INDUSTRY-SPECIFIC INTELLIGENT FIRE MANAGEMENT SYSTEM

PROJECT REPORT

TEAM MEMBERS:

Madhumitha. G

Harish Kumar. C.S.G

Anuranjann. G.T

Akshayalakshmi. E

TEAM ID:

PNT2022TMID27908

TITLE

1.INTRODUCTION:

Fire accidents in industries kill more people every year. While controlled fire helps us to save people and property, uncontrolled fire can be harmful. However, detection and control of fire can save lives and properties. Therefore, we have come up with an industry specific intelligent fire management system that serves to save people and property worth millions.

1.1 PROJECT OVERVIEW

The fire management system consists of flame sensor, temperature sensor, sprinklers, exhaust fans and GSM (Global System for Mobile communication) to detect any changes in the environment by placing the sensors at the required position. Based on the temperature readings, the sensors operate automatically. If any gases are detected, the exhaust fans are turned on. Similarly, if any flame is detected, sprinklers will be switched on. After the detection, alert messages are sent to the industry authorities and fire stations.

1.2 PURPOSE

The purpose of this project is to meet the organizations requirements and to deal with the problem of fire management in industries, where the fire prevention system is not optimized. This fire management system reduces the risk of fire accidents by detecting and preventing it and make certain of the safety of workers and properties of an industry.

2. LITERATURE SURVEY

The information related to the project are gathered and the existing solutions, technical papers and research publications are referred for the project development

LITERATURE SURVEY

TEAM:
G.Madhumitha (Team leader)
E.Akshayalakshmi (Team member)
C.S.G. Harish Kumar (Team member)
G.T. Anuranjann (Team member)
TOPIC: Industry-specific intelligent fire management system

PROJECT TITLE	AUTHOR/WEBSITE	OBJECTIVE/OUTCOME
Efficient fire detection for uncertain surveillance environment	**Publisher: IEEE** IEEE Transactions on Industrial Informatics **Author**-Khan Muhammad Salman Khan **Date of Publication:** 05 February 2019	This research paper proposes an efficient method for fire detection in uncertain environment. This is achieved using a 5G TI-enabled fire detection system for which their proposed framework fits well, considering its promising accuracy, minimum false alarm rate, and response time.
IOT-based fire fighters for disaster case management	**Publisher: IEEE** IEEE Sensors Journal **Author**-Murtaza Cicioglu Ali Calhan **Date of Publication:** 31 July 2020	In this study, the proposed IoT system can detect the gases in the environment in which the fire-fighters interfere with the fire, and give warnings and suggestions to fire-fighters accordingly (which extinguishing technique, type of gas, etc.) and send the position information with health signals of the fire-fighter to a remote central control unit. In this way, it will be possible to protect human life (both fire-fighters and victims) and avoid more dangerous situations that may occur.

Research on fire alarm computer monitoring system in fire engineering	**Publisher**: Journal of physics www.iopscience.iop.org **Author**- Xiyang Feng1 Chaofei Wang1 **Year of publication**-2021	This research in fire protection engineering is a kind of early warning monitoring system based on intelligent equipment, which judges the fire situation by detecting changes in the environment.
IOT-based smoke detection in foggy environment	**Publisher:IEEE** IEEE Internet of Things Journal **Date of Publication:** 30 January 2019	This research proposes an energy-efficient system based on deep convolutional neural networks for early smoke detection in both normal and foggy IoT environments.
IOT based fire department alerting system project	**Website:**https://www.projectsof8051.com/iot-based-fire-alerting-system-project/	This review serve as detailing the methods used to design an IOT based Fire Alerting System using Temperature and a smoke sensor which would not only signal the presence of fire in a particular premise but will also send related information through IOT.
Fire protection systems	**Website:**https://fire.nv.gov/uploadedfiles/firenvgov/content/bureaus/FST/4-ifipp-PSsm.pdf **Date of publication**: July 7 2012	This study shows properly designed, installed, operated, and maintained fire alarm system and provides the novice inspector a solid foundation on which to build.

2.1 EXISTING PROBLEM

The fire accidents in industries kill many people and damage property worth lacs. Fire management system helps to prevent machines from getting damaged and save human lives from industry fire accidents. The system is also implemented to reduce the risk of fire accidents in industries.

2.2 REFERENCES

1 Efficient fire detection for uncertain surveillance environment
Publisher: IEEE
IEEE Transactions on Industrial Informatics
Author-Khan Muhammad Salman Khan
Date of Publication: 05 February 2019

2 IOT-based fire fighters for disaster case management
Publisher: IEEE
IEEE Sensors Journal
Author-Murtaza Cicioglu Ali Calhan
Date of Publication: 31 July 2020

3 Research on fire alarm computer monitoring system in fire engineering
Publisher: Journal of physics
www.iopscience.iop.org
Author- Xiyang Feng1 Chaofei Wang1
Year of publication-2021

4 IOT-based smoke detection in foggy environment
Publisher:IEEE
IEEE Internet of Things Journal
Date of Publication: 30 January 2019

5 IOT based fire department alerting system project
Website: https://www.projectsof8051.com/io t-based-fire-alerting-system-project/

6 Fire protection systems
Website:https://fire.nv.gov/uploadedfiles/fir envgov/content/bureaus/FST/4-ifippPSsm.pdf
Date of publication: July 7 2012

2.3 PROBLEM STATEMENT DEFINITION

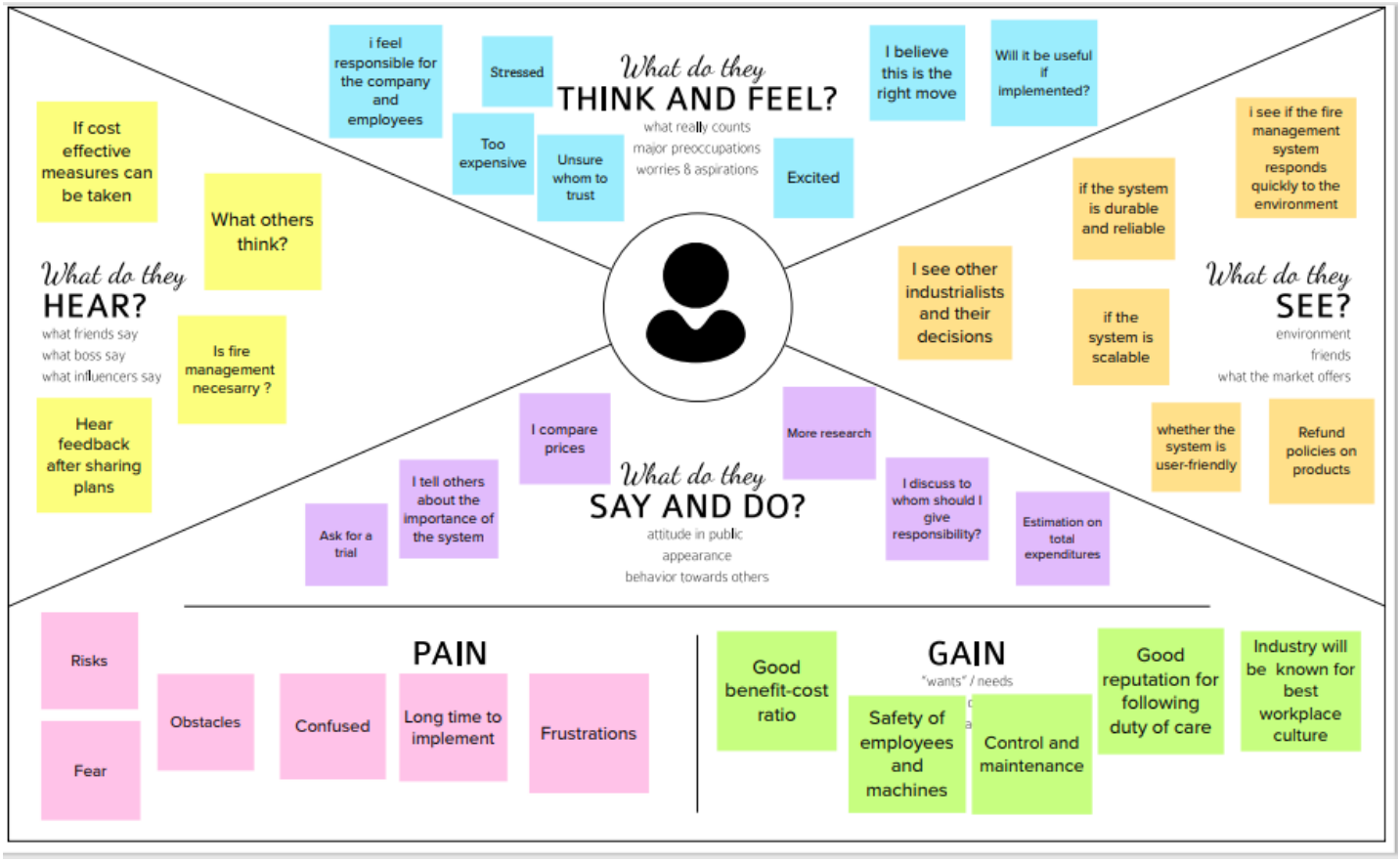
The main reason to implement the fire management system is to avoid fire accidents and deaths due to these accidents. Carelessness of some workers is one of the reasons for the industry fire accidents. Delayed response from the fire station is considered to be another factor of cause.

3. IDEATION & PROPOSED SOLUION

The ideas were specified by conducting a brainstorm and sort the top 3 ideas based on feasibility and importance. The proposed solution that consists of novelty, feasibility of idea, social impact, business model etc., are documented.

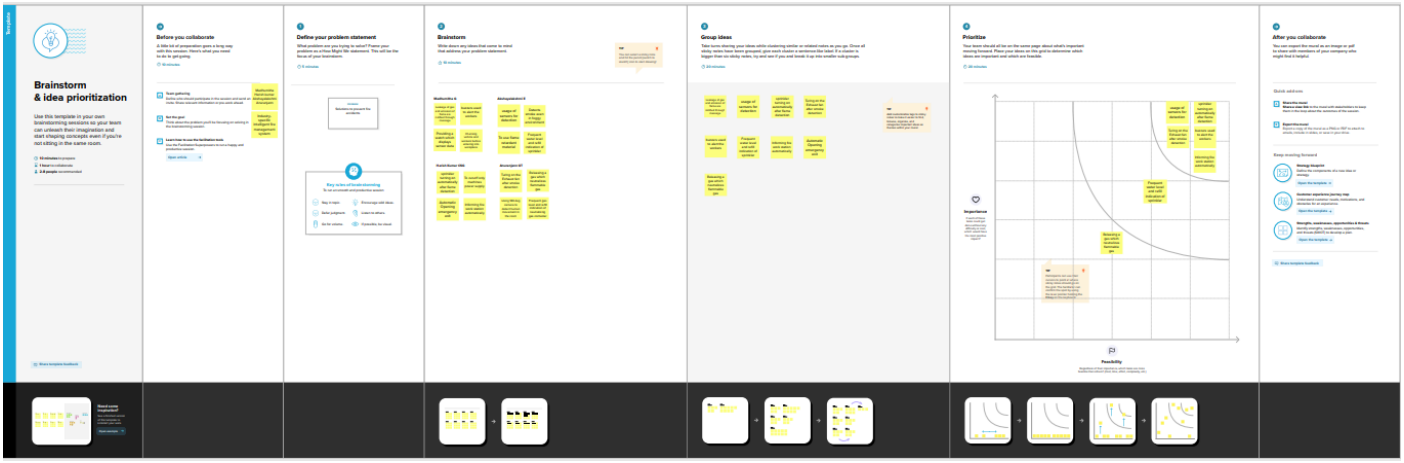
3.1 EMPATHY MAP CANVAS

The empathy maps are created to articulate and capture the customers emotions towards the product. To know how the buyers perceive the product, empathy maps are designed. The empathy maps are constructed in the canvas software.



3.2 IDEATION AND BRAINSTORMING

Several ideas related to the project are gathered by conducting brainstorming sessions. The ideas that are important and can be implemented are prioritized from the session.



3.3 PROPOSED SOLUTION

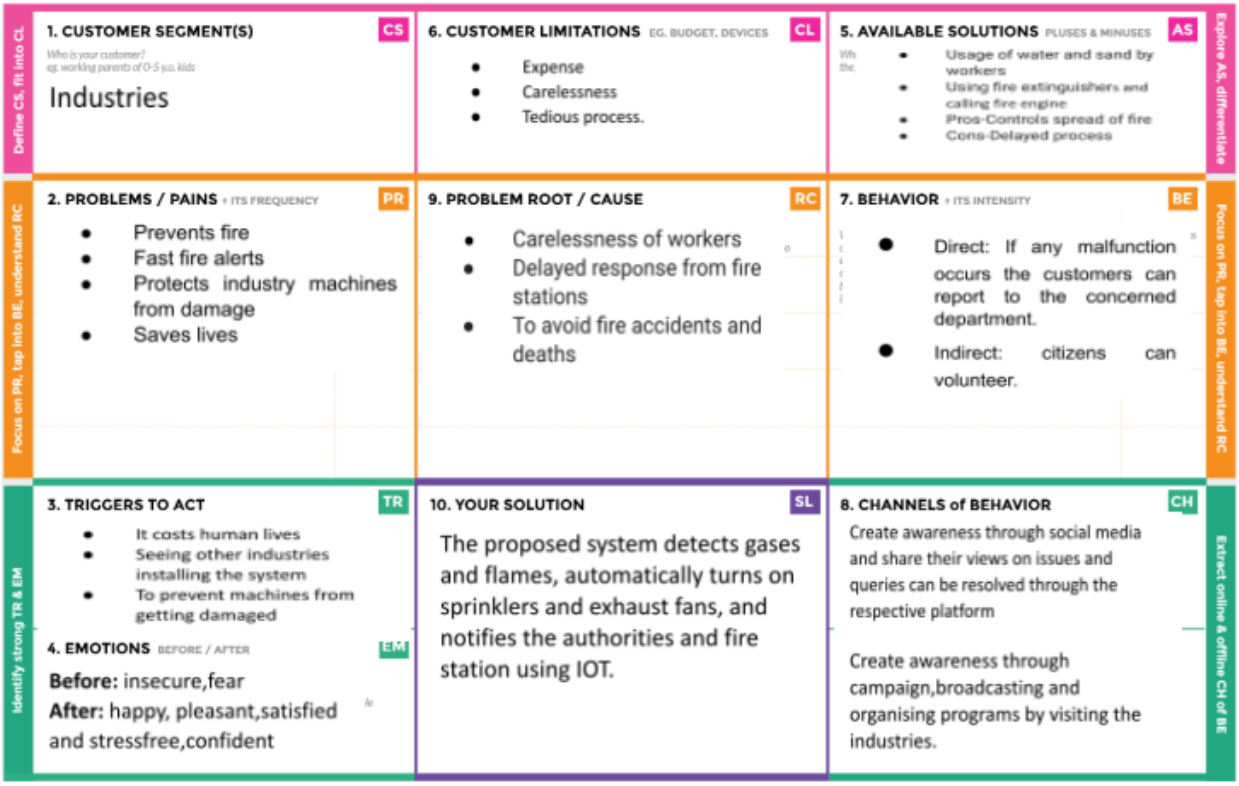
A document which includes novelty, feasibility and scalability of the solution was prepared.

Proposed Solution Template:

Project team shall fill the following information in the proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	This project deals with the problem of fire management in industries, where the fire prevention system is not optimized. This project enables the organizations to meet their needs of fire management systems. This system reduces the risk of fire by detecting and preventing it and also ensures the safety of buildings and workers.
2.	Idea / Solution description	The key research objectives are as follows: • The proposed system detects gases and flames, automatically turns on sprinklers and exhaust fans, and notifies the authorities and fire station. • The Proposed system consists of gas and flame sensors, sprinklers, exhaust fans, and GSM (Global System for Mobile communication). • In the proposed system, environmental changes are acknowledged by placing the sensors at the required position, which automatically turns on sprinklers and exhaust fans and alerts the authorities and fire stations.
3.	Novelty / Uniqueness	The uniqueness of this system is that it opens sprinklers and exhaust fans automatically after the sensor's detection and also informs the authorities and fire station without any human intervention.
4.	Social Impact / Customer Satisfaction	Managers and employees aren't aware of the risks that surround them at work every day. Social impacts are fatal accidents that have snuffed out innocent lives, blocked roads and railway lines, electricity, mobile and land telephone lines cut, destruction of homes and industries.
5.	Business Model (Revenue Model)	The global fire protection systems market size is estimated to surpass USD 70 billion by 2027.
		This growth is attributed to the rising product demand in light of the surging number of fire accidents due to the lack of fire protection systems at heritage sites. Therefore surging government standards and regulations for the improvement of the safety of individuals and industrial, commercial and residential property during fire outbreaks will augment fire protection systems market share over the assessment period.
6.	Scalability of the Solution	Fire management system is highly scalable in various aspects. Following this approach, this idea presented an efficient IoT-based and real-time fire management model for improving the safety of industries and workers. This system can be scaled by adding various technologies to it. More sensors can be used to detect flame and gas even in undesirable conditions. This can also be implemented in residential buildings, schools and colleges. Mobile apps and websites can also be used to monitor the areas frequently.

3.4 PROBLEM SOLUTION FIT



4. REQUIREMENT ANALYSIS

Requirement analysis is the process of determining user needs or conditions for a product.

4.1 FUNCTIONAL REQUIREMENTS

The functional requirements describes the features and functions of the product.

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Home Page	Description about the fire management system Guidelines to use the fire management system
FR-2	User Registration	Registration through website. Registration through application.
FR-3	User Confirmation	Confirmation via Email or OTP.
FR-4	User Login	Login through website or app. Access the app using respective username or password.
FR-5	User upload	Should be able to upload the data.
FR-6	System behavior	The sensors within the system are used for detection. The data gets updated in the cloud.
FR-7	System functionality	If flame is detected, sprinklers are turned on automatically. If gas is detected, exhaust fans are turned on automatically.
FR-8	Notification	The fire workstation and industry authorities are notified through message.

4.2 NON FUNCTIONAL REQUIREMENTS

Non-functional requirements describes the quality attributes of a system.

Non-functional Requirements:

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

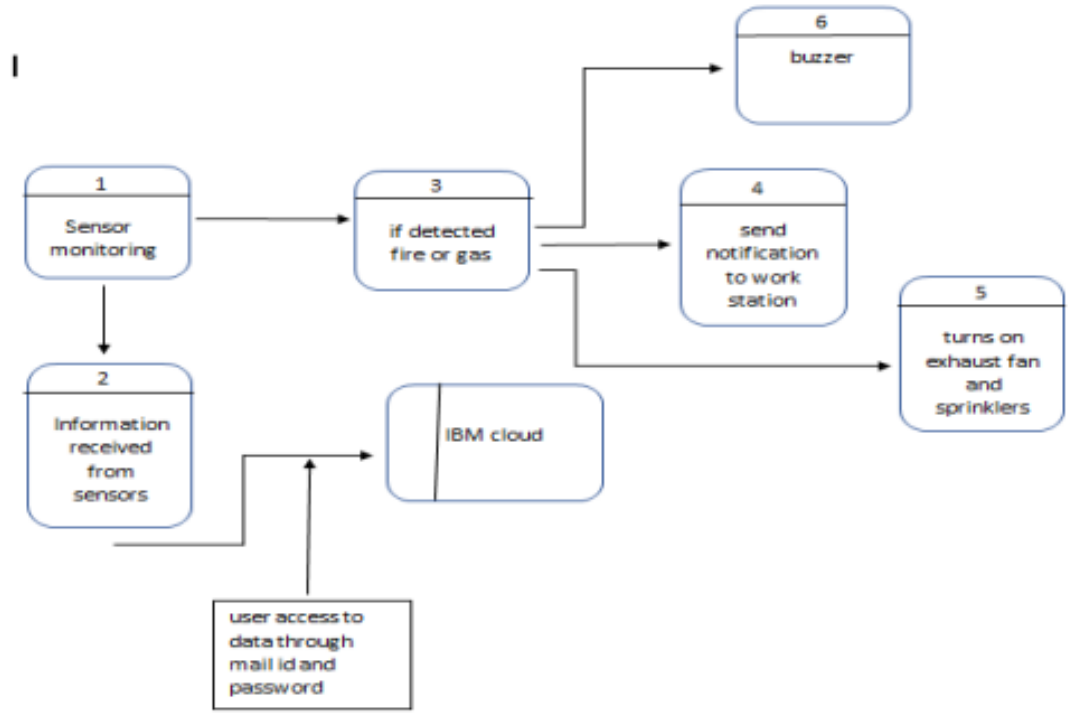
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The system is user friendly and performs instant action which is important in case of fire management.
NFR-2	**Security**	The system is able to withstand any abnormal condition. The web application is more secure.
NFR-3	**Reliability**	The system is highly reliable. The notification is sent even in case of Wi-Fi malfunction.
NFR-4	**Performance**	The sprinklers and exhaust fan starts to function and the notification is sent immediately after fire detection .
NFR-5	**Availability**	Any industries who are in need of fire management system can access the system. The system works at full length.
NFR-6	**Scalability**	More technologies can be added. This system can be implemented in large scale.

5.PROJECT DESIGN

Project design is the process of planning out the ideas, procedure and deliverables of the project.

5.1 DATA FLOW DIAGRAMS

This process maps out the flow of information and describes the system's operation.



5.2 SOLUTION AND TECHNICAL ARCHITECTURE

SOLUTION ARCHITECTURE:

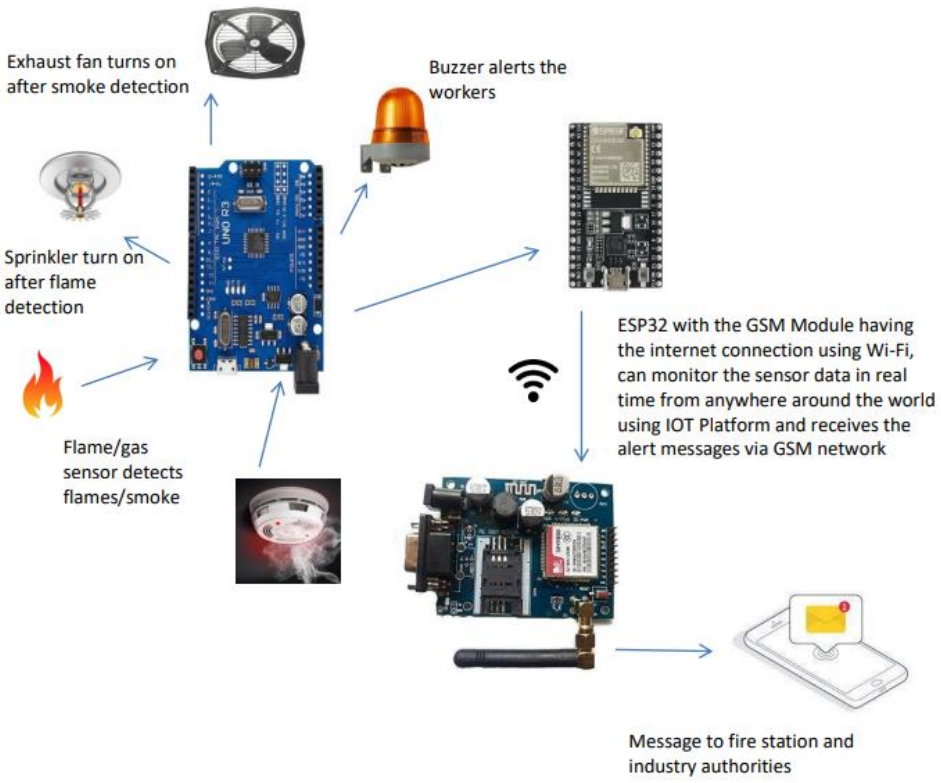
Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

• Find the best tech solution to solve existing business problems.

• Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.

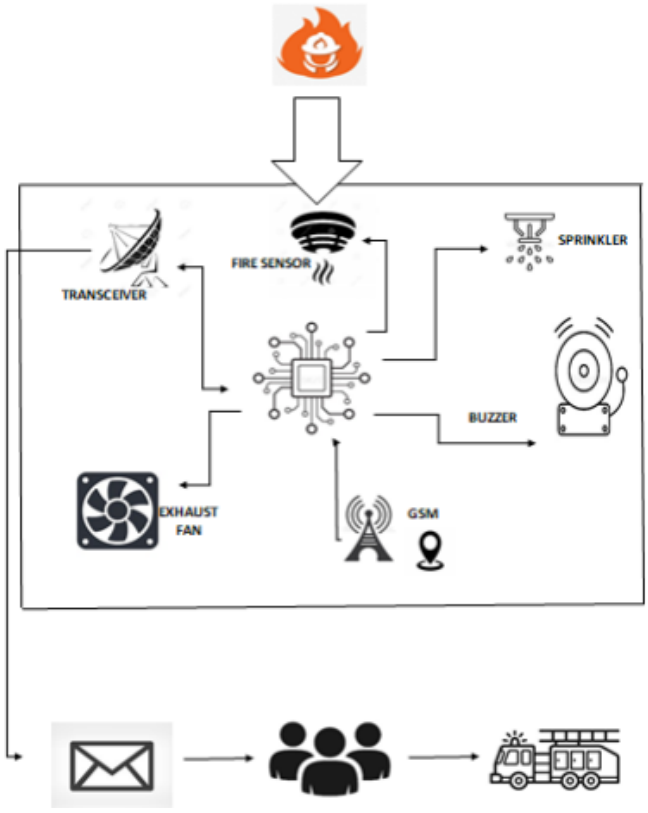
• Define features, development phases, and solution requirements.

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



TECHNICAL ARCHITECTURE

Technical architecture describes the plan, design, build, implementation and maintenance of the system.



5.3 USER STORIES

User stories is an informal explanation of a system feature written from the perspective of the customers.

User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Primary admin	login	USN-1	As a primary admin , I can buy and install the system and manage the webserver	I can access account / dashboard	Medium	Sprint-2
Secondary admin	login	USN-2	As a secondary admin, I can Monitor thesensor readings	I can access the data	High	Sprint-1
Industry employee	dashboard	USN-3	As an industry employee I can monitor the correct functional of detectors ,sprinklers and exhaust fans	I can access thesystem	Medium	Sprint-2
Fire engine driver	dashboard	USN-4	As an fire fighter, I reach to the correct destination	I can access thelocation	Medium	Sprint-2
Industry Manager	Login	USN-5	As a manager, we will look over the smooth management of the whole process	I can monitor the whole system	High	Sprint-1

6.PROJECT PLANNING & SCHEDULING

The process of planning deals with selecting the appropriate procedures for the project. Project scheduling consists of assigning start and end dates to individual tasks and allocating the resources within an estimated budget.

6.1 SPRINT PLANNING & ESTIMATION

This phase involves planning the project and estimating the duration within which the project can be completed.

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the product by entering my email and password.	1	High	Madhumitha G
Sprint-1	Flame sensor	USN-3	As a user, I will purchase the product which consists of flame sensors.	2	High	Akshayalakshmi E
Sprint-1	Sprinkler	USN-4	As a user, I can observe that the sprinkler turns on automatically as soon as the flame is detected.	1	Medium	Harish Kumar C S G
Sprint-2	Gas sensor	USN-5	As a user, I will purchase the product which consists of gas sensors.	1	Medium	Harish Kumar C S G
Sprint-2	Exhaust fan	USN-6	As a user, I can observe that the exhaust fan turns on automatically as soon as the gas is detected.	2	High	Anuranjann G T
Sprint-3	Buzzer	USN-7	As a user, I can observe that the buzzer rings after the detections in order to alert the people.	1	Medium	Akshayalakshmi E
Sprint-3	Fast SMS	USN-8	As a user, I will receive alert notification and also be notified to fire work station.	2	High	Anuranjann G T
Sprint-4	Final Deliverable	USN-9	As a user, I experience a safe environment with the help of this fire management system.	2	High	Madhumitha G

6.2 SPRINT DELIVERY SCHEDULE

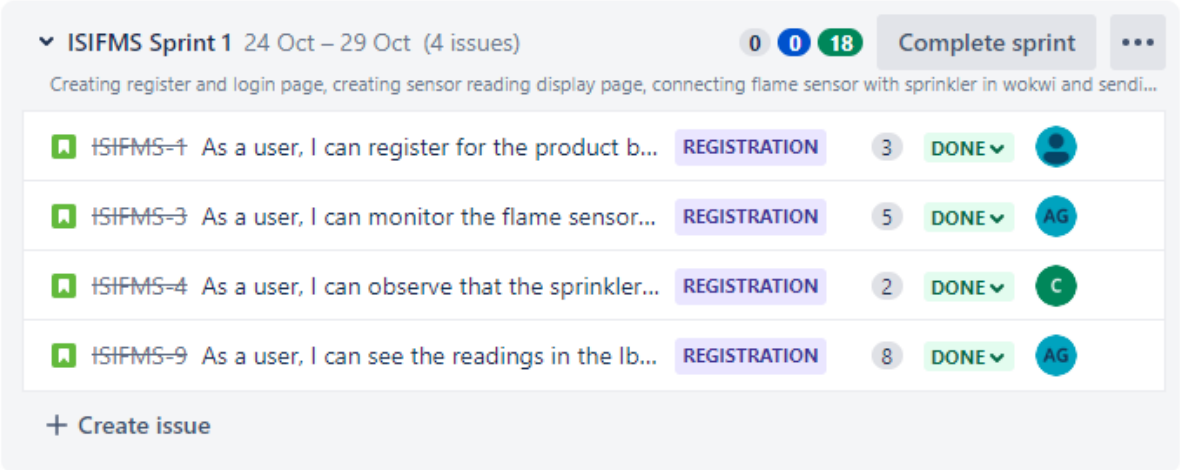
Sprint delivery schedule estimates the start date and release date of the sprints.

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

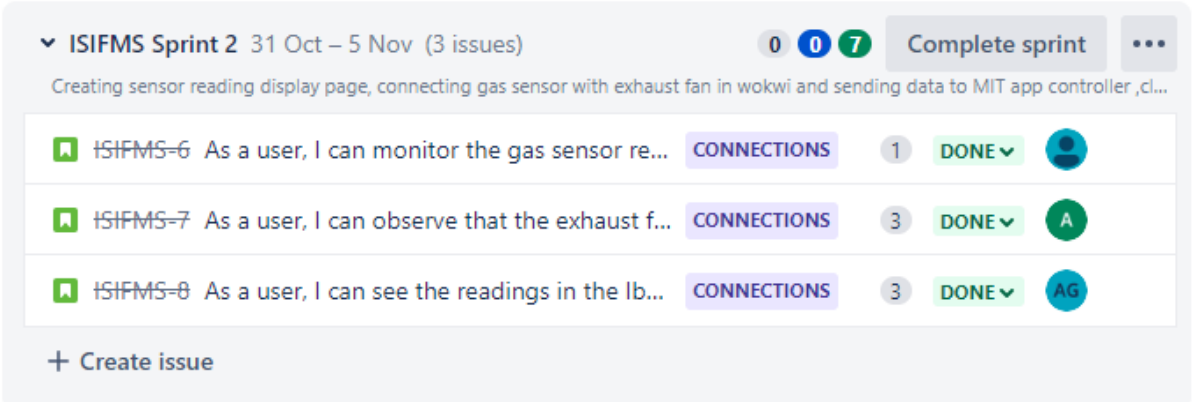
6.3 REPORTS FROM JIRA

BACKLOG

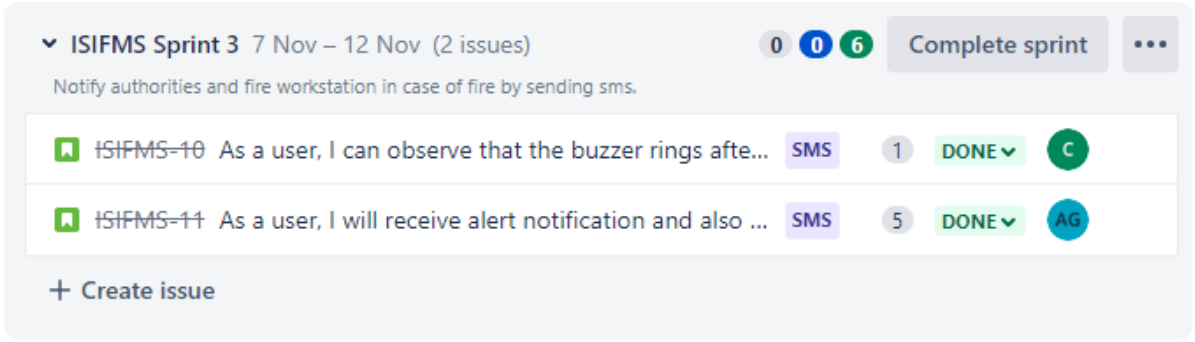
SPRINT 1



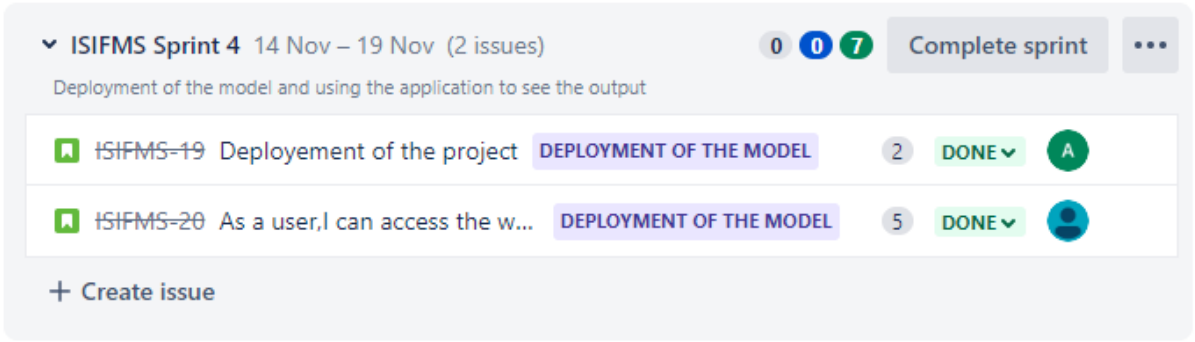
SPRINT 2



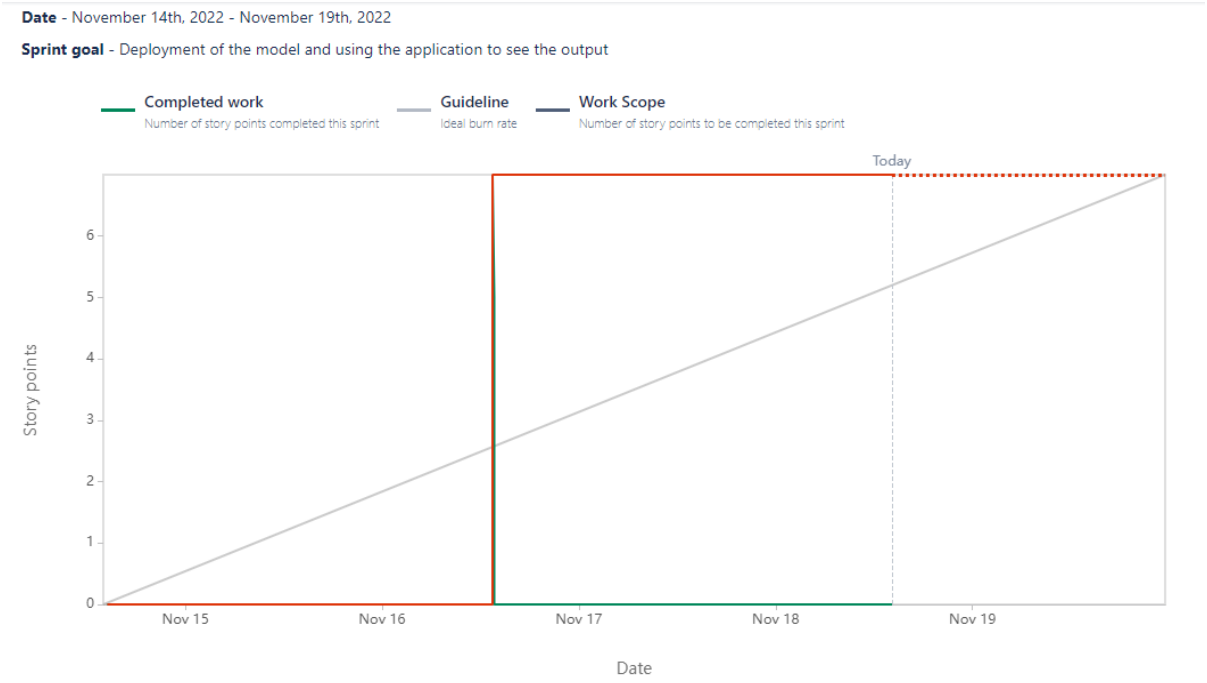
SPRINT 3



SPRINT 4

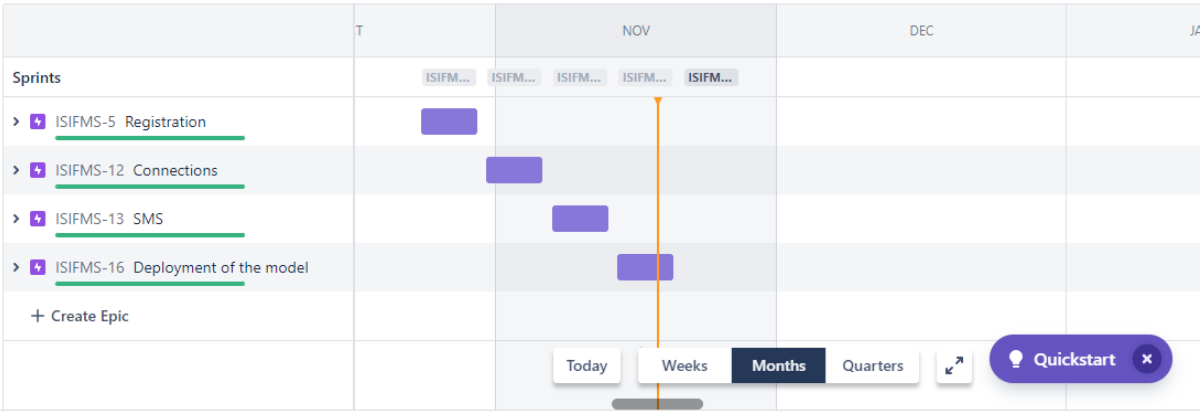


BURNUP REPORT

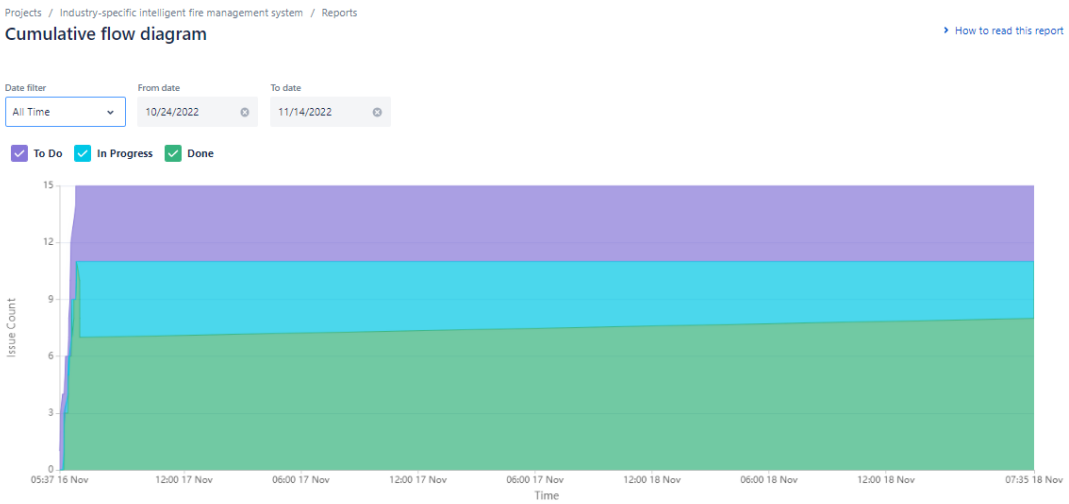


VELOCITY REPORT

ROADMAP



CUMULATIVE FLOW DIAGRAM



7. CODING AND SOLUTIONING

FEATURE 1: FLAME SENSOR, TEMPERATURE SENSOR AND GAS SENSOR

```
#include <WiFi.h>//library for wifi

#include <PubSubClient.h>//library for MQtt

#include <ESP32Servo.h>

#include "DHT.h"// Library for dht11

#include <Stepper.h>

#define DHTPIN 5    // what pin we're connected to

#define DHTTYPE DHT22   // define type of sensor DHT 11

#define SERVO_PIN 22 //servo motor connection

#define BUZZER_PIN 2//buffer connecton

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht connected

Servo servoMotor;

Servo servoMotor2;

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);


//-------credentials of IBM Accounts------


#define ORG "py0epl"//IBM ORGANITION ID

#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform

#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "12345678"     //Token

const int tempHigh=50;

const int firingHigh = 90;
```

```
const int gasHigh=400;
String gasData;
String flameData;
String tempData;
 float templevel=0;
  float flamelevel;
const int stepsPerRevolution = 200;  //
Stepper myStepper(stepsPerRevolution, 13, 12, 14, 26);


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


//-------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by passing parameter like server id,portand wificredential


void setup()// configureing the ESP32
{
  Serial.begin(115200);
 myStepper.setSpeed(60);
```

```
  pinMode(BUZZER_PIN, OUTPUT);
  servoMotor.attach(SERVO_PIN);
  dht.begin();
 delay(10);
 Serial.println();
 wificonnect();
 mqttconnect();
}

void loop()// Recursive Function
{
 templevel= dht.readTemperature();
 float analogValue = analogRead(36);
 float gaslevel=0;
 gaslevel = random(100,900);
 Serial.print(gaslevel);
 Serial.println("Sensor RAW: ");
 Serial.println(analogValue, 0);
flamelevel = map(analogValue, 0, 1024, 100, 0);
 Serial.print(flamelevel, 0);
 Serial.println("%");
 if (flamelevel >= firingHigh ) { // stoker is fully firing
 tone(BUZZER_PIN,2000);
  servoMotor.write(180);
   delay(300);
   flameData="alert";
 }
 else{
```

```
  flameData="safe";
  noTone(BUZZER_PIN);
   servoMotor.write(0);
}
if(gaslevel>= gasHigh){
  tone(BUZZER_PIN,2000);
    myStepper.step(stepsPerRevolution);
  delay(300);
  gasData="alert";
}
else{
  gasData="safe";
      myStepper.step(-stepsPerRevolution);
  noTone(BUZZER_PIN);

}
 if(templevel>= tempHigh){
  tone(BUZZER_PIN,2000);
  delay(300);
  tempData="alert";
}
else{
  tempData="safe";
  noTone(BUZZER_PIN);
}

PublishData(gaslevel,flamelevel,templevel);
delay(1000);
```

```
  if (!client.loop()) {
    mqttconnect();
  }
}


/*.....retrieving to Cloud.....*/

void PublishData(float gaslevel,float flamelevel,float templevel) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */
  String payload = "{\"gaslevel\":";
  payload += gaslevel;
  //payload += "," "\"GasMsg\":";
  //payload += gasData;
  payload += "," "\"flamelevel\":";
  payload += flamelevel;
  //payload += "," "\"FlameMsg\":";
  //payload += flameData;
  payload += "," "\"templevel\":";
  payload += templevel;
  //payload += "," "\"TemperatureMsg\":";
  //payload += tempData;
  payload += "}";
  Serial.print("Sending payload: ");
  Serial.println(payload);
```

```
  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it will print publish ok in Serial monitor or else it will print publish failed
  } else {
    Serial.println("Publish failed");
  }


}
void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }
     initManagedDevice();
     Serial.println();
  }
}
void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");


  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection
  while (WiFi.status() != WL_CONNECTED) {
```

```
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{
  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
}
```

EXPLANATION:

- Above set of code is used for the detection of flame and turns on sprinklers,detection of gas and turns on exhaust fan, detection temperature and in all cases turns on buzzer.
- This code is also used for sending data to cloud.

FEATURE 2:

```
#include <WiFi.h>//library for wifi

#include <PubSubClient.h>//library for MQtt

#include <ESP32Servo.h>

#include "DHT.h"// Library for dht11

#include <Stepper.h>

#define DHTPIN 5     // what pin we're connected to

#define DHTTYPE DHT22   // define type of sensor DHT 11

#define SERVO_PIN 22 //servo motor connection

#define BUZZER_PIN 2//buffer connecton

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht connected

Servo servoMotor;

Servo servoMotor2;

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);


//-------credentials of IBM Accounts------

#define ORG "py0epl"//IBM ORGANITION ID

#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform

#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "12345678"     //Token

const int tempHigh=50;

const int firingHigh = 90;
```

```
const int gasHigh=400;

String gasData;

String flameData;

String tempData;

 float templevel=0;

 float flamelevel;

const int stepsPerRevolution = 200;  //

Stepper myStepper(stepsPerRevolution, 13, 12, 14, 26);


//-------- Customise the above values --------

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name

char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in which data to be send

char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING

char authMethod[] = "use-token-auth";// authentication method

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id



//-------------------------------------

WiFiClient wifiClient; // creating the instance for wificlient

PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by passing parameter like server id,portand wificredential
```

```
void setup()// configureing the ESP32
{
  Serial.begin(115200);
 myStepper.setSpeed(60);
   pinMode(BUZZER_PIN, OUTPUT);
   servoMotor.attach(SERVO_PIN);
   dht.begin();
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}
void loop()// Recursive Function
{
  //int steps=200000;
  templevel= dht.readTemperature();
  float analogValue = analogRead(36);
  float gaslevel=0;
  gaslevel = random(100,900);
  Serial.print(gaslevel);
  Serial.println("Sensor RAW: ");
  Serial.println(analogValue, 0);
 flamelevel = map(analogValue, 0, 1024, 100, 0);
  Serial.print(flamelevel, 0);
  Serial.println("%");
  if (flamelevel >= firingHigh ) { // stoker is fully firing
```

```
tone(BUZZER_PIN,2000);
 servoMotor.write(180);
  delay(300);
  flameData="alert";
}
else{
  flameData="safe";
  noTone(BUZZER_PIN);
   servoMotor.write(0);
}
Serial.print("Flame Message sending to authority :");
Serial.println(flameData);
if(gaslevel>= gasHigh){
  tone(BUZZER_PIN,2000);
    myStepper.step(stepsPerRevolution);
  delay(300);
  gasData="alert";
}
else{
  gasData="safe";
      myStepper.step(-stepsPerRevolution);
  noTone(BUZZER_PIN);

}
Serial.print("Gas Message sending to authority :");
Serial.println(gasData);
```

```
 if(templevel>= tempHigh){
  tone(BUZZER_PIN,2000);
  delay(300);
  tempData="alert";
 }
 else{
  tempData="safe";
  noTone(BUZZER_PIN);
 }
 Serial.print("Temperature Message sending to authority :");
 Serial.println(tempData);
  PublishData(gaslevel,flamelevel,templevel);
 delay(1000);
 if (!client.loop()) {
  mqttconnect();
 }
}

/.....retrieving to Cloud...../
void PublishData(float gaslevel,float flamelevel,float templevel) {
 mqttconnect();//function call for connecting to ibm
 /*
   creating the String in in form JSon to update the data to ibm cloud
 */
 String payload = "{\"gaslevel\":";
 payload += gaslevel;
```

```
  //payload += "," "\"GasMsg\":";
  //payload += gasData;
  payload += "," "\"flamelevel\":";
  payload += flamelevel;
  //payload += "," "\"FlameMsg\":";
  //payload += flameData;
  payload += "," "\"templevel\":";
  payload += templevel;
  //payload += "," "\"TemperatureMsg\":";
  //payload += tempData;
  payload += "}";
  Serial.print("Sending payload: ");
  Serial.println(payload);
    if (client.publish(publishTopic, (char*) payload.c_str())) {
   Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it
will print publish ok in Serial monitor or else it will print publish failed
  } else {
    Serial.println("Publish failed");
  }
 }

void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
```

```
      delay(500);
    }
     initManagedDevice();
    Serial.println();
  }
}
void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
the connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
```

```
   Serial.println("subscribe to cmd FAILED");
  }
}
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{
  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  }
```

EXPLANATION:

- This code also detects,prevents any fire accidents,sends data to authotity and sends data to cloud.

EXTRA FEATURE ADDED IN APP DEVELOPMENT-TINYDB

- Tinydb is used to save the user credentials such as user id and password even if the user navigates to another screen
- The user id will be displayed in the user text box as it is saved
- Tinydb is used to save all the sensor readings in the same ID.

DATABASE SCHEMA-

ID:String/Number/Symbols

Password: String/Number/Symbols

Gaslevel:Float

Flamelevel:Float

Temperaturelevel:Float

Msg1:String

Msg:String

Eg. {

"_id": "2022TMID-27908",

"_rev": "434-58699e9fd80cce86bb2497e2199f5f17",

"password": "27908mid",

"gaslevel": 702,

"flamelevel": 3,

"templevel": 80,

"msg1": "logged in",

"msg": "registered"

}

8. TESTING

8.1 TESTCASES

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N	BUG ID	Executed By
RegistrationPage_TC_004	Functional	Registration page	Verify user is able to create an Id by registering if not created one.	User should install the application.	1.To register, enter the following credentials: a.Username text box b.Password text box c.Click submit to register d. Click submit to sign in	Username: abcd12 password: 1234	Application should display 'Check User ID or Register' popup if user tries to sign in without registering.	Working as expected	Pass		Y		Madhumitha
Sensor_005	Functional	Microcontrolle	Sensor data is properly taken	Check if the system is in	Activate the sensor	Sensor values are	Sending the value to the users	Working as	Pas		Y		Akshayalakshmi
Sensor_006	Functional	Microcontrolle	Sensor data is properly taken	Check if the system is in	Activate the sensor	Sensor values are	Sending the value to the users	Working as	Pas		Y		Harish Kumar
Sensor_007	Functional	Microcontroller	Sensor data is properly taken	heck if the system is in active st	Activate the sensor	ensor values are generate	Sending the value to the users a	Working as expected	Pass		Y		Anuranjann
Cloud	storage	watson cloud	sensor data is displayed	check if the system is on	Activate the sensor		sensor values are displayed	working as expected	Pass		Y		Anuranjann
Database_008	Storage	Cloudant	The received data is stored in	lode red is connected with the cl	or readings are stored automatically		readings shown in database with	Working as expected	Pass		Y		Akshayalakshmi
SMS_009	API	sms API	the sms is sent incase of fire	The Node red should be	or 'Alert' message is sent autom	Alert' or 'safe' message	Alert and safe message is sent	Working as	Pas		Y		Harish Kumar

8.2 USER ACCEPTANCE TESTING

DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	9	0	2	1	12
Duplicate	1	0	3	0	4
External	0	0	1	0	1
Fixed	19	24	25	14	82
Not Reproduced	0	0	2	0	2
Skipped	0	0	0	0	0
Won't Fix	0	5	0	0	0
Totals	28	24	30	15	97

TEST CASE ANALYSIS

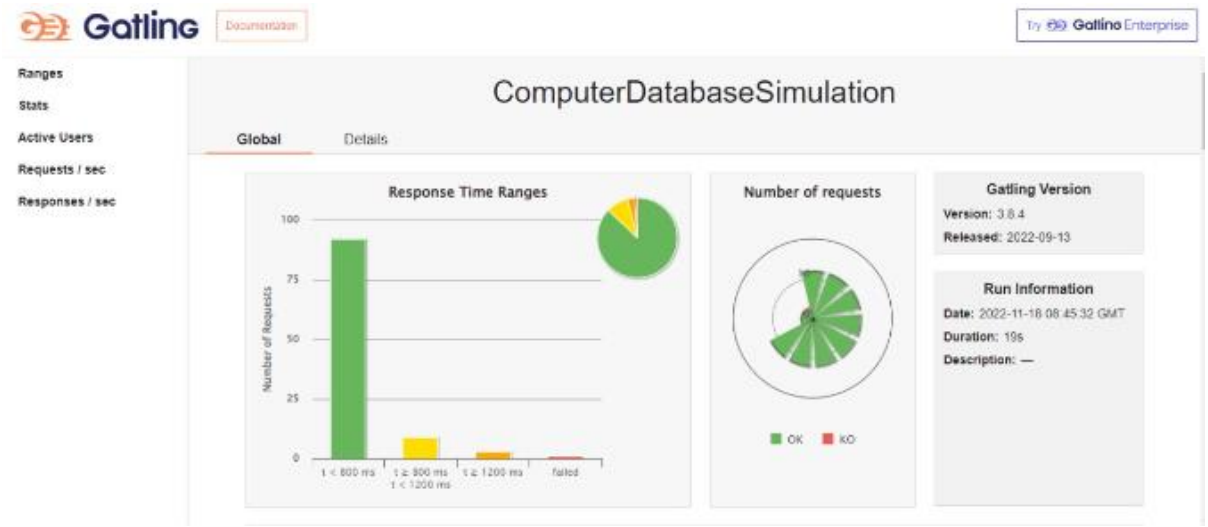
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	4	0	0	4
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	11	0	0	11
Final Report Output	5	0	0	5
Version Control	2	0	0	2

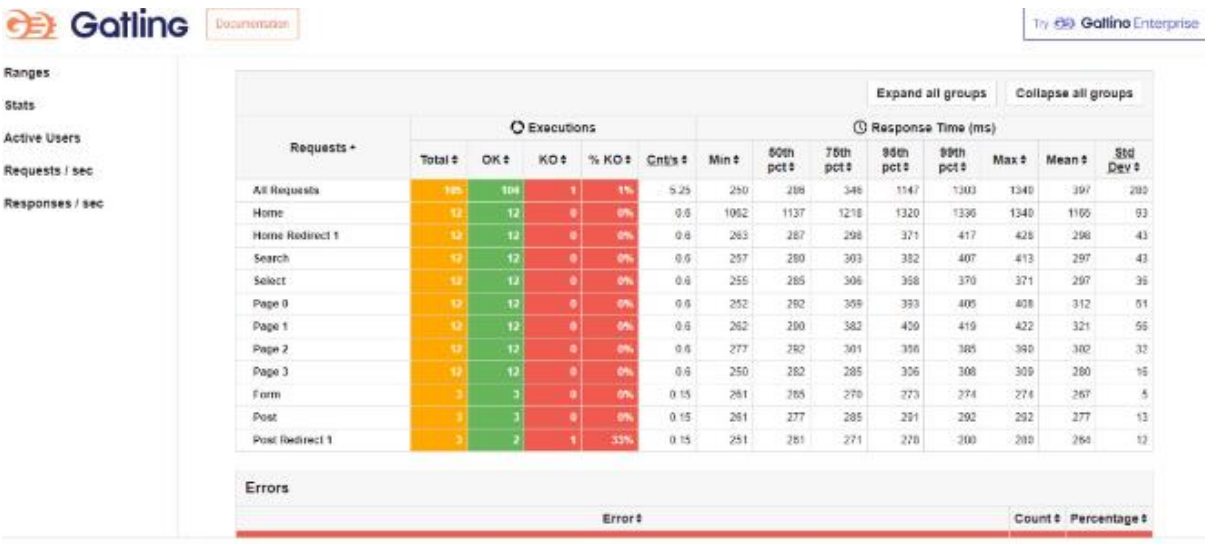
9.RESULTS

9.1 PERFORMANCE METRICS

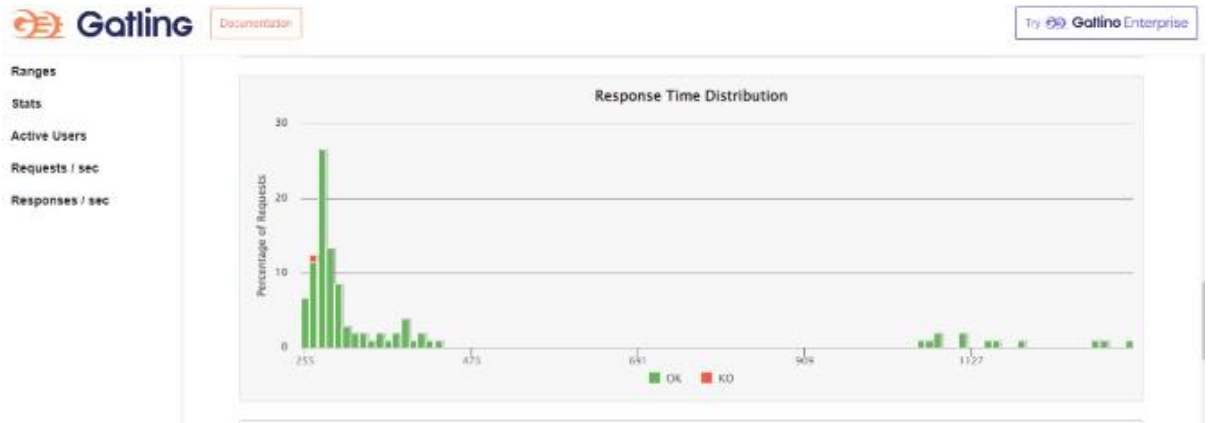
PERFORMANCE TESTING:

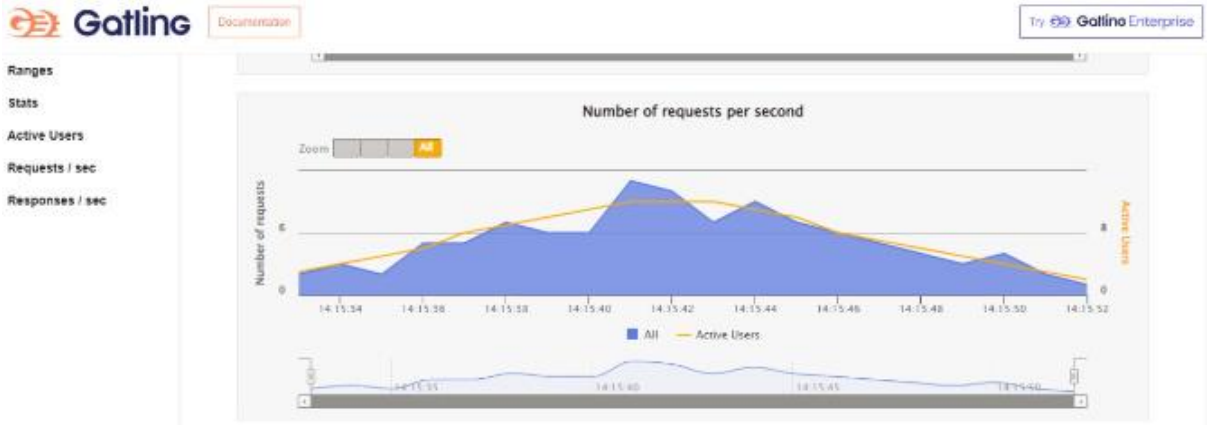
Performance testing is done to evaluate how the system works in terms of responsiveness and stability.

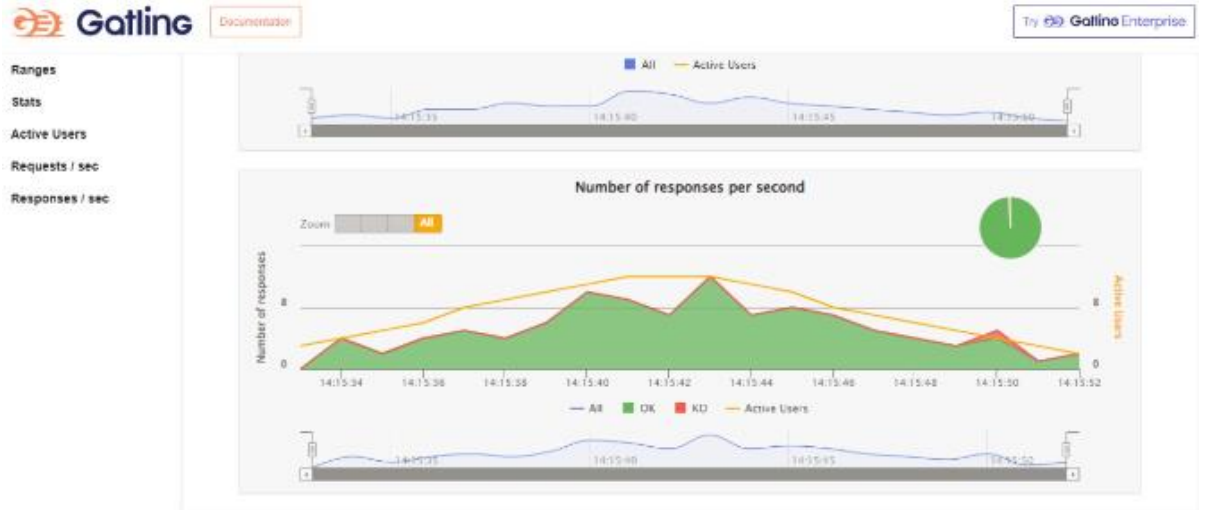



Gatling
Documentation
Try Gatling Enterprise
Ranges
Stats
Active Users
Requests / sec
Responses / sec
Expand all groups
Collapse all groups
Requests
Executions
Response Time (ms)
Total
OK
KO
% KO
Cnt/s
Min
50th pct
75th pct
95th pct
99th pct
Max
Mean
Std Dev
All Requests 105 104 1 1% 5.25 250 288 348 1147 1303 1340 397 280
Home 12 12 0 0% 0.6 1062 1137 1218 1320 1336 1340 1165 93
Home Redirect 1 12 12 0 0% 0.6 263 287 298 371 417 426 298 43
Search 12 12 0 0% 0.6 257 280 303 382 407 413 297 43
Select 12 12 0 0% 0.6 255 285 306 358 370 371 297 36
Page 0 12 12 0 0% 0.6 252 282 359 393 405 408 312 51
Page 1 12 12 0 0% 0.6 262 290 382 409 419 422 321 56
Page 2 12 12 0 0% 0.6 277 282 301 356 385 390 302 32
Page 3 12 12 0 0% 0.6 250 282 285 306 308 309 280 16
Form 3 3 0 0% 0.15 261 265 270 273 274 274 267 5
Post 3 3 0 0% 0.15 261 277 285 291 292 292 277 13
Post Redirect 1 3 2 1 33% 0.15 251 261 271 278 280 280 264 12
Errors
Error
Count
Percentage


Gatling
Documentation
Try Gatling Enterprise
Ranges
Stats
Active Users
Requests / sec
Responses / sec
Errors
Error
Count
Percentage
status.find.is(201), but actually found 200 1 100 %
Active Users along the Simulation
Admins
Users
Active Users
Zoom
All
Number of Active Users
10
5
0
14:15:34
14:15:36
14:15:38
14:15:40
14:15:42
14:15:44
14:15:46
14:15:48
14:15:50
14:15:52


Gatling
Documentation
Try Gatling Enterprise
Ranges
Stats
Active Users
Requests / sec
Responses / sec
Response Time Distribution
Percentage of Requests
30
20
10
0
255
473
691
909
1127
OK
KO





The performance testing for a project was performed in Gatling.

PERFORMANCE TESTING REPORT

			NFT - Risk Assessment					
S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Load/Voluem Changes	Risk Score	Justification
1	Industry specific	Existing	Low	No Changes	Moderate	>5 to 10%	ORANGE	As we have seen the chnages
2	IIndustry Industr	Existing	Low	Low	Moderate	>5 to 10%	ORANGE	As we have seen the chnages

	NFT - Detailed Test Plan			
S.No	Project Overview	NFT Test approach	mptions/Dependencies/R	Approvals/SignOff
1	Industry specific intelligent f	load	if login page crashes/soft	approved
2	Industry specific intelligent f	stress	if fire system crashes/ha	approved

			End Of Test Report					
S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	(Detected/Closed/Open)	Approvals/SignOff
1	Industry specific	load	Met	system working well,by usin	GO	-	closed	approved
2	Industry specific	stress	Met	system working well in all er	GO	-	closed	approved

10. ADVANTAGES

- Monitoring constantly for fire outbreaks.
- Sprinklers that turn on and off automatically.
- The exhaust fan turns on and off automatically.
- Automatic SMS messages are sent to the fire station and industry authorities.
- User can monitor the status of the system through the app.

DISADVANTAGES

- A constant internet connection is required for sending alert messages.
- The entire operation gets disturbed if the system gets damaged.

11. CONCLUSION

The industry-specific intelligent fire management system provides a solution to fire accidents that occur in industries which results in the loss of life and millions of rupees worth of property. To save time, it also sends alarm messages to the fire stations and industrial authorities in place of personally calling them.

12. FUTURE SCOPE

The present device can be adjusted to work for house use to big companies because fire incidents can result in significant loss of human life in both houses and large industries.

13.APPENDIX

SOURCE CODE

```
#include <WiFi.h>//library for wifi

#include <PubSubClient.h>//library for MQtt

#include <ESP32Servo.h>

#include "DHT.h"// Library for dht11

#include <Stepper.h>

#define DHTPIN 5     // what pin we're connected to

#define DHTTYPE DHT22   // define type of sensor DHT 11

#define SERVO_PIN 22 //servo motor connection

#define BUZZER_PIN 2//buffer connecton

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht connected

Servo servoMotor;

Servo servoMotor2;

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);


//-------credentials of IBM Accounts------

#define ORG "py0epl"//IBM ORGANITION ID

#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform
```

```
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
const int tempHigh=50;
const int firingHigh = 90;
const int gasHigh=400;
String gasData;
String flameData;
String tempData;
 float templevel=0;
 float flamelevel;
const int stepsPerRevolution = 200;  //
Stepper myStepper(stepsPerRevolution, 13, 12, 14, 26);


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id



//-------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
```

PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by passing parameter like server id,portand wificredential

```
void setup()// configureing the ESP32
{
  Serial.begin(115200);
 myStepper.setSpeed(60);
   pinMode(BUZZER_PIN, OUTPUT);
   servoMotor.attach(SERVO_PIN);
   dht.begin();
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}
void loop()// Recursive Function
{
  //int steps=200000;
  templevel= dht.readTemperature();
  float analogValue = analogRead(36);
  float gaslevel=0;
  gaslevel = random(100,900);
  Serial.print(gaslevel);
  Serial.println("Sensor RAW: ");
  Serial.println(analogValue, 0);
 flamelevel = map(analogValue, 0, 1024, 100, 0);
```

```
Serial.print(flamelevel, 0);
Serial.println("%");
if (flamelevel >= firingHigh ) { // stoker is fully firing
tone(BUZZER_PIN,2000);
 servoMotor.write(180);
  delay(300);
  flameData="alert";
}
else{
  flameData="safe";
  noTone(BUZZER_PIN);
   servoMotor.write(0);
}
Serial.print("Flame Message sending to authority :");
Serial.println(flameData);
if(gaslevel>= gasHigh){
  tone(BUZZER_PIN,2000);
    myStepper.step(stepsPerRevolution);
  delay(300);
  gasData="alert";
}
else{
  gasData="safe";
      myStepper.step(-stepsPerRevolution);
  noTone(BUZZER_PIN);
```

```
 }
 Serial.print("Gas Message sending to authority :");
 Serial.println(gasData);
  if(templevel>= tempHigh){
   tone(BUZZER_PIN,2000);
   delay(300);
   tempData="alert";
 }
 else{
   tempData="safe";
   noTone(BUZZER_PIN);
  }
 Serial.print("Temperature Message sending to authority :");
 Serial.println(tempData);
  PublishData(gaslevel,flamelevel,templevel);
 delay(1000);
 if (!client.loop()) {
   mqttconnect();
 }
}

/.....retrieving to Cloud...../
void PublishData(float gaslevel,float flamelevel,float templevel) {
 mqttconnect();//function call for connecting to ibm
 /*
    creating the String in in form JSon to update the data to ibm cloud
```

```
 */
 String payload = "{\"gaslevel\":";
 payload += gaslevel;
 //payload += "," "\"GasMsg\":";
 //payload += gasData;
 payload += "," "\"flamelevel\":";
 payload += flamelevel;
 //payload += "," "\"FlameMsg\":";
 //payload += flameData;
 payload += "," "\"templevel\":";
 payload += templevel;
 //payload += "," "\"TemperatureMsg\":";
 //payload += tempData;
 payload += "}";
 Serial.print("Sending payload: ");
 Serial.println(payload);
  if (client.publish(publishTopic, (char*) payload.c_str())) {
  Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it
will print publish ok in Serial monitor or else it will print publish failed
 } else {
  Serial.println("Publish failed");
 }
}


void mqttconnect() {
 if (!client.connected()) {
  Serial.print("Reconnecting client to ");
```

```
   Serial.println(server);
   while (!!!client.connect(clientId, authMethod, token)) {
     Serial.print(".");
     delay(500);
   }
     initManagedDevice();
    Serial.println();
  }
}
void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
the connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
```

```
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{
  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  }
```

Github link:

https://github.com/IBM-EPBL/IBM-Project-19469-1659698325

Project Demo link:

https://drive.google.com/file/d/1ihRHwxN8T37IRzS7aIDxLsB3N30k4-6R/view?usp=share_link