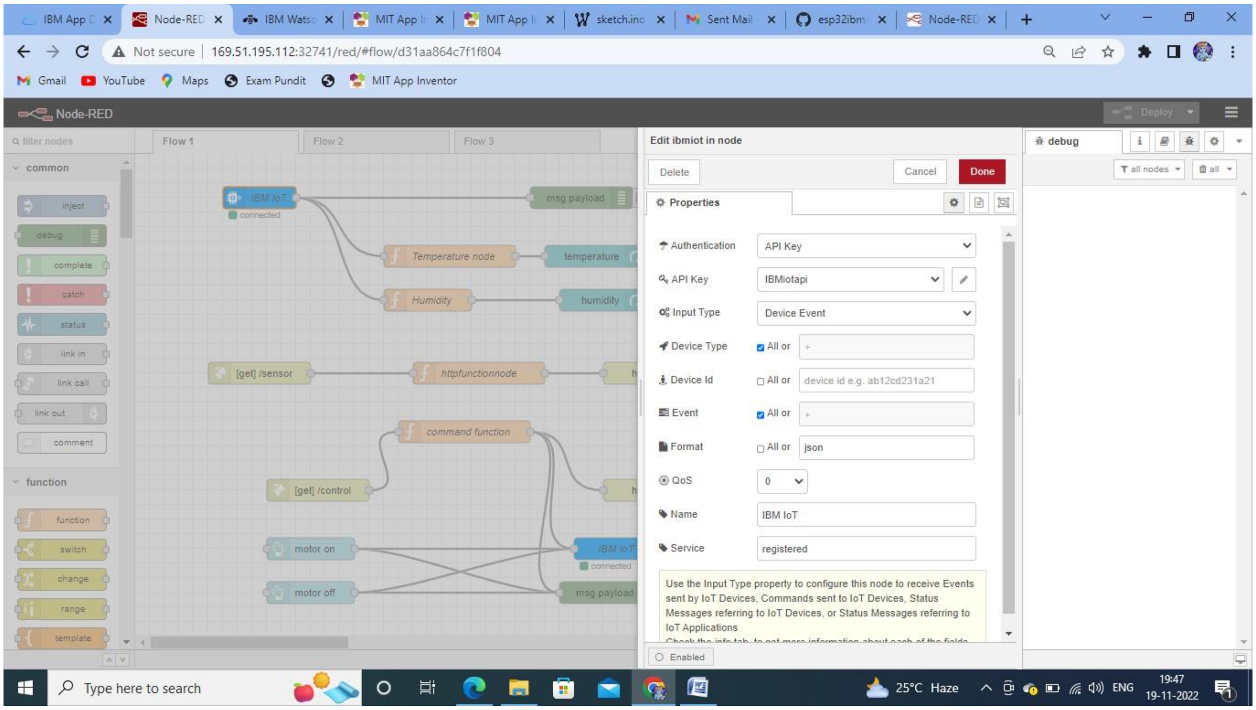
Build A Web Application Using Node-Red

Team ID	PNT2022TMID14625
Project Name	Smart Farmer-IOT Enabled Smart Farming Application

#1. First open Node RED workspace and drag IBM iot input into the workspace. It will as Ask API key, device id, device type etc.



Edit ibmiot in node

Authentication: API Key

API Key: IBMiotapi

Input Type: Device Event

Device Type: ☑ All or +

Device Id: □ All or device id e.g. ab12cd231a21

Event: ☑ All or +

Format: □ All or json

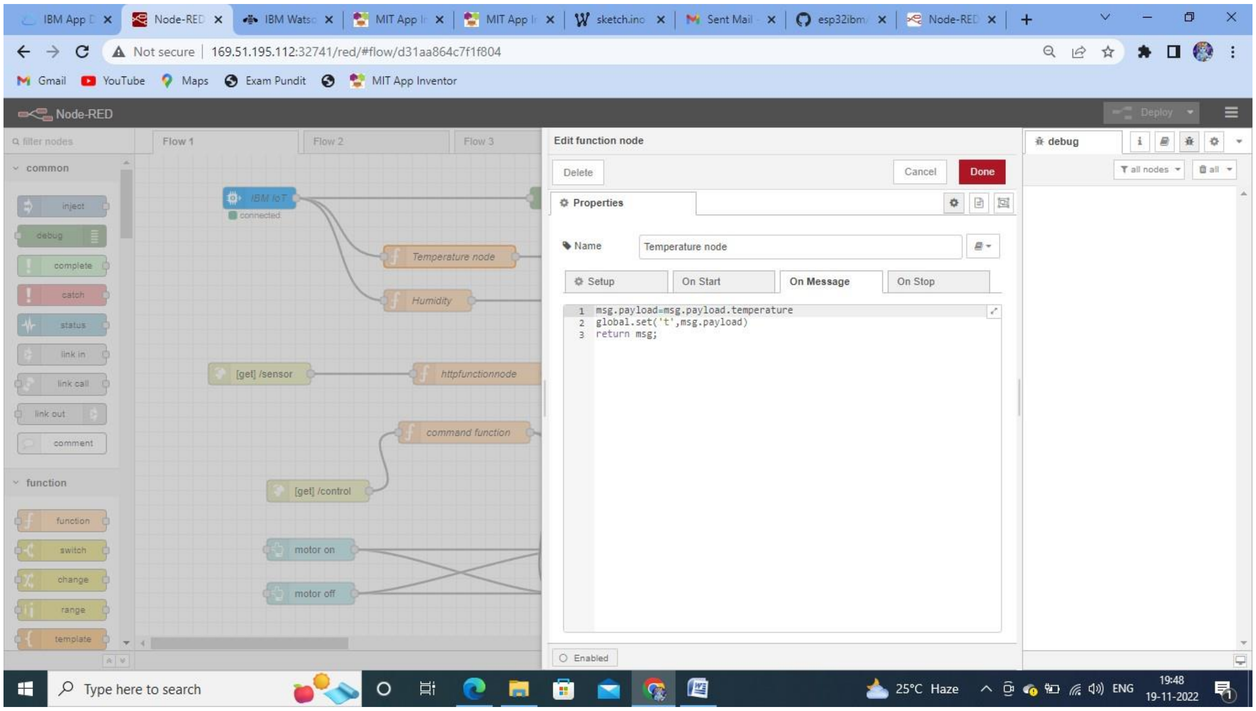
QoS: 0

Name: IBM IoT

Service: registered

Use the Input Type property to configure this node to receive Events sent by IoT Devices, Commands sent to IoT Devices, Status Messages referring to IoT Devices, or Status Messages referring to IoT Applications

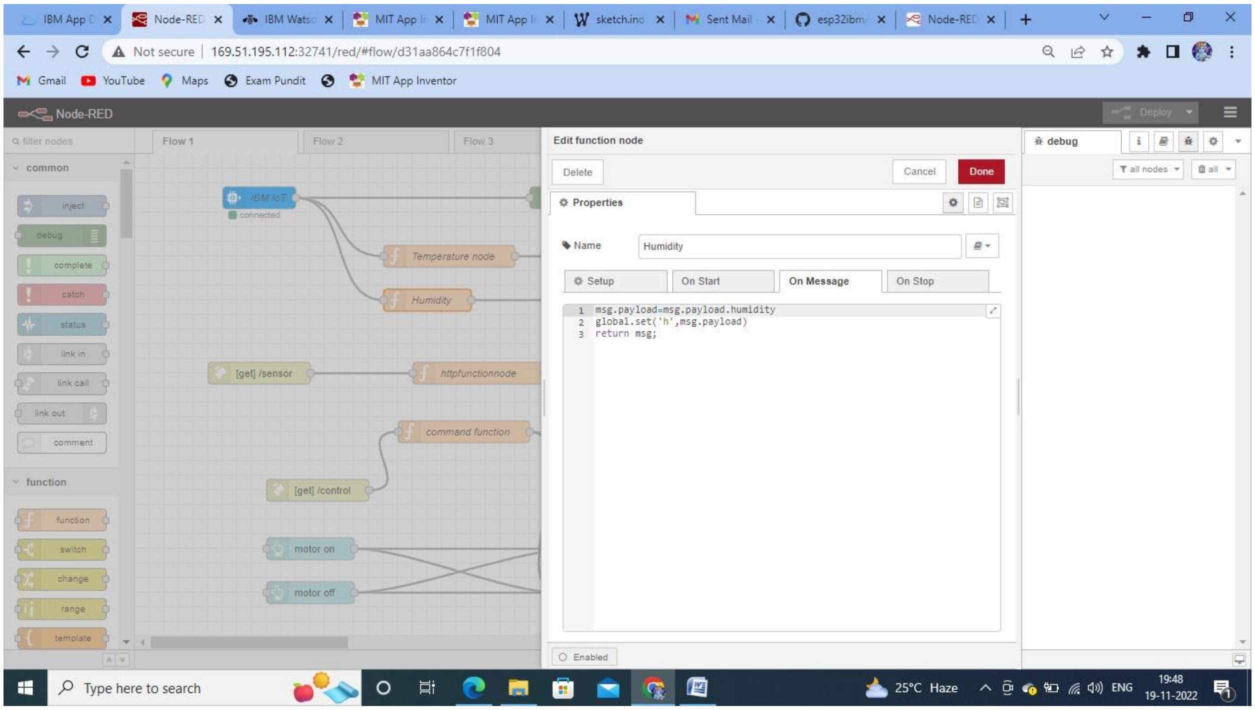
#2. Take a function node and rename it has a temperature and message in the editor.



Name: Temperature node

```
msg.payload=msg.payload.temperature
global.set('t',msg.payload)
return msg;
```

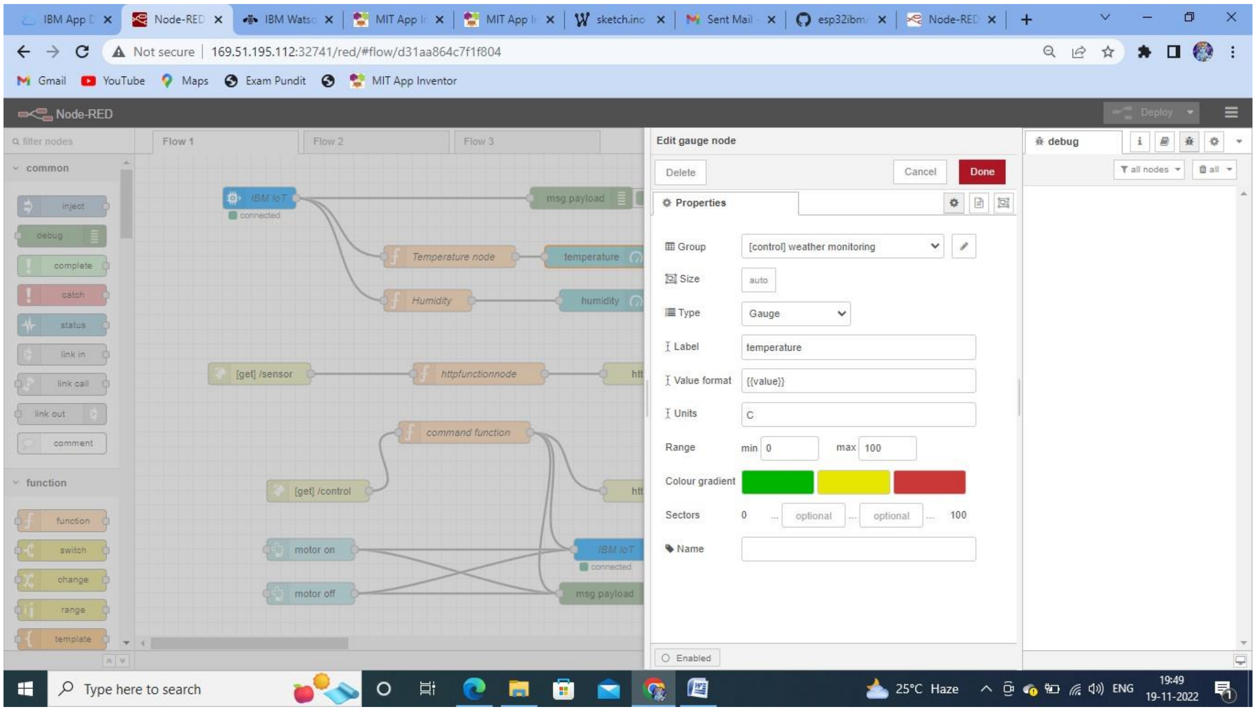
#3. Now take a function node for humidity and type message in the editor.



Name: Humidity

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

#4. Now take temperature gauge meter in the dashboard and give name as temperature and range 0 to 100.



Edit gauge node

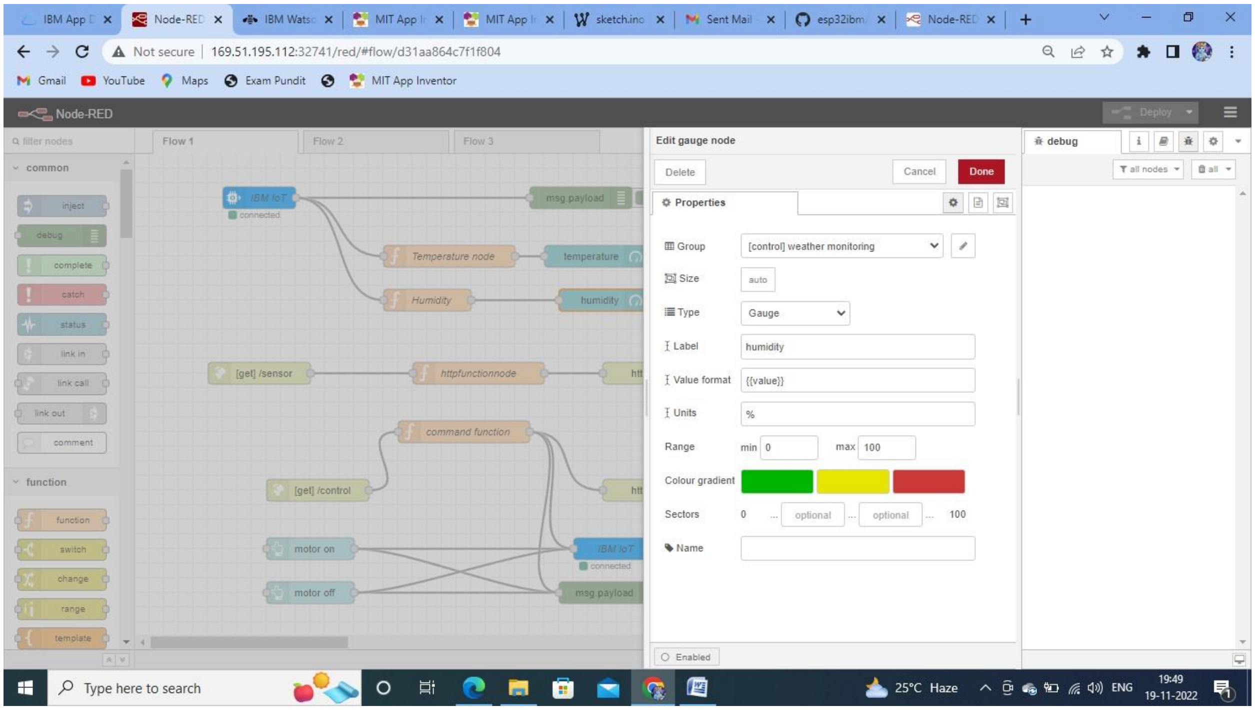
Delete | Cancel | Done

Properties

Group: [control] weather monitoring
Size: auto
Type: Gauge
Label: temperature
Value format: {{value}}
Units: C
Range: min 0 max 100
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 100
Name:

Enabled

#5. Similarly for humidity u take another gauge meter and range 0 to 100.



Edit gauge node

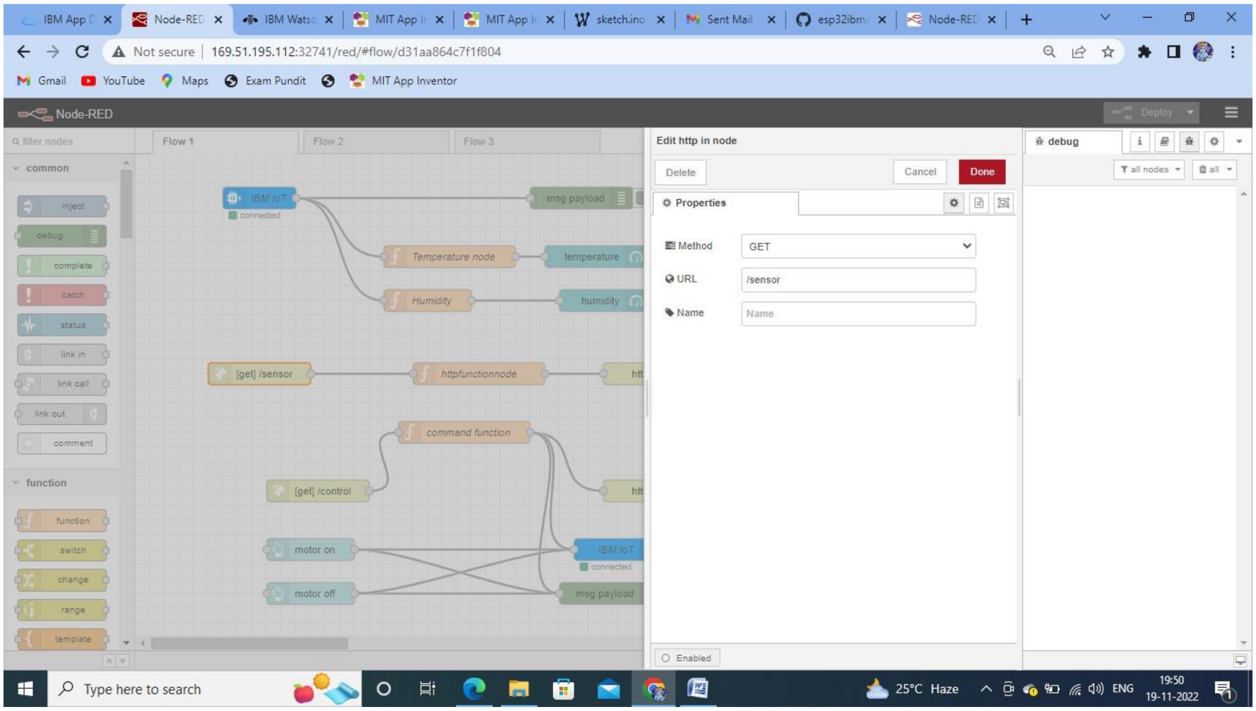
Delete | Cancel | Done

Properties

Group: [control] weather monitoring
Size: auto
Type: Gauge
Label: humidity
Value format: {{value}}
Units: %
Range: min 0 max 100
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 100
Name:

Enabled

#6. Now change the http:// in into the get /sensor.



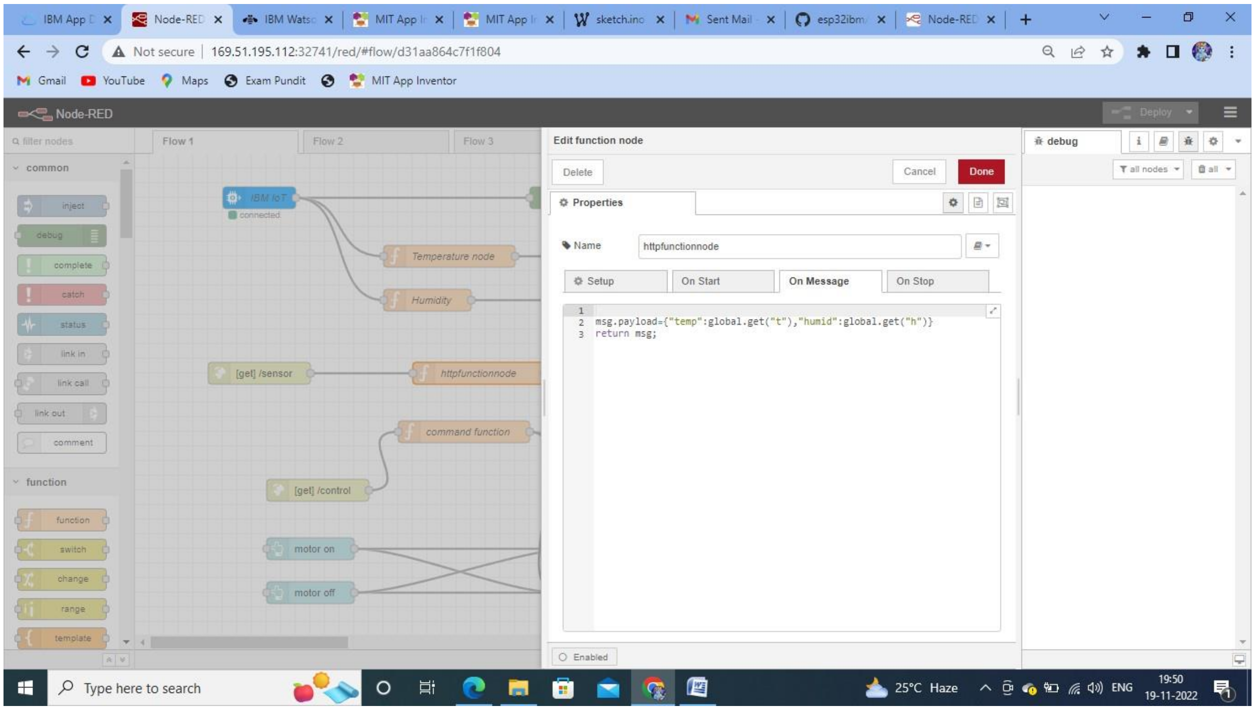
Edit http in node

Method: GET

URL: /sensor

Name: Name

#7. Now take http function and type temperature , humidity ,and soil etc.



Edit function node

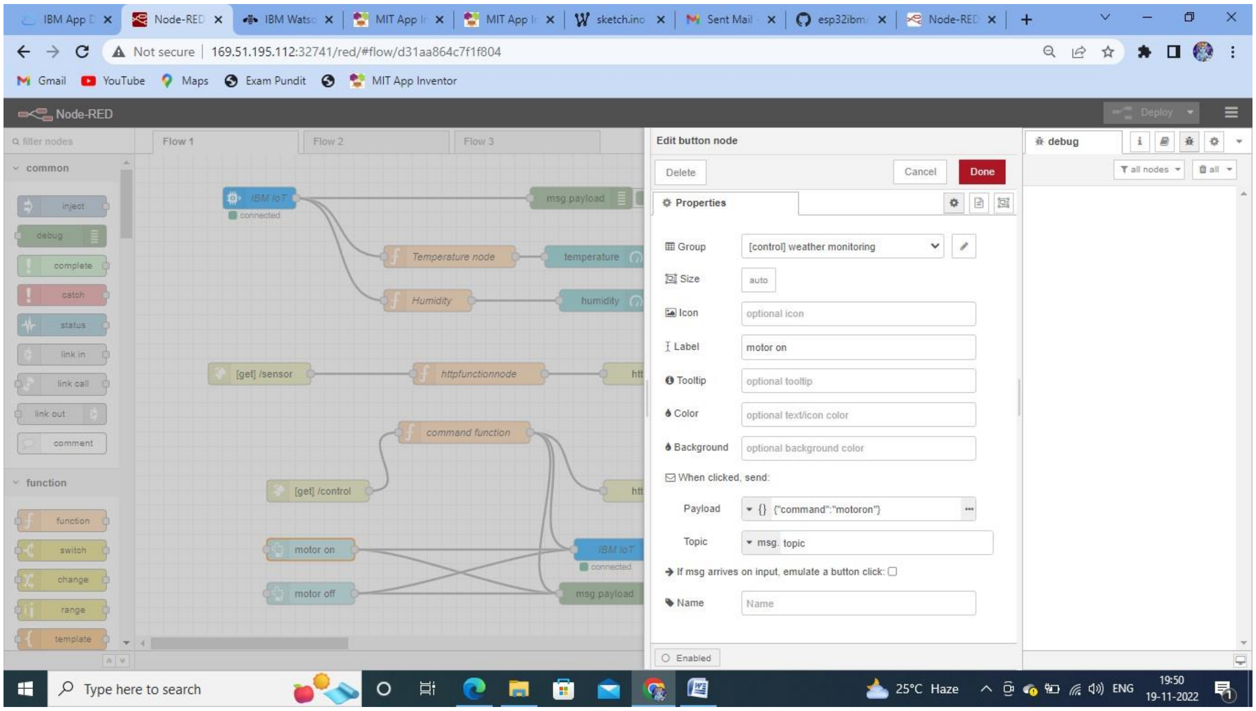
Name: httpfunctionnode

Setup | On Start | On Message | On Stop

```

msg.payload={"temp":global.get("t"),"humid":global.get("h")}
return msg;
```

#8. Now take Motor on and give command {“command”:”motoron”}



Edit button node

Group: [control] weather monitoring

Size: auto

Icon: optional icon

Label: motor on

Tooltip: optional tooltip

Color: optional text/icon color

Background: optional background color

When clicked, send:

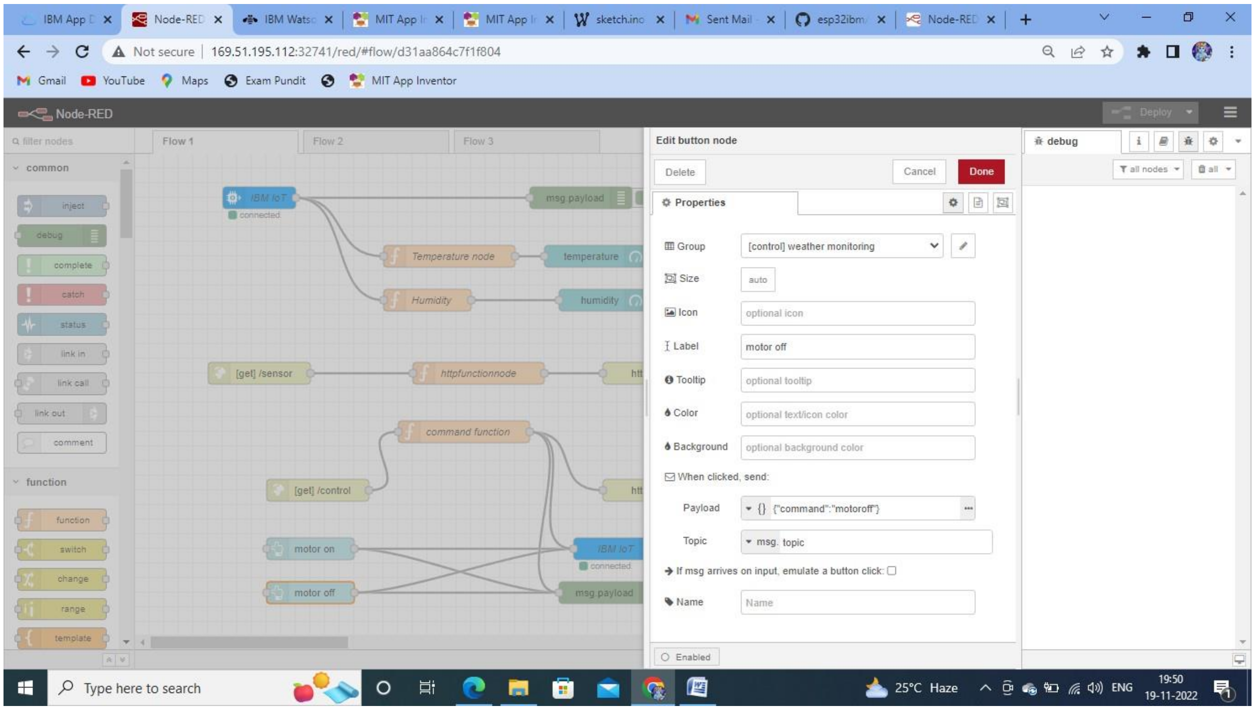
Payload: {} {"command":"motoron"}

Topic: msg. topic

If msg arrives on input, emulate a button click: □

Name: Name

#9. Now take Motor off and give command {“command”:”motoroff”}



Edit button node

Group: [control] weather monitoring

Size: auto

Icon: optional icon

Label: motor off

Tooltip: optional tooltip

Color: optional text/icon color

Background: optional background color

When clicked, send:

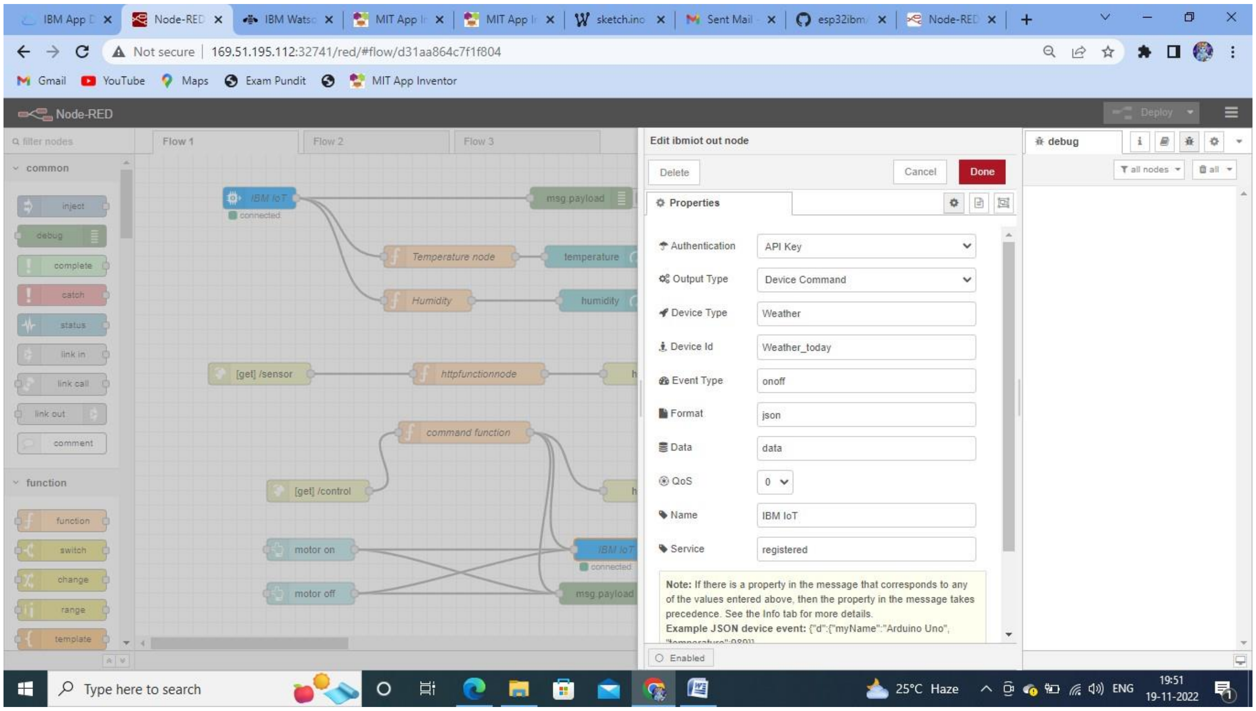
Payload: {} {"command":"motoroff"}

Topic: msg. topic

If msg arrives on input, emulate a button click: □

Name: Name

#10. Now take the IBM iot out and connect the motor on and motor off.



Edit ibmiot out node

Delete Cancel Done

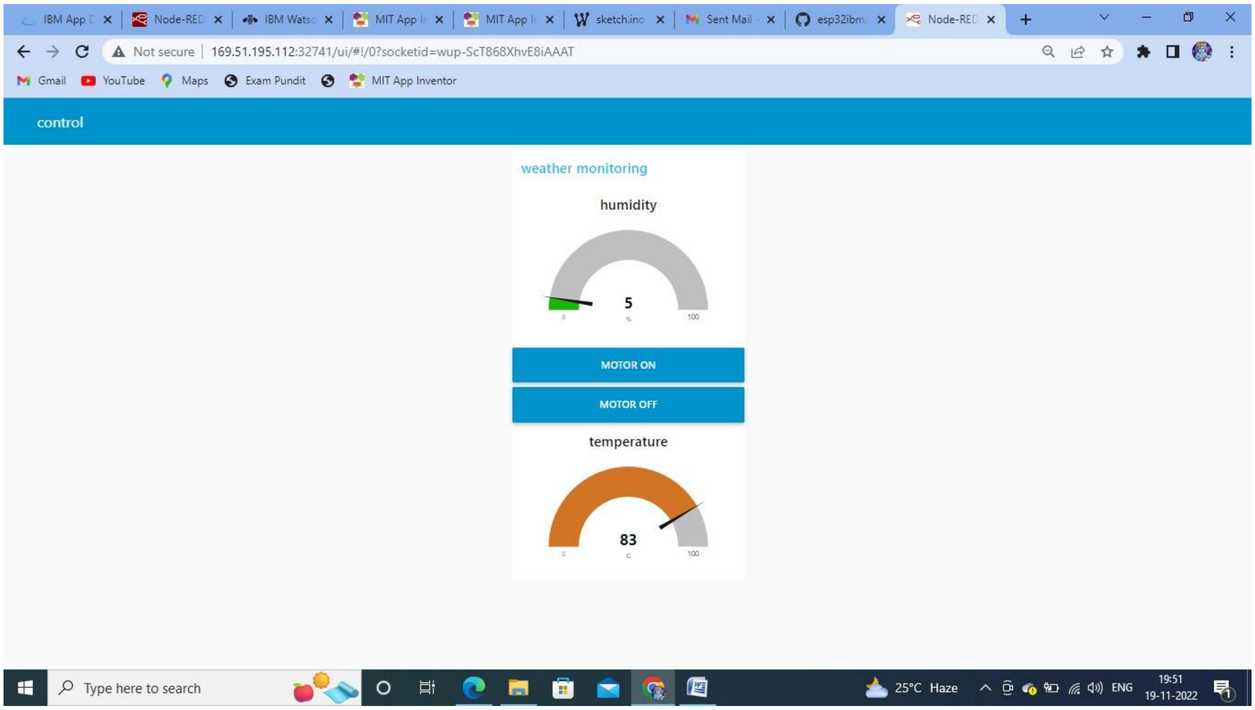
Properties

Authentication: API Key
Output Type: Device Command
Device Type: Weather
Device Id: Weather_today
Event Type: onoff
Format: json
Data: data
QoS: 0
Name: IBM IoT
Service: registered

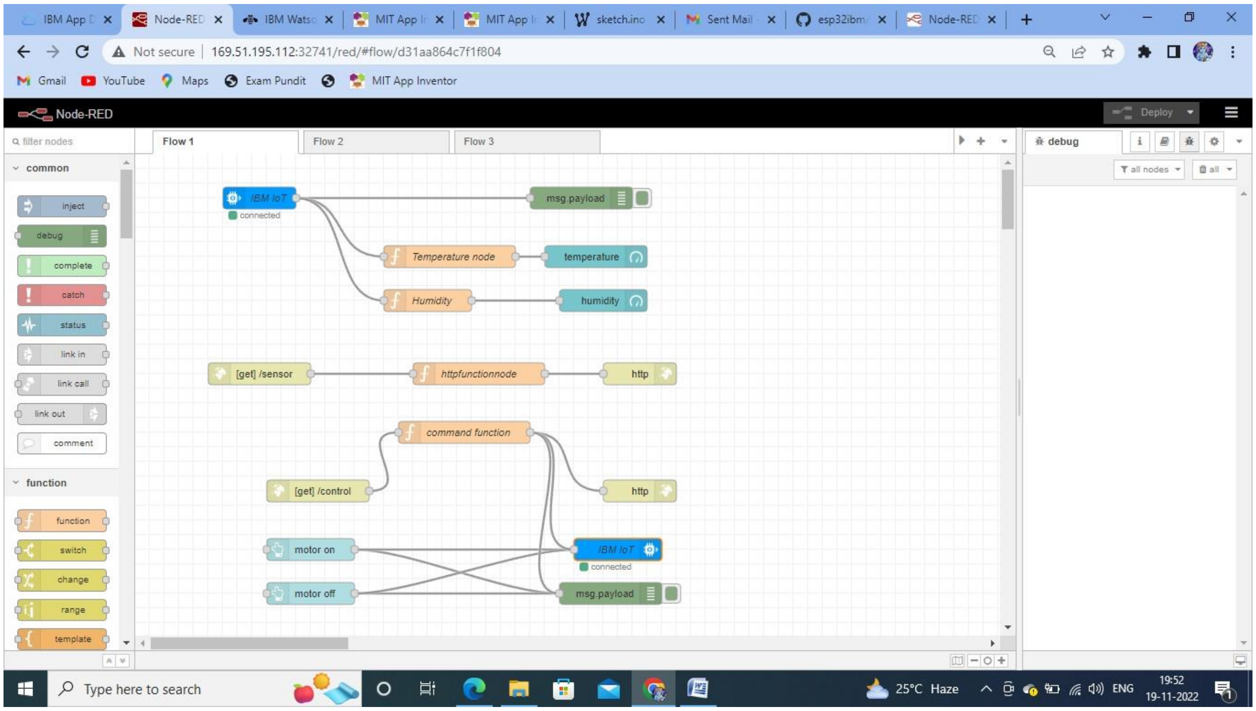
Note: If there is a property in the message that corresponds to any of the values entered above, then the property in the message takes precedence. See the Info tab for more details.
Example JSON device event: {"d":{"myName":"Arduino Uno",

○ Enabled

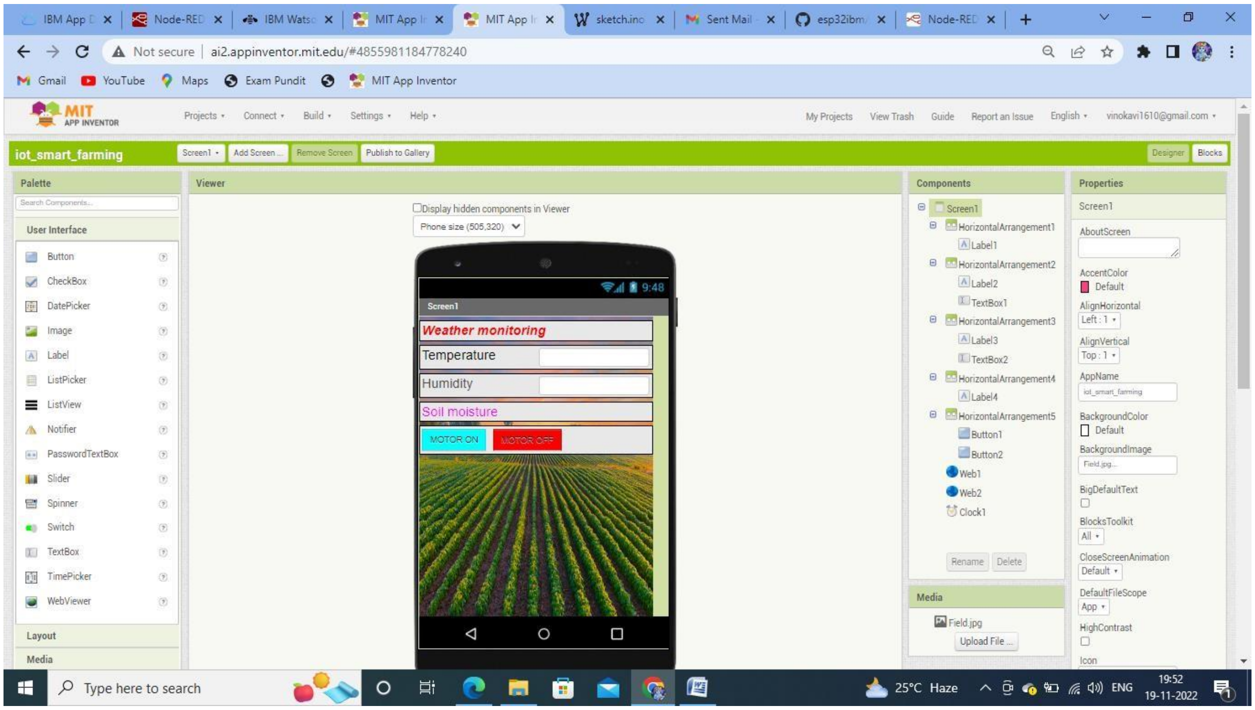
These is the node-red dashboard to see reading of temperature and humidity and soil moisture.



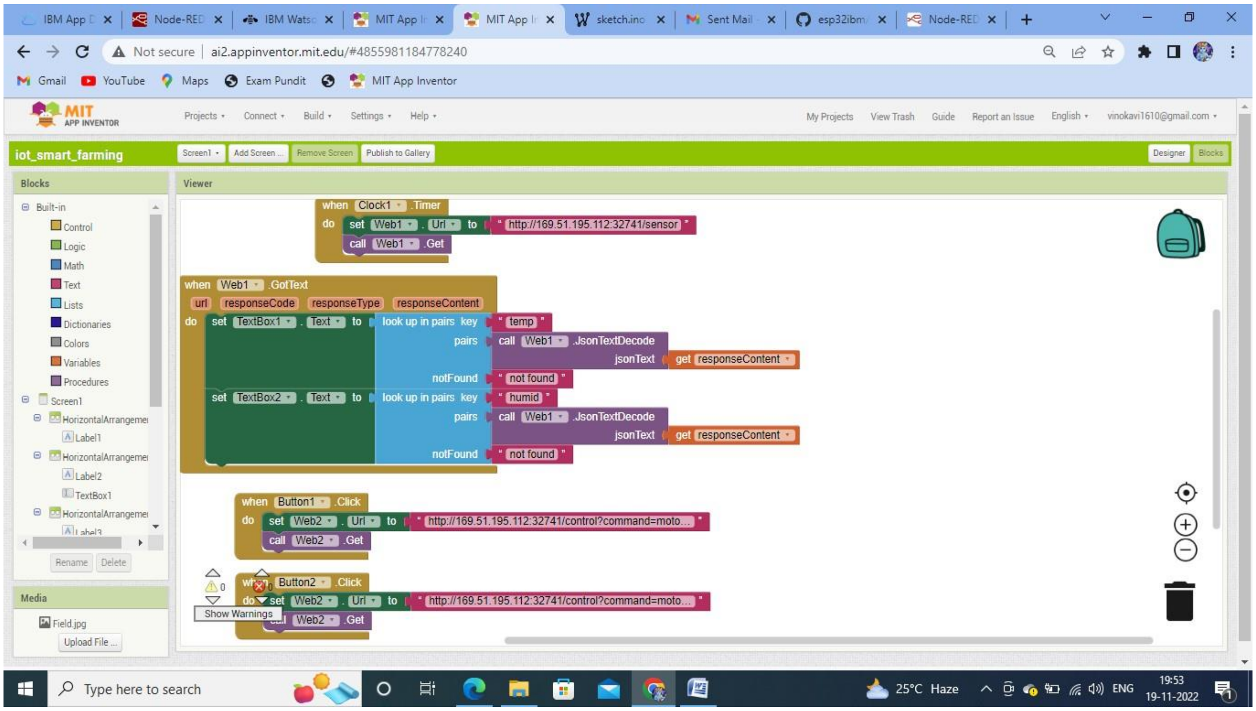
Finally we can connected as shown below:



#11. Now developing mobile application using mit app inventor.



These are the blocks of the mit app inventor.



```
when Clock1 .Timer
do  set Web1 . Url to " http://169.51.195.112:32741/sensor "
    call Web1 .Get

when Web1 .GotText
 url  responseCode  responseType  responseContent
do  set TextBox1 . Text to  look up in pairs key " temp "
                                           pairs  call Web1 .JsonTextDecode
                                                            jsonText  get responseContent
                                        notFound " not found "
    set TextBox2 . Text to  look up in pairs key " humid "
                                           pairs  call Web1 .JsonTextDecode
                                                            jsonText  get responseContent
                                        notFound " not found "

when Button1 .Click
do  set Web2 . Url to " http://169.51.195.112:32741/control?command=moto... "
    call Web2 .Get

when Button2 .Click
do  set Web2 . Url to " http://169.51.195.112:32741/control?command=moto... "
    call Web2 .Get
```