Project Development Phase
SPRINT 3

Date	12 November 2022
Team ID	PNT2022TMID53557
Project Name	Gas leakage monitoring and alerting system

Data Transfer:

As a system, it should send the data of sensor values along with latitudes and longitudes to the IBM cloud

```
#include <WiFi.h>

#include <PubSubClient.h>

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

//-------credentials of IBM Accounts------

#define ORG "oqhi1j"//IBM ORGANITION ID

#define DEVICE_TYPE "NODEMCU"//Device type mentioned in ibm watson IOT Platform

#define DEVICE_ID "BHAVAN0108"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "bharathi0503" //Token

String data3;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char publishTopic[] = "iot-2/evt/Gas/fmt/json";

char publishTopic2[] = "iot-2/evt/Loc/fmt/json";

char subscribetopic[] = "iot-2/cmd/home/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

WiFiClient wifiClient;

PubSubClient client(server, 1883, callback ,wifiClient);

const int gasSensor = A0;

#define SOUND_SPEED 0.034

int gasValue = 0;

String latitude = "0.000000";

String longitude = "0.000000";

void setup()

{
```

```cpp
  Serial.begin(115200);
  wificonnect();
  mqttconnect();
}
void loop()
{
gasValue = random(600,750);
Serial.print("Gas Value: ");
Serial.println(gasValue);
delay(1000);
PublishData(gasValue);
delay(1000);
if(gasValue > 700)
{
latitude = "13.148760";
longitude = "80.229100";
PublishString(latitude, longitude);
}
if (!client.loop())
{
mqttconnect();
}
Serial.println();
Serial.println("----------------------------------------------------------------------");
Serial.println();
delay(3000);
}
void PublishData(int gas)
{
mqttconnect();
String payload = "{\"Gas Value\":";
payload += gas;
payload += "}";
Serial.print("Sending payload Gas: ");
```

```cpp
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str()))
{
Serial.println("Gas is Published");
}
else
{
Serial.println("Gas is not Published");
}
}
void PublishString(String lat, String lon)
{
mqttconnect();
String payload2 = "{\"d\":{\"Latitude\":";
payload2 += lat;
payload2 += ","""\"Longitude\":";
payload2 +=lon;
payload2 +="}}";
Serial.print("Sending Payload Location: ");
Serial.println(payload2);
if (client.publish(publishTopic2, (char*) payload2.c_str()))
{
Serial.println("Location is Published");
}
else
{
Serial.println("Location is not Published");
}
}
void mqttconnect()
{
if (!client.connected())
{
Serial.print("Reconnecting client to ");
```

```cpp
Serial.println(server);
while (!!!client.connect(clientId, authMethod, token))
{
Serial.print(".");
delay(500);
}
initManagedDevice();
Serial.println();
}
}
void wificonnect()
{
Serial.println();
Serial.print("Connecting to ");
WiFi.begin("Wokwi-GUEST", "", 6);
while (WiFi.status() != WL_CONNECTED)
{
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}
void initManagedDevice()
{
if (client.subscribe(subscribetopic))
{
Serial.println((subscribetopic));
Serial.println("subscribe to cmd OK");
}
else
{
```

```
Serial.println("subscribe to cmd FAILED");

}

}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)

{

Serial.print("callback invoked for topic: ");

Serial.println(subscribetopic);

for (int i = 0; i < payloadLength; i++)

{

//Serial.print((char)payload[i]);

data3 += (char)payload[i];

}

Serial.println("data: "+ data3);

data3="";

}
```

```
Connecting to ...
WiFi connected
IP address:
10.10.0.2
Reconnecting client to oqhi1j.messaging.internetofthings.ibmcloud.com
iot-2/cmd/home/fmt/String
subscribe to cmd OK

Gas Value: 645
Sending payload Gas: {"Gas Value":645}
Gas is Published

-----------------------------------------------------------------

Gas Value: 672
Sending payload Gas: {"Gas Value":672}
Gas is Published

-----------------------------------------------------------------

Gas Value: 619
Sending payload Gas: {"Gas Value":619}
Gas is Published
```

```
Gas Value: 631
Sending payload Gas: {"Gas Value":631}
Gas is Published


----------------------------------------------------------------


Gas Value: 720
Sending payload Gas: {"Gas Value":720}
Gas is Published
Sending Payload Location: {"d":{"Latitude":13.148760,"Longitude":80.229100}}
Location is Published


----------------------------------------------------------------
```

Browse   Action   Device Types   Interfaces   Add Device ⊕

| 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
Loc	{"d":{"Latitude":13.14876,"Longitude":80.2291}}	json	a few seconds ago
Gas	{"Gas Value":720}	json	a few seconds ago
Gas	{"Gas Value":631}	json	a few seconds ago
Gas	{"Gas Value":658}	json	a few seconds ago
Gas	{"Gas Value":688}	json	a few seconds ago

Items per page 50 ▾ | 1–1 of 1 item   1 of 1 page   ‹   1 ▾   ›

As a cloud system, the IBM cloud should send the data to NodeRed As a system, it should collect the data from the NodeRed and give it to the backend of the mit app.
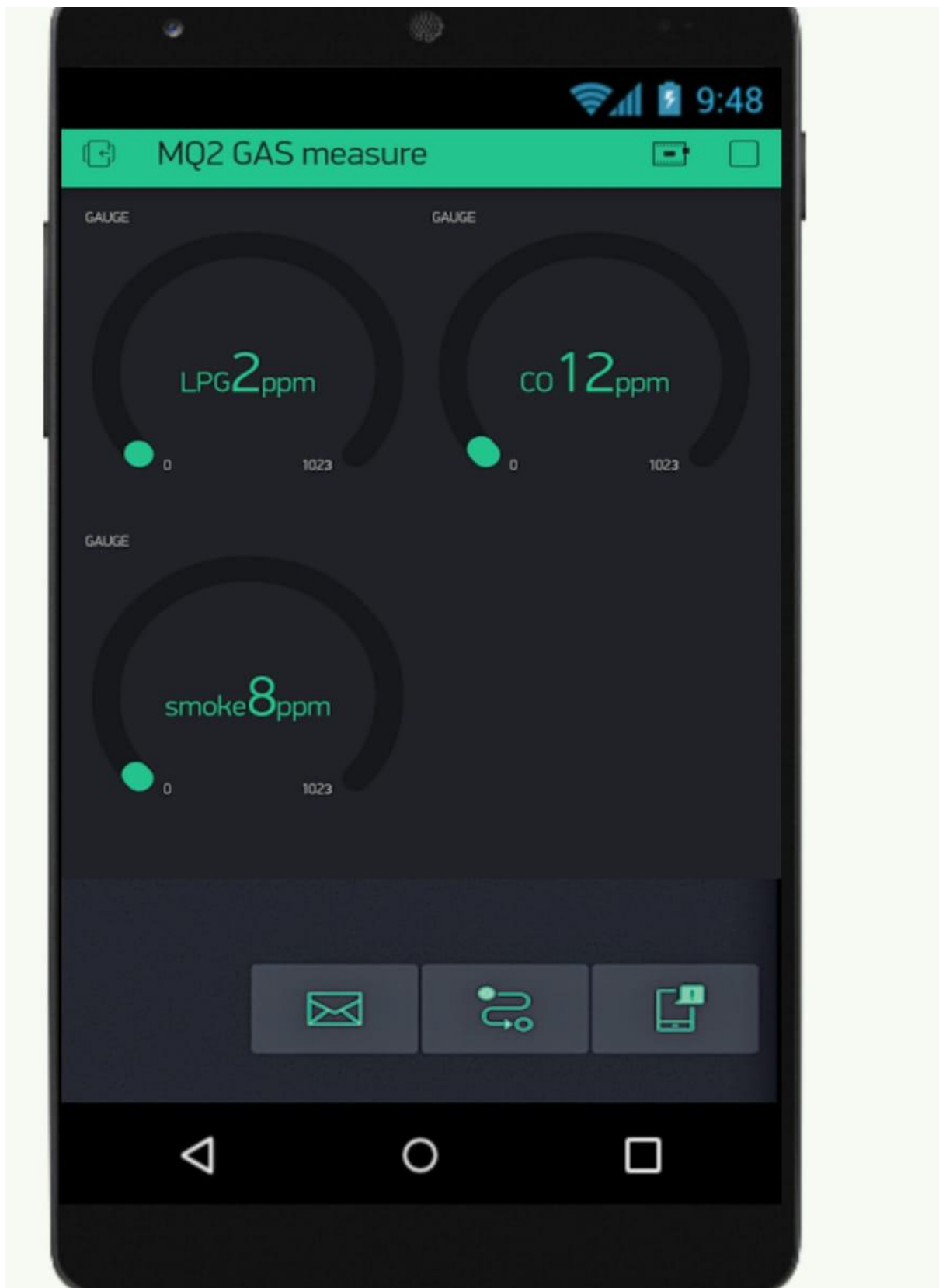
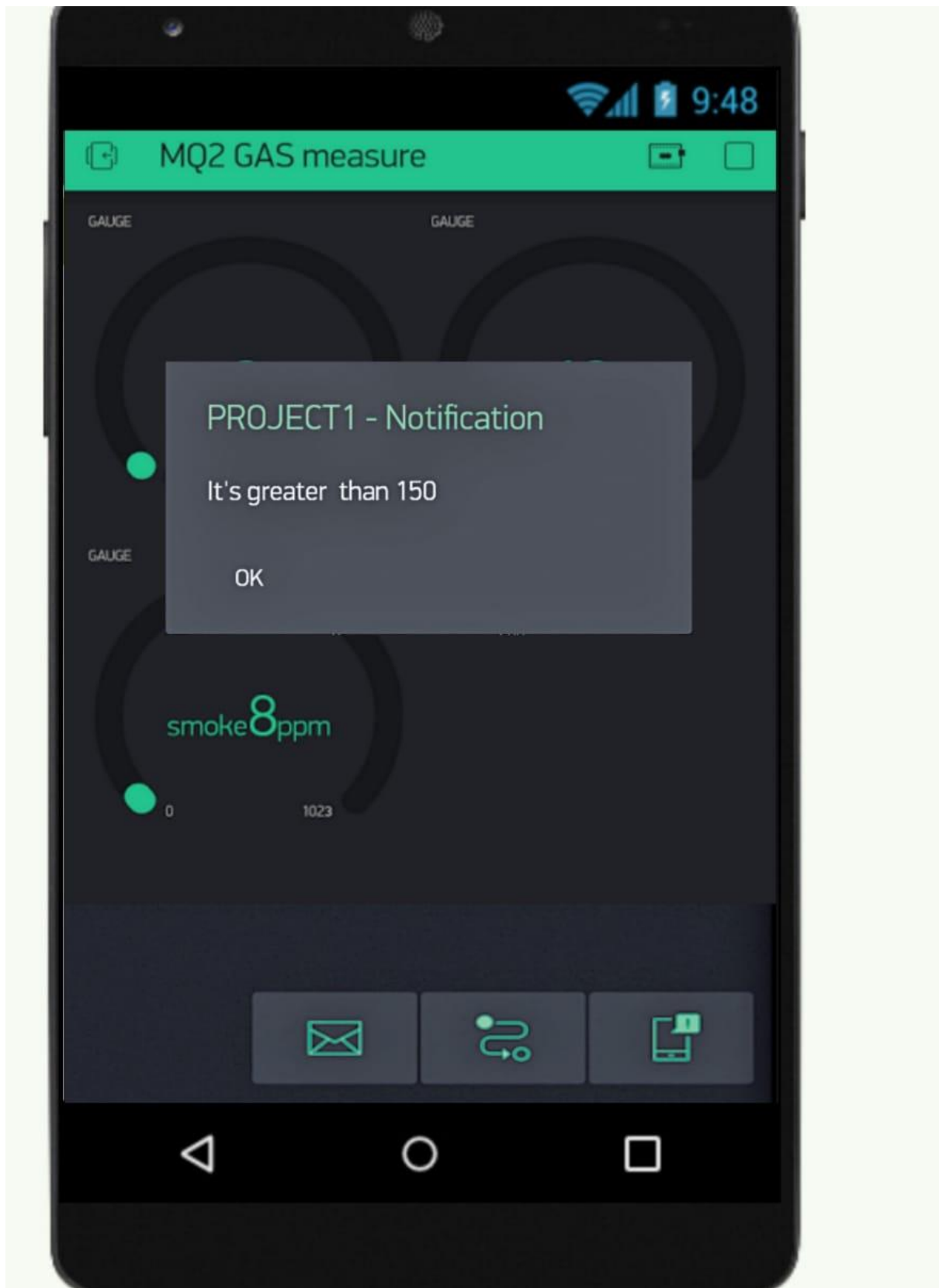## Data is brought to Node-RED
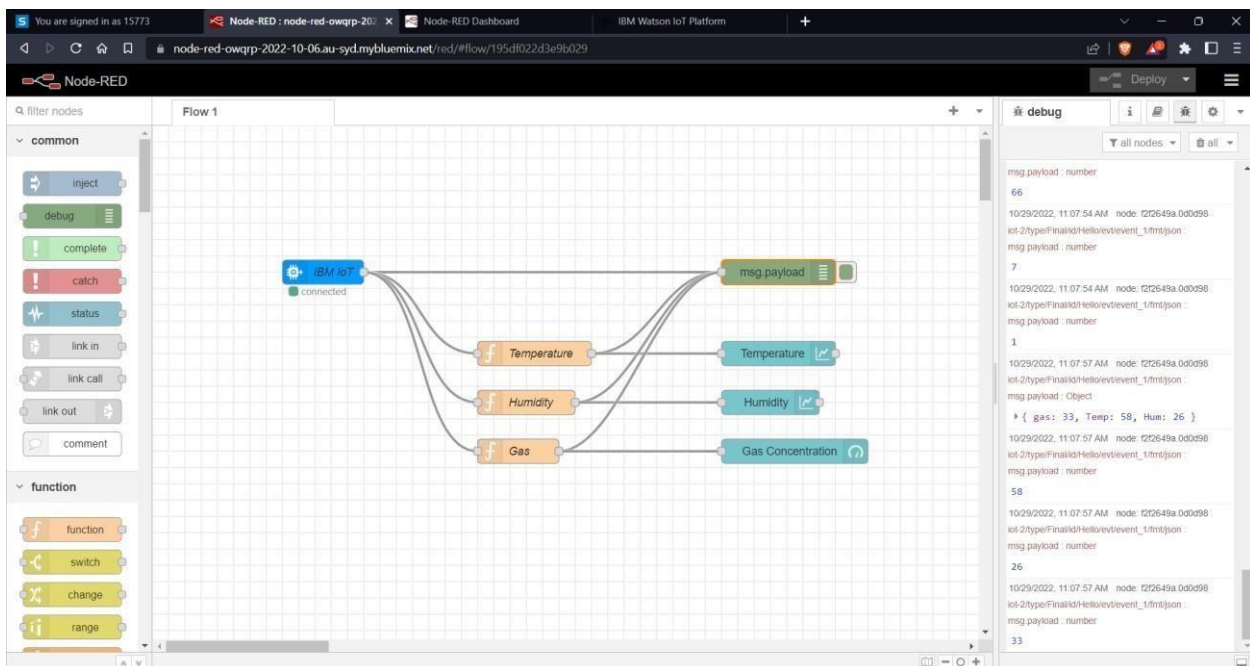
Data is displayed in Dashboard
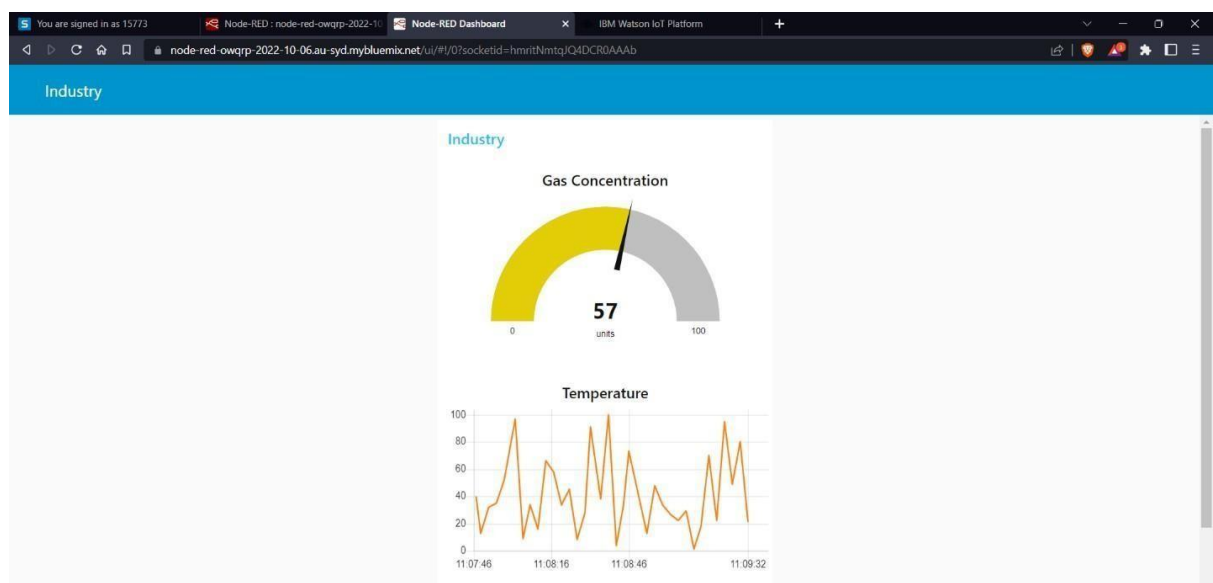
As an application, it should display the details of the gas level and other details to the user through the frontend of the mit app.