Gas Leakage Monitoring and Alerting System For Industries

Team ID	PNT2022TMID02525
Project Name	Gas Leakage Monitoring and Alerting System for Industries

1. INTRODUCTION

A gas detector is a device that detects the presence of gas in an area, often as part of a security system. This type of device is important because there are many gas that can be harmful to organic life forms such as humans and animals. Detects gas leaks and triggers warning systems to enable safety measures. Installing a gas leak detection system is a necessary investment, as some leaks are too small to smell or contain odorless gas. The use of gas in industrial and residential environments is constantly increasing. Gas is primarily used for power generation and as a process requirement in manufacturing industries. The consequences of a gas leak can be devastating regardless of the extent of the leak.

1.1 Project Overview

- Alarm when hazardous gas exceeds the threshold.
- Temperature, Humidity parameters are detected.
- Alert messages are shared through mobile phones using GSM
- The location of the gas leakage area is shared.
- Fire detection using OpenCV
- Rainwater sprinkling in case of catch fire.
- The Exhaust fan is ON when hazardous gas is detected.

1.2 Purpose

- To detect the leakage of hazardous gas in a closed environment, if any.
- To inform the user about the leakage of gas via SMS.
- To activate the alarm unit to inform neighbours about the gas leakage.
- To switch on the exhaust fan as a primary preventive measure against gas leakage.
- Produce a sound alarm upon gas leak and stop the alarm once gas leak is under control (gas presence in atmosphere is under normal range).

2. LITERATURE SURVEY

Gas Leakage Detector and Monitoring System by Yekini Nureni Adetokunbo Oloyede et al.,June 2022 in International Journal of Engineering and Manufacturing 12(4) DOI:10.5815/ijem.2022.05.05

Leakage of gas is a major issue in the industrial sector, gas-powered vehicles, and so on. The focus of this work is to propose a device that can detect gas leakage and alert the owners to avert problems due to gas leakages. The system is based on a microcontroller that employs a gas sensor as well as a GSM module, an LCD display, and a buzzer. The system was designed for gas leakage monitoring and alerts with SMS via an Arduino microcontroller with a buzzer and an MQ2 gas sensor. The circuit contains a Microcontroller MQ2 gas sensor, buzzer, LCD display, and GSM module, when the sensor detects gas leakage it transmit the information to the Microcontroller while the microcontroller makes a decision and then forwarded a warning message to the user as SMS to a mobile phone for decision to be taken accordingly. The output of this research will be significant in averting problems associated with gas leakages now and in future.

METHANE LEAKAGE MONITORING TECHNOLOGY FOR NATURAL GAS STATIONS AND ITS APPLICATION by Bing Han,Qiang Fu, et al., in IEEE 5th International Conference on Computer and Communications, DOI: 10.1109/ICCC47050.2019.9064041

The Fiber Bragg Grating sensing technology is applied to monitor key areas and equipment in natural gas stations that may suffer methane leakage, considering the shortcomings of existing leakage monitoring technologies applied for the natural gas stations, i.e. high false alarm rate, poor stability, easy to be interfered by background gas, etc. The false alarm rate can be effectively reduced by simultaneously monitoring the leakage vibration and methane concentration.

AUTOMATICS GAS DETECTION SYSTEM USING IOT by Rajat Kumar Dwibedi, Vanitha.v, et al., in International Conference on Recent Advancements in Engineering and Management, vol. 981

To detect the LPG, MQ-2 gas sensor is employed. The device would send an SMS to the appropriate authority via the GSM module, to investigate the leakage, alert the people in the house via Buzzer, and open windows automatically. The system consists of a sensor (MQ-2) that is highly sensitivity to propane (C3H8) and butane (C4H10), an Buzzer, a UNO Arduino (microcontroller), a GSM SIM 800C module, two channel relay module, 16x2 LCD display and a Servo motor. Also shuts down the supply in order to mitigate pollution, accident and leakage expenses.

2.1 Existing Problem

Gas leakage causes a variety of consequences, including financial loss and human life. Gas leakage can be discovered by neighboring humans but if no one is around, it cannot be found. But occasionally, a human with a poor sense of smell cannot notice it and as a result, this device will aid in the detection of gas leakage.Harmful gas leakage causes global warming and causes health problems.

2.2 References

[1] H. Bainand S. Zhu, H. Huang, "A Greenhouse Remote Monitoring System Based on GSM," in Proc. of IEEE International Conference on information management, pp. 357-360, 2011.

[2] S. Shinde, S. B. Patil and A. J. Patil, "Development of movable gas tanker leakage detection using wireless sensor IJSER International Journal of Scientific & Engineering Research Volume 9, Issue 7, July-2018 1851 ISSN 2229-5518 IJSER © 2018 http://www.ijser.org network based on embedded system," International Journal of Engineering Research and Applications.

[3] H.G. Rodney Tan, C. H. Lee and V. H. Mok, "Automatic Gas Leakage Detection and Control Using GSM Network" IPEC 2007, International power engineering conference, PP. 465 – 469, Dec 3-6, 2007

[4] Li Kaicheng, Liu Jianfeng, Yue Cong Yuan, Zhang Ming, "Gas Leakage System Based on ARM microprocessor" CPEM 2008, Conference on Precision Electromagnetic Measurements digest, PP. 216 –217, June 8-13, 2008.

[5] V. Ramya and B. Palaniappan, "Embedded system for Hazardous gas detection and Alerting," in Proc. of International Journal of Distributed and parallel system(IJDPS), vol. 3, no. 3, May 2012.

[6] Yu - Wei Huang, Shun - Chien Chang and Chih - Hung Wu, "GPRS - Based Embedded Gas Sensing System" Sensors and Industry Conference 2005, PP. 105 – 110, Feb 8 - 10, 2005.

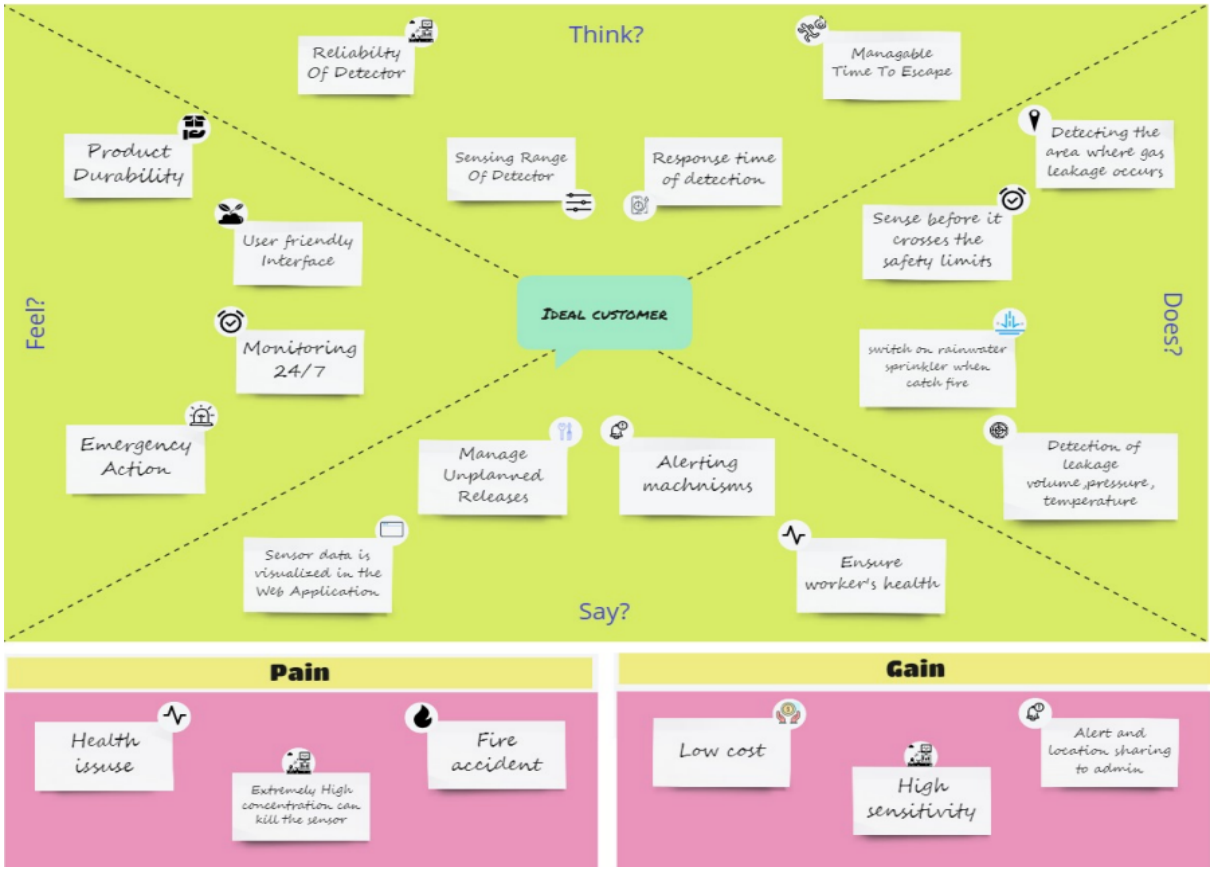
2.3 Problem Statement Definition

Gas leaks have many consequences, including economic loss and human life. Gas leaks can be detected by others, but they cannot be detected without them. However, people with a weak sense of smell may not notice it, so it can help detect gas leaks. Leakage of toxic gases causes global warming and health problems.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

The empathy map canvas is below.



3.2 Ideation & Brainstorming

1. Alarm when hazardous gas exceeds the threshold.
2. Alert messages are shared through the website.
3. The location of the gas leakage area is shared.
4. Alert messages are shared through mobile phones using GSM.
5. After the spread of hazardous gas, windows automatically open by IOT.
6. Rain sprinkling in case of catch fire.
7. Database of previous gas leakage using cloud.
8. Real time parameters of gas are displayed.
9. Additional inverter LED bulbs are fixed in the rooms incase of power shutdown due to gas leakage and for fixing of leaked valves.
10. Additional valve is attached to the opening of the gas cylinder, if in case the gas is leaked, the valve will automatically close.

3.3 Proposed Solution

Following table is the proposed solutions.

S.NO	Parameter	Description
1	Problem Statement (Problem to be solved)	Gas leakage causes a variety of consequences, including financial loss and human life. Gas leakage can be discovered by neighboring humans but if no one is around, it cannot be found. But occasionally, a human with a poor sense of smell cannot notice it and as a result, this device will aid in the detection of gas leakage. Harmful gas leakage causes global warming and causes health problems.
2	Idea / Solution description	To build a prototype that can detect the gas leakage and measure the humidity and temperature which will be uploaded on the IBM cloud for cloud computation and to determine the level of gas leakage and alert the concerned person depending upon the gas leakage via Node red Dashboard.
3	Novelty / Uniqueness	System provides real time alerts about the presence in the atmosphere and real-time updates about leakages with the location.
4	Social Impact / Customer Satisfaction	The environment has a critical impact on human health issues in everyday life. Fire, Suffocation and explosion risks are all determined by physical attributes like flammability and toxicity. As the leaked gas is detected and alert is given prior to the safety limit, it provides enough time to evacuate workers from the particular area. The rate of risk to workers is lowered.
5	Business Model (Revenue Model)	Receive instant notifications of the presence of gases in the atmosphere. Prevent explosions and fire dangers. Monitor the levels of gas concentration. Maintain the health of the workforce. Installation is affordable.
6	Scalability of the Solution	The system can also be used in Mining industries, Semiconductor manufacturing industries, Wastewater treatment, Chemical industry and Oil and gas sector.

3.4 Problem Solution Fit

Following table is the problem solution fit.

<table>
<tr><td>1. CUSTOMER SEGMENT(S)

Gas & Oil sector, Chemical Industries, Wastewater treatment, Semiconductor manufacturing, Mining Industries.</td><td>6. CUSTOMER LIMITATIONS

No real time updates and alerts about leakage with location, only one gas can be measured with each instrument.</td><td>5. AVAILABLE SOLUTION(S)

Provides alert by means of Alarm.</td></tr>
<tr><td>2. PROBLEMS / PAINS

Health issues like breathing difficulties, Fire accident, Financial loss.</td><td>9. PROBLEM ROOT / CAUSE

Faulty pipeline, Poor ventilation, Fuel Pressure regulator issues, Fitting and connection issues.</td><td>7. BEHAVIOR

Less response time, only one particular gas can be detected.</td></tr>
<tr><td>3.TRIGGERS TO ACT

Monitoring 24/7 , High sensitivity, Emergency action, User friendly interface</td><td rowspan="2">10. SOLUTION

Provides real time updates and alerts about leakage with location to the website. Detection of temperature and humidity. Emergency actions like water sprinkler is used.</td><td rowspan="2">8. CHANNELS OF BEHAVIOR

Social media platforms like YouTube, Instagram and Facebook.</td></tr>
<tr><td>4. EMOTIONS

Before - Risk of financial loss and human life.
After - No risk of financial loss and human life.</td></tr>
</table>

4. REQUIREMENT ANALYSIS

4.1 Functional Requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirements (Epic)	Sub Requirements (Story/ Sub-Task)
FR-1	User Entry	Scan the QR code
FR-2	User Access	Real time monitoring of gas leakage systems through web portals for authorized users.
FR-3	User Alert	User receives an alert through the website.
FR-4	Review and Feedback	Suggestion boards that allow customers to submit Feedback.
FR-5	Gas Sensor	Gas sensors are used to detect and identify different types of gas leaked.
FR-6	Temperature Sensor	Temperature sensor is used to measure the temperature of its environment and converts the input data into electronic data
FR-7	Open CV	OpenCV technology accesses a camera to detect the fire (may occur due to external entities like rise of temperature) and interface with an arduino to control the water sprinkler.
FR-8	Relay	Relays whose control is taken by arduino to switch ON or OFF the water sprinkler motor.
FR-9	Water Pump Motor	Water sprinkler system that discharges water when the fire has been detected, such as when a predetermined temperature has been exceeded.
FR-10	Arduino	Open-source electronic prototyping platform interfaces with sensors, OpenCV and delivers desired alerts, data , monitoring of unsafe places and takes control measures.

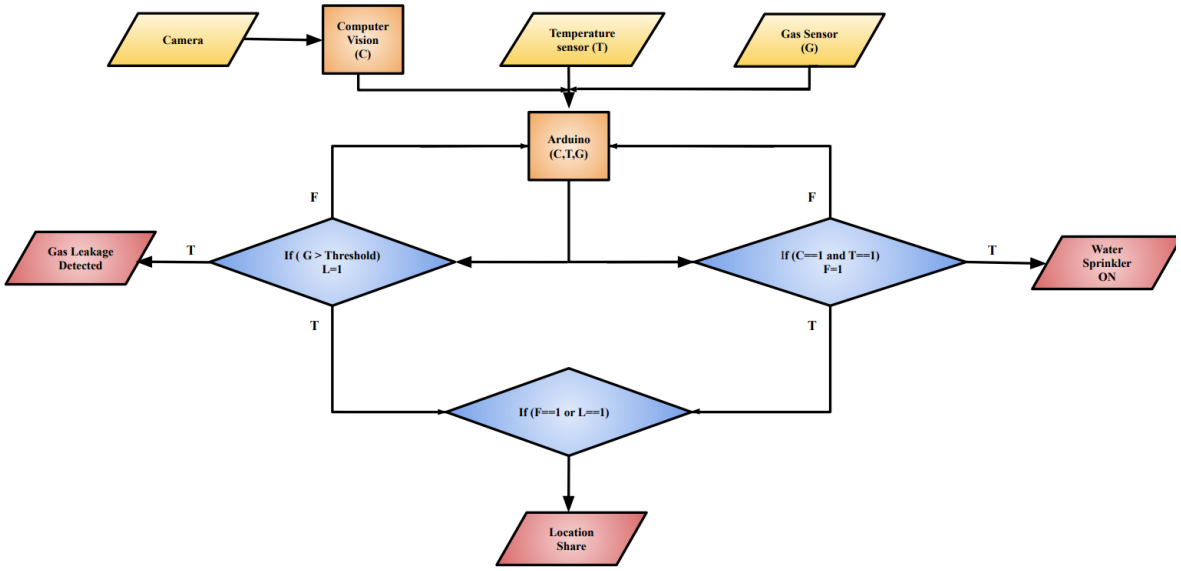
4.2 Non-Functional Requirements

Following are the non-functional requirements of the proposed solution.

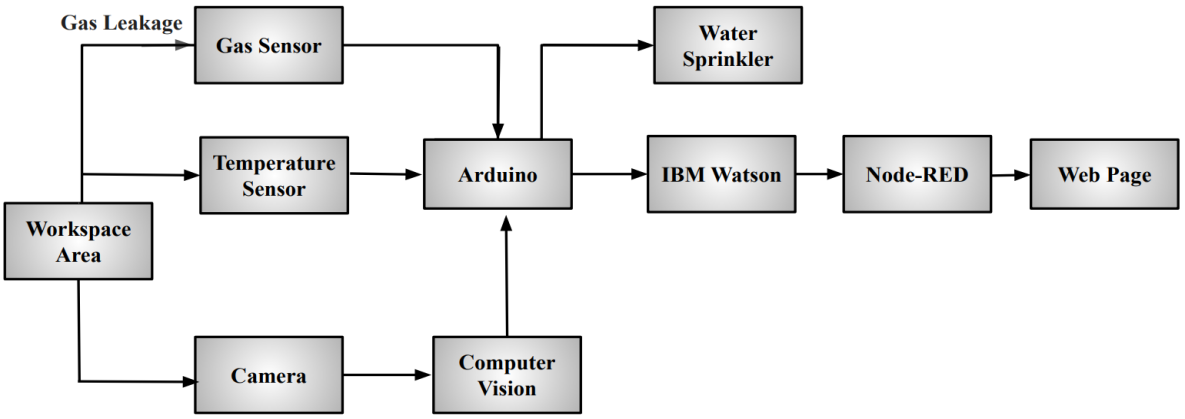
NFR No.	Non Functional Requirements	Description
NFR-1	Usability	Usage of UI template of Node - Red could make well understandable representation of data like usage of gauge representation, Graphical representation, Sliders etc
NFR-2	Security	Only authorized persons have access to the system and also data transmission and handling through secured protocols
NFR-3	Reliability	Gas Sensor has an anti-explosion network which ensures the heating element does not cause explosion on interfacing with flammable gas.
NFR-4	Performance	Faster response and high accuracy of gas leakage detection in localized areas
NFR-5	Availability	Real time monitoring system and also the user can access the website 24/7
NFR-6	Scalability	The system is scalable even in case of many gas sensors or in case of many supervisors.

5. PROJECT DESIGN

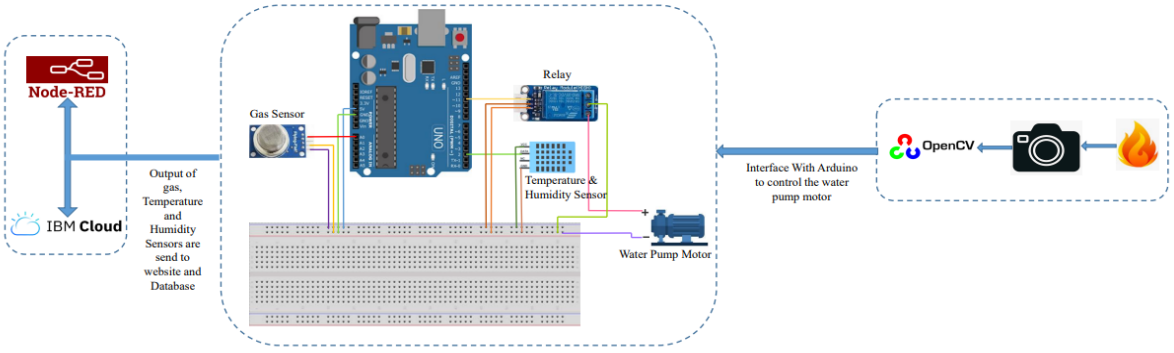
5.1 Data Flow Diagram



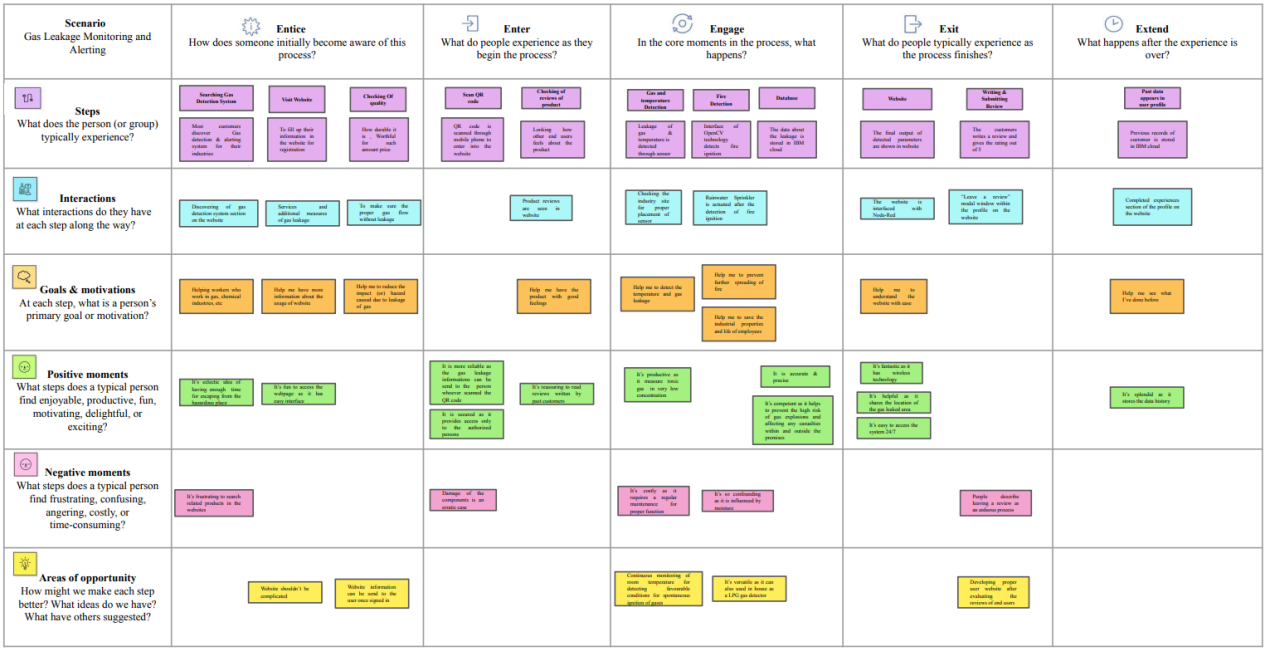
5.2 Block Diagram



5.3 Solution and Technical Architecture



5.3 User Stories



6. PROJECT PLANNING AND SCHEDULING

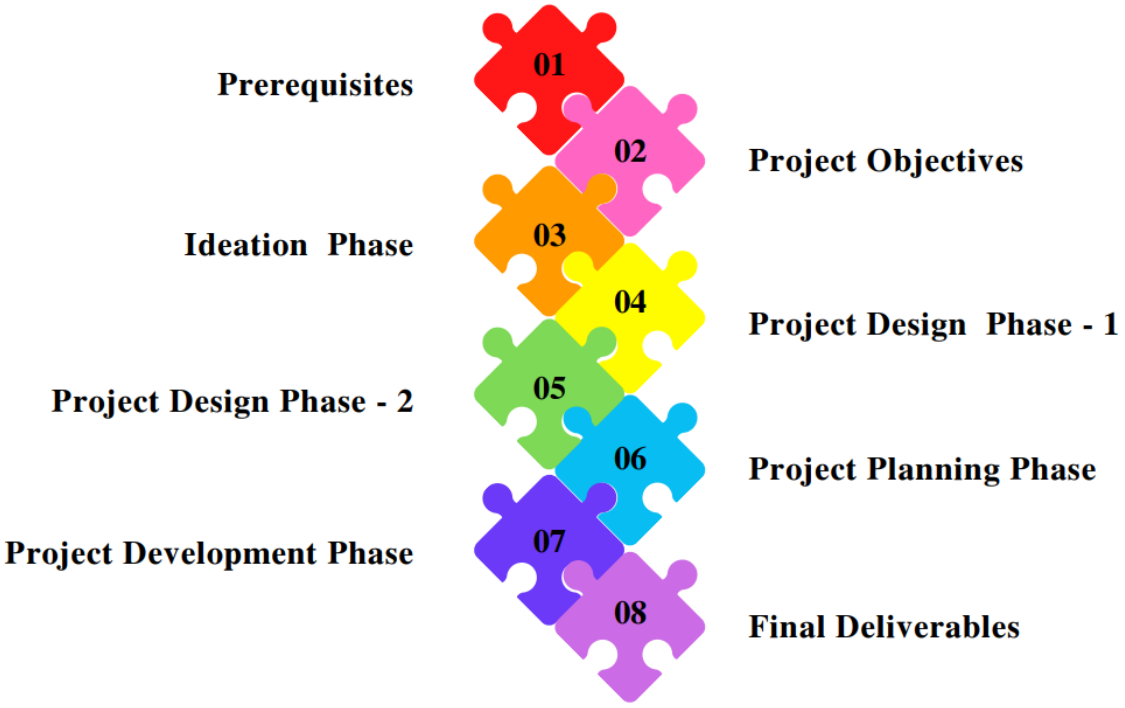
6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Objective	USN-1	As a system, the access is done through QR code	2	High	Sandhiya V, Sarumathi M, Priyadharshini V, Pavithra P
Sprint-1	Features	USN-2	As a system, the gas sensor should detect the gas	2	Low	Sandhiya V
Sprint-1	Features	USN-3	As a system, the fire should be detected using open CV	1	Medium	Priyadharshini V
Sprint-1	Features	USN-4	As a system, after the detection of Fire the rainwater sprinkler should actuated	2	High	Sarumathi M, Pavithra P
Sprint-2	Focus	USN-5	As a system, as soon as the detected gas reaches the threshold level, the alert should be turned ON	1	Low	Sandhiya V, Sarumathi M
Sprint-2	Focus	USN-6	As a system, as soon as the detected gas reaches the threshold level, the alert should send to admin via website	2	Medium	Priyadharshini V, Pavithra P

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-2	Features	USN-7	As a system, it should the send the location where the gas is detected	2	High	Sandhiya V
Sprint-3	Features	USN-8	As a program, it should retrieve the API key of the IBM cloud to send the details of the system.	1	Low	Priyadharshini V
Sprint-3	Data Transfer	USN-9	As a cloud system, the IBM cloud should send the data to NodeRed	2	High	Sarumathi M
Sprint-3	Data Transfer	USN-10	As a system, it should collect the data from the NodeRed and give it to the backend of the website	2	High	Pavithra P
Sprint-3	Data Transfer	USN-11	As an application, it should display the details of the gas level and other details to the user through the frontend of the website.	2	High	Sandhiya V
Sprint-4	Data Transfer	USN-12	As a user, I can access the dashboard andmake use of available resources.	2	Medium	Priyadharshini V
Sprint-4	Data Transfer	USN-13	As a user, I must receive an alert once the leakage is detected.	1	High	Pavithra P
Sprint-4	Registration	USN-14	As an admin, I must receive information about the leakage along with location and share exact location and route to the person.	2	High	Sarumathi M

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-4	Registration	USN-15	As an admin, I must allot particular person to look after the leakage in a particular location.	2	Medium	Sandhiya V, Sarumathi M, Priyadharshini V, Pavithra P

6.2 Sprint Delivery Schedule



7. CODING & SOLUTION

7.1 Feature 1

Detection of Gas Leakage above threshold, Temperature, Humidity, Alert when gas leaked with location.

CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTPIN 15
#define DHTTYPE DHT22
#define LED 2

DHT dht (DHTPIN, DHTTYPE);
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

//-------credentials of IBM Accounts------
#define ORG "z0kljz"//IBM ORGANITION ID
#define DEVICE_TYPE "DHT11"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "Temp_Humid"//Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "a8DT4SKUC+IC4kmC8@" //Token
String data3;
float h, t;
long gas_random;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/test/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);

void setup() {
```

```
Serial.begin(115200);
dht.begin();
pinMode(LED,OUTPUT);
delay(10);
Serial.println();
wificonnect();
mqttconnect();
}

void loop()
{
h = dht.readHumidity();
t = dht.readTemperature();
gas_random = random(0,100);
Serial.print("temperature:");
Serial.println(t);
Serial.print("Humidity:");
Serial.println(h);
Serial.print("Gas PPM:");
Serial.println(gas_random);


PublishData(t, h, gas_random);
delay(1000);
if (!client.loop()) {
mqttconnect();
 }
}

void PublishData(float temp, float humid, long gas) {
mqttconnect();
String payload = "{\"temperature\":";
payload += temp;
payload += "," "\"humidity\":";
payload += humid;

if (gas>=30){
  payload += "," "\"Alert !!! Detected gas PPM\":";
  payload += gas;
```

```
  digitalWrite(LED,HIGH);
}
else {
  payload += ","\" Safe !!! Detected gas PPM\":";
  payload += gas;
  digitalWrite(LED,LOW);
}

payload += "}";

Serial.print("Sending payload: ");
Serial.println(payload);

if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Publish ok");
} else {
Serial.println("Publish failed");
 }
}
void mqttconnect() {
if (!client.connected()) {
Serial.print("Reconnecting client to ");
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);
if(data3=="lighton")
{
  Serial.println(data3);
  digitalWrite(LED,HIGH);
}
else{
  Serial.println(data3);
  digitalWrite(LED,LOW);
}
data3="";
}
```

7.2 Feature 2

Detection of fire accidents using OpenCV in python.

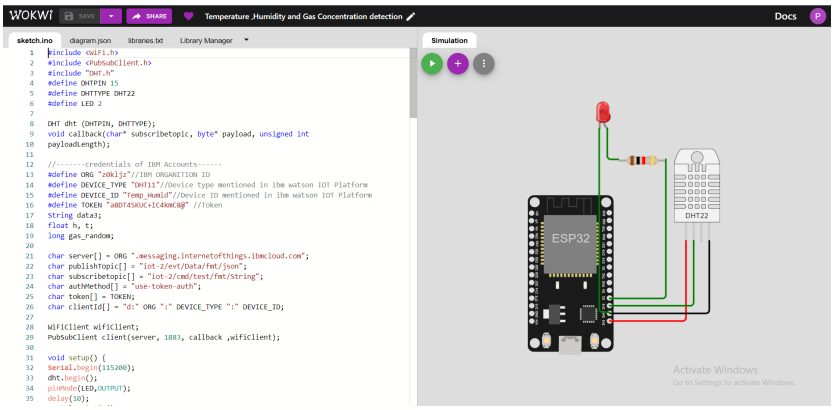
CODE:

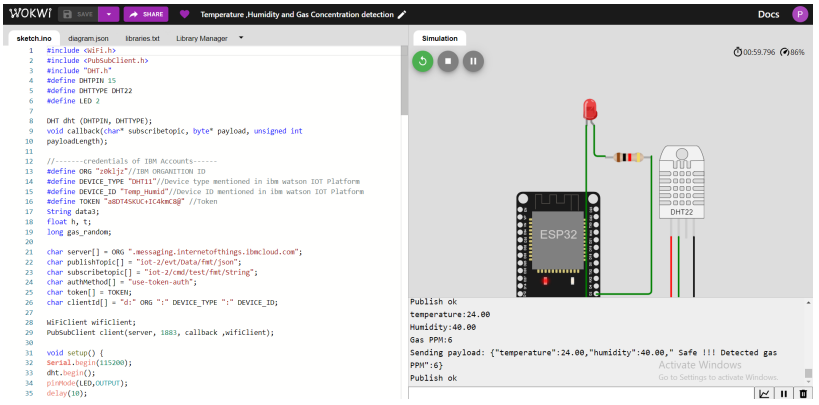
```
import cv2
import time
import sys
import smtplib
import numpy as np
import threading
Alarm_Status = False
Email_Status = False
Fire_Reported = 0

video = cv2.VideoCapture(0) # If you want to use webcam use Index like 0,1

while True:
    (grabbed, frame) = video.read()
    if not grabbed:
        break

    frame = cv2.resize(frame, (850, 540))

    blur = cv2.GaussianBlur(frame, (21, 21), 0)
    hsv = cv2.cvtColor(blur, cv2.COLOR_BGR2HSV)

    lower = [18, 50, 50]
    upper = [35, 255, 255]
    lower = np.array(lower, dtype="uint8")
    upper = np.array(upper, dtype="uint8")

    mask = cv2.inRange(hsv, lower, upper)
    output = cv2.bitwise_and(frame, hsv, mask=mask)

    no_red = cv2.countNonZero(mask)

    if int(no_red) > 15000:
        Fire_Reported = Fire_Reported + 1
    cv2.imshow("output",output)
    cv2.imshow("video",frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cv2.destroyAllWindows()
video.release()
```

8. TESTING

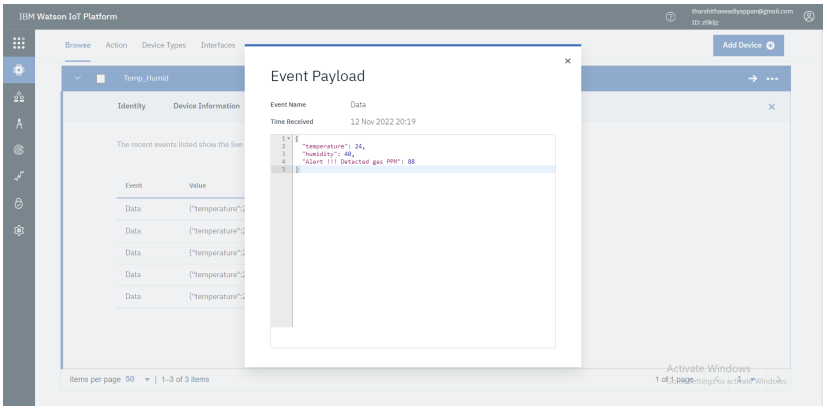
8.1 Test Cases

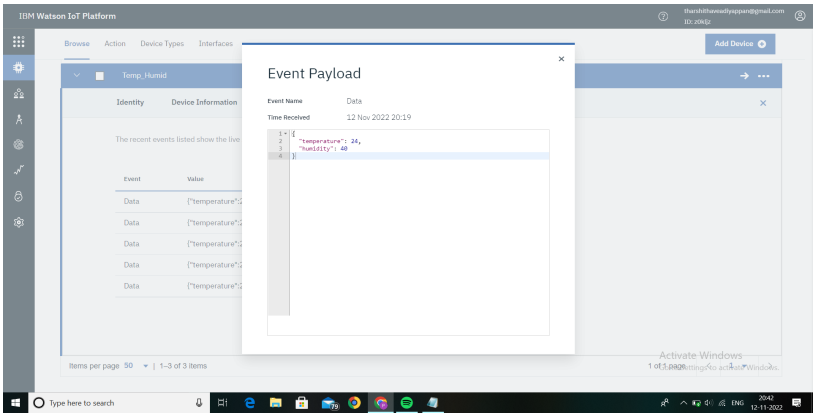
Wokwi Output (Connection of ESP32 with DHT11 and Random Generation Of Gas leakage level)



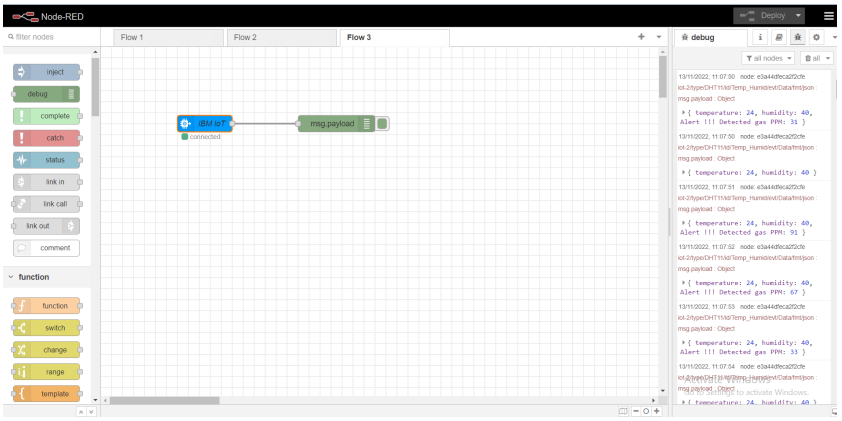


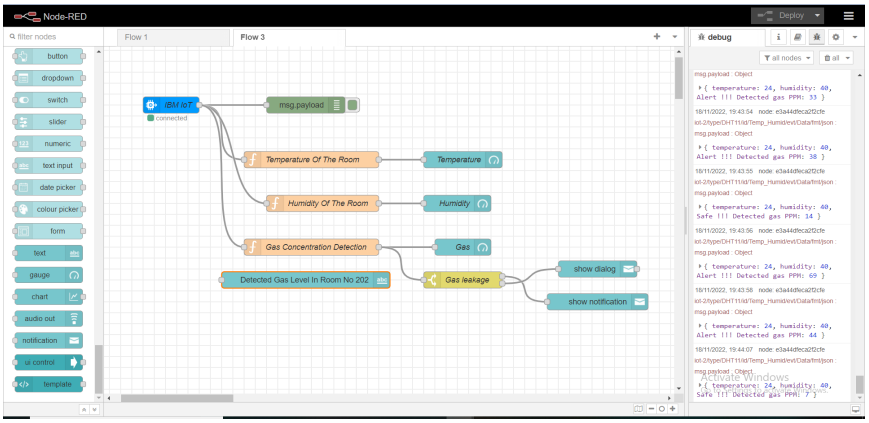
IBM Watson IOT Platform Output (Connection of Wokwi to IBM Cloud)



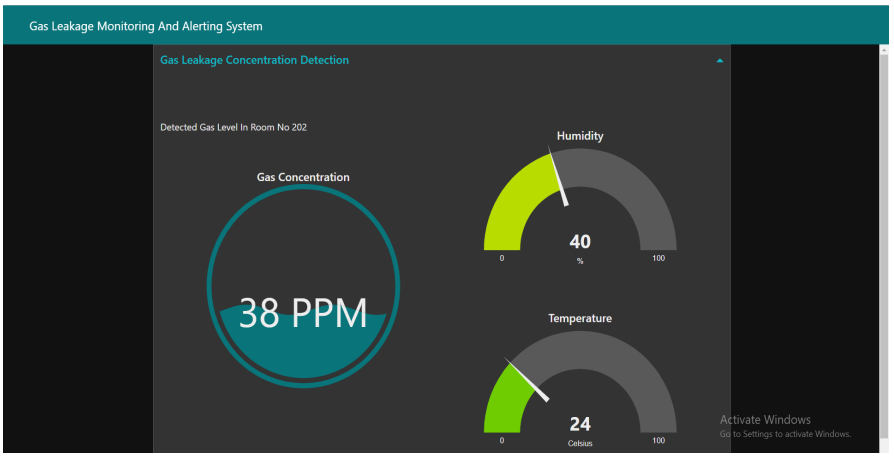


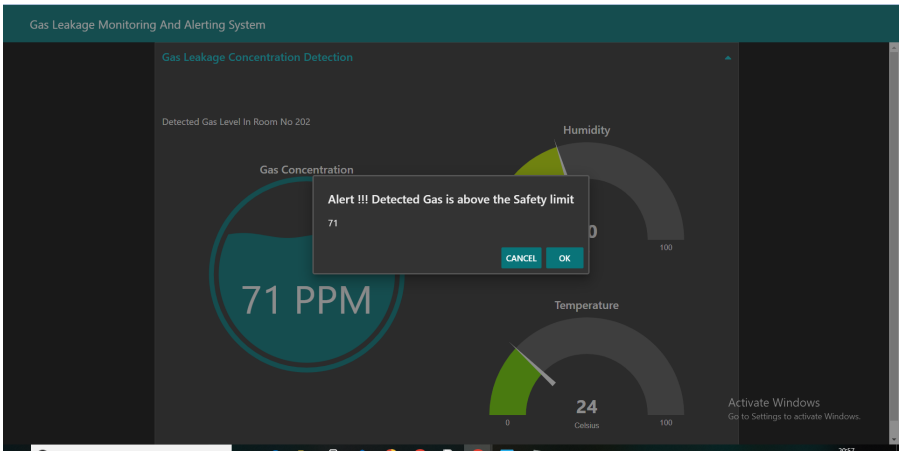
Node Red Output (Connection Of Node Red from IBM Cloud)





Webpage Output (By Using Node Red Dashboard)

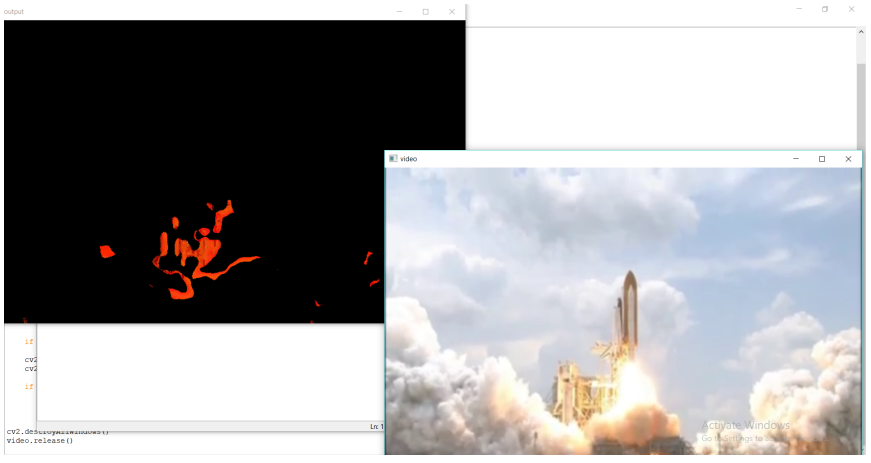


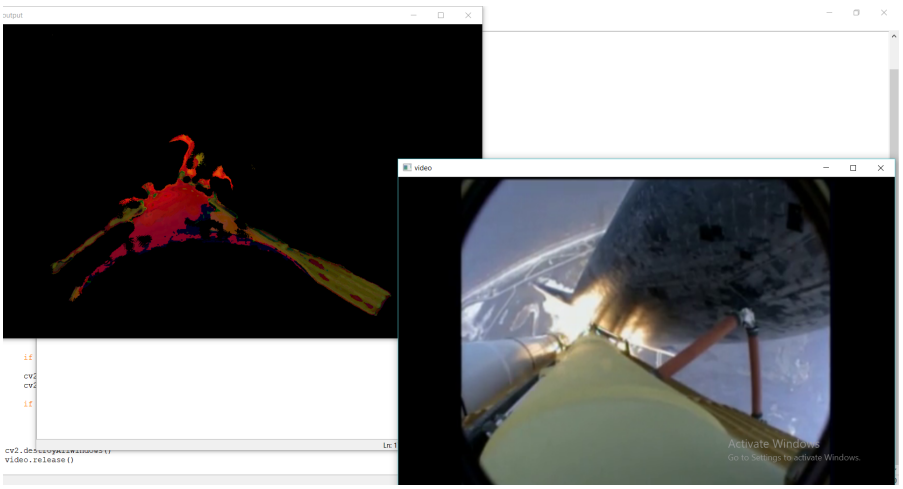


OpenCV Outputs

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

Output 1: (Detection Of Fire In Video Input)





8.2 User Acceptance Testing

Output : (Detection Of Fire In Live Video)

9. ADVANTAGES & DISADVANTAGES

Advantages:

- Low power consumption and reliability.
- Automatically controlled and easy to use.
- It is accurate and precise as it is digital.
- Measure toxic gas in low concentration.

Disadvantages:

- Equipment requires regular maintenance by a qualified person.
- Damage of the component in an erratic case.

10. CONCLUSION

Gas leakage leads to severe accidents resulting in material losses and human injuries. The main causes of gas leaks are poor maintenance of facilities and lack of awareness of people. Thus the system provides safety and control measures helping the humans.

11. FUTURE SCOPE

- The system can be integrated with GSM to provide alerts through SMS.
- Electricity can be shut down when hazardous gas is detected.
- Additional inverter LED bulbs can be fixed in the rooms incase of power shutdown due to gas leakage and for fixing of leaked valves.
- The Exhaust fan can be turned ON when hazardous gas is detected.
- Additional valve can be attached to the opening of the gas cylinder, if in case the gas is leaked, the valve will automatically close.

12. APPENDIX

SOURCE CODE

Wokwi Stimulator Connection to IBM Cloud Link & Code:

Wokwi Link : https://wokwi.com/projects/348132773889835602

Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTPIN 15
#define DHTTYPE DHT22
#define LED 2

DHT dht (DHTPIN, DHTTYPE);
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

//-------credentials of IBM Accounts------
#define ORG "z0kljz"//IBM ORGANITION ID
#define DEVICE_TYPE "DHT11"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "Temp_Humid"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "a8DT4SKUC+IC4kmC8@" //Token
String data3;
float h, t;
long gas_random;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/test/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);

void setup() {
        Serial.begin(115200);
```

```
        dht.begin();
        pinMode(LED,OUTPUT);
        delay(10);
        Serial.println();
        wificonnect();
        mqttconnect();
}

void loop()
{
        h = dht.readHumidity();
        t = dht.readTemperature();
        gas_random = random(0,100);
        Serial.print("temperature:");
        Serial.println(t);
        Serial.print("Humidity:");
        Serial.println(h);
        Serial.print("Gas PPM:");
        Serial.println(gas_random);
        PublishData(t, h, gas_random);
        delay(1000);
        if (!client.loop()) {
                mqttconnect();
         }
}

void PublishData(float temp, float humid, long gas) {
        mqttconnect();
        String payload = "{\"temperature\":";
        payload += temp;
        payload += "," "\"humidity\":";
        payload += humid;

        if (gas>=30){
                payload += "," "\"Alert !!! Detected gas PPM\":";
                payload += gas;
                digitalWrite(LED,HIGH);
        }

        else {
                payload += ","""\" Safe !!! Detected gas PPM\":";
                payload += gas;
                digitalWrite(LED,LOW);
```

```
        }

        payload += "}";

        Serial.print("Sending payload: ");
        Serial.println(payload);

        if (client.publish(publishTopic, (char*) payload.c_str())) {
                Serial.println("Publish ok");
        }
         else {
                Serial.println("Publish failed");
         }
}
void mqttconnect() {
        if (!client.connected()) {
                Serial.print("Reconnecting client to ");
                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);
        if(data3=="lighton")
        {
                Serial.println(data3);
                digitalWrite(LED,HIGH);
        }
        else{
                Serial.println(data3);
                digitalWrite(LED,LOW);
        }
        data3="";
}
```

OpenCV Code:

```
import time
import sys
import cv2
import numpy as np
import smtplib
import threading

Fire_Reported = 0

video = cv2.VideoCapture("Rocket Launch - 228.mp4") # If you want to use a webcam use
Index like 0,1.

while True:
   (grabbed, frame) = video.read()
   if not grabbed:
      break
```

```
        frame = cv2.resize(frame, (850, 540))

        blur = cv2.GaussianBlur(frame, (21, 21), 0)
        hsv = cv2.cvtColor(blur, cv2.COLOR_BGR2HSV)

        lower = [110, 50, 50]    # threshold value for fire colour
        upper = [130, 255, 255]
        lower = np.array(lower, dtype="uint8")
        upper = np.array(upper, dtype="uint8")

        mask = cv2.inRange(hsv, lower, upper)

        output = cv2.bitwise_and(frame, hsv, mask=mask)

        no_red = cv2.countNonZero(mask)

        if int(no_red) > 15000:
            Fire_Reported = Fire_Reported + 1
        cv2.imshow("output",output)
        cv2.imshow("video",frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):   #For killing the program
            break

    cv2.destroyAllWindows()
    video.release()
```

GITHUB LINK : https://github.com/IBM-EPBL/IBM-Project-20358-1659717978
SCAN QR CODE:

PROJECT DEMO LINK :
https://drive.google.com/file/d/1ej5SGXuiUcXylVTPP-JpVJ8wzjdxTHGA/view?usp=share_link
WOKWI LINK: https://wokwi.com/projects/348132773889835602