

**Project Development Phase**  
**Sprint III**

Date	17 November 2022
Team ID	PNT2022TMID19266
Project Name	Signs with Smart Connectivity for better road safety

**Sprint Targets:**

<b>Sprint</b>	<b>Functional Requirement (Epic)</b>	<b>User Story Number</b>	<b>User Story/Task</b>	<b>Story Points</b>	<b>Priority</b>	<b>Team Members</b>
Sprint-3	Login	USN-5	I need a website account because I'm the administrator.	7	Low	Kishor M Kishore S M Krishna M Naveen V
Sprint-3	Dashboard	USN-6SSS	I should be able to watch over and add sign nodes as an administrator.	13	Medium	Kishor M Kishore S M Krishna M Naveen V

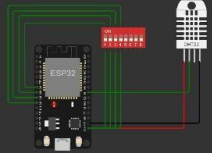
## WOWKI Simulation:

WOWKI SAVE SHARE final\_iot.ino copy Docs

sketch.ino diagram.json libraries.txt Library Manager

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #include "DHT.h" // Library for dht11
4 #define DHTPIN 5 // what pin we're connected to
5 #define DHTTYPE DHT22 // define type of sensor DHT 11
6
7 DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and type of dht connect
8
9 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
10
11 //-----credentials of IBM Accounts-----
12
13 #define ORG "psh4py" //IBM ORGANIZATION ID
14 #define DEVICE_TYPE "alert-device" //Device type mentioned in ibm watson IOT Platform
15 #define DEVICE_ID "4571" //Device ID mentioned in ibm watson IOT Platform
16 #define TOKEN "12345678" //Token
17 String data3;
18 float h, t;
19
20
21 //----- Customise the above values -----
22 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
23 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event perform a
24 char subscribetopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT command type AND
25 char authMethod[] = "use-token-auth"; // authentication method
26 char token[] = TOKEN;
27 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
28
29
30 //-----
31 WiFiClient wificlient; // creating the instance for wificlient
32 PubSubClient client(server, 1883, callback, wificlient); //calling the predefined client
33
34
```

Simulation 01:47.514 98%



temp:37.40  
humidity:86.00  
Sending payload:  
{ "temp":37.40,"humidity":86.00,"North":0,"South":0,"East":0,"West":0}  
Publish ok  
Reconnecting client to psh4py.messaging.internetofthings.ibmcloud.com  
.....

Type here to search 23°C Cloudy 08:23 13-11-2022

## IoT Device – IoT Platform

The screenshot displays the 'Recent Events' tab for a device with ID 0001. The interface includes a top navigation bar with 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area shows a table of recent events, each containing an event name, a JSON value, the format, and the time received. A status box at the bottom right indicates '1 Simulation running'.

Device ID: 0001, Status: Disconnected, Device Type: edge-device-1, Class ID: Device, Date Added: Nov 5, 2022 8:56 PM

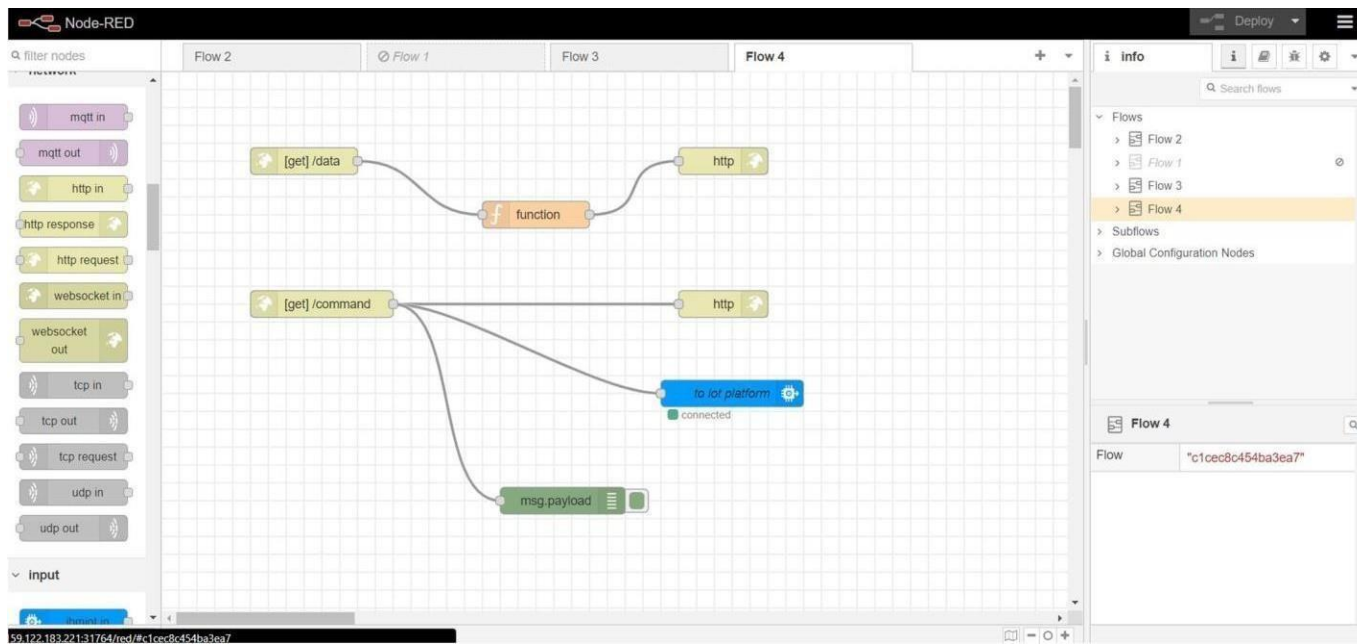
Identity | Device Information | **Recent Events** | State | Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
rnd_number	{"Lane_1":5,"Lane_2":83,"Lane_3":30,"Lane_4":...	json	a few seconds ago
rnd_number	{"Lane_1":59,"Lane_2":59,"Lane_3":94,"Lane_4":...	json	a few seconds ago
rnd_number	{"Lane_1":93,"Lane_2":88,"Lane_3":49,"Lane_4":...	json	a few seconds ago
rnd_number	{"Lane_1":2,"Lane_2":61,"Lane_3":21,"Lane_4":...	json	a few seconds ago
rnd_number	{"Lane_1":70,"Lane_2":11,"Lane_3":69,"Lane_4":...	json	a few seconds ago

1 Simulation running

## Node Red – Connect with MIT App Inventor:



## Edit function node

Delete

Cancel

Done

### Properties

Name

Name

Setup

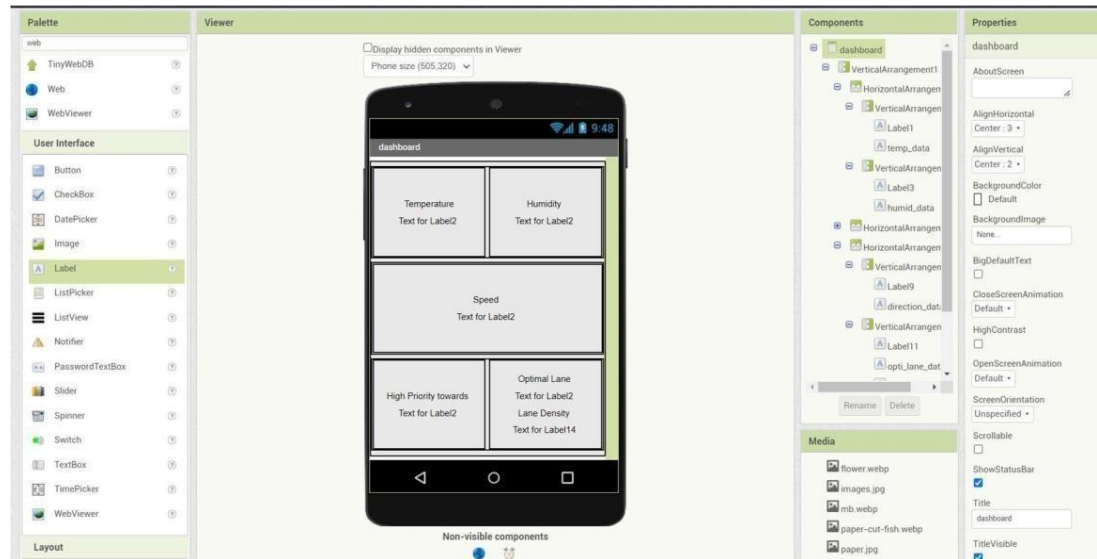
On Start

On Message

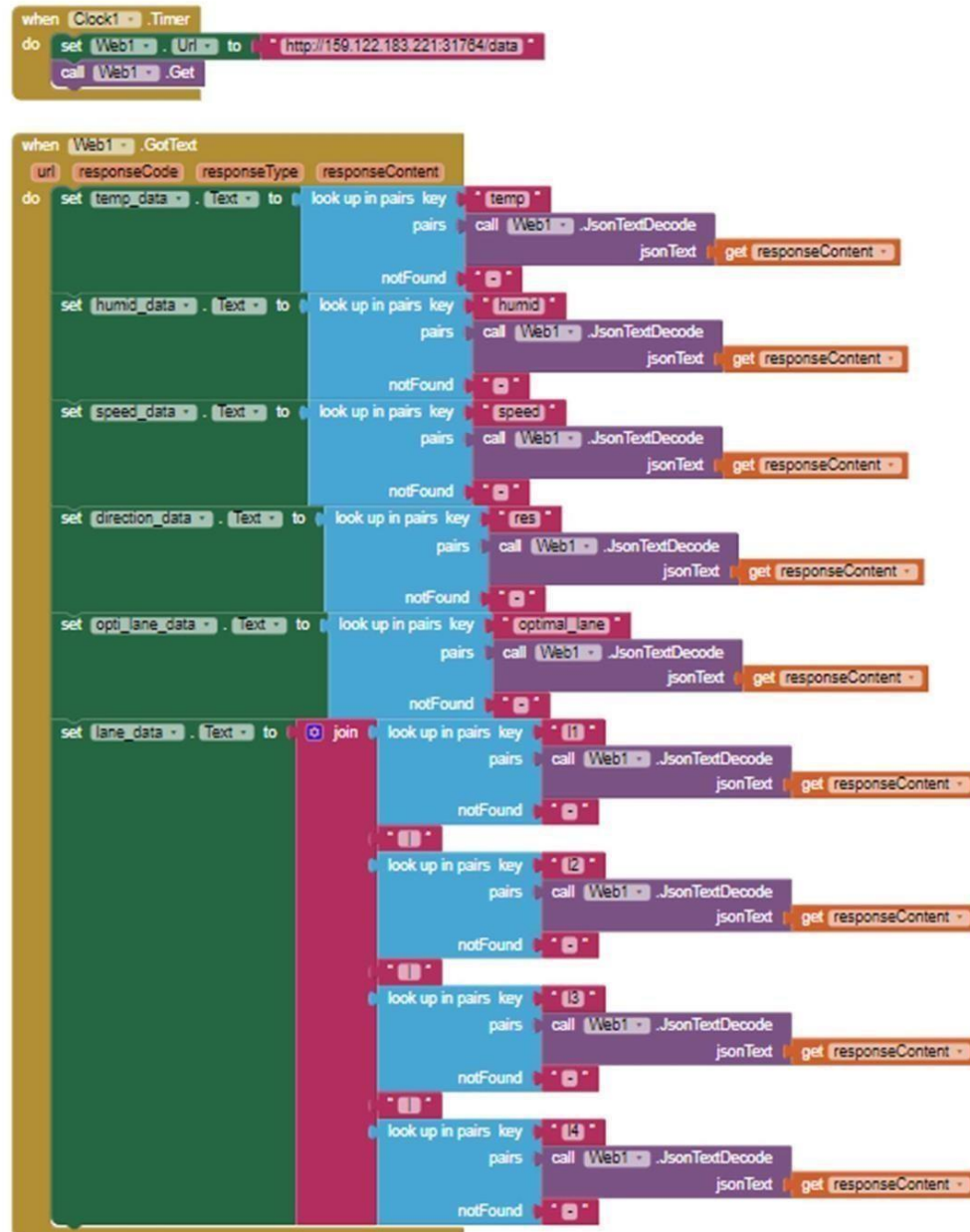
On Stop

```
1 msg.payload = {  
2   "temp":global.get("temp"),  
3   "humid":global.get("humid"),  
4   "speed":global.get("speed"),  
5   "n":global.get("n"),  
6   "s":global.get("s"),  
7   "e":global.get("e"),  
8   "w":global.get("w"),  
9   "res":global.get("res"),  
10  "l1":global.get("l1"),  
11  "l2":global.get("l2"),  
12  "l3":global.get("l3"),  
13  "l4":global.get("l4"),  
14  "optimal_lane":global.get("optimal_lane")  
15 };  
16  
17  
18 return msg;
```

## MIT App Inventor UI design:



## MIT App Inventor Backend design:



**Sprint 3 delivery:**

**(OUTPUT) Display from MIT App:**

