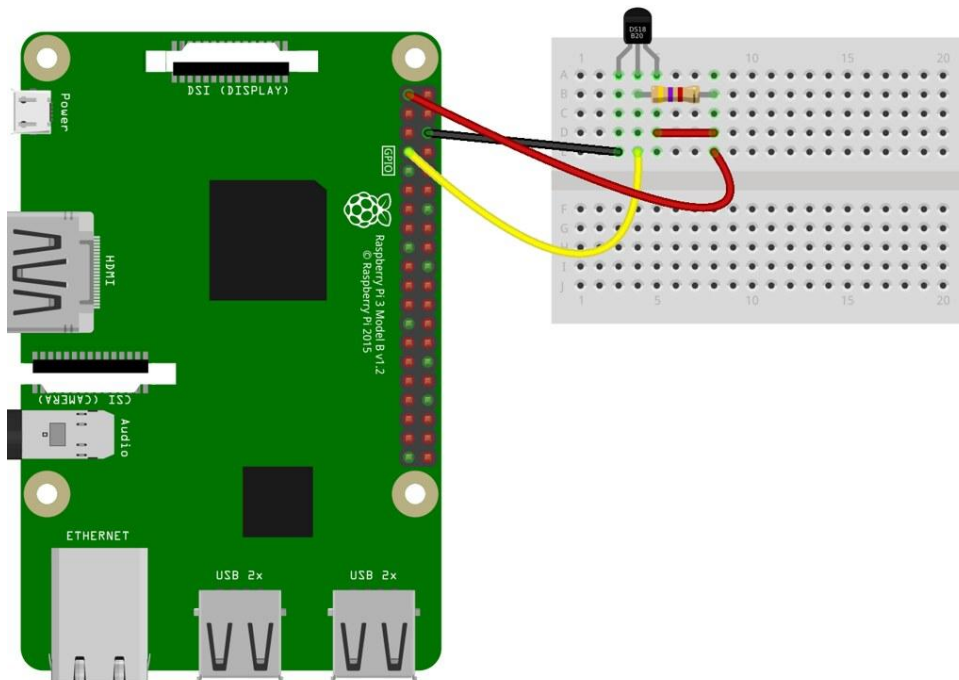


SPRINT 1

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 wlthermsensor import WlThermSensor
#sensor = WlThermSensor()
#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 screenshot shows the IBM IoT Platform interface. At the top, there are tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A 'Delete' button is visible on the left, and an 'Add Device' button is on the right. The main content area displays a table of devices. The first device, with ID '3c7c3f5b666d', is in a 'Disconnected' state. Below the table, a detailed view of the selected device is shown, including its identity, device information, recent events, state, and logs. The 'Device Information' section shows the device ID, type (RPI), date added (Nov 17, 2022 8:17 PM), and added by (ryanrynjah.23cs@licet.ac.in). The 'Connection Status' section shows the device is 'Disconnected' with details on the last connection attempt.

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
3c7c3f5b666d	Disconnected	RPI	Device	Nov 17, 2022 8:17 PM	

Device Information

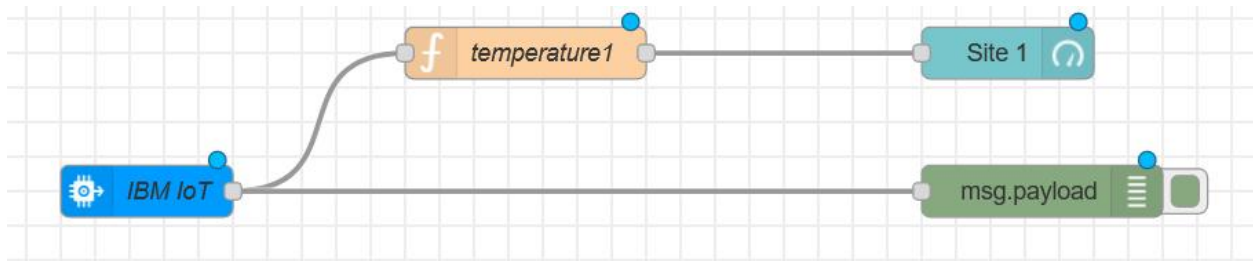
- Device ID: 3c7c3f5b666d
- Device Type: RPI
- Date Added: Nov 17, 2022 8:17 PM
- Added By: ryanrynjah.23cs@licet.ac.in
- Connection Status: **Disconnected**
Last Connected: Nov 19, 2022 5:22 AM
Client Address: 49.204.113.93 Insecure
Duration: 22 minutes
Data Transferred: 53.3 KB

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.

The screenshot shows the IBM IoT Platform interface with the same device now in a 'Connected' state. The 'Recent Events' tab is selected, showing a live stream of data. The events are listed in a table with columns for Event, Value, Format, and Last Received. The events show temperature data being published at regular intervals.

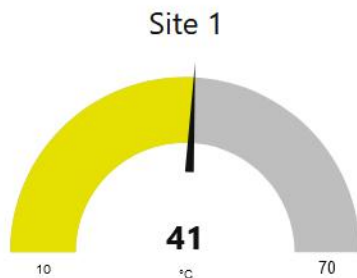
Event	Value	Format	Last Received
status1	{"d":{"temperature":42}}	json	a few seconds ago
status1	{"d":{"temperature":32}}	json	a few seconds ago
status1	{"d":{"temperature":64}}	json	a few seconds ago
status1	{"d":{"temperature":45}}	json	a few seconds ago
status1	{"d":{"temperature":70}}	json	a few seconds ago

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.



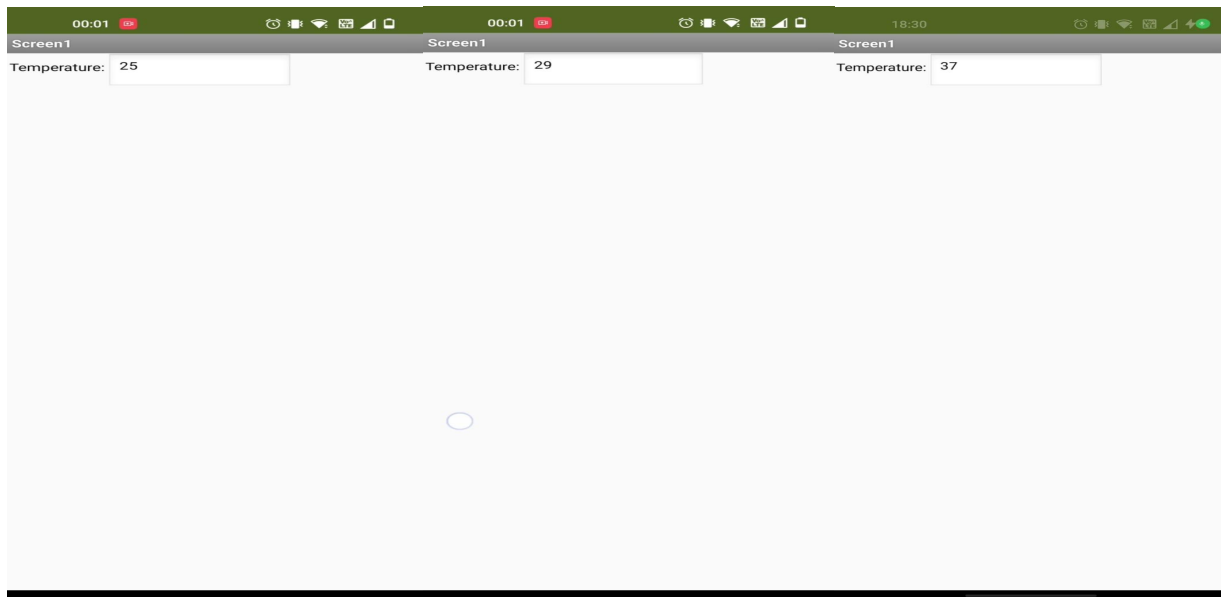
5. [get]/data is a GET HTTP method which sends the json data through the http node.

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

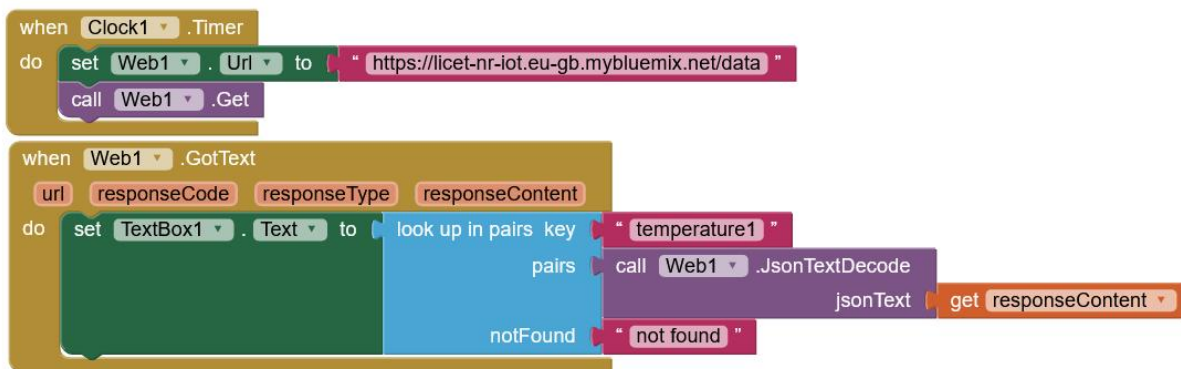
6. This allows us to export the data through HTTP for use by other applications.

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.