

PROJECT REPORT

PROJECT NAME	GAS LEAKAGE MONITORING & ALERTING SYSTEM FOR INDUSTRIES
TEAM ID	PNT2022TMID18536
TEAM MEMBERS	SHANMUGAM S VIGNEASHWARAN B VISHNU V THIRUMURUGAN M
BRANCH	COMPUTER SCIENCE AND ENGINEERING

CONTENTS

1.INTRODUCTION

1.1 Overview

1.2 Purpose

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem statement

3.IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation and Brain Storming

3.3 Proposed Solution

3.4.Problem Solution Fit

4. THEORITICAL ANALYSIS

4.1 Functional requirement

4.2 Non Functional requirement

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution and Technical Architecture

6. PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

7.2 Feature 2

8. TESTING

8.1 Test Cases

9. RESULTS

9.1 Performance Metrics

10. ADVANTAGES/DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

14. GITHUB and DEMO LINK

1.INTRODUCTION

1.1 Project Overview:

The internet of Things is a developing topic of technical, social, and economic significance. The usage of the gas brings great problems in the domestic as well as working places. The inflammable gas, which is excessively used in the work places (Industries). The leakage of the gas causes destructible impact to the lives and as well as to the heritage of the people. Most of the societies have fire safety mechanism. But it can use after the fire exists. As a result, a system for detecting and monitoring gas leaks is required. Through a flame sensor, the system will sense fire and flame. The buzzer begins to ring when a fire is detected. Once the leakage is detected it will alert the user about the leakage. The performance that was produced showed that it was successful in detecting the gas leakage

1.2 Purpose:

The design of a sensor-based automatic gas leakage detector with an alert system has been proposed. This is an affordable, less power using, lightweight, portable, safe, user friendly, efficient, multi featured and simple system device for detecting gas. To monitor this gas leak, the system includes an MQ6 gas detector. This sensor detects the amount of leaking gas present in the surrounding atmosphere. In this way, the consequences of an explosion or gas leak can be avoided.

2.LITERATURE SURVEY :

2.1 Existing Problem:

Gas leakage is nothing but the leak of any gaseous molecule from a pipeline, or cylinder etc in the industries. Gas Leakages in open or closed areas can prove to be dangerous .This can occur either purposefully or even unintendedly. As we are aware that these kinds of leaks are dangerous to our health, and when it becomes explosive it could cause great danger to the people, industry and the environment. Therefore, we have used IoT technology to make a Gas Leakage Detector for society which has Smart Alerting techniques involving sending a text message to the concerned authority and the ability to view their gas level to prevent the gas leak. This will detect the harmful gases in the environment and alerting everyone through sending notifications.

2.2 References:

1. Shital Imade, Priyanka Rajmanes, Aishwarya Gavali , Prof. V. N. Nayakwadi "GAS LEAKAGE DETECTION AND SMART ALERTING SYSTEM USING IOT"
<https://www.pramanaresearch.org/gallery/22.%20feb%20ijirs%20-%20d539.pdf>
2. Kumar Keshamoni and Sabbani Hemanth. "Smart Gas Level Monitoring, Booking

& Gas Leakage Detector over IoT " International Advance Computing Conference IEEE, 2017.

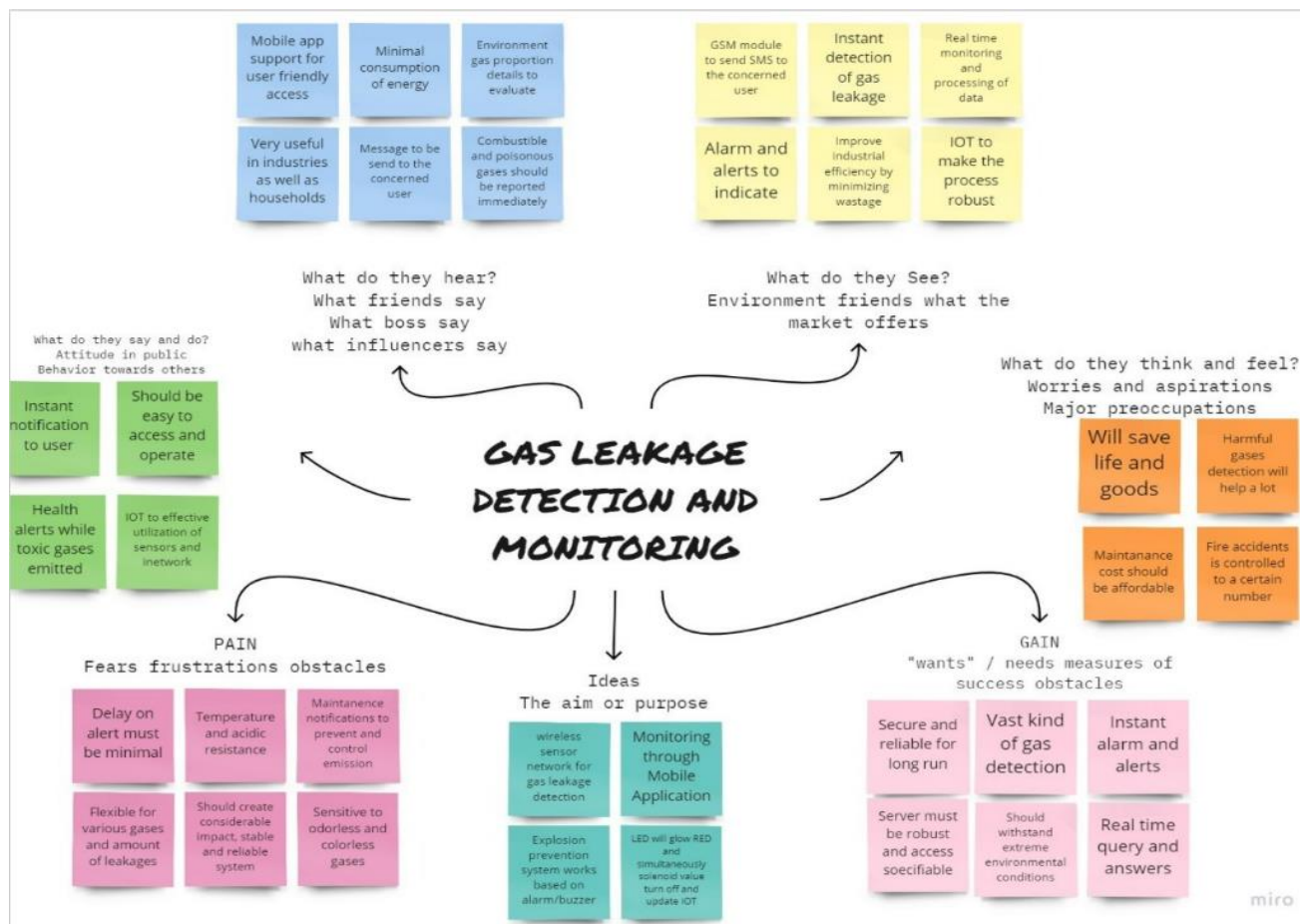
3. Petros Spachos , Liang Song and Dimitrios Hatzinakos. "Gas Leak Detection and Localization System Through Wireless Sensor Networks" The 11th Annual IEEE Consumer Communications and Networking Conference - Demos. IEEE, 2014.
4. "Design and Implementation of an Economic Gas Leakage Detector" National Institute of Health (2004). What you need to know about natural gas detectors.
Available:http://www.nidcd.nih.gov/health/smelltaste/gas_dtctr.asp.
5. Prof.M.Amsaveni, A.Anurupa, R.S.Anu Preetha, C.Malarvizhi,M.Gunasekaran
"Gsm based LPG leakage detection and controlling system" the International Journal of Engineering and Science (IJES) ISSN (e): 2319 – 1813 ISSN (p):2319 – 1805 Pages 112-116 March- 2015.
6. Srinivasan,Leela,Jeyabharathi,Kirthika,Rajasree"GAS LEAKAGE DETECTION AND CONTROL" Scientific Journal of Impact Factor(SJIF): 3.134.
7. Pal-Stefan Murvaya, IoanSileaa "A survey on gas leak detection and localization techniques".
8. Ch. Manohar Raju, N. Sushma Rani, "An android based automatic gas detection and indication robot. In International Journal of Computer Engineering and Applications. 2014;8(1).
9. Falohun A.S., Oke A.O., Abolaji B.M. "Dangerous Gas Detection using an Integrated Circuit and MQ-9" in International Journal of Computer Applications (0975 –8887) Volume 135 – No.7, February 2016.
- 10.Ashish Shrivastava,Ratnesh Prabhaker, Rajeev Kumar and Rahul Verma "GSM BASED GAS LEAKAGE DETECTION SYSTEM" in International Journal of Technical Research and Applications e-ISSN: 2320- 8163,www.ijtra.com Volume 1, Issue 2 (may-June 2013).
- 11.C.Selvapriya, S.Sathyaprabha, M.Abdulrahim," LPG leakage monitoring and multilevel alerting system", published in 2013.
- 12.Falohun A.S., Oke A.O., Abolaji B.M. "Dangerous gas detection using an integrated circuit and MQ-9. In International Journal of Computer Applications. 2016; 135(7).

2.3 Problem Statement Definition:

In most industries, one of the key parts of any safety plan for reducing risks to personnel and plant is the use of early-warning devices such as gas detectors. These can help to provide more time in which to take remedial or protective action. They can also be used as part of a total, integrated monitoring and safety system for an industrial plant. Rapid expansion of oil and gas industry leads to gas leakage incidents which are very serious and dangerous. Solutions need to be found out at least to minimize the effects of these incidents since gas leaks also produce a significant financial loss. Solution should be in a way to reduce the gas leak by obtaining gas level and alerting the concerned authority by sending alert messages.

3. IDEATION & PROPOSED SOLUTION:

3.1 Empathy Map Canvas:



3.2 Ideation & Brainstorming:

[illegible]

3

Group ideas

Take some sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is large enough to fill a sticky note, try and see if you can break it up into smaller sub-groups.

20 minutes

Tip

Add a convenient page or sticky notes to allow ideas to be added, changed, moved, and rearranged. You can also use a large sheet of paper as a drawing surface for your ideas.

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

5

After you collaborate

You can request the result as an image or as help to share with members of your company who might find it helpful.

Quick add-ons

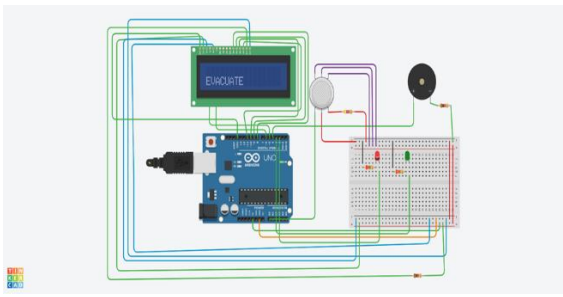
- Show the result**
Share a note like the input with stakeholders to help them in their work about the outcomes of the session.
- Report the result**
Report a copy of the result as a PDF or PPT to obtain its content, create a video, or share it with your team.

Keep moving forward

- Strategy blueprint**
Define the components of a new idea or strategy.
- Customer experience journey map**
Understand customer needs, emotions, and objectives for an experience.
- Map the results**
- Strategy, weaknesses, opportunities & threats**
Identify strengths, weaknesses, opportunities, and threats (SWOT) to structure a plan.

Show template feedback

3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To detect the gas leakage to alert the user through notification
2.	Idea / Solution description	In order to have a control over such conditions we proposed system that uses sensors which is capable of detecting the gases such as LPG, CO ₂ , CO and CH ₄ . This system will not only able to detect the leakage of gas but also alerting through audible alarms.
3.	Novelty / Uniqueness	<ul style="list-style-type: none">• Ability to predict the hazardous situation• Low cost
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none">• This model is vital for the society as there are lot of people unable to detect the gas leakage prior the fire accident.• we have used the IoT technology to make a Gas Leakage Detector for society which having Smart Alerting techniques involving sending text message to the concerned authority and an ability performing data analytics on sensor readings.
5.	Business Model (Revenue Model)	

6.	Scalability of the Solution	Develop a proposed system which include some safety factors.
----	-----------------------------	--

3.4 Problem Solution fit:

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS <p>The industrialists are the users or customers, who are engaged with the production of gases for their manufacturing. Here industrial worker is the user or customer, who are engaged with gas related production.</p>	6. CUSTOMER CC <p>High cost of installing the products make them to move far from recent technologies. It is difficult to know failures. Ability to detect the wide range of gases</p>	5. AVAILABLE SOLUTIONS AS <p>The monitoring and detecting the leakage of gas could be done by the manpower. Automatic cut off gas supply. In early days they used to identify the leakage of gas by sensing the smell of particular gas. Even though man power could reduce electricity cost and monitor properly, it may cause high risk for their life.</p>	Explore AS, differentiate
	2. JOBS-TO-BE-DONE / PROBLEMS J&P <ul style="list-style-type: none"> Gas leakage leads to many diseases and also increases the fatality rate. Heavy budget problems on buying and installing a gas detecting system Having no proper maintenance or monitoring the system Flammable gas leakage may lead to Secondary accident such as fire and explosion, while toxic gas. 	9. PROBLEM ROOT CAUSE RC <ul style="list-style-type: none"> Improperly installed tube fittings /poor tubing selection. Improper use of gas furnace, stove, or appliance, including leaking due to gas lines being hooked up incorrectly. Use of defective equipment. Behind this gas leakage problem there could be many reasons like atomic reactions between molecules and material quality. 	7. BEHAVIOUR BE <ul style="list-style-type: none"> If the gas leaked is heavily toxic, there is a chance of causing hereditary health hazards. Monitoring the system regularly. To determine the gas leakage area and alerts through by warning message or alerting sound. Using manpower as the source of monitoring the leakage causes high hazards. 	
Identify strong TR & EM	3. TRIGGERS TO ACT TR <p>Identification of gas leakage will be done immediately and urges them to find out a solution as soon as possible. Health issues due to the toxic gases urges them to find out a solution</p>	10. YOUR SOLUTION SL <ul style="list-style-type: none"> Develop a cost efficient IoT based gas leakage detecting system which can be easily accessed by the workers. If there is gas leak then it will alert the workers by sending SMS. 	8. CHANNELS OF BEHAVIOUR CH ONLINE: <p>Promoting through social media, With the help of social media influencer. Users can also easy to monitor the live reports.</p>	Extract online & offline CH of BE
	4.EMOTIONS: BEFORE / AFTER EM <p>Before: The leakage of gases causes heavy losses and made them feel depressed & guilt and also lose the recognition of their products. After: Creating awareness and safety precautions to the workers to work without any fear.</p>		OFFLINE: <p>Identifying the leakage area and take precautionary actions manually. It makes call to user. Frequently check the leakage of gas</p>	

4.

REQUIREMENT ANALYSIS:

4.1 Functional requirement:

Business Requirements	User Requirements	Product Requirements
The gas leakage detection system can be deployed in homes, hotels, factory units, LPG cylinder storage areas, and so on. The main advantage of this IoT and Arduino-based application is that it can determine the leakage and send the data over to a site. It can be monitored, and preventive measures can be taken to avoid any disaster.	The gas leakage detection system can be optimized for detecting toxic gasses along with upgrading them with smoke and fire detectors to identify the presence of smoke and fire. Ensuring worker safety is important but making using of the right technology is even more vital.	Detecting gasses is necessary regardless of your business role or individual purpose. Certain technologies at play make such IoT devices what they are, and if you want to indulge in IoT application development, you must know what they are and what purpose they can fulfill.

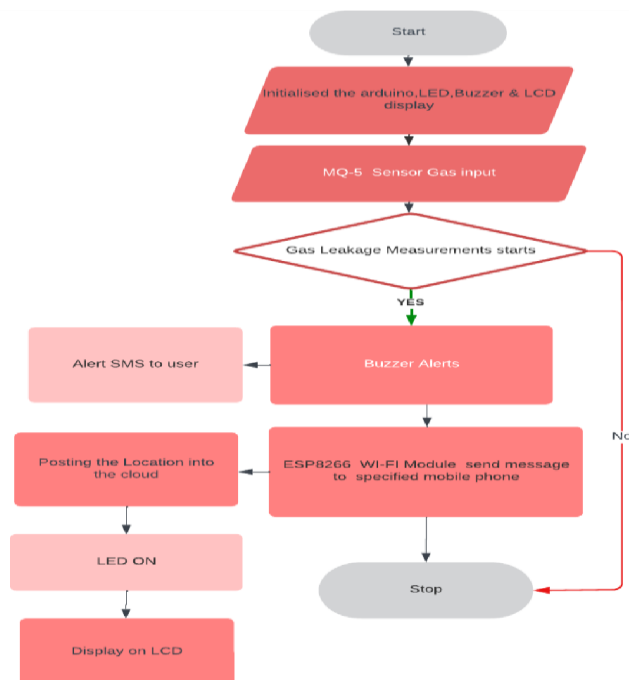
4.2 Non-Functional requirements:

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The sensors used to detect the gas leakage which helps to prevent the high risk of gas explosion and also can prevent the causalities within and outside the covering area of the industries.
NFR-2	Security	The device is intended for the use of industries or factories, where there is a use of explosive gas is a source of risk. This device will help and secures from the causes.
NFR-3	Reliability	Gas leakage detecting system detects the gas leakage at industries or factories which detects the small amount of gas leakage as soon and sends the alerting SMS to users.
NFR-4	Performance	The Gas leakage detecting system is a device with an alarm setting. Whenever there is a gas leak ,which is greater than the threshold level, the inbuild sensor detects and alerts the user within a minute much before it can cause any accidents.

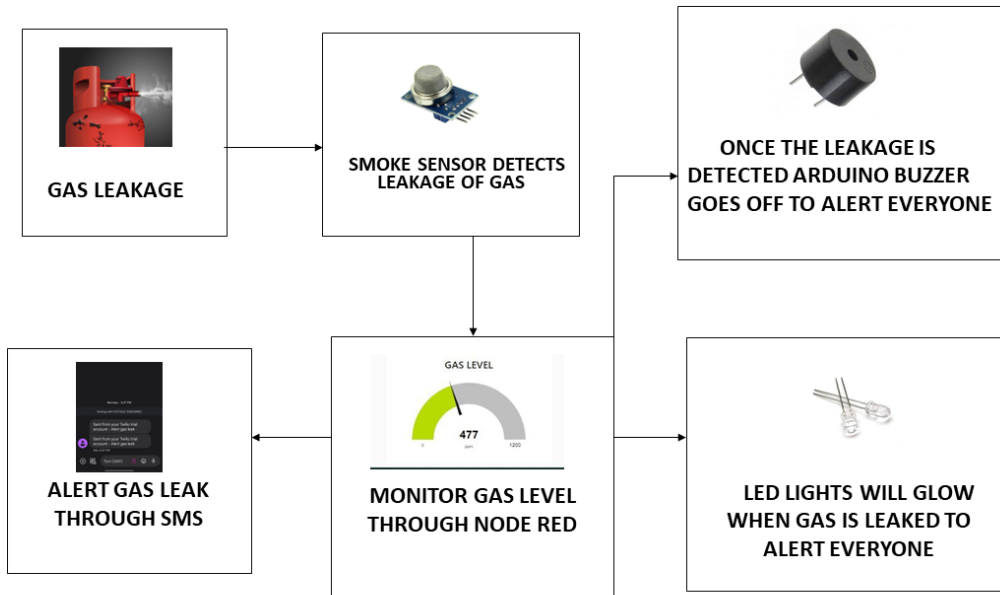
NFR-5	Availability	The gas leakage detecting system is readily available in the market which is extremely expensive, but here we are providing a low-cost circuit for gas leakage detecting system and also it is user friendly
NFR-6	Scalability	The system is very simple and easy to maintain with cost efficient. A backup power supply will be included in the design to prevent from the power failure conditions. It has the capability to works for a period of time without any damage in the system components.

PROJECT DESIGN:

5.1 Data Flow Diagrams:



5.2 Solution & Technical Architecture:



5.3 User Stories:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Monitor the gas leakage	USN-1	The Industrialist have own industries so the industry owner must take of workers .The workers have family so the industries give security assurance of workers.	2	High	Shanmugam S Vigneashwaran B Vishnu V Thirumurugan M
Sprint-2	Avoid From Disaster	USN-2	The gas leakage occur at the time fire service will take care to protect the people from the disaster.	1	High	Shanmugam S Vigneashwaran B Vishnu V Thirumurugan M
Sprint-3	Detect the gas	USN-3	We have monitor the gas by 24/7 hrs. To avoid leakage ,the industry have quality pipes to transfer the gas and proper maintenance service once in a month. The industry must take care of what are the necessary process to avoid the gas leakage.	2	Low	Shanmugam S Vigneashwaran B Vishnu V Thirumurugan M
Sprint-4	The model is trained and tested by sample dataset.	USN-4	The programmer design the model to detect the gas leakage.	2	Medium	Shanmugam S Vigneashwaran B Vishnu V Thirumurugan M
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-5	Warning message	USN-5	Incase any gas leakage occur, the device give the alarm and alert message to concerned user within a minute.	1	High	Shanmugam S Vigneashwaran B Vishnu V

6 PROJECT PLANNING & SCHEDULING:

6.1 Sprint Planning & Estimation:

6.1.1 SPRINT PLAN

6.1.2 ANALYZE THE PROBLEM

6.1.3 PREPARE an ABSTRACT, PROBLEM STATEMENT

6.1.4 LIST A REQUIRED OBJECT NEEDED

6.1.5 CREATE A PROGRAM CODE AND RUN IT

6.1.6 MAKE A PROTOTYPE TO IMPLEMENT

6.1.7 TEST WITH THE CREATED CODE AND CHECK THE DESIGNED PROTOTYPE

6.2 Sprint Delivery Schedule:

Sprint	Functional Requirement (Epic)	User Story	User Story / Task	Story Point	Priority
Sprint-1	Create	US-1	Create the IBM Cloud services which are being used in this project.	5	High
Sprint-1	Configure	US-2	Configure the IBM Cloud services which are being used in completing this project.	1	Medium
Sprint-1	Create	US-3	IBM Watson IoT platform acts as the mediator to connect the web application to IoT devices, so create the IBM Watson IoT platform.	1	Medium
Sprint-1	Configure	US-4	Configure the IBM Watson IoT which are being used to display the output.	13	High

Sprint-2	Create	US-1	In order to connect the IoT device to the IBM cloud, create a device in the IBM Watson IoT platform and get the device credentials.	13	High
----------	--------	------	---	----	------

Sprint-2	Configure	US-2	Configure a device in the IBM Watson IoT platform and get the device credentials.	3	Medium
Sprint-2	Create	US-3	Create a Node-RED service.	3	High
Sprint-2	Configure	US-4	Configure the connection security and create API keys that are used in the Node- RED service for accessing the IBM IoT Platform.	1	Medium
Sprint-3	Develop	US-1	Develop a python script to publish random sensor data such as temperature, Flame level and Gas level to the IBM IoTplatform	1 3	High

Sprint-3	Configure	US-2	After developing python code and commands just run the code	1	Medium
Sprint-3	Print	US-3	Print the statements which represent the control of the devices.	1	Low
Sprint-3	Publish	US-4	Publish Data to The IBM Cloud	5	High
Sprint-4	Create	US-1	Create Web UI in Node- Red	5	High
Sprint-4	Configure	US-2	Configure the Node-RED flow to receive data from the IBMIoT platform	5	High
Sprint-4	Configure	US-3	Use cloudant DB nodes to store the received sensor data in the cloudant DB	5	High
Sprint-4	Publish	US-4	Publish the received data in webapplication	5	High

7. CODING & SOLUTIONING:

```
import time
import sys
import random
import wiotp.sdk.device# IBM IoT Watson Platform Module
import ibmiotf.device
import tkinter as tk # Python GUI Package
from tkinter import ttk # Python GUI
import time
from threading import Thread

organization = "venslr"
deviceType = "GASLEAKAGE"
deviceId = "1234"
authMethod = "token"
authToken = "12345678"
# Tkinter root window
root = tk.Tk()
root.geometry('350x300') # Set size of root window
root.resizable(True, True) # root window non-resizable
root.title('Gas Leakage Monitoring And Alerting System for Industries ')

# Layout Configurations
root.columnconfigure(0, weight=1)
```

```

root.columnconfigure(1, weight=3)

current_gas = tk.DoubleVar()

def get_current_gas(): # function returns current gas level value
    return '{: .2f}'.format(current_gas.get())

def slider_changed(event): # Event Handler for changes in sliders
    print('-----')
    print('Gas Level: {: .2f}'.format(current_gas.get()))
    print('-----')
    gas_label.configure(text=str(get_current_gas()) + " ppm") # Displays current gas level as
label content

# Tkinter Labels

# label for the gas level slider /
slider_gas_label = ttk.Label(root, text='Set Gas Level:')
slider_gas_label.grid(column=0, row=0, sticky='w')

# Gas Level slider
slider_gas = ttk.Scale(root, from_=0, to=1300, orient='horizontal',
command=slider_changed, variable=current_gas)
slider_gas.grid(column=1, row=0, sticky='we')

# current gas level label

```

```
current_gas_label = ttk.Label(root,text='Current Gas Level:')
current_gas_label.grid(row=1,columnspan=2,sticky='n',ipadx=10,ipady=10)
```

```
# Gas level label (value gets displayed here)
gas_label = ttk.Label(root,text=str(get_current_gas()) + " ppm")
gas_label.grid(row=2,columnspan=2,sticky='n')
```

```
def publisher_thread():
    thread = Thread(target=publish_data)
    thread.start()

def publish_data():
    # Exception Handling
    try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-
method": authMethod,
                        "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        # .....

    except Exception as e:
        print("Caught exception connecting device: %s" % str(e))
        sys.exit()
```

```
deviceCli.connect() # Connect to IBM Watson IoT Platform
```

```
while True:
```

```
    gas_level = int(current_gas.get())
```

```
    temp_level = random.randint(10,80)
```

```
    hum=random.randint(0,10)
```

```
    pre=random.randint(0,20)
```

```
    data={
```

```
'gas_level' : gas_level,
```

```
'Temperature':temp_level,
```

```
'Humidity':hum,
```

```
'Pressure':pre
```

```
}
```

```
def myOnPublishCallback():
```

```
    if ( gas_level >600):
```

```
        print("Gas Level = %s ppm" % gas_level, "Automatic sprinkler turned on")
```

```
    print("Published Gas Level = %s ppm" % gas_level, "to IBM Watson")
```

```
    print("Published humidity = %s ppm" % hum, "to IBM Watson")
```

```
    print("Published temperature = %s ppm" % temp_level, "to IBM Watson")
```

```
    print("Published pressure = %s ppm" % pre, "to IBM Watson")
```

```
        success = deviceCli.publishEvent("event", "json", data, qos=0,  
on_publish=myOnPublishCallback)
```

```
    if not success:
```

```
        print("Not connected to IoTTF")
```

```
    time.sleep(1)
```

```
publisher_thread()
```

```
root.mainloop() # startup Tkinter GUI
```

```
# Disconnect the device and application from the cloud
```

```
deviceCli.disconnect()
```

CODE:

```
"ibmmain.py - C:\Users\Sunmugam\Desktop\ibmmain.py (3.9.1)"
File Edit Format Run Options Window Help

import time
import sys
import random
import wiotp.adk.device# IBM IoT Watson Platform Module
import ibmiotf.device
import tkinter as tk # Python GUI Package
from tkinter import ttk # Python GUI
import time
from threading import Thread

organization = "venslr"
deviceType = "GASLEAKAGE"
deviceId = "1234"
authMethod = "token"
authToken = "12345678"
# Tkinter root window
root = tk.Tk()
root.geometry('350x300') # Set size of root window
root.resizable(True, True) # root window non-resizable
root.title('Gas Leakage Monitoring And Alerting System for Industries ')

# Layout Configurations
root.columnconfigure(0, weight=1)
root.columnconfigure(1, weight=3)

current_gas = tk.DoubleVar()

def get_current_gas(): # function returns current gas level value
    return '{: .2f}'.format(current_gas.get())

def slider_changed(event): # Event Handler for changes in sliders
    print('-----')
    print('Gas Level: ({: .2f})'.format(current_gas.get()))
    print('-----')

Ln: 17 Col: 0
```

```
"ibmmain.py - C:\Users\Sunmugam\Desktop\ibmmain.py (3.9.1)"
File Edit Format Run Options Window Help

    print("Caught exception connecting device: %s" % str(e))
    sys.exit()

deviceCli.connect() # Connect to IBM Watson IoT Platform

while True:
    gas_level = int(current_gas.get())
    temp_level = random.randint(10,80)
    hum=random.randint(0,10)
    pre=random.randint(0,20)

    data={
        'gas_level': gas_level,
        'Temperature':temp_level,
        'Humidity':hum,
        'Pressure':pre
    }

    def myOnPublishCallback():
        if ( gas_level >600):
            print("Gas Level = %s ppm" % gas_level, "Automatic sprinkler turned on")

            print("Published Gas Level = %s ppm" % gas_level, "to IBM Watson")
            print("Published humidity = %s ppm" % hum, "to IBM Watson")
            print("Published temperature = %s ppm" % temp_level, "to IBM Watson")
            print("Published pressure = %s ppm" % pre, "to IBM Watson")

    success = deviceCli.publishEvent("event", "json", data, qos=0, on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoT")
        time.sleep(1)

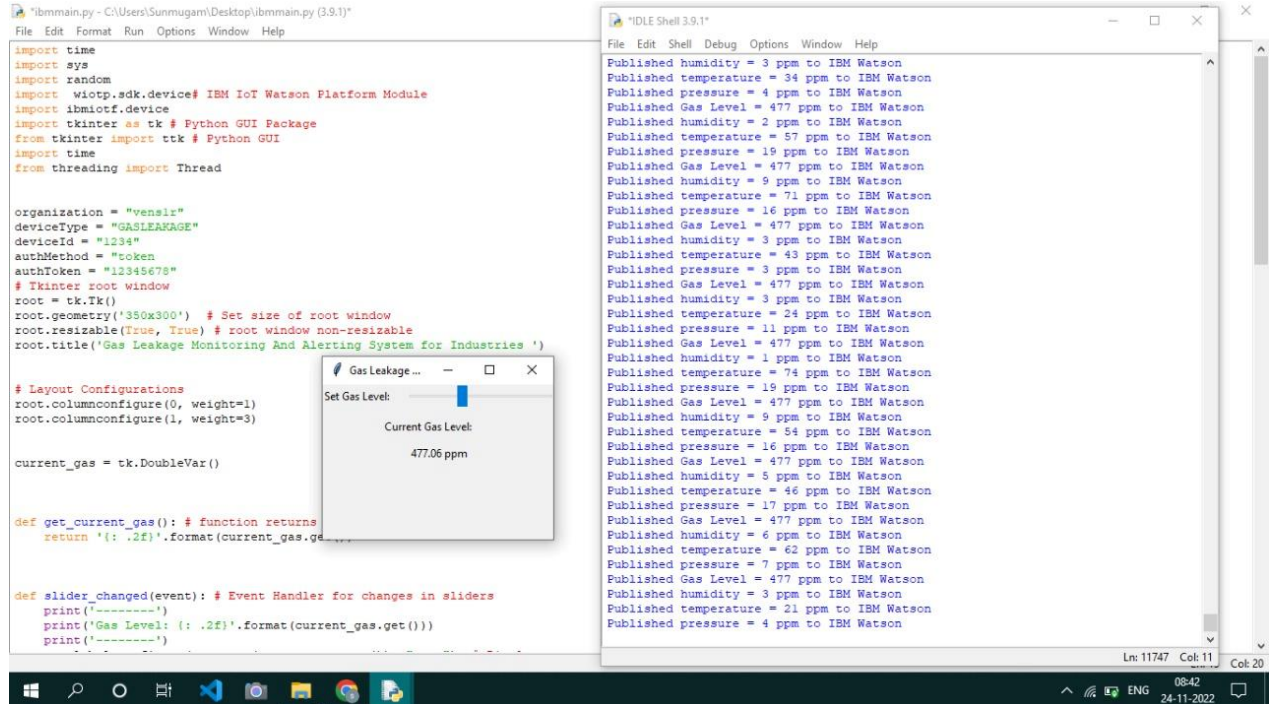
publisher_thread()

root.mainloop() # startup Tkinter GUI

# Disconnect the device and application from the cloud
deviceCli.disconnect()

Ln: 17 Col: 0
```

OUTPUT:



```
ibmmain.py - C:\Users\Sunmugam\Desktop\ibmmain.py (3.9.1)*
File Edit Format Run Options Window Help

import time
import sys
import random
import wiotp.sdk.device # IBM IoT Watson Platform Module
import ibmiotf.device
import tkinter as tk # Python GUI Package
from tkinter import ttk # Python GUI
import time
from threading import Thread

organization = "venslr"
deviceType = "GASLEAKAGE"
deviceId = "1234"
authMethod = "token"
authToken = "12345678"
# Tkinter root window
root = tk.Tk()
root.geometry('350x300') # Set size of root window
root.resizable(True, True) # root window non-resizable
root.title('Gas Leakage Monitoring And Alerting System for Industries ')

# Layout Configurations
root.columnconfigure(0, weight=1)
root.columnconfigure(1, weight=3)

current_gas = tk.DoubleVar()

def get_current_gas(): # function returns
    return '{: .2f}'.format(current_gas.get())

def slider_changed(event): # Event Handler for changes in sliders
    print('-----')
    print('Gas Level: ({: .2f})'.format(current_gas.get()))
    print('-----')
```

```
IDLE Shell 3.9.1*
File Edit Shell Debug Options Window Help

Published humidity = 3 ppm to IBM Watson
Published temperature = 34 ppm to IBM Watson
Published pressure = 4 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 2 ppm to IBM Watson
Published temperature = 57 ppm to IBM Watson
Published pressure = 19 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 9 ppm to IBM Watson
Published temperature = 71 ppm to IBM Watson
Published pressure = 16 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 3 ppm to IBM Watson
Published temperature = 43 ppm to IBM Watson
Published pressure = 3 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 3 ppm to IBM Watson
Published temperature = 24 ppm to IBM Watson
Published pressure = 11 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 1 ppm to IBM Watson
Published temperature = 74 ppm to IBM Watson
Published pressure = 19 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 9 ppm to IBM Watson
Published temperature = 54 ppm to IBM Watson
Published pressure = 16 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 5 ppm to IBM Watson
Published temperature = 46 ppm to IBM Watson
Published pressure = 17 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 6 ppm to IBM Watson
Published temperature = 62 ppm to IBM Watson
Published pressure = 7 ppm to IBM Watson
Published Gas Level = 477 ppm to IBM Watson
Published humidity = 3 ppm to IBM Watson
Published temperature = 21 ppm to IBM Watson
Published pressure = 4 ppm to IBM Watson

Ln: 11747 Col: 11 Col: 20
```

08:42
24-11-2022

8. Testing:

				Team ID	PNT2022TMD18536					
				Project Name	GAS LEAKAGE MONITORING & ALERTING SYSTEM FOR INDUSTRIES					
				Maximum Marks	4 marks					
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	
HomePage_TC_009	Functional	Home Page	Verify the IBM Watson connection	IBM WATSON CLOUD	1.Run the python code 2.In IBM Watson Connection is enabled	Connection and Python Code	Connected Successfully	Working as expected	Pass	
PredictPage_TC_010	Functional	Predict Page	Run the node red and increase the gas level	Node Red Editor	1.Enter the url 2.Gas Level will be displayed.	gas_level:660, Temperature: 20, Pressure:10, Humidity:5	User can view the gas level and warning popup message	Working as expected	Pass	
PredictPage_TC_011	Functional	Predict Page	Send the SMS	Twilio Account	1.Enter the twilio account details 2.Enter the message which you want to send.	Alert gas Leak	SMS sent Successfully	Working as expected	Pass	

UAT TESTING:

Defect Analysis

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	19
Duplicate	1	0	3	0	4
External	2	3	0	1	6

Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	24	9	10	24	67

TESTING SCREENSHOTS:

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various IoT functions. The main content area shows details for a device with ID '1234', which is 'Connected' and named 'GASLEAKAGE'. The 'Recent Events' tab is active, displaying a table of live data events.

Event	Value	Format	Last Received
event	{"gas_level":608,"Temperature":44,"Humidity":0...	json	a few seconds ago
event	{"gas_level":608,"Temperature":12,"Humidity":7...	json	a few seconds ago
event	{"gas_level":608,"Temperature":37,"Humidity":1...	json	a few seconds ago

At the bottom of the dashboard, there is a status bar indicating 'Items per page: 50' and '1 of 1 page'.

Application Details - Node-RED : node-red - IBM Watson IoT Platform - IBM-EPBL/IBM-Proje - Node-RED Dashboard - (1) WhatsApp

node-red-iswxp-2022-11-21.eu-gb.mybluemix.net/red/#flow/844ceed33037c78

Node-RED

Flow 1

debug

11/24/2022, 8:10:59 AM node: f2c2649a.0d0d98
iot-2/type/GASLEAKAGE/id/1234/evt/event/fmt/json :
msg.payload : string[11]
"Gas Leakage"

11/24/2022, 8:11:00 AM node: f2c2649a.0d0d98
iot-2/type/GASLEAKAGE/id/1234/evt/event/fmt/json :
msg.payload : Object
{ gas_level: 608, Temperature: 45, Humidity: 10, Pressure: 11 }

11/24/2022, 8:11:00 AM node: f2c2649a.0d0d98
iot-2/type/GASLEAKAGE/id/1234/evt/event/fmt/json :
msg.payload : string[11]
"Gas Leakage"

11/24/2022, 8:11:00 AM node: f2c2649a.0d0d98
iot-2/type/GASLEAKAGE/id/1234/evt/event/fmt/json :
msg.payload : Object
{ gas_level: 608, Temperature: 37, Humidity: 7, Pressure: 18 }

11/24/2022, 8:11:00 AM node: f2c2649a.0d0d98
iot-2/type/GASLEAKAGE/id/1234/evt/event/fmt/json :
msg.payload : string[11]
"Gas Leakage"

Application Details - Node-RED : node-red - IBM Watson IoT Platform - IBM-EPBL/IBM-Proje - Node-RED Dashboard - WhatsApp - https://node-red -

node-red-iswxp-2022-11-21.eu-gb.mybluemix.net/ui/#/07/socketid=ckED_6rAfsGuorizAAB3

Gas Leakage Monitoring

Message

Gas Leakage

Gas Leakage ...

Set Gas Level:

Current Gas Level:
681.82 ppm

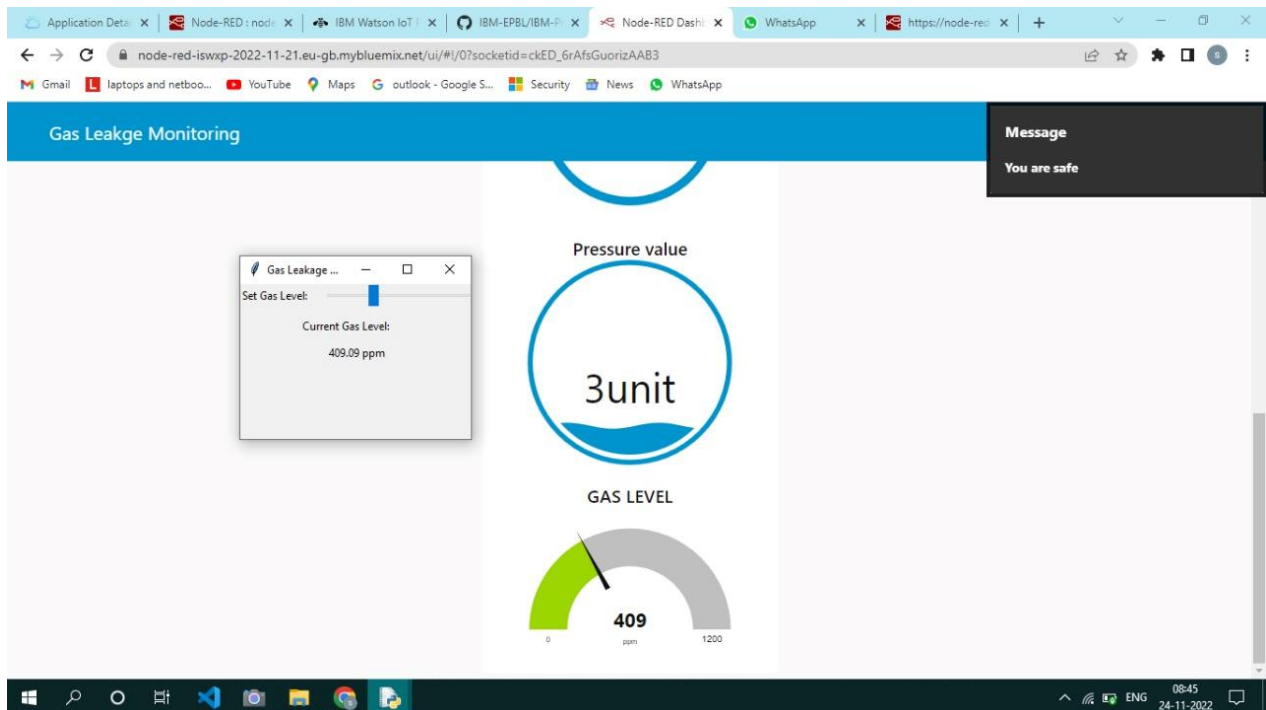
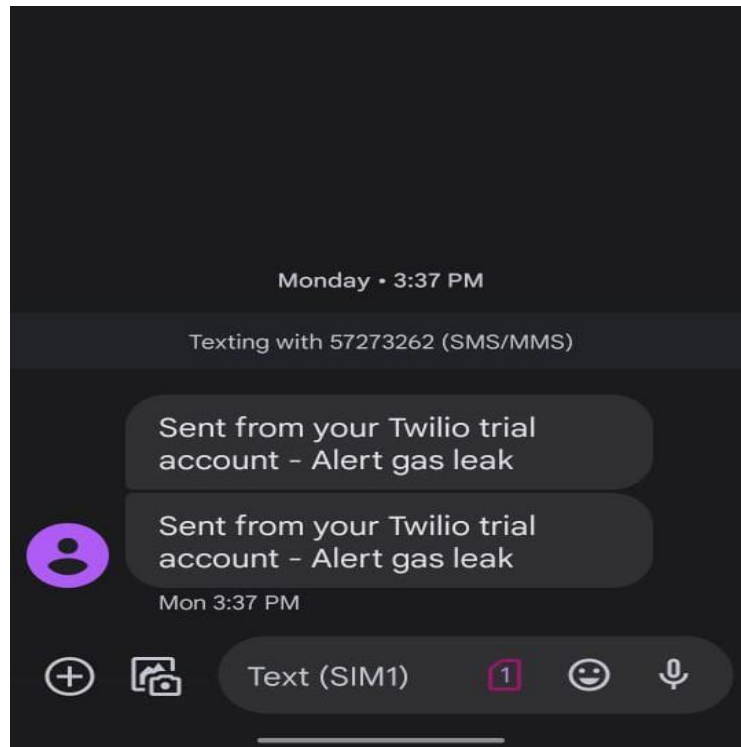
Pressure value

2unit

GAS LEVEL

681

0 1200



9. Result:

The system can be taken as a small attempt in connecting the existing primary gas detection methods to detect gas leakage. The gases are sensed in an area of 1m radius of the sensor. The accuracy of sensors is not up to the mark thus stray gases are also detected which creates an amount of error in the outputs of the sensors, especially in case of methane. Further the availability and storage of toxic gases like hydrogen sulphide also creates problems for testing the assembled hardware. As the system operates outside the pipeline, the complication of system maintenance and material selection of the system in case of corrosive gases is reduced. Thus, the system at this stage can only be use data primary indicator of leakage inside a plant. The system monitors the gas level and alert the concerned authority through SMS.

10. Advantages/Disadvantages:

10.1 Advantages:

1. Get real-time alerts about the gaseous presence in the atmosphere.
2. Prevent fire hazards and explosions.
3. Supervise gas concentration levels.
4. Ensure worker's health.
5. Real-time updates about leakages.
6. Cost-effective installation.
7. Data analytics for improved decisions.
8. Measure oxygen level accuracy.
9. Get immediate gas leak alerts.

10.2 Disadvantages:

1. It requires air or oxygen to work.
2. It gets reacted due to heating of wire.
3. It can be poisoned by lead, chlorine and silicon

11.CONCLUSION:

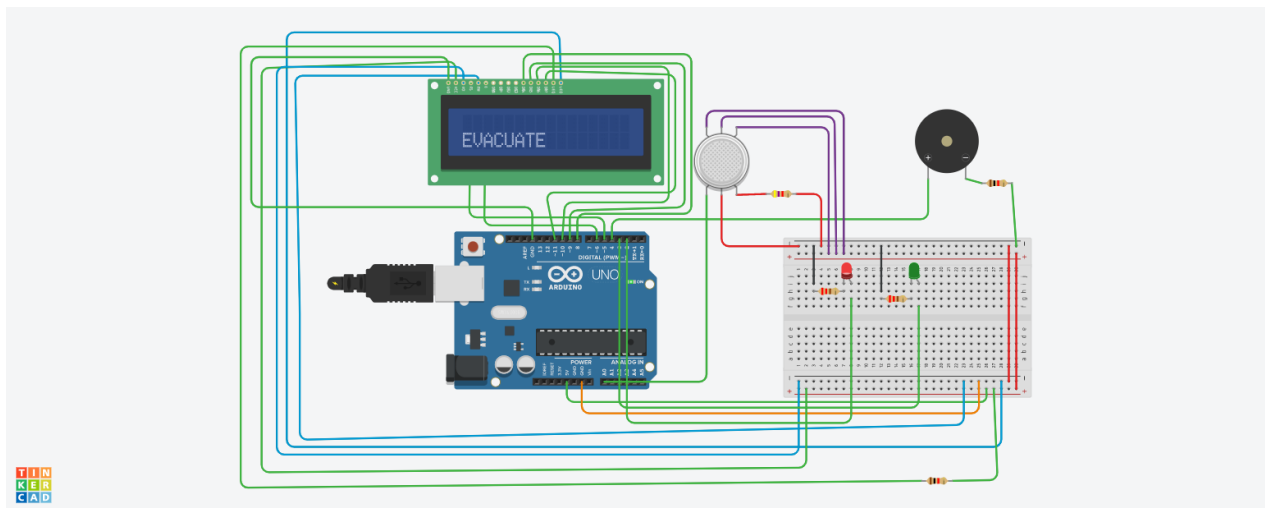
This gas leak detector system contains two features, this includes the SMS Gateway feature for only sending warning information regarding the gas leak to user, and the gas level alert for the warning alert. There is some improvement which can be applied for the future work, such as regarding the SMS Gateway, it need to enhance with feature such as notifying the user whenever the remaining credit balance is insufficient. Therefore, it is recommended to add this kind of features in the future work for better refinement.

12. FUTURE SCOPE:

We propose to build the system using an MQ6 gas detection sensor and interface it with an Arduino Uno microcontroller along with an LCD Display. This system uses the gas sensor to detect any gas leakages. The gas sensor sends out a signal to the microcontroller as soon as it encounters a gas leakage. The microcontroller processes this signal and a message is displayed on the LCD to alert the user.

13. APPENDIX:

13.1 Circuit Diagram:



13.2 Components:

The design of a sensor-based automatic gas leakage detector with an alert and control system. The components are

S.NO	NAME OF THE COMPONENT	QUANTITY
1	Arduino Uno R3	1
2	LCD 16x2	1
3	Piezo	1
4	Gas sensor	1
5	1 k ohm Resistor	1
6	2.3 k ohm Resistor	1
7	4.7 k ohm Resistor	1
8	Red LED	1
9	Green LED	1

13.3 Source Code:

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(5,6,8,9,10,11);

int redled = A5; int
greenled = A3; int
buzzer = 4; int sensor =
A0; int sensorThresh =
400; void setup()
{
```

```
pinMode(redled, OUTPUT);
pinMode(greenled,OUTPUT);
pinMode(buzzer,OUTPUT);
pinMode(sensor,INPUT);
Serial.begin(9600);
lcd.begin(16,2);
}
void loop()
{
  int analogValue = analogRead(sensor);
  Serial.println(analogValue);
  if(analogValue>sensorThresh)
  {
    digitalWrite(redled,HIGH);
    digitalWrite(greenled,LOW);
    tone(buzzer,1000,10000);
    lcd.clear();
    lcd.setCursor(0,1);
    lcd.print("ALERT");
    Serial.print("ALERT");
    delay(1000);
    lcd.clear();
    lcd.setCursor(0,1);
    lcd.print("EVACUATE");
    Serial.println(" -- EVACUATE");
    delay(1000);
  }
  else
  {
    digitalWrite(greenled,HIGH);
    digitalWrite(redled,LOW);
    noTone(buzzer);
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("SAFE");
    Serial.print("SAFE");
```



```
    delay(1000);  
    lcd.clear();  
    lcd.setCursor(0,1);  
    lcd.print("ALL CLEAR");  
    Serial.println(" -- ALL CLEAR");  
    delay(1000);  
}  
}
```

13.4 GITHUB:

Link : <https://github.com/IBM-EPBL/IBM-Project-26465-1660027284>

Demo Link:

https://drive.google.com/file/d/18eNKX0_1Ztpr4hsuy7uQm1o1hfDtAnbu/view?usp=drivesdk