PROJECT BASED EXPERIENTIAL LEARNING PROGRAM

(NALAIYA THIRAN)

INDUSTRY- SPECIFIC INTELLIGENT FIRE MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

AKSHAYA H (2019105508)
DIVYAPRIYA D (2019105529)
JANANI K R (2019105539)
SOWMIYAA S U (2019105579)

TEAM ID: PNT2022TMID35457

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

COLLEGE OF ENGINEERING GUINDY
(ANNA UNIVERSITY)

1. INTRODUCTION:

1.1 PROJECT OVERVIEW:

Fire Alarm System is designed to alert us to an emergency so that we can take action to protect ourselves, staff and the general public.Fire alarms are found in Offices, Factories, and public buildings,they are a part of our everyday routine but are often overlooked until there is an emergency at which point, they might just save our lives.we are going to build a fire alarm using multiple sensors.This project will send an alert whenever it detects fire and smoke.It has also connected with a buzzer and an LED which will act as a audio and visual indication for alert.

1.2 PURPOSE:

The primary purpose of fire alarm system is to provide an early warning of fire so that people can be evacuated & immediate action can be taken to stop or eliminate of the fire effect as soon as possible. Alarm can be triggered by using detectors or manual call point (Remotely)

A complete fire management system ensures legal compliance and protection of lives and assests.

2. LITERATURE SURVEY:

2.1 EXISTING PROBLEM:

When fire alarm panels are in trouble condition, it can be difficult to find the root cause of the problem. Trouble signals occur due to ground faults, circuit problems, battery faults, or other failures within the system. Device failures include smoke detectors above our heads or other devices that can fail out there can cause a trouble from dust and contamination. The sensors inside the detectors can be a problem. Addressable devices can self-diagnose problems and tell you that there's a problem. That would be one reason for a trouble

2.2 REFERENCE:

S. No	Author/ Publication Year	Title	Methodology	Advantages	Disadvantages
1.	Chen Yueping, Gan Fangcheng 2007	Design and Realization of Fire Alarm System Based on CAN Bus	In this fire alarm and monitoring system, intelligent and multi-sensor fire detectors and interface boards of CAN bus communication are considered as key core. Intelligent and multi-sensor fire detector is design by used microprocessor. Multi-fire detector is connected to CAN bus with communication interface boards	Its primary goal was to enable faster communication between electronic devices and modules in vehicles while reducing the amount of wiring (and the amount of copper) necessary.	It does not support a maximum number of nodes. It can connect only up to 64 nodes because of electrical loading.

			of CAN.		
2.	Huide Liu; Suwei Li; Lili Gao; Junfu Li 2010	CRT graphic display system of automatic fire alarm system based on GIS	CRT graphic display system of automatic fire alarm system based on GIS for the ready position fire locations, to select the best rescue route, about the fire scene surrounding environment plays a very important role in. This system is used for fire alarm and linkage of equipment management, control, monitoring, data, graphics, mainly displays text and other related fire and linkage information. Commonly used for large construction projects of fire control center.	CRT graphic display system intuitive an machine interface can be a comprehensive, real-time visual display of the fire scene, and the scene surrounding the situation. Help to fully grasp the situation on the ground rescue personnel quickly arrived at the fire scene.	The CRT's Gaussian beam profile produces images with softer edges that are not as sharp as an LCD at its native resolution

3.	Rajendra Prasad Behera; N. Murali; S.A.V. Satya Murty 2015	Development of Tele-Alarm and Fire Protection system using Remote Terminal Unit for Nuclear Power Plant	Tele-Alarm System (TAS) is used to ensure that the operator in every shift performs area surveillance of all the equipment to check their healthiness and identify maintenance requirements. During the surveillance, maintenance personnel operate area surveillance key that is provided at strategic locations such that no area in the plant is left unattended in every shift, as some of the equipment/buildings are unmanned normally. Water logging detectors are mounted near sumps, in pits, trenches/tunnels and other areas in the plant where water is likely to be accumulated and affect other systems.	The micro-controller based Tele-Alarm and Fire Protection System with distributed architecture have been successfully designed, developed and tested for 500MWe Prototype Fast Breeder Reactor being constructed at Kalpakkam project site. The hardware has been qualified to meet the environmental condition, EMI/EMC and seismic guidelines for a nuclear reactor.	The main drawback was poor wireless network sometimes and low power consumption.
4.	S. R. Vijayalakshmi; S. Muruganand 2018	Fire alarm based on spatial temporal analysis of fire in video	Fire alarm system is based on detection of fire from video acquisition input data. This is done with the help of digital image processing techniques and embedded vision. It is based on vision-based fire detection	The proposed technique can be incorporated with a fully automatic surveillance system monitoring open spaces of interest for early fire	Difficulties encountered in this research is the difficulty to determine the accuracy of the success of fire detection / fire danger. Future work development can be focused

			system. This approach integrates colour, spatial, temporal and motion information to locate fire regions in video frames.	warning system. The detection rate is increased by combine image processing technique along with sensing technique using sensors.	to generate a better formula to measure system performance and flicker into current system to achieve more robust fire detection.
5.	Mingyu Song; Wuxing Li; Xiaomin Zhang; Li Liu; Yanke Ci; Xushan Peng; Yongping Li; Haosong Chen 2020	Design of Distributed Factory Fire Alarm System	The distributed plant fire alarm system can quickly detect the fire and issue an alarm to reduce the damage caused by the fire. The fire alarm system is a control system that integrates signal detection, transmission, processing and control. It mainly completes the basic functions of fire, smoke and temperature module monitoring fire, and studies the multi-point communication of nRF2401 wireless transceiver module to realize the function of transmitting data at multiple points simultaneously.	It analyzes the characteristics of the existing intelligent monitoring system on the market, finds defects and deficiencies to improve, and draws advantages.	it cannot be used for non-carbon fires as well as only being able to detect fires that emits both the UV/IR radiation not individually

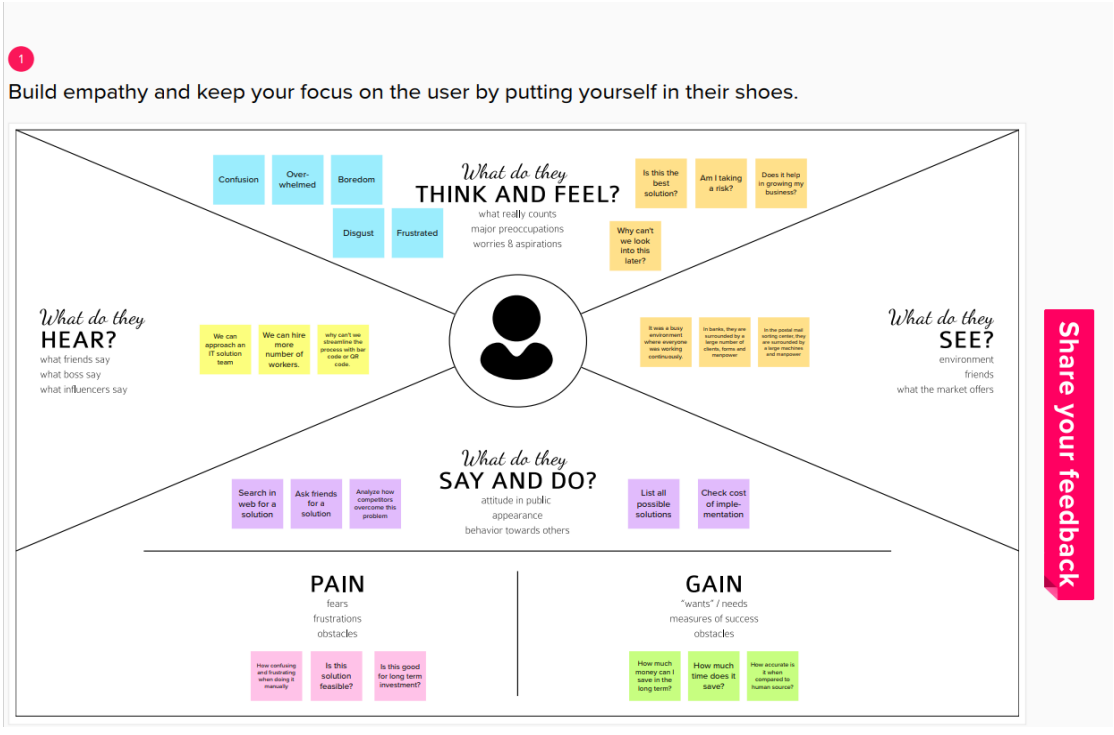
2.3 Problem statement :

A person who is an industrialist needs a rapid fire alarm system because the industrial material are easy fire catching. So to detect the fire accurately in the respective place and the machines can't get damaged . For that industrial specific fire alarm system must be needed.

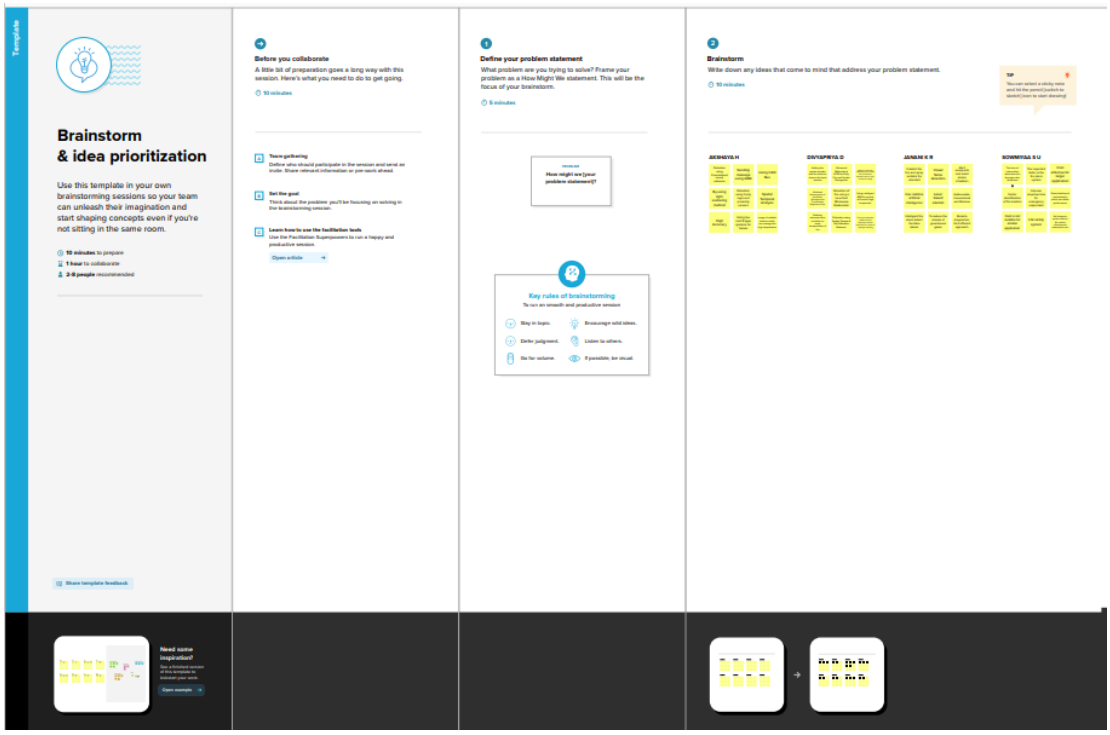
Who does the problem affect?	Working people, Proprietor
What are the boundaries of the problem?	Hospitals, Industries
What is the issue?	Very sensitive, which can leads to false alarm. The batteries need to be changed properly. Lack of maintenance
When does the issue occurs?	The materials used in industries may cause fire
Where is the issue occurring?	Industries
Why is it important that we fix the problem?	Safety measures, Provides 24/7 protection

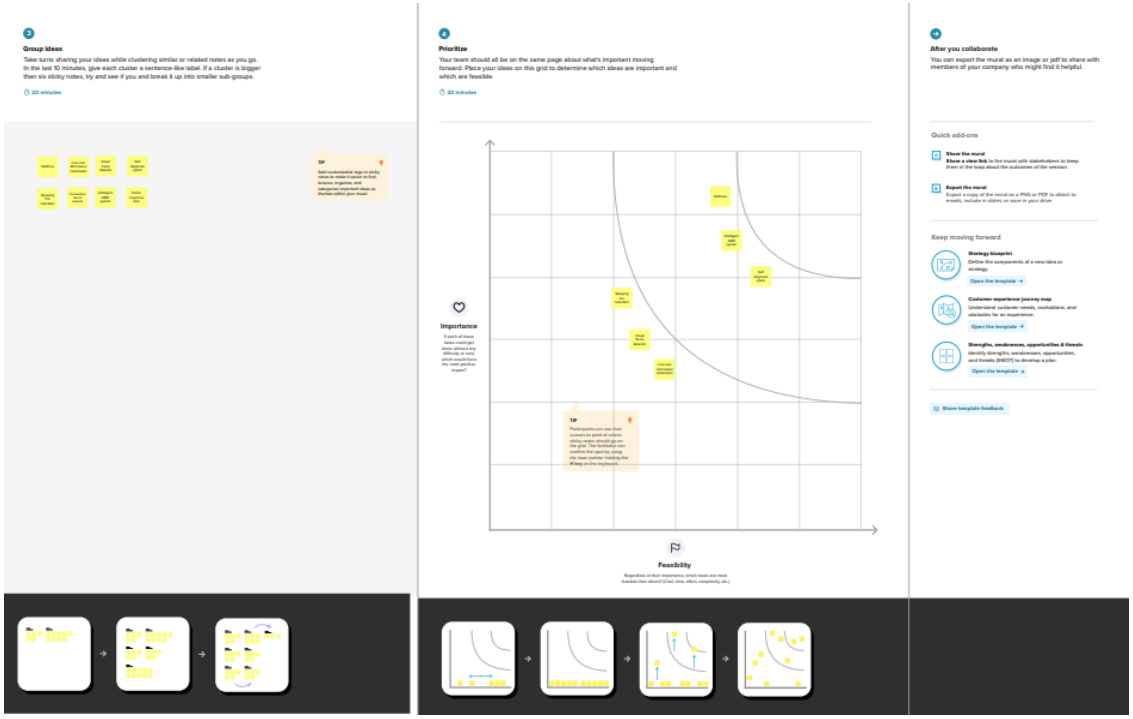
3. IDEATION & PROPOSED SOLUTION:

3.1 EMPATHY MAP CANAVAS:



3.2 IDEATION & BRAINSTROMING:



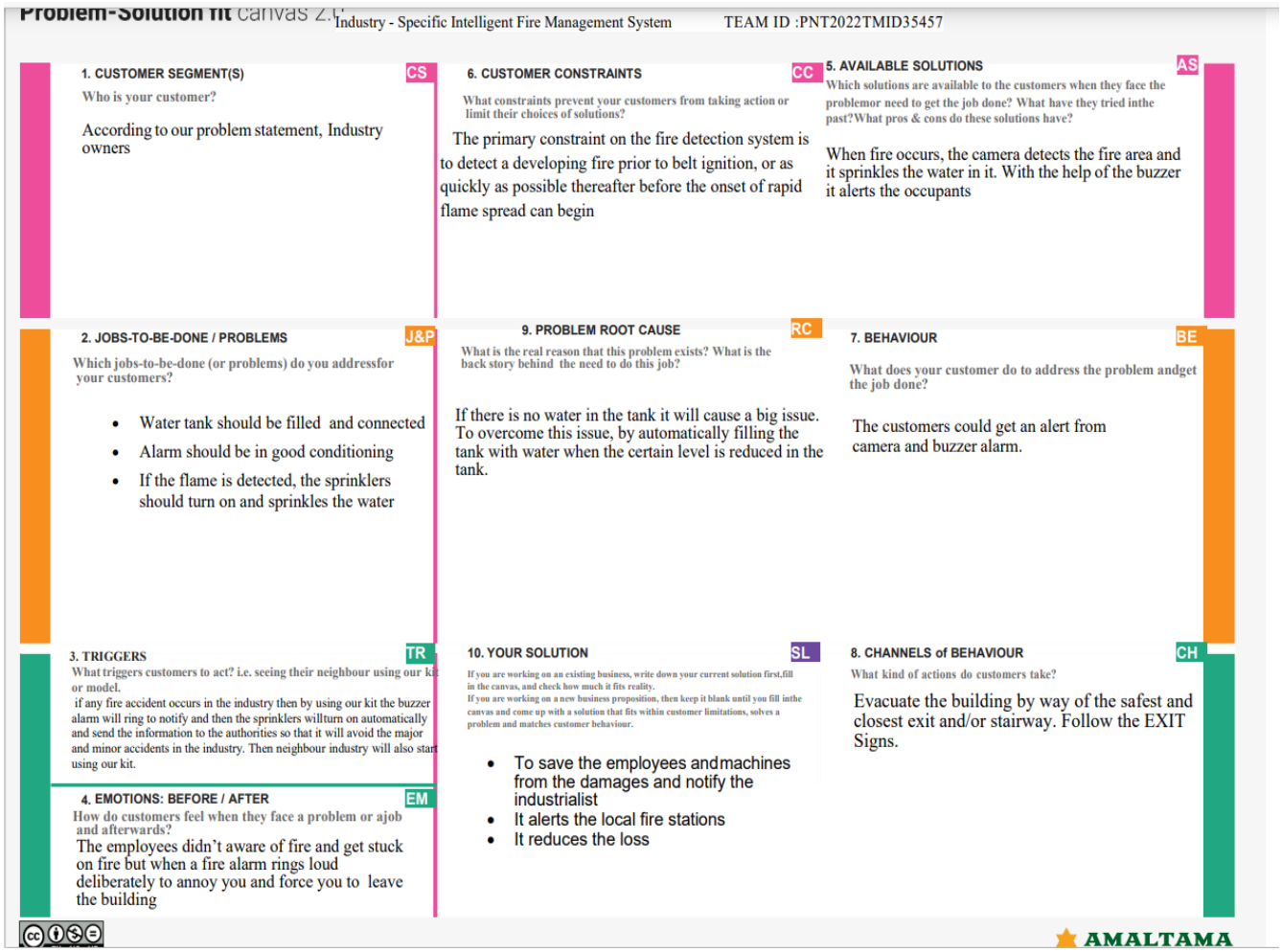


3.3 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Fire alarm systems are only effective if they can generate reliable and fast fire alerts with exact location of fire. There is a direct correlation between the amount of damage caused by fire and interventions time in various fire alarm systems. As the time of intervention decreases, the damage also decreases.
2.	Idea / Solution description	• The smart fire management system includes a gas sensor, flame sensor, humidity sensor and temperature sensors to detect any changes in the environment. • Based on the temperature readings and if any gases are present the exhaust fans are powered ON. • If any flame is detected the sprinklers will be switched on automatically. • Emergency alerts are notified to the authorities and Fire station.
3.	Novelty / Uniqueness	• Temperature Sensor • Flame sensor • Smoke sensor • Humidity sensor • Automatic water sprinkler • Buzzer • Cloud DB to store Data
4.	Social Impact / Customer Satisfaction	• Fire alarms save lives: A combination of smoke and heat detectors, sirens and bells, and strobe lights detect fires and alert building occupants, giving them ample time to evacuate in an orderly fashion.

		• Fire alarms reduce property loss: Monitored fire alarm systems automatically notify emergency responders and fire trucks dispatch to your location without delay. • To avoid the fire accidents that happen in the industries. • Alerts the local fire department
5.	Business Model (Revenue Model)	Less building damage means shorter downtime until you can reopen for business. This cuts your losses from the fire even more, allowing you to return to business as usual before long.
6.	Scalability of the Solution	• IBM Platform • Sensors • Python

3.4 PROBLEM SOLUTION FIT:



4.REQUIREMENT ANALYSIS:

4.1 FUNCTIONAL REQUIREMENT:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Gmail Registration through certain app
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Monitoring	As a user I ensure that the battery fully charged
FR-4	Detection	Using smoke sensor we can detect the smoke and we can perform the certain actions
FR-5	Alarm system	With the help of buzzer system we can alert the occupants

FR-6	Actions	Once received alert signal, the sprinkler sprinkles the suitable retardant

4.2 NON FUNCTIONAL REQUIREMENT:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Simplest user interface for ease of use
NFR-2	**Security**	It provides protection to our own life and equipment in the industry
NFR-3	**Reliability**	It quickly respond and perform certain action when the fire detect
NFR-4	**Performance**	The sprinkler sprinkles the suitable retardant at the early stage
NFR-5	**Availability**	Provide early warning and sensitive to visual particle of smoke
NFR-6	**Scalability**	The performance of the system depends upon application

5.PROJECT DESIGN:

5.1 DATA FLOW DIAGRAM:

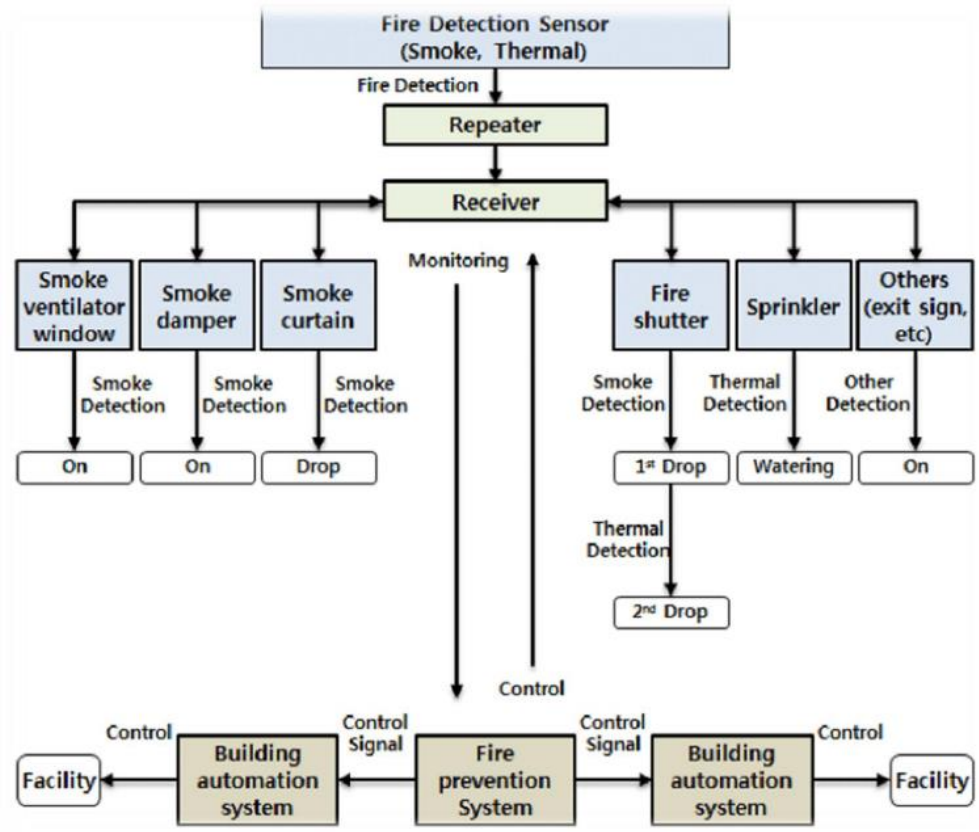
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

FLOW:

❖ High-Rise and Large Office Building has potential elements of fire occurrence, it expects casualties and property damage in case of fire.

❖ For this reason, buildings build fire prevention system which included in the BAS(Building Automation System). But it operates separately and remains simple monitoring and control. Therefore, this research introduces the integrated platform for fire suppression which is developed with reviews of BAS and fire prevention systems in Intelligent Building.

❖ As a result, the proposed integrated platform for fire suppression consists of the fire detection algorithm using temperature data, the occupancy detecting algorithm using RFID, CCTV and PIR(Pyroelectric Infrared Radial) sensor, the fire suppression algorithm using HVAC system, and connections with Building Management System in BAS for fire situation.



5.2 SOLUTION & TECHNICAL ARCHITECTURE:

SOLUTION ARCHITECTURE:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- ❖ Find the best tech solution to solve existing business problems.
- ❖ Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders
- ❖ Define features, development phases, and solution requirements.

- ❖ Provide specifications according to which the solution is defined, managed, and delivered.

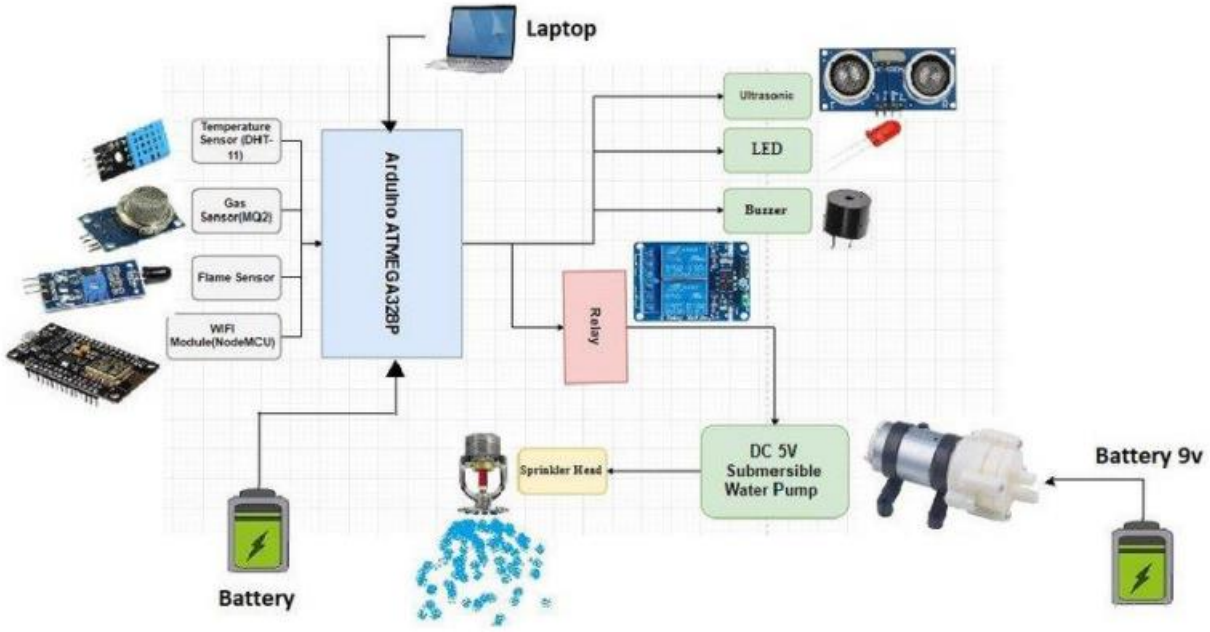


Figure 1: Solution Architecture of Industry-specific intelligent fire management system

TECHNICAL ARCHITECTURE:

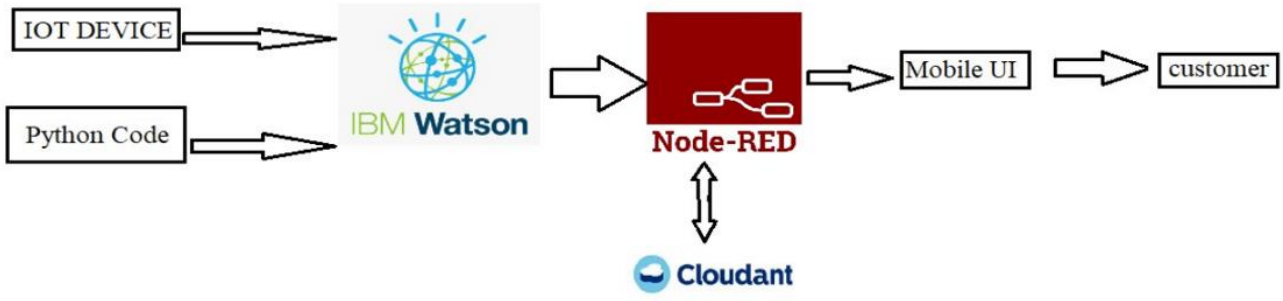


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface		HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
9.	External API-2	Purpose of External API used in the application	Aadhar API, etc.
10.	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Opensource framework
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-	Technology used

S.No	Characteristics	Description	Technology
		services)	
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Technology used
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Technology used

5.3 USER STORIES:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Industry owner)	Installation	USN-1	As a user, I will analyse the feasible positions to fix the fire detection sensor/system	I can easily identify the place	High	Sprint-1
		USN-2	As a user, I will make proper wiring andpiping connections	I make sure that the wire is free from faults	High	Sprint-1
	Fault finding	USN-3	As a user,I ensure that I have installedproperly	an receive information without attenuation	Medium	Sprint-2

	Monitoring system	USN-4	As a user, I ensure that the battery fullycharged	I can ensure,even without power supply the system will work	Medium	Sprint-2
	Sprinkler system	USN-5	As a user, I can ensure that the sprinkler sprinkles the suitable retardant when the system detects fire	I can ensure that the retardant tank is filled for controlling the fire	Medium	Sprint-1
	Sensor system	USN-6	As a user,I will install the various sensors(temperature sensor,smoke detection sensor)	I make sure thate the sensor detects fire	High	Sprint-1
	Alarm system	USN-7	As a user,I will use the buzzer for alertingwhen the system detects fire	I can confirm that thefire is detected	High	Sprint-1
	Alert system	USN-8	As a user,I will use Google API for identifying the location and alert the firestation using GSM	I make sure that I have proper internet connection	High	Sprint-1

6. PROJECT PLANNING AND SCHEDULING:

6.1 SPRINT PLANNING & ESTIMATION:

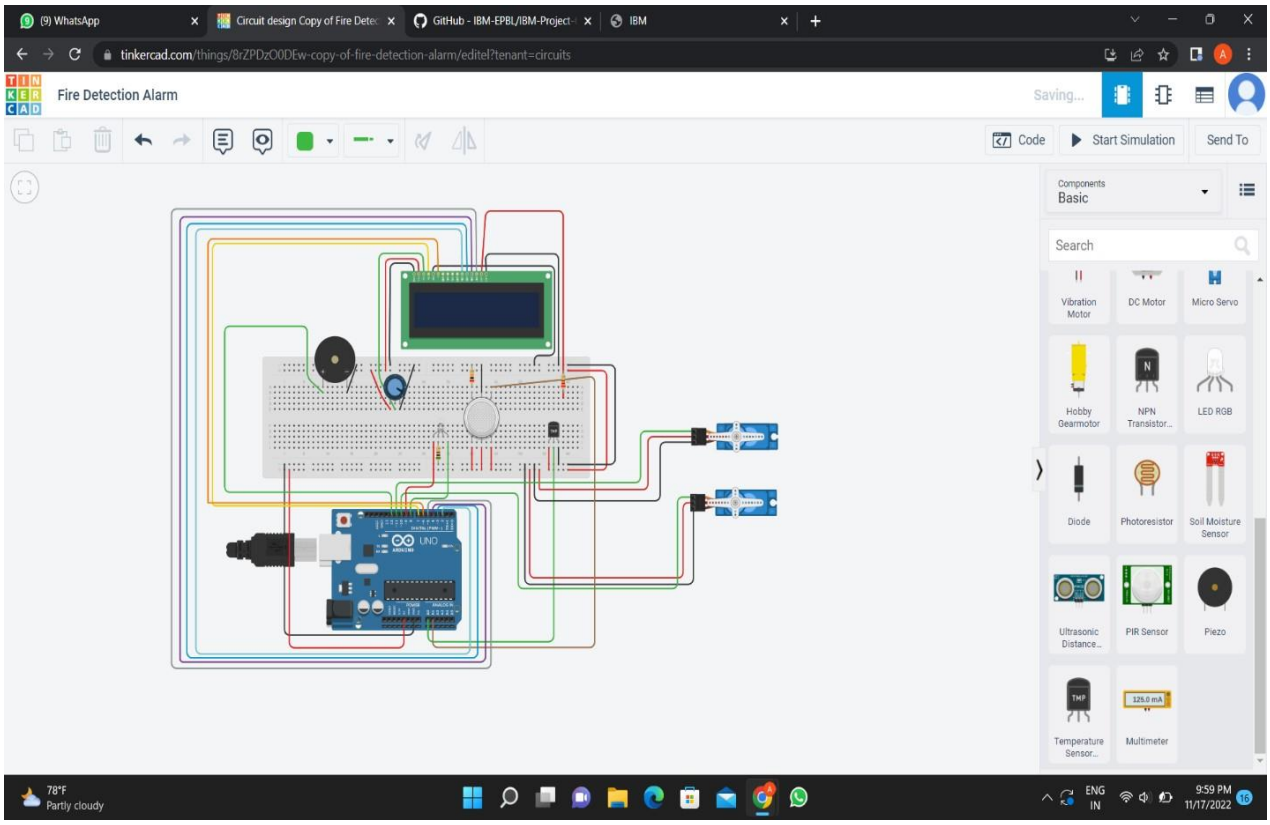
Sprint-1:

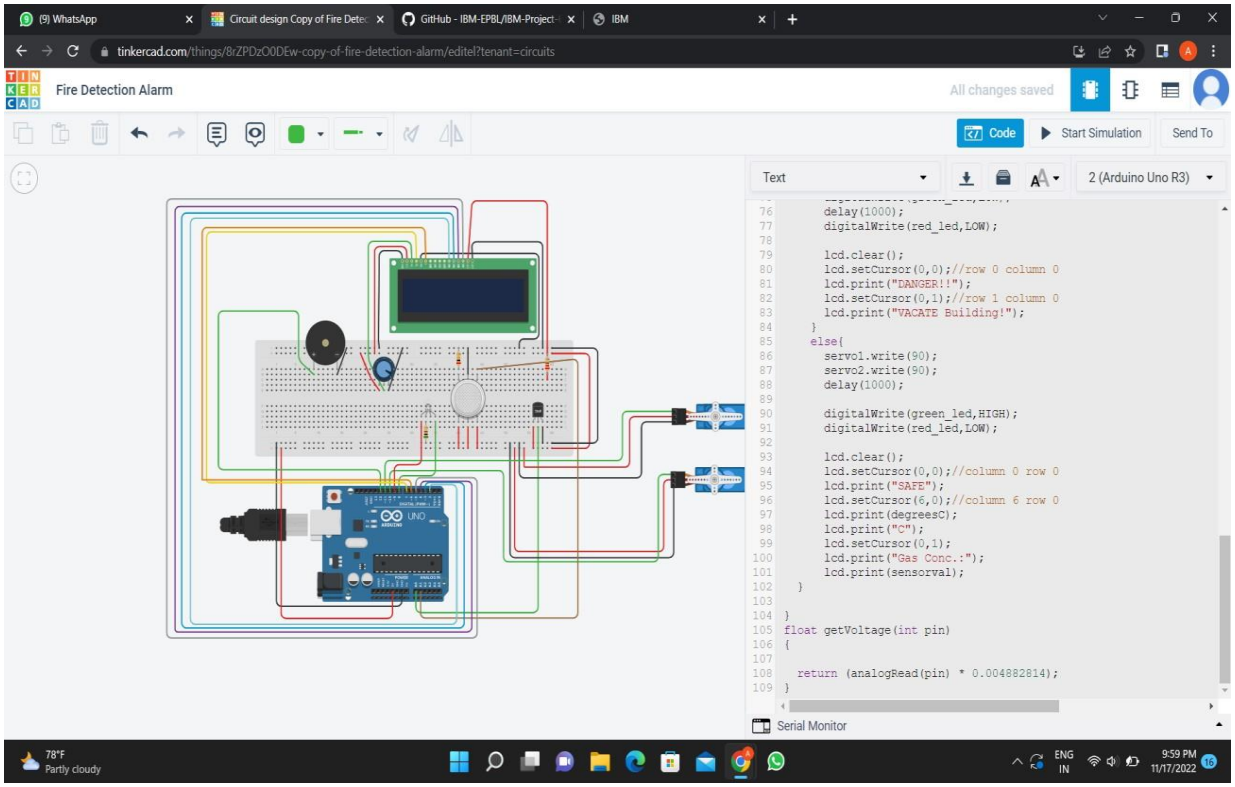
PLATFORM : Tinkercad

WORKFLOW:

- Create a Tinkercad student account and start a new circuit with a name
- Select the required components for the project with your model andConnect the components properly
- Then code the required program to proceed for your Project and execute it
- Finally the output is shown like in the below figure

CIRCUIT CONNECTION:





CODE:

```
#include<Servo.h>//header file for servo
#include <LiquidCrystal.h>//header file for
LCD
//first of all we will use the TMP36 which is a temperature sensor that outputs
//a voltage that's proportional to the ambient temperature.
// We'll use analog input 0 to measure the temperature sensor's signal pin.
//Temparature Sensor
const int temperaturePin = 0; //The output of tmp36 is connected to A0 of
arduinoconst int buzzer = 12; //buzzer is connected to D12 on the arduino
//Gas Sensor
```

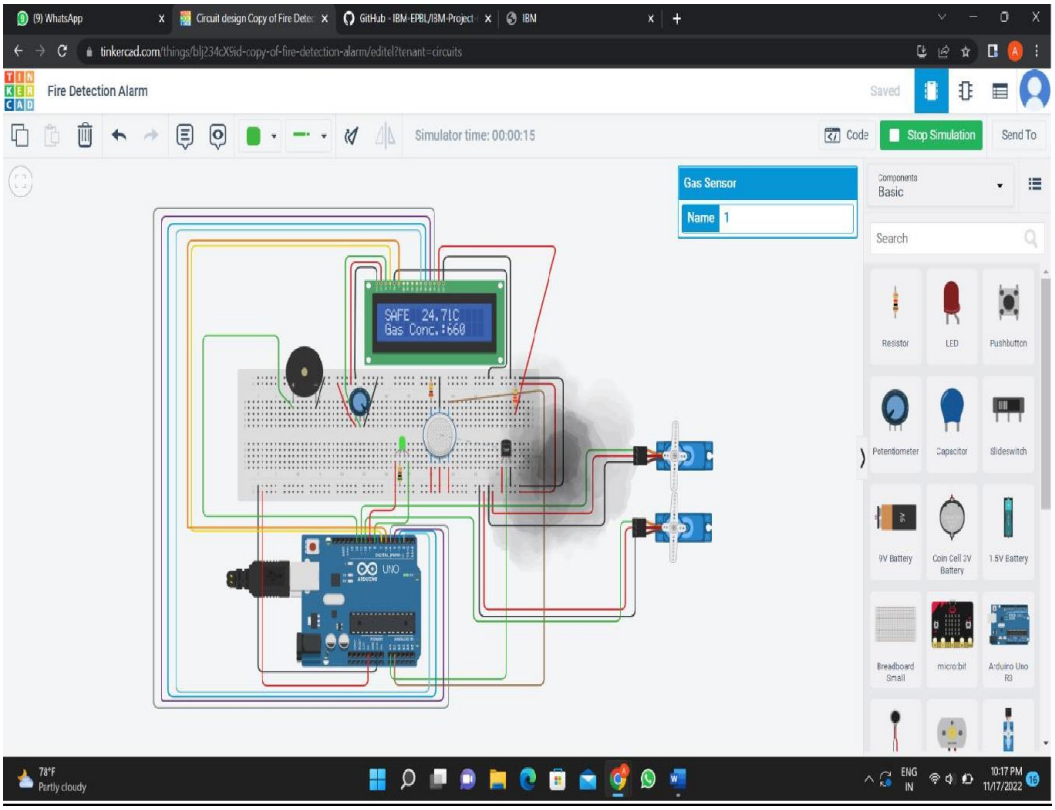
```
int gasSensorPin=A1;//Gas sensor output is connected to A1 of
Arduinoint sensorval;//For storing the value sensed by gas sensor


//Doors
Servo
servo1,servo2;int
servo1Pin=11;
int servo2Pin=10;


//RGB LED
int red_led=9;//Red terminal of RGB LED is connected to D9 of Arduino
int green_led=8;//Green terminal of RGB LED is connected to D8 of Arduino


//LCD
LiquidCrystal lcd(7, 6, 2, 3, 4, 5);//Sets the interfacing pins on Arduino that are
connected to LCD


//7-Rs,6-E(Enable), 5,4,3,2 are the inputs->4 bit
modevoid setup()
{
  pinMode(buzzer, OUTPUT);//set the pin connected to the buzzer as an output
  servo1.attach(servo1Pin);
  servo2.attach(servo2Pin);
  servo1.write(90);//Intially both doors are closed(i.e, 90 degrees)
```

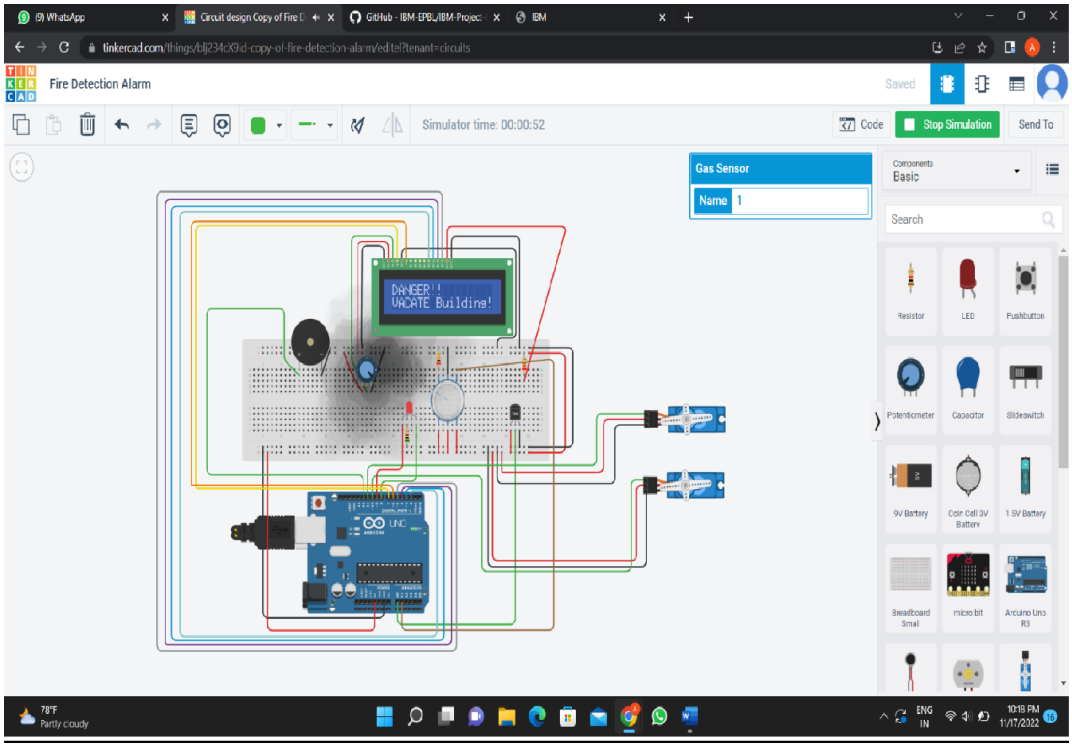
```
  servo2.write(90);
  delay(2000);
  pinMode(red_led,OUTPUT);
  pinMode(green_led,OUTPUT);
  //Serial.begin(9600);
  lcd.begin(16,2);//initialisation of 16*2
  LCD
}
void loop()
{
        //for buzzer and tmp36 temp
        sensorfloat voltage, degreesC;
        voltage =
        getVoltage(temperaturePin);
        degreesC = (voltage - 0.5) * 100.0;
        sensorval=analogRead(gasSensorPin
        );
        //Serial.print(sensorval);
       if(degreesC>37 ||
       sensorval>700)
        {
    digitalWrite(buzzer,
    LOW); tone(buzzer, 800,
    800); delay(200);
```

```
//delay
tone(buzzer,600,800);
delay(200);
servo1.write(0);
servo2.write(0);
delay(1000);
digitalWrite(red_led,HIG
H);
digitalWrite(green_led,LO
W);delay(1000);
digitalWrite(red_led,LOW);
lcd.clear();
lcd.setCursor(0,0);//row 0
      column 0
      lcd.print("DANGER!!");
lcd.setCursor(0,1);//row 1
column 0lcd.print("VACATE
Building!");
    }
    else{
servo1.write(9
0);
servo2.write(9
0);
```

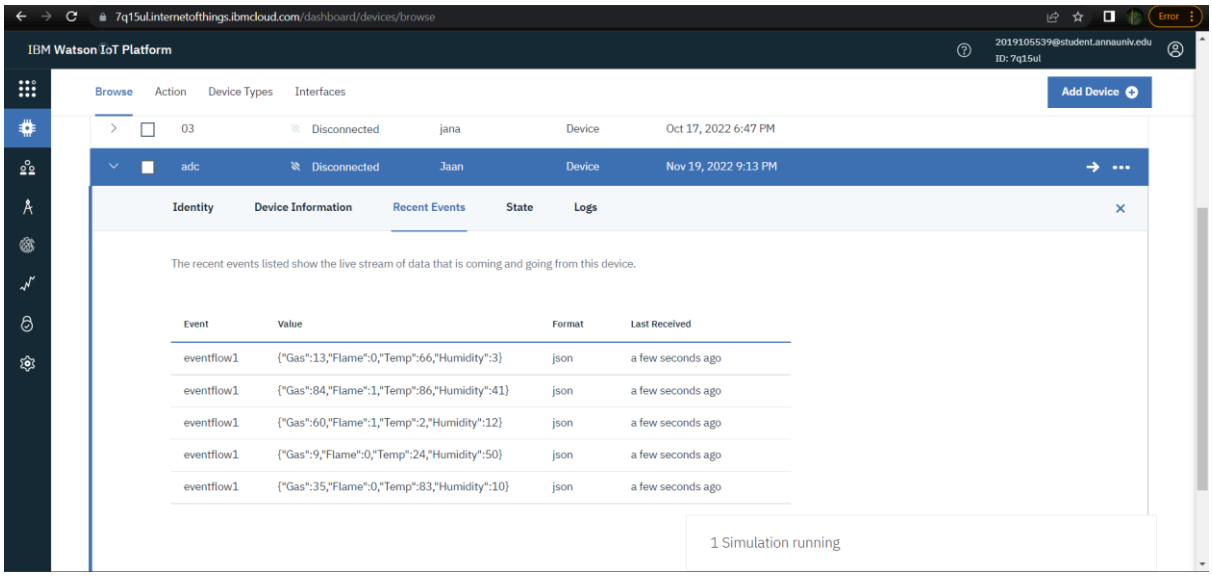
```
      delay(1000);
       digitalWrite(green_led,HIGH);
      digitalWrite(red_led,LOW);
      lcd.clear();
      lcd.setCursor(0,0);//column 0
      row 0lcd.print("SAFE");
      lcd.setCursor(6,0);//column 6
      row 0lcd.print(degreesC);
      lcd.print("C");
      lcd.setCursor(0,1);
      lcd.print("Gas
      Conc.:");
      lcd.print(sensorval);
  }


}
float getVoltage(int pin)
{


  return (analogRead(pin) * 0.004882814);
}
```

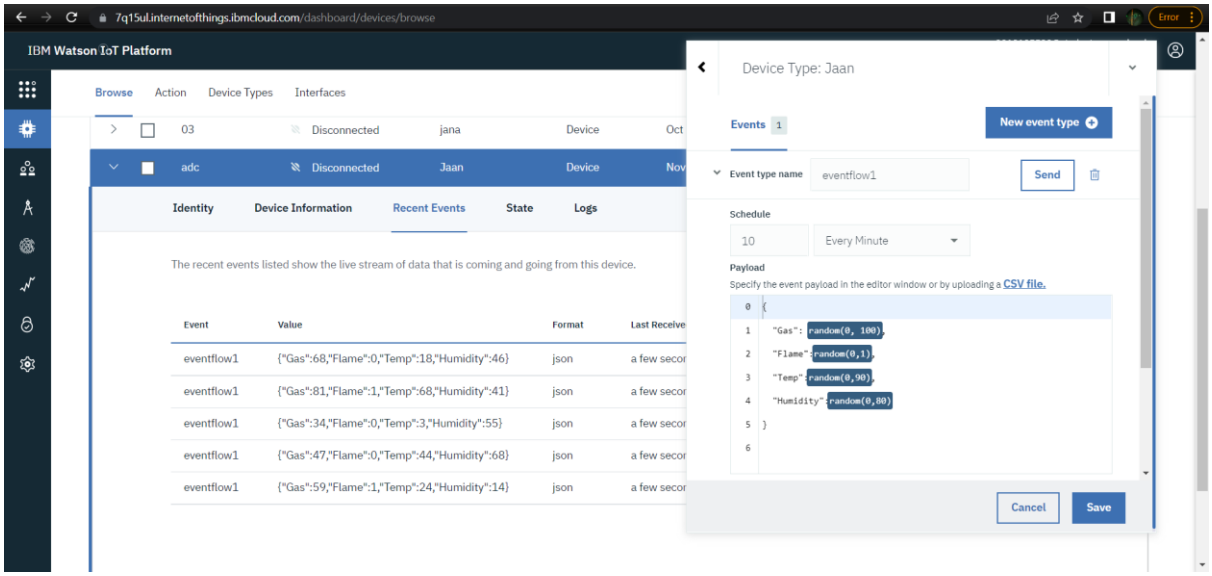
OUTPUT:

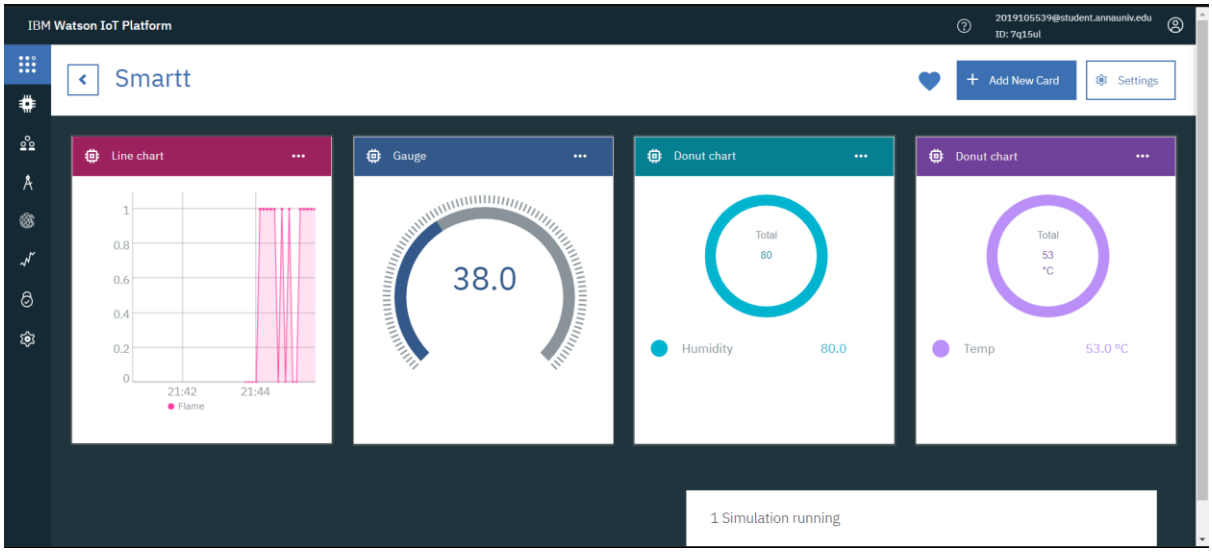

Fire Detection Alarm
Simulator time: 00:00:15
Stop Simulation
Send To
Gas Sensor
Name 1
SAFE 24.71C
Gas Conc.:660

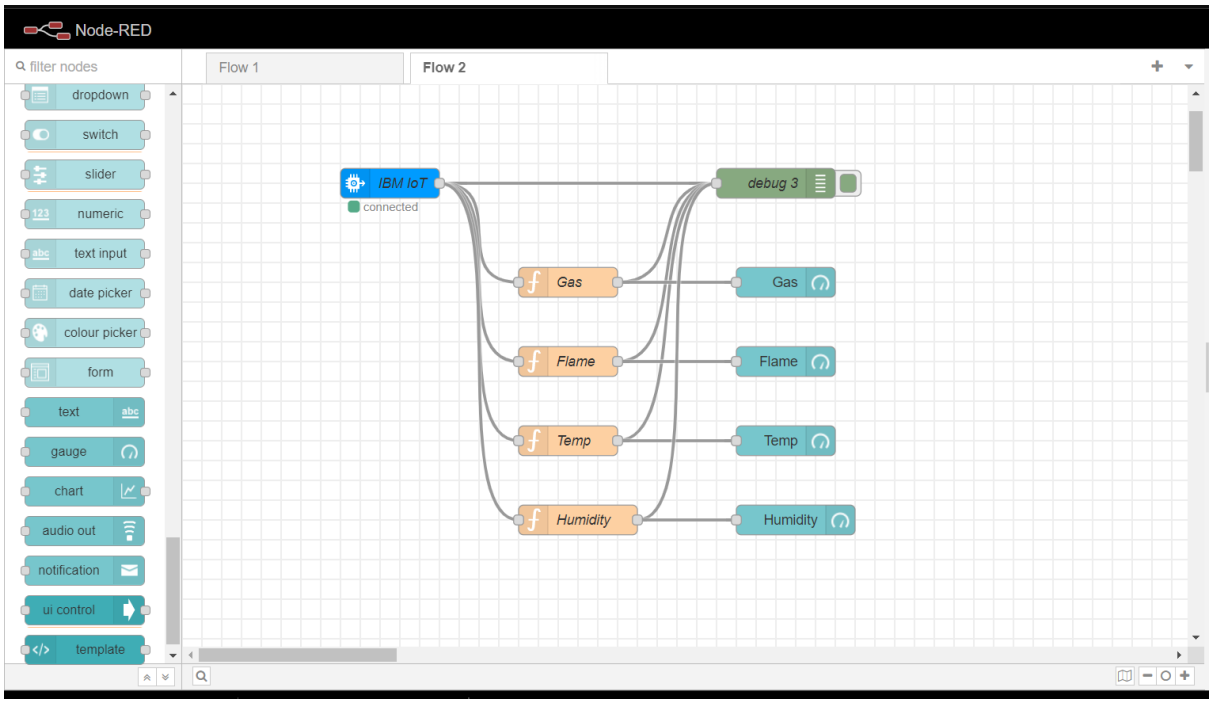

Fire Detection Alarm
Simulator time: 00:00:52
Stop Simulation
Send To
Gas Sensor
Name 1
DANGER!!
VACATE Building!

SPRINT 2 :

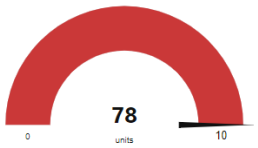

7q15ul.internetofthings.ibmcloud.com/dashboard/devices/browse
IBM Watson IoT Platform
2019105539@student.annauniv.edu
ID: 7q15ul
Browse
Action
Device Types
Interfaces
Add Device
03 Disconnected jana Device Oct 17, 2022 6:47 PM
adc Disconnected Jaan Device Nov 19, 2022 9:13 PM
Identity
Device Information
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
eventflow1 {"Gas":13,"Flame":0,"Temp":66,"Humidity":3} json a few seconds ago
eventflow1 {"Gas":84,"Flame":1,"Temp":86,"Humidity":41} json a few seconds ago
eventflow1 {"Gas":60,"Flame":1,"Temp":2,"Humidity":12} json a few seconds ago
eventflow1 {"Gas":9,"Flame":0,"Temp":24,"Humidity":50} json a few seconds ago
eventflow1 {"Gas":35,"Flame":0,"Temp":83,"Humidity":10} json a few seconds ago
1 Simulation running


7q15ul.internetofthings.ibmcloud.com/dashboard/devices/browse
IBM Watson IoT Platform
Device Type: Jaan
Events 1
New event type
Event type name
eventflow1
Send
Schedule
10
Every Minute
Payload
Specify the event payload in the editor window or by uploading a CSV file.
{
"Gas": random(0, 100),
"Flame": random(0,1),
"Temp": random(0,90),
"Humidity": random(0,80)
}
Cancel
Save

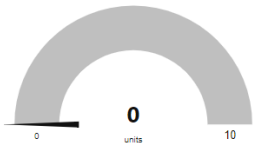

IBM Watson IoT Platform
2019105539@student.annauniv.edu
ID: 7q15ul
Smartt
Add New Card
Settings
Line chart
Gauge
Donut chart
Donut chart
38.0
Total
80
Humidity
80.0
Total
53
°C
Temp
53.0 °C
1 Simulation running


Node-RED
filter nodes
Flow 1
Flow 2
dropdown
switch
slider
numeric
text input
date picker
colour picker
form
text
gauge
chart
audio out
notification
ui control
template
IBM IoT
connected
debug 3
Gas
Gas
Flame
Flame
Temp
Temp
Humidity
Humidity

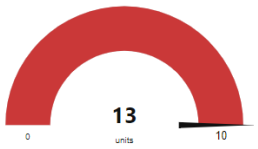
Gas



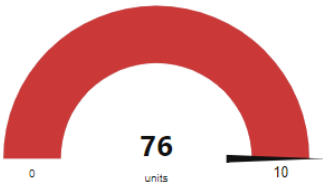
Flame



Temp



Humidity



Temp

SPRINT 3:

CODE:

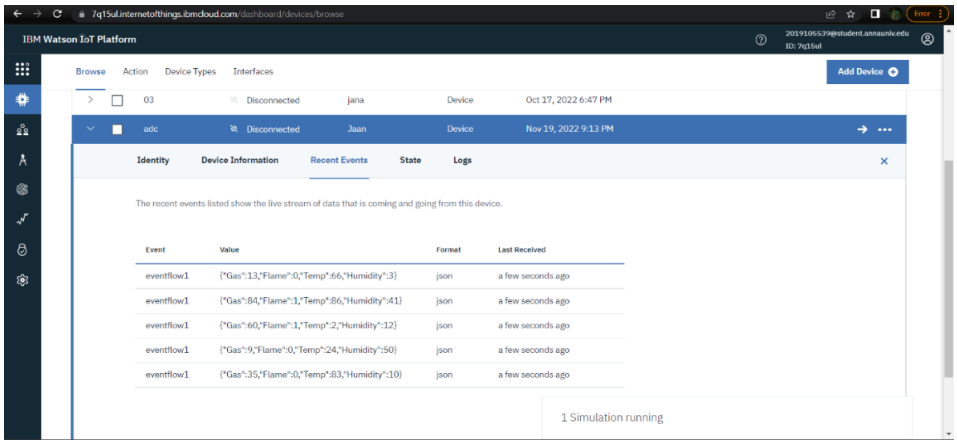
```
#include<WiFi.h>
#include <Wire.h>
#include<Liquidcrystal.h>
#include<ESP32Servo.h>
#include<WiFiClient.h>

unsigned long myChannelNumber = 1;
 const char * myWriteAPIKey= "a-7nqq26-ymfksmglqp";
 int led_pin = 30;buzzer_pin= 10;
Mq1 = 6;
int value = 0;


   //Flame int flame_sensor_pin = 11 ;output
   pin int  flame_pin = HIGH ;

char sid[] = "JANANI";
char  pass[]= "JANANI";
WiFiClientclient;
#define adc_vref_mV 3520.0
#define adc_resolution 4563.0
#define relay_pin17
#define relay_pin1  27

void setup()
{
  Serial.begin(136200); pinMode(relay_pin,
output); pinMode(relay_pin1, output);
   Serial.print("Connecting to ");
 Serial.println(sid); WIFI.begin(sid,
 pass);
  int wifi_ctr = 0;
    while (WIFI.status() != wl_connected)
    {
    delay(1000);
  Serial.print(".");
```

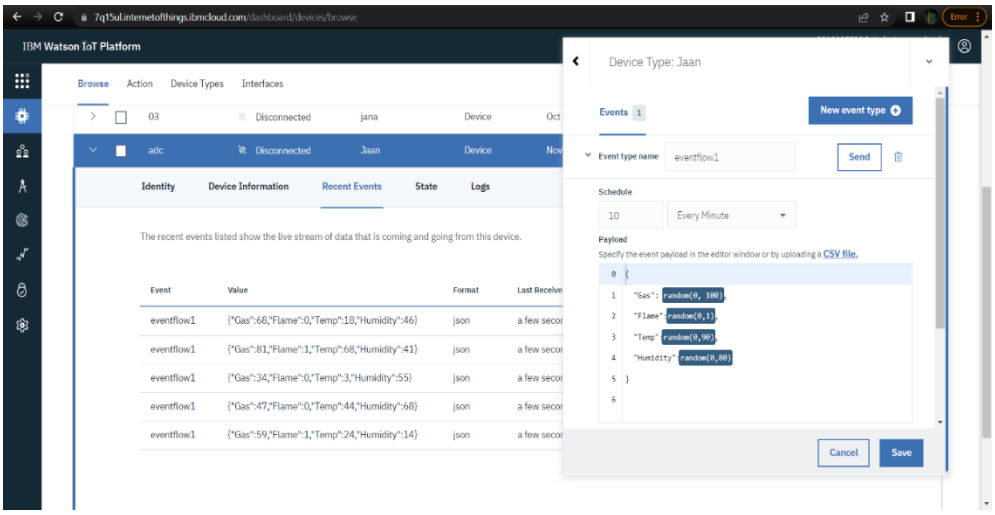
```
    }
    Serial.println("WIFI connected");
  pinMode(led_pin, output); pinMode(mq1,
  input);
  pinMode ( flame_sensor_pin , input );
  pinMode(buzzer_pin, output);
}

void temperature()
{
    int adcVal = analogRead(pin_LM32);
    float milliVolt = adcVal *(adc_vref_mVadc_resolution);float
    tempC = milliVolt /10;
 Serial.print("Temperature: ");
Serial.print(tempC); Serial.print("°C");

    if(tempC > 60)
    {
     Serial.println("Alert");
     digitalWrite(buzzer_pin, HIGH);
    }
    else
    {
      digitalWrite(buzzer_pin, LOW); // turn on
    }



void GasSensor()
{
    //Mq1

    int gassensorAnalogMq1 =analogRead(Mq1);
    Serial.print("Mq1 Gas Sensor: ");
    Serial.print(gassensorAnalogMq1);

    Serial.print("\t");
    Serial.print("\t");
    Serial.print("\t");
```

```
    if (gassensorAnalogMq1 > 1500)
    {
      Serial.println("Mq1Gas");
      Serial.println("Alert");
      digitalWrite(relay_pin1, HIGH);
      delay(100);
    }
    else
    {
      Serial.println("No Mq1Gas");
      digitalWrite(relay_pin1,LOW);
      delay(100);


    }
}

void flamesensor()
  {
  flame_pin = digitalRead ( flame_sensor_pin ) ;
  (flame_pin == LOW
{
  Serial.println ( "FLAME IS DETECTED" ) ;digitalWrite
  (buzzer_pin,HIGH )
   }
  else
{
Serial.println ( " NO FLAME DETECTED " ) ;digitalWrite
(buzzer_pin , LOW ) ;
}
int value = digitalRead(flame_sensor_pin);

    if (value ==LOW)
    {
    Serial.print("FLAME");
    digitalWrite(relay_pin, HIGH);
    }
    else
    {
```

```
      Serial.print("NO FLAME");
   digitalWrite(relay_pin_, LOW);
    }

}
void loop()
{
temperature();
GasSensor();
flamesensor();
```

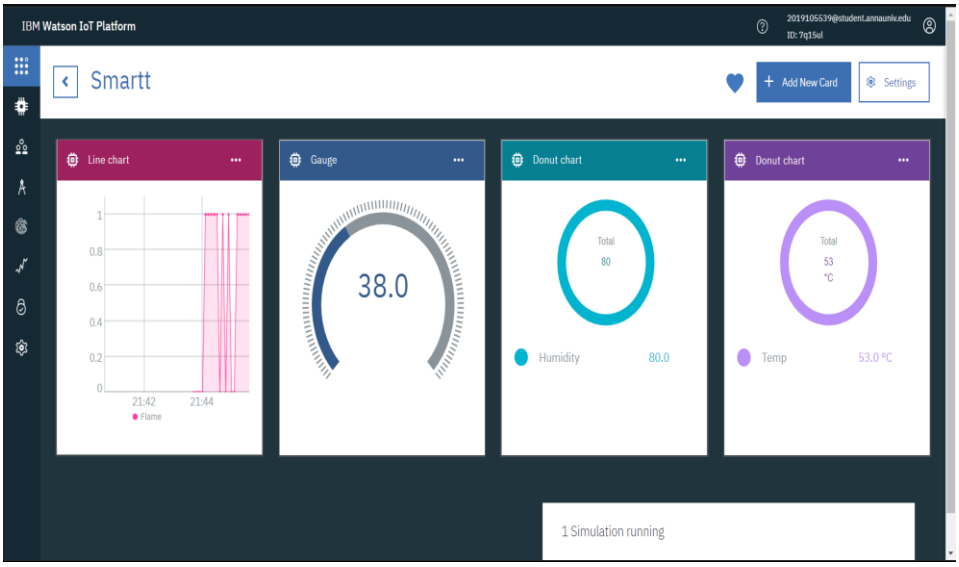
SPRINT 4:

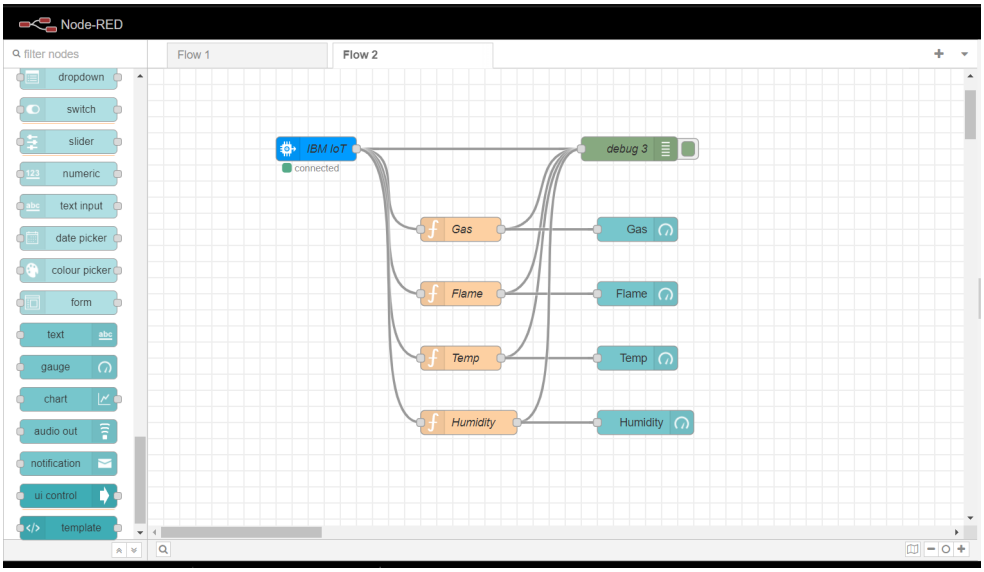




CODE:

```
#include<WiFi.h>
#include<Wire.h>
#include <Liquidcrystal.h>
#include<ESP32Servo.h>
#include<WiFiClient.h>

unsigned long myChannelNumber = 1;
const char *  myWriteAPIKey= "a-7nqq26-ymfksmglqp";
int led_pin = 30;
buzzer= 10;
const int Mql = 4;
  int value = 0;

  Flame int flame_sensor_pin = 11 ;
  output pin int  flame_pin = HIGH ;

char ssid[] = "JANANI";
char  pass[]= “JANANI";
WiFiClient client;
#define pin_lm35 39
#define adc_vref_mV 3520.0
#define adc_resolution 4563.0
#define relay_pin 17
#define relay_pin1  27


void setup()
{
   Serial.begin(136200);
```



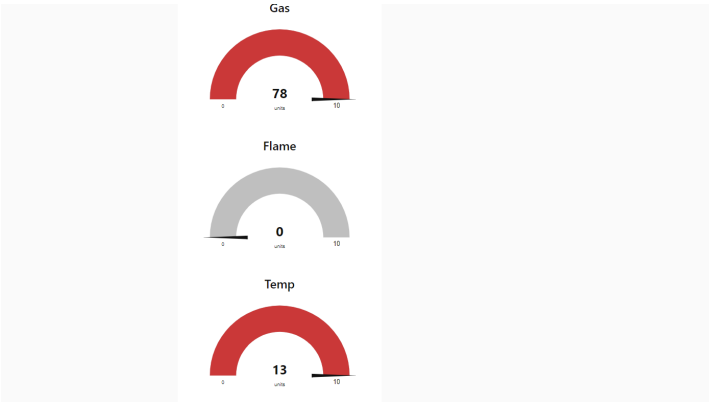


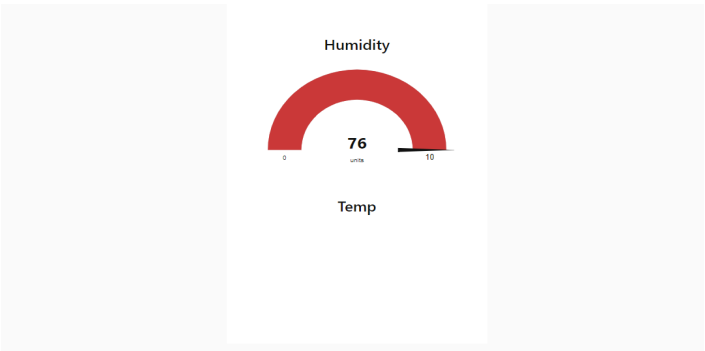

```
pinMode(relay_pin, output);
pinMode(relay_pin1, output);
Serial.print("Connecting to ");
Serial.println(Sid);
  WiFi.begin(Sid, pass);
  int wifi_ctr = 0;
   while (WiFi.sts() != wi_connected)
   {
   delay(1000);
  Serial.print(".");
   }
```

```
Serial.println("wi_connected ");
  Liquidcrystal.begin(client);
  pinMode(led_pin, output);
  pinMode(Mq2, input);
  pinMode ( flame_sensor_pin , input );
  pinMode(buzzer, output);
}


void temperature()
{
int adcVal = analogRead(pin_LM35);
float milliVolt = adcVal * (adc_vref_mV / adc_resolution);
float tempC = milliVolt /10;
Serial.print("Temperature: ");
Serial.print(tempC);
Serial.print("°C");
if(tempC> 60)
    {
     Serial.println("Alert"); digitalWrite(buzzer,high); //
     turn on
    }  else
    {
      digitalWrite(buzzer, low); // turn on
    }
     int x = Liquidcrystal.writeField(myChannelNumber,1, tempC, myWriteAPIKey);
     }


void GasSensors()
{

    int gassensorAnalogMq1 =analogRead(Mq1);
    Serial.print("Mq1 Gas Sensor: ")
    Serial.print(gassensorAnalogMq1);
    Serial.print("\t");
    Serial.print("\t");
```

```
    Serial.print("\t");
    if (gassensorAnalogMq1 > 1000)
    {
     Serial.println("Mq1Gas");
     Serial.println("Alert");
     digitalWrite(relay_pin1, high);
     delay(100);
    }  else
    {
   Serial.println("No Mq1Gas");
   digitalWrite(relay_pin1,low);
   delay(100);

    }
  int a = Liquidcrystal.writeField(myChannelNumber,4, gassensorAnalogMq1,
myWriteAPIKey);

}

void flamesensor()
  { flame_pin = digitalRead ( flame_sensor_pin ) ;
  if (flame_pin == LOW ) // applying condition
{
  Serial.println ( " alert: flame is detected" ) ;
  digitalWrite (buzzer,high ) ;/
   if state is high,
  then turn high the buzzer
    }
  else
{
Serial.println ( " no flame detected " ) ;


digitalWrite (buzzer , low ) ; // otherwise turn it low
}

int value = digitalRead(flame_sensor_pin); // read the analog value from sensor
```

```
  if (value ==low)
  {
  Serial.print("flame");
  digitalWrite(relay_pin, high);
     }
     else
     {
   Serial.print("no flame");
   digitalWrite(relay_pin, low);
     }
}
void loop() {
temperature();
GasSensors();
flamesensor();
     }
```





6.2 SPRINT DELIVERY SCHEDULE:

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Installation	USN-1	As a user, I will analyse the feasible positions to fix the fire detection sensor/system	2	High	Akshaya , Divyapriya
Sprint-1		USN-2	As a user, I will make proper wiring and piping connections	1	High	Janani, Soumiyaa
Sprint-2	Fault finding	USN-3	As a user,I ensure that I have installed properly	2	Medium	Akshaya , Divyapriya
Sprint-1	Monitoring system	USN-4	As a user, I ensure that the battery fully charged	2	Medium	Janani, Soumiyaa
Sprint-2	Sprinkler system	USN-5	As a user, I can ensure that the sprinkler sprinkles the suitable retardant when the system detects fire	2	High	Akshaya , Divyapriya
Sprint-3	Sensor system	USN-6	As a user,I will install the various sensors(temperature sensor,smoke detection sensor)	2	High	Janani, Soumiyaa
Sprint-4	Alarm system	USN-7	As a user,I will use the buzzer for alerting when the system detects fire	2	High	Akshaya , Divyapriya

Sprint-4	Alert system	USN-8	As a user,I will use Google API for identifying the location and alert the fire station using GSM	2	High	Janani, Soumiyaa

Project Tracker, Velocity & Burndown Chart:

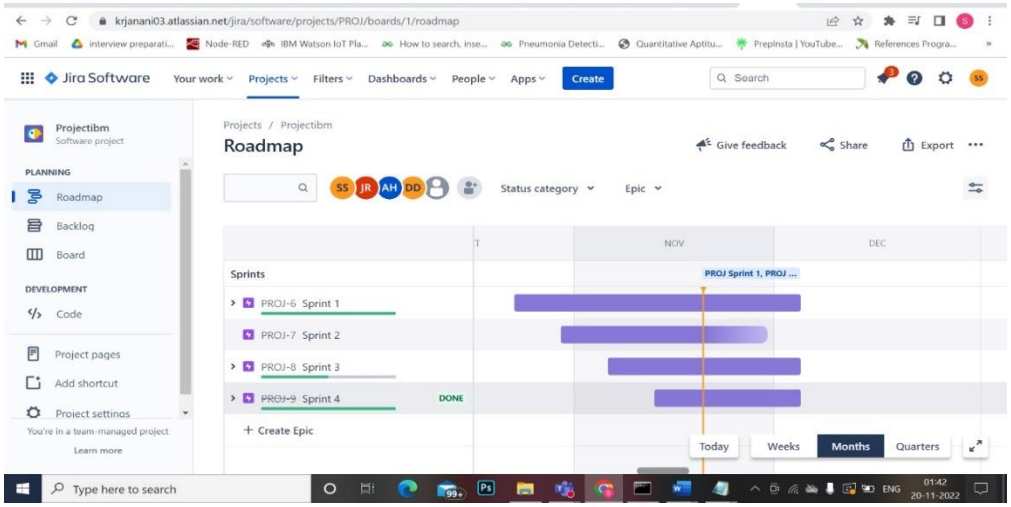
Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	31 Oct 2022
Sprint-3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

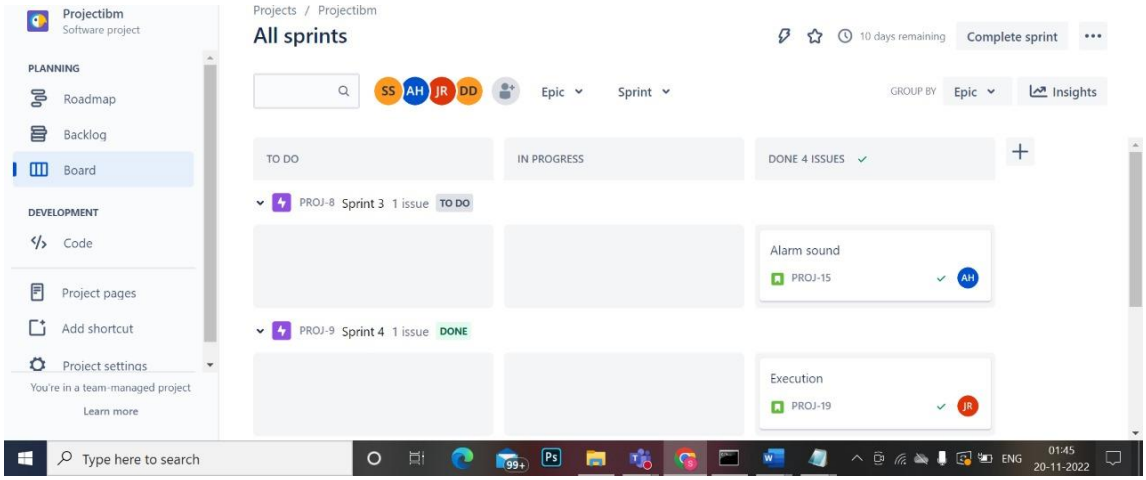
Velocity:

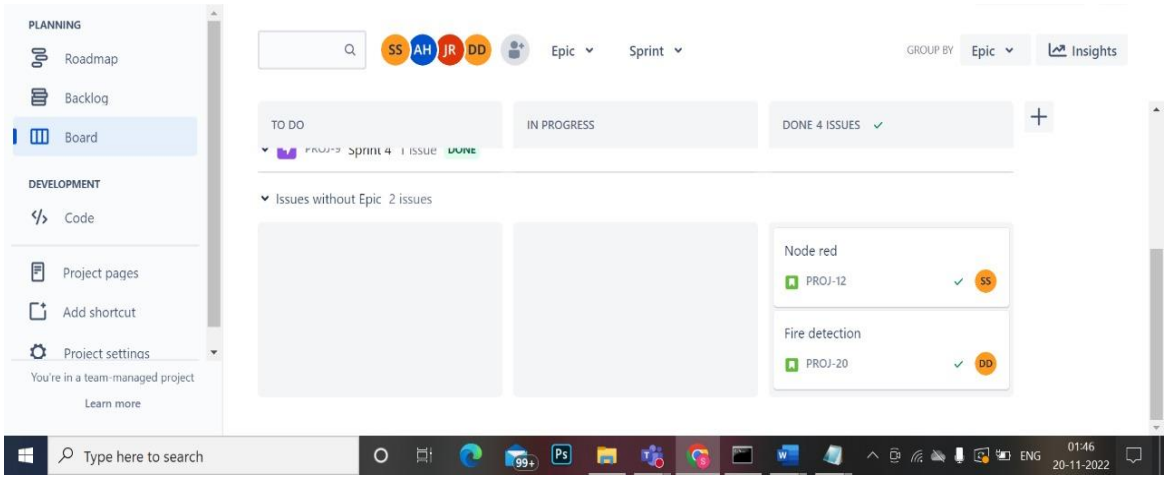
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

6.3 REPORT FROM JIRA:







7.CODING AND SOLUTIONING:

7.1 FEATURE 1:

Develop a python code for publishing data to the IBM IOT Platform

CODE:

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


#Provide your IBM Watson Device Credentials
organization = "7q15ul"
deviceType = "jana"
deviceId = "03"
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")
   else :
      print ("led 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)

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

```
        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(10)

        deviceCli.commandCallback = myCommandCallback

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

7.2 FEATURE 2:

CODE:

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


#Provide your IBM Watson Device Credentials
organization = "7q15ul"
```

```
deviceType = "jaan"
deviceId = "adc"
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")
   else :
      print ("led 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(0,90)
       Humidity=random.randint(0,80)
       Gas=random.randint(0,100)
       Flame=random.randint(0,1)
```

```
    data = { 'Temp' : Temp, 'Humidity': Humidity,'Gas': Gas, 'Flame': Flame }
    #print data
    def myOnPublishCallback():
      print ("Published Temperature = %s C" % Temp, "Humidity = %s %%" % Humidity,
"Gas = %s " % Gas, "Flame = %s " % Flame, "to IBM Watson")

    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
    if not success:
      print("Not connected to IoTF")
    time.sleep(10)
    deviceCli.commandCallback = myCommandCallback
```

8.TESTING:

8.1 TEST CASES:

F14 — 1.Simualte the fire in the simulator
2. click the sent alert button in the dashboard ui

Date	18-Nov-22
Team ID	PNT2022TMID35457
Project Name	Industry-specific intelligent fire
Team Members	Akshaya H, Divyapriya D, Janani K R, Sowmiyaa S U

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N	BUG ID	Executed By
Sensor_003	Functional	Microcontroller	Sensor data is taken	The microcontroller should receive data from the sensors	1.Open the simulator in wokwi. 2.Click on run button.	Random values generated	Get the values and print it in the serial out	Working as expected	Pass		N		Janani K R
Sensor_004	Functional	Microcontroller	Sensor data is taken	Sensor should be connected properly	1.Open the simulator in wokwi. 2.Click on run button.	Random values generated	Get the values and print it in the serial out	Working as expected	Pass		N		Sowmiyaa S U
Work_001	Functional	Microcontroller and IBM IOT watson platform	The data should be sent to IBM IoT platform using mqtt protocol using json format	The device setup is completed and necessary libraries are installed	1.Start the simulation in wokwi. 2.Go to IBM IoT platform. 3.Click on the device id which is	Random values generated	The values are shown in recent events tab in json format	Working as expected	Pass		N		Akshaya H
Work_002	Functional	Node-red	The data should be sent to Node-red platform	The necessary packages should be installed in node-red for ibm iot And the iot node have to be connected and configured with ibm iot	1.Login to node red editor 2.Configure the environment 3.Make function nodes to get the data from the iot device 4.verify it with a debug node	values got from the iot device	The debug area should show the correct values.	Working as expected	Pass		N		Divyapriya D
Work_003	Functional	Node-red	Verify that the json data is parsed in correct way.	A configured node-red with IBM watson	1.Login to node red editor 2.Attach a debug node to the iot node 3.Deploy the app.	values got from the iot device	the debug menu shows the output as proper json format	Working as expected	Pass		N		Janani K R
Database_001	Storage	Cloudant	The received data is stored in database in a key value pair	The node red is connected with cloudant node	1.login to cloudant dashboard. 2.create new database. 3. connect the database with node red and then give the database name in required field	values got from the iot device	After sending the data the data is stored in cloudant	Working as expected	Pass		N		Sowmiyaa S U
Apikey_001	API	API key	The sms is sent when there is fire alert	The node red should be configured to send a post request	1.Simualte the fire in the simulator 2. click the sent alert button in the dashboard ui	Alerts when fire is detected	sms will be sent when fire is dete	Working as expected	Pass		N		Akshaya H
Work_004	Functional	UI	Even at times of emergency sometimes manual control is required	the dashboard interaction elements is connected to the node-red	1.in the dashboard enter the correct pin 2.click the action to be done	The action by user	manual command system works	Working as expected	Pass		N		Divyapriya D
Ver_001	Functional	UI	Verify that the correct pin is entered	test filed is given in dashboard to enter pin	1.The correct pin is entered 2.then necessary action is required	12345678	command is sent successfull	working as expected	Pass		N		Janani K R
Apikey_002	Functional	Microcontroller	Verify that the message is not sent continuously	the sms funtionality should be implemented	1.Simulate a fire accident scenario 2.or click the send alert button on the dashboard 3.wait for the message to be sent	the event is simulated or triggered	Simulated without an errors	Working as expected	Pass		N		Sowmiyaa S U

Testcases | TestScearnios

8.2 USER ACCEPTING TESTING:

Purpose of Document:

The purpose of this document is to briefly explain the test coverage and open issues ofthe Industry-specific intelligent fire management system 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, andhow they were resolve

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By design	15	7	5	9	36
External	8	6	9	16	39
Fixed	2	6	10	8	26
Not Reproduced	12	9	4	5	30
Skipped	4	1	5	3	13
Won't Fix	1	7	3	5	16
Totals	42	36	36	46	160

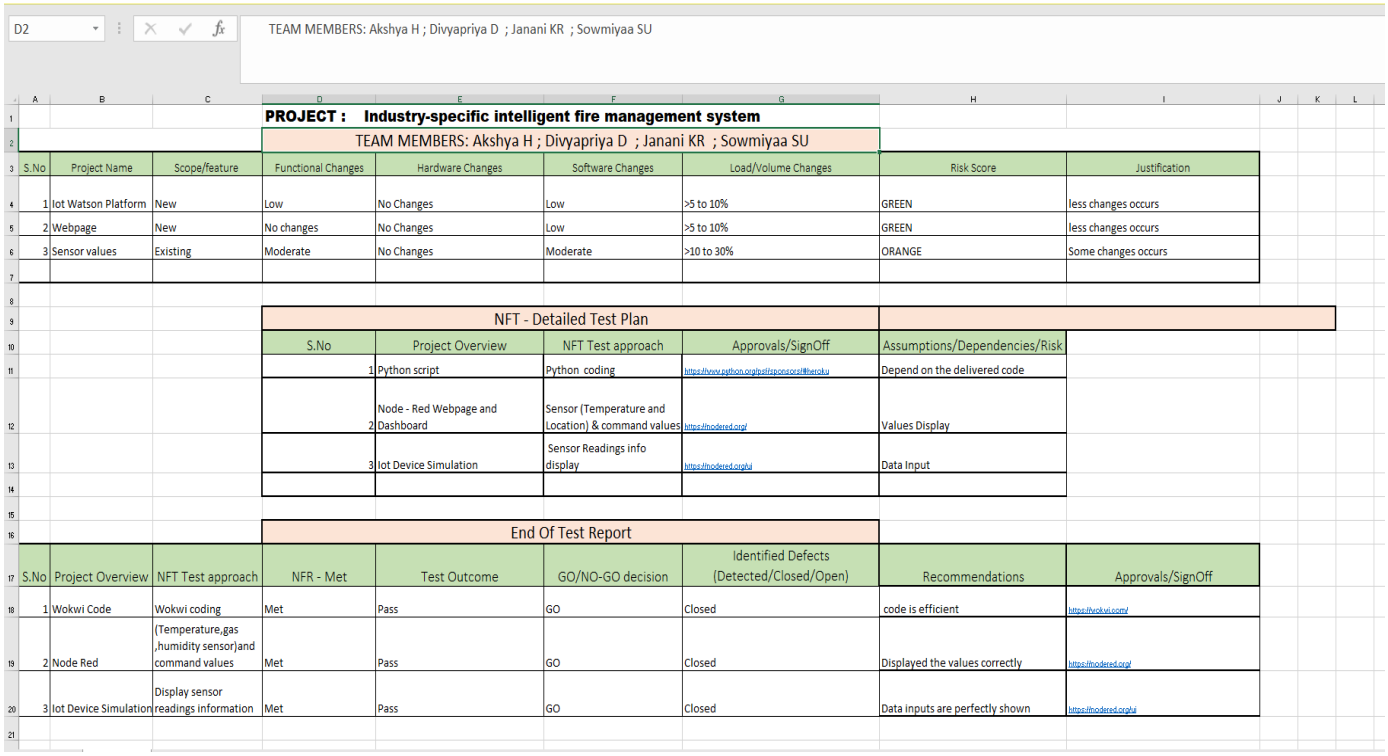
Test Case Analysis:

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

Section	Total cases	Not Tested	Fail	Pass
Client Application	5	0	0	5
Security	3	0	0	3
Exception Reporting	13	0	0	13
Final Report Output	6	0	0	6

9.RESULT:

9.1 PERFORMANCE METRICS:



D2 | TEAM MEMBERS: Akshya H ; Divyapriya D ; Janani KR ; Sowmiyaa SU

PROJECT : Industry-specific intelligent fire management system

TEAM MEMBERS: Akshya H ; Divyapriya D ; Janani KR ; Sowmiyaa SU

S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Load/Volume Changes	Risk Score	Justification
1	Iot Watson Platform	New	Low	No Changes	Low	>5 to 10%	GREEN	less changes occurs
2	Webpage	New	No changes	No Changes	Low	>5 to 10%	GREEN	less changes occurs
3	Sensor values	Existing	Moderate	No Changes	Moderate	>10 to 30%	ORANGE	Some changes occurs

NFT - Detailed Test Plan

S.No	Project Overview	NFT Test approach	Approvals/SignOff	Assumptions/Dependencies/Risk
1	Python script	Python coding	https://www.python.org/psf/sponsors/#thero.ku	Depend on the delivered code
2	Node - Red Webpage and Dashboard	Sensor (Temperature and Location) & command values	https://nodered.org/	Values Display
3	Iot Device Simulation	Sensor Readings info display	https://nodered.org/ui	Data Input

End Of Test Report

S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Identified Defects (Detected/Closed/Open)	Recommendations	Approvals/SignOff
1	Wokwi Code	Wokwi coding	Met	Pass	GO	Closed	code is efficient	https://wokwi.com/
2	Node Red	(Temperature,gas ,humidity sensor)and command values	Met	Pass	GO	Closed	Displayed the values correctly	https://nodered.org/
3	Iot Device Simulation	Display sensor readings information	Met	Pass	GO	Closed	Data inputs are perfectly shown	https://nodered.org/ui

10. ADVANTAGES & DISADVANTAGES:

ADVANTAGES:

- Fire alarms provide early detection of fire.
- They monitor for 24/7
- Fire alarms can save you money on your insurance to a large extent
- Avoid Smoke Inhalation and Early detection
- Wireless fire alarm system is it operates off of a battery
- Easy & Affordable
- Less prone to false alarms from cooking fumes or shower steam
- Ideal for detecting dense smoke

DISADVANTAGES:

- Sensitive to dust particles and insects, meaning that regular maintenance is needed
- Expensive to maintain
- Require more current to operate (they are typically wired to a 110-volt power source)
- The system is essentially useless if the batteries aren't charged, since it won't work properly.

11.CONCLUSION:

A fire alarm is a device that detects the presence of fire and atmospheric changes relating to smoke. The fire alarm operates to alert people to evacuate a location in which a fire or smoke accumulation is present. When functioning properly, a fire alarm will sound to notify people of an immediate fire emergency. This distinct sound exists to allow the notification to be heard The fire alarm constructed by this project work is reliable at low cost.

12.FUTURE SCOPE:

Fire detection technologies have been slow to evolve compared to rapidly advancing smart devices. Understandably, global companies focus their efforts on developing high-return products, especially ones that connect consumers with popular trends. While fire alarms aren’t exactly at the forefront of social advancement, innovative companies are developing new methods of approaching fire and gas-related threats.

Upcoming Technologies includes Sensor-Assisted Fire Fighting,High-Pressure Water Mist,Drones,Fireballs.

As new technologies emerge, dealers should be sure to leverage both timeless and emerging technologies to target more customers. Devices are becoming more and more capable, and regardless their application, they are all evolving to connect to the internet and cooperate with other devices. Consumers look for smart home technology along with commercial products that work in a cohesive environment. Along with investigating which products to sell, we encourage dealers to take a serious look at reliable partners to provide wholesale alarm monitoring services for devices so that consumers never go without protection.

13.APPENDIX:

SOURCE CODE:

```
#include<Servo.h>//header file for servo
#include <LiquidCrystal.h>//header file for
LCD
//first of all we will use the TMP36 which is a temperature sensor that outputs
//a voltage that's proportional to the ambient temperature.
// We'll use analog input 0 to measure the temperature sensor's signal pin.
//Temparature Sensor
const int temperaturePin = 0; //The output of tmp36 is connected to A0 of
arduinoconst int buzzer = 12; //buzzer is connected to D12 on the arduino
//Gas Sensor
int gasSensorPin=A1;//Gas sensor output is connected to A1 of
Arduinoint sensorval;//For storing the value sensed by gas sensor


//Doors
Servo
servo1,servo2;int
servo1Pin=11;
int servo2Pin=10;


//RGB LED
int red_led=9;//Red terminal of RGB LED is connected to D9 of Arduino
```

```
int green_led=8;//Green terminal of RGB LED is connected to D8 of Arduino
//LCD
LiquidCrystal lcd(7, 6, 2, 3, 4, 5);//Sets the interfacing pins on Arduino that are
connected to LCD
//7-Rs,6-E(Enable), 5,4,3,2 are the inputs->4 bit
modevoid setup()
{
  pinMode(buzzer, OUTPUT);//set the pin connected to the buzzer as an output
  servo1.attach(servo1Pin);
  servo2.attach(servo2Pin);
  servo1.write(90);//Intially both doors are closed(i.e, 90 degrees)
  servo2.write(90);
  delay(2000);
  pinMode(red_led,OUTPUT);
  pinMode(green_led,OUTPUT);
  //Serial.begin(9600);
  lcd.begin(16,2);//initialisation of 16*2
  LCD
}
void loop()
{
        //for buzzer and tmp36 temp
        sensorfloat voltage, degreesC;
        voltage =
```

```
    getVoltage(temperaturePin);
    degreesC = (voltage - 0.5) * 100.0;
    sensorval=analogRead(gasSensorPin
    );
    //Serial.print(sensorval);
  if(degreesC>37 ||
  sensorval>700)
    {
digitalWrite(buzzer,
LOW); tone(buzzer, 800,
800); delay(200);
//delay
tone(buzzer,600,800);
delay(200);
servo1.write(0);
servo2.write(0);
delay(1000);
digitalWrite(red_led,HIG
H);
digitalWrite(green_led,LO
W);delay(1000);
digitalWrite(red_led,LOW);
lcd.clear();
lcd.setCursor(0,0);//row 0
```

```
      column 0
      lcd.print("DANGER!!");
lcd.setCursor(0,1);//row 1
column 0lcd.print("VACATE
Building!");
    }
    else{
servo1.write(9
0);
servo2.write(9
0);
delay(1000);
 digitalWrite(green_led,HIGH);
digitalWrite(red_led,LOW);
lcd.clear();
lcd.setCursor(0,0);//column 0
row 0lcd.print("SAFE");
lcd.setCursor(6,0);//column 6
row 0lcd.print(degreesC);
lcd.print("C");
lcd.setCursor(0,1);
lcd.print("Gas
Conc.:");
lcd.print(sensorval);
```

```
  }


}

float getVoltage(int pin)

{

  return (analogRead(pin) * 0.004882814);

}
```

PROJECT DEMO LINK:

https://drive.google.com/drive/folders/1x8q9yzxnCbwkrcTGJ4o-CsEU9o4kDAXv?usp=sharing

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-6681-1658834440