UNIVERSITY of RWANDA

COLLEGE OF SCIENCE AND TECHNOLOGY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

Design and Prototype a Coal Mine Monitoring System

Students

ASIFIWE Emmanuel 217062253

NYIRAMUNYURA Alice 217135552

TUYIZERE Vincent 217048684

January 2021

Undergraduate Final year Project

Bachelor of Science (Honors) in Electronics and Telecommunication Engineering

Supervisor: Dr. Louis Sibomana

Project ID: **EEE/ETE/2019-2021/06**

Dedication

We dedicate this project to:

Our almighty God

Our beloved parents

Our brothers and sisters

Our relatives, friends, and classmates

Approval and Certification

We ASIFIWE Emmanuel (217062253), NYIRAMUNYURA Alice (217135552), TUYIZERE Vincent (217048684) hereby declare that the report titled "**Design and Prototype of a Coal Mine Monitoring System"** has been carried out at the University of Rwanda, College of Science and Technology, School of Engineering, Department of Electrical and Electronics Engineering for the award of Bachelor's degree in Electronics and Telecommunication Engineering under the supervision of Dr. Louis SIBOMANA. We solemnly declare that to the best of our knowledge, is our original work and has never been presented or submitted here or elsewhere in a previous application for the award of any academic qualification. All sources have been duly acknowledged as a complete references.

1. ASIFIWE EMMANUEL 217062253 ....

2. NYIRAMUNYURA Alice 217135552...

3. TUYIZERE Vincent 217048684..

Supervisor: Dr. Louis SIBOMANA . .

Head of Department EEE: Dr. BAKUNZIBAKE Pierre......................................

Acknowledgment

We would like to thank God Almighty for his support, safety, and grace, for helping us to complete this project. We would like to extend our heartfelt appreciation to Dr. Louis SIBOMANA for his oversight, assistance, suggestions, encouragement, and guidance. Throughout this work, he has been a continual source of inspiration to us, we count ourselves incredibly lucky for having the ability to study and work under him over the entire period.

We would also like to use this opportunity to share our appreciation and heartfelt gratitude to our classmates for their support and motivation. Finally, our sincere gratitude to all our families for their support, commitment, and continuing encouragement in our efforts to do our job.

Thank you so much all of you, God bless you

Abstract

The welfare of miners is a serious problem today. The health and lives of miners are unsafe due to many important problems, including not just the working environment, but also its after-effects. Mining operations release dangerous and toxic gases that expose the associated employees to the danger of life. This puts the mining industry under a lot of pressure to ensure the health and wealth of the miners.

Incident conditions that occur in underground mines are becoming harmful to the miner's health such as the toxic gases, temperature, humidity, smoking, and collapses of the mining surface. All those conditions shouldn't be detected easily by the human's senses. The coal mine monitoring system by using a wireless sensor network which is composed of multiple sensors has been developed to monitor the surrounding parameters in the underground mine environment.

This system is designed to provide an early warning alarm and real-time monitoring inside the mine to save the lives of miners. The alarm triggered when the sensor's value crossing the threshold value. It also provides an online monitoring system by saving the values of data collected by sensors into the webserver and the saved data can be used for future use.

Table of Contents

List of figures

List of tables

Abbreviation

ASP: Active Server Page

API: Application Programming Interface

AHS: Asynchronous HTTP Service

CSS: Cascading Style Sheets

CMMS: Coal Mine Monitoring System

DB: Database

HTML: Hypertext Markup Language

HTTP: Hypertext Transfer Protocol

IDE: Integrated Development Environment

IoT: Internet of Thing

JS: JavaScript

LPG: Liquefied Petroleum Gas

PaS: platform as Service

PHP: Personal Home Page

SDI: Serial Digital Interface

SVG: Scalable Vector Graphics

URL: Uniform Resource Locator

XML: Extensible Markup Language

XHTML: Extensible Hypertext Markup Language

Chapter 1. INTRODUCTION

1.1 Background

Underground mining operations prove to be a risky activity as far as the safety and health of workers are concerned. These risks are due to different techniques used for extracting different minerals. The deeper the mine, the greater is the risk. These safety issues are of grave concern especially in the case of coal industries. Thus, the safety of workers should always be of major consideration in any form of mining, whether it is coal or any other minerals [1]. The safe production level of the coal mine is still low, disasters in the coal mine are due to the complexity of the mine environment and the variety of work carried out in coal mine, as such it is very necessary to monitor the working environment of the coal mine [2]. The underground mining industry comes to the category, where every parameter such as methane gas, rising temperature, humidity, fire accidents, and so on has to be monitored regularly [3]. It is very necessary to monitor the mine working environment to overcome those incident conditions and come up with a new solution to the challenges found in the mining site.

Rwanda Mines, Petroleum, and Gas Board developed Mining safety standards to provide the best practice guide for the management of health and safety in a particular underground mining environment and the open-pit mines and the standards developed to show that the monitoring will be done manually by specialists for providing a framework for the management of the hazards and associated risks that are inherent in the sector without real-time data collections [4]. The wireless sensor network can solve the key issues of the working area in real-time monitoring, synchronization monitoring, and so on.

We have designed and developed a prototype for Coal Mine Monitoring System to monitor unsafe conditions that should occur in the underground mine environment to improve the level of monitoring production safety and reduce accidents in the coal mine. This monitoring system is composed of different sensors to collect the rising temperature and humidity, the explosion of toxic gases, and smoke in the underground mine. Sensors are connected to ESP32 using the SDI-12 protocol to collect the various parameters in the environment of the mining surface. If the collected data goes above the safety threshold level creates an alarm to the miners. Coal Mine Monitoring System has been developed to save the received data from ESP32 to the webserver and show it in real-time on web app and analyze all kinds of information for decision function.

1.3 Aims of the project

1.3.1 Main objective

The main objective of our project is to design and prototype a Coal Mine Monitoring System in the underground coal mines for keeping the safety of miners.

1.3.2 The specific objectives

1. Collect the data by sensor network in underground mine then store them on the webserver for the future inspection and research.
2. Gather the data by sensors and use those data for monitoring and alerting the miners automatically inside the mine.
3. Develop a web application to monitor the coal mine extraction online wherever you are.

1.3.3 The scope of the project

In our project, we have designed and prototype the coal mine monitoring system and develop a web application that is used to store the data gathered by sensors. This work was achieved by using a wireless sensor network, buzzer, and ESP32 microcontroller which can be used as a Wi-Fi module to provide a web interface.

Chapter 2: LITERATURE REVIEW

In an underground mine, the monitoring systems are very essential systems that provide the way to monitor the miners for being healthy and wealthy. The incident condition which can occur in underground mine needs to be monitored such as rising in temperature and humidity, emitted gases, smoking achieved by using sensors, Wi-Fi module with microcontroller to provide the real-time monitoring to the miners. The new and developing communication system between sensors and microcontrollers utilized to monitor the miners more efficiently.

2.1. Previous work

Previously, researchers were working on different monitoring systems for the safety of the community and they have used different technologies to design this system. Here below, we have to highlight a number of those previous works.

Sumit Kumar Srivastava [1] proposed a design of a real-time monitoring system for mine safety using a wireless sensor network (Multi-Gas Detector). This work mostly was dealing with detecting the gas explosion, temperature, and humidity that occur in underground mining for the safety of the miners. This monitoring system did not consider storing the data gathered by sensors that can be used for future inspection and research.

Tan et al [5] designed the system called WSN based on mine safety system to the mining environment. It is the system providing the pre-warning for the explosion of gas or fire and proficient in real-time monitoring to the environment. This was not considered to measure the rising temperature and humidity that occur in an underground mine and to store the data from sensors for future inspection and research.

Mo Li, Yunhao Liu [6] proposed a system of underground structure monitoring with wireless sensor networks. Where the information should be gathered by the sensor network. And such information send to the monitor on the ground surface monitoring system to provide the alarm for the safety of the miners. This structure monitoring system did not develop the storage of the data gathered by sensors which can be essential in future inspection and research and online monitoring wherever you are around the world.

Gao Dan1, Li Weiwei1, and Dai Kun [7] proposed a design of a coal mine intelligent monitoring system. Focusing on two key issues of intelligent control and ZigBee wireless sensor networks were studied. Coal mine intelligent monitoring system can be used to monitor the concentration of carbon monoxide, carbon dioxide concentration, oxygen concentration, wind speed, air pressure, smoke, temperature, etc. this was not considering an online monitoring and the future inspection and research where the data gathered by sensors can be stored on the webserver and be accessed wherever you are.

2.2. Comprehensive parameters to be monitored

2.2.1 Temperature in Underground.

There are many large underground mines, defined as such based on production, horizontal extent, and depth. However, deep mining has led to an increase in the thermal hazard, which affects the production of conventional mine, poses potential risks to the health and wellbeing of miners, and threatens mining efficiency[8].

Thermal pollution in mines varies regularly due to seasonal changes in surface air temperatures. Working areas of mines experience thermal pollution when airflow temperatures exceed specified safety values, particularly in the summer. Airflow temperature is a critical parameter for assessing the degree of thermal pollution. Seasonal variation in surface air temperature has a considerable effect on the temperature field of surrounding rocks in the mine roadway, and also on airflow temperature[8]. Therefore, studying the influence of seasonal variations in ground air temperature on airflow and surrounding rock temperatures will help to evaluate, prevent, and control thermal pollution in underground mines. The present study combined measured temperatures and temperature measurements obtained from coal mine monitoring system web application stored to be used for study or researches.

2.2.2 Gases inside underground.

The air we breathe on the surface is a mixture of several gases including oxygen, nitrogen, argon, carbon dioxide, and other gases in trace amounts. We breathe easiest with 21% oxygen present in the air[9]. When other gases contaminant the air, the oxygen levels drop, and that is when the trouble begins. The air in mines can be contaminated by the presence of other gases such as carbon monoxide, hydrogen sulfide, methane, and excess carbon dioxide[9]. Due to being in a confined space these gases are not always able to disperse and can therefore build up in the mine, and due to their combustible, explosive, or toxic qualities, this is a serious issue. Rather than one particular gas, they are a toxic or explosive mixture of different gases that have a varying effect on human health and miner's safety[9]. These damps are produced or released during mining operations including drilling and blasting, by mining machineries such as diesel and gasoline motors, and by other means such as the decay of timbers, the aftereffects of mine fires, and chemical processes like oxidation[9].

The most efficient way of preventing these gases in mines is the incorporation of high-quality mining ventilation systems as well as the use of early detection devices as a reason why coal mine monitoring system was built to solve the issues like this. The most dangerous gases in mining are CH_4(methane), CO (Carbon Monoxide), CO_2 (Carbon Dioxide) and, H_2S (Hydrogen Sulfide). The best prevention is to provide high-quality mining ventilation systems and a wide range of detection devices including Gas monitors, Air samples, and chemical analysis[9].

2.3. Innovation of our system

By comparing to the previous work in [1, 5, 6, 7], our project is also the design of a coal mine monitoring system, but we have added the web application to store the data on the webserver to alert and warn alarm the miners in an underground mine and online monitoring wherever you are around the world to improve the safety to the miners in the mining industry in real-time. This system is containing ESP32 Node MCU, sensors, and buzzer.

ESP32 is a highly integrated solution for Wi-Fi-and-Bluetooth Internet of Thing (IoT) applications, with around 20 external components. ESP32 integrates an antenna switch, power amplifier, low-noise receive amplifier, filters, and power management modules [10]. This Wi-Fi module helped to send the data on the webserver and being used as a microcontroller.

The buzzer is a simple indicator that will be used as an alarm for providing the audible sound to alert the miners of any changes situation occur.

Sensors are used to maintain the data from the environmental changes like sensing the temperature and humidity, the methane gas, and smoking all those conditions should be sensed by sensors because they cannot be monitored by human sense.

2.4. Wireless sensor network

A Wireless sensor network can be defined as a network of devices that can communicate the information gathered from a monitored field through wireless links. The data is forwarded through multiple nodes, and with a gateway, the data is connected to other networks like wireless Ethernet. WSN is a wireless network that consists of base stations and numbers of nodes (wireless sensors). These networks are used to monitor physical or environmental conditions like sound, pressure, temperature, humidity and co-operatively pass data through the network to the main location [11].

2.4.1. Gas sensor (MQ-2 sensor)

MQ2 gas sensor is an electronic sensor used for sensing the concentration of gases in the air such as LPG, propane, methane, hydrogen, alcohol, smoke, and carbon monoxide. MQ2 gas sensor is also known as chemiresistor. It contains a sensing material whose resistance changes when it comes in contact with the gas. This change in the value of resistance is used for the detection of gas [12]. Besides, there are several merits caused by the emitted gas which could be very dangerous to the people work in underground mine such as:

Methane gas causes headaches, reduces the oxygen level in the physical structure. If the oxygen level reduces to less than 12%, the individual can get to be unconscious and turn out to be dead in some cases.

These gas symptoms are Nausea and vomiting, heart palpitations (which causes a painful sensation of the heart beating), memory loss, poor judgment, dizziness, and blurred vision [1].

Figure 1: MQ-2 Sensor

Using an MQ-2 sensor detects a gas very easily. You can either use the digital pin or the analog pin to accomplish this. Simply power the module with 5V and you should notice the power LED on the module glow and when no gas is detected the output LED will remain turned o meaning the digital output pin will be 0V [13]. MQ-2 Sensor has four pins where 1 pin is for VCC, 2 pin is ground, 3 pin is a digital output pin and 4 pins is an analog output pin.

Parameter name	**Technical condition**
Circuit voltage	5V±0.1
Heating voltage	5V±0.1
Load resistance	Can adjust
Heater resistance	33Ω±5%
Heating consumption	Less than 800mw
Using temperature	-20℃-50℃
Storage temperature	-20℃-70℃
Related humidity	less than 95%Rh
Oxygen concentration	21%(standard condition)Oxygen concentration can affect sensitivity
Sensing resistance	3KΩ-30KΩ (1000ppm iso-butane)
Standard detecting condition	Temp: 20℃±2℃ Humidity: 65%±5%

Table 1: Technical Specification of MQ-2 Sensor

2.4.2. Temperature and humidity sensor (DHT11 Sensor)

DHT11 is a Humidity and Temperature Sensor, which generates calibrated digital output. DHT11 can be interface with any microcontroller like Arduino, Raspberry Pi, etc., and get instantaneous results.DHT11 is low-cost, long-term stability, relative humidity, and temperature measurement, fast response, long-distance signal transmissions like in 20 meters, anti-interference ability, digital signal output, precise calibration, and high reliability. There are the two types of DHT sensor which are both used to measure the humidity and temperature such as DHT11 and DHT22. However, they have a little bit different in characteristics but mostly they are the same [14]. In our project, we have picked out DHT11 with the ultra-low-cost than DHT22 which will sense the temperature and humidity in underground mine spaces.

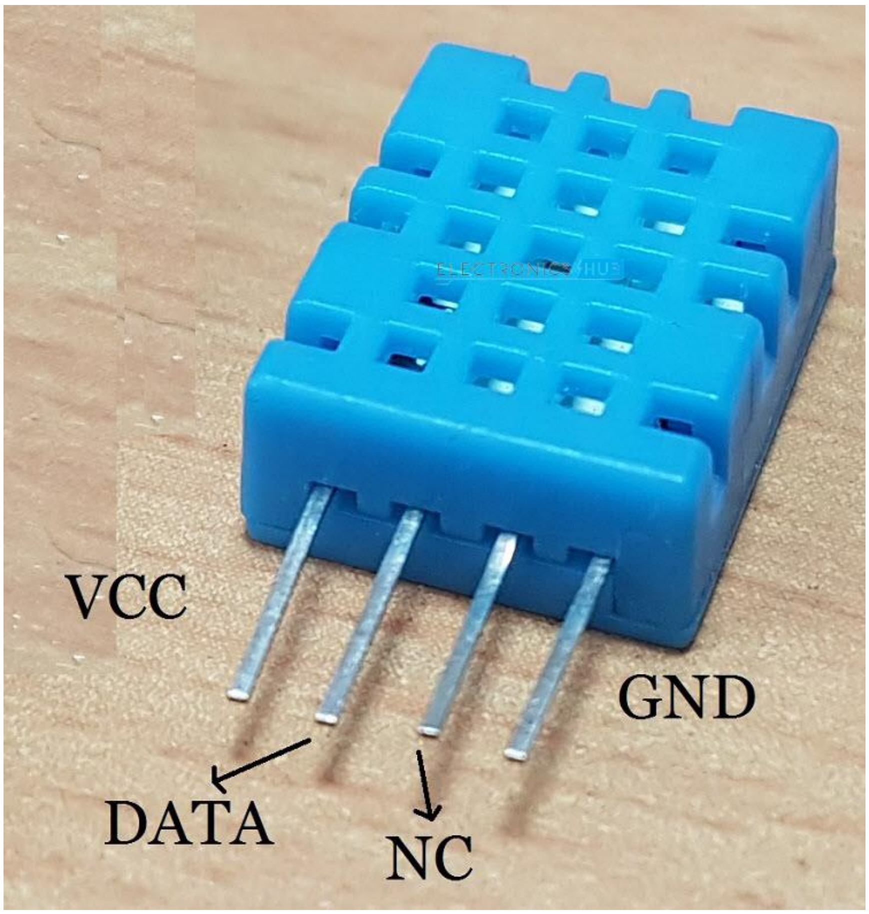

Figure 2: DHT11 Sensor

Relative Humidity	**Technical condition**
Repeatability	±1%RH
Accuracy	25℃ ±5%RH
Interchangeability	Fully interchangeable
Response time	1/e (63%) 25℃ 6s 1m/s Air 6s
Hysteresis	<±0.3% RH
Temperature	
Resolution	16Bit
Repeatability	±1℃
Accuracy	25℃ ±2℃
Response time	1/e (63%) 10S
Electrical Characteristics	
Power supply	DC 3.3～5.5V
Supply current	Measure 0.3mA Standby 60µA
Sampling period	Secondary Greater than 2 seconds

Table 2: Description of DHT11 Sensor

DHT11 has four pins one is connected to VCC, the second one is for data, the third is not connected and the fourth one is for grounding, negative power.

The DHT11 Humidity and Temperature Sensor consists of 3 main elements. A resistive type humidity sensor, an NTC (negative temperature coefficient) thermistor (to measure the temperature), and an 8-bit microcontroller, which converts the analog signals from both the sensors and sends out a single digital signal. Any microcontroller will read this digital signal for additional analyses [15].

Due to the low cost, high reliability, fast response, and being used at long-distance signal transmission, it is mostly used by many manufactures and in various application such as measuring humidity and temperature values of heats, air conditioning system and being used by weather stations to predict the weather conditions.

2.5. BUZZER

A buzzer is an audio signaling system, which may be mechanical, electromechanical, and piezoelectric. Buzzer and common uses beepers include systems for alarms, times, and a confirmation of user input.

By simply powering it using a DC power supply varying from 4V to 9V, this buzzer can be used. It is also possible to use a simple 9V battery but a controlled +5V or +6V DC supply is preferred. The buzzer is usually connected to a switching circuit to turn the buzzer on or off at the appropriate time and involves an internal.

2.6 Resistor

The resistor is a passive electrical component used in the electrical current flow to produce resistance. They can be present in virtually all electrical networks and electronic circuits. The calculation of resistance is in unit ohms [16].

Resistors are utilized by several applications. A few examples include electric current delimitation, voltage division, heat generation, circuit matching and loading, control gain, and constant time fixing. They are commercially available with resistance values over a range of more than nine orders of magnitude. They can be used as electric brakes to dissipate kinetic energy from trains or be smaller than a square millimeter for electronics.

2.7 Wi-Fi Module ESP32

ESP32 is a single 2.4 GHz Wi-Fi-and-Bluetooth combo chip designed with the TSMC ultra-low-power 40 nm technology. It is designed to achieve the best power and RF performance, showing robustness, versatility, and reliability in a wide variety of applications and power scenarios [10].

It is designed for mobile, wearable electronics, and Internet-of-Things (IoT) applications. It features all the state-of-the-art characteristics of low-power chips, including fine-grained clock gating, multiple power modes, and dynamic power scaling. The output of the power amplifier is also adjustable, thus contributing to an optimal trade-off between communication range, data rate, and power consumption [17].

Figure 3: ESP32 WI-FI Module

Thus, we have chosen this device to be used in our monitoring system as a microcontroller where the different sensors will be connected on it so that the data taken by this controller will be sent and stored on the webserver (database) we have created and even this device will also work as a microcontroller and a Wi-Fi module in the underground mine.

2.8 Database (Mongo DB)

Mongo DB Atlas is a fully managed cloud database developed by the same people that build Mongo DB. Atlas handles all the complexity of deploying, managing, and healing your deployments on the cloud service provider of your choice (AWS, Azure, and GCP). Mongo DB Realm provides a first-class service interface for Mongo DB Atlas that lets you securely one or more Atlas data sources. You can use standard Mongo DB query language syntax to access your data directly in your client application code or from a Realm Function. The Mongo DB service secures your data with a dynamic, role-based Rules engine that proxies and modifies incoming queries based on rules that you define. There are three types of Mongo DB collection rules: roles, filters, and Realm Schema [18].

2.9 Heroku

Great apps come from inspired and productive developers. The right tools and services will increase your development pace and help you bypass mundane tasks, remove friction, and simplify or automate processes. Not only can you get the job done and iterate quickly, but you're freer to let the inspiration flow and produce your best work. Heroku is a cloud-based, platform-as-a-service (PaaS) based on a managed container system for building, running, and managing modern apps. Heroku's platform, tools, integrated services, and ecosystem are meticulously designed to support the best possible developer experience. That's why Heroku has become a favorite app platform for hundreds of thousands of developers. Heroku gives you a set of powerful capabilities that deliver higher-order value. The Heroku platform is fully managed, meaning that we take care of servers. Hardware, and infrastructure. So you can stay focused on your app. The platform's flexibility allows you to build apps using your preferred language or framework and using popular architectural patterns, such as microservices. Deploying apps on Heroku is fast and streamlined. With built-in workflows that support your team's continuous integration and continuous delivery practices. This is why it is our choice to put our app here to be accessed [19].

2.10 Web application.

Web applications is a computer program that utilizes web browsers and web technology to perform tasks over the internet. Web applications use a combination of server-side scripts (PHP, Node js, and ASP) to handle the storage and retrieval of the information, and client-side scripts (JavaScript and HTML) to present information to users. This allows users to interact with the company using online forms, content management systems, shopping carts, and more [20].

The web application requires a web server to manage requests from the client, an application server to perform the tasks requested, and then a database store information.

Here's what a typical web application flow looks like:

- The user triggers a request to the web server over the internet, either through a web browser or the application's user interface.
- The web server forwards this request to the appropriate web application server.
- Web application server performs the requested tasks- such as querying the database or processing the data – then generates the results of the requested data.
- Web application server sends results to the webserver with the requested information or processed data.
- The web server responds to the client with the requested information that then appears on the user's display.

2.11 Power system

Electrical and Electronic devices require electrical power to work properly. The process of providing power to the system that will work real-time monitoring requires an appropriate electrical power supply for its normal working. This will require a controlled power supply with the ability to store power at the appropriate amount of time.

- For proper working of CMMS, it is required to use portable batteries. In the underground, more than one battery will be the perfect option for providing power to the miners.
- Proper electrical power supply for site operation devices, the installation of electrical power from power distribution nearby will be the best options for proper functions.

In the case of our project, we have used a battery of 5v and we may use a 5v adapter for powering CMMS. The following figure 6 shows the adapter and battery to be used.

Figure 4: Battery and Adapter of Powering ESP32

Chapter 3: METHODOLOGY

3.1 Introduction

This chapter discusses the methodology used while conducting this work. The data collection method such as documentation and internet has led us to gather and identify requirements for the best of designing and prototyping coal mine monitoring system as it concern mining safety.

3.2 Methods

By seeking more understanding to conduct our project we have used books, websites, and previous work related to this project. To build and implement the proposed project, the required software, development kits, development frameworks, modules, programming languages, and libraries are discussed in subsection

3.2.1 Documentation

In this project books, websites, journal, and articles helped in knowing the structure of mining environment especially different parameters found in mining which are hazardous to human health. It helped to know more about already implemented Systems to improve the health of miners inside, the challenges, and required improvements. It also helped to know the difference between those implemented systems and the coal mine monitoring system. We also used some published papers about mining monitoring systems to refer to for literature review and design system.

3.3 System Development

3.3.1. Underground Coal Mine Monitoring system

Required software, development kits, programming languages, and libraries that are required to develop and test the proposed project are explained in the following.

3.3.1.1 ESP32 Microcontroller

Documentation of ESP32 Microcontroller found in [x] allowed us to choose this microcontroller over other microcontrollers like Arduino board, Pocket Beagle, Raspberry Pi, and so on[17][10]. because of its ability of multifunction that meets the need of underground coal mine monitoring system which is to collects measured data from sensors, prepares collected measurements and then sends them to the web application, and in case of emergency, to provide alarming. this is the module we have used as a microcontroller and Wi-Fi module to communicate remotely with our webserver.

C and C++ languages have been used to program ESP32 Microcontroller to configure inputs as sensors, manipulation of data, and the functions of sending data with help of libraries like HTTP Client, Adafruit sensor, WIFI, DHT, and MQ2 using Arduino IDE.

3.3.1.2 Arduino IDE

it is the main text editing program we have used for Arduino programming before uploading it to the microcontroller.

3.3.1.3 Wireless sensor network

It can be defined as a network of devices that can communicate information collected through wireless connections from a monitored field.

3.3.1.4 Temperature and Humidity Sensor

Documentation of DHT11 sensor which states that is a Humidity and Temperature sensor, which generates calibrated digital output and it can be easily interfaced with any microcontroller[15], it is a good option for coal mine monitoring system to provide measurements of temperature and humidity as they are parameters required to be monitored to provide safety of miners. It is a low-cost digital temperature and humidity sensor. It measures the surrounding air and sends measured data to ESP32.

3.3.1.5 Gas concentration sensor

Documentation of MQ2 gas states that it is an electronic sensor used for sensing the concentration of gases in the air such as LPG, propane, methane, hydrogen, alcohol, smoke, and carbon monoxide[12]. Some of the parameters mentioned will be highly monitored by the coal mine monitoring system. We have used this sensor for detecting LPG, CO, and Smoke then sends them to ESP32. With high sensitivity and fast response time, the measurement can be taken as soon as possible.

3.3.1.6 Alarming devices

Audio signaling is the best option for alarming people in the mining site. Documentation of Buzzer provides information of buzzer as an audio signaling system, which may be mechanical, electromechanical, and piezoelectric[21]. Buzzer and common uses beepers include systems for alarms, times, and a confirmation of user input. The active buzzer has been used to generate a sound and getting a command from ESP32.

3.3.2. Web app Coal Mine Monitoring system

With help of HTML, CSS, and JavaScript, we were able to build a web application that meets the requirements of a coal mine monitoring system. This app can get data from different mining sites, it was built to allow real-time monitoring of different mining sites, and then store data for future use in different sectors.

Documentation of HTML state that it is the most basic building block of the web. It defines the meaning and structure of web content. HTML uses “markup” to annotate text, images, and other content for display in a web browser [20].

Documentation of CSS states that CSS describes how elements should be rendered on screen, on paper, in speech, or on other media[22].

Documentation of JavaScript states that JavaScript (JS) is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive [23].

Mongo DB is a document-oriented No SQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, Mongo DB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in Mongo DB [24].

Node js is an open-source, cross-platform, back-end JavaScript runtime environment that executes JavaScript code outside a web browser [25].

The combination of HTML, CSS, JavaScript, Node js, Mongo DB, to create CMMS web application explanation is demonstrated in chapter 4, subsection 4.4.2 which is CMMS web application design.

3.4 Expected result

The outcome of this project is a prototype of the coal mine monitoring system which can be a benefit to the safety and health of miners in the underground mine.

This monitoring system can be implemented in the mining industry to improve the safety of miners working in underground mine activity.

3.5. Limitations of the study

This project research has a limitation on the following guidelines:

1. Filter design in case of noise in measuring process of parameters inside the coal mine environment.
2. Power amplifier design to increase the amplitude of signal measured by sensors in case the signal is too low to be processed by microcontroller.
3. This project is limited to prototyping with manipulated coal mining site for simulation of functionality.

CHAPTER 4: DESIGN AND PROTOTYPING OF COAL MINE MONITORING SYSTEM

This chapter provides an overview of the design and prototyping process for the project.

4.1. CMMS planning phases

The development of CMMS has different project planning phases. Figure 6 shows all of the phases in detail and the output results of each phase. This process aims to ensure that the necessary information is generated in the atmosphere of the mining environment site.

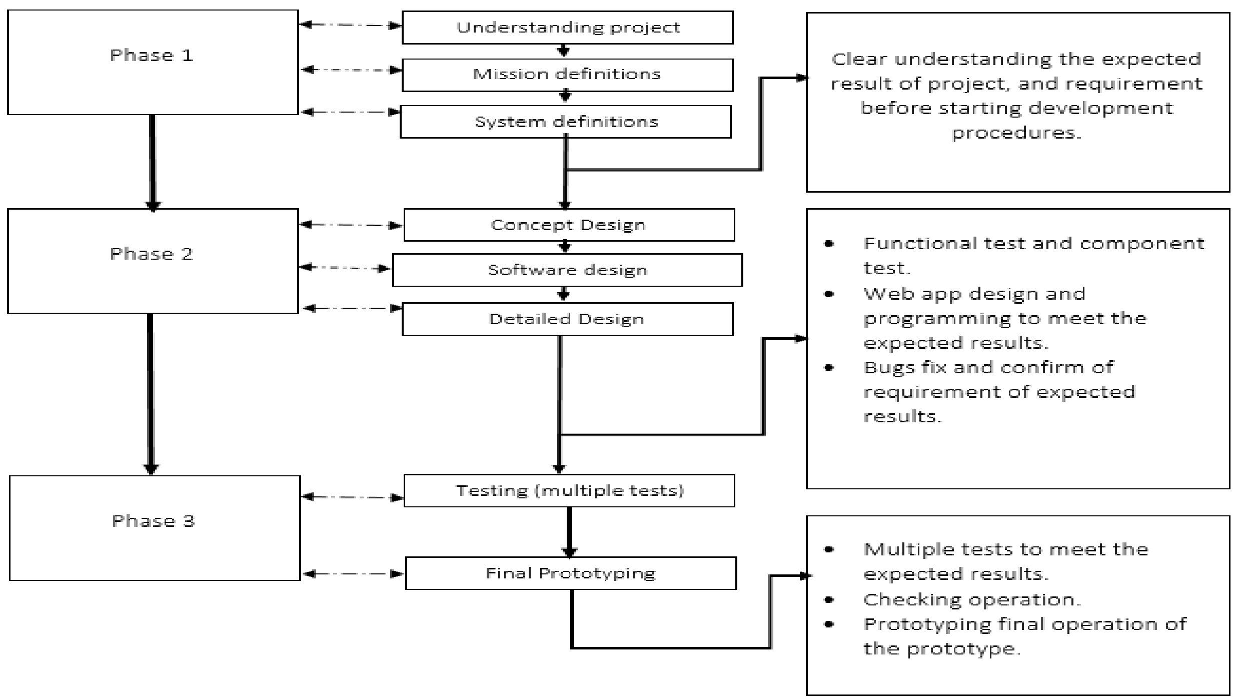

Figure 5: CMMS Planning phase

All phases were processed to develop the prototype of CMMS. Identifying the complexity of mining infrastructure and welfare of miners to determine the environment to experiment with the CMMS, identification of required tools to move to the next phase of design. Phase two describes the components used to design CMMS to build a web application, integration of sensors, microcontroller, and web application to test the functionality of components including experiment environment. Finally, Phase Three is to confirm the operation of the developed prototype and do many multiple tests to meet the expected results and compare its result to the main objective of CMMS.

4.2. CMMS block diagram

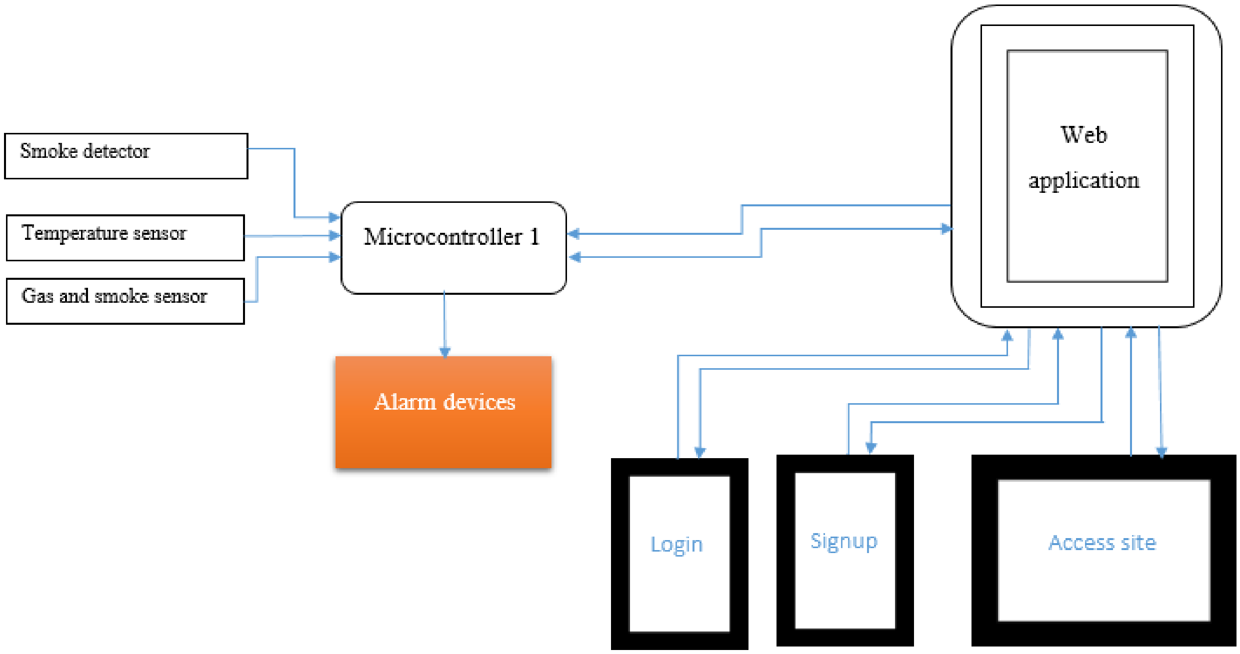

Figure 6: CMMS Block Diagram

CMMS consists of a web application and ESP32 to collect rising temperature and humidity from the DHT11 sensor, the explosion of toxic gases and smoke from the MQ-2 Gas sensor in the underground mine using SDI-12 protocol. ESP32 sends collected data to be displayed on the web application in real-time. The web application receives data and saves them into the Mongo DB database as a persistent data storage. It also allows nearby users to access the data and alarming functionality using a computer or phone by providing the IP address of ESP32. Based on functionality and specifications, subsections 4.3, 4.4, and 4.5 will explain the usage of each component to fulfill the objective of safety to the miners.

4.3. Hardware Implementation.

To test the designed real-time CMMS using a wireless sensor network, the normal environment has been used to test our system and check how it can work while we need to implement it. We have used this environment because accessing the mining site was not possible due to budget and short time. We designed the complete Prototype system on the breadboard which is presented in Figure 7.

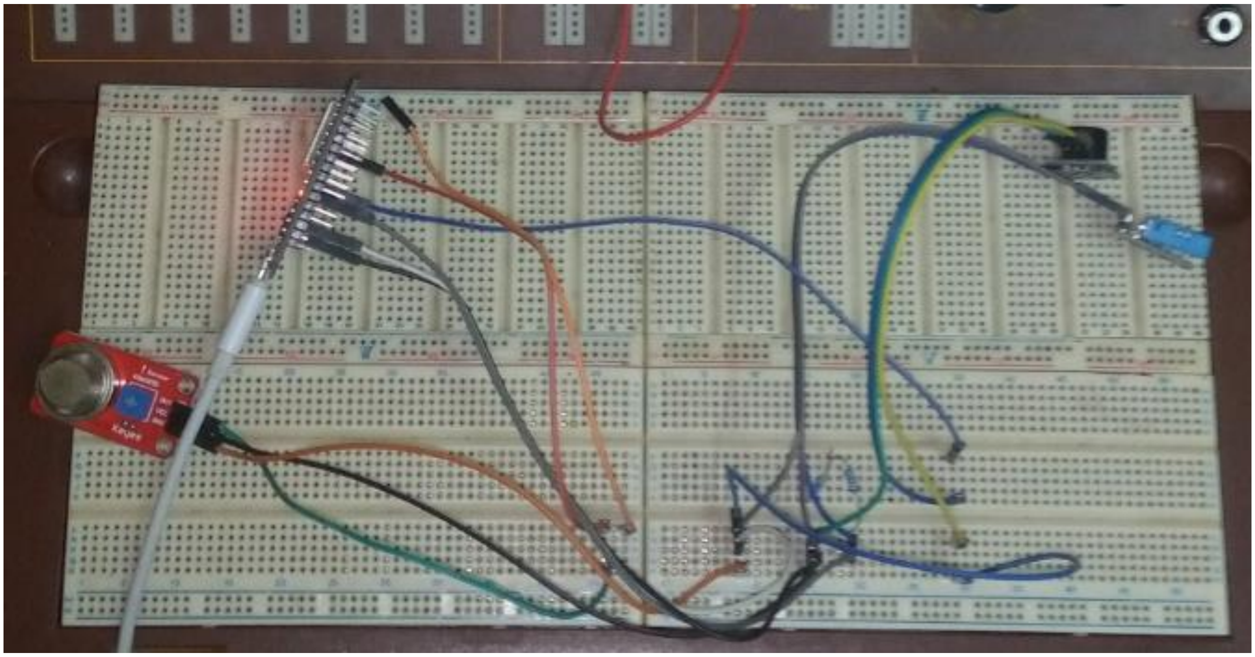

Figure 7: Hardware Implementation

The system on above Figure 7 consists of the following components

1. Microcontroller – ESP32: collects measured data from sensors, and sends them to the web application, and provides alarming.
2. Temperature and Humidity Sensor – DHT11: low-cost digital temperature and humidity sensor. It measures the surrounding air and sends measured data to ESP32.
3. Gas concentration sensor – MQ-2: used for detecting LPG, CO, and Smoke then sends them to ESP32. With high sensitivity and fast response time, the measurement can be taken as soon as possible.
4. Alarming device – Buzzer: audio signaling device which may be mechanical, electromechanical, or piezoelectric. The active buzzer has been used to provide a beeper and getting a command from ESP32.

Further, Figure 8 shows physical connections between components used to build the prototype using Fritzing version 0.9.3 design software. The schematic diagram shows all sensors such as MQ-2 Gas sensors, DHT11 sensor (temperature and humidity sensor) connected to ESP32, and the buzzer physically connected on a breadboard.

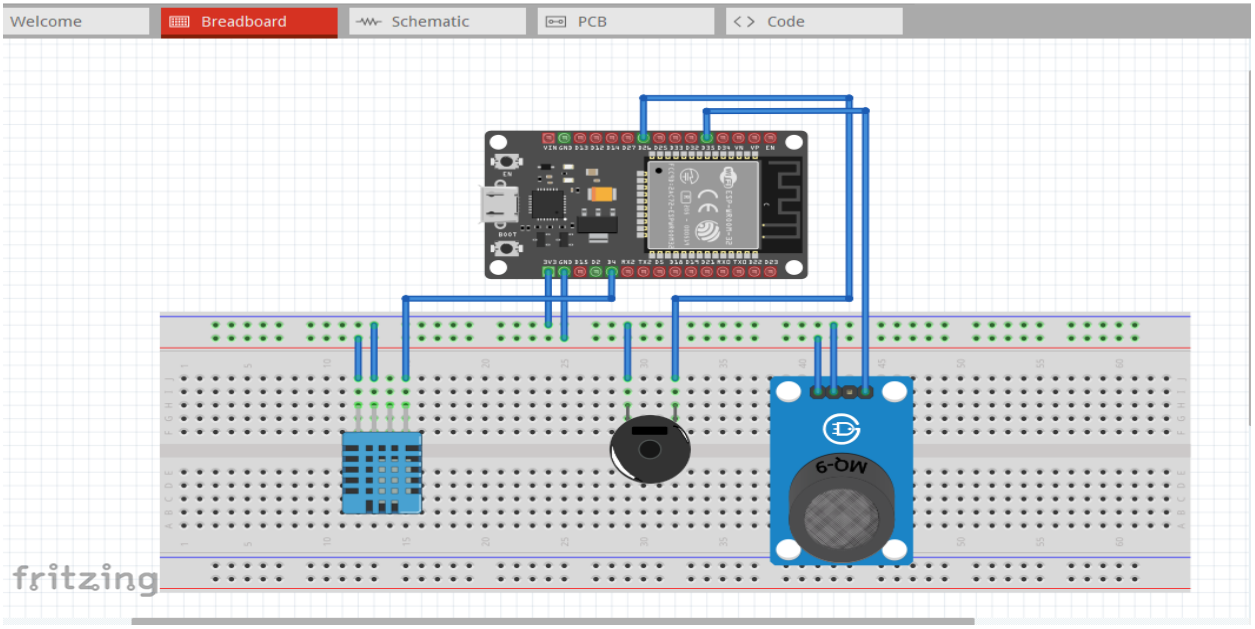

Figure 8: Schematic diagram of CMMS

4.4 Software Implementation**.**

4.4.1 ESP32 coding part

ESP32 is Arduino microcontroller-based board. So for programming the ESP32 board, Arduino IDE 1.8.13 (Integrated Development Environment) software is used which supports C and C++ programming languages. This software makes it easy to write code.

The CMMS prototype was tested in a normal environment. The data collected are different from the data that can be collected from a real mining environment. The collected data are automatically sent to the webserver to be stored for future inspection using HTTP GET request with help of the ability of ESP32 to process this request. To process the HTTP GET request, ESP32 requires a Wi-Fi connection and an API URL (Uniform Resource Locator**)** path to make this request. The necessary libraries, configuration, and manipulation of data used in coding to accomplished the task of collecting data and send them to a web application can be seen in (Appendix A).

Figure 9 shows the different sensors' value, Wi-Fi connection, and their corresponding HTTP GET request in Serial monitor using Arduino IDE 1.8.13 software. During design and experiment, we displayed the collected data measured by sensors to make sure they're accurate and helpful to the mine environment to decide in real-time. The data sent to the database, have been also displayed on the Serial monitor to be sure that the response is successful or failed.

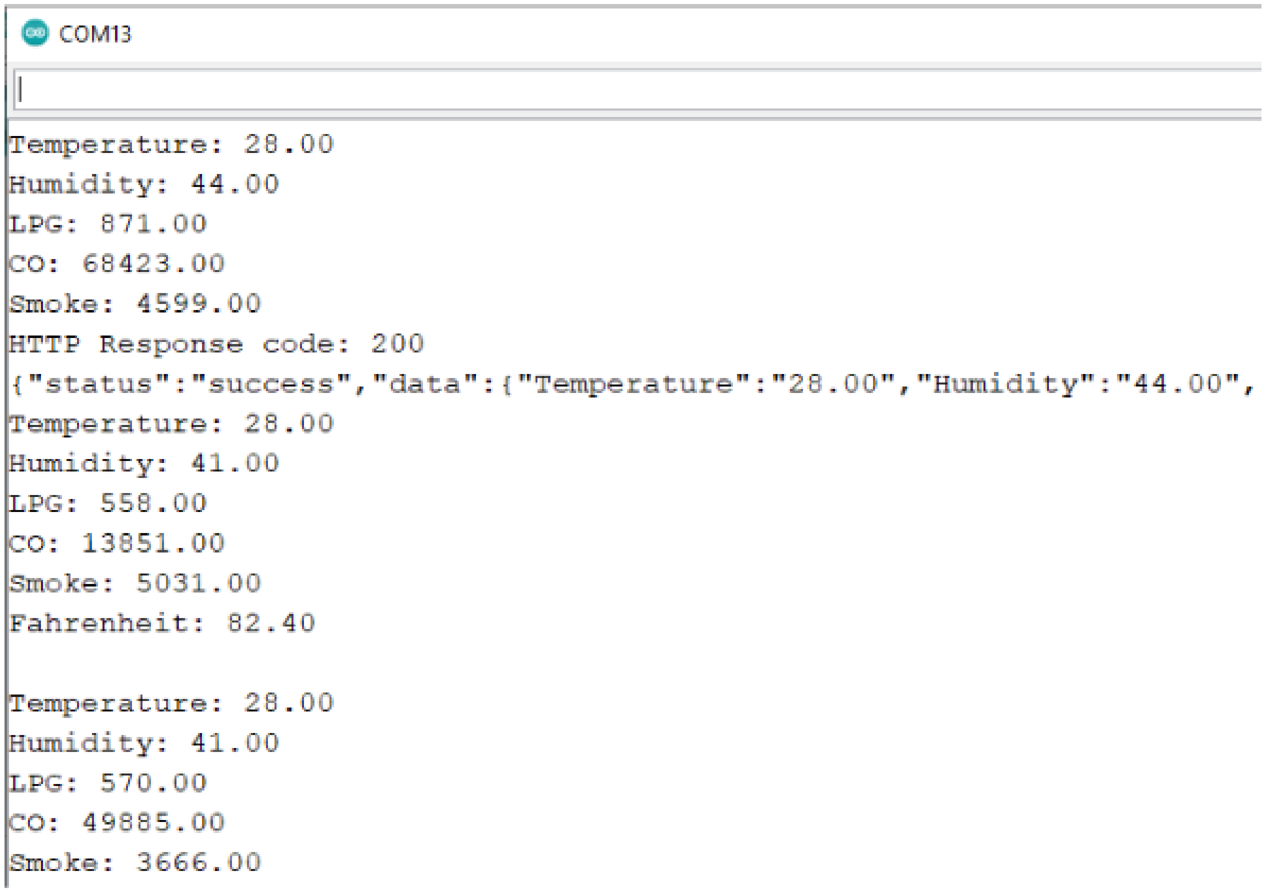

Figure 9: Parameter's value in Arduino

4.4.2 CMMS web application design

CMMS web app was designed using HTML, CSS, and JavaScript on the front-end, and Node js was used on the server-side design.

Node js is an open-source, cross-platform, back-end JavaScript runtime environment that executes JavaScript code outside a web browser [25].

HTML (Hypertext Markup Language) is the most basic building block of the web. It defines the meaning and structure of web content. HTML uses "markup" to annotate text, images, and other content for display in a web browser [20].

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML, or XHTML).

CSS describes how elements should be rendered on screen, on paper, in speech, or on other media [22]. We have used CSS to describe how elements should look on screen by adding style to multiple pages.

JavaScript (JS) is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive [23].

Mongo DB is a document-oriented No SQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, Mongo DB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in Mongo DB [24]. CMMS comprises of Mongo DB database to store collected data for online monitoring and future use. CMMS web application has been deployed on the Heroku platform. Heroku is a container-based cloud platform as Service (PaS). Developers use Heroku to deploy, manage, and scale modern apps. This platform is easy to use, offering developers the simplest path to getting their apps to market [26].

We have combined HTML, CSS, JavaScript, and Node js to create a CMMS web app using codes that can be accessed in (Appendix B). Figure 10 shows the CMMS web application created that can be accessed through https://cmms-app.herokuapp.com/. To access data on the CMMS web application requires a user to first be authenticated (log in or signup) before accessing the dashboard of collected data. All of this design of the web app was done by our group members to provide the best and easy way to access the collected data.

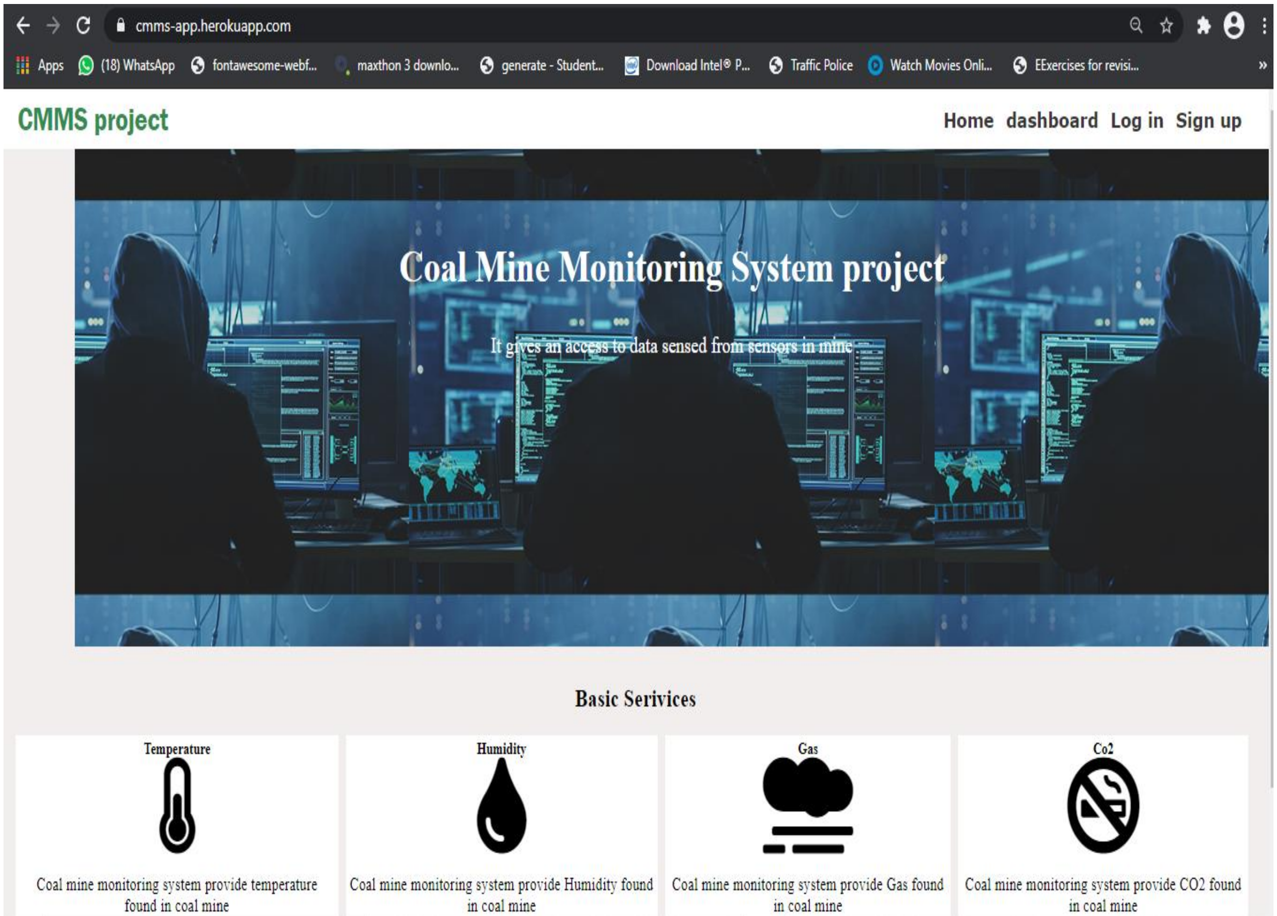

Figure 10: CMMS web application

4.4.3 CMMS web Server.

We built a web server that updates the readings automatically without the need to refresh the web page. We have used HTTP client, Wi-Fi, and ESP Async WebServer libraries that provide an easy way to build an asynchronous web server. The Asynchronous HTTP Server (AHS) is an HTTP server capable of handling long-lived asynchronous XML HTTP Requests in a scalable fashion [27]. This server provides the advantages of handling more than one connection at the same time. When you send the response, you are immediately ready to handle other connections while the server is taking care of sending the response in the background. Figure 11 shows the created Asynchronous web server accessed using the phone. We created this webserver for online monitoring and alarming in case of failure of sensors or other unpredicted circumstances that are not being measured by the system. The team leader or one in control of online monitoring will use this system ON button to alarm the miners in case of emergence. Through the configuration of an Asynchronous web server. We have provided an IP address that the user can utilize to access the data and alarming functionality, the IP is 192.168.1.109.

Figure 11: CMMS web server accessed by using the phone

4.5 Results and Discussion

Following the combination of work done in subsection 4.2, and subsection 4.3, different mine gases, temperature, and humidity were collected successfully, and all of those data are stored in the database each time of collection to be visualized on the dashboard and to be analyzed for future inspection of coal mine safety. Figure 12 shows a sample of data stored in the MongoDB database. CMMS stored data are provided in real-time and historical to support decision making and provision of availability of mining sites.

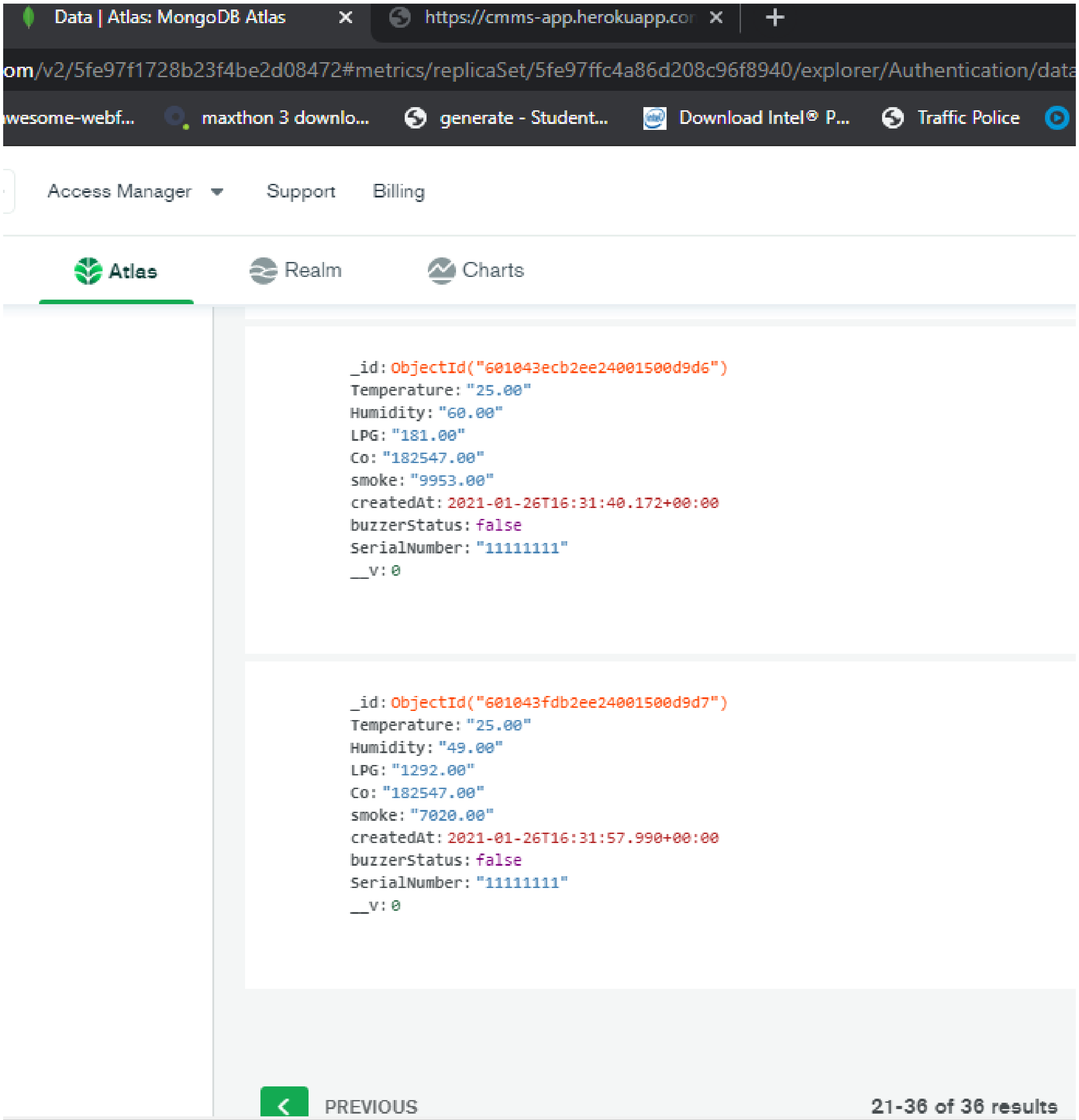

Figure 12: data stored in the database

The production safety of miners by alarming is done automatically when the sensor's measured value exceeds the threshold value of normal conditions for each collected value of Temperature, humidity, and gases. The CMMS prototype was tested in a normal environment. We have selected a threshold value of 32.00C for temperature, 100.00C for Humidity, 1000.00ppm for LPG, 70000.00ppm for CO, 5200.00ppm for Smoke to use in the experiment which may be different from the ones that can be selected and used in the mining site. An online alarming monitoring system provided by Async Webserver was created and can be accessed using the IP address provided to see the current data and providing alarming to the miners. Pressing the ON button shown on phone in Figure 11 or any computer that has access using provided IP address to alarm the miners in the underground mining site provides alarming functionality as needed.

Chapter 5: Conclusion and Recommendation

5.1 Conclusion

We have developed a prototype of a Coal Mine Monitoring System, for underground monitoring in the coal mine to prevent happening of the accident inside the mine. This was achieved by using WSN and ESP32 microcontroller to measure the parameters that occur in underground mine such as toxic gases, rising temperature and humidity, and smoking. The coal mine monitoring system developed can provide the alarm automatically when the data gathered by sensors crossing the threshold value and online monitoring via a web interface to improve the safety and health of the miners in the mining industry. By using the artificial environment we have tested our system so that the data collected by sensors stored on the CMMS web application developed with the purpose of future inspection and research to the interested one.

5.2 Recommendation

Due to the short time, budget constraint together with the impact of the COVID-19 pandemic, we only simulated the prototype and we did not achieve our targeted objective to make it in the real environment of mining sites. In this manner, our future work will be to implement the project in the real environment, design, and implementation of filter and power amplifier for signal processing functionality. Also, we recommend to the University of Rwanda to provide a long time and resources such as equipment required to conduct the practical project. Besides, we recommend the University of Rwanda to make the follow-up to the project submitted by the students and highlight the project which can be essential to the community and being implemented in the countrywide application. This will inspire the student to work hard to do more research and encourage them to know the digital technology of today’s generation.

REFERENCE

[1] S. K. Srivastava, “Real Time Monitoring System for Mine Safety using Wireless Sensor Network (Multi-Gas Detector),” 2015.

[2] S. Kumari, T. . Divya, K. Keerthana, S. Nisha, and S. Pallavi, “IOT based Coal Mine Safety Monitoring and Controlling,” *Int. J. Adv. Eng. Res. Sci.*, vol. 7, no. 3, pp. 395–399, 2020, doi: 10.22161/ijaers.73.56.

[3] A. M. Projects, N. M. Projects, S. E. Projects, I. M. Projects, S. R. Projects, and A. Projects, “MQTT based Monitoring System for Coal Mine using Raspberry,” pp. 1–7, 2021.

[4] C. Treatment, “Republic of Rwanda Paediatric,” no. September, pp. 1–50, 2012.

[5] V. Henriques, R. Malekian, and D. Capeska Bogatinoska, “Mine safety system using wireless sensor networks,” *2017 40th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2017 - Proc.*, vol. 4, pp. 515–520, 2017, doi: 10.23919/MIPRO.2017.7973480.

[6] M. Li and Y. Liu, “Underground Structure Monitoring with Wireless Sensor Networks,” pp. 69–78, 2007, doi: 10.1109/ipsn.2007.4379666.

[7] Y. Wang and L. Li, “Design of Intelligent Monitoring System Based on ZigBee Wireless Network,” no. Icmmse, pp. 340–345, 2015, doi: 10.2991/isrme-15.2015.146.

[8] X. Yi *et al.*, *Effects of seasonal air temperature variation on airflow and surrounding rock temperature of mines*, vol. 6, no. 3. 2019.

[9] T. Must, B. Something, and I. The, “The Most Dangerous Gases In Mining Firedamp - Methane Gas (CH4).”

[10] E. Systems, “ESP32 Series,” 2020.

[11] S. C. Systems, “Wireless Sensor Networks and Their Application A Survey,” *Environment*, pp. 1–12, 2004.

[12] P. Ideas, “MQ2 Gas Sensor Working and Its Applica,” pp. 1–7, 2021.

[13] S. Diagram, “used laptop i7,” pp. 1–11, 2020.

[14] T. Sensor and R. Pi, “ELECTRONICS,” pp. 1–29, 2020.

[15] Components101, “DHT11 Temperature and Humidity Sensor,” *Components101.Com*, pp. 1–6, 2019.

[16] T. Ohm, “Resistors & Circuits,” pp. 1–8, 2015.

[17] G. L. Iot and D. Loggers, “WiFi / Bluetooth Module ESP32,” 2021.

[18] G. Feedback, G. Started, A. Operations, and C. Rules, “MongoDB Atlas Overview,” pp. 1–8, 2021.

[19] P. D. Support, “What is Heroku ? Design is part of Heroku ’ s DNA , it ’ s at the heart of everything we do . We approach our work using the guiding principles of utility , simplicity , elegance , and quality . Every single person at Heroku sweats the details and contrib,” pp. 1–7, 2021.

[20] H. M. Language *et al.*, “Looking to become a front-end web developer?,” pp. 1–4, 2021.

[21] H. Electromechanical, P. Types, E. Mechanical, P. Modern, J. Henry, and B. Titanate, “A buzzer or beeper is an audio signalling device, [1] which may be mechanical, electromechanical, or piezoelectric (,” pp. 6–8, 2021.

[22] C. S. Sheets, F. Css, C. S. S. I. If, C. S. S. Tutorials, O. Css, and C. S. S. R. Our, “CSS : Cascading Style Sheets Key resources,” pp. 1–4, 2021.

[23] H. Reactor, “What is JavaScript Used For ? What is JavaScript ? What is JavaScript used for ?,” pp. 1–5, 2021.

[24] M. Features, “What is MongoDB ? Introduction , Architecture , Features & Example What is MongoDB ?”

[25] T. Node *et al.*, “Node.js,” pp. 1–11, 2021.

[26] P. D. Support, “What is Heroku ? our work using the guiding principles of utility , simplicity , elegance , and quality . Every single person at Heroku sweats the details and contributes to the Heroku Our philosophy,” pp. 1–7, 2021.

[27] T. Asynchronous *et al.*, “Introduction to the Asynchronous HTTP Server,” pp. 1–36, 2021.

Appendix A

```
#include <HTTPClient.h>
#include <Adafruit_Sensor.h>
#include <WiFi.h>
#include <DHT.h>
#include <MQ2.h>

// define DHTTYPE in our case is DHT11
#define DHTTYPE DHT11
// define required pins
#define DHTPIN 4 // Digital pin connected to the DHT sensor
#define MQGASPIN 35
//#define BUZZERPIN 26

// initialize MQ2 and DHT
DHT dht(DHTPIN, DHTTYPE);
MQ2 mq2(MQGASPIN);

// Replace with WIFI network credentials
const char* ssid = "CANALBOX-B53A";
const char* password = "6383905460";

// HTTP status
HTTPClient http;
// creation of webser
// check Web server port for work
```

```
WiFiServer server(80);
// we have to set variable to store the HTTP request
String header;
// declaration of output state(Auxiliar variables to store the current output state
String output26State = "off";
// Assign output variables to GPIO poins
const int output26 = 26;

// we need to keep track with time.
// current time
unsigned long currentTime = millis();
// previous time
unsigned long previousTime = 0;
// define timeout time in milliseconds
const long timeoutTime = 2000;

// end of Emmanuel try

// declaration of variables required.
String temperature;
String humidity;
String lpg;
String co;
String smoke;
String buzzerState;
// url to make a request
String url = "https://cmms-app.herokuapp.com/monitoring/update-data?data=";

float tempThreshold = 32.00;
float humiThreshold = 100.00;
float lpgThreshold = 1000.00;
float coThreshold = 70000.00;
float smokeThreshold = 5200.00;
```

```
//required function to collecting data
// temperature sensing function that return value of temperature
String readDHTTemperature() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  // Read temperature as Celsius
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  //float t = dht.readTemperature(true);
  // Check if any reads failed and exit early (to try again).
  if (isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return "--";
  }
  else {
    Serial.print("Temperature: ");
    Serial.println(t);
    return String(t);
  }
}

// Humidity sensing function that return value of humidity
String readDHTHumidity() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  if (isnan(h)) {
    Serial.println("Failed to read from DHT sensor!");
    return "--";
  }
  else {
    Serial.print("Humidity: ");
    Serial.println(h);
    return String(h);
  }
}
```

```
String readDHTTemperatureF() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float f = dht.readTemperature(true);
  if (isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return "--";
  }
  else {
    Serial.print("Fahrenheit: ");
    Serial.println(f);
    return String(f);
  }
}

// LPG sensing function with help of MQ2 gas sensor that return value of lpg value
String readMQLPG() {
  // Sensor reading require few seconds to read.
  float lpg = mq2.readLPG();
  if (isnan(lpg)) {
    Serial.println("Failed to read the value of lpg from MQ2 sensor!");
    return "--";
  }
  else {
    Serial.print("LPG: ");
    Serial.println(lpg);
    return String(lpg);
  }
}

// CO2 sensing funcdtion with help of MQ2 gas sensor that return value of co value
String readMQCo() {
  // Sensor reading with few seconds
  float co = mq2.readCO();
```

```
  if (isnan(co)) {
    Serial.println("Failed to read the value of co2 from MQ2 sensor!");
    return "--";
  }
  else {
    Serial.print("CO: ");
    Serial.println(co);
    return String(co);
  }
}

String readMQSmoke() {
  // Sensor reading with few seconds
  float smoke = mq2.readSmoke();
  if (isnan(smoke)) {
    Serial.println("Failed to read the value of smoke from MQ2 sensor!");
    return "--";
  }
  else {
    Serial.print("Smoke: ");
    Serial.println(smoke);
    return String(smoke);
  }
}
void setup()
{
  // buzzer pin setting
  pinMode(output26, OUTPUT);
  digitalWrite(output26, LOW);
  // setup serial to print required strings
  Serial.begin(115200);

  dht.begin();
  mq2.begin();
```

```
  delay(500);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
  }

  // start Async webserver
  server.begin();
}

void loop()
{
  // read all required data from functions repetitively to display on Async webserver.
  String tempValue = readDHTTemperature();
  String humiValue = readDHTHumidity();
  String lpgValue = readMQLPG();
  String coValue = readMQCo();
  String smokeValue = readMQSmoke();
  String tempFValue = readDHTTemperatureF();
  Serial.println(" ");

  // give access to the client to Async webserver.
  WiFiClient client = server.available();

  if (client) {
    currentTime = millis();
    previousTime = currentTime;
    Serial.println("New Client.");
    String currentLine = "";
```

```
    while (client.connected() && currentTime - previousTime <= timeoutTime) {
      currentTime = millis();
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        header += c;
        if (c == '\n') {
          if (currentLine.length() == 0) {
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println("Connection: close");
            client.println();

            if (header.indexOf("GET /26/on") >= 0) {
              Serial.println("GPIO 26 on");
              output26State = "on";
              digitalWrite(output26, HIGH);
            } else if (header.indexOf("GET /26/off") >= 0) {
              Serial.println("GPIO 26 off");
              output26State = "off";
              digitalWrite(output26, LOW);
            }

            // Display the HTML web page
            client.println("<!DOCTYPE html><html>");
            client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");
            client.println("<link rel=\"icon\" href=\"data:,\">");
            // CSS to style the on/off buttons
            client.println("<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}");
            client.println(".button { background-color: #4CAF50; border: none; color: white; padding: 16px 40px;");
            client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}");
```

```
          client.println(".button2 {background-color: #555555;}</style></head>");

          // Web Page Heading
          client.println("<body><h1>CMMS Web Server</h1>");
          client.println("<body><h1>CMMS Data Collected</h1>");
          client.println("<p>Temperature Value: " + tempValue + "</p>");
          client.println("<p>Humidity Value: " + humiValue + "</p>");
          client.println("<p>LPG Value: " + lpgValue + "</p>");
          client.println("<p>CO2 Value: " + coValue + "</p>");
          client.println("<p>Smoke Value: " + smokeValue + "</p>");
          client.println("<p>Fahrenheit Value: " + tempFValue + "</p>");

          // Give Access to the buzzer
          client.println("<p>GPIO 26 - State " + output26State + "</p>");
          if (output26State == "off") {
            client.println("<p><a href=\"/26/on\"><button class=\"button\">ON</button></a></p>");
          } else {
            client.println("<p><a href=\"/26/off\"><button class=\"button
button2\">OFF</button></a></p>");
          }

          client.println("</body></html>");

          // The HTTP response ends with another blank line
          client.println();
          // Break out of the while loop
          break;
        } else {
          currentLine = "";
        }
      } else if (c != '\r') {
        currentLine += c;
      }
    }
```

```
    }
    header = "";
    // Close the connection
    client.stop();
    Serial.println("Client disconnected.");
    Serial.println("");
  }

  // read value generated by functions
  temperature = readDHTTemperature();
  humidity = readDHTHumidity();
  lpg = readMQLPG();
  co = readMQCo();
  smoke = readMQSmoke();
  delay(1000);

  if ((temperature.toFloat() > tempThreshold) || (humidity.toFloat() > humiThreshold) || (lpg.toFloat() > lpgThreshold) || (co.toFloat() > coThreshold) || (smoke.toFloat() > smokeThreshold)) {
    output26State = "on";
    digitalWrite(output26, HIGH);
  } else {
    output26State = "off";
    digitalWrite(output26, LOW);
  }

  // reconnect just in case
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }

  String path = url + "11111111" + "*" + temperature + "*" + humidity + "*" + lpg + "*" + co + "*" + smoke;
```

```
 http.begin(path.c_str());

 //send get request to the web

 int httpResponseCode = http.GET();

 if (httpResponseCode > 0) {
  Serial.print("HTTP Response code: ");
  Serial.println(httpResponseCode);
  String payload = http.getString();
  Serial.println(payload);
 }
 else {
  Serial.print("Error code: ");
  Serial.println(httpResponseCode);
 }
 // Free resources
 http.end();
 delay(15000);
}
```

Appendix B

- Visiting this link: https://cmms-app.herokuapp.com/ to be able to access the CMMS Web APP.
- To access all codes used in the design. Click this link https://github.com/Coal-Mine-Monitoring-System to navigate to the codes.