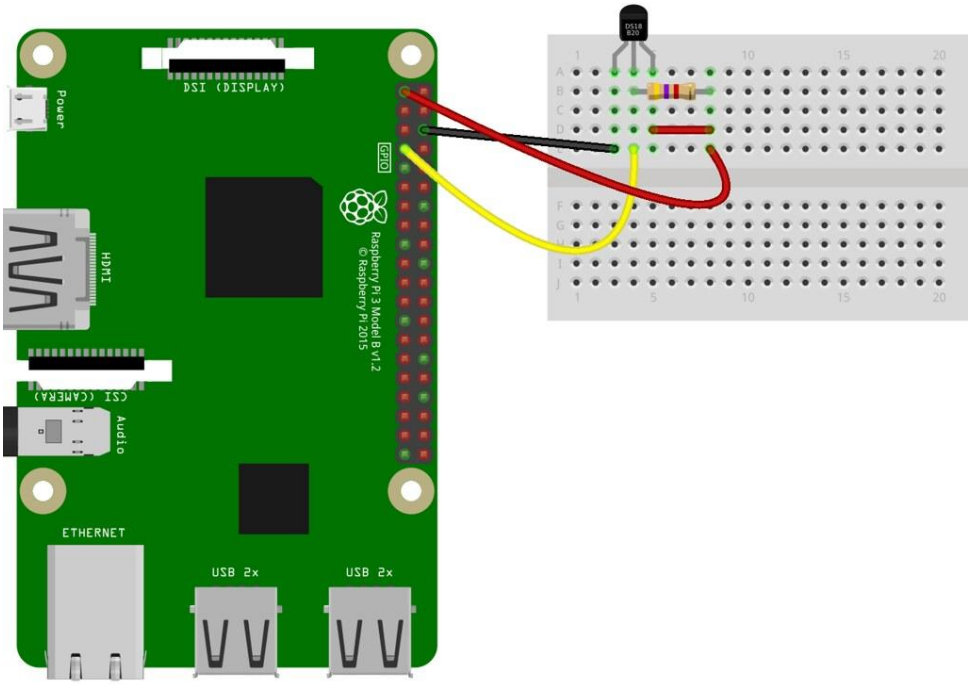
SPRINT 1

Date	18 October 2022
Team ID	PNT2022TMID27462
Project Name	Project - Hazardous Area Monitoring for Industrial Plant powered by IoT

Sprint 1 focuses on allowing users to get local data from beacons on their devices.

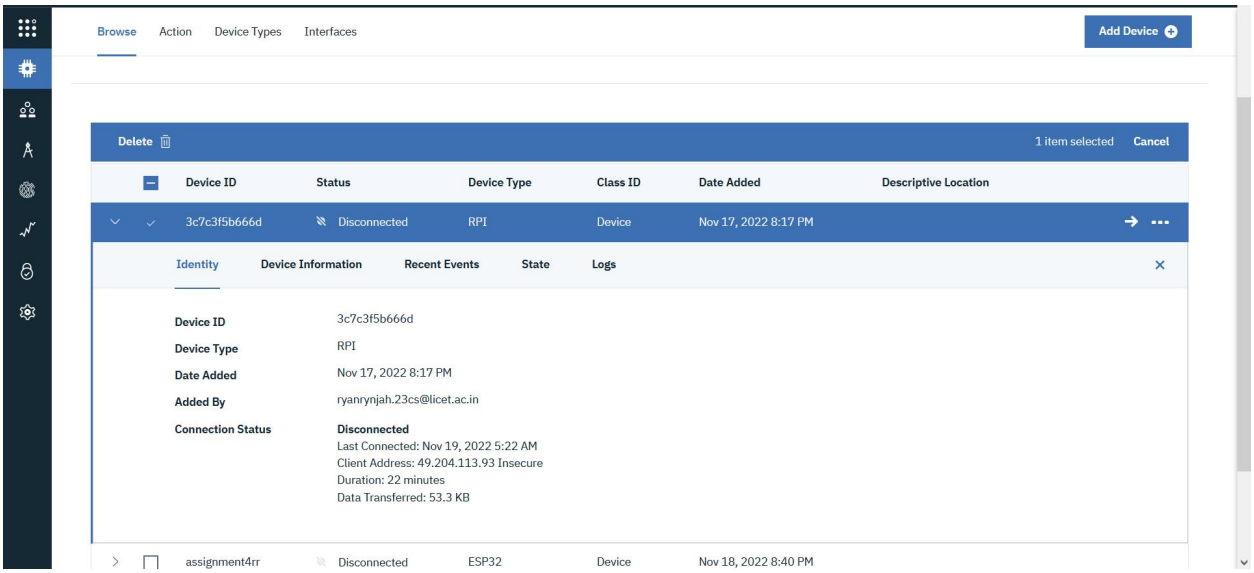
IoT device - Raspberry Pi 3B+



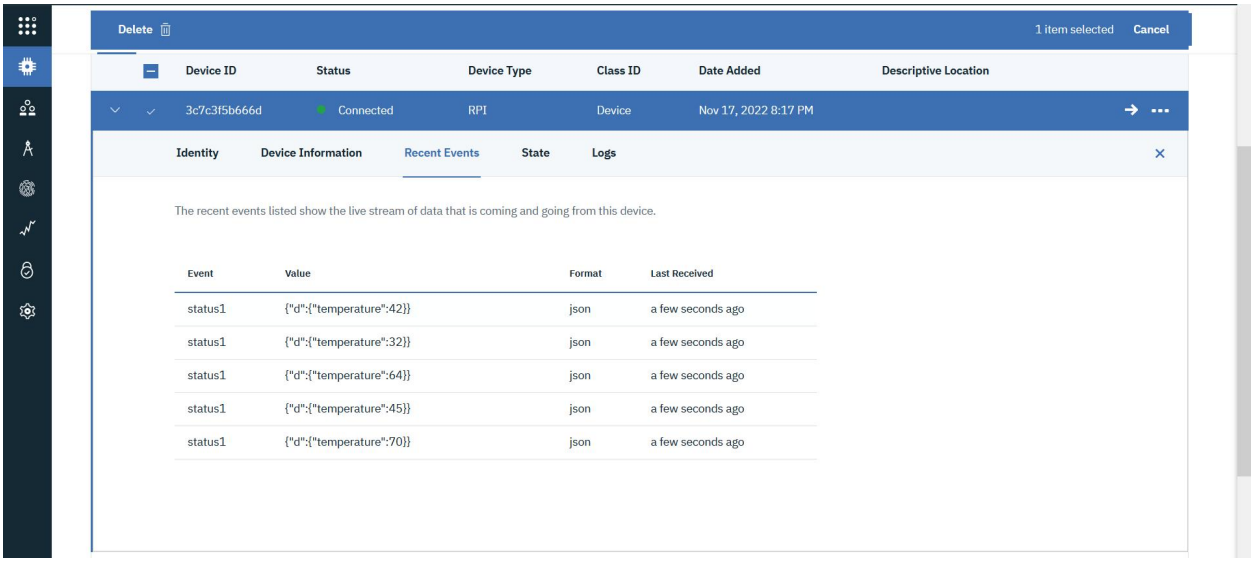

```
import time
import random
import paho.mqtt.client as mqtt
import json

#from w1thermsensor import W1ThermSensor
#sensor = W1ThermSensor()
#def Temp():
    #return temperature = sensor.get_temperature()
```

```
#The above lines of code would be used when getting temperature data from
a DS18B20 sensor.
#Due to hardware limitations we are simulating values using random
function.

def Temp():
    return random.randint(0,99);

ORG = "csgusn"
DEVICE_TYPE = "RPI"
TOKEN = "1123581321"
DEVICE_ID = "3c7c3f5b666d"  #Credentials of device as per created on IBM
IoT platform.

server = ORG + ".messaging.internetofthings.ibmcloud.com";
pubTopic1 = "iot-2/evt/status1/fmt/json";   #event named status 1 in JSON
format

authMethod = "use-token-auth";
token = TOKEN;
clientId = "d:" + ORG + ":" + DEVICE_TYPE + ":" + DEVICE_ID;

mqttc = mqtt.Client(client_id=clientId)
mqttc.username_pw_set(authMethod, token)
mqttc.connect(server, 1883, 60)         #Connecting using MQ Telemetry
Transport Protocol

while True:

    tempDict = { "d": {"temperature": Temp()} };    #Temporary storage in
a dictionary

    tempJson = json.dumps(tempDict);    #Conversion from dictionary to
JSON

    mqttc.publish(pubTopic1, tempJson)      #Publish payload
    print("Reading Taken");

    time.sleep(5);
```

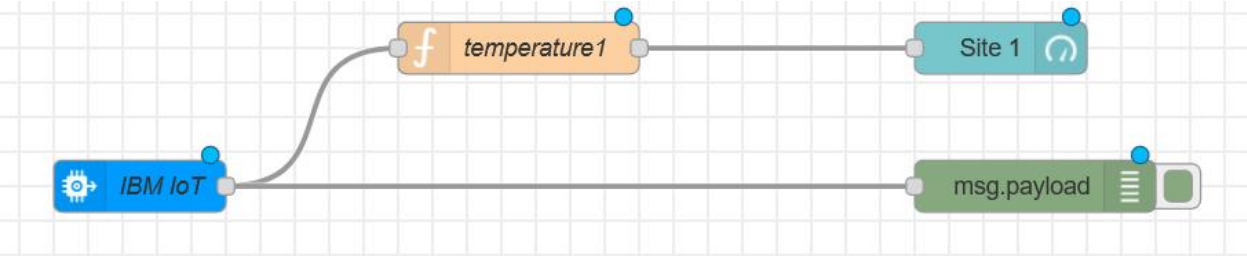
IBM IoT Platform



The device as created in IBM watson platform. On running the program, Status changes to Connected and published temperature data is shown in recent events.

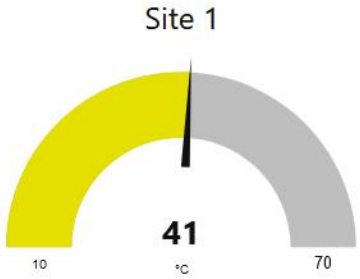


Node-RED



1. IBM IoT IN node allows us to get data from IBM IoT device.
2. Temperature is a function that extracts the payload.
   global.set("temperature1",msg.payload.d.temperature1)
   msg.payload=msg.payload.d.temperature1
   return msg;

3. Site is a gauge to view values on the Node-RED dashboard.



4. msg.payload allows for debugging.

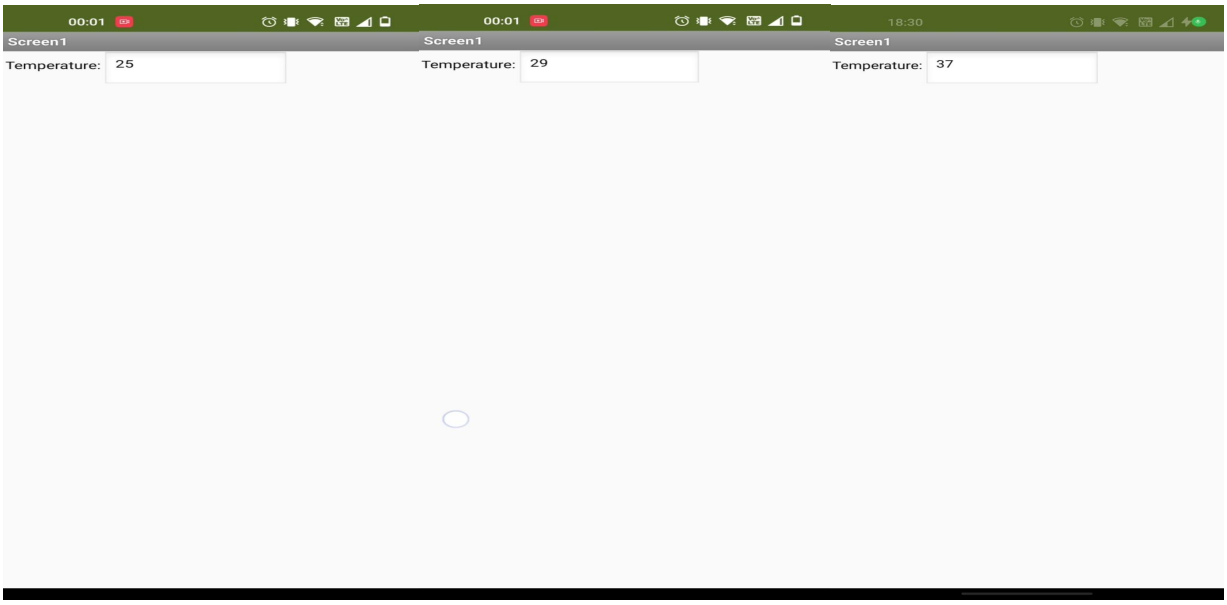
5. [get]/data is a GET HTTP method which sends the json data through the http node.

```
{
  "d": {
    "temperature": 42
  }
}
```

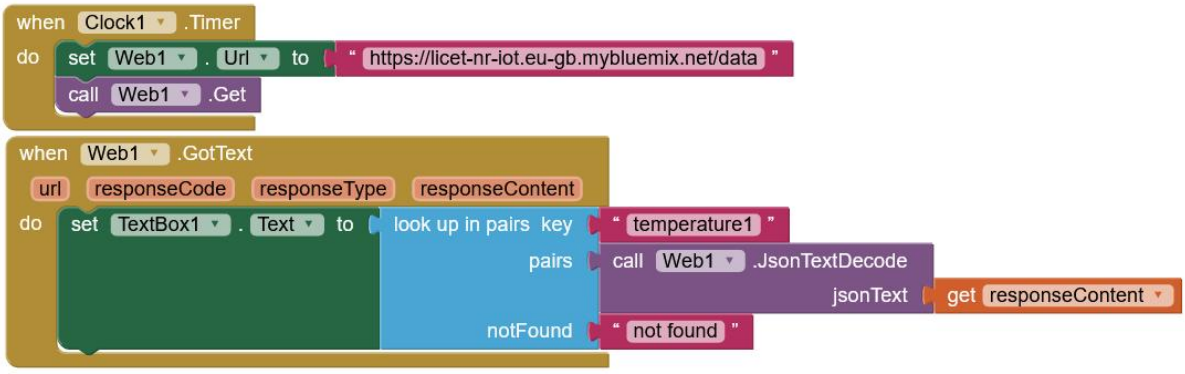
6. This allows us to export the data through HTTP for use by other applications.

User Device

Employee user interface is an Android Application.



MIT App Inventor



The app fetches the JSON data from the HTTP out from Node-RED, extracts the relevant data and pastes it in a text box.