



## IBM PROJECT REPORT

PROJECT NAME	GAS LEAKAGE MONITORING & ALERTING SYSTEM FOR INDUSTRIES
TEAM ID	PNT2022TMID16564
TEAM MEMBERS	SANTHOSH S SARAN S PRATHAP A SUNIL KUMAR K

# **CONTENTS**

## **1. INTRODUCTION**

- a. Project Overview
- b. Purpose

## **2. LITERATURE SURVEY**

- a. Existing problem
- b. References
- c. Problem Statement Definition

## **3. IDEATION & PROPOSED SOLUTION**

- a. Empathy Map Canvas
- b. Ideation & Brainstorming
- c. Proposed Solution
- d. Problem Solution fit

## **4. REQUIREMENT ANALYSIS**

- a. Functional requirement
- b. Non-Functional requirements

## **5. PROJECT DESIGN**

- a. Data Flow Diagrams
- b. Solution & Technical Architecture
- c. User Stories

## **6. PROJECT PLANNING & SCHEDULING**

- a. Sprint Planning & Estimation
- b. Sprint Delivery Schedule
- c. Reports from JIRA

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

- a. Feature 1
- b. Feature 2
- c. Database Schema (if Applicable)

## **8. TESTING**

- a. Test Cases
- b. User Acceptance Testing

## **9. RESULTS**

- a. Performance Metrics

## **10. ADVANTAGES & DISADVANTAGES**

## **11. CONCLUSION**

## **12. FUTURE SCOPE**

## **13. APPENDIX**

Source Code

GitHub & Project 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. Tests have shown that the system can keep track of the wastage of gas and leaks and notify the user. The performance that was produced showed that it was successful in reducing the amount of gas that was wasted.

## **1.2 Purpose:**

The design of a sensor-based automatic gas leakage detector with an alert and control 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 perform data analytics on sensor readings. Our main aim is to propose a gas leakage system for a society where each flat has gas leakage detector hardware. This will detect the harmful gases in the environment and alerting to society members through the alarm and 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](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:**

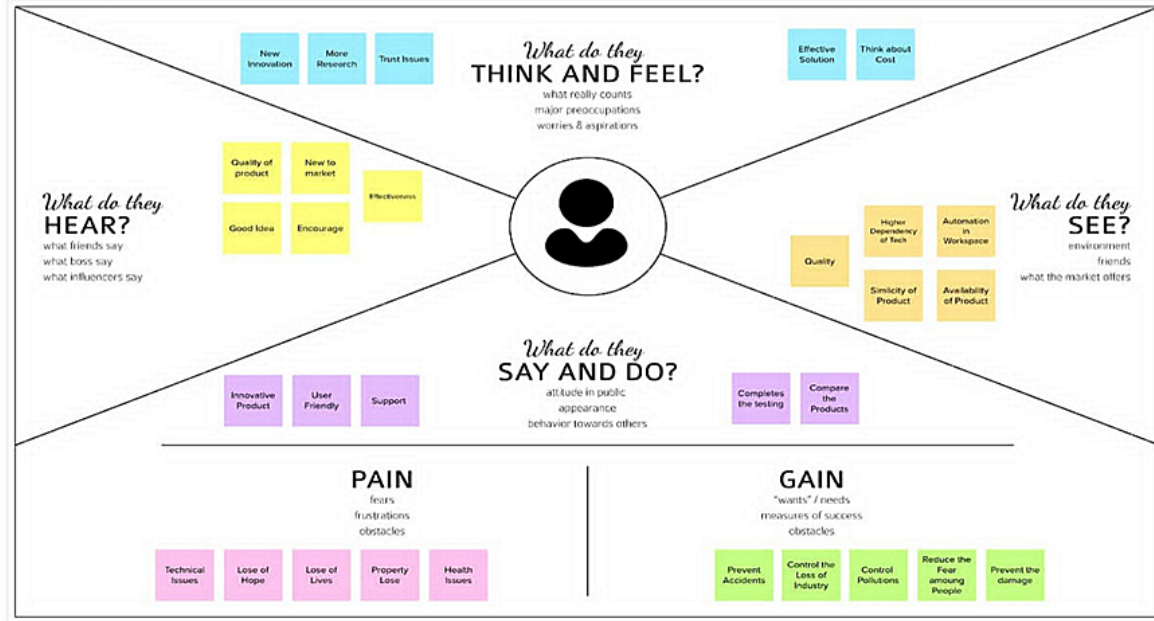
In most industries, one of the key parts of any safety plan for reducing risks to personnel and plant is the use of early-warning devices such as gas detectors. These can help to provide more time in which to take remedial or protective action. They can also be used as part of a total, integrated monitoring 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. The challenges are not only to design a prototype of the device that can only detect but also automatically respond to it whenever the leakage occurs.

### 3. IDEATION & PROPOSED SOLUTION

#### 3.1 Empathy Map Canvas:

1

Build empathy and keep your focus on the user by putting yourself in their shoes.



Share your feedback

### 3.2 Ideation & Brainstorming:

[illegible]

### 3 Group Ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence that states if it cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

⌚ 10 minutes

### 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 less so.

⌚ 10 minutes

### 5 Share the story

Share a story about the story you've just told. It's a chance to share your ideas with others and get feedback. You can also use this time to ask questions and clarify your ideas.

⌚ 10 minutes

### 6 After your collaboration

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

⌚ 10 minutes



### 3.3 Proposed Solution:

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	Gas leakage leads to various accidents resulting in loss of human lives and industry properties. Sometimes, the gas leakage cannot be detected by human that has a low sense of smell. Thus, this system will help to detect the presence of gas leakage and alert the users.
2.	Idea / Solution description	It detects the gas leakage by using various sensors. If the gas leakage level is above the threshold level, it sends the alert message through SMS to the user by using GSM module and buzzer the alarm.
3.	Novelty / Uniqueness	We use location tagging and alert service so that the admin and fire department team will be notified the exact location. The system provides constant monitoring and detection of gas leakage along with storage of data in database for predictions and analysis.
4.	Social Impact / Customer Satisfaction	By implementing real-time gas leak detection, industries can monitor their environmental performance, ensure better occupational health. Also, early detection of gas leaks can trigger concerned engineers to curtail the spread and keep a safe environment for better health and safety.

5.	Business Model (Revenue Model)	The product can be made compact, cost efficient and easily installable so that all the industries from small scale to large scale can able to buy the product .
6.	Scalability of the Solution	The system is very simple and easy to maintain and cost efficient. It has the capability to works for a period of time without any damage in the system components.

### 3.4 Problem Solution fit

Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> <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>	<b>6. CUSTOMER</b> <span>CC</span> <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>	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> <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
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span> <ul style="list-style-type: none"> <li>Gas leakage leads to many diseases and also increases the fatality rate.</li> <li>Heavy budget problems on buying and installing a gas detecting system</li> <li>Having no proper maintenance or monitoring the system</li> <li>Flammable gas leakage may lead to Secondary accident such as fire and explosion, while toxic gas.</li> </ul>	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> <ul style="list-style-type: none"> <li>Improperly installed tube fittings /poor tubing selection.</li> <li>Improper use of gas furnace, stove, or appliance, including leaking due to gas lines being hooked up incorrectly.</li> <li>Use of defective equipment.</li> <li>Behind this gas leakage problem there could be many reasons like atomic reactions between molecules and material quality.</li> <li></li> </ul>	<b>7. BEHAVIOUR</b> <span>BE</span> <ul style="list-style-type: none"> <li>If the gas leaked is heavily toxic, there is a chance of causing hereditary health hazards.</li> <li>Monitoring the system regularly.</li> <li>To determine the gas leakage area and alerts through by warning message or alerting sound.</li> <li>Using manpower as the source of monitoring the leakage causes high hazards.</li> </ul>	

Focus on J&P, tap into BE, understand

Focus on J&P, tap into BE, understand

Identify strong TR & EM	<b>3. TRIGGERS TO ACT</b> <span>TR</span> 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	<b>10. YOUR SOLUTION</b> <span>SL</span> <ul style="list-style-type: none"> <li>Develop a cost efficient IoT based gas leakage detecting system which can be easily accessed by the workers.</li> <li>If there is gas leak then it will alert the workers by sending SMS.</li> </ul>	<b>8. CHANNELS OF BEHAVIOUR</b> <span>CH</span> <b>ONLINE:</b> Promoting through social media, With the help of social media influencer. Users can also easy to monitor the live reports.	Extract online & offline CH of BE
	<b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span> <b>Before:</b> The leakage of gases causes heavy losses and made them feel depressed & guilt and also lose the recognition of their products. <b>After:</b> Creating awareness and safety precautions to the workers to work without any fear.		<b>OFFLINE:</b> Identifying the leakage area and take precautionary actions manually. It makes call to user. Frequently check the leakage of gas	

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form  Registration through Gmail  Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email  Confirmation via OTP
FR-3	GPS Access	GPS access to know the location
FR-4	Business Requirements	The device is intended for the use of industries or factories and also for cylinder storage areas. It detects the leakage of gas and sends the data over to a site and preventive measures can be taken to avoid the loss of properties.
FR-5	User Requirements	The Gas leakage detecting system with upgrading technologies which identifies the leakage of gas and also ensures the workers safety.

## 4. REQUIREMENT ANALYSIS

### 4.1 Functional Requirements:

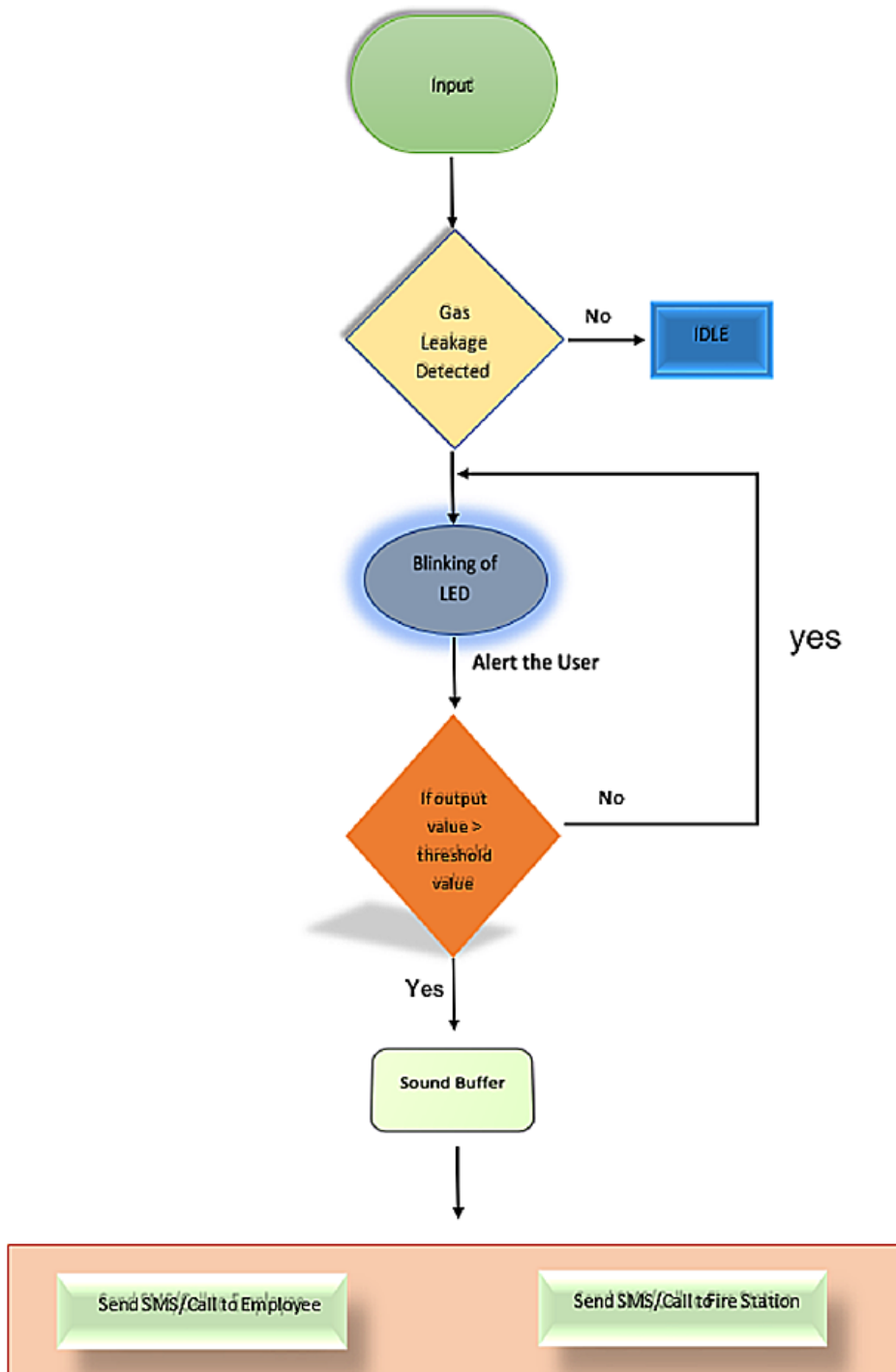
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Methodology	It is a well way to get rapid results in a short time.
FR-2	Impact	Sensor has excellent sensitivity combined with a quick fast response time , possible to get instantaneous results.
FR-3	sensitivity	Specialised of the gas in all similar systems
FR-4	WIFI -Module	Can communicate directly with industrial scientific, consumer technology that is web friendly with no use of shields or any peripherals.
FR-5	operation	The system be operated in android operating system.
FR-6	User Interface	Emergency call, message with application systems

## 4.2 Non-Functional requirements:

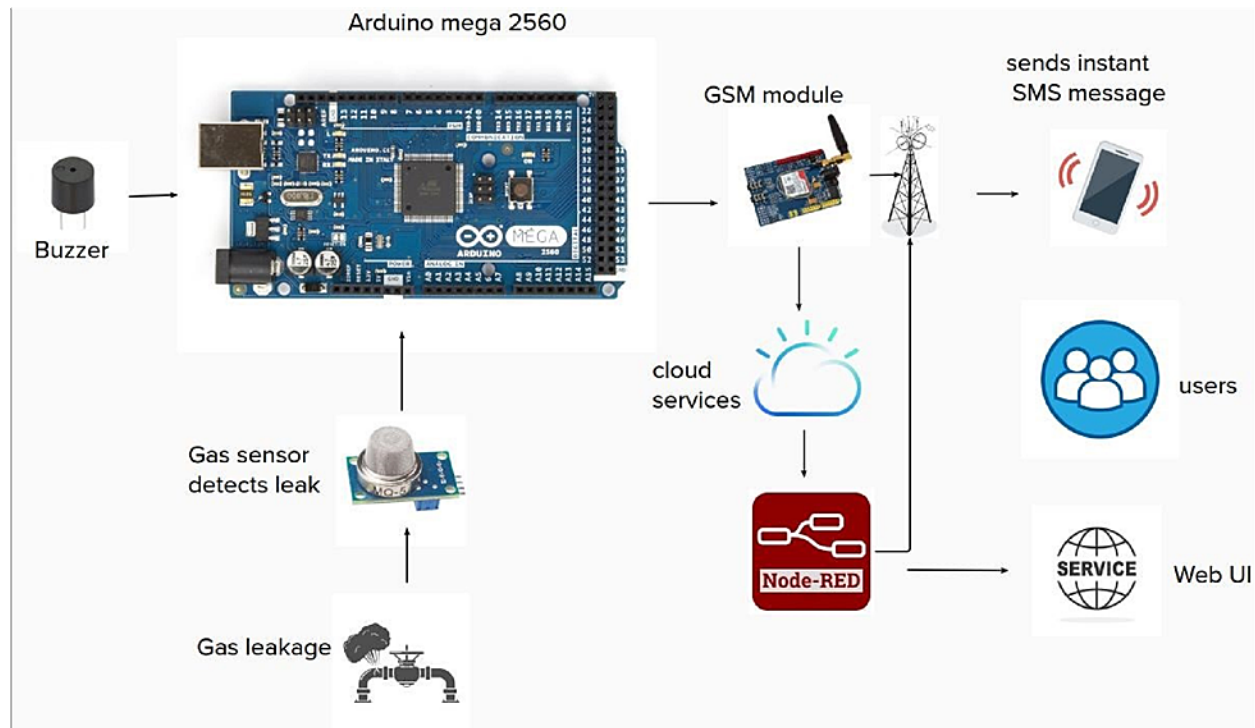
FR No.	Non-Functional Requirement	Description
NFR-1	<b>Usability</b>	It helps prevent the high risk of gas explosions
NFR-2	<b>Security</b>	The system should not display the homeowner personal information to anyone.
NFR-3	<b>Reliability</b>	Unsafe behaviour of personnel has the greatest impact on the probability of gas leakage.
NFR-4	<b>Performance</b>	Arduino response time will be fast.
NFR-5	<b>Availability</b>	The system should work 24 hours 7 days a week.
NFR-6	<b>Scalability</b>	The system interface should be easy and effective(user-friendly).

## 5. PROJECT DESIGN

### 5.1 Data Flow Diagrams:



## 5.2 Solution & Technical Architecture:



## 5.3 User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can create an account in the application provided.	I can access my account/ dashboard	High	Sprint-1
		USN-2	As a user, I registered using my Gmail.	I can receive confirmation email.	High	Sprint-1
		USN-3	As a user, I can successfully install the app.	I can register and access the dashboard.	Low	Sprint-2

	Login	USN-4	As a user, I can login using my Gmail and password easily.	The login process was easy and simple to access the dashboard.	High	Sprint-1
Customer (Web user)	Registration	WUSN-1	As a web user I can login to web dashboard just like a website.	I can register and access the dashboard.	High	Sprint-2
	Dashboard	WUSN-2	As a user I can view the alert/warning SMS in the web application.	I can login to the website using my login credentials	High	Sprint-2
Customer Care Executive		CCE-1	A customer care executive will always be available for the interaction with the customer to clarify the queries.	An executive will clarify the doubts and note down the complaints of the application if any.	High	Sprint-2
Administrator		ADMIN-1	I as an Admin can access and view the data or information provided by the application & can also check, analyse the threshold value of the gas.	The details of the gas leakage level of the gas are provided to the users through SMS when an alerting sound is received.	High	Sprint-1



## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation:

1. SPRINT PLAN
2. ANALYZE THE PROBLEM
3. PREPARE an ABSTRACT, PROBLEM STATEMENT
4. LIST A REQUIRED OBJECT NEEDED
5. CREATE A PROGRAM CODE AND RUN IT
6. MAKE A PROTOTYPE TO IMPLEMENT
7. TEST WITH THE CREATED CODE AND CHECK THE DESIGNED PROTOTYPE

### 6.2 Sprint Delivery Schedule:

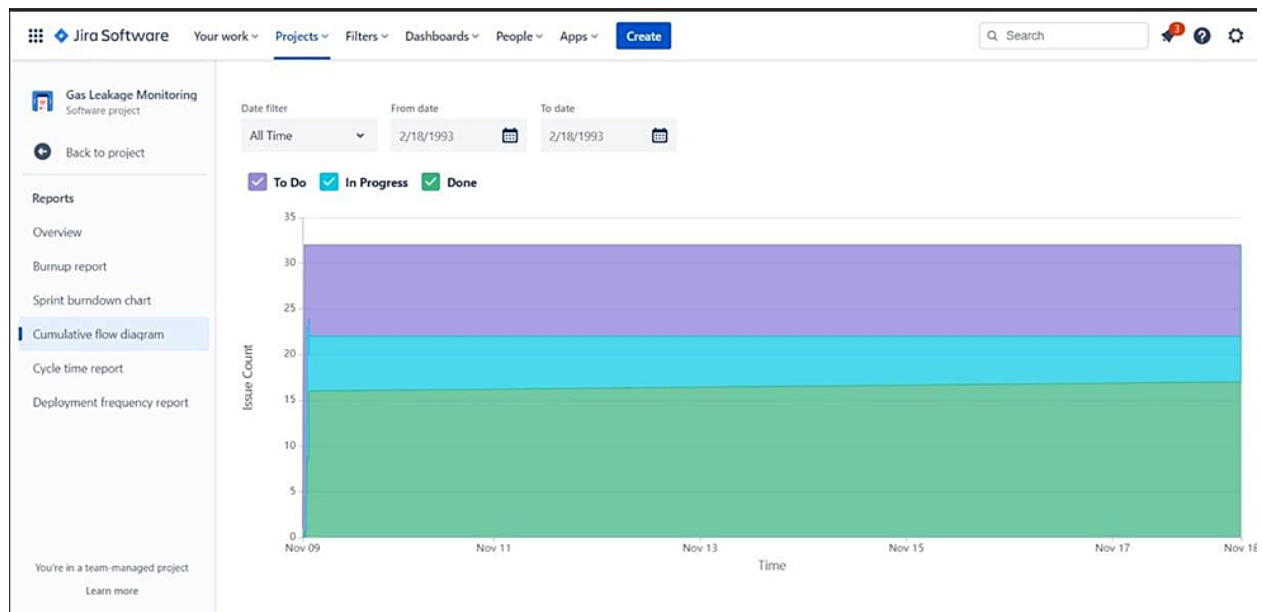
Sprint	Functional Requirement (Epic)	User Story	User Story / Task	Story Point	Priority
Sprint-1	Create	US-1	Create the IBM Cloud services which are being used in this project.	5	High

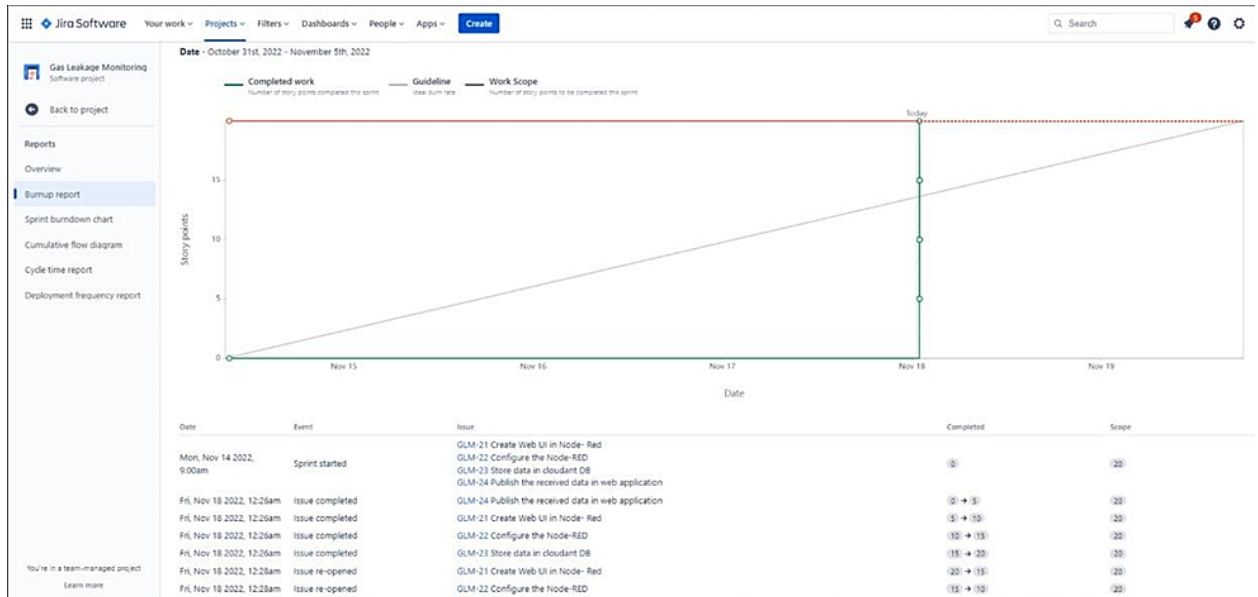
Sprint-1	Configure	US-2	Configure the IBM Cloud services which are being used in completing this project.	1	Medium
Sprint-1	Create	US-3	IBM Watson IoT platform acts as the mediator to connect the web application to IoT devices, so create the IBM Watson IoT platform.	1	Medium
Sprint-1	Configure	US-4	Configure 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 webapplicati on	5	High

## 6.3 Report from JIRA:





Jira Software

Gas Leakage Monitoring  
Software project

Back to project

Reports

Overview

Burnup report

Sprint burndown chart

Velocity report

Cumulative flow diagram

Cycle time report

Deployment frequency report

You're in a team-managed project  
Learn more

All issues in sprint have been completed

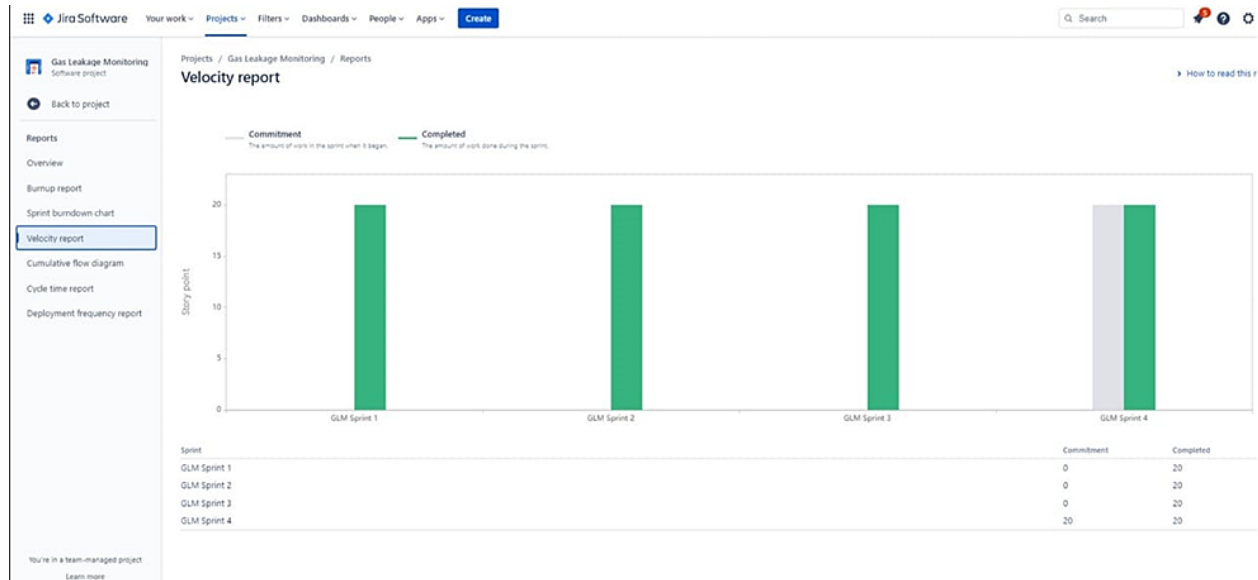
View in issue navy

Key	Summary	Issue type	Epic	Status	Assignee	Story points
GLM-21	Create Web UI in Node- Red	Task		DONE	SR	5
GLM-22	Configure the Node-RED	Task		DONE	SR	5
GLM-23	Store data in cloudant DB	Task		DONE		5
GLM-24	Publish the received data in web application	Task		DONE	SR	5

Issues completed outside of sprint

Key	Summary	Issue type	Epic	Status	Assignee	Story points
-----	---------	------------	------	--------	----------	--------------

No issues have been completed outside of the sprint



## 7. CODING & SOLUTIONING:

# Importing Required modules

import time

import sys

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 = "ioz5i8" # Organization ID

deviceType = "raspberrypi" # Device type

deviceId = "123456" # Device ID

authMethod = "token" # Authentication Method

authToken = "a-ioz5i8-dl5lboxjraw" #Replace the authtoken

# Tkinter root window

root = tk.Tk()

root.geometry('350x300') # Set size of root window

root.resizable(False, False) # 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=3000,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,
"authmethod": 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())
        data = {'gas_level' : gas_level}
        def myOnPublishCallback():
            print("Published Gas Level = %s ppm" % gas_level, "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()

```

## CODE:

```

File Edit Format Run Options Window Help
# Importing Required modules
import time
import sys
import wiop.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 = "0tus0f" # Organization ID
deviceType = "ESP32" # Device type
deviceId = "01" # Device ID
authMethod = "token" # Authentication Method
authToken = "Gowtham$nk18" # Replace the authToken

# Tkinter root window
root = tk.Tk()
root.geometry('350x300') # Set size of root window
root.resizable(False, False) # root window non-resizable
root.title('Gas Leakage Monitoring And Alerting System for Industries (PNT2022TMD4227)')

# 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())

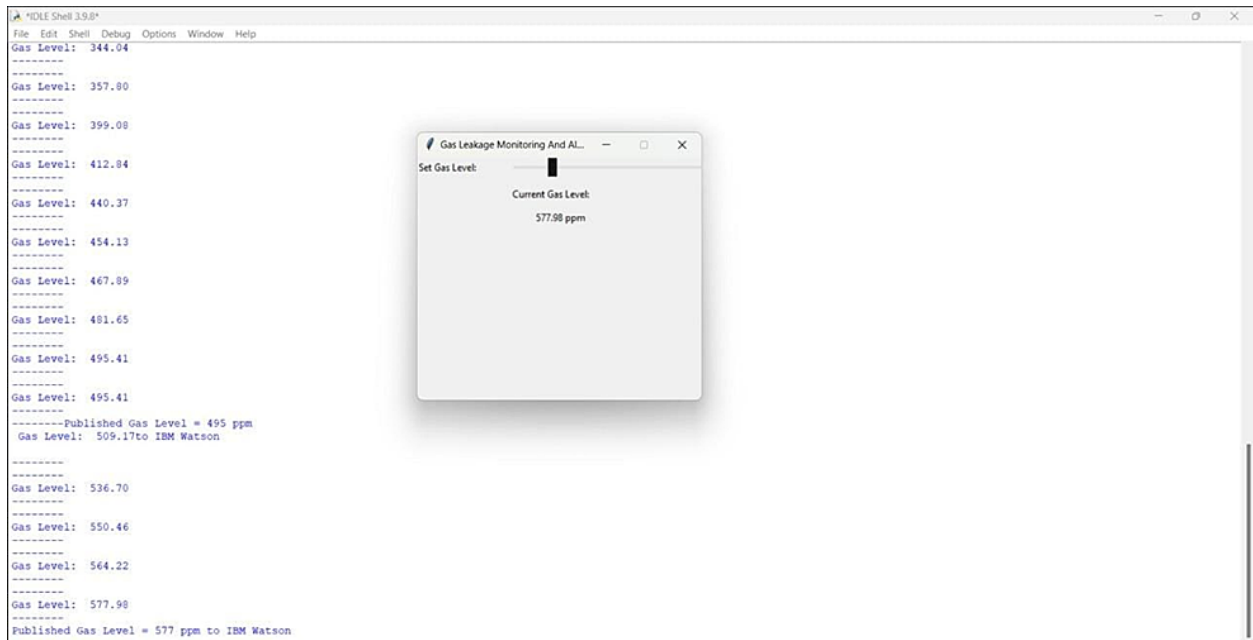
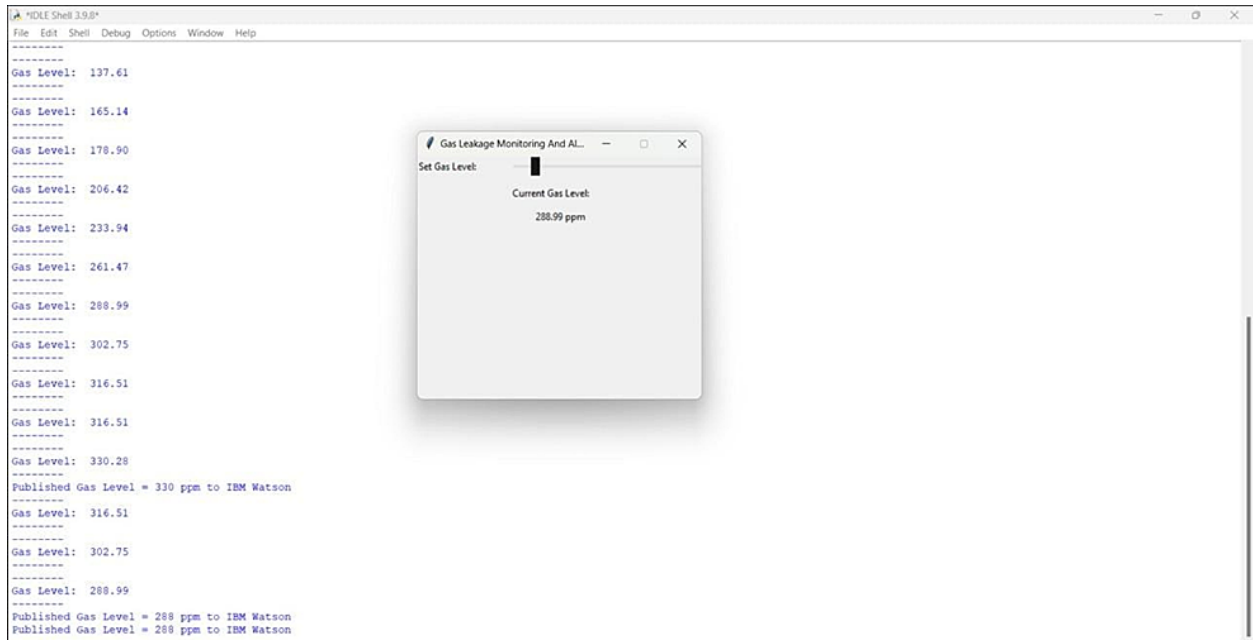
def slider_changed(event): # Event Handler for changes in sliders
    print('-----')
    print('Gas Level: {: .2f}'.format(current_gas.get()))
    print('-----')
    gas_label.configure(text=str(get_current_gas()) + " ppm") # Displays current gas level as label content

# 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')

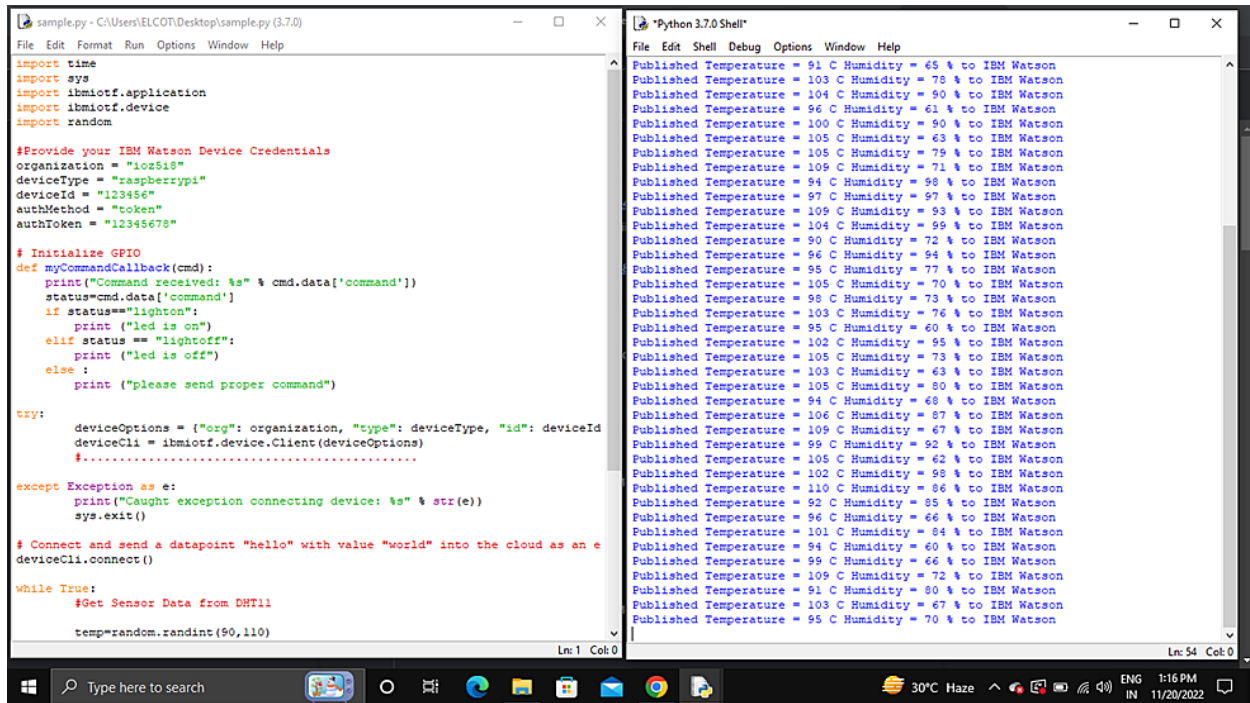
```



## OUTPUT:



## 8. Testing:



The image shows a Python script in a file editor and its execution output in a terminal. The script, named `sample.py`, is designed to connect to the IBM Watson IoT Platform and send sensor data. It includes imports for `time`, `sys`, `ibmiotf.application`, `ibmiotf.device`, and `random`. It defines organization, device type, device ID, authentication method, and token. A callback function `myCommandCallback` is defined to handle incoming commands like 'led on', 'led off', or 'lighton'. The script then initializes the device, connects to the cloud, and enters a loop where it generates random temperature and humidity data and sends it to the cloud as a datapoint.

```
sample.py - C:\Users\ELCOT\Desktop\sample.py (3.7.0)
File Edit Format Run Options Window Help

import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "ioz5i8"
deviceType = "raspberrypi"
deviceId = "123456"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    elif status == "lightoff":
        print ("led is off")
    else :
        print ("please send proper command")

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #.....

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

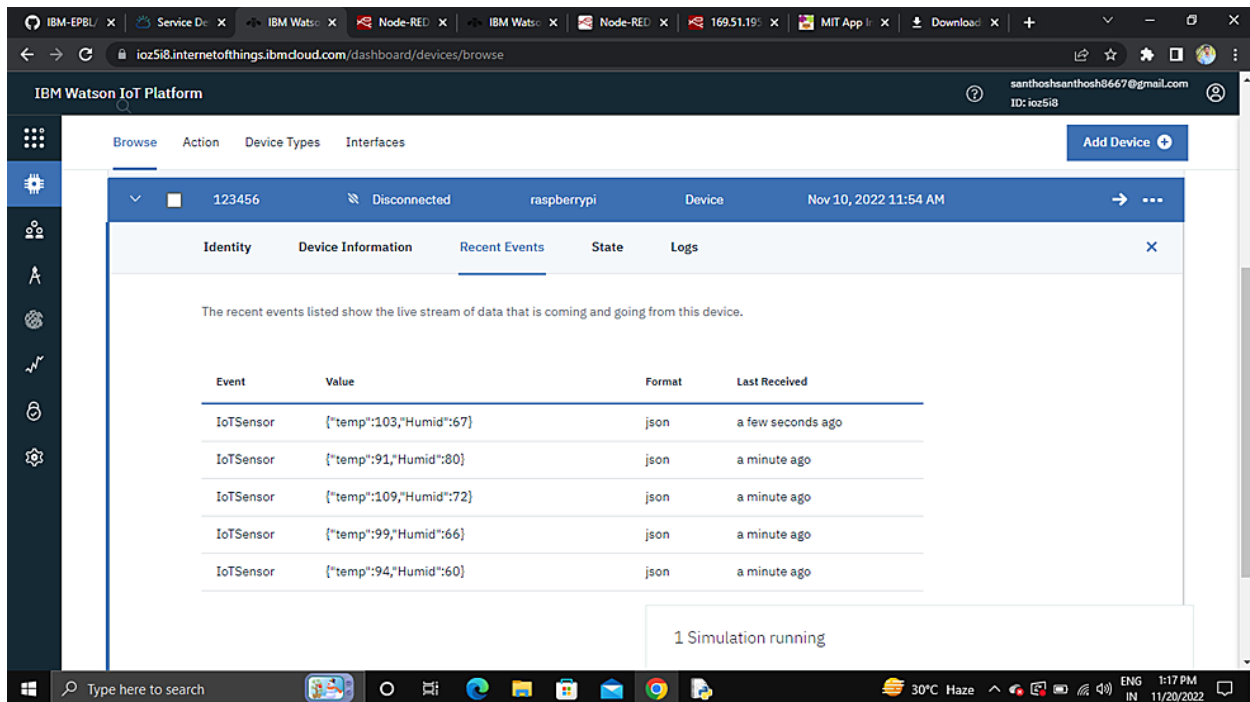
# Connect and send a datapoint "hello" with value "world" into the cloud as an e
deviceCli.connect()

while True:
    #Get Sensor Data from DHT11
    temp=random.randint(90,110)
```

The terminal output shows a continuous stream of data points being published to the IBM Watson IoT Platform:

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help

Published Temperature = 91 C Humidity = 65 % to IBM Watson
Published Temperature = 103 C Humidity = 78 % to IBM Watson
Published Temperature = 104 C Humidity = 90 % to IBM Watson
Published Temperature = 96 C Humidity = 61 % to IBM Watson
Published Temperature = 100 C Humidity = 90 % to IBM Watson
Published Temperature = 105 C Humidity = 63 % to IBM Watson
Published Temperature = 105 C Humidity = 79 % to IBM Watson
Published Temperature = 109 C Humidity = 71 % to IBM Watson
Published Temperature = 94 C Humidity = 98 % to IBM Watson
Published Temperature = 97 C Humidity = 97 % to IBM Watson
Published Temperature = 109 C Humidity = 93 % to IBM Watson
Published Temperature = 104 C Humidity = 99 % to IBM Watson
Published Temperature = 90 C Humidity = 72 % to IBM Watson
Published Temperature = 96 C Humidity = 94 % to IBM Watson
Published Temperature = 95 C Humidity = 77 % to IBM Watson
Published Temperature = 105 C Humidity = 70 % to IBM Watson
Published Temperature = 98 C Humidity = 73 % to IBM Watson
Published Temperature = 103 C Humidity = 76 % to IBM Watson
Published Temperature = 95 C Humidity = 60 % to IBM Watson
Published Temperature = 102 C Humidity = 95 % to IBM Watson
Published Temperature = 105 C Humidity = 73 % to IBM Watson
Published Temperature = 103 C Humidity = 63 % to IBM Watson
Published Temperature = 105 C Humidity = 80 % to IBM Watson
Published Temperature = 94 C Humidity = 60 % to IBM Watson
Published Temperature = 106 C Humidity = 87 % to IBM Watson
Published Temperature = 109 C Humidity = 67 % to IBM Watson
Published Temperature = 99 C Humidity = 92 % to IBM Watson
Published Temperature = 105 C Humidity = 62 % to IBM Watson
Published Temperature = 102 C Humidity = 98 % to IBM Watson
Published Temperature = 110 C Humidity = 86 % to IBM Watson
Published Temperature = 92 C Humidity = 85 % to IBM Watson
Published Temperature = 96 C Humidity = 66 % to IBM Watson
Published Temperature = 101 C Humidity = 84 % to IBM Watson
Published Temperature = 94 C Humidity = 60 % to IBM Watson
Published Temperature = 99 C Humidity = 66 % to IBM Watson
Published Temperature = 109 C Humidity = 72 % to IBM Watson
Published Temperature = 91 C Humidity = 80 % to IBM Watson
Published Temperature = 103 C Humidity = 67 % to IBM Watson
Published Temperature = 95 C Humidity = 70 % to IBM Watson
```



The image shows the IBM Watson IoT Platform dashboard. The top navigation bar includes links for Browse, Action, Device Types, and Interfaces. A sidebar on the left contains icons for various platform features. The main content area displays details for a specific device with ID 123456, which is currently disconnected. Below the device information, there is a section for 'Recent Events' showing a live stream of data. The events are listed in a table with columns for Event, Value, Format, and Last Received. A notification at the bottom indicates that 1 simulation is running.

IBM Watson IoT Platform

santhoshsanthosh8667@gmail.com  
ID: ioz5i8

Browse Action Device Types Interfaces

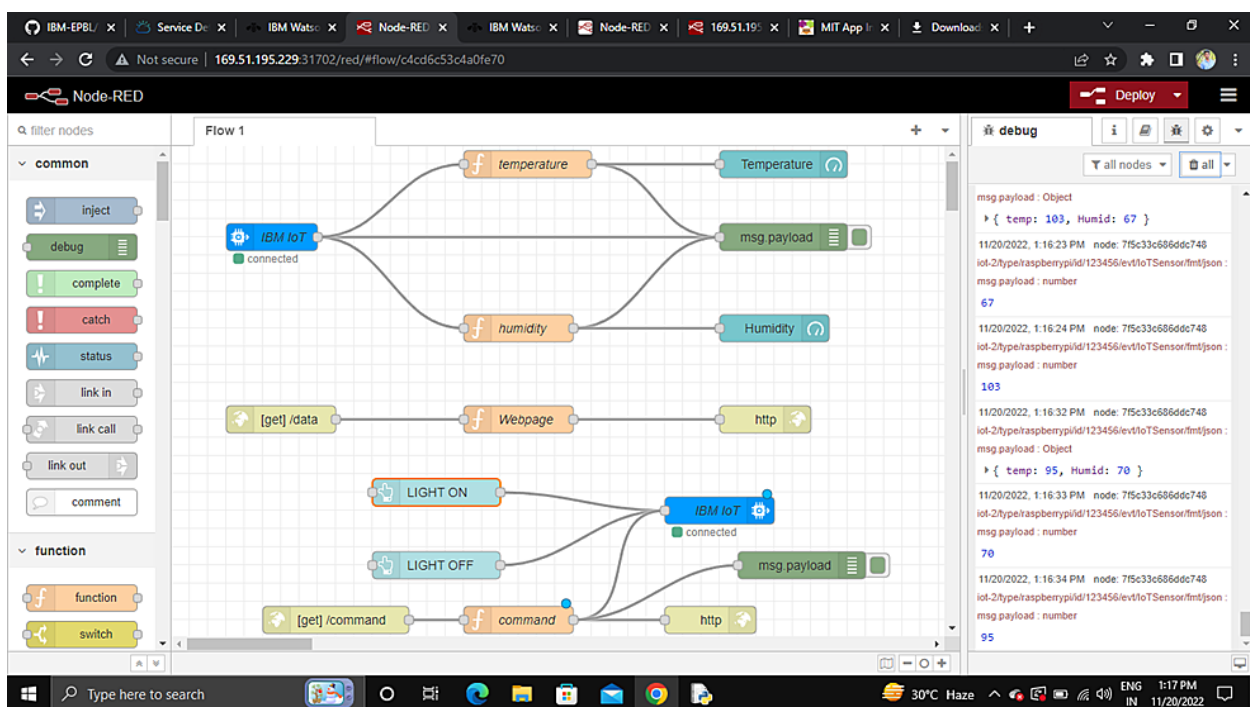
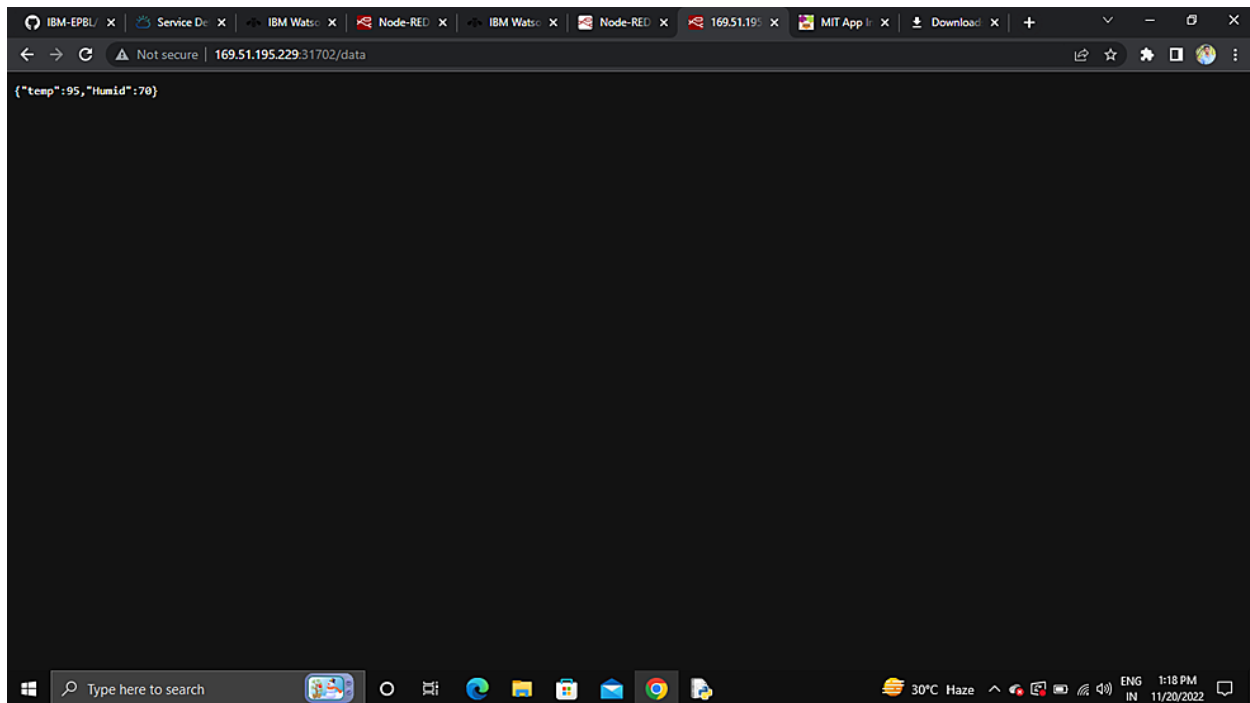
123456 Disconnected raspberrypi Device Nov 10, 2022 11:54 AM

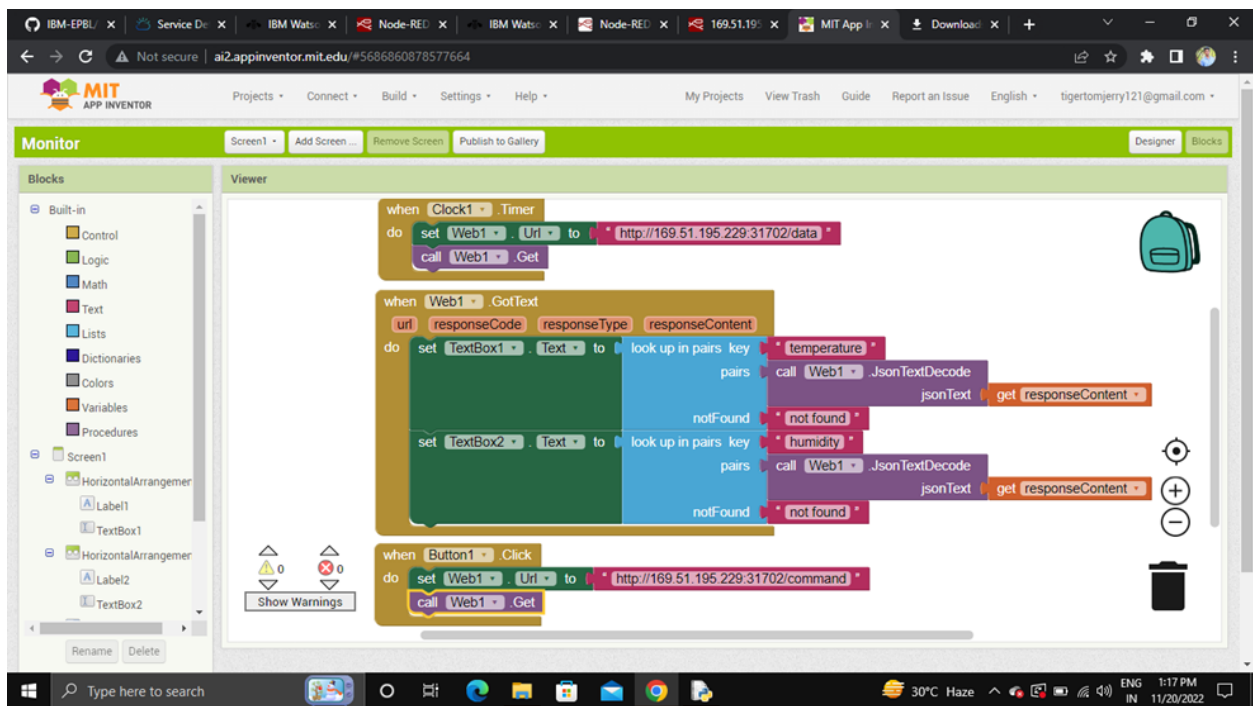
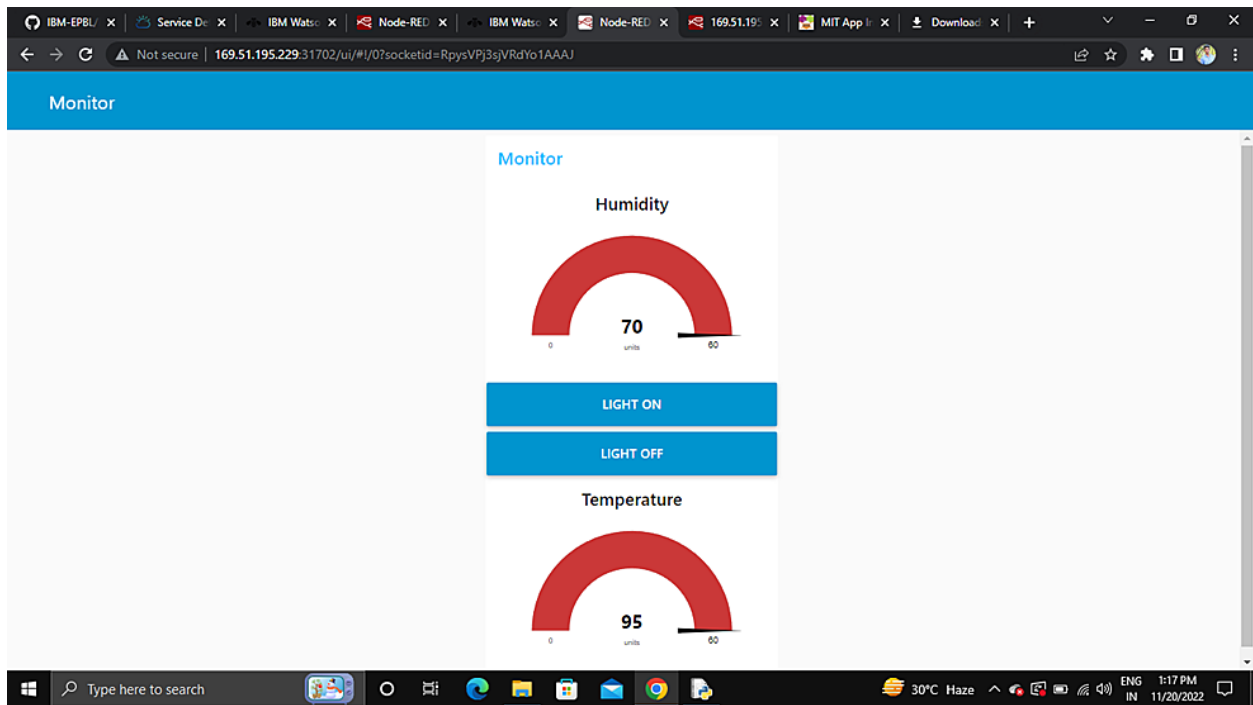
Identity Device Information Recent Events State Logs

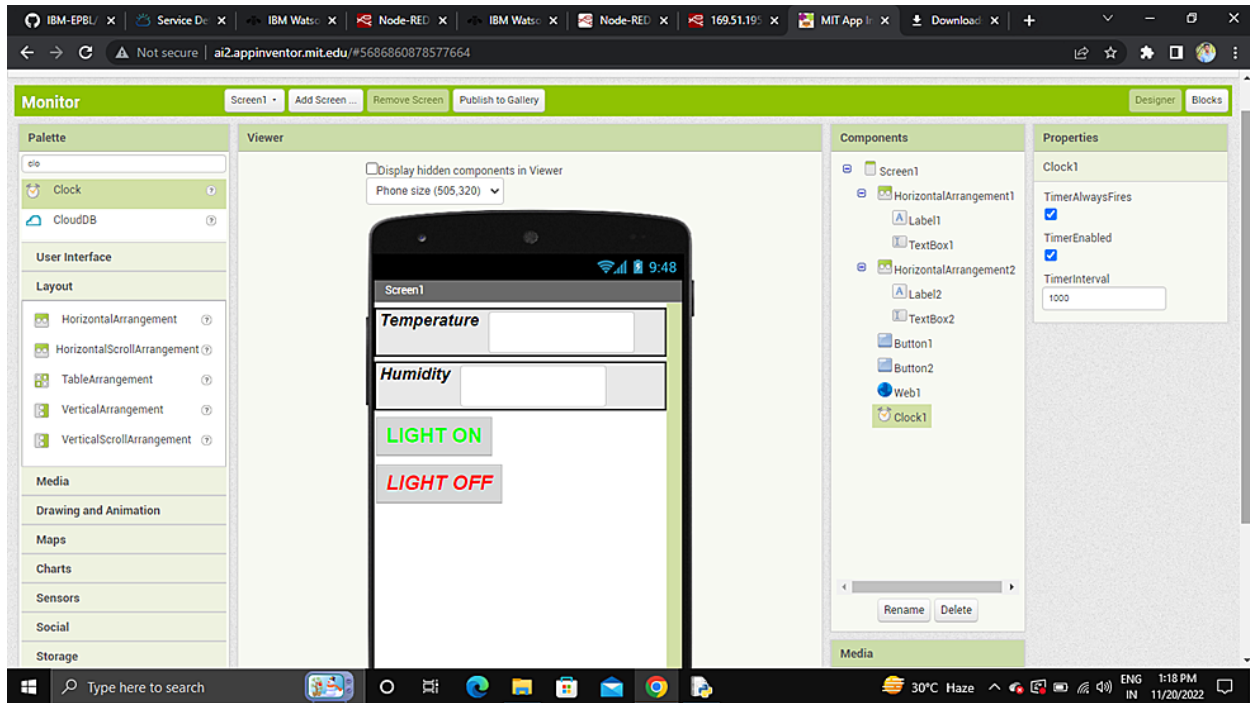
The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
IoTSensor	{"temp":103,"Humid":67}	json	a few seconds ago
IoTSensor	{"temp":91,"Humid":80}	json	a minute ago
IoTSensor	{"temp":109,"Humid":72}	json	a minute ago
IoTSensor	{"temp":99,"Humid":66}	json	a minute ago
IoTSensor	{"temp":94,"Humid":60}	json	a minute ago

1 Simulation running







## 9. Result:

The system can be taken as a small attempt in connecting the existing primary gas detection methods to a mobile platform integrated with IoT platforms. The gases are sensed in an area of 1m radius of the rover and the sensor output data are continuously transferred to the local server. 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.

## **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:**

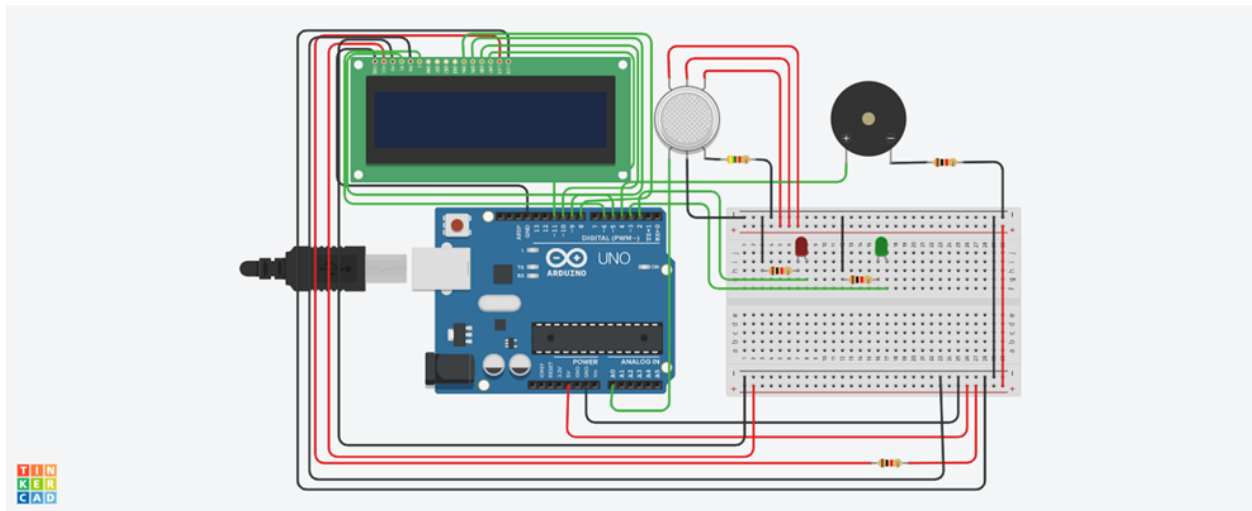
This gas leak detector system contains two features, this includes the SMS Gateway feature for only sending warning information regarding the gas leak to user, and the alarm for the warning alert. There is some improvement which can be applied for the future work, such as regarding the SMS Gateway, it need to enhance with feature such as notifying the user whenever the remaining credit balance is insufficient. Another thing which can be enhanced is regarding the sensor, the sensors in this module do not include somewhat notification for notifying the user whenever the sensor not working properly or not connected to the micro-controller for some cases, 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 Aurdino 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 = 2;
int greenled = 3;
int buzzer = 4;
int sensor = A0;
int sensorThresh = 100;
```

```
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.print(analogValue);  
  if(analogValue>sensorThresh)  
  {  
    digitalWrite(redled,HIGH);  
    digitalWrite(greenled,LOW);  
    tone(buzzer,1000,10000);  
    lcd.clear();  
    lcd.setCursor(0,1);  
    lcd.print("ALERT");  
    delay(1000);  
    lcd.clear();  
    lcd.setCursor(0,1);  
    lcd.print("PLEASE EVACUATE");  
    delay(1000);  
  }  
  else  
  {  
    digitalWrite(greenled,HIGH);  
    digitalWrite(redled,LOW);  
    noTone(buzzer);  
    lcd.clear();  
    lcd.setCursor(0,0);  
    lcd.print("SAFE");  
    delay(1000);  
    lcd.clear();  
    lcd.setCursor(0,1);  
    lcd.print("ALL CLEAR");  
  }  
}
```

```
    delay(1000);  
  }  
}
```

#### 13.4 GITHUB:

Link: <https://github.com/IBM-EPBL/IBM-Project-21040-1659770715>

#### 13.5 Project Demo Link :

1.Link:[https://drive.google.com/file/d/1IEBF9QZo0IJ3Egmh\\_plczpDOvLCAL2Oj/view](https://drive.google.com/file/d/1IEBF9QZo0IJ3Egmh_plczpDOvLCAL2Oj/view)

2.Link:<https://www.tinkercad.com/things/6kjOhED25pb-gas-leakage-monitoring-and-alerting-system/editel?sharecode=Gle6clcCKeR3foOhvoj1MLQykM0BeNvGsToLSkJDxNA>