

Sprint 4

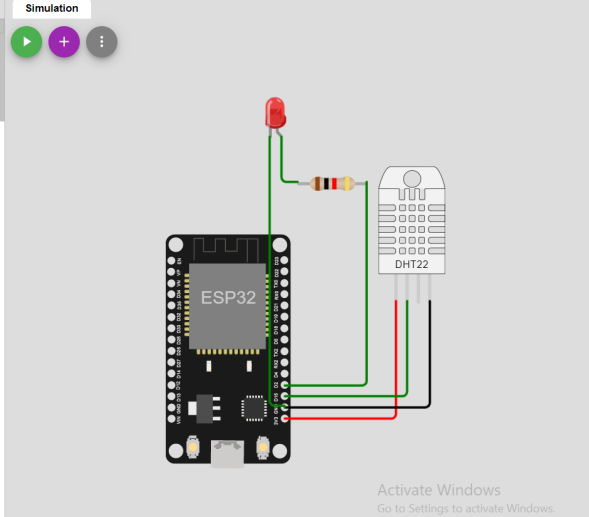
Wokwi Output (Connection of ESP32 with DHT11 and Random Generation Of Gas leakage level)

WOKWI SAVE SHARE Temperature ,Humidity and Gas Concentration detection Docs P

sketch.ino diagram.json libraries.txt Library Manager

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 #include "DHT.h"
4 #define DHTPIN 15
5 #define DHTTYPE DHT22
6 #define LED 2
7
8 DHT dht (DHTPIN, DHTTYPE);
9 void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
10
11
12 //-----credentials of IBM Accounts-----
13 #define ORG "z0kljz"//IBM ORGANITION ID
14 #define DEVICE_TYPE "DHT11"//Device type mentioned in ibm watson IOT Platform
15 #define DEVICE_ID "Temp_Humid"//Device ID mentioned in ibm watson IOT Platform
16 #define TOKEN "a8DT4SKUC+IC4kmC8@" //Token
17 String data3;
18 float h, t;
19 long gas_random;
20
21 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
22 char publishTopic[] = "iot-2/evt/Data/fmt/json";
23 char subscribetopic[] = "iot-2/cmd/test/fmt/String";
24 char authMethod[] = "use-token-auth";
25 char token[] = TOKEN;
26 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
27
28 WiFiClient wificlient;
29 PubSubClient client(server, 1883, callback ,wificlient);
30
31 void setup() {
32   Serial.begin(115200);
33   dht.begin();
34   pinMode(LED,OUTPUT);
35   delay(10);
```

Simulation



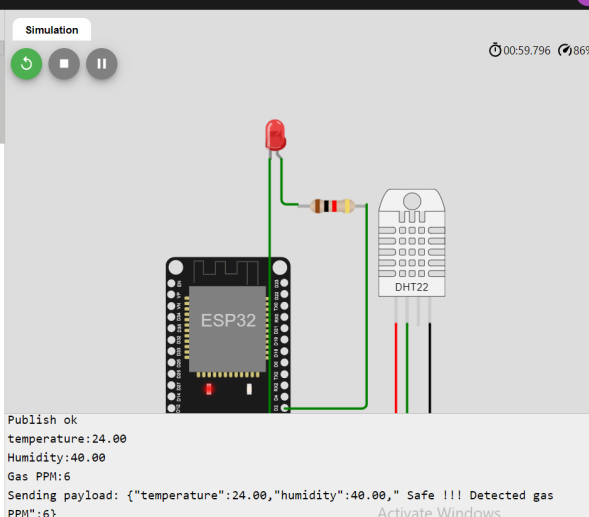
Activate Windows
Go to Settings to activate Windows.

WOKWI SAVE SHARE Temperature ,Humidity and Gas Concentration detection Docs P

sketch.ino diagram.json libraries.txt Library Manager

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 #include "DHT.h"
4 #define DHTPIN 15
5 #define DHTTYPE DHT22
6 #define LED 2
7
8 DHT dht (DHTPIN, DHTTYPE);
9 void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
10
11
12 //-----credentials of IBM Accounts-----
13 #define ORG "z0kljz"//IBM ORGANITION ID
14 #define DEVICE_TYPE "DHT11"//Device type mentioned in ibm watson IOT Platform
15 #define DEVICE_ID "Temp_Humid"//Device ID mentioned in ibm watson IOT Platform
16 #define TOKEN "a8DT4SKUC+IC4kmC8@" //Token
17 String data3;
18 float h, t;
19 long gas_random;
20
21 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
22 char publishTopic[] = "iot-2/evt/Data/fmt/json";
23 char subscribetopic[] = "iot-2/cmd/test/fmt/String";
24 char authMethod[] = "use-token-auth";
25 char token[] = TOKEN;
26 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
27
28 WiFiClient wificlient;
29 PubSubClient client(server, 1883, callback ,wificlient);
30
31 void setup() {
32   Serial.begin(115200);
33   dht.begin();
34   pinMode(LED,OUTPUT);
35   delay(10);
```

Simulation



00:59.796 86%

Publish ok
temperature:24.00
Humidity:40.00
Gas PPM:6
Sending payload: {"temperature":24.00,"humidity":40.00," Safe !!! Detected gas PPM":6}
Publish ok

Activate Windows
Go to Settings to activate Windows.

IBM Watson IOT Platform Output (Connection of Wokwi to IBM Cloud)

The screenshot displays the IBM Watson IoT Platform interface. A modal window titled "Event Payload" is open, showing the following details:

- Event Name:** Data
- Time Received:** 12 Nov 2022 20:19
- Payload:**

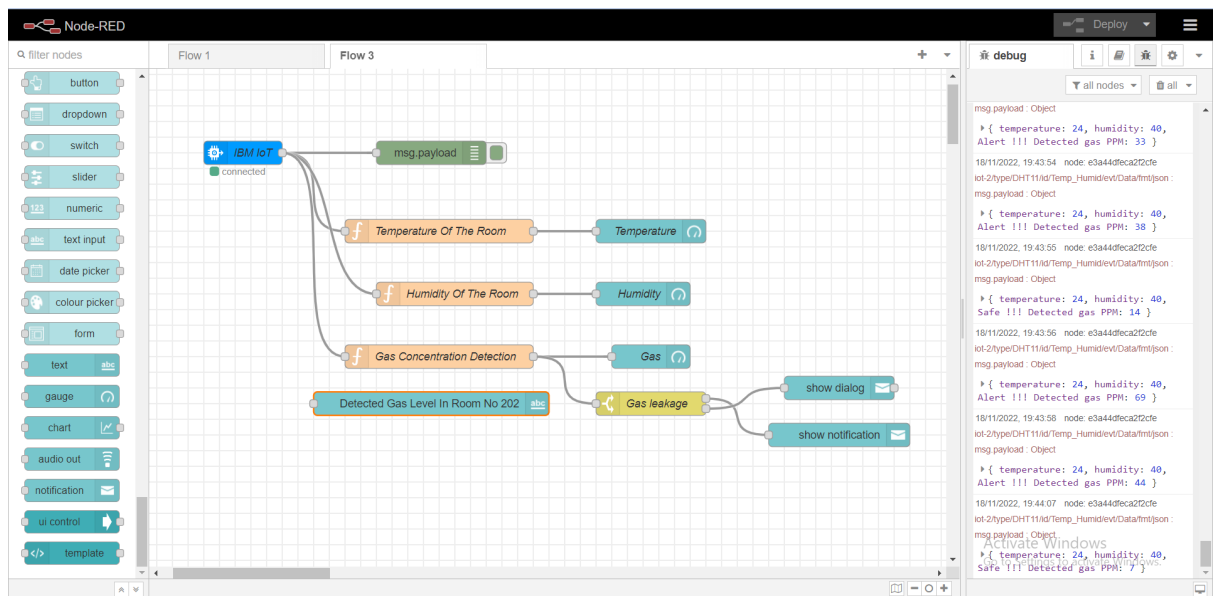
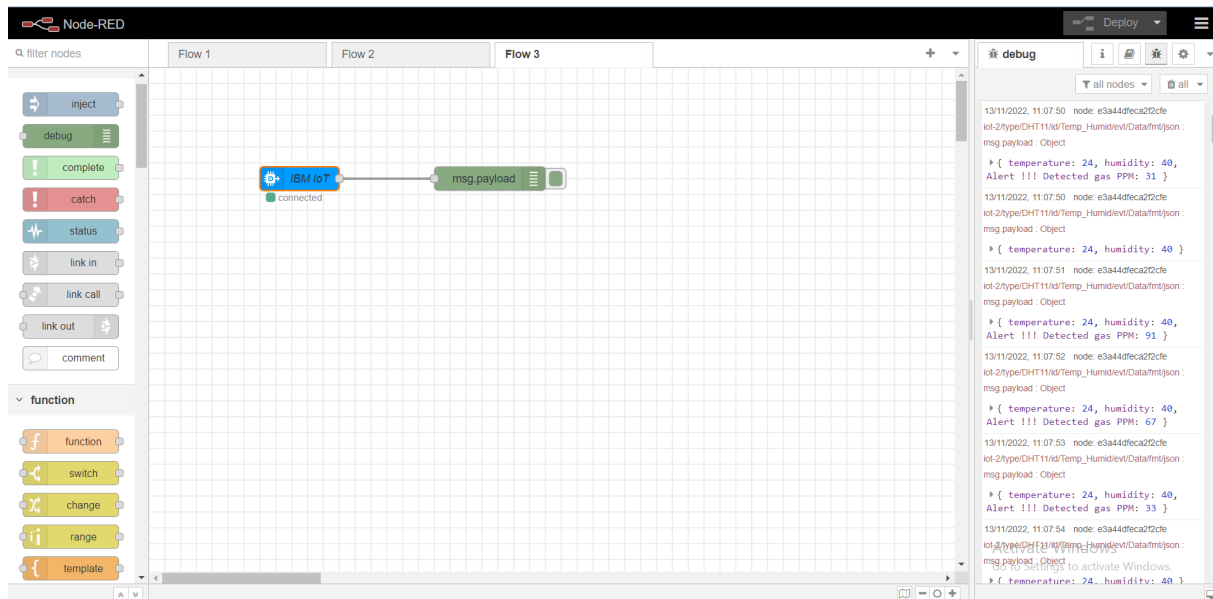
```
1 - {  
2   "temperature": 24,  
3   "humidity": 40,  
4   "Alert !!! Detected gas PPM": 88  
5 }
```

In the background, the "Temp_Humid" device page is visible, showing a table of recent events:

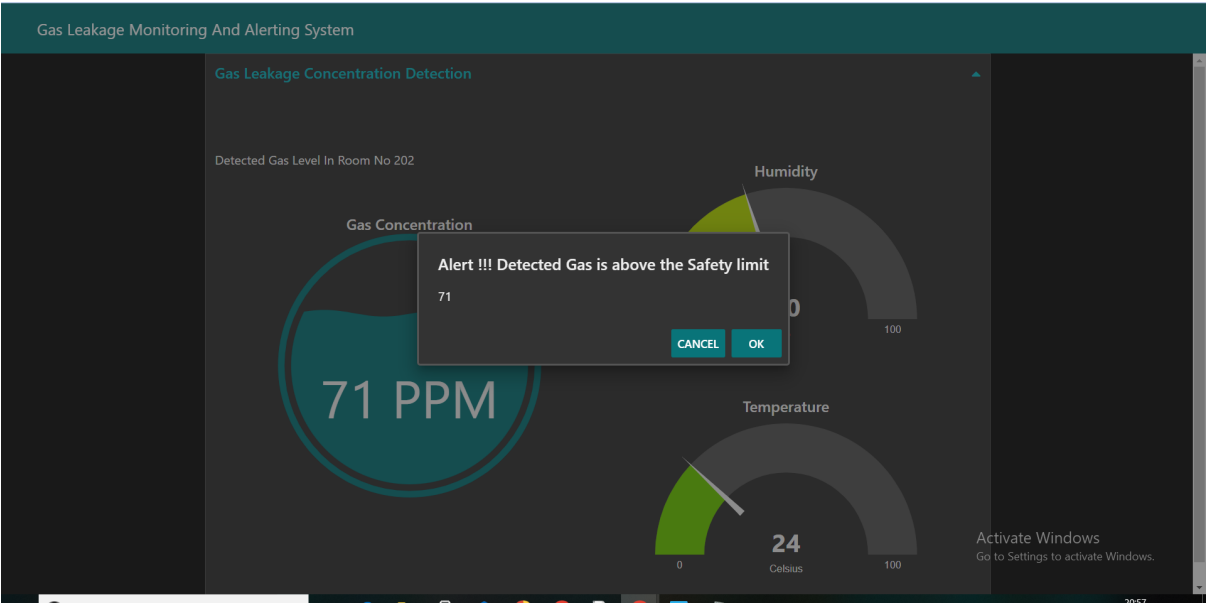
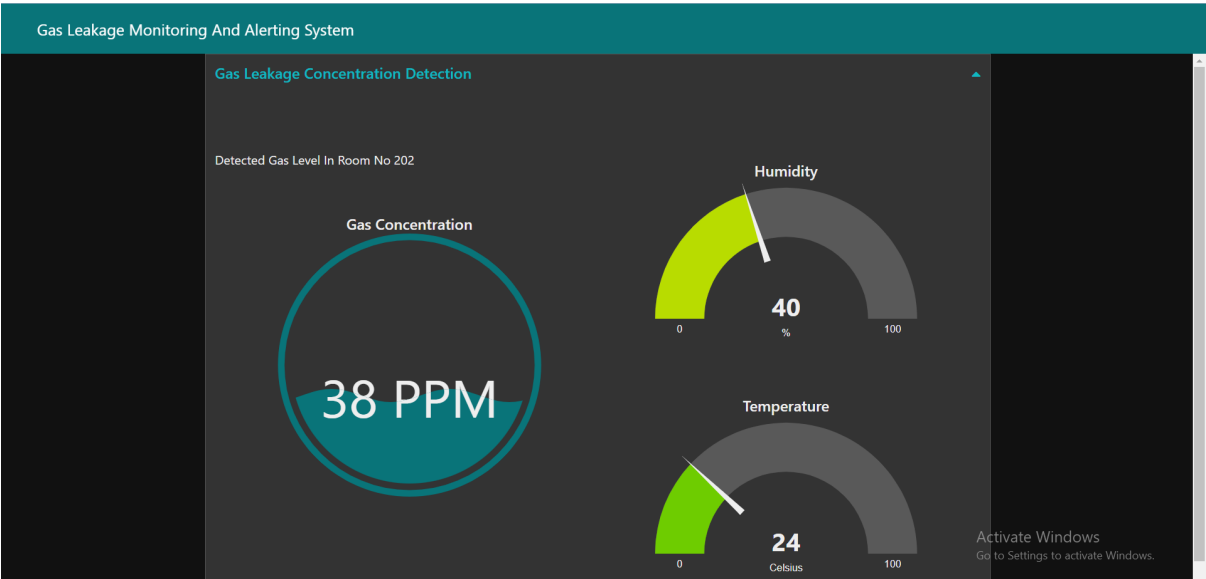
Event	Value
Data	{"temperature": 24, "humidity": 40, "Alert !!! Detected gas PPM": 88}
Data	{"temperature": 24, "humidity": 40, "Alert !!! Detected gas PPM": 88}
Data	{"temperature": 24, "humidity": 40, "Alert !!! Detected gas PPM": 88}
Data	{"temperature": 24, "humidity": 40, "Alert !!! Detected gas PPM": 88}
Data	{"temperature": 24, "humidity": 40, "Alert !!! Detected gas PPM": 88}

This screenshot is similar to the one above, showing the IBM Watson IoT Platform interface with the "Event Payload" modal open. The payload data is the same as in the first image. However, this version includes a Windows taskbar at the bottom of the screen, indicating it was captured on a Windows machine. The taskbar shows various application icons and the system clock displays the date as 2042-12-11-2022.

Node Red Output (Connection Of Node Red from IBM Cloud)



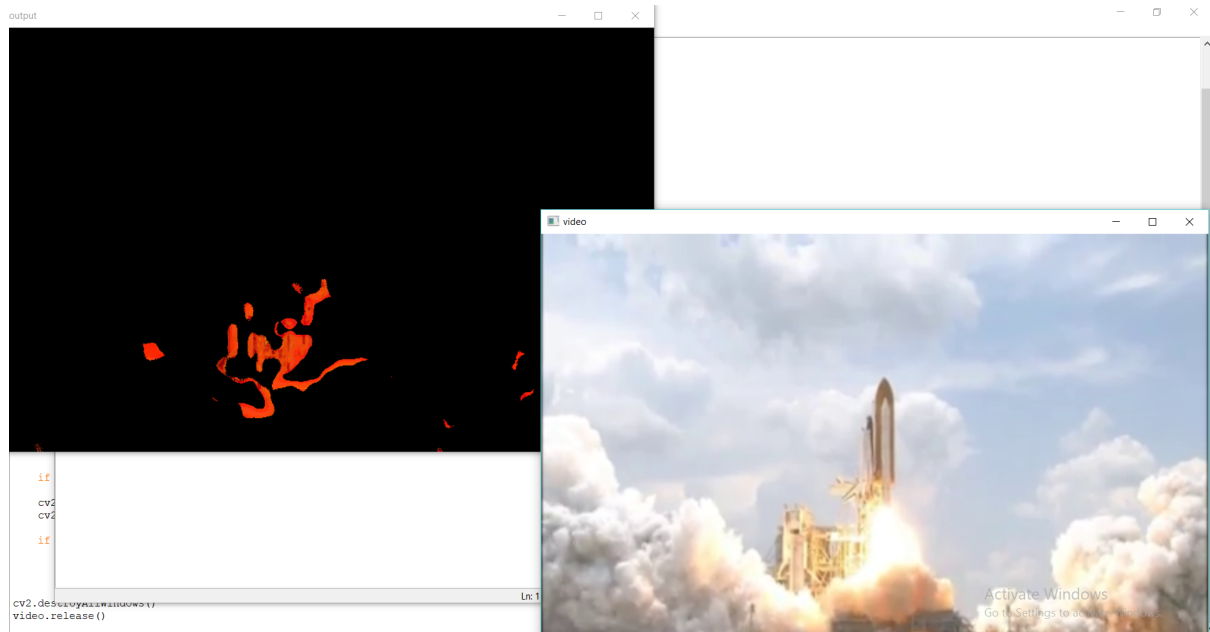
Webpage Output (By Using Node Red Dashboard)



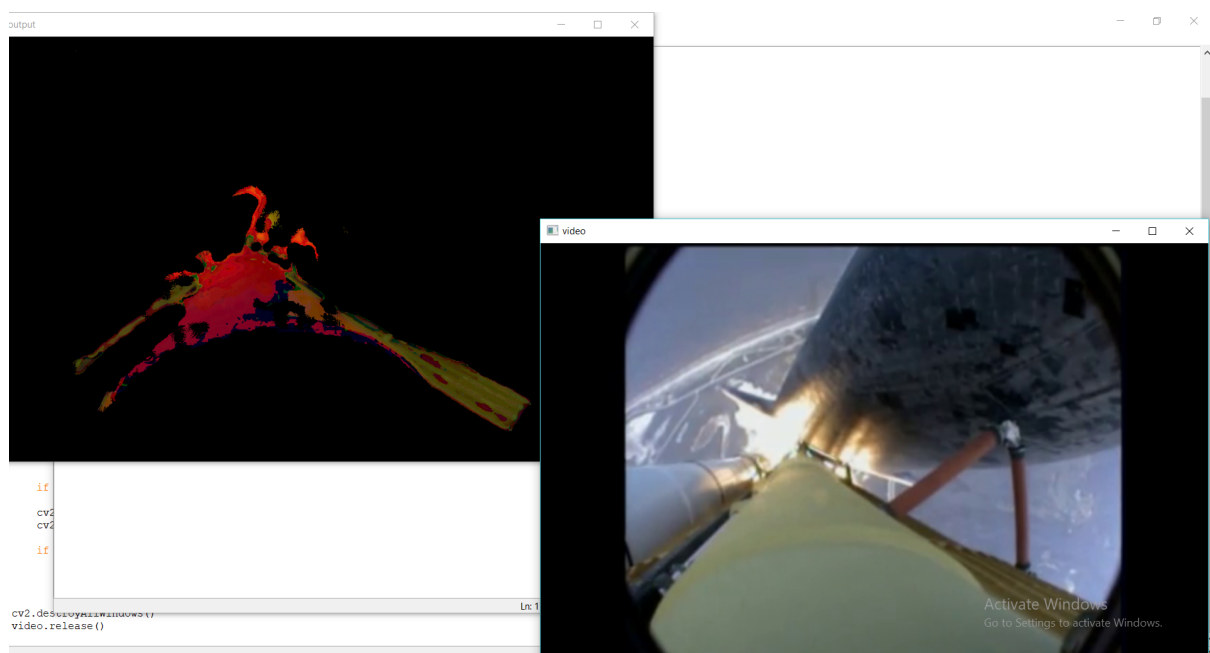
OpenCV Outputs

On right down , original input video and on the left top , OpenCV output video

Output 1: (Detection Of Fire In Video Input)



Output 2:



Output 3: (Detection Of Fire In Live Video)

