

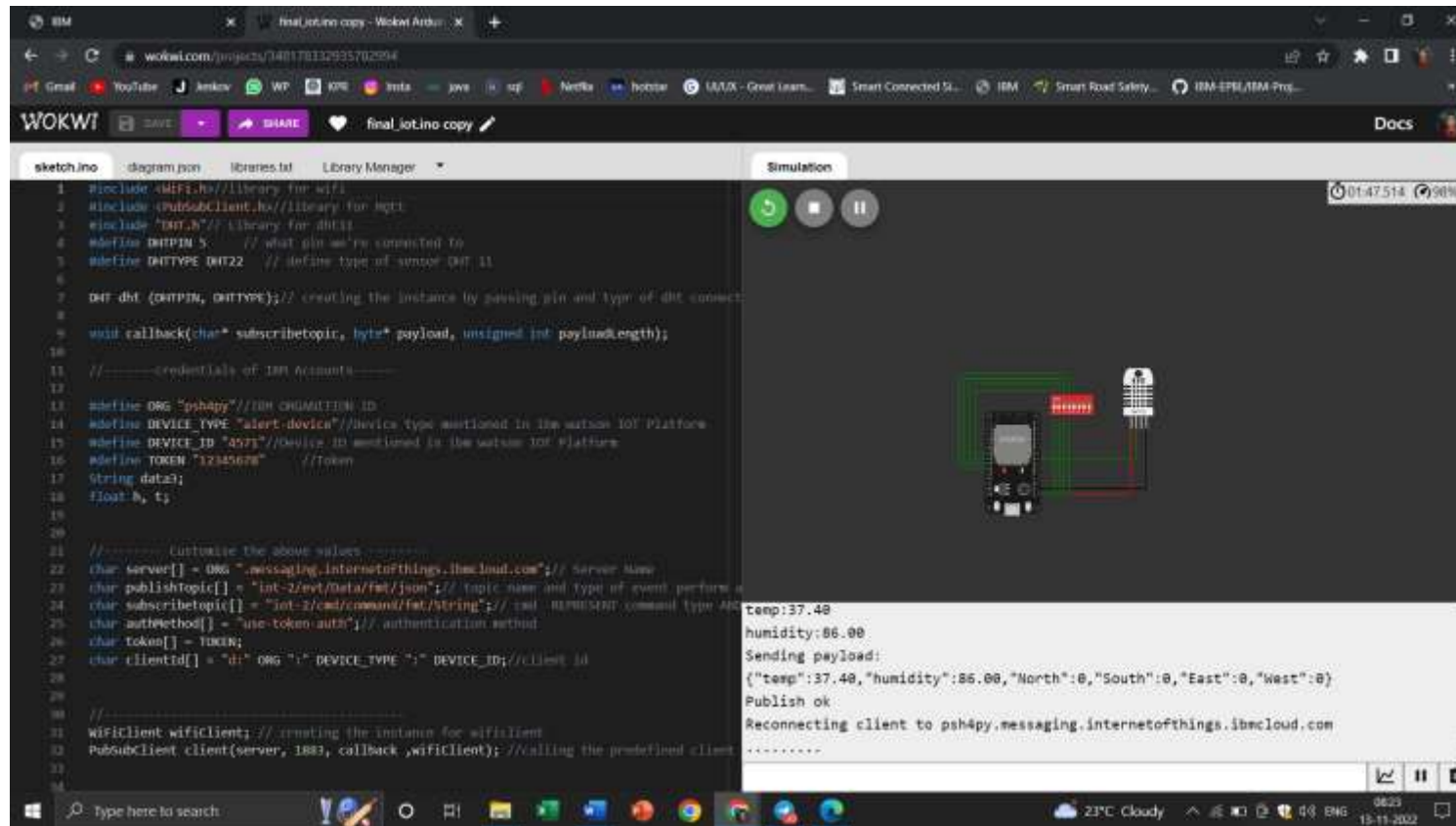
**Project Development Phase**  
**Sprint III**

Date	13 November 2022
Team ID	PNT2022TMID41867
Project Name	Signs with Smart Connectivity for better road safety

**Sprint Targets :**

Sprint	Functional Requirement (Epic)	UserStory Number	User Story /Task	Story Points	Priority	Team Members
Sprint-3	Login	USN-5	As an administrator , I should have an account of the website	7	Low	Saranya Sandhiya Santhiya Sakthikala Kariyaraman
Sprint-3	Dashboard	USN-6SSS	As an admin , I should be able to monitor and add sign nodes	13	Medium	Saranya Sandhiya Santhiya Sakthikala Kariyaraman

Wokwi Simulation: <https://wokwi.com/projects/348178332935782994>



The screenshot displays the Wokwi web interface for an Arduino simulation. The top navigation bar includes 'WOKWI', 'SAVE', 'SHARE', and the project name 'final\_lotino copy'. Below this, tabs for 'sketch.ino', 'diagram.json', 'libraries.txt', and 'Library Manager' are visible. The 'sketch.ino' tab is active, showing the following code:

```
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* subscribtopic, byte* payload, unsigned int payloadLength);
10
11 //----- credentials of IBM Accounts -----
12
13 #define ORG "psh4py" // IBM ORG/PROJECT ID
14 #define DEVICE_TYPE "alert-device" // Device type mentioned in the Watson IoT Platform
15 #define DEVICE_ID "4571" // Device ID mentioned in the 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[] = "int-2/evt/Data/fat/json"; // topic name and type of event perform a
24 char subscribtopic[] = "int-2/cmd/command/fat/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 wifi client
32 PubSubClient client(server, 1883, callback, wifiClient); // calling the predefined client
33
34
```

The 'Simulation' tab is active, showing a circuit diagram of an Arduino Uno connected to a DHT22 sensor. The terminal window displays the following output:

```
temp:37.48
humidity:86.00
Sending payload:
{"temp":37.48,"humidity":86.00,"North":0,"South":0,"East":0,"West":0}
Publish ok
Reconnecting client to psh4py.messaging.internetofthings.ibmcloud.com
.....
```

The bottom status bar shows the system clock as 08:23 on 13-11-2022, and the weather as 23°C Cloudy.

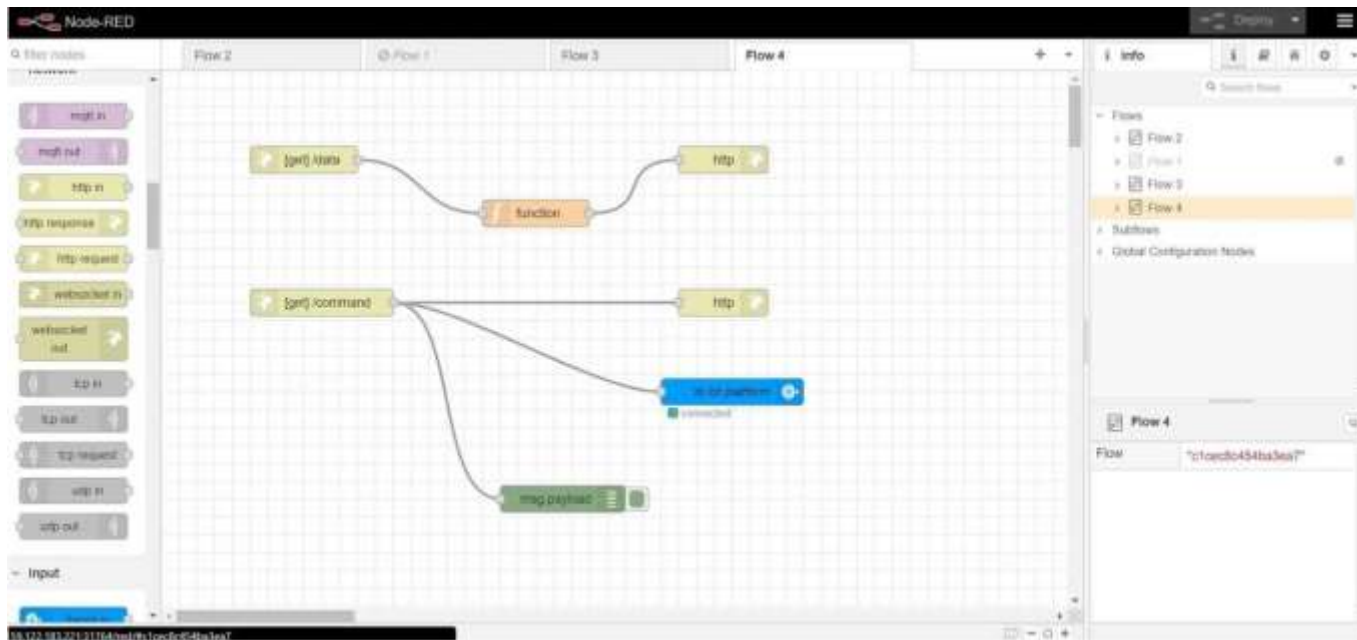
## IoT Device – IoT Platform

The screenshot displays the 'Recent Events' tab for a device named '0001'. The device is currently 'Disconnected' and is an 'edge-device-1' of type 'Device'. It was last seen on 'Nov 9, 2022 8:56 PM'. The interface includes a sidebar with navigation icons and a top navigation bar with tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A table lists recent events, each containing an 'Event' type, a 'Value' (a JSON array of lane numbers), a 'Format' (JSON), and a 'Last Received' timestamp. A status box at the bottom right indicates '1 Simulation running'.

Event	Value	Format	Last Received
mdl_number	["Lane_1":5,"Lane_2":83,"Lane_3":30,"Lane_4":...	json	a few seconds ago
mdl_number	["Lane_1":58,"Lane_2":99,"Lane_3":94,"Lane_4":...	json	a few seconds ago
mdl_number	["Lane_1":93,"Lane_2":88,"Lane_3":69,"Lane_4":...	json	a few seconds ago
mdl_number	["Lane_1":2,"Lane_2":61,"Lane_3":21,"Lane_4":...	json	a few seconds ago
mdl_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 AppInventor



## 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;
```

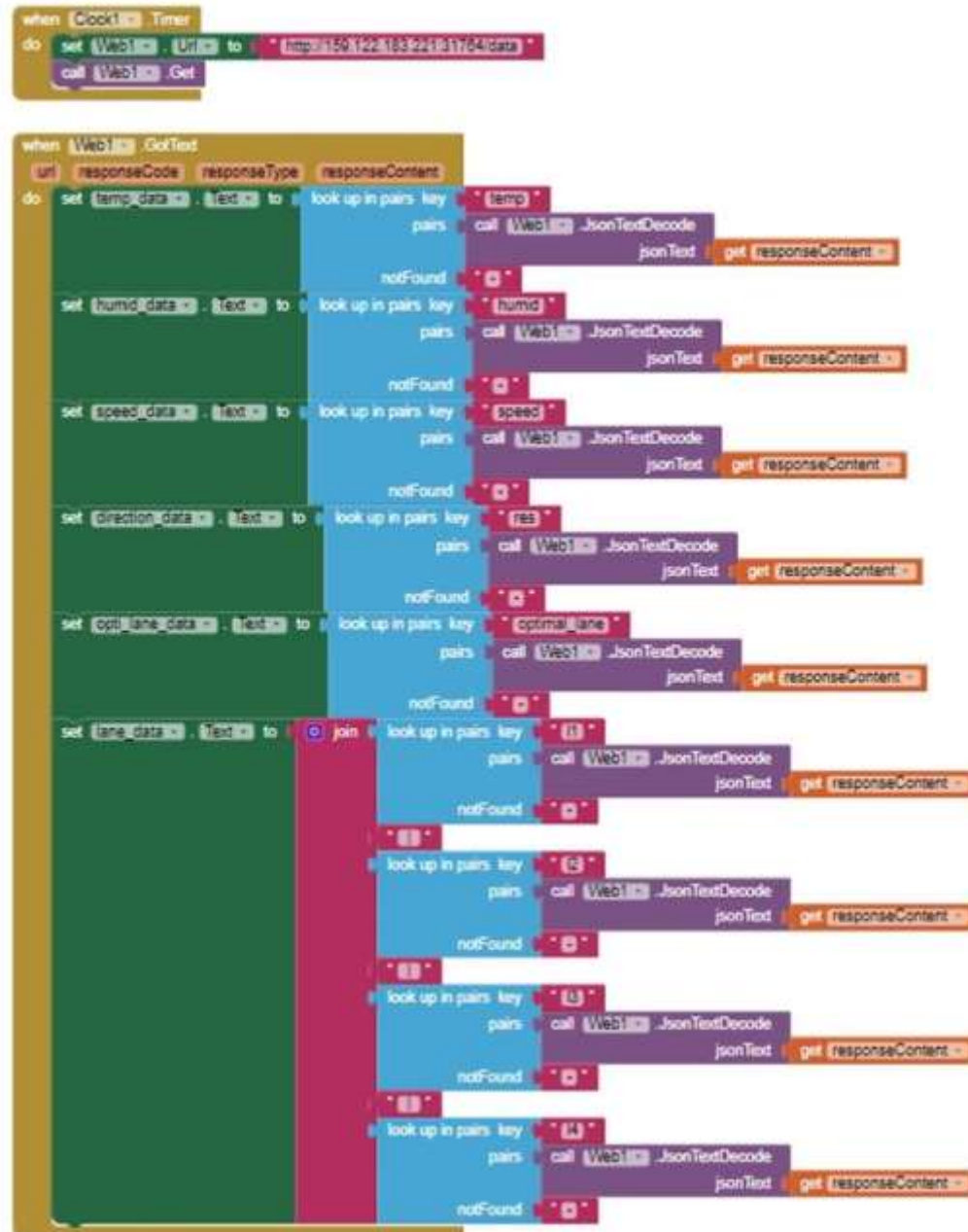
Output from Node red:

```
["temp":14.9,"humid":86,"speed":89,"n":0,"s":0,"e":0,"w":1,"res":"West","11":69,"12":99,"13":19,"14":40,"optimal_lane":"Lane 3"]
```

MIT App Inventor UI design:



## MIT App Inventor Backend design:



**Sprint 3 delivery:**

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

