MAHENDRAINSTITUTEOF TECHNOLOGY

(AUTONOMOUS)

Mahendhirapuri,Mallasamudram, Namakkal 637503

Office of the Controller of Examinations

A PROJECT REPORT

Submitted by

INDHUMATHI B	**(611619104042)**
JEEVAJOTHI D	**(611619104046)**
JEEVITHA C	**(611619104047)**
KANISHKA K	**(611619104048)**

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

MAHENDRA INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

MAHENDHIRAPURI, MALLASAMUDRAM, NAMAKKAL- 637 503

MAHENDRA INSTITUTE OF TECHNOLOGY

(Autonomous)

Mahendhirapuri, Mallasamudram, Namakkal DT- 637 503

Department of Computer Science and Engineering

BONAFIDE CERTIFICATE

Certified that this project report "**IoT Based Gas Leakage Monitoring and Alerting System"** is the Bonafide work of our team **"INDHUMATHI B(611619104042), JEEVAJOTHI D (611619104046), JEEVITHA C(611619104047), KANISHKA K (611619104048)"**who carried out the project work under my supervision.

……	**……**
SIGNATURE	**SIGNATURE**
Dr. J. STANLY JAYAPRAKASH	**Mr.K. MEIYALAKAN**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Professor	**Assistant professor**
Department of CSE	**Department of CSE**
Mahendra Institute Technology	**Mahendra Institute of Technology**
Namakkal - 637 503	**Namakkal - 637 503**

Submitted for the end semester viva voce examination held on ……..

……..	……..
Internal Examiner	External Examiner

MAHENDRA INSTITUTE OF TECHNOLOGY

(Autonomous)

Mahendhirapuri, Mallasamudram, Namakkal DT -637 503

Department of Computer Science and Engineering

CERTIFICATE OF PROJECT APPROVAL

This is to certify that the Project report titled **"IoT Based Safety Gas Leakage Monitoring and Alerting System"** is the approved record of work done by **"INDHUMATHI B (611619104042), JEEVAJOTHI D (611619104046), JEEVITHA C (611619104047), KANISHKA K (611619104048) "** in partial fulfilment for the award of the Degree of B.E Computer Science and Engineering during the academic year 2019-2023.

SUPERVISOR **HEAD OF THE DEPARTMENT**

Date: (Signature with seal)

Submitted for the end semester viva voce examination held on _____

Internal Examiner External Examiner

ACKNOWLEDGEMENT

We would like to take this opportunity to say our thanks to the people who have helped us make this project a reality.

We wish to express our sincere thanks to our honorable chairman, **Shri. M.G.Bharath Kumar B.Ed., M.A., M.I.S.T.E.,** of our Educational trust, Kalipatty and the Managing Directors **Er.Ba.Mahendiran B.E.,** and **Er.Maha Ajay Prasad B.E.,** providing an extraordinary infrastructure.

We would like to express our sincere thanks to **Dr.T.Elango M.E., Ph.D.,** the principal of our college, for their kind encouragement and blessings to do this project.

We also thank **Dr.J.Stanly Jayaprakash M.E., Ph.D.,** Head of the Department, Department of Computer Science and Engineering for the encouragement, valuable suggestions and support in doing this project.

We would like to thank our internal guide **Mrs.C.Gayathri.M.E,(Ph.D).,** Department of Computer Science and Engineering for the kind co-operative and support rendered in making our project as success.

We would like to say our sincere thanks to all other faculties, Department of Computer Science and Engineering for their active and kind guidance and advices for our project.

Above all we would like to express my sincere gratitude and thanks to our parents for their valuable comments and suggestions for making success.

CONTENTS

ABSTRACT

Leakage of any kind of gas has been a concern in recent years, whether it is in a residential setting, a business, a cafe, or a canteen. In this paper development of an IoT based gas wastage monitoring, leakage detecting and alerting system is proposed. This paper elaborates design such an intelligent system that will help save gas and smartly prevent accidents. The system needs to be integrated with the cooker. The technology includes ultrasonic sensors that determine if the cooker is being utilized for cooking purposes or not. If it is discovered that the cooker is not in use, the system uses an automatic switching off mechanism to cut off the gas supply. The moment gas leakage will probably be recognized, users will be informed via SMS through GSM, and so that user can solve the issue as soon as possible. The system will monitor flame and fire through flame sensor. When a fire is detected, the buzzer begins to sound. Aside from that, the system also has a cloud storage capability. The usage of gas for each user each day may be tracked with the aid of this cloud storage solution. At the end of the day, this procedure will assist in detecting per user natural gas usage. The system has been tested and it is able to monitor gas wastage, leakage and send a SMS to the user. The resulting performance indicated its effectiveness toward saving a significant portion of the wasted gas in domestic

1. INTRODUCTION

1.1 Project Overview:

This project helps the industries in monitoring the emission of harmful gases. In several areas, the integration of gas sensors helps in monitoring the gas leakage. If in any area gas leakage is detected the admins will be notified along with the location. In the web application, admins can view the sensor parameters.

1.2 Purpose:

Inhaling concentrated gas can lead to asphyxia and possible death.To overcome these disasters, we designed a system for monitoring and alerting the leakage of those harmful gases. This makes the industrialists get rid of the fear of any disasters caused by the gases.

2. LITERATURE SURVEY

2.1 Existing Problem:

The number of sensors is unpredictable and the positioning of equipment is improper and also the affordable of the system is high and the systems are sometimes causing heavy disasters.

2.2 References:

i. Bing Han, Qiang Fu, Hanfang Hou, 'Methane Leakage Monitoring Technology For Natural Gas Stations and Its Application', IEEE 5th International Conference on Computer and Communications,2001.

Shruthi Unnikrishnan,1 Mohammad Razil, Joshua Benny, Shelvin

Varghese and C.V. Hari, 'LPG Monitoring And Leakage Detection System',
Department of Applied Electronics and Instrumentation Engineering,
Rajagiri School of Engineering and Technology, Rajagiri Valley, Kakkanad,
Kochi, India. J.Vijayalakshmi, Dr.G.Puthilibhai, S.R.Leoram Siddarth,
'Implementation Of Ammonia Gas Leakage Detection & Monitoring System Using Internet Of Things', West Tambaram, Chennai. Makiko Kawada, Tadao Minagawa, Eiichi Nagao, Mitsuhito Kamei,
Chieko Nishida and Koji Ueda, 'Advanced Monitoring System For Gas Density Of GIS', Mitsubishi Electric Corporation.

2.3 **Problem statement definition:**

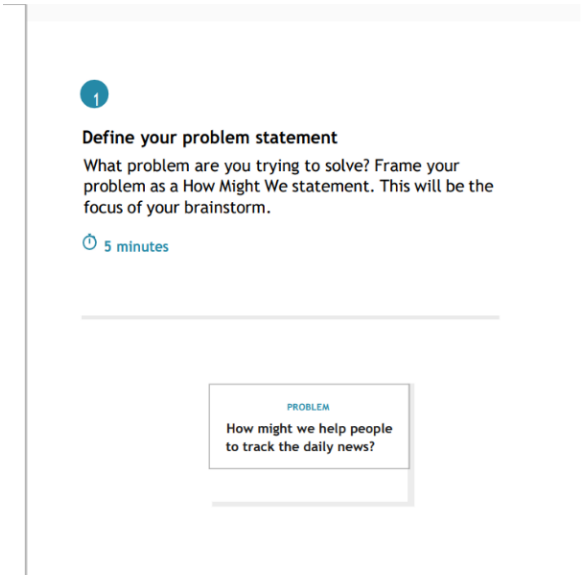
Since the number of sensors is unpredictable, the industrialists feel in secure in handling the gases. Also, the cost price of the products and the complications in installing the systems are high. This makes the customers feel disappointed sometimes.

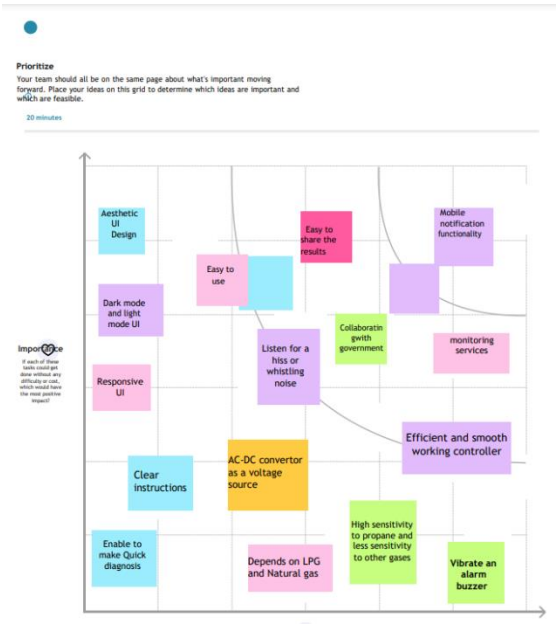
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:

Example:-

3.2 Ideation & Brainstorming:



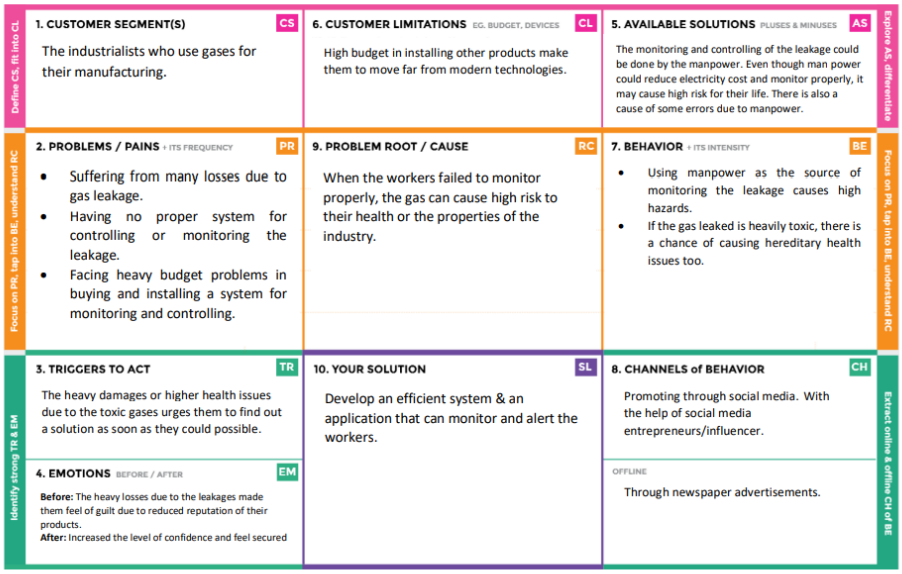


3.3 Proposed Solution:

S.No.	Parameter	Description
	Problem Statement	Develop an efficient system & an application that can monitor and alert the users(workers)
	Idea / Solution description	This product helps the industries in monitoring the emission of harmful gases In several areas, the gas sensors will be integrated to monitor the gas leakage. If in any area gas leakage is detected the admins will be notified along with the location In the web application, admins can view the sensor parameters.
	Novelty / Uniqueness	Fastest alerts to the workers User friendly
	Social Impact / Customer Satisfaction	Cost efficient Easy installation and provide efficient results Can work with irrespective of fear
	Business Model (Revenue Model)	The product is advertised all over the platforms. Since it is economical, it even helps small scale industries from disasters.As the product usage can be understood by everyone, it is easy for them to use it properly for their safest organization.
	Scalability of the Solution	Since the product is cost-efficient, it can be placed in many places in the industry. Even when the gas leakage is more, the product senses the accurate values and

		alerts the workers effectively.

3.4 Problem Solution Fit:



4.REQUIREMENT ANALYSIS

4.1Functional Requirement:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Visibility	The level of gas can be monitored by users if there is any leakage, alerts can be sent through messages.
FR-2	User Reception	The data like the level of gas can be sent through messages
FR-3	User Understanding	The user can monitor the level of gas with the help of the data. If there is an increase in gas level, then the alert will be given. They also get notified by the alert.

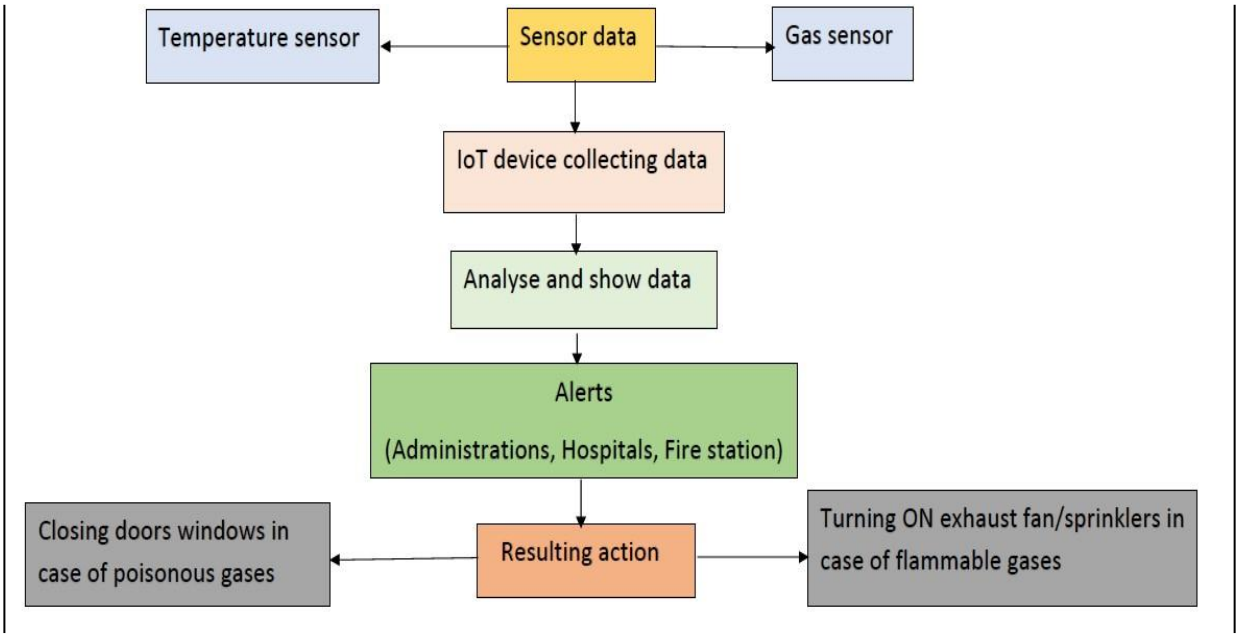
FR-4	User Convenience	Through messages we can easily get data of gas level and in case of gas leakage, it can directly send notifications to nearby police stations and hospitals.
FR-5	User Performance	When the user gets notified, he could turn on the exhaust fan/sprinkler.

4.2 Non-Functional Requirement:

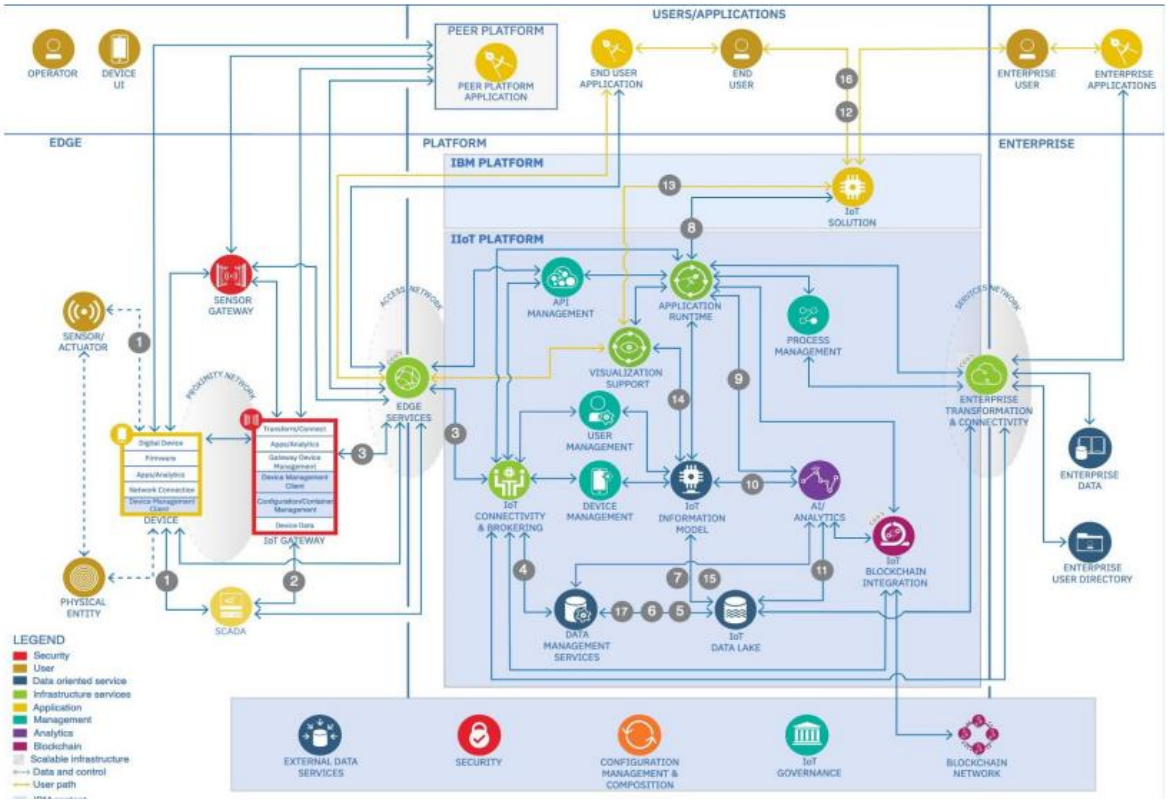
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	It updates the data regularly as well as protects the workers.
NFR-2	Security	As a result of emergency alert, we can be able to protect both humans and properties.
NFR-3	Reliability	Can be able to provide accurate values. It might have a capacity to recognize the smoke accurately and does not give a false
NFR-4	Performance	Sprinklers and exhaust fans are used in case of emergency.
NFR-5	Availability	It can be used for everyday; it includes day and nights.
NFR-6	Scalability	Sensors can be replaced every time it fails.

5.PROJECT DESIGN

5.1 Data Flow Diagram



5.2 Solution & Technical Architecture:



5.3User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	User can enter into the web application	I can access my account / dashboard	High	Sprint-1
		USN-2	Users can register their credentials like email id and password	I can receive confirmation email and click confirm	High	Sprint-1
	Login	USN-3	User can log in to the application by entering email and password	I can login to my account	High	Sprint-1
	Dashboard	USN-4	User can view the temperature	I can view the data given by the device	High	Sprint-2
		USN-5	User can view the level of gas	I can view the data given by the device	High	Sprint-2
Customer (Web user)	Usage	USN-1	User can view the webpage and get the information	I can view the data given by the device	High	Sprint-3
Customer	Working	USN-1	User act according to the alert given by the device	I can get the data work according to it	High	Sprint-3

		USN-2	User turns ON the exhaust fan/sprinkler when the leakage occurs	I can get the data work according to it	High	Sprint-4
Customer Care Executive	Action	USN-1	User solve the problems when someone faces any usage issues	I can solve the issues when someone fails to understand the procedure	High	Sprint-4
Administrator	Administration	USN-1	User stores every information	I can store the gained information	High	Sprint-4

6.PROJECT PLANNING AND SCHEDULING

6.1. Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Create	US-1	Create the IBM Cloud services which are being used in this project.	6	High
Sprint-1	Configure	US-2	Configure the IBM Cloud services which are being used in completing this project.	4	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.	5	Medium
Sprint-1	Create	US-4	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.	5	High
Sprint-2	Configure	US-1	Configure the connection security and create API keys that are used in the Node-RED service for accessing the IBM IoT Platform.	10	High
Sprint-2	Create	US-2	Create a Node-RED service.	10	High
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 IoT platform	7	High
Sprint-3	Configure	US-2	After developing python code, commands are received just print the statements which represent the control of the devices.	5	Medium
Sprint-3	Publish	US-3	Publish Data to The IBM Cloud	8	High
Sprint-4	Create	US-1	Create Web UI in Node- Red	10	High
Sprint-4	Configure	US-2	Configure the Node-RED flow to receive data from the IBM IoT platform and use Cloudant DB nodes to store the received sensor data in the cloudant DB	10	High

6.2 Sprint Delivery Schedule:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports From JIRA:

Reports from JIRA regarding sprint delivery

7. CODING AND SOLUTIONING

7.1 Feature 1

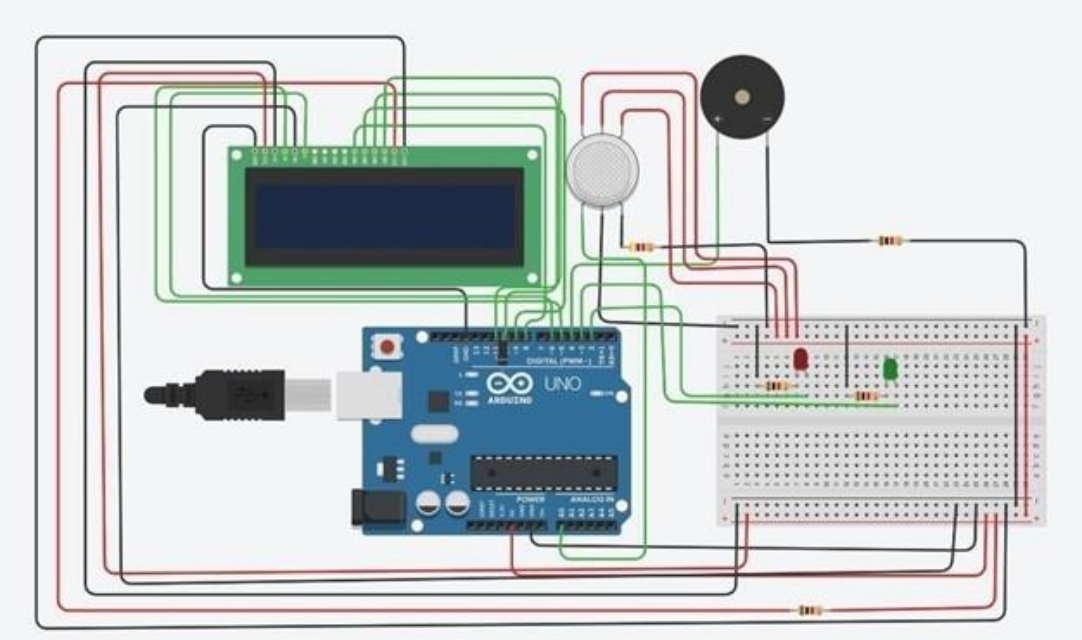
```
import time import sys

import ibmiotf.application

import ibmiotf.device import

random



#Provide your IBM Watson Device Credentials

organization = "vq4nsy" deviceType =

"PNT2022TMID47483" deviceId =

"PNT2022TMID47483DEVICEID" authMethod =

"token" authToken = "0vZoxRf8LrhADWKjb!"



# Initialize GPIO

def myCommandCallback(cmd):     print("Command received: %s" % cmd.data['command'])
status=cmd.data['command']     if status=="alarmon":         print ("Alarm is on")     elif (status == "alarmoff") :
print ("Alarm is off")     elif status == "sprinkleron":         print("Sprinkler is ON")

   elif status == "sprinklerOFF":

print("Sprinkler is OFF")

   #print(cmd)

  try:

          deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method":
authMethod, "auth-token": authToken}

          deviceCli = ibmiotf.device.Client(deviceOptions)
```

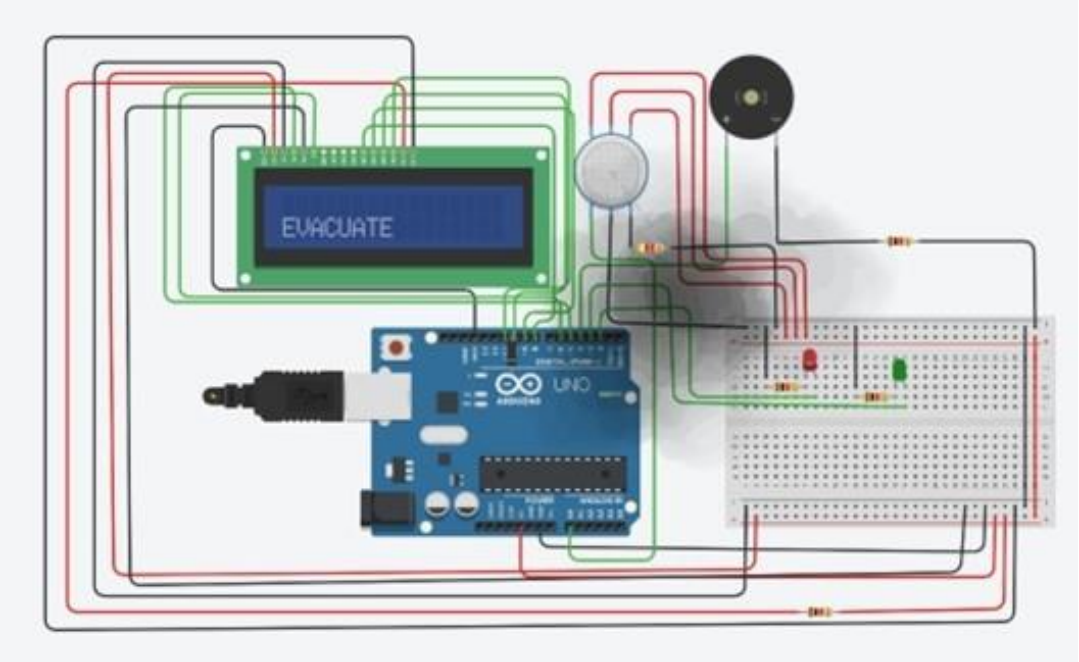
```
        #.....................................

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

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting"
10 times deviceCli.connect()

while True:
    #Get Sensor Data from DHT11

    temp=random.randint(0,100)
Humid=random.randint(0,100)      gas=random.randint(0,100)

    data = { 'temp' : temp, 'Humid': Humid, 'gas' : gas }
    #print data      def
myOnPublishCallback():
      print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "Gas_Level = %s %%"
%gas, "to IBM Watson")

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

    deviceCli.commandCallback = myCommandCallback

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

7.2 Feature 2: (Python Output)

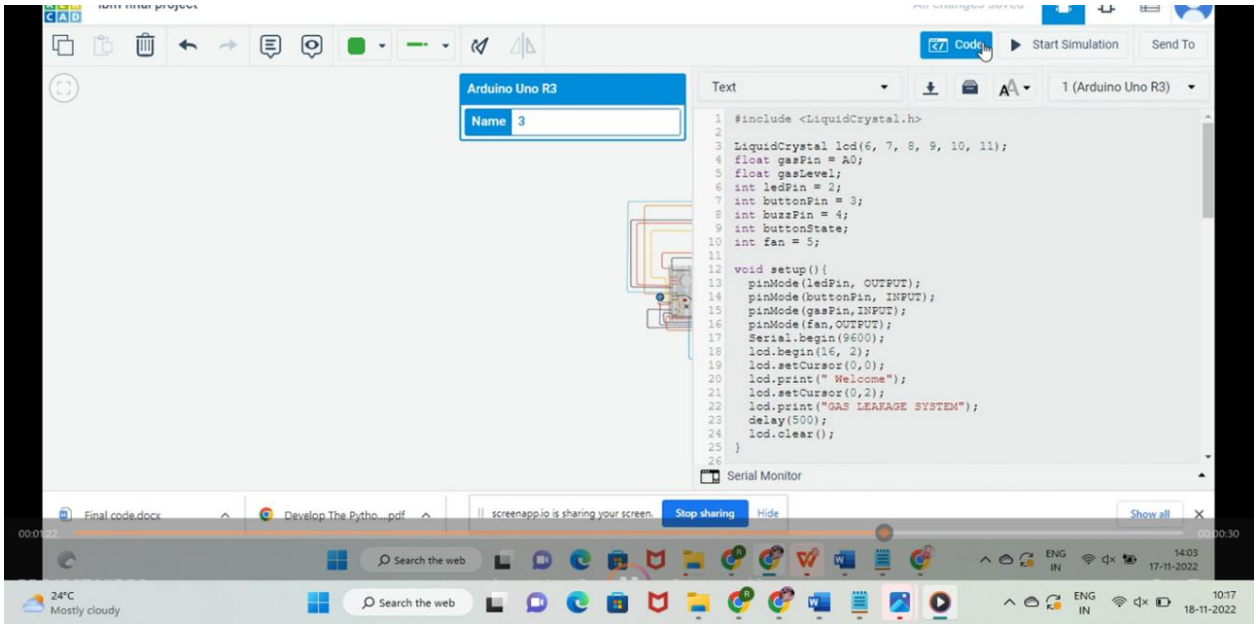
```
File  Edit  Shell  Debug  Options  Window  Help
Published Temperature = 72 C Humidity = 38 % Gas_Level = 93 % to IBM Watson
Published Temperature = 29 C Humidity = 58 % Gas_Level = 63 % to IBM Watson
Published Temperature = 71 C Humidity = 14 % Gas_Level = 87 % to IBM Watson
Published Temperature = 5 C Humidity = 32 % Gas_Level = 92 % to IBM Watson
Published Temperature = 51 C Humidity = 20 % Gas_Level = 82 % to IBM Watson
Published Temperature = 87 C Humidity = 10 % Gas_Level = 62 % to IBM Watson
Published Temperature = 35 C Humidity = 14 % Gas_Level = 19 % to IBM Watson
Published Temperature = 8 C Humidity = 28 % Gas_Level = 81 % to IBM Watson
Published Temperature = 69 C Humidity = 90 % Gas_Level = 50 % to IBM Watson
Published Temperature = 39 C Humidity = 0 % Gas_Level = 51 % to IBM Watson
Published Temperature = 88 C Humidity = 62 % Gas_Level = 27 % to IBM Watson
Published Temperature = 76 C Humidity = 89 % Gas_Level = 98 % to IBM Watson
Published Temperature = 99 C Humidity = 90 % Gas_Level = 12 % to IBM Watson
Published Temperature = 93 C Humidity = 36 % Gas_Level = 7 % to IBM Watson
Published Temperature = 98 C Humidity = 23 % Gas_Level = 40 % to IBM Watson
Published Temperature = 32 C Humidity = 72 % Gas_Level = 62 % to IBM Watson
Published Temperature = 55 C Humidity = 7 % Gas_Level = 80 % to IBM Watson
Published Temperature = 100 C Humidity = 74 % Gas_Level = 29 % to IBM Watson
Published Temperature = 64 C Humidity = 86 % Gas_Level = 13 % to IBM Watson
Published Temperature = 55 C Humidity = 5 % Gas_Level = 17 % to IBM Watson
Published Temperature = 72 C Humidity = 28 % Gas_Level = 37 % to IBM Watson
Published Temperature = 10 C Humidity = 54 % Gas_Level = 65 % to IBM Watson
Published Temperature = 30 C Humidity = 82 % Gas_Level = 82 % to IBM Watson
Published Temperature = 40 C Humidity = 95 % Gas_Level = 57 % to IBM Watson
Published Temperature = 28 C Humidity = 18 % Gas_Level = 17 % to IBM Watson
Published Temperature = 47 C Humidity = 66 % Gas_Level = 50 % to IBM Watson
Published Temperature = 58 C Humidity = 86 % Gas_Level = 50 % to IBM Watson
Published Temperature = 98 C Humidity = 19 % Gas_Level = 87 % to IBM Watson
Published Temperature = 12 C Humidity = 81 % Gas_Level = 40 % to IBM Watson
Published Temperature = 32 C Humidity = 79 % Gas_Level = 75 % to IBM Watson
Published Temperature = 37 C Humidity = 80 % Gas_Level = 24 % to IBM Watson
Published Temperature = 73 C Humidity = 59 % Gas_Level = 40 % to IBM Watson
Published Temperature = 51 C Humidity = 69 % Gas_Level = 34 % to IBM Watson
Published Temperature = 96 C Humidity = 13 % Gas_Level = 68 % to IBM Watson
Published Temperature = 28 C Humidity = 62 % Gas_Level = 7 % to IBM Watson
Published Temperature = 86 C Humidity = 69 % Gas_Level = 34 % to IBM Watson
Published Temperature = 48 C Humidity = 5 % Gas_Level = 40 % to IBM Watson
Published Temperature = 20 C Humidity = 51 % Gas_Level = 78 % to IBM Watson
Published Temperature = 60 C Humidity = 2 % Gas_Level = 91 % to IBM Watson
Published Temperature = 42 C Humidity = 86 % Gas_Level = 64 % to IBM Watson
Published Temperature = 95 C Humidity = 47 % Gas_Level = 99 % to IBM Watson
Published Temperature = 49 C Humidity = 16 % Gas_Level = 84 % to IBM Watson
Published Temperature = 59 C Humidity = 25 % Gas_Level = 66 % to IBM Watson
Published Temperature = 85 C Humidity = 100 % Gas_Level = 56 % to IBM Watson
Published Temperature = 65 C Humidity = 73 % Gas_Level = 13 % to IBM Watson
Published Temperature = 48 C Humidity = 38 % Gas_Level = 38 % to IBM Watson
```

7.3 Database Schema (If Applicable)




EVACUATE


Simulator time: 00:00:02
Code
Stop Simulation
Send To
Arduino Uno R3
Name 3
Basic
Search


Code
Start Simulation
Send To
Arduino Uno R3
Name 3
Text
1 (Arduino Uno R3)
#include <LiquidCrystal.h>
LiquidCrystal lcd(6, 7, 8, 9, 10, 11);
float gasPin = A0;
float gasLevel;
int ledPin = 2;
int buttonPin = 3;
int buzzPin = 4;
int buttonState;
int fan = 5;
void setup(){
pinMode(ledPin, OUTPUT);
pinMode(buttonPin, INPUT);
pinMode(gasPin,INPUT);
pinMode(fan,OUTPUT);
Serial.begin(9600);
lcd.begin(16, 2);
lcd.setCursor(0,0);
lcd.print(" Welcome");
lcd.setCursor(0,2);
lcd.print("GAS LEAKAGE SYSTEM");
delay(500);
lcd.clear();
}
Serial Monitor
Final code.docx
Develop The Pytho...pdf
screenapp.io is sharing your screen.
Stop sharing
Hide
Show all
Search the web
ENG
IN
14:03
17-11-2022
24°C
Mostly cloudy
Search the web
ENG
IN
10:17
18-11-2022

8.TESTING

8.1 Test Cases:

This report shows the number of test cases that have passed, failed, and untested

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
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

8.2 User Acceptance Testing:

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Gas Leakage Monitoring and Alerting System for Industries project at the time of the release to User Acceptance Testing (UAT).

Defect Analysis

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	20
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	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

9.RESULTS:

9.1 PERFORMANCE MATRICS

After this project performance can conclude that the detection of the LPG gas leakage is incredible in the project system. Applicable usefully for industrial and domestic purposes. In dangerous situations, we can save the life by using this system. An alert is indicated by the GSM module. A sensor node senses gas like CO2, oxygen, and propane. The estimated range of transmission and consumption of power is obtained. The simple procedures and Arduino UNO Micro controller area used to build the sensor.

10.ADVANTAGES AND DIASADVANTAGES:

Advantages:

1. Because of the very narrow 0.3 nm line width of the laser emission, there is no interference from other gases.
2. Response times are in the order 1 second.

Disadvantages:

1. Only one gas can be measured with each instrument.
2. When heavy dust, steam or fog blocks the laser beam, the system will not be able to take measurements.

11.CONCLUSION:

In danger situations we have to save the life by using this system. An alert is indicated by the GSM module. A sensor node senses gas like CO2, oxygen, propane. The estimated range of transmission and consumption of power is obtained. The simple procedures and Arduino UNO

Micro controller area used to build the sensor.

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.

Our 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

Source Code:

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



#Provide your IBM Watson Device Credentials
```

```
organization = "vq4nsy"
deviceType = "PNT2022TMID47483"
deviceId = "PNT2022TMID47483DEVICEID"
authMethod = "token"
authToken = "0vZoxRf8LrhADWKjb!"


# Initialize GPIO




def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="alarmon":
        print ("Alarm is on")
    elif (status == "alarmoff") :
        print ("Alarm is off")
    elif status == "sprinkleron":
        print("Sprinkler is OFF")
    elif status == "sprinkleron":
        print("Sprinkler is ON")
    #print(cmd)







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()


# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type
"greeting" 10 times
```

```
deviceCli.connect()


while True:
        #Get Sensor Data from DHT11

        temp=random.randint(0,100)
        Humid=random.randint(0,100)
        gas=random.randint(0,100)

        data = { 'temp' : temp, 'Humid': Humid, 'gas' : gas }
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "Gas_Level =
%s %%" %gas, "to IBM Watson")


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

        deviceCli.commandCallback = myCommandCallback


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

GitHub and Project Demo Link:

GitHub link: IBM-EPBL/**IBM-Project-29627-1660127854**