Gas Pipeline Monitoring System for Hospitals

Professional Readiness for Innovation, Employability and Entrepreneurship

PROJECT REPORT

Submitted by Team ID: NM2023TMID14262

RICKY CHARLES

YUVARAJ S K

RAKESH KUMAR M

SHYAM SUNDAR R

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

KCG COLLEGE OF TECHNOLOGY, KARAPAKKAM

ANNA UNIVERSITY: CHENNAI 600 025

ABSTRACT

It is essential to maintain proper levels of gas, as it is used in operating theatres in order to improve patient's condition. Oxygen is the most common and important gas that has to be checked at regular intervals for availability. Leakage of these gases can pose a severe threat to the safety of lives in hospitals. Through this smart system, staff will now be able to check the level of gas present for use and will be able to take safety measures when alerted during a gas leak. Sensors are used to record the levels of gases, which are stored in the cloud. These values can also be displayed on a mobile application for the users.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Project Overview

- ➢ Users can monitor the gas levels present in the surrounding environment using mobile application.

- ➢ Alertion is indicated with a buzzer, LED, and LCD Display when the gas level reaches the threshold value.

- ➢ Saving and Visualization of gas, temperature, humidity in IBM Watson IOT Platform.

- ➢ Connection with IOT device and MIT App Inventor is done by using Node Red Flow.

1.2 Purpose

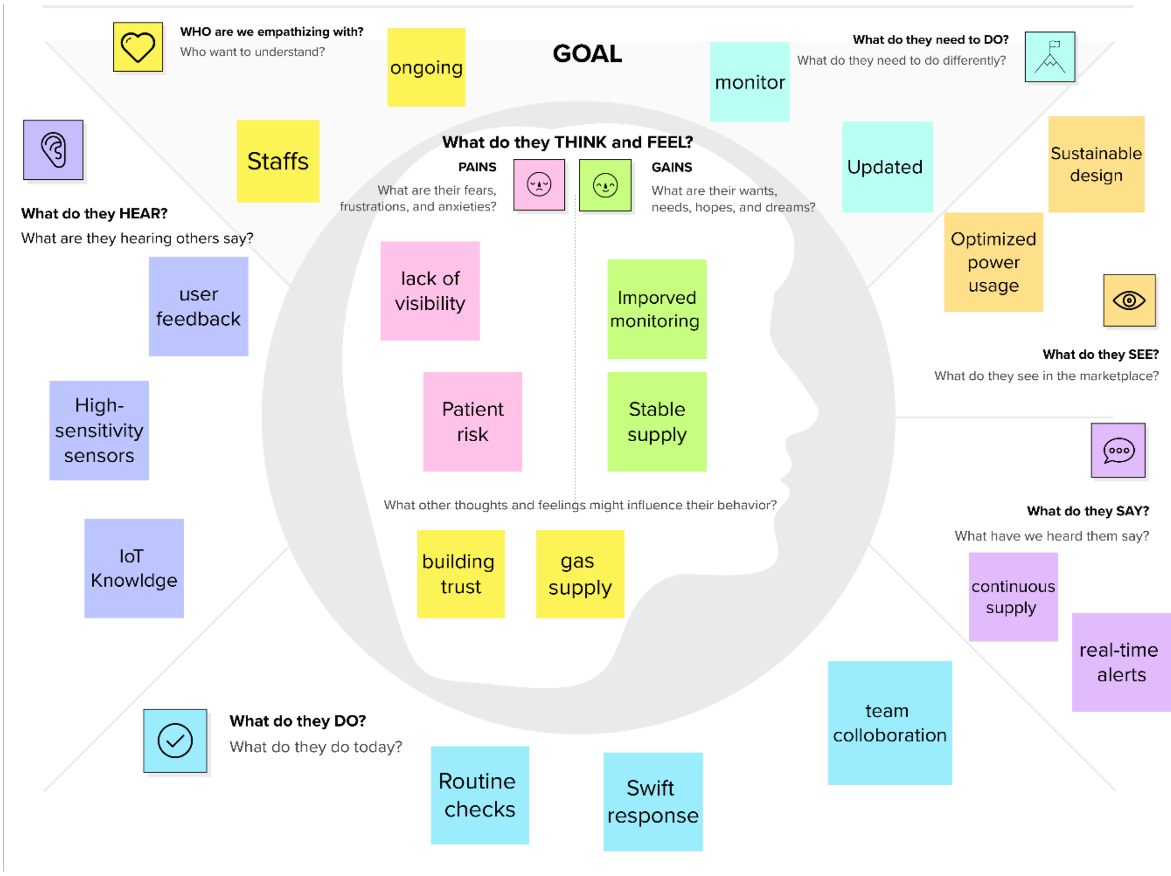
The purpose of developing a smart gas management system for operating theatres is to enhance patient safety by ensuring the availability of essential gases, particularly oxygen, while mitigating the risks associated with gas leaks. The system aims to provide real-time monitoring of gas levels, timely detection of leaks, and prompt alerts to healthcare staff. By enabling staff to take immediate safety measures and collaborate with maintenance teams, the system aims to prevent disruptions in gas supply, minimize potential hazards, and improve overall patient care. Additionally, the system promotes eco-friendly design, durability, and optimized power consumption to support sustainable healthcare practices. Ultimately, the purpose of this system is to establish a reliable and efficient gas management solution that safeguards patient well-being and ensures the smooth functioning of operating theatres.

2. IDEATION & PROPOSED SOLUTION

2.1 PROBLEM STATEMENT DEFINITION

The problem at hand is the need to ensure proper gas management in operating theatres to enhance patient safety and well-being. Specifically, the challenge lies in maintaining adequate levels of gases, especially oxygen, while promptly detecting and addressing gas leaks that can potentially jeopardize lives within the hospital setting. The existing manual monitoring processes may be inefficient and prone to human error, posing a risk of inadequate gas supply or delayed response to gas leaks. Therefore, there is a demand for a smart system that enables real-time monitoring of gas levels, alerts healthcare staff during gas leaks, and utilizes cloud-based storage and a mobile application for easy access to gas level data. Addressing this problem is essential to safeguard patients' health, optimize gas utilization, and establish a robust gas management solution in operating theatres.

2.2 Empathy map Canvas



2.3 IDEATION & BRAINSTORMING

IBM PROJECT

Brainstorm & idea prioritization

Gas Pipeline Monitoring System for Hospitals

•This Project helps the hospitals in monitoring the emission of gases.

•In some area, the gas sensors will be integrated to monitor the gas leakage.

•If in any area gas leakage is detected the staffs will be notified along with location.

•In web application, staffs can view the sensors parameters.

Share template feedback

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

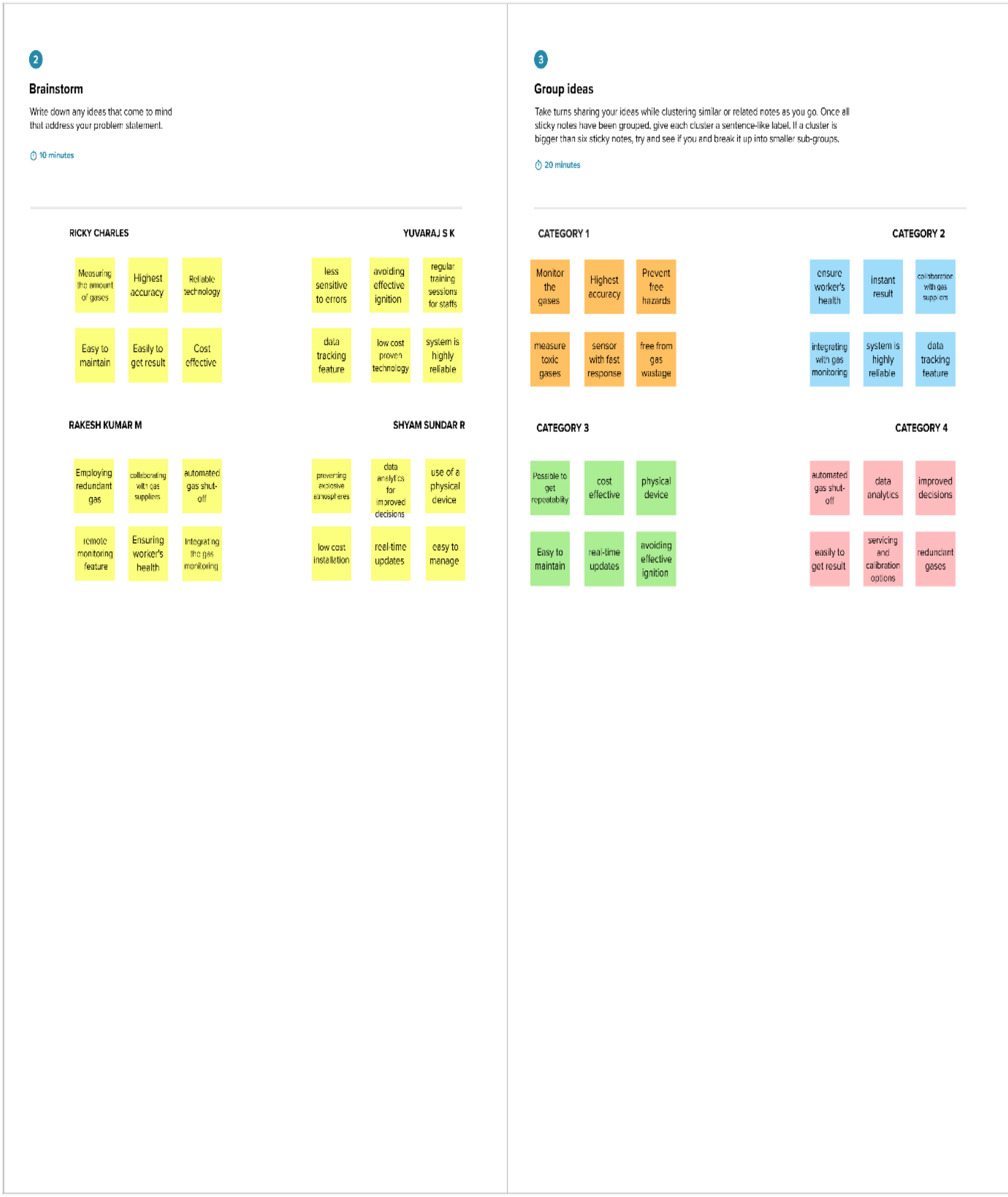
PROBLEM STATEMENT

It is essential to maintain proper levels of gas, as it is used in operating theatres in order to improve patient's condition. Oxygen is the most common and important gas that has to be checked at regular intervals for availability. Leakage of these gases can pose a severe threat to the safety of lives in hospitals. Through this smart system, staff will now be able to check the level of gas present for use and will be able to take safety measures when alerted during a gas leak. Sensors are used to record the levels of gases, which are stored in the cloud. These values are displayed on a mobile application for the users.

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

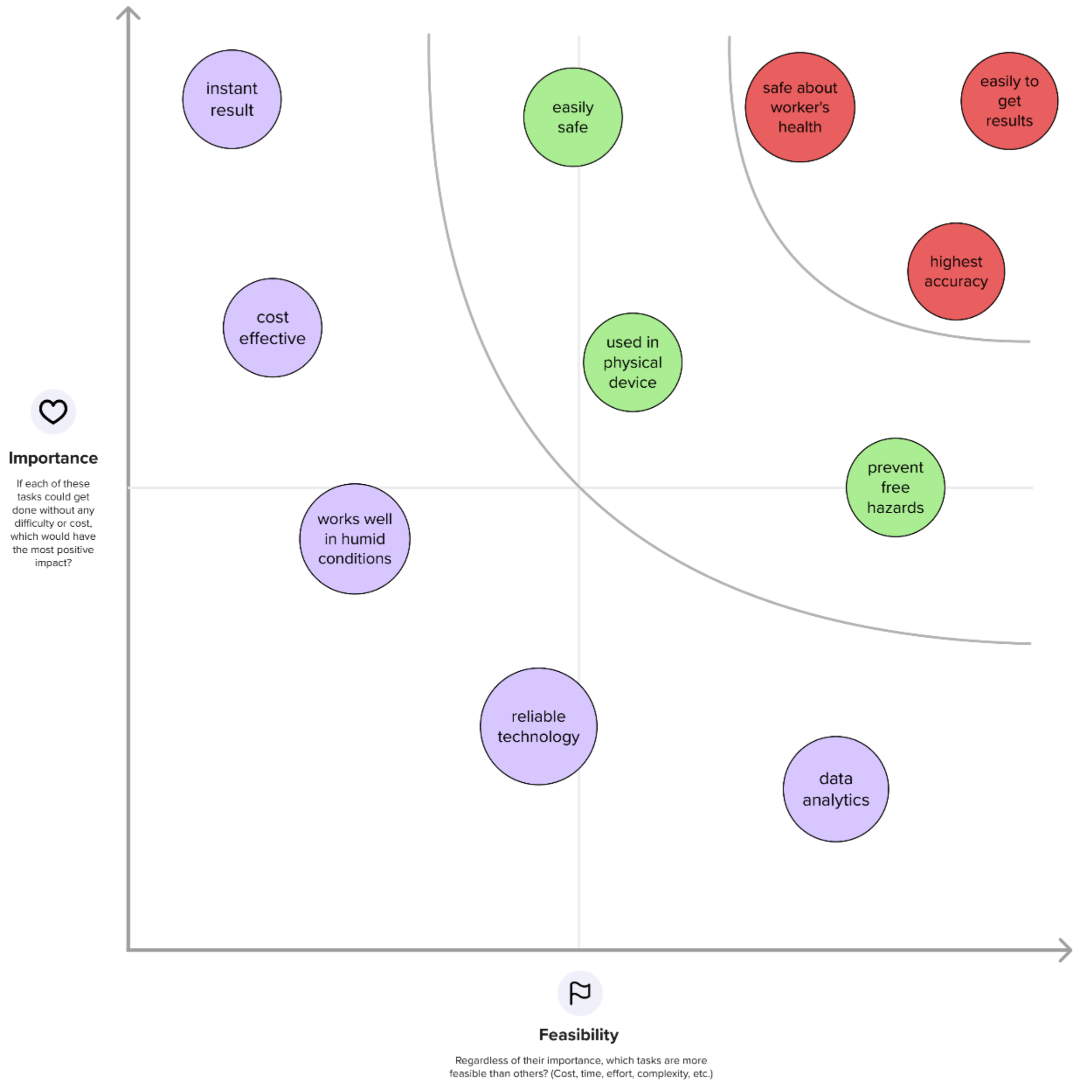

2
Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes
RICKY CHARLES
Measuring the amount of gases
Highest accuracy
Reliable technology
Easy to maintain
Easily to get result
Cost effective
YUVARAJ S K
less sensitive to errors
avoiding effective ignition
regular training sessions for staffs
data tracking feature
low cost proven technology
system is highly reliable
RAKESH KUMAR M
Employing redundant gas
collaborating with gas suppliers
automated gas shut-off
remote monitoring feature
Ensuring worker's health
Integrating the gas monitoring
SHYAM SUNDAR R
preventing explosive atmospheres
data analytics for improved decisions
use of a physical device
low cost installation
real-time updates
easy to manage
3
Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.
20 minutes
CATEGORY 1
Monitor the gases
Highest accuracy
Prevent free hazards
measure toxic gases
sensor with fast response
free from gas wastage
CATEGORY 2
ensure worker's health
instant result
collaboration with gas suppliers
integrating with gas monitoring
system is highly reliable
data tracking feature
CATEGORY 3
Possible to get repeatablity
cost effective
physical device
Easy to maintain
real-time updates
avoiding effective ignition
CATEGORY 4
automated gas shut-off
data analytics
improved decisions
easily to get result
servicing and calibration options
redundant gases

4

Prioritize

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

20 minutes



2.4 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	It is essential to maintain proper levels of gas, as it is used in operating theatres in order to improve patient's condition. Oxygen is the most common and important gas that has to be checked at regular intervals for availability. Leakage of these gases can pose a severe threat to the safety of lives in hospitals. Through this smart system, staff will now be able to check the level of gas present for use and will be able to take safety measures when alerted during a gas leak. Sensors are used to record the levels of gases, which are stored in the cloud. These values can also be displayed on a mobile application for the users.
2.	Idea / Solution description	• When the gas leakage is detected it will alert the user • It can send the notification to the user • Detection of the gas leakage is important and halting leakage is important equally.
3.	Novelty / Uniqueness	• Instant detection of gas leakage • Send notification to the concerned user • Easy to access and operate
4.	Social Impact / Customer Satisfaction	• Cost efficient • Easy to access and operate • Easy installation and detect the gas leakage faster • Prevent fires and explosions
5.	Business Model (Revenue Model)	•The product is advertised all over the platforms. Since it is economical, even helps small scale industries from disasters. •As the product usage can be understood by everyone, it is easy for them to use it properly for their safest organization
6.	Scalability of the Solution	•Since the product is cost efficient, it can be placed in many places in the hospitals. •Even when the gas leakage is more, the product sense the accurate values and alerts the workers effectively

3. REQUIREMENT ANALYSIS

3.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	➢ Registration through Form ➢ Offline Registration
FR-2	User Confirmation	➢ Confirmation via Email ➢ Confirmation via OTP
FR-3	Detection	The said system can be deployed in homes, hotels, factory units, LPG cylinder storage areas, and so on. The main advantage of this Arduino based application is that it can determine the leakage and send the data over to a site.
FR-4	Monitoring	The leakage can be monitored and can be optimized for detecting leakage of gasses.
FR-5	User Alert	The leakage can be monitored and can be optimized for detecting toxic gasses.
FR-6	Communication	The registered user is able to get alert from the system through a SMS and can also be able to get notification in app.

3.2 Non- Functional requirement

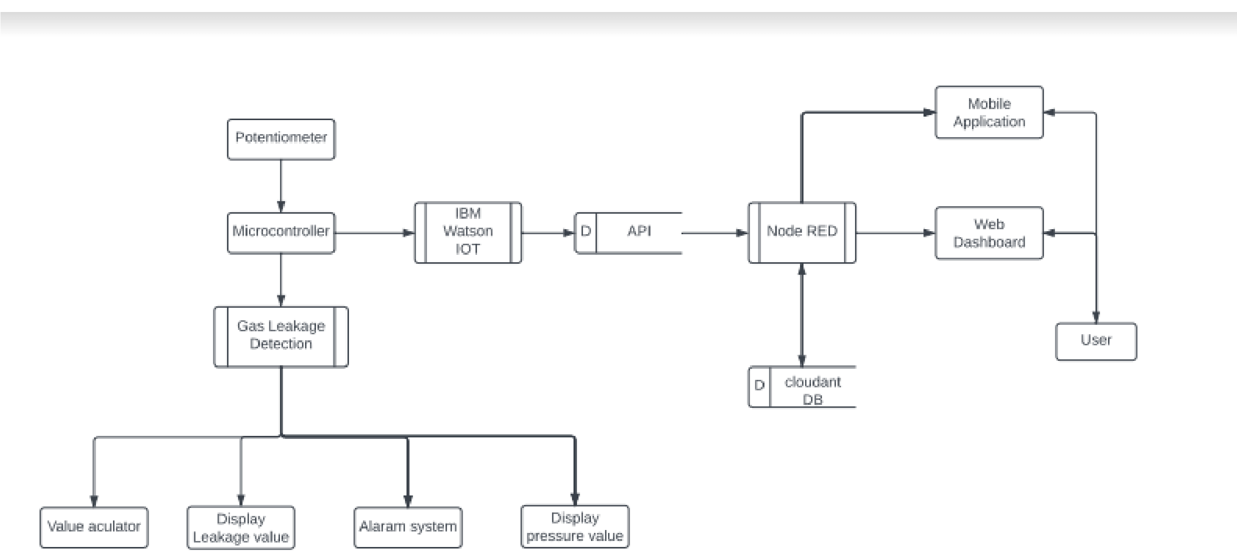
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	➢ Easier Installation process, and Realtime Monitoring Service.
NFR-2	**Security**	➢By identifying the danger of hazardous gas leakage with prior notification people can evacuate in time. ➢Data encryption & Cloud security.
NFR-3	**Reliability**	➢ Only authorised personnel have access to the system. ➢ Assured Data Security and Information conciseness. ➢ Longer Lifetime of Product/Service.

NFR-4	**Performance**	In this technique the gas sensor sends the signal to the Arduino UNO after detecting the gas leakage. Arduino to other externally connected devices such as buzzer and GSM send vigorous signals. SMS is sent by GSM module to the provided mobile number as a result. In practice, results are noticed by the people surrounding by the area are alerted by buzzer sound indicate the danger to the people by making beep sound.
NFR-5	**Availability**	➢ The user can access the System 24/7. ➢ Realtime monitoring system.
NFR-6	**Scalability**	By using this system that detects the gas leakage applicable usefully in the industrial and domestic purpose. In danger situations we are able save lives by using this system.

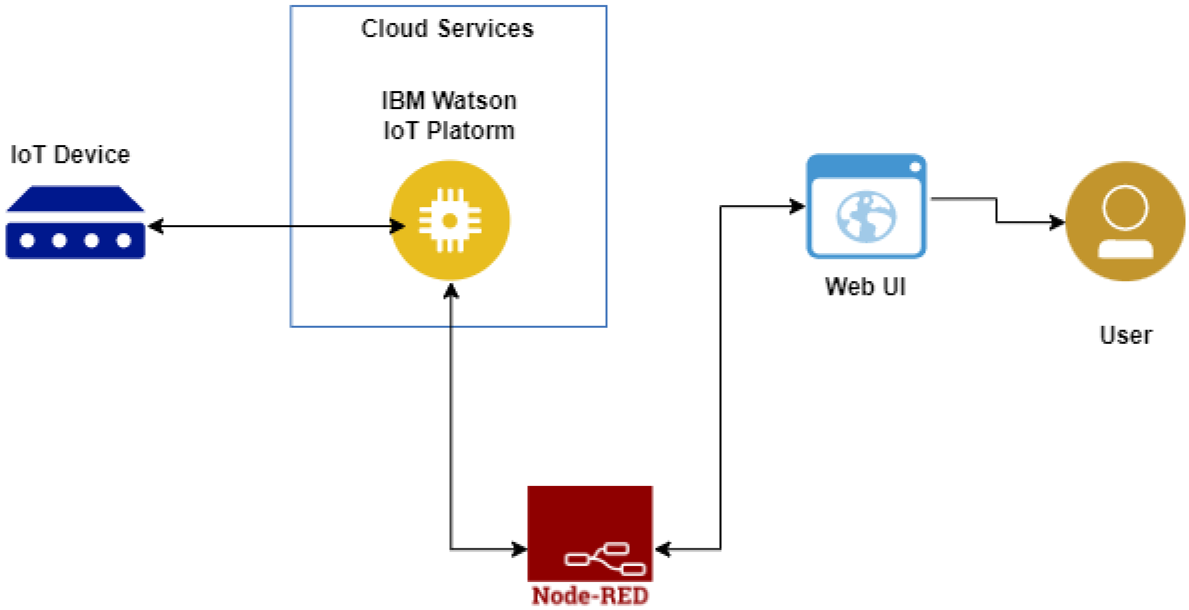
4. PROJECT DESIGN

4.1 Data Flow Diagrams

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.



4.2 Solution & Technical Architecture



4.2 User Stories

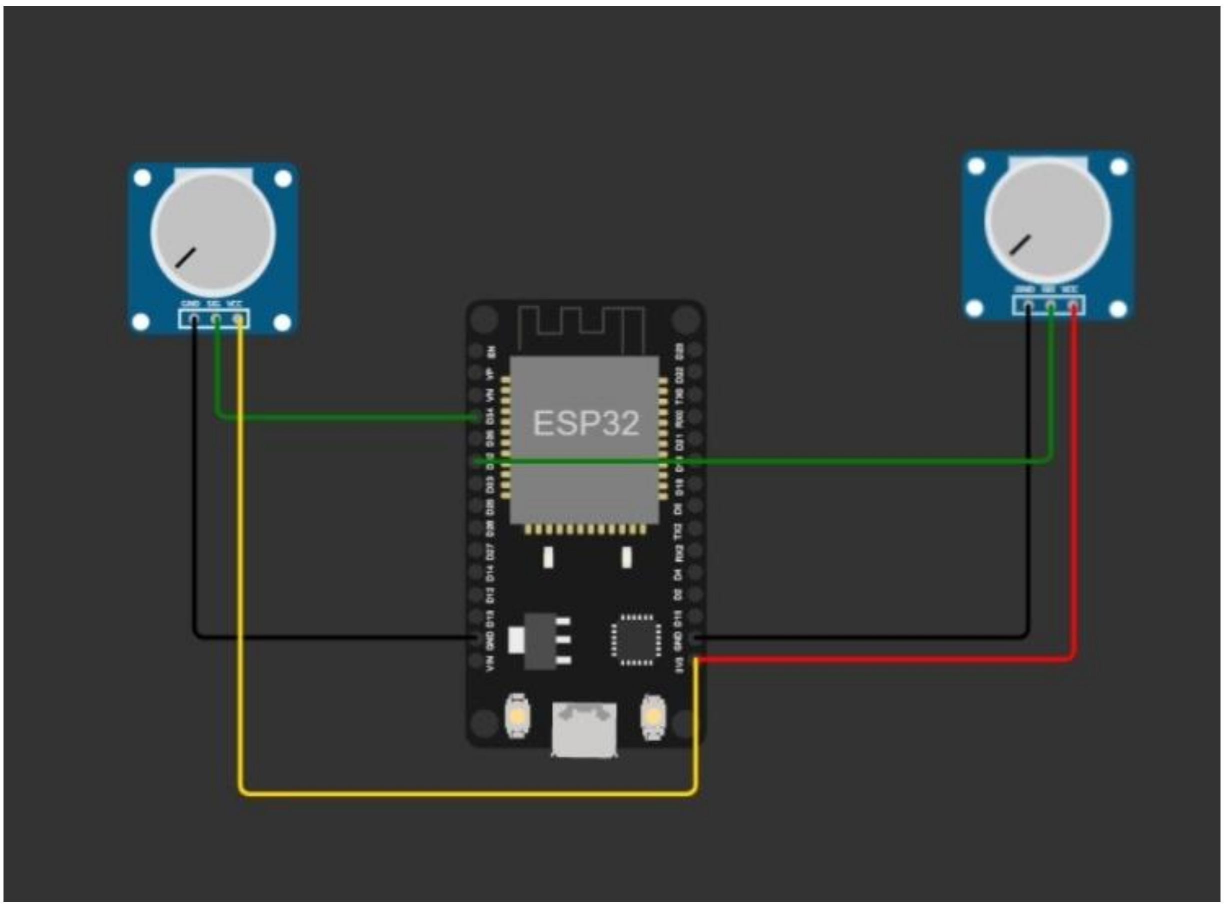
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
Customer (Hospital staff)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Shyam sundar
Customer (Hospital staff)	Confirmation	USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Rakesh kumar
Customer (Hospital staff)	Authorize	USN-3	As a user, I will enable the supervisor to monitor the gas leakage system status.	I can provide access to supervisor.	High	Yuvaraj
Customer (Supervisor)	Login	USN-4	As a user, I can log into the application by entering email & password.	I can get access to dashboard	High	Ricky charles
Customer (Supervisor)	Monitor	USN-5	As a user, I can monitor the status of the gas leakage system.	I can view the status of gas leakage system	High	Rakesh kumar
Customer (Line Workers)	Notification	USN-6	As a user, I can get (alarm system) alert about gas leakage	I can get alert about gas leak alert	High	Yuvaraj
Customer (Supervisor)	Notification	USN-7	As a user, I can get notification & alarming alert about gas leakage.	I can get notification about gas leak	Medium	Shyam sundar
Customer (Hospital staff)	Notification	UNS-8	As a user, I can get notification about gas leakage.	I can get notification alert about gas leakage	Medium	Ricky charles
Customer (Hospital staff)	Sign-up	USN-9	As a user, I can sign up using login	I can sign-up in application	Low	Shyam sundar
Customer (Supervisor)	Sign-up	UNS-10	As a user, I can sign up using login	I can sign-up in application	Low	Yuvaraj
Administrator	Service request	USN-11	As a user, I can request for service in case of any issue with gas leakage monitoring system	I can get service from provider	Low	Rakesh kumar
Administrator	Increased service	USN-12	As a user, I can request for scaling up the gas leakage monitoring system	I can get service from provider	Low	Shyam sundar

5. CODING & SOLUTIONING

5.1 Feature 1

Displaying current level of gas present

The system will provide a feature that displays the current level of gas present in the operating theatre. This feature will offer real-time visibility into the gas levels, allowing healthcare staff to easily monitor and assess the availability of gases, particularly oxygen. The gas level display will be clear and intuitive, providing an instant overview of the current status. This feature ensures that staff can quickly ascertain whether gas levels are within the desired range, enabling proactive management and timely actions to maintain an adequate gas supply for patient care.



CODE:

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

```
float pressure;
float leakage;


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

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

#define ORG "0129bl"//IBM ORGANITION ID
#define DEVICE_TYPE "Gas"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12345" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
//float h, t;


//-------- 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/test/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);


  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
```

```
{

  pressure=analogRead(34);
  leakage=analogRead(32);

  Serial.print("Pressure: ");
  Serial.println(pressure);
  Serial.print("Leakage: ");
  Serial.println(leakage);
  delay(1000);

  PublishData(pressure,leakage);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}



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

void PublishData(float pressure,float leakage) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */

  String payload = "{\"pressure\":";
  payload += pressure;
  payload += "," "\"leakage\":";
  payload += leakage;
  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");
  }
```

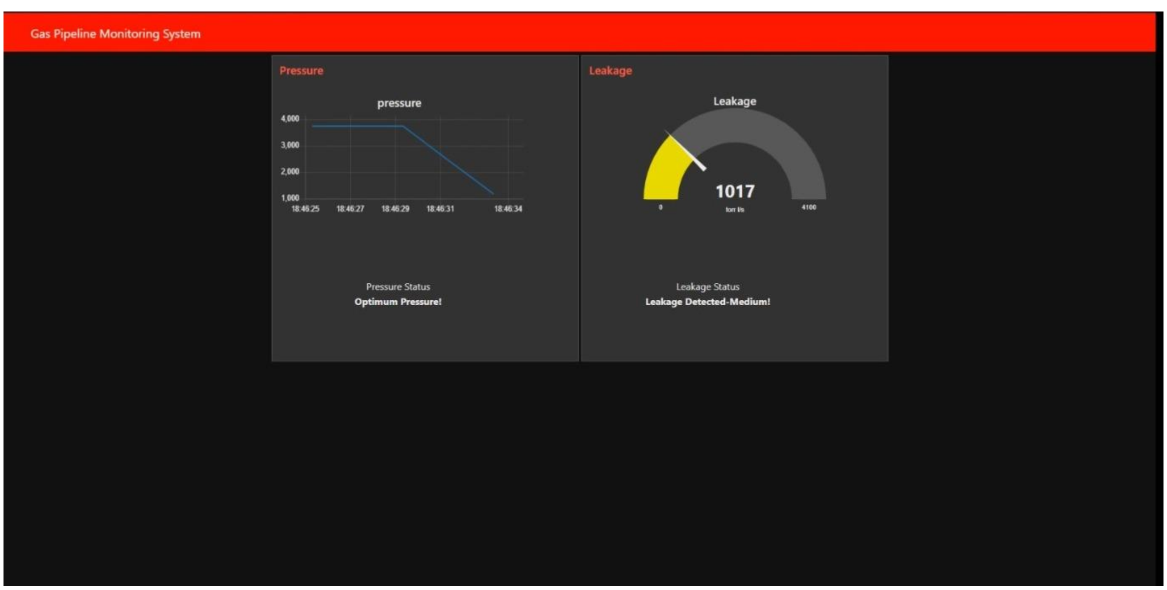
```
}
void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }

     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);

  for (int i = 0; i < payloadLength; i++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }

  Serial.println("data: "+ data3);

data3="";


}
```

5.2 Feature 2

Displaying alert message when there is a leakage

The system will include a feature that displays an alert message when a gas leakage is detected. Upon detecting a gas leak, the system will generate an immediate notification or alarm, signaling the presence of a potentially hazardous situation. The alert message will be prominently displayed on the user interface, drawing the attention of healthcare staff to take immediate safety measures. This feature ensures that gas leaks are promptly identified, allowing for swift response and minimizing the risk of harm to patients, staff, and the overall hospital environment.



CODE:

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

float pressure;
float leakage;


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

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

#define ORG "0129bl"//IBM ORGANITION ID
#define DEVICE_TYPE "Gas"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12345" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
//float h, t;


//-------- 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/test/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);


  delay(10);
```

```
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{

  pressure=analogRead(34);
  leakage=analogRead(32);

  Serial.print("Pressure: ");
  Serial.println(pressure);
  Serial.print("Leakage: ");
  Serial.println(leakage);
  delay(1000);

  PublishData(pressure,leakage);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}



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

void PublishData(float pressure,float leakage) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */

  String payload = "{\"pressure\":";
  payload += pressure;
  payload += "," "\"leakage\":";
  payload += leakage;
  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);

  for (int i = 0; i < payloadLength; i++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }

  Serial.println("data: "+ data3);

data3="";


}
```

6. RESULTS

6.1 Performance Metrics

NFT - Risk Assessment

S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volume Changes	Risk Score	Justification
1	Monitoring System	New	Moderate	No Changes	No Changes	10.5	>70 to 100%	ORANGE	As we have seen the chnages

NFT - Detailed Test Plan

S.No	Project Overview	NFT Test approach	Assumptions/Dependencies/Risks	Approvals/SignOff
1	Indication of Gas leakage	Done	In case of absence it may lead to improper detection/ it depends on the application owner/ it may lead to risk lives of many peoples.	Approved

End Of Test Report

S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	Identified Defects (Detected/Closed/Open)	Approvals/SignOff
1	Indication of Gas leakage	Done	Done	Detects the level of gas present and alerts the user if there is any leakage	GO	It is recommended to send the alert meassage directly to the users mobile	Closed	Approved

7. ADVANTAGES & DISADVANTAGES

Advantages:

1. Enhanced Patient Safety: The system significantly improves patient safety by ensuring the availability of gases, monitoring gas levels in real-time, and promptly alerting staff in case of gas leaks. This proactive approach reduces the risk of accidents, injuries, and adverse health effects.

2. Timely Response to Gas Leaks: The system enables quick detection of gas leaks and provides immediate alerts, allowing staff to take prompt safety measures. This helps prevent the escalation of gas leaks, minimizing potential hazards and their impact on patients and staff.

3. Improved Efficiency and Resource Management: By monitoring gas levels and usage patterns, the system helps optimize resource allocation, enabling efficient use of gases and reducing waste. This contributes to cost savings and a more sustainable approach to gas management.

4. Remote Access and Monitoring: The mobile application allows staff to remotely access gas level information, receive alerts, and take necessary actions. This feature facilitates convenient monitoring, particularly in large healthcare facilities or during off-hours, enhancing operational efficiency.

Disadvantages:

1. Initial Implementation Cost: Implementing a comprehensive gas monitoring system involves upfront costs for equipment, sensors, software development, and installation. The initial investment can be a potential drawback, particularly for healthcare facilities with limited budgets.

2. Maintenance and Upkeep: The system requires regular maintenance, including sensor calibration, software updates, and troubleshooting. Adequate resources and personnel

must be allocated to ensure the system operates optimally, adding to ongoing operational costs.

3. Technical Challenges: Technical issues, such as sensor malfunctions, connectivity disruptions, or false alarms, may arise. These challenges can impact the reliability and trustworthiness of the system, requiring prompt attention and technical expertise to address.

4. Staff Training and Adaptation: Introducing a new system requires staff training and adaptation to the technology. The learning curve and resistance to change may temporarily affect workflow efficiency until users become fully accustomed to the system.

5. Dependence on Power and Connectivity: The system relies on a stable power supply and internet connectivity to function effectively. Power outages or connectivity issues can temporarily disrupt the system's monitoring capabilities, necessitating backup solutions or manual checks during such instances.

8. CONCLUSION

In conclusion, the implementation of a smart gas monitoring system in operating theatres is a vital step towards ensuring patient safety and optimizing gas management in healthcare facilities. This system provides real-time display of gas levels, enabling healthcare staff to monitor the availability of critical gases like oxygen and promptly address any shortages or potential risks. The system's ability to generate alerts during gas leaks allows for immediate response and minimizes the danger posed to patients and staff.

Additionally, the smart gas monitoring system promotes efficient resource management by optimizing gas usage and reducing wastage. The convenience of remote access and monitoring through a mobile application empowers staff to monitor gas levels from anywhere within the hospital, enabling quick decision-making and proactive measures. While careful consideration should be given to initial costs, maintenance requirements, and staff training, the benefits of enhanced patient safety, minimized disruptions, and optimized resource utilization make the implementation of a smart gas monitoring system a valuable investment for healthcare facilities.

9. FUTURE SCOPE

The implementation of a smart gas monitoring system in operating theatres opens up exciting possibilities for future enhancements. Some potential areas for future development include integrating the system with hospital management systems for seamless data sharing and improved coordination. Advanced analytics and predictive maintenance capabilities can be incorporated to analyze usage patterns, predict gas shortages, and optimize resource allocation. Integration with other IoT devices and systems within the hospital can create an interconnected infrastructure for automated responses to gas leaks. Enhancements to the user interface and reporting features can provide comprehensive insights and customized dashboards for better decision-making. Compliance with regulatory standards and integration with energy management systems are also important considerations for future scope.

10. APPENDIX

Source Code

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

float pressure;
float leakage;
```

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

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

#define ORG "0129bl"//IBM ORGANITION ID
#define DEVICE_TYPE "Gas"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12345" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
//float h, t;


//-------- 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/test/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);



  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{

  pressure=analogRead(34);
  leakage=analogRead(32);

  Serial.print("Pressure: ");
  Serial.println(pressure);
  Serial.print("Leakage: ");
  Serial.println(leakage);
  delay(1000);

  PublishData(pressure,leakage);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}
```

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

```
void PublishData(float pressure,float leakage) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */

  String payload = "{\"pressure\":";
  payload += pressure;
  payload += "," "\"leakage\":";
  payload += leakage;
  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);

  for (int i = 0; i < payloadLength; i++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }

  Serial.println("data: "+ data3);

data3="";

}
```

Github & Project Video Demo Link

Github link:

https://github.com/naanmudhalvan-SI/PBL-NT-GP-17216-1683004727

Project Video Demo Link:

https://drive.google.com/file/d/1GHoQKAJFeoBoSiaW87use4hULmCSU4cf/view?usp=share_link