**IOT BASED SAFETY KIT FOR INDUSTRIAL WORKERS**

# A PROJECT REPORT

***Submitted by***

# JERROLD GIDEON S - 312319104058

**JERRY BRITTO J – 312319104059**

***in partial fulfilment for the award of the Degree of***

# BACHELOR OF ENGINEERING

# *In*

**COMPUTER SCIENCE AND ENGINEERING**



# St. JOSEPH’S COLLEGE OF ENGINEERING

**(An Autonomous Institution)**

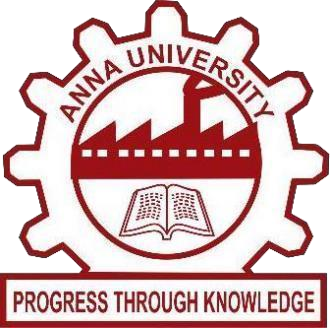
**St. Joseph’s Group of Institutions**

**OMR, Chennai 600 119**

# ANNA UNIVERSITY:: CHENNAI 600 025

**MARCH 2023**

# ANNA UNIVERSITY :: CHENNAI



**BONAFIDE CERTIFICATE**

*Certified that this project report on* ***IOT BASED SAFETY KIT FOR INDUSTRIAL WORKERS*** *is the bonafide work of* ***S.******JERROLD GIDEON***

***(312319104058)*** *and* ***J. JERRY BRITTO(312319104059)*** *who carried out the project under my supervision during the academic year 2022 - 2023.*

# Signature of the Head of the Department

# Dr. A. Chandrasekar M.E., Ph. D HOD

Department of Computer Science and Engineering

St. Joseph’s College of Engineering OMR,

Chennai- 600119.

# Signature of the Supervisor

**Mr. S. Vinu M.E.,**

# Assistant Professor

Department of Computer Science and Engineering

St. Joseph’s College of Engineering OMR,

Chennai- 600119.

# ACKNOWLEDGEMENT

At the outset, we would like to express our sincere gratitude to our beloved ***Chairman,* Dr.B. Babu Manoharan M.A.,M.B.A.,Ph.D *St.Joseph’s Group of Institutions*** for his constant guidance and support to the student community and the Society.

We would like to express our hearty thanks to our respected ***Managing Director,* Mrs. S. Jessie Priya M.Com. *St. Joseph’s Group of Institutions*** for her kind encouragement and blessings.

We wish to express our sincere thanks to our ***Executive Director* Mr. B. Shashi Sekar, M.Sc. *St. Joseph’s Group of Institutions*** for providing ample facilities in the institution.

We would like to express sincere gratitude to our beloved and highly respected ***Principal* Dr. Vaddi Seshagiri Rao M.E., M.B.A., Ph.D., F.I.E.**for his inspirational ideas during the course of the project.

We express my sincere gratitude to our beloved ***Dean (Student Affairs)* Dr. V. Vallinayagam M.Sc., M.Phil., Ph.D.,** *and* ***Dean (Academics*) Dr. G. Sreekumar M.Sc., M.Tech., Ph.D.,** for their ideas during the course of the project.

We also express my thanks and most heartfelt sense of gratitude to **Dr. A. Chandrasekar, M.E., Ph.D., *Head of the Department of Computer Science and Engineering and Dean (Research)*** for his dedication, support and encouragement for the completion of project.

We would like to acknowledge our gratitude to our supervisor **Mr. S. VINU M.E.,.,** for his excellent guidance and connoisseur's suggestion throughout the study carried out successfully.

Finally, we thank the **Faculty Members** and **our Family**, who helped and encouraged us constantly to complete the project successfully.

**ABSTRACT**

IoT (Internet of Things) safety kits are designed to provide users with a reliable and easy-to-use solution for monitoring and detecting potential safety hazards in various environments. These kits typically include a range of sensors and communication modules that are used to monitor conditions such as temperature, humidity, gas levels, and motion. In the event that any of these sensors detect a potential hazard, the kit can send alerts to the user's smartphone or other connected device, allowing them to take appropriate action. The human body's exposure to hazardous environments can have a negative impact on our health. We may be exposed to abnormal environments in our daily lives, such as industrial settings, which may be tolerable at first but may cause harmful side effects in our health in the long run. These can be avoided by monitoring the industrial areas and alerting the workers when their work environment gets abnormal; this can help the workers take the required measures. A Desktop application can be provided as a dashboard for the administrators to check and monitor with human supervision at anytime whenever necessary.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | iv |
|  | **LIST OF FIGURES** | vii |
|  | **LIST OF ABBREVIATIONS** | viii |
| **1** | **INTRODUCTION** | 1 |
|  | 1.1 Introduction to the Project | 1 |
|  | 1.2 Understanding the background and topic | 3 |
|  | 1.3 Problem statement | 4 |
|  | 1.4 Project goals | 4 |
| **2** | **LITERATURE SURVEY** | 5 |
|  | 2.1 Existing System | 5 |
|  | 2.1.1 Safety Training Through Computer-Aided Technologies | 5 |
|  | 2.1.2 Integrating BIM and Safety | 5 |
|  | 2.1.3 Proximity detection devices | 6 |
|  | 2.1.4 Wearable sensing devices | 6 |
|  | 2.1.5 Problems in the existing system | 6 |
|  | 2.2 Related Works | 7 |
|  | 2.3 Proposed System | 8 |
|  | 2.3.1 Modules in proposed system | 9 |
| **3** | **SYSTEM REQUIREMENTS** | 10 |
|  | 3.1 Hardware Requirements | 10 |
|  | 3.2 Software Requirements | 10 |
|  | 3.3 Software Features | 11 |
| **4** | **SYSTEM DESIGN** | 15 |
|  | 4.1 System Architecture | 15 |
|  | 4.2 Data Flow | 16 |
|  | 4.3 Module Design | 16 |
| **5** | **SYSTEM IMPLEMENTATION** | 18 |
|  | 5.1 Data collection using Arduino Mega | 18 |
|  | 5.2 Uploading data to IBM Watson cloud | 19 |
|  | 5.3 Retrieving data to local Node-RED server | 20 |
|  | 5.4 Displaying the data in the application | 21 |
| **6** | **RESULTS AND EVALUATION** | 22 |
|  | 6.1 Results and Discussions | 22 |
|  | 6.2 Performance Analysis | 24 |
| **7** | **CONCLUSION** | 25 |
|  | 7.1 Conclusion | 25 |
|  | 7.2 Limitation | 26 |
|  | 7.3 Future Enhancement | 26 |
|  | **REFERENCES** | 28 |
|  | **APPENDICES** | 29 |

**LIST OF FIGURES**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **FIGURE NO** | **FIGURE DESCRIPTION** | **PAGE NO** | | **2.1** | Hardware Components | 9 | | **4.1** | System Architecture Diagram | 15 | | **4.2** | Data Flow Diagram | 16 | | **4.3** | Module Design Level 1 | 16 | | **4.4** | Module Design Level 2 | 17 | | **6.1** | Output of desktop app. | 22 | | **6.2** | Graphical representation of temperature | 23 | | **6.3** | Graphical representation of MQ135 value | 23 | | **6.4** | Graphical representation of performance of Sensors | 24 | |  |  |  |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| IOT | INTERNET OF THINGS |
| DHT | DIGITAL HUMIDITY AND TEMPERATURE |
| IBM | INTERNATIONAL BUSINESS MACHINES |
| UWP | UNIVERSAL WINDOWS PLATFORM |
|  |  |

**CHAPTER 1**

**INTRODUCTION**

* 1. **INTRODUCTION TO PROJECT**

IoT (Internet of Things) safety kits are designed to provide users with a reliable and easy-to-use solution for monitoring and detecting potential safety hazards in various environments. These kits typically include a range of sensors and communication modules that are used to monitor conditions such as temperature, humidity, gas levels, and motion. In the event that any of these sensors detect a potential hazard, the kit can send alerts to the user's smartphone or other connected device, allowing them to take appropriate action.

IoT safety kits can be designed for a range of applications, from home security to workplace safety and fire detection. For example, a home security kit might include sensors to detect motion, door and window opening/closing, and smoke or gas levels, along with a siren or other alert system to deter intruders or warn occupants of potential fires or gas leaks.

In the workplace, an IoT safety kit might be used to monitor conditions such as temperature, humidity, and air quality in order to ensure the safety and comfort of employees. In a factory setting, for example, the kit might include sensors to detect hazardous gases or chemicals, while in an office building it might monitor conditions such as indoor air quality and temperature to ensure optimal working conditions.

This project is mainly for detecting, monitoring these temperature parameters and displaying it to the industrial workers and providing timely alert to the industrial workers and admins via desktop application.

We know that India is in the list of developing countries. As in developing phase the nation is moving towards infrastructural and urbanization development. The both directly lead to increase in construction activities. Approximately 150+ employee caught in accidents out of 1000 in India.

The mass employee at site is difficult to manage with regards to safety norms and regulations. So Wearable Sensing Devices (WSD) and the Internet of Things (IoT) have been identified as emerging technologies with great potential to managing and coordinating the mass.

In Industrial plants there are some hazardous areas which contains potentially life-threatening levels of temperature parameters. On Industries wearable sensors and systems can be used for physiological monitoring, environmental sensing, proximity sensing, and location tracking of a wide range to restrict the construction hazards. Based on the results of the completed review, recommendations are made on how WSD and IoT can be effectively implemented to improve safety performance on construction sites.

Overall, IoT safety kits offer a flexible and adaptable solution for a wide range of safety applications, and are becoming increasingly popular as more and more users seek to take advantage of the benefits of smart, connected devices.

**1.2 UNDERSTANDING THE BACKGROUND AND TOPIC**

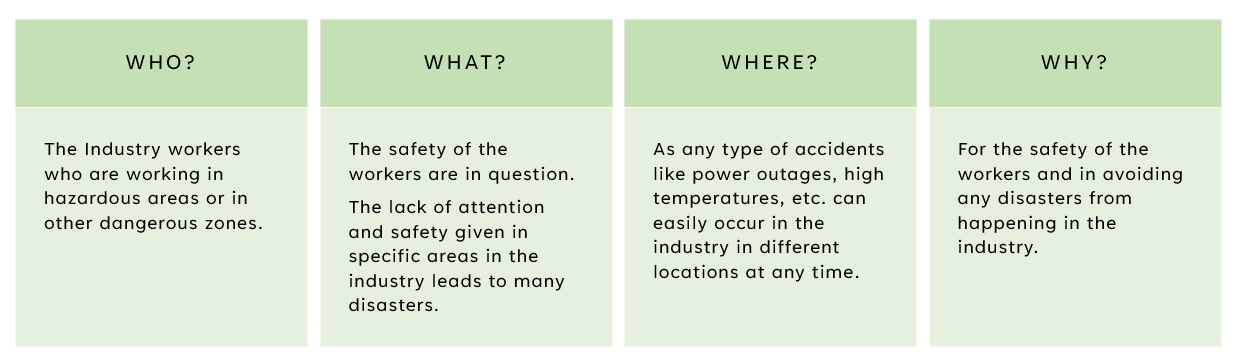
The Internet of Things or IoT is the placement of related personal computers, electronic and computing devices, objects, creatures, or people. Equipped with a unique identifier (UID), you can send information about your organization without the need for personal or PC intervention.

The most well known scratches in the work environment are slips, power outages and accidents. To solve these problems, IoT is currently used in the development business. Gadgets and sensors embedded in the development materials allow the chief to track and verify ship status via GPS, identify work examples, inactivity times, and other measurements, misuse, robbery, or fuel robbery.

A research paper “An estimate of fatal accidents in Indian construction” by Dilip kumar Patel and Kumar Neeraj Jha reports that every year 24.20% of the deaths occurring at occupational sites are caused by the construction industry, causing it to be the highest in the nation every year. It is also reported that about 80% of the construction sites in India are unsafe.

In a work environment that helps you avoid the problem and make preparations and mappings successful, taking into account the ever-changing factors. Aid workers are protected by the IoT through wearable devices that check their body and health and provide persistent data. Send alerts when dangerous areas are approaching to notify site administrators of unexpected site changes so that you can better monitor your work efficiency.

**1.3 PROBLEM STATEMENT**



**1.4 PROJECT GOALS**

1. To save the lives of workers working in hazardous areas in the industries.

2. To collect and study IoT in various sectors of Industries.

3. Providing more than one way of detecting and monitoring temperature levels in industrial plants.

4. More modern and cost-efficient approach of detecting and monitoring temperature parameters in industrial plants.

5. An efficient way for administrators working on industrial plants in detecting temperature parameters and in monitoring them at all times.

6. Easier and methodical way of alerting the workers of toxic gases, high temperatures, flammable gases, etc. present in that area.

7. Making it easier and user-friendly to use the wearable device to obtain information at all times.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 EXISTING SYSTEM**

A strong system of safety data collection, analysis, and sharing will assist the industry to understand the root causes of an event, explore existing and potential hazards, and continuously improving existing safety programs. Different countries and industries have conducted multiple reporting systems to collect, analyze, and share information with the public.

**2.1.1 Safety Training Through Computer-Aided Technologies**:

The development of virtual reality (VR), augmented reality (AR), and mixed reality (MR) have embedded worker training systems and become significant cost-effective and safer ways to educate workers. The immersive VR/AR/MR environments within computer generated simulations have also gained popularity in safety training to identify the potential hazards as well as educate moving vehicle operators on the job site. Hazardous construction scenarios can be simulated interactively with the working environment, workers’ behavior, high-risk equipment, and working sequence.

**2.1.2 Integrating BIM and Safety**:

Numerous studies and industrial applications evidenced that safety and BIM integration can assist in safety planning and execution of projects, for example to automatic checking of construction models and schedules for preventing fall-related accidents; automated scaffolding-related safety hazard identification, visualization & prevention, blind spots identification and mapping, etc.

**2.1.3 Proximity detection devices**:

Many proximity avoidance systems have been developed by utilizing various technologies, such as an ultrasonic based sensor, radio frequency identification (RFID) sensing technology, radar, GPS, and magnetic field generators, to prevent contact accidents, particularly for accidents due to being struck by equipment.

**2.1.4 Wearable sensing devices**:

A wide range of wearable devices has been applied across different industrial sectors including health care, manufacturing, etc. Some of these devices have proven to be very useful and beneficial to these industries and efforts are being made by both researchers and industry practitioners to improve on these technologies and learn from their initial implementation. With the attention being gained by wearable device worldwide, mobile devices are becoming part of everyday life and the number, types and forms of wearable devices are increasing exponentially in recent years.

**2.1.5 Problems in the existing system**

1.Cost of various types of sensors used and number of devices connected to it.

2. Using devices in an In-efficient approach for temperature detection & monitoring.

3. Utilization of VR, AR, MR systems require a large sum of money for the industry.

4. No easier (or) more modern approach for alerting the workers.

5. Multiple ways for displaying the temperature for the workers and admins to monitor.

**2.2 Related Works**

1. Jamal Mabrouki , Mourade Azrour, Driss Dhiba, Yousef Farhaoui, and Souad El Hajjaji proposed a system “IoT-Based Data Logger for Weather Monitoring Using Arduino based Wireless Sensor Networks with Remote Graphical Application and Alert” In this paper, we propose an automatic weather monitoring system that allows having dynamic and real-time climate data of a given area. The main objective of this system is sensing the climate parameters. The captured values can then be sent to remote applications or databases.No information about where we can implement this, just the monitoring thing is explained and done.
2. Pulishetty Prasad et all proposed a system “Zigbee Based Intelligent Helmet For Coal Miners” in 2017. It discovers real time monitoring with timely warning intelligence when there is a leakage of gas, rise or drop in temperature and informs control station by using new age Zigbee wireless technology.The major disadvantage of this system is that it is particularly focused on workplace safety rather on workers.
3. Mangala Nandhini. V et all proposed a system "IOT based Smart Helmet for Ensuring Safety in Industries" in 2018.This system provides actual time detection of threatening gases like CO,CH4, LPG , temperature and humidity and provides emergency alert to the control station. Wi-Fi is used to transfer data from the helmet to the monitoring station. The reliability and durability of the communication system is poor.
4. Risa Augusta Moorthy et all[3] proposed a system “Android-Based Dam Management and Monitoring System: Wireless Communication using Google Firebase” in which In a flood-prone area, water level sensors measure the water level so that drivers can be aware of the condition of the area. If the water level exceeds a predetermined limit, drivers are advised to use caution when crossing the road. The application uses google fire base to direct both drivers to pass another shortcut if the water level exceeds the limit.
5. K. Divya et all proposed a system "A Smart Helmet For Improving Safety In Mining Industry" in 2017.In case of any poisonous gas detection the helmet provides oxygen by opening a valve. It detects hazardous event, monitors and provides oxygen supplements to avoid the inhalation of poisonous gases. The helmet is too heavy since it contains oxygen cylinders which is uncomfortable to work with.

**2.3 Proposed System**

This Project comprises of an ESP32 board, LM35(Temperature), DHT-11, MQ-135, MQ-7, Flame sensor(IR) and Node RED as a cloud platform.

1. First we will make IoT kits for each Industrial room.
2. Each and every device will be connected to the cloud services.
3. We will also have an Arduino Mega to get data from these devices and send them together as one single device.
4. Through this Arduino we can get the data offline.
5. Through the cloud services we can access the data online.
6. We will be having a desktop application which can toggle between online and offline modes.
7. We can access the data via serial port in our application in offline mode.

**2.3.1 Modules in proposed system**

The different modules present in our product are:

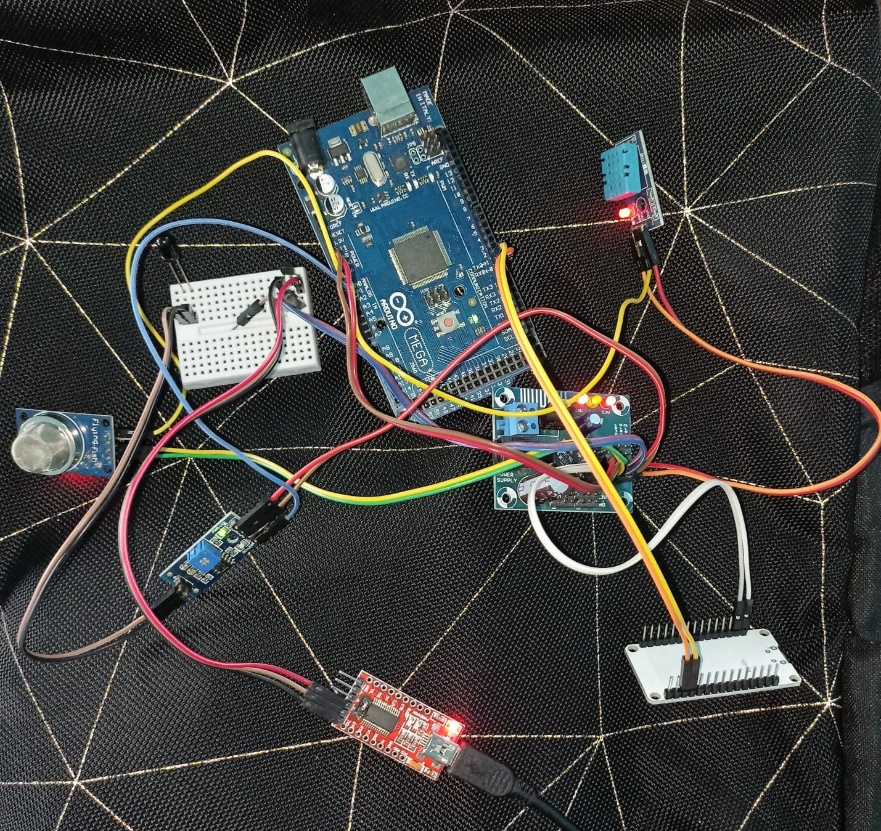
1. Temperature sensor module

2. Humidity sensor module

3. Air quality sensor module

4. Carbon Monoxide gas sensor module

5. Infra-red radiation intensity sensor module



**FIG. 2.1 HARDWARE COMPONENTS**

**CHAPTER 3**

**SYSTEM REQUIREMENTS**

**3.1 HARDWARE REQUIREMENTS**

Hardware: (minimal)

* Processor: 8th gen i3 Dual core 2.50Ghz
* RAM: 4GB
* Memory: 128GB

Hardware: (optimal)

* Processor: 10th gen i5 Dual core 3.50Ghz
* RAM: 8GB
* Memory: 256GB

**3.2 SOFTWARE REQUIREMENTS**

* OS: Windows 10, also compatible with Xbox, HoloLens, Windows Mobile
* Frontend: XAML (Extensible Application Markup Language), UI and UX
* Backend: C#
* Arduino – Embedded C++

**3.3 SOFTWARE FEATURES**

Various software used in our project are:

**1. VS community 2022:**

Microsoft Visual Studio Community 2022 is a free, fully-featured Integrated Development Environment (IDE) that is widely used by developers to create applications for Windows, web, mobile, and cloud platforms. Here are some of the new features and improvements that are included in Visual Studio Community 2022:

1. Enhanced User Interface: Visual Studio Community 2022 comes with an updated user interface that provides a more modern and streamlined experience.
2. Improved Performance: Visual Studio Community 2022 has improved performance compared to previous versions, thanks to various performance improvements such as faster solution loading, faster code analysis, and improved IntelliSense.
3. .NET 6 Support: Visual Studio Community 2022 has full support for .NET 6, the latest version of the .NET Framework. This includes support for the latest C# 10 language features.
4. Improved Debugging: Visual Studio Community 2022 includes improvements to the debugging experience, including better performance, new debugging tools, and improved support for remote debugging.
5. Code Analysis: Visual Studio Community 2022 includes improved code analysis tools that help developers identify and fix common coding issues.

**2. Arduino IDE:**

The Arduino Integrated Development Environment (IDE) is an open-source software tool used to program Arduino boards. It provides a user-friendly interface for writing, compiling, and uploading code to an Arduino board. Here are some of the features of the Arduino IDE:

1. Code Editor: The Arduino IDE includes a code editor with syntax highlighting and code completion. The editor is designed to make it easy for beginners to write and edit code.
2. Board Manager: The Arduino IDE includes a board manager that makes it easy to install and manage different boards, such as the Arduino Uno, Arduino Mega, and Arduino Nano.
3. Library Manager: The Arduino IDE includes a library manager that makes it easy to install and manage libraries. Libraries are collections of code that can be used to add functionality to your project.
4. Serial Monitor: The Arduino IDE includes a serial monitor that allows you to communicate with your board through the serial port.

**3. IBM Watson IOT:**

IBM Watson IoT Platform is a cloud-based platform that allows organizations to securely connect and manage IoT devices and collect and analyze data from those devices. Here are some of the features of IBM Watson IoT Platform:

1. Device Management: IBM Watson IoT Platform provides tools for managing IoT devices, including device registration, device monitoring, and over-the-air (OTA) updates.
2. Data Management: IBM Watson IoT Platform includes tools for collecting and managing data from IoT devices, including data filtering, aggregation, and transformation.
3. Analytics: IBM Watson IoT Platform includes tools for analyzing IoT data, including real-time analytics, predictive analytics, and machine learning.
4. Security: IBM Watson IoT Platform provides advanced security features, including secure device communication, data encryption, and access control.
5. Visualization: IBM Watson IoT Platform includes tools for visualizing IoT data, including dashboards, charts, and graphs.
6. Open Standards: IBM Watson IoT Platform supports open standards, including MQTT and HTTPS, making it easy to connect and communicate with a wide range of IoT devices.

**4. Node-RED:**

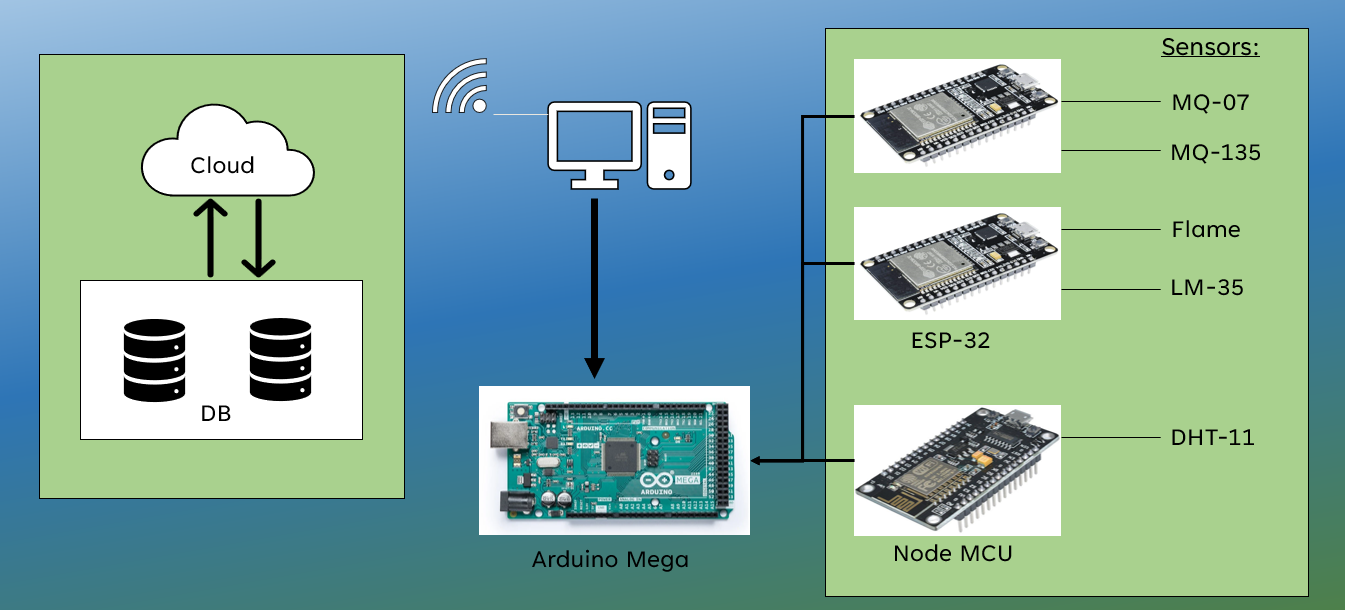
Node-RED is an open-source flow-based programming tool designed for IoT applications. Here are some of the key features of Node-RED:

1. Visual Programming: Node-RED provides a visual programming interface that allows users to create flows by connecting nodes together. This makes it easy to create complex IoT applications without need for traditional coding.
2. Easy Integration: Node-RED includes built-in support for a wide range of protocols, including MQTT, HTTP, and WebSocket. It also integrates with a wide range of IoT devices, such as sensors and actuators.
3. Large Library of Nodes: Node-RED has a large library of nodes that can be used to build IoT applications, including nodes for data transformation, device control, and data visualization.
4. Dashboard UI: Node-RED includes a dashboard UI that allows users to create custom web interfaces for their IoT applications. The dashboard UI can be used to display real-time data, control devices, and trigger actions.
5. Debugging Tools: Node-RED includes a built-in debugging tool that allows users to monitor the flow of data through their IoT applications and debug any issues.
6. Scalability: Node-RED is designed to be scalable and can be run on a single device or in a distributed environment. This makes it suitable for a wide range of IoT applications, from simple home automation to industrial-scale systems.

**CHAPTER 4**

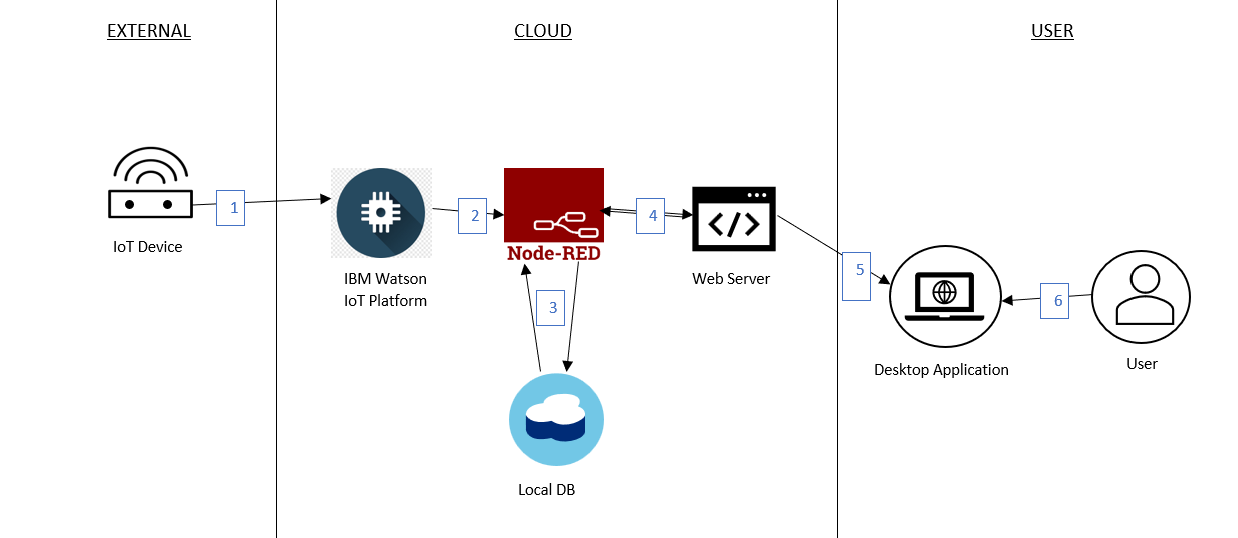
**SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE**

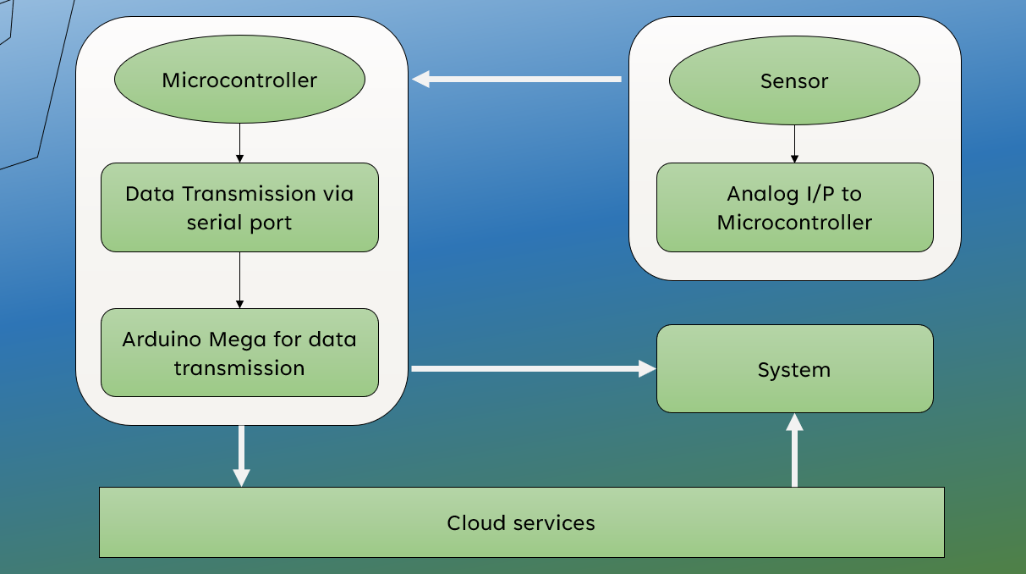


**FIG. 4.1 SYSTEM ARCHITECTURE DIAGRAM**

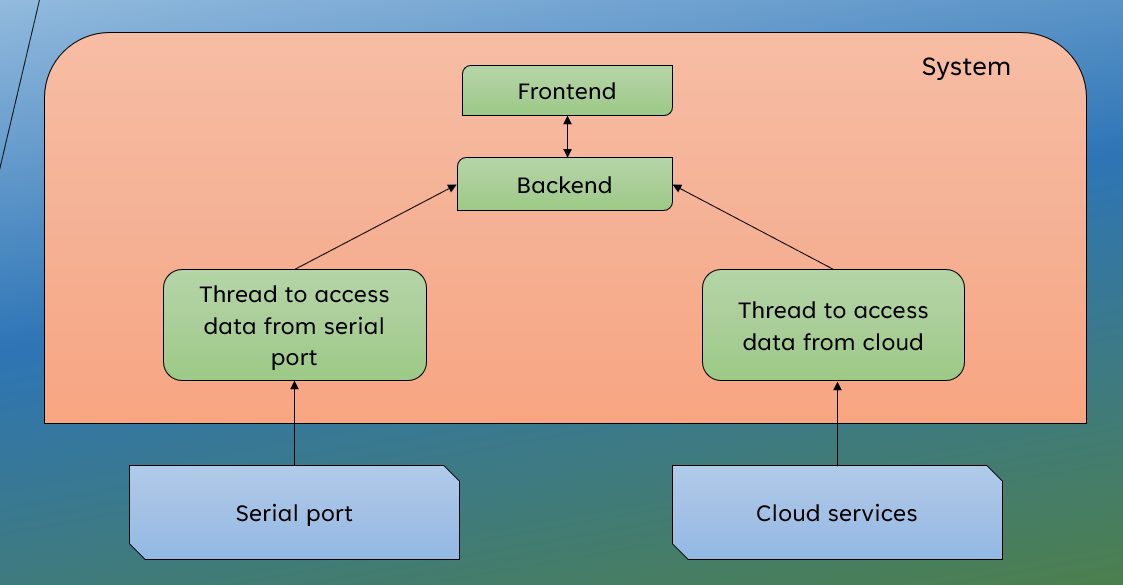
**4.2** **DATA FLOW**



**FIG. 4.2 DATA FLOW DIAGRAM**

**4.3 MODULE DESIGN**

**FIG. 4.3 MODULE DESIGN LEVEL 1**

****

**FIG. 4.4 MODULE DESIGN LEVEL 2**

**CHAPTER 5**

**SYSTEM IMPLEMENTATION**

**5.1 DATA COLLECTION USING ARDUINO MEGA**

**1. Determine the type of data you want to collect**: Before you start collecting data using Arduino Mega, you need to determine the type of data you want to collect. This could be temperature, humidity, light intensity, sound, etc.

**2. Select the appropriate sensor:** Once you have determined the type of data you want to collect, you need to select the appropriate sensor that can measure the data. Arduino Mega supports a wide range of sensors that can be used for different types of data.

**3. Connect the sensor to Arduino Mega:** After selecting the appropriate sensor, you need to connect it to Arduino Mega. This can be done by connecting the sensor to one of the input/output (I/O) pins on the board. The connection can be made using wires or connectors.

**4. Write the code:** Once the sensor is connected, you need to write the code that will enable Arduino Mega to collect data from the sensor. The code should include instructions on how to read data from the sensor and how to store the data in a suitable format.

**5. Test the system:** After writing the code, you need to test the system to ensure that it is working as expected. This can be done by uploading the code to Arduino Mega and monitoring the data that is being collected.

**6. Store the data:** Once the system is working correctly, you need to store the data that is being collected. This can be done by storing the data in a file on an SD card or transmitting it to a computer or server using a wireless or wired connection.

**5.2 UPLOADING DATA TO IBM WATSON CLOUD**

**1. Create a Watson Studio service:** After signing up, create a Watson Studio service by selecting "Create Resource" from the IBM Cloud dashboard and searching for "Watson Studio". Follow the prompts to create the service.

**2. Create a project:** Once you have created the Watson Studio service, create a new project by selecting "Create a Project" from the Watson Studio dashboard. Give the project a name and select the appropriate options for your data.

**3. Add data assets:** After creating the project, add the data assets you want to upload by selecting "Add to Project" and then "Data Asset". Choose the type of data asset you want to upload, such as a file or database, and follow the prompts to upload the data.

**4. Add a data source:** Once you have uploaded your data assets, add a data source to connect to your data. Select "Add to Project" and then "Connection" to create a connection to your data source.

**5. Import data:** After creating the connection, import the data by selecting "Import" from the data source. Choose the data asset you want to import and follow the prompts to import the data.

**6. Analyze and visualize data:** Once you have imported the data, you can analyze and visualize it using Watson Studio tools such as Jupyter Notebooks, SPSS Modeler, and Dashboarding.

* 1. **RETRIEVING DATA TO LOCAL NODE-RED SERVER**

**1. HTTP Request node:** Use this node to make HTTP requests to external APIs or web services and retrieve data in JSON or other formats. You can specify the request method (GET, POST, etc.), URL, headers, and payload in the node configuration.

**2. MQTT In node:** Use this node to subscribe to a MQTT broker and retrieve data from MQTT topics published by IoT devices or other sources. You can specify the broker configuration and topic name in the node configuration.

**3. File In node:** Use this node to read data from local files in various formats, such as CSV, JSON, or XML. You can specify the file path and encoding in the node configuration.

**4. Database nodes:** Use various database nodes available in Node-RED to connect to databases such as MySQL, MongoDB, or PostgreSQL and retrieve data from tables or collections. You can specify the database connection details, query, and output format in the node configuration.

**5.4 DISPLAYING THE DATA IN THE APPLICATION**

1. Create a new UWP project in Visual Studio and add the required references and packages to connect to the data source, such as a REST API, a database, or a local file.

2. Define the user interface using XAML controls such as ListView, GridView, or DataGrid, depending on the type and structure of the data. You can also use custom controls or third-party libraries to enhance the user experience.

3. Define the data model and the data access layer using C# classes and methods that can retrieve the data from the source and map it to the model objects. You can also use data binding or MVVM (Model-View-ViewModel) architecture to separate the data from the UI.

4. Populate the UI controls with the data retrieved from the source using data binding or code-behind methods that update the UI elements with the data values. You can also add sorting, filtering, or paging features to the UI controls to enable the user to interact with the data.

5. Handle any errors or exceptions that may occur during the data retrieval or display process and provide appropriate feedback to the user. You can use dialog boxes, notifications, or log files to report errors and debug the application.

**CHAPTER 6**

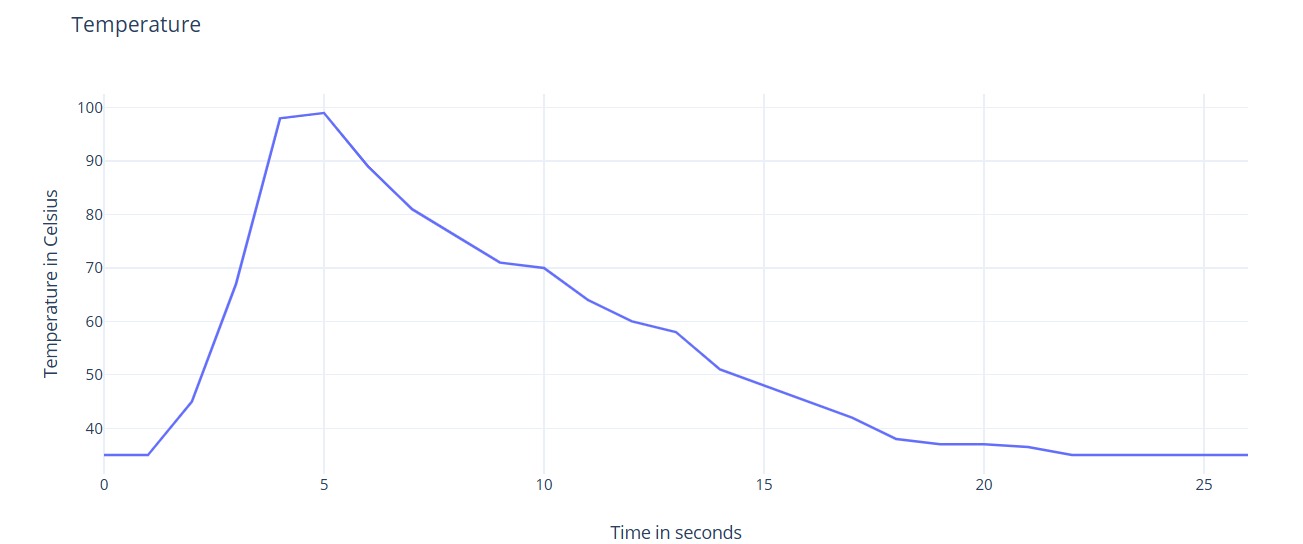
**RESULTS AND EVALUATION**

**6.1 RESULTS AND DISCUSSIONS**

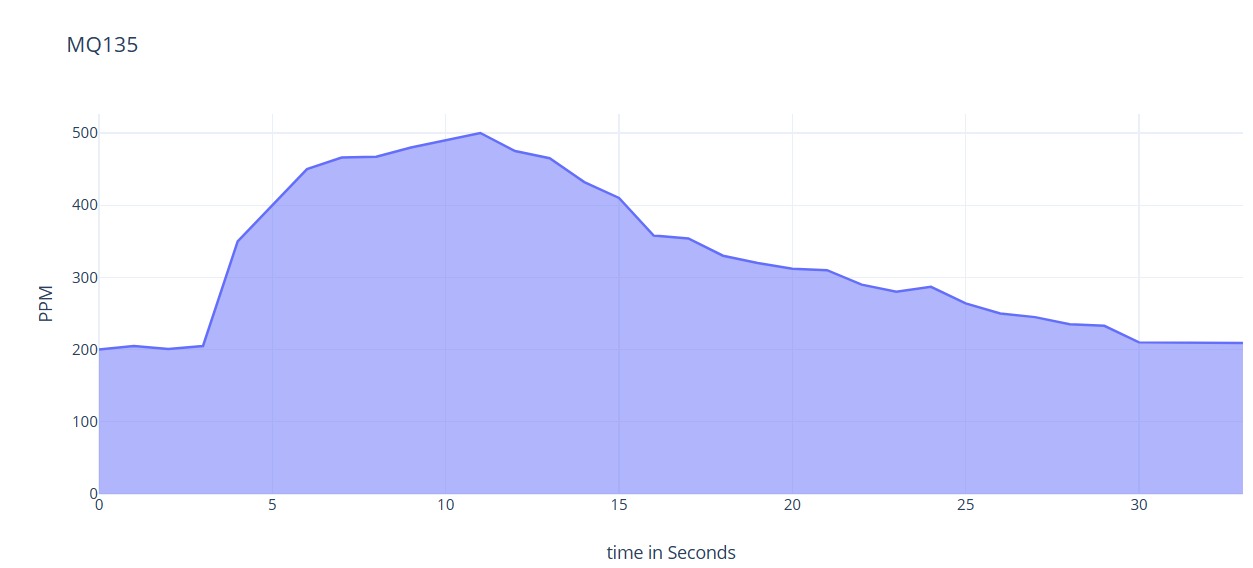
The image below shows the result of the system of our project and the values are correctly displayed on the desktop application and works successfully.

****

**FIG. 6.1 OUTPUT OF DESKTOP APPLICATION**



**FIG. 6.2 GRAPHICAL REPRESENTATION OF TEMPERATURE**



**FIG. 6.3 GRAPHICAL REPRESENTATION OF MQ135 VALUE**

**6.2 PERFORMANCE ANALYSIS**

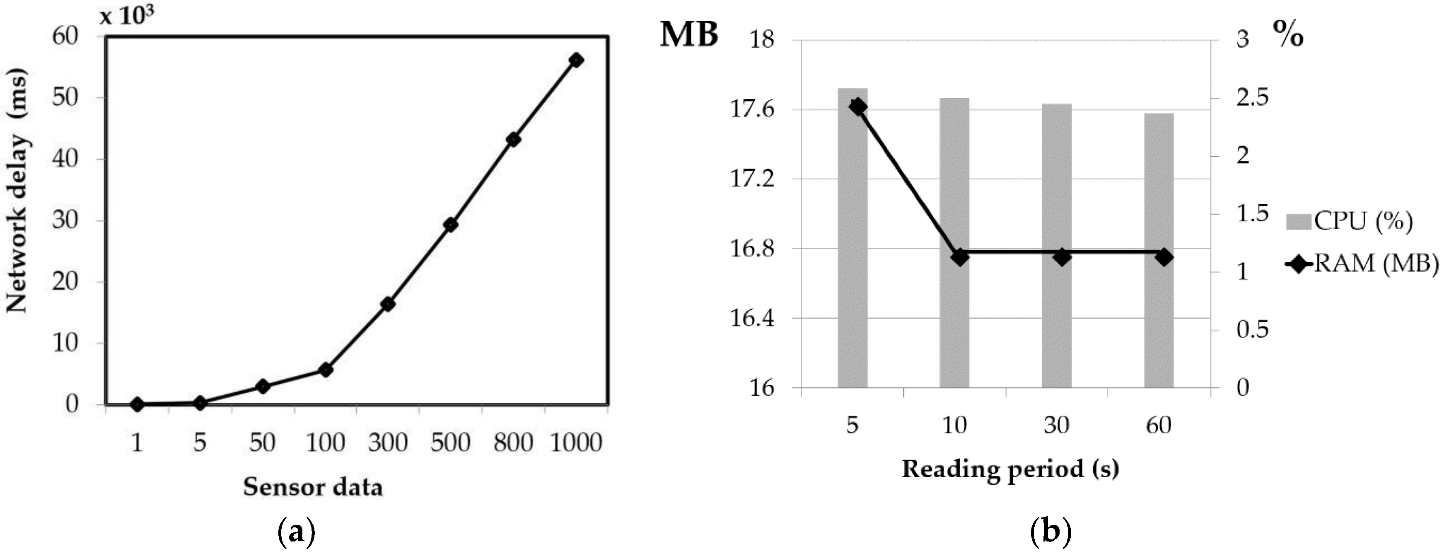
It was discovered through the literature review that the field of safety kits based on IoT is well-established. In order to highlight the benefits of our approach, we contrasted it with some existing approaches in terms of temperature, toxic gas detection, accuracy, mobility, and comfort. All of these approaches took the differentiation between similar events into account, despite the fact that their experiment protocols and datasets for detection vary. Our system, however, has a separate local server that gathers data. also has the ability to show it to the user on the desktop program. The safety gear is portable. A desktop application to display various information collected via sensors.

This project entails:

1) Detecting temperature, Humidity, Air quality, Carbon Monoxide gas, using a smart kit.

2) Transmitting the data from ESP32 to Node-RED(local server).

3) Storing and monitoring the data using our own desktop application.



**FIG. 6.4 GRAPHICAL REPRESENTATION OF PERFORMANCE OF SENSORS**

**CHAPTER 7**

**CONCLUSION**

**7.1 CONCLUSION**

We hope to gain hands-on experience with the trending technologies of "Embedded System" and "Internet of Things" through this project. IoT enabled industrial monitoring systems have become increasingly popular in a variety of industries because they improve safety standards by providing real-time monitoring of critical parameters such as temperature, humidity, and smoke, as well as alerting officials and workers regularly. The implementation is not only for safety reasons, but it also has the potential to increase industry yields. In our project, the Internet of Things (IoT) is used to collect data and communicate through the internet.

We hope that our project will be beneficial enough to be implemented in industries across India, saving lives and property from accidents and risks that are often overlooked by industry personnel and users. Companies in the industrial and logistics sectors can better meet the new era of instant needs by utilizing the Industrial Internet of Things (IoT). IoT technologies are used in manufacturing processes and across supply chains in the Industrial Internet of Things.

By using this system the administrator can use this smart kit to constantly monitor all of the workers engaged in the construction process. This would guarantee the safety of all construction employees on the site. Furthermore, we can gather information about the construction workers' health in order to provide them with instant assistance in the event of an emergency.

Thus, we can decrease the death rate of construction workers and improve their safety at work. This application would be extremely useful in hastening the rescue process, saving precious lives from suffocation and death. There are no products that combine our unique characteristics - to reduce the number of fatalities in the building industry - at the moment, so our product is unique.

**7.2 LIMITATIONS**

* Lack of internet connection will prohibit our data transfer.
* Frequent checks for sensors should be done as they may malfunction sometimes.
* Both the Node-Red server and the application should be running in the same network as we are running them locally and we have not deployed them in the cloud.
* Currently we have no analytical approach to find the abnormal conditions.

**7.3 FUTURE ENHANCEMENT**

Industrial IoT strategy should include machine learning and big data technology in addition to data from devices and sensors, harnessing the combination of existing sensor data, machine to machine (M2M) connectivity, and automation technologies to deliver greater insight back to the business. We are planning to make this project more cost-efficient and more reliable in the future. We are also planning on making the devices more effective by adding new features which makes it more faster and accessible by all the users. More new informations regarding the temperature parameters and various other details are planned to be implemeted in the future. Security features are planned to be made more robust and reliable to avoid system failure. New and various other methods for sending out alert and warnings to the user are planned to be added in the future. We are planning to add more IoT based technologies which would enhance the safety aspects in an industrial environments and also planning to make the system more flexible and easy-to-use by adding more options.

**REFERENCES**

1. Jamal Mabrouki , Mourade Azrour, Driss Dhiba, Yousef Farhaoui, and Souad El Hajjaji - IoT-based data logger for weather monitoring using arduino-based wireless sensor networks with remote graphical application and alerts - Published in: [Big Data Mining and Analytics](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=8254253) ( Volume: 4, [Issue: 1](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=9321194&punumber=8254253), March 2021) DOI: [10.26599/BDMA.2020.9020018](https://doi.org/10.26599/BDMA.2020.9020018)
2. LONG-PHUOC TÔN, LAM-SON LÊ,(Member, IEEE), AND MINH-SON NGUYEN - Micraspis: A Computer Aided Proposal Toward Programming and Architecting Smart IoT Wearables - Published in: [IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)(Volume:9)DOI: [10.1109/ACCESS.2021.3096749](https://doi.org/10.1109/ACCESS.2021.3096749)
3. ABDULL AH H. ALTOWAIJ RI, MOHAMM ED S. ALFAIFI, TARIQ A. ALSHAWI, (Member, IEEE), AHMED B. IBRAHIM, AND SALEH A. ALSHEBEI LI - A Privacy-Preserving Iot-Based Fire Detector - Published in: [IEEE Access](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639) ( Volume: 9) DOI: [10.1109/ACCESS.2021.3069588](https://doi.org/10.1109/ACCESS.2021.3069588)
4. Mangala Nandhini. V , Padma Priya G.V , Nandhini. S, Mr. K.Dinesh - IOT based Smart Helmet for Ensuring safety in Industries – Published in: 2018 IJERT - DOI : 10.17577/IJERTCONV6IS04068
5. [Risa Augusta Murti](https://ieeexplore.ieee.org/author/37089173772); [Annisa Firly Aprilia Putri](https://ieeexplore.ieee.org/author/37089175520); [Bayu Arivia Putra](https://ieeexplore.ieee.org/author/37089174015); [Salma Salsabilla](https://ieeexplore.ieee.org/author/37089165895); [Ayaturrahman Akrabullah](https://ieeexplore.ieee.org/author/37089173226) and Sujito - Android-Based Dam Management and Monitoring System: Wireless Communication using Google Firebase - Published in: [2021 7th International Conference on Electrical, Electronics and Information Engineering(ICEEIE)](https://ieeexplore.ieee.org/xpl/conhome/9616616/proceeding)DOI:[10.1109/ICEEIE52663.2021.9616787](https://doi.org/10.1109/ICEEIE52663.2021.9616787)
6. Kun Han and Xiangdong Zeng - Deep Learning-Based Workers Safety Helmet Wearing Detection on Construction Sites Using Multi-Scale Features - date of publication December 24, 2021, date of current version January 5, 2022. Digital Object Identifier 10.1109/ACCESS.2021.3138407 IEEE Access
7. Raghavendra Rao B, Karthik NS, NA Poojitha, Divya L and Nandini N - SIRASTRANA -A Smart Helmet for Air Quality and Hazardous Event Detection for the Mining Industry - International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 5, Issue 5, May 2018
8. Pulishetty Prasad , Dr. K. Hemachandran and H.Raghupathi - ZIGBEE BASED INTELLIGENT HELMET FOR COAL MINERS – International Journal of advanced research in [1] Science and engineering – Volume number 6, issue number 10, October

**APPENDICES**

**A.1 Code to collect data using Arduino Mega**

**Arduino Mega code**

#include <WiFi.h>

#include <PubSubClient.h>

#include <DHT.h>

WiFiClient wifiClient;

String data3;

#define DHTTYPE DHT11

#define DHTPIN 15

#define MQTPIN 34

DHT dht(DHTPIN, DHTTYPE);

#define ORG "2on79n"

#define DEVICE\_TYPE "ESP32"

#define DEVICE\_ID "Furnace"

#define TOKEN "Auth-Token"

#define speed 0.034

void callback(char\* topic, byte\* playload, unsigned int payloadLength);

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

char publishTopic[] = "iot-2/evt/Data/fmt/json";

char topic[] = "iot-2/cmd/test/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;

PubSubClient client(server, 1883, callback , wifiClient);

void publishData();

String command;

String data = "";

long duration;

float dist;

const int motorPin = 2;

void setup()

{

Serial.begin(115200);

dht.begin();

pinMode(motorPin, OUTPUT);

wifiConnect();

mqttConnect();

}

void loop() {

publishData();

delay(2000);

if (!client.loop()) {

mqttConnect();

}

}

void wifiConnect() {

Serial.print("Connecting to "); Serial.print("Wifi");

WiFi.begin("Wi-Fi SSID","password");

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.print("WiFi connected, IP address: "); Serial.println(WiFi.localIP());

}

void mqttConnect() {

if (!client.connected()) {

Serial.print("Reconnecting MQTT client to "); Serial.println(server);

while (!client.connect(clientId, authMethod, token)) {

Serial.print(".");

delay(500);

}

initManagedDevice();

Serial.println();

}

}

void initManagedDevice() {

if (client.subscribe(topic)) {

Serial.println("IBM subscribe to cmd OK");

} else {

Serial.println("subscribe to cmd FAILED");

}

}

void publishData()

{

int sensorValue = analogRead(MQTPIN); //MQT 135 connected to GPIO 34 (Analog ADC1\_CH6)

int humid = dht.readHumidity();

float temp = dht.readTemperature();

String payload = "{\"Humidity\":";

payload += humid;

payload += "}";

if (client.publish(publishTopic, (char\*) payload.c\_str())) {

Serial.println("Publish OK ");

Serial.println(payload);

}

delay(500);

payload = "{\"Temperature\":";

payload += temp;

payload += "}";

if (client.publish(publishTopic, (char\*) payload.c\_str())) {

Serial.print("Publish OK ");

Serial.println(payload);

}

}

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++) {

data3 += (char)payload[i];

}

}

**A.2 UWP desktop application code**

**MainPage.xaml**

using System;

using Windows.ApplicationModel.Core;

using Windows.Media.Core;

using Windows.Media.Playback;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Media.Animation;

// The Blank Page item template is documented at https://go.microsoft.com/fwlink/?LinkId=402352&clcid=0x409

namespace Final\_Year\_Project

{

public sealed partial class MainPage : Page

{

public MainPage()

{

this.InitializeComponent();

MediaPlayer mediaPlayer = new MediaPlayer();

mediaPlayer.Source = MediaSource.CreateFromUri(new Uri("ms-appx:///Assets/Doofenshmirtz.mp3"));

mediaPlayer.Play();

MyFrame.Navigate(typeof(HomePage), null, new SuppressNavigationTransitionInfo());

CoreApplication.GetCurrentView().TitleBar.ExtendViewIntoTitleBar = true;

Window.Current.SetTitleBar(AppTitleBar);

var tBar = CoreApplication.GetCurrentView().TitleBar;

tBar.LayoutMetricsChanged += OnTitleBarLayoutMetricsChanged;

PageNameTextBlock.Text = "Home";

}

public void OnTitleBarLayoutMetricsChanged(CoreApplicationViewTitleBar sender, object args)

{

var bar = sender as CoreApplicationViewTitleBar;

}

private void HomePageNavigationButton\_Click(object sender, RoutedEventArgs e)

{

MyFrame.Navigate(typeof(HomePage), null, new SuppressNavigationTransitionInfo());

PageNameTextBlock.Text = "Home";

}

private void FurnacePageNavigationButton\_Click(object sender, RoutedEventArgs e)

{

MyFrame.Navigate(typeof(FurnacePage), null, new SuppressNavigationTransitionInfo());

PageNameTextBlock.Text = "Furnace";

}

private void BoilerPageNavigationButton\_Click(object sender, RoutedEventArgs e)

{

MyFrame.Navigate(typeof(BoilerPage), null, new SuppressNavigationTransitionInfo());

PageNameTextBlock.Text = "Boiler";

}

private void FumeHoodPageNavigationButton\_Click(object sender, RoutedEventArgs e)

{

MyFrame.Navigate(typeof(FumeHoodPage), null, new SuppressNavigationTransitionInfo());

PageNameTextBlock.Text = "FumeHood";

}

}

}

**FumeHood.xaml**

using System;

using System.Threading;

using Windows.System;

using Windows.UI.Core;

using Windows.UI.Xaml.Controls;

namespace Final\_Year\_Project

{

public sealed partial class FumeHoodPage : Page

{

public FumeHoodPage()

{

var dispatcherQueue = DispatcherQueue.GetForCurrentThread();

this.InitializeComponent();

dispatcherQueue.CreateTimer();

Thread t = new Thread(new ThreadStart(GetCOConc));

t.Start();

Thread t1 = new Thread(new ThreadStart(GetAirQuality));

t1.Start();

}

private async void GetCOConc()

{

while (true)

{

Thread.Sleep(40);

var uri = new Uri("http://127.0.0.1:1880/data/furnace/irintensity");

using (var httpClient = new Windows.Web.Http.HttpClient())

{

try

{

string result = await httpClient.GetStringAsync(uri);

Dispatcher.RunAsync(CoreDispatcherPriority.Normal, () => SetValueCallback(result));

}

catch { }

}

}

}

private void SetValueCallback(string text)

{

try

{

double val = Convert.ToDouble(text);

this.MQ07Gauge.Value = val;

this.MQ07Value.Text = val.ToString();

}

catch{

}

}

private async void GetAirQuality()

{

while (true)

{

Thread.Sleep(40);

var uri = new Uri("http://127.0.0.1:1880/data/furnace/temperature");

using (var httpClient = new Windows.Web.Http.HttpClient())

{

try

{

string result = await httpClient.GetStringAsync(uri);

Dispatcher.RunAsync(CoreDispatcherPriority.Normal, () => SetTextCallback(result));

}

catch { }

}

}

}

private void SetTextCallback(string text)

{

try

{

double val = Convert.ToDouble(text);

this.MQ135Gauge.Value = val;

this.MQ135Value.Text = val.ToString();

}

catch

{

}}}