Gas Pipeline Leakage Detection at Hospital

Project Team Members :
Jyoti Prakash Behura (20BCE7355)
Abhijit Bose Das (20BCE7142)
Binit Nayak (20BCE7420)
Siddhant Samanta Singhar (20BCE7212)

1 INTRODUCTION

1.1 Overview

Gas pipeline leakage detection at hospitals is crucial for ensuring the safety of patients, staff, and visitors. Advanced sensor systems are employed to continuously monitor gas pipelines, detecting any leaks or abnormalities in real time. These systems utilize sophisticated technologies such as acoustic sensors, infrared cameras, and pressure sensors to identify potential leaks promptly. Immediate alerts are generated to alert the appropriate personnel, enabling them to take swift action and prevent any hazardous situations. Timely detection of gas pipeline leaks in hospitals ensures the maintenance of a secure environment, minimizing the risk of accidents and ensuring the well-being of everyone within the facility.

1.2 Purpose

The purpose of gas pipeline leakage detection at hospitals is to ensure the safety and well-being of patients, staff, and visitors. The detection systems are designed to promptly identify any gas leaks within the hospital premises, minimizing the risk of fire, explosions, or other hazardous incidents. By continuously monitoring the gas pipelines using advanced sensors and technologies, potential leaks can be detected in real-time. This enables immediate action to be taken, such as shutting off the gas supply, evacuating affected areas, and addressing the issue to prevent any harm. The primary purpose is to maintain a secure environment within the hospital, protecting the lives and health of all individuals present.

2 LITERATURE SURVEY

2.1 Existing problem

Gas pipeline leakage detection at hospitals faces a significant challenge due to the potential risks associated with undetected leaks. The existing problem lies in the possibility of gas leaks leading to fire hazards, explosions, and health risks for patients, staff, and visitors. The conventional detection methods may not be robust enough to provide timely alerts, leaving the facility vulnerable to accidents. There is a need for more advanced and reliable detection systems that can accurately identify leaks in real-time, ensuring the safety of everyone within the hospital premises. Addressing this problem is crucial for maintaining a

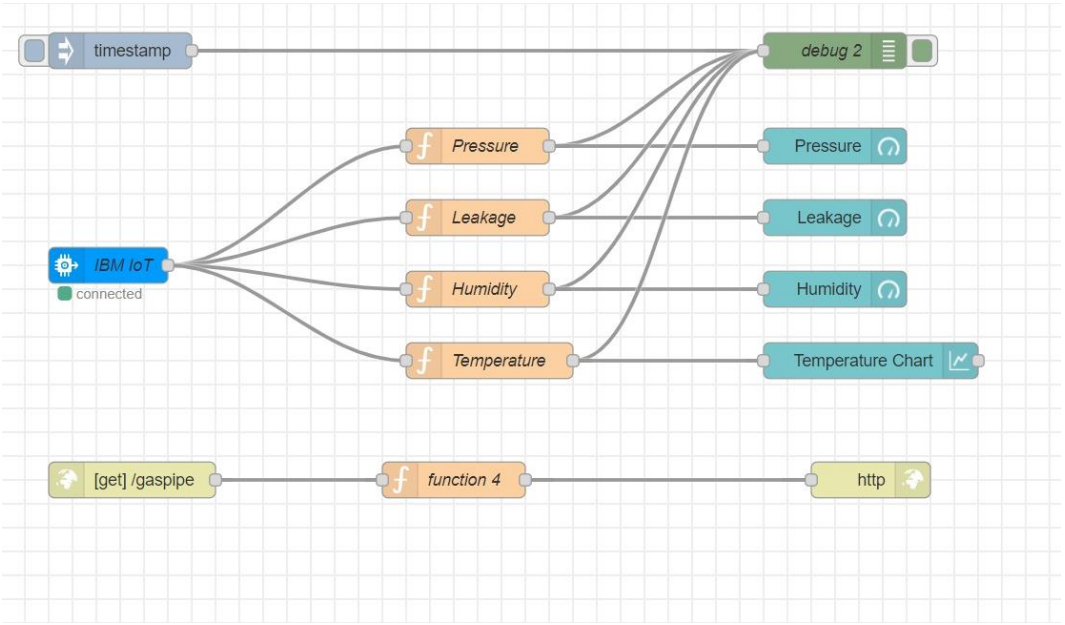
secure environment and preventing any potential harm.

2.2 Proposed solution

A proposed solution for gas pipeline leakage detection at hospitals is the implementation of an integrated and intelligent monitoring system. This system would utilize advanced sensor technologies, such as acoustic, infrared, and pressure sensors, to continuously monitor the gas pipelines in real-time. The collected data would be processed through machine learning algorithms and anomaly detection techniques to identify potential leaks promptly. Immediate alerts would be sent to designated personnel, enabling quick response and appropriate actions, such as shutting off the gas supply and evacuating affected areas. This solution ensures early detection and mitigation of gas leaks, minimizing the risks and ensuring the safety of patients, staff, and visitors within the hospital.

3 THEORETICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software Designing

Hardware to be used to make:- ESP32, Gas Sensors, Relay Modules, Power Supply, Wires

Wokwi Online Micro-Controller Simulator, IBM Cloud, MIT APP Inventory Component used:- ESP32, DHT22, Potentiometer

4 EXPERIMENTAL INVESTIGATIONS

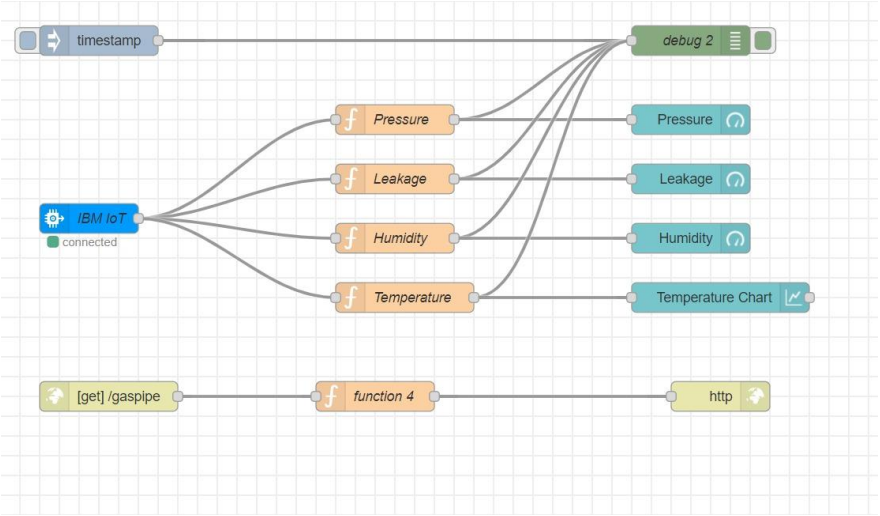
1. Sensor Selection: Choose appropriate gas sensors based on the specific gases to be monitored, such as methane or carbon

monoxide. Consider factors like sensitivity, accuracy, and response time.

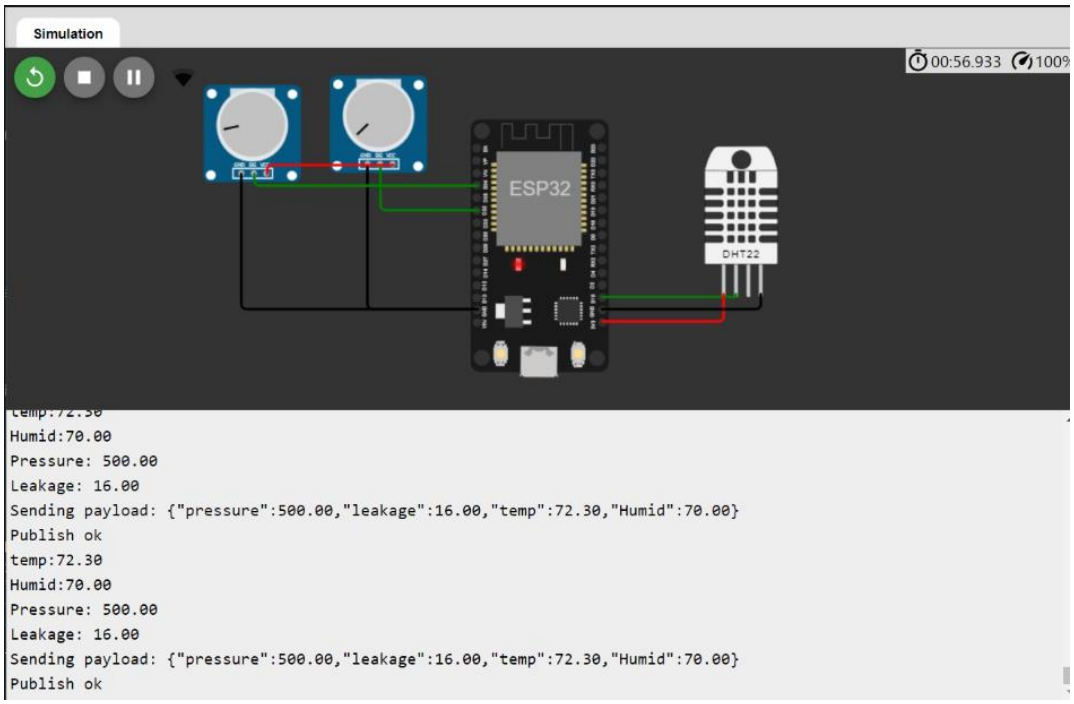
2. Hardware Setup: Connect the gas sensors to an IoT-enabled device like an ESP32 microcontroller. Ensure proper power supply and wiring connections.
3. Data Acquisition: Develop a program to collect data from the gas sensors. This may involve periodic sampling or real-time streaming of sensor readings.
4. Data Transmission: Establish a communication protocol, such as Wi-Fi or Bluetooth, to transmit the acquired data from the microcontroller to a central server or cloud platform for further processing
5. Cloud Integration: Set up a cloud-based platform to receive and store the sensor data. Services like AWS IoT, Azure IoT, or Google Cloud IoT can be utilized for this purpose. Here we have made use of IBM Cloud.
6. Data Processing: Implement data processing algorithms on the cloud platform to analyze the received sensor data. This may involve threshold-based detection, anomaly detection, or machine learning techniques.
7. Alert Generation: Define criteria for gas leak detection and generate alerts or notifications when a gas leak is detected. This can be accomplished via email, SMS, or push notifications to relevant personnel.
8. Visualization and Reporting: Develop a user interface or dashboard to visualize real-time sensor data, display gas leak events, and generate reports for further analysis.
9. Testing and Validation: Conduct controlled experiments to simulate gas leaks and evaluate the performance of the system. Verify the accuracy, sensitivity, and response time of the gas leakage detection.
10. Iterative Improvements: Based on the experimental results and feedback, refine the system, make necessary adjustments to improve accuracy, optimize power consumption, and enhance overall performance.

5 FLOWCHART

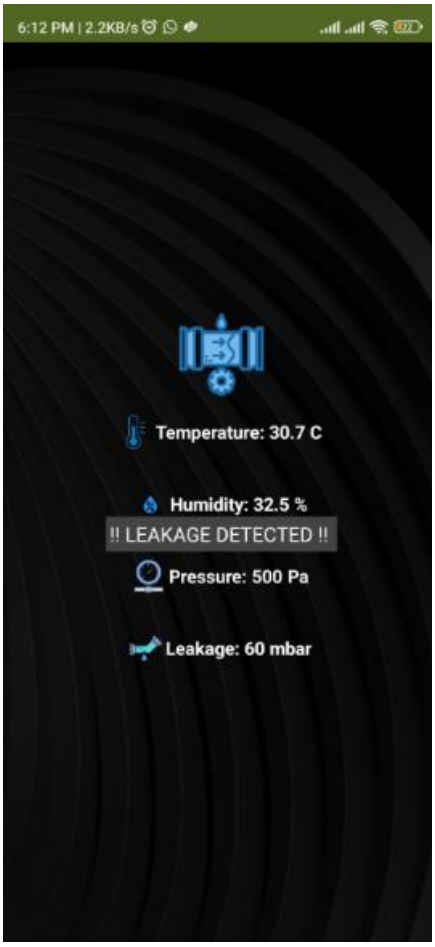
Gas Pipeline Leakage Detection at Hospital



Circuit:

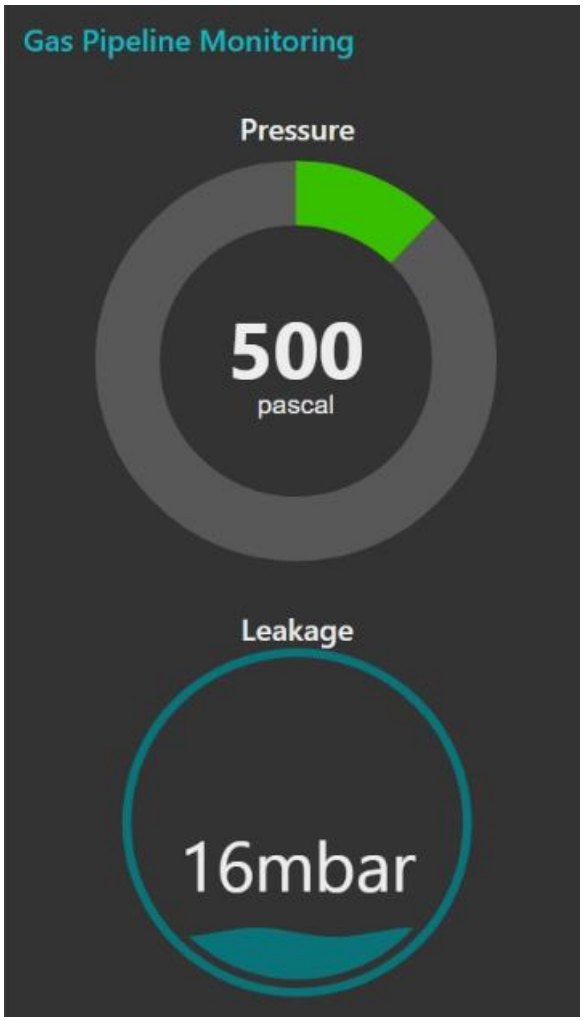

Simulation
00:56.933
100%
ESP32
DHT22
Humid:70.00
Pressure: 500.00
Leakage: 16.00
Sending payload: {"pressure":500.00,"leakage":16.00,"temp":72.30,"Humid":70.00}
Publish ok
temp:72.30
Humid:70.00
Pressure: 500.00
Leakage: 16.00
Sending payload: {"pressure":500.00,"leakage":16.00,"temp":72.30,"Humid":70.00}
Publish ok

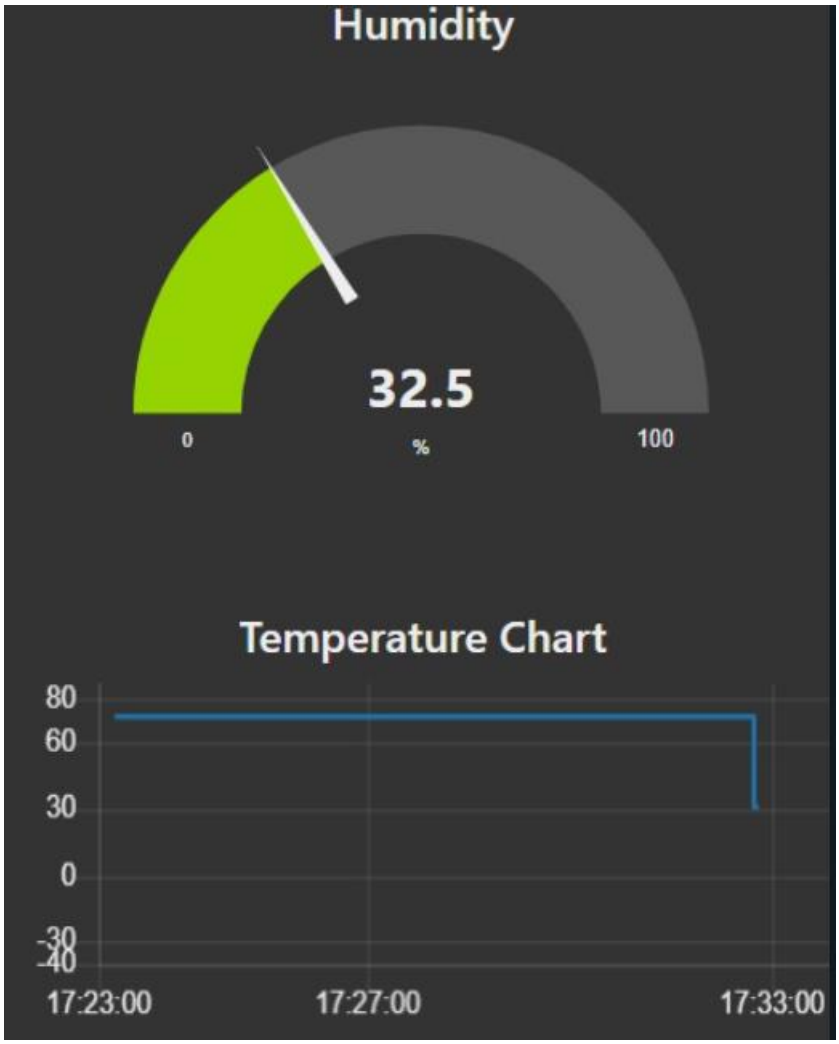
6 RESULT


6:12 PM | 2.2KB/s
Temperature: 30.7 C
Humidity: 32.5 %
!! LEAKAGE DETECTED !!
Pressure: 500 Pa
Leakage: 60 mbar

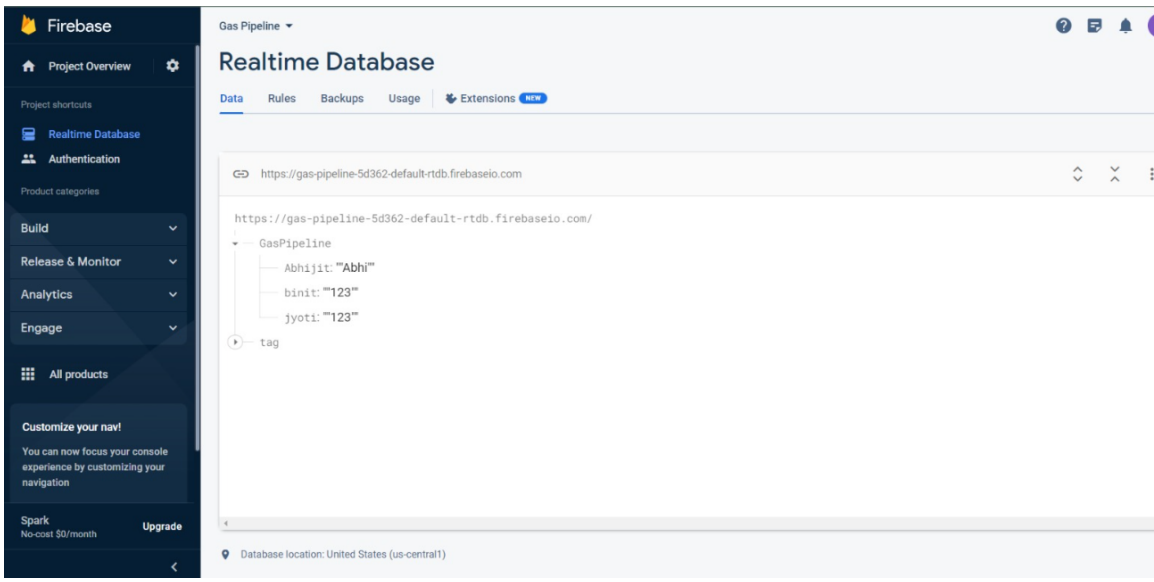
```
Pressure: 4095.00
Leakage: 92.00
Sending payload:
{"pressure":4095.00,"leakage":92.00,"temp":80.00,"Humid":100.00}
Publish ok
```

Node Red Dashboard:





Firebase Database:



Advantage: -

1. Reduction in the cost of transportation is very significant.
2. Supply through pipelines is very reliable.
3. In the case of underground pipelines, the land in which the pipeline is laid can still be used.
4. It ensures supply in remote areas where roadways are not very good, also it provides safe and secure supply for defense needs.
5. Gas on the pipeline is 30% cheaper than an LPG cylinder.

Disadvantage: -

1. Illegal pilferage and wastage due to leak is a problem in pipelines.
2. Like other big linear structures patrolling and maintenance of pipelines is a huge task.
3. In the case of chemicals and petroleum pipelines any leak can cause an accident.

8 APPLICATIONS

1. Residential and Commercial Buildings: Gas pipeline leakage detection systems are crucial in residential and commercial buildings to ensure the safety of occupants. Timely detection of gas leaks can prevent accidents, such as fire or explosions, and protect lives and property.
2. Industrial Facilities: Industries that utilize or store large quantities of gases, such as chemical plants, refineries, and manufacturing facilities, employ gas pipeline leakage detection systems to mitigate the risk of hazardous incidents. Early detection helps prevent accidents, environmental contamination, and potential health hazards to workers.
3. Power Plants: Gas pipeline leakage detection is essential in power plants that use natural gas or other gases as a fuel source. Detecting and addressing gas leaks promptly helps maintain safe operations, prevent equipment damage, and protect workers' well-being.
4. Transportation: Gas pipeline leakage detection is relevant in transportation sectors, particularly for vehicles using compressed natural gas (CNG) or liquefied petroleum gas (LPG). Ensuring the integrity of gas pipelines and detecting leaks in vehicles is vital for transportation safety.
5. Oil and Gas Industry: Gas pipeline leakage detection is critical in the oil and gas industry to monitor the integrity of pipelines and prevent leaks. It helps reduce the risk of environmental pollution,

equipment damage, and worker safety hazards.

6. Healthcare Facilities: Gas pipeline leakage detection is essential in hospitals, clinics, and laboratories to ensure the safety of patients, staff, and visitors. Prompt detection of gas leaks helps prevent accidents, maintain a secure environment, and protect individuals from potential health risks.
7. Environmental Monitoring: Gas pipeline leakage detection systems contribute to environmental monitoring efforts. Detecting leaks in natural gas pipelines or storage facilities helps prevent the release of greenhouse gases and minimizes environmental impact.

9 CONCLUSION

In conclusion, gas pipeline leakage detection at hospitals is of utmost importance to ensure the safety and well-being of patients, staff, and visitors. Timely detection of gas leaks can prevent accidents, fire hazards, and health risks within hospital premises. By implementing advanced sensor technologies, such as acoustic, infrared, and pressure sensors, integrated with IoT systems, hospitals can continuously monitor gas pipelines in real time. This enables the prompt identification of potential leaks, triggering immediate alerts and enabling swift actions to mitigate risks. Gas pipeline leakage detection systems enhance the security of hospitals, minimizing the potential harm to individuals and maintaining a safe environment for all. Continued research, development, and implementation of such systems are essential for enhancing hospital safety protocols and protecting lives.

10 FUTURE SCOPE

1. Enhanced Sensor Technologies: Continuous research and development of more advanced sensor technologies can lead to increased sensitivity, accuracy, and reliability in detecting gas leaks. Innovations in sensor design and material composition can improve detection capabilities and reduce false alarms.

2. Integration with Artificial Intelligence: The integration of artificial intelligence (AI) techniques, such as machine learning and pattern recognition, can enhance the gas pipeline leakage detection

system's intelligence. AI algorithms can learn from historical data, identify complex patterns, and improve the accuracy of leak detection, reducing false positives and optimizing the system's performance.

3. Wireless Sensor Networks: The use of wireless sensor networks can enable seamless connectivity between multiple sensors spread across a hospital facility. This network can facilitate real-time data transmission, centralized monitoring, and quicker response to gas leaks.

4. Predictive Maintenance: Implementing predictive maintenance techniques can help hospitals proactively monitor the health of gas pipelines, identify potential weaknesses or vulnerabilities, and perform necessary maintenance before leaks occur. This approach can reduce downtime, enhance system reliability, and prevent unforeseen incidents.

5. Data Analytics and Visualization: Developing advanced data analytics and visualization tools can provide actionable insights from the collected sensor data. Real-time dashboards, trend analysis, and predictive analytics can help hospitals identify potential areas of concern, optimize resource allocation, and make informed decisions regarding maintenance and safety measures.

11 BIBILOGRAPHY

- https://www.robotique.tech/robotics/intelligent-gas-leak-detection-system-with-esp32/
- https://iotstarters.com/esp32-based-gas-leakage-detection-using-email-notification/
- https://wokwi.com/projects/348681809497686611
- http://ijiird.com/wp-content/uploads/EnTC024.pdf
- https://research.ibm.com/publications/preventive-leak-detection-for-high-pressure-gas-transmission-networks
- https://www.mdpi.com/1424-8220/21/2/367
- https://upikptss.edu.my/ojs/index.php/IJRIM/article/download/144/49/
- https://www.researchgate.net/publication/328605096_Development_of_ESP32-based_Wi-Fi_Electronic_Nose_System_for_Monitoring_LPG_Leakage_at_Gas_Cylinder_Refurbish_Plant

APPENDIX

A. Source Code

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11

DHT dht (DHTPIN, DHTTYPE);

float pressure;
float leakage;


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

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

#define ORG "9f8wlx"//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
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/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);
  dht.begin();
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{

  pressure=analogRead(34);
  leakage=analogRead(32);
  h = dht.readHumidity();
  t = dht.readTemperature();

  Serial.print("temp:");
  Serial.println(t);
  Serial.print("Humid:");
  Serial.println(h);

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


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

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

void PublishData(float pressure,float leakage,float temp, float humid) {
  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 += "," "\"temp\":";
  payload += temp;
  payload += "," "\"Humid\":";
  payload += humid;
  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="";


}
```