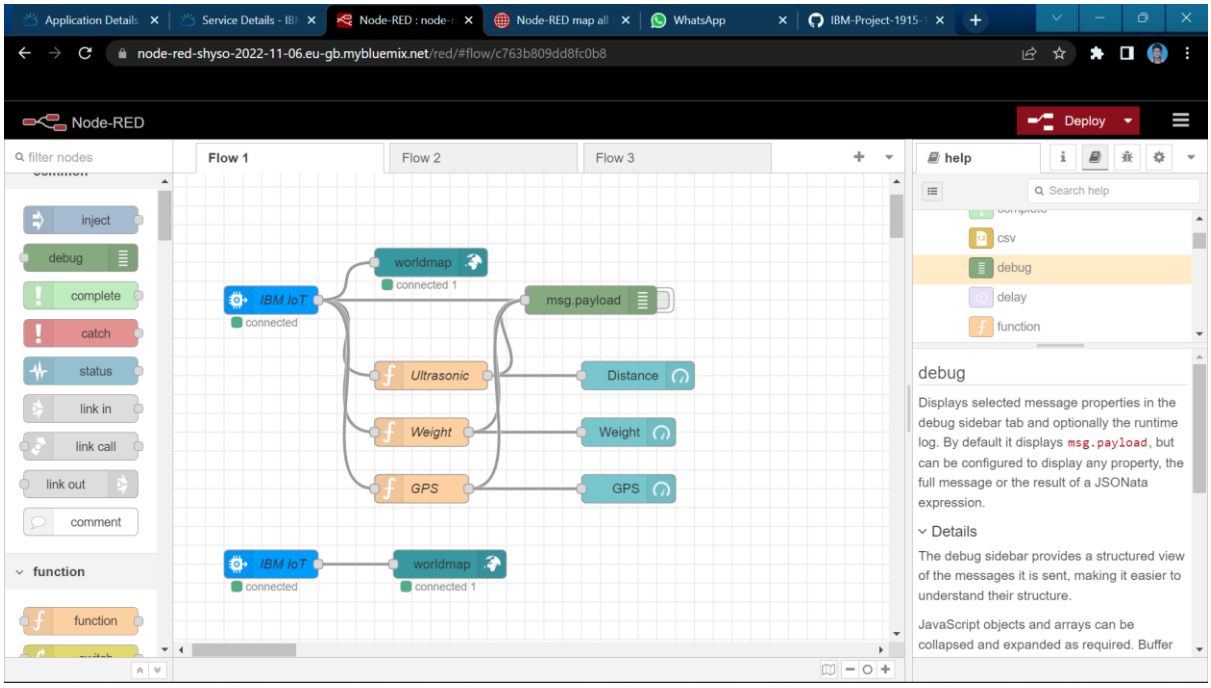
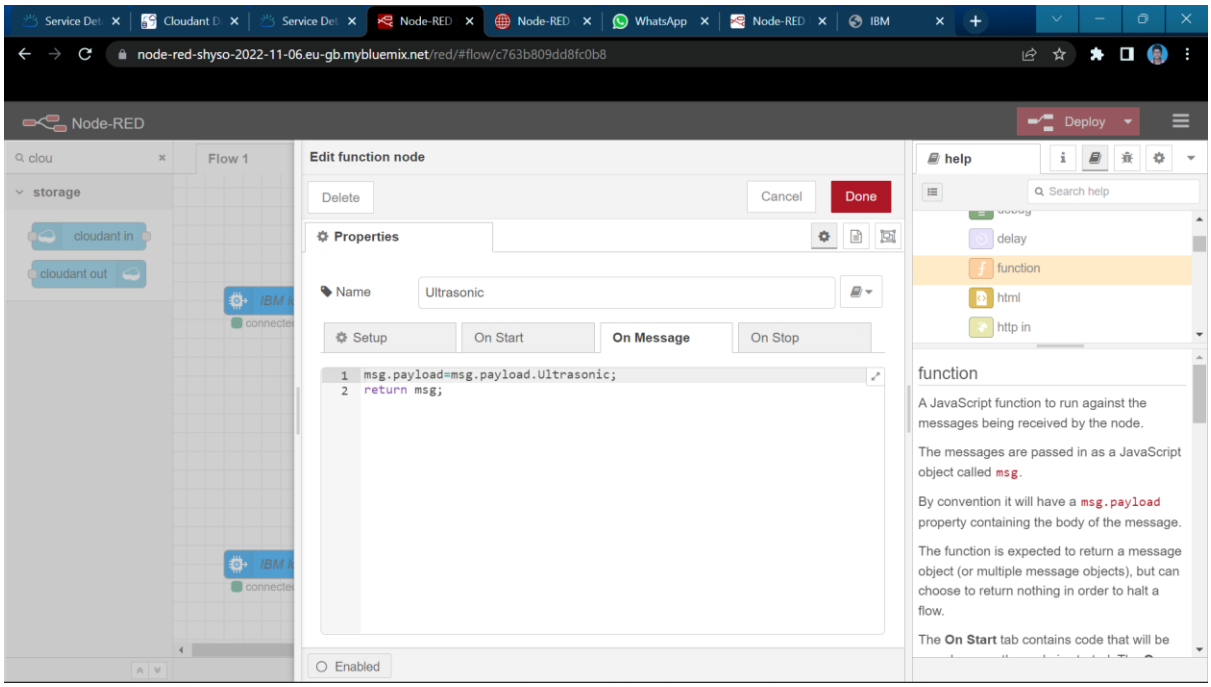
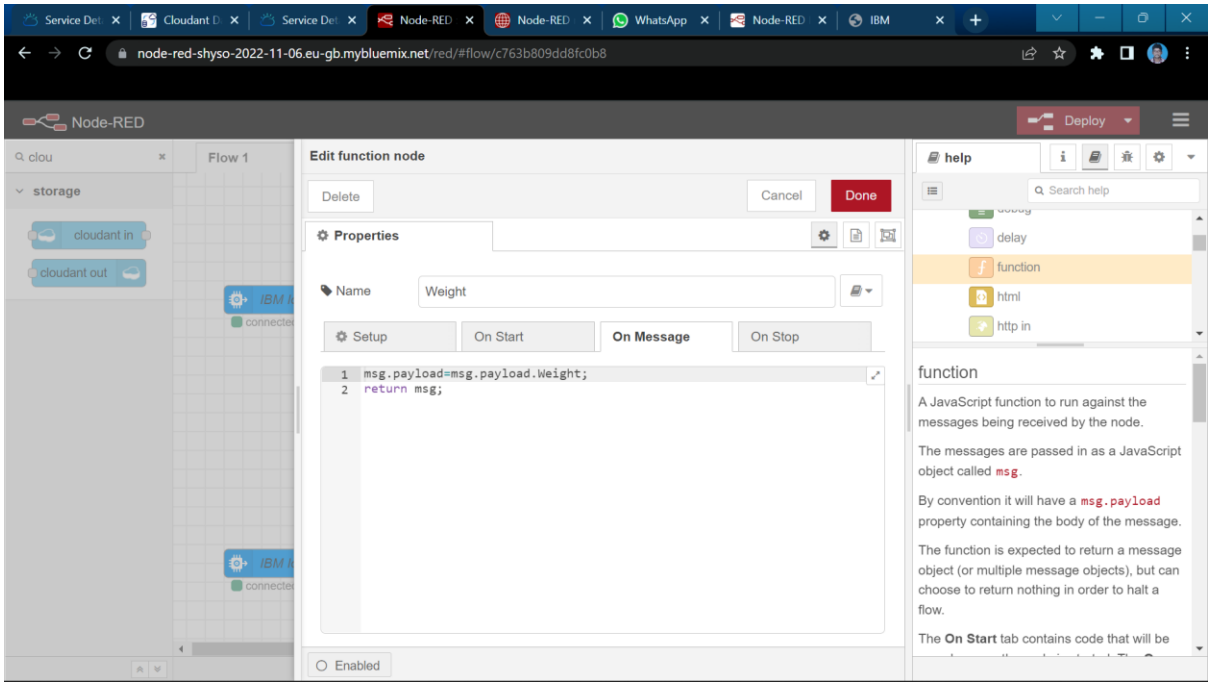
DEVELOP THE WEB APPLICATION USING NODE RED

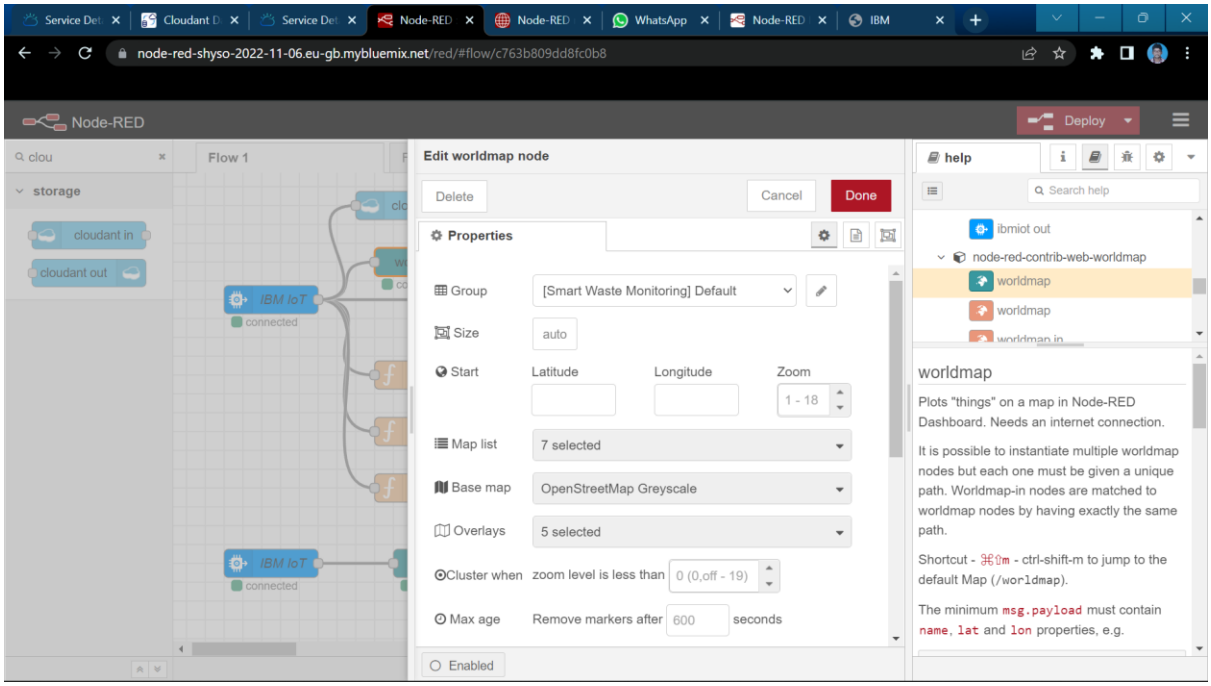
Date	14 November 2022
Team ID	PNT2022TMID25449
Project Name	SMART WASTE MANAGEMENT SYSTEM
Maximum Marks	2 Marks

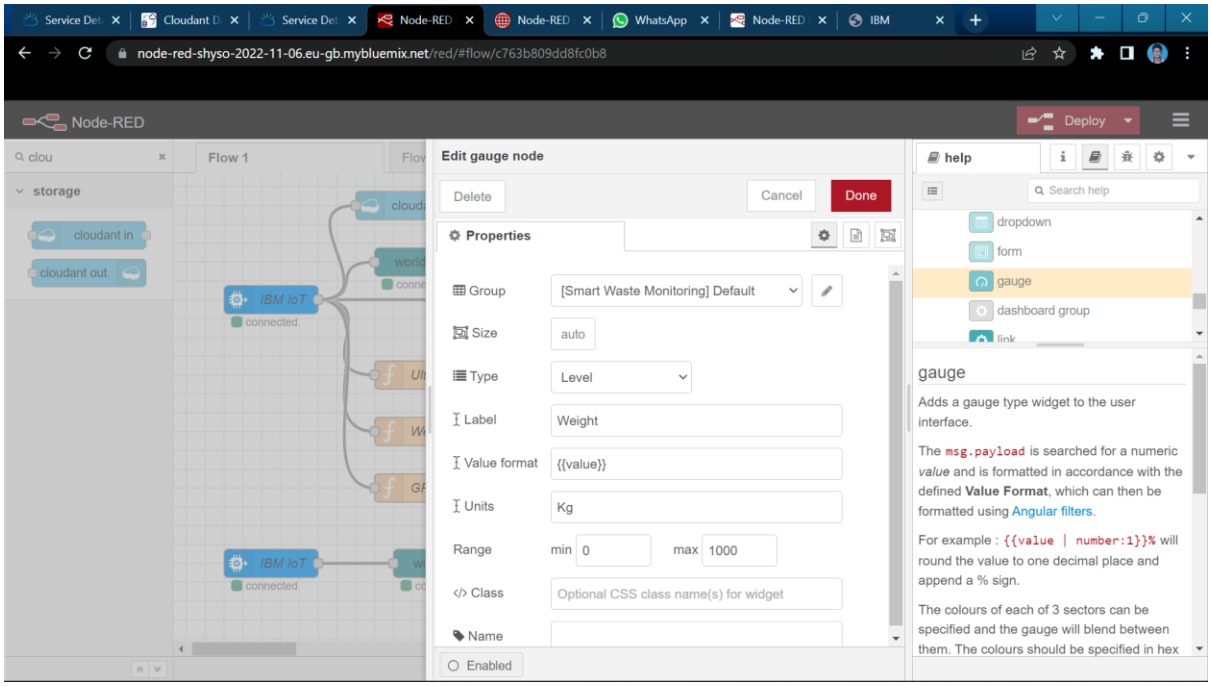
NODE RED APPLICATION

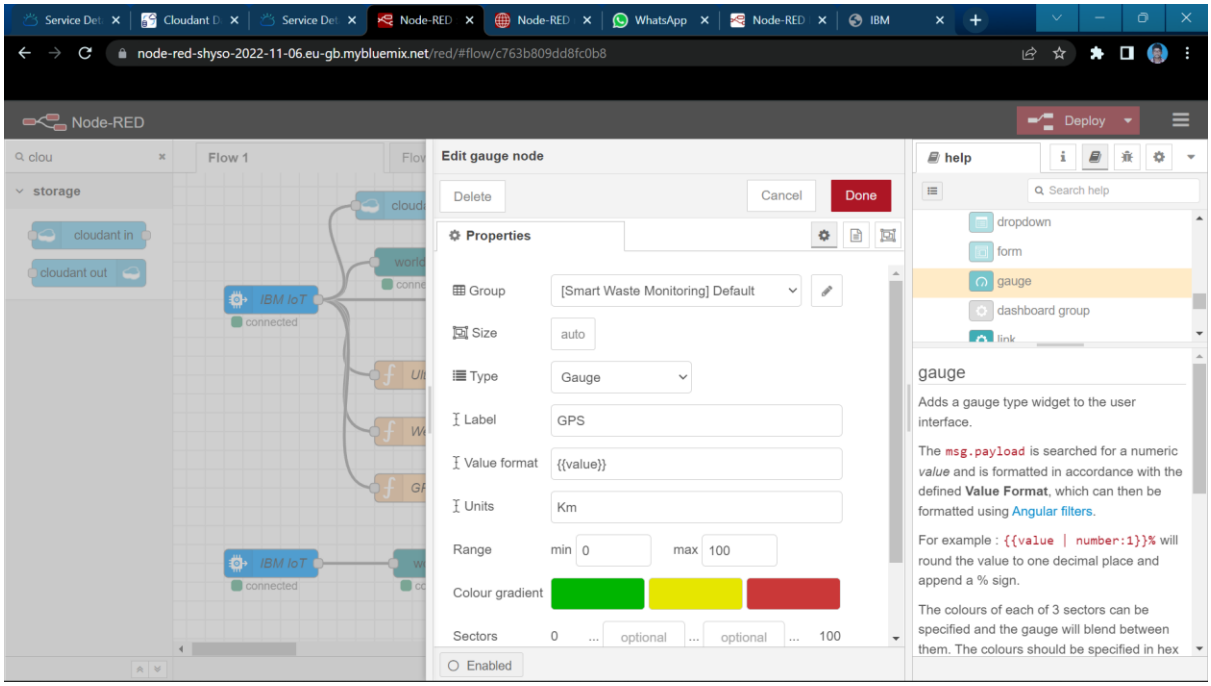


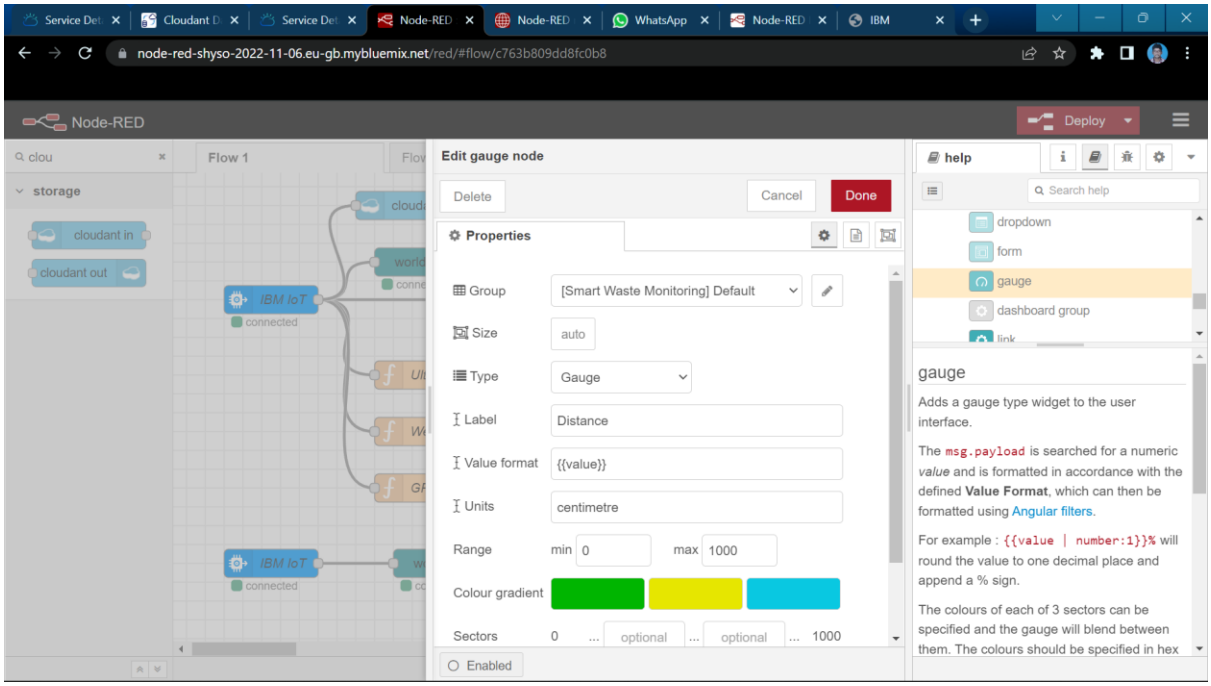


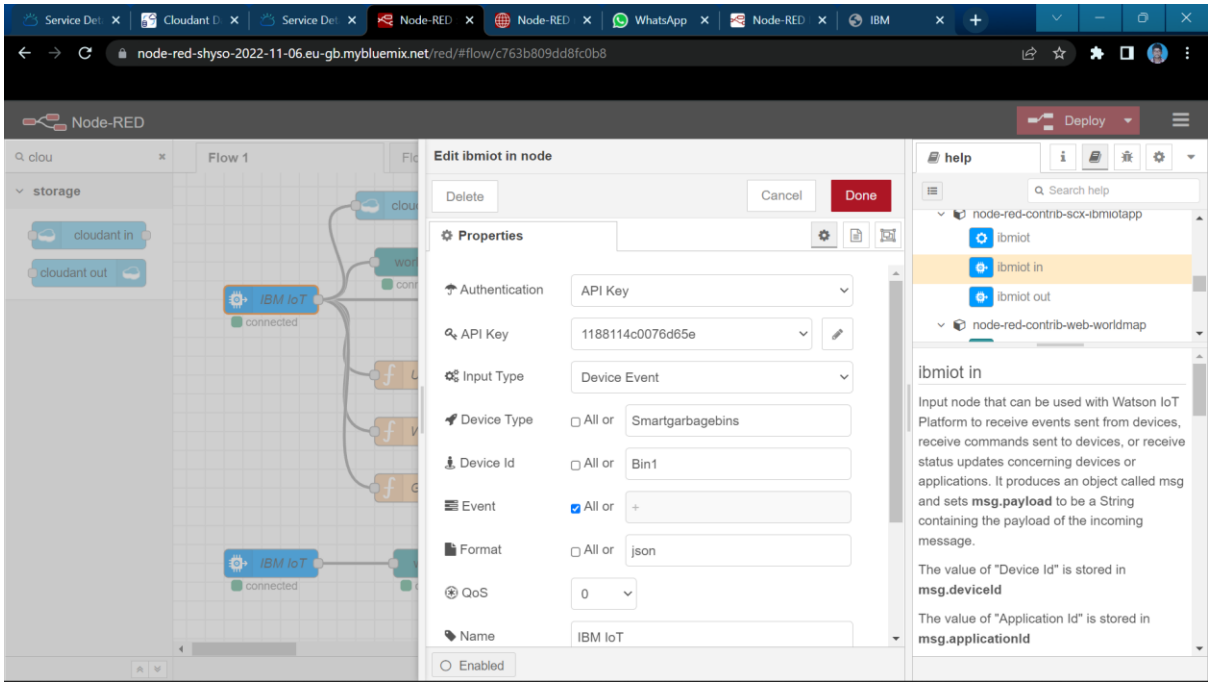

node-red-shyso-2022-11-06.eu-gb.mybluemix.net/red/#flow/c763b809dd8fc0b8
Node-RED
Deploy
Flow 1
storage
cloudant in
cloudant out
Edit function node
Delete
Cancel
Done
Properties
Name
Weight
Setup
On Start
On Message
On Stop
1 msg.payload=msg.payload.Weight;
2 return msg;
Enabled
help
Search help
delay
function
html
http in
function
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.
The On Start tab contains code that will be


node-red-shyso-2022-11-06.eu-gb.mybluemix.net/red/#flow/c763b809dd8fc0b8
Node-RED
Deploy
Flow 1
storage
cloudant in
cloudant out
IBM IoT
connected
Edit worldmap node
Delete
Cancel
Done
Properties
Group
[Smart Waste Monitoring] Default
Size
auto
Start
Latitude
Longitude
Zoom
1 - 18
Map list
7 selected
Base map
OpenStreetMap Greyscale
Overlays
5 selected
Cluster when
zoom level is less than
0 (0,off - 19)
Max age
Remove markers after
600
seconds
Enabled
help
Search help
ibmiot out
node-red-contrib-web-worldmap
worldmap
worldmap
worldmap
Plots "things" on a map in Node-RED Dashboard. Needs an internet connection.
It is possible to instantiate multiple worldmap nodes but each one must be given a unique path. Worldmap-in nodes are matched to worldmap nodes by having exactly the same path.
Shortcut - ⌘⇧m - ctrl-shift-m to jump to the default Map (/worldmap).
The minimum msg.payload must contain name, lat and lon properties, e.g.


node-red-shyso-2022-11-06.eu-gb.mybluemix.net/red/#flow/c763b809dd8fc0b8
Node-RED
Deploy
Flow 1
storage
cloudant in
cloudant out
IBM IoT
connected
Edit gauge node
Delete
Cancel
Done
Properties
Group
[Smart Waste Monitoring] Default
Size
auto
Type
Level
Label
Weight
Value format
{{value}}
Units
Kg
Range
min 0
max 1000
Class
Optional CSS class name(s) for widget
Name
Enabled
help
Search help
dropdown
form
gauge
dashboard group
gauge
Adds a gauge type widget to the user interface.
The msg.payload is searched for a numeric value and is formatted in accordance with the defined Value Format, which can then be formatted using Angular filters.
For example : {{value | number:1}}% will round the value to one decimal place and append a % sign.
The colours of each of 3 sectors can be specified and the gauge will blend between them. The colours should be specified in hex


node-red-shyso-2022-11-06.eu-gb.mybluemix.net/red/#flow/c763b809dd8fc0b8
Node-RED
Deploy
Flow 1
storage
cloudant in
cloudant out
IBM IoT
connected
Edit gauge node
Delete
Cancel
Done
Properties
Group
[Smart Waste Monitoring] Default
Size
auto
Type
Gauge
Label
GPS
Value format
{{value}}
Units
Km
Range
min 0
max 100
Colour gradient
Sectors
0
optional
optional
100
Enabled
help
Search help
dropdown
form
gauge
dashboard group
gauge
Adds a gauge type widget to the user interface.
The msg.payload is searched for a numeric value and is formatted in accordance with the defined Value Format, which can then be formatted using Angular filters.
For example : {{value | number:1}}% will round the value to one decimal place and append a % sign.
The colours of each of 3 sectors can be specified and the gauge will blend between them. The colours should be specified in hex





CODE:

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

```
import sys
#Provide your IBM Watson Device Credentials
organization = "a7mbs7"
deviceType = "Smartgarbagebins"
deviceId = "Bin1"
authMethod = "token"
authToken = "Sakthi@2001"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status =="lighton":
        print("led in on")
    else :
        print ("led is off")
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 oftype
"greeting" 10 times
deviceCli.connect()
while True:
    #Get Sensor Data from DHT11
    time.sleep(5)
    Ultrasonic=random.randint(0,80)
    Weight=random.randint(0,100)
    lat = round(random.uniform(11.03, 11.50), 6)
```

```
    long = round(random.uniform(76.80, 76.90), 6)
    GPS = str(lat) + str(',') + str(long)
    myData = {'Ultrasonic' : Ultrasonic, 'Weight' : Weight , 'GPS' : GPS}
    #print data
    def myOnPublishCallback():
        print ("Published Ultrasonic = %s Cm" %Ultrasonic, "Weight:%s kg " %Weight, "GPS: %s"%GPS)
    success = deviceCli.publishEvent("IoTSensor", "json", data=myData, qos=0,
    on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")
    time.sleep(1)
    deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

RESULT:

Thus the node-red application is successfully developed