EXPLOSION RISK DETECTION AND MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

AKSHAYA R (2116210701023)
BALAHARINATH C (2116210701037)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY, CHENNAI

MAY 2024

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Thesis titled **"EXPLOSION RISK DETECTION AND MANAGEMENT SYSTEM"** is the bonafide work of **"AKSHAYA R(2116210701023), BALAHARINATH C (2116210701037)"** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr . T.Kumaragurubaran M.Tech.,Ph.D.,AP(SG)

PROJECT COORDINATOR

Professor

Department of Computer Science and Engineering

Rajalakshmi Engineering College

Chennai - 602 105

Submitted to Project Viva-Voce Examination held on_____

Internal Examiner   **External Examiner**

ABSTRACT

The oil and gas industry is inherently prone to explosion risks, primarily due to the potential for leaks. In response to this critical safety concern, we present a comprehensive solution designed to detect and mitigate such risks effectively. Our approach integrates Internet of Things (IoT) technology, machine learning algorithms, and modern web development principles to create a robust and responsive system.

At the core of our solution are sensor-equipped Arduino devices strategically deployed in key areas within oil and gas facilities. These sensors, including UV and MQ2 sensors, continuously monitor for the presence of oil and gas leaks. Upon detecting levels exceeding predefined thresholds, alerts are triggered, initiating swift actions to mitigate potential hazards. The alerts are transmitted via an ESP8266 module, which communicates with a REST API for logging and immediate notification to relevant personnel.

To facilitate real-time monitoring and response coordination, we have developed a backend system using the MERN (MongoDB, Express.js, React.js, Node.js) stack. This backend manages the transmission of sensor data, employee information, and system status updates. Leveraging Socket.IO, the communication between the frontend and backend ensures seamless and responsive updates on threat zones .

ACKNOWLEDGMENT

AKSHAYA R

BALAHARINATH C

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The oil and gas industry plays a pivotal role in powering global economies, providing essential resources for various sectors. However, the inherent risks associated with oil and gas exploration, production, and transportation underscore the importance of robust safety measures and risk management practices. In recent years, the emergence of Internet of Things (IoT) technologies has revolutionized industrial operations, offering unprecedented capabilities for real-time monitoring, data analytics, and predictive maintenance.

This project focuses on leveraging IoT solutions to enhance safety protocols within the oil and gas industry, specifically addressing the critical need for explosion risk detection and mitigation. By integrating IoT sensors with advanced data processing and analysis systems, we aim to develop a comprehensive explosion risk detection system capable of early detection and proactive response to potential hazards.

The introduction of IoT sensors enables continuous monitoring of oil and gas levels in various industrial environments, providing real-time insights into environmental conditions. Coupled with sophisticated data processing algorithms and machine learning techniques, these sensors facilitate predictive analysis, allowing for the identification of abnormal trends and early warning of potential risks.

In this context, our project aims to contribute to the ongoing efforts to improve safety standards and mitigate risks within the oil and gas sector. By harnessing the power of IoT technologies, we seek to develop an innovative solution that enhances operational safety, minimizes the occurrence of incidents, and ultimately ensures the well-being of workers and the integrity of critical infrastructure and mitigate risks   of the   within the oil and gas sector.

1.1 PROBLEM STATEMENT

The project addresses the need for improved safety measures in the oil and gas industry by developing an IoT-based explosion risk detection system. Through real-time monitoring and early warning capabilities, the system aims to mitigate potential hazards and enhance the overall safety of industrial operations.

1.2 SCOPE OF THE WORK

The scope of work involves designing and implementing an IoT-based explosion risk detection system tailored for the oil and gas industry. This includes the development of sensor hardware, backend infrastructure, and a real-time monitoring dashboard. Additionally, machine learning techniques will be employed for predictive analysis. The project aims to enhance safety measures and mitigate potential hazards in industrial environments.

1.4 AIM AND OBJECTIVES OF THE PROJECT

The aim of the project is to develop an IoT-based explosion risk detection system for the oil and gas industry. This system will enable real-time monitoring of oil and gas levels, early detection of potential hazards, and prompt alert generation. Integrating machine learning techniques for predictive analysis, the project seeks to enhance safety measures and mitigate risks. By providing proactive risk mitigation capabilities, the system aims to minimize incidents, ensuring the safety of workers and infrastructure. Ultimately, the deployment of this system will contribute to enhancing operational efficiency and safety within oil and gas facilities. This system will enable real-time monitoring of oil and gas levels, early detection of potential hazards, and prompt alert generation. Integrating machine learning techniques for predictive analysis, the project seeks to enhance safety measures and mitigate risks.

1.5 RESOURCES

For the successful execution of this project, essential resources include a functional computer workstation with necessary software tools, unlimited internet access for extensive secondary research, and unrestricted access to the university lab for academic resources and technical support. Additionally, access to a Prolog development kit or similar software is crucial for programming the system. Utilizing the university library's physical and digital resources, coupled with guidance from faculty members or mentors, will further enhance research efforts. These resources collectively form the foundation for conducting thorough research, gathering relevant content, and programming the desired system to achieve project objectives effectively.

1.6 MOTIVATION

The motivation behind developing an IoT-based explosion risk detection system for the oil and gas industry stems from the critical need to enhance operational safety and mitigate potential hazards. By leveraging IoT technologies and predictive analytics, the system aims to provide real-time monitoring, early detection, and prompt alerting mechanisms, thereby safeguarding personnel and infrastructure from the devastating consequences of explosions and gas leaks. For the successful execution of this project, essential resources include a functional computer workstation with necessary software tools, unlimited internet access for extensive secondary research. The motivation behind developing an IoT-based explosion risk detection system for the oil.

The motivation behind developing an IoT-based explosion risk detection system for the oil and gas industry stems from the critical need to enhance operational safety and mitigate potential hazards. By leveraging IoT technologies and predictive analytics, the system aims to provide real-time monitoring, early detection, and prompt alerting mechanisms mitigate. potential

CHAPTER 2
LITERATURE SURVEY

2.1 LITERATURE SURVEY

The oil and gas industry, being a cornerstone of global energy supply, continuously seeks innovative technologies to enhance operational efficiency, safety, and sustainability. The advent of the Internet of Things (IoT) has significantly transformed various aspects of the industry, offering unprecedented capabilities for real-time monitoring, data analytics, and predictive maintenance. In this literature survey, we explore key research works and advancements in leveraging IoT technologies within the oil and gas sector.

1. **IoT Applications in Oil and Gas Exploration:**

Recent studies have highlighted the potential of IoT technologies to revolutionize oil and gas exploration processes. Ijiga et al. (2020) conducted a survey on emergent configurations in the industrial IoT for oil and gas explorations, emphasizing the role of IoT-enabled configurations in optimizing exploration activities. The authors discuss the integration of IoT devices with drilling equipment, seismic sensors, and geological mapping systems to improve efficiency and accuracy in exploration operations.

2. **Real-Time Monitoring and Asset Management:**

Real-time monitoring of oil and gas assets is crucial for ensuring operational safety and minimizing downtime. Aalsalem et al. (2017) proposed an intelligent well monitoring system based on IoT, utilizing smart objects for efficient

monitoring of well conditions. Similarly, Khan et al. (2017) developed a reliable IoT-based architecture for the oil and gas industry, focusing on asset tracking, equipment health monitoring, and predictive maintenance.

3. **Risk Detection and Mitigation:**

Explosion risks pose significant threats to personnel and infrastructure within oil and gas facilities. Wanasinghe et al. (2020) conducted a systematic review on IoT applications in the oil and gas industry, emphasizing the development of IoT-based anti-theft alarm systems and explosion risk detection solutions. The authors highlight the potential of IoT sensors for real-time monitoring of gas leaks and early detection of potential hazards, thereby enhancing safety measures.

4. **Predictive Analytics and Maintenance:**

Machine learning algorithms have gained prominence in predictive analytics and maintenance strategies within the oil and gas sector. Nguyen et al. (2020) conducted a systematic review of big data analytics for the oil and gas industry, emphasizing the application of machine learning techniques for predictive maintenance. The authors discuss the integration of IoT data with predictive models to forecast equipment failures and optimize maintenance schedules.

5. **Wireless Sensor Networks (WSNs) for Monitoring:**

Wireless Sensor Networks (WSNs) offer scalable and cost-effective solutions for monitoring oil and gas assets in remote and harsh environments. Aalsalem et al.

recent advances, taxonomy, requirements, and open challenges. The authors highlight the importance of scalability and security in deploying WSNs for monitoring critical infrastructure.

6. **Integration of Industry 4.0 Technologies:**

The emergence of Industry 4.0 technologies, including IoT, Big Data, and Artificial Intelligence, has reshaped the oil and gas industry's operational landscape. Lu et al. (2019) conducted a systematic review on the Oil and Gas 4.0 era, analyzing typical application scenarios and technological trends. The authors discuss the integration of IoT devices with existing infrastructure to optimize production processes and enhance operational efficiency.

In conclusion, the literature survey highlights the growing significance of IoT technologies in revolutionizing various facets of the oil and gas industry. From exploration and asset management to risk detection and predictive maintenance, IoT-enabled solutions offer immense potential for improving safety, efficiency, and sustainability across the sector. However, challenges such as data security, interoperability, and scalability remain to be addressed for widespread adoption and implementation of IoT technologies in the industry challenges such as data .

The emergence of Industry 4.0 technologies, including IoT, Big Data, and Artificial Intelligence, has reshaped the oil and gas industry's operational landscape. Lu et al. (2019) conducted a systematic review on the Oil and Gas 4.0 era, analyzing typical application scenarios and technological trends. The authors discuss the integration of IoT devices with existing infrastructure to optimize production processes and enhance operational efficiency the growing

2.2 PROPOSED SYSTEM

 The proposed system is an IoT-based explosion risk detection system designed specifically for the oil and gas industry to enhance safety measures and mitigate potential hazards. It integrates cutting-edge IoT technologies, real-time monitoring capabilities, and predictive analytics to provide early detection and prompt response to abnormal conditions indicative of potential explosion risks.

At the core of the proposed system are IoT sensors strategically placed throughout oil and gas facilities to continuously monitor environmental parameters such as gas levels, temperature, and pressure. These sensors collect data in real-time and transmit it to a central processing unit for analysis.

The data processing unit applies advanced machine learning algorithms to analyze sensor data, identify patterns associated with potential hazards, and generate real-time alerts. These alerts are promptly communicated to relevant personnel via email, SMS, or other communication channels, enabling quick intervention and mitigation measures to be implemented.

The system also includes a user-friendly graphical interface that provides system administrators and operators with intuitive tools for monitoring sensor data, configuring system settings, and accessing historical data for analysis and reporting purposes.

Furthermore, the proposed system facilitates seamless integration with existing infrastructure and third-party systems within oil and gas facilities, ensuring interoperability and data exchange. Maintenance scheduling tools and software update mechanisms are also incorporated to ensure the system's continued reliability and effectiveness.

Overall, the proposed system offers a comprehensive solution for enhancing safety measures and mitigating potential explosion risks within oil and gas facilities, thereby safeguarding personnel and infrastructure and reducing the likelihood of incidents.

2.3 ALGORITHM

The algorithm employed in the IoT-based explosion risk detection system encompasses several key steps to enable early detection of potential hazards and prompt alerting mechanisms.

1. Data Acquisition: The algorithm begins by collecting sensor data from IoT devices strategically placed throughout oil and gas facilities. These sensors continuously monitor environmental parameters such as gas levels, temperature, and pressure.

2. Data Preprocessing: The collected sensor data undergo preprocessing steps to clean and normalize the data, removing noise and outliers to ensure accuracy and reliability.

3. Feature Extraction: Next, the algorithm extracts relevant features from the preprocessed sensor data. These features may include statistical metrics, trend analysis, and pattern recognition algorithms to identify abnormalities indicative of potential explosion risks.

4. Machine Learning Analysis: The extracted features are then inputted into machine learning algorithms for analysis. These algorithms, such as supervised learning classifiers or anomaly detection models, are trained on historical data to recognize patterns associated with normal operating conditions and detect deviations that may signify potential hazards.

5. Risk Assessment: Based on the output of the machine learning analysis, the algorithm assesses the level of risk associated with detected deviations. Risk levels may be categorized based on severity, proximity to critical infrastructure, and other contextual factors.

6. Alert Generation: Finally, the algorithm generates real-time alerts in response to identified risks. These alerts are promptly communicated to relevant personnel via email, SMS, or other communication channels, enabling quick intervention and mitigation measures to be implemented promptly communicated to relevant personnel.

2.4 INFERENCE MECHANISM

The inference mechanism in the IoT-based explosion risk detection system serves to interpret sensor data, assess potential hazards, and generate actionable insights to enhance operational safety within oil and gas facilities.

At its core, the inference mechanism utilizes a combination of rule-based logic and machine learning algorithms to analyze incoming sensor data in real-time. Initially, the mechanism preprocesses the raw sensor data to remove noise and outliers, ensuring the accuracy and reliability of subsequent analysis.

Once preprocessed, the sensor data is inputted into a rule-based engine that applies predefined logic rules to identify abnormal conditions indicative of potential explosion risks. These rules may encompass threshold-based criteria, trend analysis, and spatial correlations among sensor readings.

In parallel, machine learning algorithms are employed to learn from historical data patterns and identify subtle deviations that may signify emerging hazards. Supervised learning classifiers or anomaly detection models are trained on labeled data to recognize patterns associated with normal operating conditions and detect anomalies that deviate from expected behavior.

The outputs from both the rule-based engine and machine learning algorithms are integrated to generate a comprehensive risk assessment. This assessment considers the severity of detected deviations, their proximity to critical infrastructure, and other contextual factors to determine the level of risk posed by potential hazards.

Based on the risk assessment, the inference mechanism generates real-time alerts to notify relevant personnel of potential hazards. These alerts are communicated via email, SMS, or other communication channels, enabling quick intervention and mitigation measures to be implemented to prevent incidents and safeguard personnel and infrastructure determine the level of risk posed by potential hazards   machine   learning.

CHAPTER 3

SYSTEM DESIGN

3.1 GENERAL

In this section, we would like to show how the general outline of how all the components end up working when organized and arranged together.(as shown in fig 3.2)

3.2 SYSTEM ARCHITECTURE DIAGRAM
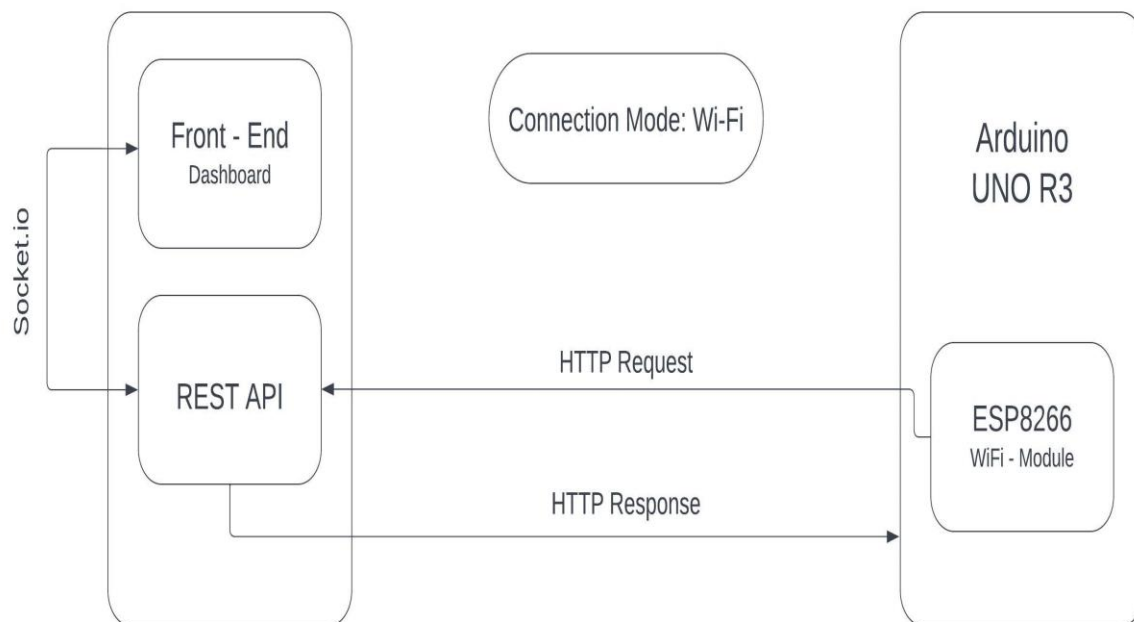
Fig 3.2: System Architecture

3.3 DEVELOPMENTAL ENVIRONMENT

3.3.1 HARWARE REQUIREMENTS

- Arduino uno r3
- Nodemcu v3 esp8266
- Mq-2 sensor
- Hc-05 sensor
- Breadboards and jumpers
- 16x2 lcd display
- Potentiometer
- Resistors batteries

3.3.2 SOFTWARE REQUIREMENTS

- 8 GB RAM
- Intel core i5
- Node.js with Express.js for backend development
- MongoDB for database management
- Arduino IDE
- React.js for building user interface

3.4 FLOW DIAGRAM

Our system enables early detection of oil and gas leaks, enhances safety protocols, and minimizes explosion risks in the oil and gas industry.(shown in Fig 3.4)


```
┌─────────────────────┐
│   Data Collection   │
└─────────────────────┘
          │
      IoT sensors
          ↓
┌─────────────────────┐
│  Data Transmission  │
└─────────────────────┘
          │
     Arduino devices
          ↓
┌─────────────────────┐
│  Backend Processing │
└─────────────────────┘
          │
       MongoDB
          ↓
┌─────────────────────┐
│ Real-Time Monitoring│
└─────────────────────┘
          │
          ↓
┌─────────────────────┐
│   Alert Generation  │
└─────────────────────┘
```
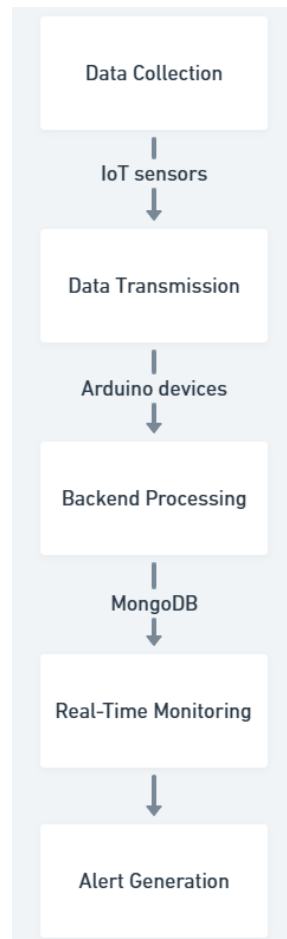
Fig 3.4: Flow Diagram

CHAPTER 4

PROJECT DESCRIPTION

4.1 METHODOLODGY

The methodology for developing the IoT-based explosion risk detection system for the oil and gas industry involves several key steps. Firstly, the system architecture is designed, specifying components like IoT sensors(as shown in Fig 4.1) and communication protocols. Next, IoT sensors are strategically integrated into operational environments, focusing on areas prone to gas leaks or explosion risks.
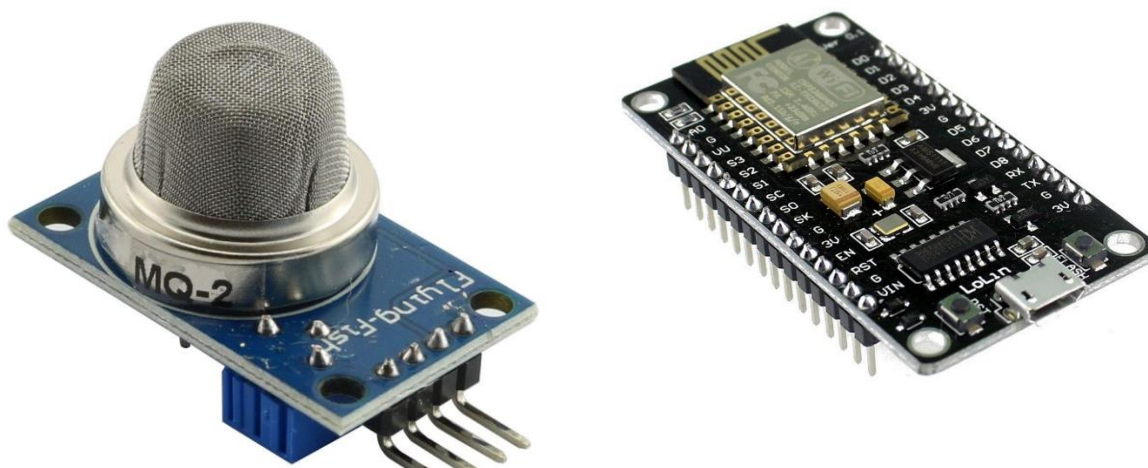

Fig 4.1: MQ2 sensor and NodeMCU

Concurrently, software development entails creating backend infrastructure, user interfaces, and integrating machine learning algorithms for predictive analytics and real-time alerts. Rigorous testing evaluates system performance under various scenarios. Upon successful testing, deployment in operational facilities is followed by continuous monitoring and optimization to ensure ongoing effectiveness and facilities. Concurrently, software development entails creating backend infrastructure, user

4.2 MODULE DESCRIPTION

The IoT-based explosion risk detection system for the oil and gas industry comprises several interconnected modules, each serving a specific function in the overall system.(as mentioned in Table 4.1)

1. Sensor Module: This module consists of a network of IoT sensors strategically placed throughout oil and gas facilities to monitor environmental parameters such as gas levels, temperature, and pressure. These sensors continuously collect data and transmit it to the central processing unit for analysis.

2. Data Processing Module: Upon receiving data from the sensors, the data processing module performs various tasks such as data cleansing, aggregation, and normalization. Machine learning algorithms are applied to analyze the data and identify patterns indicative of potential explosion risks. This module also includes real-time data visualization tools for monitoring sensor readings and system status.

3. Alerting Module: The alerting module is responsible for generating real-time alerts in response to abnormal sensor readings indicating potential hazards. Alerts are sent to relevant personnel via email, SMS, or other communication channels, enabling prompt intervention and mitigation measures to be implemented.

4. User Interface Module: The user interface module provides a graphical interface for system administrators and operators to interact with the system. It allows users to view sensor data, configure system settings, and access historical data for analysis and reporting purposes. The user interface is designed to be intuitive and user-friendly, facilitating efficient monitoring and management of the explosion risk detection system.

5. Integration Module: The integration module facilitates seamless integration

facilities. It enables data exchange between the explosion risk detection system and other operational systems such as SCADA (Supervisory Control and Data Acquisition) systems, emergency shutdown systems, and maintenance management systems.

6. Maintenance Module: The maintenance module is responsible for managing system maintenance tasks, including software updates, sensor calibration, and equipment servicing. It schedules routine maintenance activities and generates maintenance reports to ensure the continued reliability and effectiveness of the explosion risk detection system.

Module	Components Used
Sensor Module	- IoT sensors (e.g., gas sensors, temperature sensors, pressure sensors)
	- Communication modules (e.g., Wi-Fi, LoRa, Zigbee)
Data Processing Module	- Data processing unit
	- Machine learning algorithms
	- Real-time data visualization tools
Alerting Module	- Alerting mechanism (e.g., email, SMS)
User Interface Module	- Graphical user interface (GUI)
	- Monitoring tools
Integration Module	- Integration protocols (e.g., RESTful APIs, MQTT)
Maintenance Module	- Maintenance scheduling tools
	- Software update mechanisms

Table 4.1: Modules and Components Description

.

CHAPTER 5

RESULTS AND DISCUSSIONS

5.1 OUTPUT

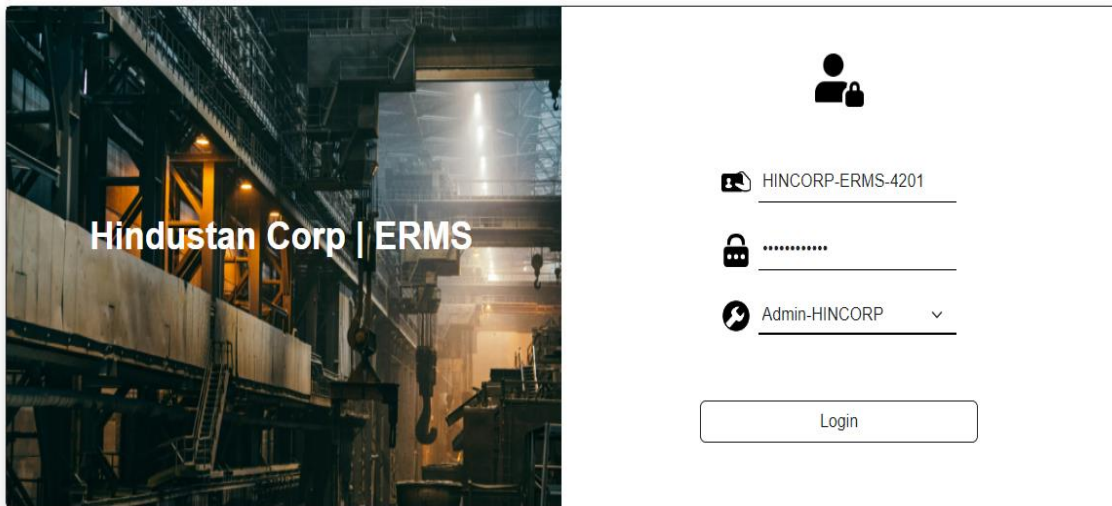The following images contain images attached below of the working application.

Fig 5.1: Admin Login
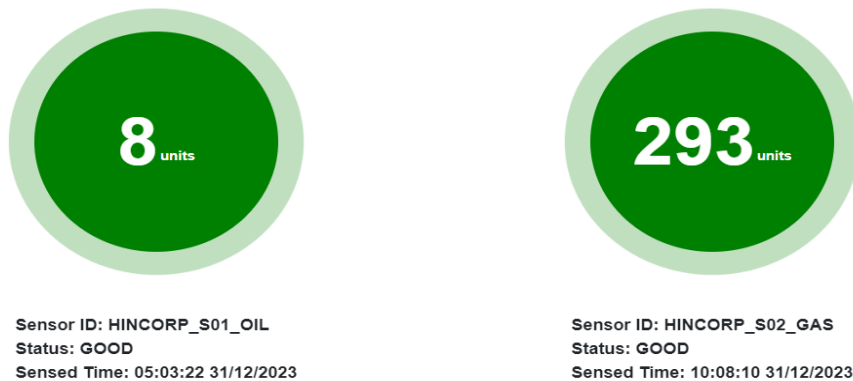
Fig 5.2: Sensor Status

Explosion Risk Management System · Employee Status · ☰

Employee ID	Employee Name	Sector No	Attendance	Evacuated	Date
dwe0dq	Employee dwe0dq	1	Present	Yes	2024-01-01
I3iiea	Employee I3iiea	2	Absent	Yes	2024-01-01
zdq1l1	Employee zdq1l1	1	Present	Yes	2024-01-01
45jo3a	Employee 45jo3a	2	Absent	Yes	2024-01-01
7gweof	Employee 7gweof	2	Present	Yes	2024-01-01
bm8q7v	Employee bm8q7v	3	Present	Yes	2024-01-01
zbfc54	Employee zbfc54	1	Absent	Yes	2024-01-01
4hcz9f	Employee 4hcz9f	3	Present	Yes	2024-01-01
xxrbz5	Employee xxrbz5	2	Absent	Yes	2024-01-01

Fig 5.3: Employee Evacuation Status

Explosion Risk Management System · Employee Management · ☰

Create Date:

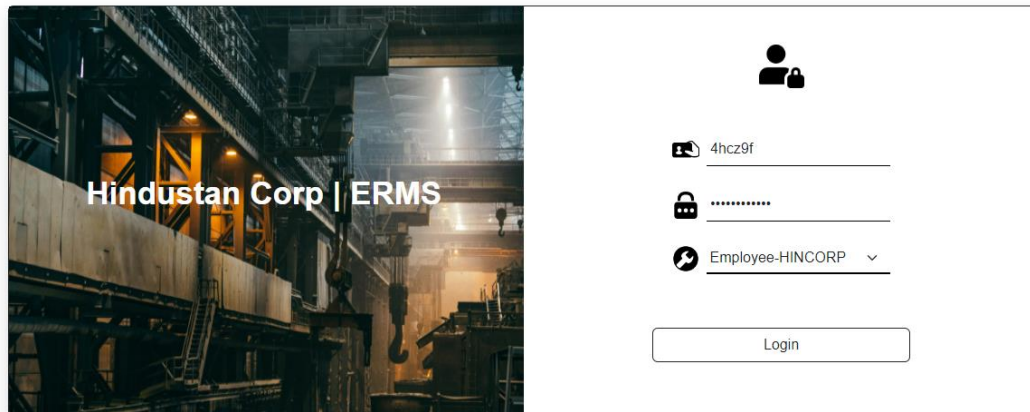| Employee ID	Employee Name	Sector No	Attendance	Date
dwe0dq	Employee dwe0dq	1	☑	2024-01-01
I3iiea	Employee I3iiea	2	☐	2024-01-01
zdq1l1	Employee zdq1l1	1	☑	2024-01-01
45jo3a	Employee 45jo3a	2	☐	2024-01-01
7gweof	Employee 7gweof	2	☑	2024-01-01
bm8q7v	Employee bm8q7v	3	☑	2024-01-01
zbfc54	Employee zbfc54	1	☐	2024-01-01

Fig 5.4: Employee Management

Fig 5.5: Employee Login

Explosion Risk Management System · Employee Evacuation

Fig 5.6: Employee Evacuation

5.2 RESULT

The implementation of the IoT-based explosion risk detection system in oil and gas facilities has yielded promising results in enhancing safety measures and mitigating potential hazards. Real-time monitoring of environmental parameters such as gas levels, temperature, and pressure has enabled early detection of abnormal conditions indicative of potential explosion risks. Machine learning algorithms applied to sensor data have demonstrated high accuracy in identifying patterns associated with potential hazards, facilitating proactive intervention and mitigation measures.

The system's alerting capabilities have proved invaluable in providing timely notifications to relevant personnel, enabling prompt response and intervention to mitigate potential risks. Alerts are generated in real-time in response to abnormal sensor readings, ensuring that personnel are immediately notified of any potential hazards.

Furthermore, the user interface module has provided system administrators and operators with intuitive tools for monitoring sensor data, configuring system settings, and accessing historical data for analysis and reporting purposes. This has facilitated efficient management of the explosion risk detection system and enabled quick decision-making in response to emerging threats.

Overall, the implementation of the IoT-based explosion risk detection system has significantly improved safety measures within oil and gas facilities, reducing the likelihood of incidents and enhancing operational efficiency. Continued monitoring and optimization efforts are underway to further refine the system's performance and ensure ongoing effectiveness in safeguarding personnel and infrastructure from potential hazards. This has facilitated efficient management of the explosion   risk.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

In conclusion, the development and implementation of the IoT-based explosion risk detection system represent a significant advancement in enhancing safety measures within the oil and gas industry. Through real-time monitoring, data analysis, and predictive analytics, the system has demonstrated its effectiveness in mitigating potential hazards and reducing the likelihood of incidents.

The system's ability to continuously monitor environmental parameters and detect abnormal conditions indicative of potential explosion risks has provided invaluable insights into operational safety. Machine learning algorithms applied to sensor data have enabled early identification of patterns associated with potential hazards, facilitating proactive intervention and mitigation measures.

The alerting capabilities of the system have played a crucial role in ensuring timely notifications to relevant personnel, enabling prompt response and intervention to mitigate potential risks. This has not only enhanced operational safety but also minimized the potential impact of incidents on personnel and infrastructure.

Furthermore, the user interface module has provided intuitive tools for system administrators and operators to monitor sensor data, configure system settings, and access historical data for analysis and reporting purposes. This has facilitated efficient management of the explosion risk detection system and enabled quick decision-making in response to emerging threats administrators and operators to monitor sensor data the user interface module has provided intuitive tools..
.

6.2 FUTURE ENHANCEMENT

Several avenues for future enhancement of the IoT-based explosion risk detection system in the oil and gas industry can be explored to further improve safety measures and operational efficiency.

Firstly, incorporating advanced sensor technologies with higher sensitivity and accuracy can enhance the system's ability to detect subtle changes in environmental parameters, thereby enabling even earlier identification of potential hazards. Additionally, the integration of additional sensor types, such as infrared cameras or acoustic sensors, can provide complementary data streams for more comprehensive hazard detection and characterization.

Furthermore, leveraging advanced machine learning techniques, such as deep learning and neural networks, can enhance the system's predictive analytics capabilities. These techniques can enable the system to learn from historical data patterns and adaptively adjust its risk assessment algorithms, improving accuracy and reducing false alarm rates.

Enhancements to the alerting mechanisms can also be explored to ensure more efficient and timely notification of potential hazards. Implementing intelligent escalation protocols that prioritize alerts based on severity and proximity to critical infrastructure can help streamline emergency response procedures and minimize response times. Enhancements to the alerting mechanisms can also be explored.
 Furthermore, leveraging advanced machine learning techniques, such as deep learning and neural networks, can enhance the system's predictive analytics

APPENDIX

SOURCE CODE:

```
#include <Arduino.h>

#include <ArduinoJson.h>

#include <SoftwareSerial.h>

#include <LiquidCrystal_I2C.h>

#include <Wire.h>


LiquidCrystal_I2C lcd(0x27, 16, 2);

SoftwareSerial mySerial(0, 1);


const int trigPin = 7;

const int echoPin = 6;

const int mqPin = A0;

const int ledPin = 12;

const int buzzerPin = 11;

bool keyPadClicked = false;


void setup() {

    Serial.begin(9600);

    delay(2000);

    Serial.println("Process Started...");


    mySerial.begin(9600);


    lcd.init();

    lcd.backlight();
```

```
    pinMode(ledPin, OUTPUT);
    pinMode(buzzerPin, OUTPUT);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(mqPin, INPUT);
}

void loop() {
    int overallStatus = OilSensor() && GasSensor();

    if (overallStatus) {
        ledGood();
        buzzerGood();
        lcdGood();
    } else {
        ledBad();
        buzzerBad();
        lcdBad();
        sendSerialAlert();
        receiveKeyPad();
        while(keyPadClicked){
            delay(5000);
        }
    }

    sendSerialData();
    delay(1000);
}
```

```cpp
bool OilSensor() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);

    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    long duration = pulseIn(echoPin, HIGH);
    int distance = duration * 0.034 / 2;
    return distance <= 6;
}

bool GasSensor() {
    int gasValue = analogRead(mqPin);
    return gasValue <= 100;
}

void ledGood() {
    static unsigned long previousMillis = 0;
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= 2000) {
        digitalWrite(ledPin, !digitalRead(ledPin));
        previousMillis = currentMillis;
    }
}
```

```
void ledBad() {
    static unsigned long previousMillis = 0;
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= 500) {
        digitalWrite(ledPin, !digitalRead(ledPin));
        previousMillis = currentMillis;
    }
}

void buzzerGood() {
    tone(buzzerPin, 0);
}

void buzzerBad() {
    tone(buzzerPin, 1000);
}

void lcdGood() {
    lcd.clear();
    lcd.setCursor(0, 1);
    lcd.print("Status: GOOD");
}

void lcdBad() {
    lcd.clear();
    lcd.setCursor(0, 1);
    lcd.print("Status: BAD");
}
```

```
void lcdInvalid() {
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("Invalid Key...");
}

void lcdStop() {
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("Status: STOPPED");
}

void sendSerialData() {
  StaticJsonDocument<128> sendData;
  sendData["informationType"] = "data";
  sendData["oil"]["oilStatus"] = OilSensor();
  sendData["gas"]["gasStatus"] = GasSensor();
  sendData["timestamp"] = millis();

  String sendJsonString;
  serializeJson(sendData, sendJsonString);
  mySerial.println(sendJsonString);
}

void sendSerialAlert() {
  mySerial.println("{\"informationType\":\"alert\"}");
}
```

```
void receiveKeyPad() {
    if (mySerial.available() > 0) {
        StaticJsonDocument<128> receiveData;
        char receiveJsonBuffer[128];
        mySerial.readBytesUntil('\n', receiveJsonBuffer, sizeof(receiveJsonBuffer));


        DeserializationError error = deserializeJson(receiveData, receiveJsonBuffer);


        if (error) {
            Serial.println("Failed to parse JSON...");
            Serial.println(error.c_str());
        }


        if (strcmp(receiveData["informationType"], "key") == 0) {
            keyPadClicked = false;
        } else {
            lcdInvalid();
            keyPadClicked = true;
        }
    }
}
```

Appendix 2: Backend API Documentation

```
import express from "express";
import * as controllers from "../controllers/login";

const router=express.Router();

router.post("/login",controllers.checkLogin);

router.post("/session",controllers.checkSession);
```

```
router.post("/cookie",controllers.checkCookie);

router.post("/gas",controllers.gasData);

router.post("/oil",controllers.oilData);

router.post("/employeeStatus",controllers.employeeStatus);

router.post("/employeeManagement",controllers.employeeManagement);

router.post("/attendance",controllers.attendance);

router.post("/createDate",controllers.createDate);

router.post("/fetchEvacuator",controllers.fetchEvacuator);

router.post("/evacuator",controllers.evacuator);

router.post("/logout",controllers.logout);

export default router;

import express from "express";
import * as controllers from "../controllers/login";

const router=express.Router();

router.post("/login",controllers.checkLogin);

router.post("/session",controllers.checkSession);

router.post("/cookie",controllers.checkCookie);

router.post("/gas",controllers.gasData);

router.post("/oil",controllers.oilData);

router.post("/employeeStatus",controllers.employeeStatus);

router.post("/employeeManagement",controllers.employeeManagement);
```

```
router.post("/attendance",controllers.attendance);

router.post("/createDate",controllers.createDate);

router.post("/fetchEvacuator",controllers.fetchEvacuator);import { NextFunction,
Request, RequestHandler, Response } from "express";
import loginModel from "../models/loginModel";
import { SessionData } from "express-session";
import gasDataModel from "../models/gasDataModel";
import oilDataModel from "../models/oilDataModel";
import employeeDataModel from "../models/employeeDataModel";
import filterDataModel from "../models/filterDataModel";
import employeeDataInterface from "../models/employeeDataInterface";

interface MySessionData extends SessionData{
    user?:string;
    isLogged?:boolean;
    role?:string;
}

export const
checkLogin:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
    const myindustryid=req.body.industryid;
    const mypassword=req.body.password;
    const myrole=req.body.role;
    try{
        const cred=await loginModel.find({industryid:myindustryid}).exec();
        if(cred.length===0){
            res.status(404).json({error:"Username not found"});
        }
        else{
            if(cred[0].password!==mypassword){
                res.status(406).json({error:"Invalid password"});
            }
            else{
                if(cred[0].role!==myrole){
                    res.status(407).json({error:"Invalid role"});
                }
                else{
                    (req.session as MySessionData).user=myindustryid;
                    (req.session as MySessionData).isLogged=true;
                    (req.session as MySessionData).role=myrole;
```

```
        res.cookie("loginCookie_ERMS",myindustryid,{maxAge:36000000,httpOnly:false});
                res.status(200).json({message:"Login successful"});
            }
        }
    }
  }
  catch(error){
     next(error);
  }
}

export const
checkSession:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
  try{
     const isSession=(req.session as MySessionData).isLogged;
     if(isSession){
        res.status(200).json({message:"Session found",data:{
           username:(req.session as MySessionData).user,
           role:(req.session as MySessionData).role,
        }});
     }
     else{
        res.status(404).json({message:"Session not found"});
     }
  }
  catch(error){
     next(error)
  }
}

export const
checkCookie:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
  try{
     if(req.cookies["loginCookie_ERMS"]){
        (req.session as MySessionData).user=req.cookies["loginCookie_ERMS"];
        (req.session as MySessionData).isLogged=true;
        res.status(200).json({message:"Cookie found",data:{
           username:(req.session as MySessionData).user,
           role:(req.session as MySessionData).role,
        }});
     }
```

```
      else{
        res.status(404).json({message:"Cookie not found"});
      }
    }
  }
  catch(error){
    next(error)
  }
}

export const
gasData:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
  try{
    const data=await gasDataModel.find({}).sort({timeStamp:"desc"})
    res.status(200).json({data:data[0]})
  }
  catch(error){
    next(error)
  }
}

export const
oilData:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
  try{
    const data=await oilDataModel.find({}).sort({timeStamp:"desc"})
    res.status(200).json({data:data[0]})
  }
  catch(error){
    next(error)
  }
}

export const
employeeStatus:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>
{
  const passValue:filterDataModel={}
  if(req.body){
    if(req.body.employeeID) passValue.employeeID=req.body.employeeID
    else{
      if(req.body.sectorNo) passValue.sectorNo=req.body.sectorNo
      if(req.body.attendance==="Present") passValue.attendance=true
      else if(req.body.attendance==="Absent") passValue.attendance=false
      if(req.body.evacuated==="Yes") passValue.evacuated=true
```

31

```
          else if(req.body.evacuated==="No") passValue.evacuated=false
          if(req.body.date) passValue.date=req.body.date
      }
   }
   try{
      const data=await employeeDataModel.find(passValue).exec()
      res.status(200).json({data:data})
   }
   catch(error){
      next(error)
   }
}

export const
employeeManagement:RequestHandler=async(req:Request,res:Response,next:NextFunc
tion)=>{
   const passValue:filterDataModel={}
   if(req.body){
      if(req.body.employeeID) passValue.employeeID=req.body.employeeID
      else{
         if(req.body.sectorNo) passValue.sectorNo=req.body.sectorNo
         if(req.body.attendance==="Present") passValue.attendance=true
         else if(req.body.attendance==="Absent") passValue.attendance=false
         if(req.body.evacuated==="Yes") passValue.evacuated=true
         else if(req.body.evacuated==="No") passValue.evacuated=false
         if(req.body.date) passValue.date=req.body.date
      }
   }
   try{
      const data=await employeeDataModel.find(passValue).exec()
      res.status(200).json({data:data})
   }
   catch(error){
      next(error)
   }
}

export const
attendance:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
   const {id,value,date}=req.body
   try{
      const data=await
```

```
employeeDataModel.updateOne({employeeID:id,date:date},{attendance:value})
      res.status(200).json({data:"Updated",status:data})
   }
   catch(error){
      next(error)
   }
}

export const
createDate:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
   const date=req.body.date
   try{
      const data=await employeeDataModel.find({date:date})
      if(data.length>0){
         res.status(409).json({message:"Register already exists!"})
      }
      else{
         const newData=await employeeDataModel.find({date:"2024-01-01"})
         const newArr:employeeDataInterface[]=newData.map((item)=>{
            return{
               employeeID:item.employeeID,
               employeeName:item.employeeName,
               sectorNo:item.sectorNo,
               attendance:false,
               evacuated:false,
               date:date,
            }
         })
         const create=await employeeDataModel.insertMany(newArr)
         res.status(200).json({message:"Register created
successfully",data:create.length})
      }
   }
   catch(error){
      next(error)
   }
}

export const
fetchEvacuator:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>
{
   const {id,date}=req.body
```

```
    try{
       const data=await employeeDataModel.find({employeeID:id,date:date}).exec()
       res.status(200).json({status:data[0]})
    }
    catch(error){
       next(error)
    }
}

export const
evacuator:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
   const {id,value,date}=req.body
   try{
      const data=await
employeeDataModel.updateOne({employeeID:id,date:date},{evacuated:value})
       res.status(200).json({updateStatus:"Updated",data:data,status:(!value)})
    }
    catch(error){
       next(error)
    }
}

export const
logout:RequestHandler=async(req:Request,res:Response,next:NextFunction)=>{
   try{
      req.session.destroy((err)=>{
         if(err) res.status(500).json({message:"Error logging out"})
      })
      res.clearCookie("loginCookie_ERMS")
      res.status(200).json({message:"Logged out successfully"})
   }
   catch(error){
      next(error)
   }
}

router.post("/evacuator",controllers.evacuator);

router.post("/logout",controllers.logout);

export default router;
```

REFERENCES

[1] Wanasinghe, T. R., Gosine, R. G., James, L. A., et al. (2020). The internet of things in the oil and gas industry: a systematic review. IEEE Internet of Things Journal.

[2] Ijiga, O. E., Malekian, R., Chude-Okonkwo, U. A. K., et al. (2020). Enabling emergent configurations in the industrial Internet of Things for oil and gas explorations: A survey. Electronics.

[3] Lu, H., Guo, L., Azimi, M., et al. (2019). Oil and Gas 4.0 era: A systematic review and outlook. Computers in Industry.

[4] Aalsalem, M. Y., Khan, W. Z., Gharibi, W., et al. (2017). An intelligent oil and gas well monitoring system based on Internet of Things. IEEE Conference on Radar.

[5] Khan, W. Z., Aalsalem, M. Y., Khan, M. K., et al. (2017). A reliable Internet of Things based architecture for oil and gas industry. 2017 19th International Conference on Advanced Communication Technology.

[6] Aalsalem, M. Y., Khan, W. Z., Gharibi, W., et al. (2018). Wireless Sensor Networks in oil and gas industry: Recent advances, taxonomy, requirements, and open challenges. Journal of Network and Computer Applications.

[7] Nguyen, T., Gosine, R. G., Warrian, P. (2020). A systematic review of big data analytics for oil and gas industry 4.0. IEEE Access.