Website:
www.aceiot.ur.ac.rw
Mail: aceiot@ur.ac.rw

College of Science and Technology

AFRICAN CENTER OF EXCELLENCE IN INTERNET OF THINGS (ACEIoT)

Developing a Machine Learning (ML) Model for an Integrated Industrial Process Control System

Case study: Industrial Air compressor

A dissertation submitted in partial fulfilment of the requirements for the award of Masters of Science degree in internet of things: Embedded Computing System

Submitted by:

UZABAKIRIHO Pascal (**Ref No:** 222023003)

August, 2025

DECLARATION

I, UZABAKIRIHO Pascal, Master ‘student from African Center of Excellence in internet of things (Embedded Computing Systems), at University of Rwanda. I declare that this research thesis is my own original work and it has never been presented before anywhere in the world.

UZABAKIRIHO Pascal

Ref: 222023003

Signed:

Date: 27/08/2025

BONAFIDE CERTIFICATE

This is to certify that the research work entitled "**Developing a Machine Learning (ML) Model for an Integrated Industrial Process Control System**" is a record of the original work done by **UZABAKIRIHO Pascal** (**222023003**), MSc. IoT- ECS Student at the University of Rwanda / College of Science and Technology / African Center of Excellence in Internet of Things, the Academic year 2023/2025.

This work has been submitted under the supervision of **Dr. KABANDANA Innocent** and **Dr. MUSONI Wilson.**

Main Supervisor: Dr. KABANDANA Innocent	Co-Supervisor: Dr. MUSONI Wilson
Date:	Date:
Signature:	Signature:

The Head of Masters studies and Trainings

Dr. James Rwigema

Signature:

Date:

Signature…………………

ACKNOWLEDGMENTS

My special thanks to the Almighty God and my family for their incredible support during the entire studies and research period. My gratitude for their time and support they offered me till the completion of this research.

I thank my supervisors, **Dr. KABANDANA Innocent** and **Dr. MUSONI Wilson**, for their ongoing support, supervision, advice, and guidance that helped this project come to a successful conclusion.

I would like to thank the entire management team at the ACEIoT for their invaluable advice and help. I appreciate all of the professors and classmates who provided every kind of assistance needed to complete this research.

Although it would be hard to list everyone who helped me here, their support is appreciated.

Thank you and May God bless you All!

ABSTRACT

Industrial automation's dependence on conventional human techniques has long been linked to inefficiencies, higher operating costs, and a higher chance of equipment failure. This demonstrates the urgent need for creative solutions that can lower costs and improve operational efficiency in the processing and manufacturing sectors. The creation of a Smart Industrial Machine Control and Monitoring System is a crucial step in the advancement of industrial automation to handle these issues. The goal of this system is to update the operation and management of vital machinery, such industrial air compressors, which are essential to many different sectors. Lack of predictive maintenance, lack of real-time monitoring, and restricted control over operational parameters are the main problems this technology attempts to solve. These difficulties have a major impact on productivity and profitability because they increase downtime, energy inefficiency, and non-compliance with safety regulations. To address these issues, the suggested approach makes use of machine learning, IoT technologies, and microcontroller-based control systems. While machine learning algorithms evaluate data to forecast problems and improve maintenance plans, sensors keep an eye on critical parameters like temperature, vibration, oil level, and voltage supply. NodeMCU modules offer remote access and real-time data collecting, which facilitates effective administration and prompt intervention through alert notifications. This technology improves energy efficiency, operational lifespan, and cost-effectiveness in addition to guaranteeing adherence to safety rules. The study shows how digital solutions in industrial automation may revolutionize air compressor management, opening the door to greater sustainability and competitiveness in the industrial sector.

Keywords: Industrial Automation, Real-time Monitoring, Predictive Maintenance, IoT (Internet of Things) and Machine Learning.

Table of Contents

LIST OF FIGURES

LIST OF ACRONYMS

AC: Alternating Current

AI: Artificial Intelligence

EEPROM: Electrically Erasable Programmable Read-Only Memory

GSM: Global System for Mobile Communications

HC-SR04: Ultrasonic Distance Sensor (specific model)

IoT : Internet of Things

LCD: Liquid Crystal Display

MCU: Microcontroller Unit

ML: Machine Learning

Wi-Fi: Wireless Fidelity

CHAPTER 1: GENERAL INTRODUCTION

1.1 Introduction

Automation, digital technology, and the Internet of Things (IoT) are radically changing the industrial landscape. Rwandan industries are up against many obstacles in their mission to modernize and become globally competitive. An overview of the Smart Industrial Machine Control and Monitoring System, which is especially intended to meet the requirements of Rwandan industries, is given in this chapter, with a focus on air compressor control. Through the integration of cutting-edge technologies like machine learning, NodeMcu, IoT sensors and the Internet of Things, this system seeks to improve safety protocols, lower costs, and increase operational efficiency in industrial settings.

1.2 Background of the study

The outdated manual labor techniques and insufficient monitoring systems pose significant obstacles to Rwandan industries, especially those that depend on industrial air compressors. These compressors are essential for many different industrial processes, but inefficiencies like poor predictive maintenance, insufficient real-time monitoring, and forgetting to monitor crucial parameters like temperature, vibration, oil cooling levels, and voltage supply can result in serious operational issues. These difficulties not only reduce output but also raise operating expenses and raise the possibility of equipment failure, which could lead to expensive downtime and decreased productivity [1].

This study aims to investigate how conventional operations can be changed by a Smart Industrial Machine Control and Monitoring System designed especially for industrial air compressors. The system will monitor important parameters with the help of NodeMCU, the central microcontroller: temperature to prevent overheating, oil cooling levels to guarantee effective operation, compressor vibration to identify mechanical problems, and voltage supply to prevent electrical defects. These elements are essential because imbalances can result in equipment failure, higher energy usage, and safety risks [2].

The suggested system seeks to boost Rwandan industry through the utilization of cutting-edge technologies, particularly machine learning algorithms for predictive analytics. Data gathered from these metrics can be analyzed by machine learning to find trends, foresee probable breakdowns, and suggest prompt maintenance procedures. This proactive strategy encourages safety and dependability in industrial operations in addition to improving operational efficiency [3].

1.3. Problem Statement

There are significant obstacles to growth and competitiveness facing Rwanda's and Africa's industrial landscapes. Many industries continue to use conventional management techniques, especially when it comes to running necessary machinery like air compressors. These environments have severe operational inefficiencies due to a lack of predictive maintenance and real-time monitoring, which raises hazards and increases operating expenses. Frequent unplanned equipment failures have a detrimental impact on the overall performance of these industries by causing production delays, wasteful energy use, and possible safety risks.

Furthermore, the inability of stakeholders to implement proactive management methods is disadvantaged by the lack of integrated digital solutions. Reactive decision-making is common, frequently relying on evidence from the past rather than current ideas. This strategy restricts the possibility of optimizing operations and delays Rwandan industries from implementing Industry 4.0 methods [4].

In order to overcome these obstacles, this research aims to pinpoint the crucial constraints that Rwandan and African industries must overcome. It then goes on to show how a machine learning-enhanced Smart Industrial Machine Control and Monitoring System can alleviate these problems, ultimately promoting economic expansion and boosting competitiveness internationally [5].

1.4. Research Objective

1.4.1. General Objective

This study's main goal is to create and put into place a smart industrial machine control and monitoring system that is specifically suited for Rwanda's efficient industrial air compressor management. This system uses real-time monitoring, predictive maintenance, and machine learning-driven insights to increase operational efficiency, lower costs, and improve safety standards.

1. 4.2. Specific objectives

- To identify the key operational challenges faced by Rwandan industries in managing air compressors.
- To develop a prototype of the smart industrial machine control and monitoring system using IoT, and ML technologies.

- To evaluate the performance of the system in terms of operational efficiency, maintenance optimization, and cost reduction.

1.5 Hypotheses

The first hypotheses, there would be a significant decrease in the operational costs related to air compressor management with the installation of a smart industrial machine control and monitoring System. The second hypotheses predicts computable improvements in operational efficiency and a decrease in the frequency of equipment failure as a result of machine learning algorithms supporting real-time monitoring and predictive maintenance. Lastly, third hypotheses is that the incorporation of cutting-edge technology, including machine learning, would improve safety standards and legal compliance in Rwandan industry, which will support overall sustainability and operational excellence.

1.6 Study Scope

The scope of my study is limited to the development and evaluation of the smart industrial machine control and monitoring system in the context of industrial air compressors within Rwanda. The research focuses on the technical design, implementation, and performance assessment of the system. It does not encompass other types of industrial machinery or broader economic factors.

1.7 Significance of the Study

My study holds significant importance for both academia and the industrial sector in Rwanda. By providing insights into the application of advanced technologies, particularly machine learning, in industrial management, it contributes to the body of knowledge in the fields of industrial automation and IoT. Furthermore, the findings of this research can serve as a blueprint for future initiatives aimed at modernizing the Rwandan industrial sector, ultimately fostering economic growth, technological advancement, and sustainable practices.

1.8. Organization of the study

The study is organized into six chapters, each addressing a key component of the research. Chapter 1 provides a general introduction, outlining the study's background, problem statement, objectives, scope, hypotheses, significance, and an overview of the organization. Chapter 2 focuses on the literature review, discussing related works and identifying gaps in existing research. Chapter 3 details the research methodology, including hardware and software requirements, the cloud platform, and the methodological approach. Chapter 4 covers system analysis and design, presenting the system's architecture, hardware and software designs, circuit diagrams, and flowcharts. Chapter 5 presents the results and analysis, showcasing data collection, system functionality, and a system dashboard for insights. Finally, Chapter 6 concludes the study with key findings and offers recommendations for future research. Supplementary sections include references and appendices for additional information.

1.9 Gantt chart

The Gantt chart organizes a six-month project timeline. It begins with literature review and system design in the first two months. Hardware development follows in months three and four, overlapping with software implementation. Testing and validation occur in months five and six, ensuring the system's reliability. Finally, results analysis and report writing are completed in the last month. This structured plan ensures timely progress and alignment with project objectives.

Task	Month 1	Month 2	Month 3	Month 4	Month 5	Month 6
Literature Review	✓	✓				
System Design		✓	✓			
Hardware Development			✓	✓		
Software Implementation				✓	✓	
Testing and Validation					✓	✓
Results Analysis and Writing						✓

CHAPTER 2: LITERATURE REVIEW

2.0 Introduction

Literature Review introduces existing research and developments related to smart industrial machine control and monitoring systems. It focuses on air compressor management, IoT integration, and machine learning for predictive maintenance. The review identifies gaps in current systems and highlights the potential benefits of using advanced technologies to enhance efficiency, reduce costs, and improve equipment longevity. It provides a foundation for understanding the significance of implementing such systems in industrial operations.

2.1 .Definitions

2.1.1. Internet of Things

The **IoT** refers to a system of interconnected devices, equipped with sensors, software, and communication technologies, that collect, exchange, and act on data over the internet. These devices operate in real time, enabling seamless interaction and automation [8]. IoT time insights into their performance. By embedding smart sensors in equipment such as air compressors, pumps, and conveyors, industries can collect critical data on operational conditions, which can then be analyzed to improve efficiency, predict maintenance needs, and optimize overall system performance. Internet of Things integration refers to the interconnection of physical devices, machines, and sensors to the internet or local networks, enabling them to collect and exchange data. In industrial settings, IoT integration allows for the remote monitoring and control of machinery and systems [8].

2.1.2 Applications of IoT Integration

IoT integration has vast applications across various industries, from manufacturing to healthcare. In industrial automation, IoT allows for the monitoring of production lines, equipment health, and energy consumption. For example, in air compressor systems, IoT-enabled sensors can measure parameters like pressure, temperature, and vibration, sending this data to centralized platforms for analysis. Similarly, smart factories use IoT integration to enable automated processes and predictive maintenance. In agriculture, IoT helps monitor soil moisture, weather conditions, and crop health, while in healthcare, IoT devices track vital signs and improve patient care by enabling remote monitoring [8].

2.1.3 Advantages of IoT Integration

The integration of IoT technologies brings several significant advantages, especially in industrial settings. One of the primary benefits is real-time monitoring. IoT allows operators to track the performance and health of machines continuously, identifying potential issues before they escalate. This leads to predictive maintenance, where maintenance tasks are scheduled based on data-driven insights rather than fixed intervals, reducing downtime and repair costs. Additionally, IoT integration enhances efficiency by automating tasks and optimizing energy use, leading to lower operational costs [7].

Another key advantage is improved decision-making. IoT systems provide valuable data that can be analyzed to make informed decisions about resource allocation, system adjustments, and process improvements. IoT also promotes remote management, allowing operators to monitor and control machines from any location, which is especially beneficial in large-scale or geographically dispersed operations. Overall, IoT integration drives cost savings, increased productivity, and enhanced operational safety, making it a critical component in the digital transformation of industries [9].

2.1.4. Introduction to Air Compressor

An air compressor is a mechanical device that converts power, usually from an electric motor or internal combustion engine, into potential energy stored in compressed air. The compressor works by drawing in air, compressing it, and then storing it in a tank or delivering it directly to a system that requires it. The compression process involves reducing the volume of air while increasing its pressure, making it a highly efficient form of energy that can be used in various applications. Air compressors come in different sizes and types, each suited to specific needs, ranging from small portable models to large industrial machines [6].

2.1.5 Applications of Air Compressors

Air compressors are crucial in various industrial, commercial, and residential applications due to their versatility. In industrial settings, they are commonly used to power pneumatic tools like drills, hammers, and wrenches, as well as machinery that requires compressed air for operation. They are also essential for spray painting and surface finishing tasks, where a steady, pressurized air supply is needed. In the manufacturing sector, air compressors help drive automation systems and actuators, improving production efficiency. Similarly, in construction and mining, they are used to operate heavy equipment and tools, making them integral to many industrial processes.

In commercial applications, air compressors play a vital role in powering refrigeration and cooling systems, as well as inflating tires at service stations. They are also used in cleaning tasks, especially in workshops or factories where high-pressure air can remove dust and debris from machinery. For medical purposes, air compressors provide clean, dry air for dental equipment, respiratory devices, and other medical instruments. In residential settings, air compressors are used in many projects, such as operating small tools or inflating objects like tires or air mattresses[15].

2.1.3 Types of Air Compressors

There are two main types of air compressors: positive displacement compressors and dynamic compressors. Positive displacement compressors, such as reciprocating compressors and rotary screw compressors, work by trapping a certain volume of air and reducing its space, thus increasing pressure. Reciprocating compressors use pistons to compress air in a cylinder, while rotary screw compressors employ two helical screws that continuously compress air. Scroll compressors, another type of positive displacement compressor, use interlocking scrolls to compress the air, offering smooth and efficient operation.

On the other hand, dynamic compressors increase the speed of air, converting its kinetic energy into pressure. These compressors are typically used in larger applications, such as turbines and jet engines. Centrifugal compressors, a subtype of dynamic compressors, use a rotating impeller to accelerate air, while axial compressors are designed for applications requiring very high flow rates, such as in jet propulsion systems [6].

2.1.4 Key Features and Advantages

Air compressors offer several advantages, including versatility, making them suitable for a wide range of applications, from powering industrial tools to inflating tires. Their energy efficiency is another important feature, as they can efficiently convert electrical or mechanical power into high-pressure air. Compressors are also known for their durability, designed to withstand demanding environments and continue functioning reliably for extended periods. Additionally, they are highly adaptable, with various types and sizes available to meet specific needs, whether it be for small scale tasks or large industrial operations.

With their broad applications and reliable performance, air compressors are indispensable in industries worldwide, enabling efficient operations, reducing costs, and improving overall productivity.

2.2.5 Machine learning

Machine learning (ML), a branch of artificial intelligence, enables systems to learn from data and make predictions or decisions without being explicitly programmed. It utilizes a variety of algorithms such as Decision Trees, Random Forests, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Naïve Bayes, and Convolutional Neural Networks (CNNs) to solve a wide range of tasks, including classification, regression, and pattern recognition. Deep learning, a subset of ML, has shown exceptional performance in applications like human activity recognition using wearable sensors. Models such as CNNs, Recurrent Neural Networks (RNNs), and Transformer networks excel at capturing complex patterns in data, offering accurate and real-time insights in dynamic environments [11].

Building on these capabilities, machine learning is increasingly being applied to industrial process control to improve automation, efficiency, and decision-making. Developing an ML model for an integrated industrial process control system involves leveraging historical and real-time data to monitor operations, detect anomalies, predict equipment failures, and optimize system performance. By shifting from traditional rule-based methods to intelligent, adaptive solutions, industries can achieve more responsive, cost-effective, and reliable control systems that align with the goals of smart manufacturing and Industry 4.0.

2.3 Related works

A lot of research has been done on the incorporation of digital technology into industrial machine control, especially when it comes to real-time monitoring and predictive maintenance. According to Lamb [6], industrial automation systems can take the role of antiquated human labor methods, increasing productivity and cutting expenses in developed nations. Industrial machinery can be operated with little assistance from humans by utilizing sensors and control algorithms.Goes into more detail on how predictive maintenance can be enhanced by combining machine learning and IoT (Internet of Things), allowing systems to foresee problems and reduce equipment downtime. However, the majority of the focus of both studies is on established industrial settings, which leaves a large vacuum about the applicability of these technologies in African industry, especially in Rwanda.

Lamb Talks about the value of real-time monitoring in air compressor systems, where it's necessary to continuously evaluate things like vibration, temperature, and oil levels to avoid operational breakdowns. His work demonstrates how the reactive nature of standard approaches to air compressor maintenance makes them insufficient. On the other hand, [9] highlights that IoT-enabled systems can drastically cut downtime by using predictive analytics that notify operators of possible problems before they arise. Even though these studies are essential to expanding our knowledge of industrial automation, they ignore the financial and technological barriers that exist in African nations, where the expense of putting such sophisticated systems into place is frequently unaffordable.

The integration of machine learning and artificial intelligence into industrial processes is the subject of more recent research. Schwab, K Investigates how artificial intelligence may transform industrial processes by enhancing advanced data analytics for safety standards and decision-making. By using machine learning models, one can anticipate when equipment will break, allowing for prompt repair. Although Schwab talks about the potential of AI in industrial automation, his research does not offer specific answers for businesses in underdeveloped countries like Rwanda that have restricted access to technology. Predictive maintenance is a crucial part of the "smart factory" concept, according to Kumar in which every piece of equipment is connected to a central system that continuously checks performance. They draw attention to how important predictive maintenance is to reducing downtime and increasing the life of industrial equipment. Despite these developments, the literature lacks information about how these systems could be modified for smaller companies in Africa because the focus of their research has been on large-scale industrial operations in nations with robust technology ecosystems.

An intriguing viewpoint on the use of data science and IoT in industrial applications is presented by [12]. His research demonstrates how real-time analytics and predictive maintenance capabilities provided by the Internet of Things when combined with machine learning can revolutionize industrial process optimization. The report does not, however, address the particular difficulties faced by African companies, such as the high expense of infrastructure and restricted access to cutting-edge technology. Lastly, Kumar clarifies the challenges that developing-nation industries especially those in Africa face. In order to solve problems like an unstable power supply and a shortage of specialized workers for maintaining complex industrial systems, he emphasizes the necessity for affordable, scalable solutions. Kumar's effort, nevertheless, falls short of offering comprehensive fixes specifically designed for air compressors a vital part of many sectors.

2.4 Literature Gaps

Despite advancements in industrial automation, studies concentrating on African businesses, particularly those in Rwanda, are conspicuously lacking. Studies that have already been done on the subject of integrating machine learning with low-cost, scalable technologies like NodeMCU for real-time monitoring and predictive maintenance in African industrial settings are inadequate. There is still a need for a customized, reasonably priced Smart Industrial Machine Control and Monitoring System that can handle the unique difficulties associated with air compressor management in Rwanda.

By creating a smart industrial machine control and monitoring system for air compressors in Rwandan industries, this research seeks to close the gap. This system will monitor vital indicators temperature, vibration, oil cooling levels, and power supply using low-cost technologies NodeMCU. The integration of machine learning algorithms will enable predictive maintenance, thereby mitigating operational inefficiencies and augmenting safety standards. This study will help create more sustainable and effective industrial operations by addressing the unique requirements of Rwanda's industrial landscape.

2.5 Comparative Summary of My Research and Existing Studies

Research Work	**Focus**	**Identified Gaps**	**Proposed Solutions in This Research**
Frederick Lamb (2021) Industrial Automation	Industrial automation for productivity.	Limited to developed nations; lacks applicability to African industries.	Use of affordable technologies (NodeMCU) tailored for Rwandan industrial needs.
Muhammad A. Khan (2021) Machine Learning for IoT	IoT and ML for predictive maintenance.	High implementation costs; not tailored for resource-constrained regions.	Low-cost IoT solutions for real-time monitoring and predictive maintenance.

Pratik Gohil (2021) Hands-On IIoT	Real-time monitoring in air compressors.	Ignores cost and technical limitations in developing countries.	Incorporation of low-cost sensors to monitor air compressor parameters.
Perry Lea (2020) IoT and Edge Computing for Architects	AI for industrial decision-making.	Inapplicability to underdeveloped countries with limited tech access.	Simplified AI models optimized for low-power, cost-effective implementations.
Klaus Schwab (2021) Impact of AI on Business & Society	Smart factory concepts and predictive maintenance.	Focused on large-scale industries; lacks scalability for smaller firms.	Design of scalable, modular solutions for small and medium-sized enterprises (SMEs).
Chhabra, J. K (2023) Data Science for IoT and Industrial Application.	IoT and data science in optimization.	High infrastructure costs and lack of affordability for African industries.	Cost-efficient hardware and open-source software integration to reduce costs.

CHAPTER 3: RESEARCH METHODOLOGY

3.1. Introduction

Research methodology refers to the systematic, documented approach used in managing a project, that includes procedures, definitions, and explanations of techniques and tools used to collect, analyze, store, and present data as part of research [16]. This section discusses and gives detailed information on the study area and Scope of the project, the research method to be used, the target population, Sample size and sampling techniques, Data collection tools, and the System Development Approach that will be used to conduct this study and allow findings to be replicated to validate them or deduce a conclusion from that analysis.

3.2. Research Methods

The research methodology for developing ML model for an Integrated Industrial Process Control System employs an experimental and prototype-based approach. This method supports iterative testing and refinement of both hardware and software components, ensuring the system's reliability, accuracy, and real-time responsiveness in industrial environments. The experimental aspect allows for evaluating sensor data accuracy and ML model performance under simulated process conditions, while prototype development ensures the integration and functionality of all system components.The prototype is centered around the NodeMCU (ESP8266), which serves as the primary microcontroller responsible for data acquisition, processing, and communication. The system incorporates a suite of sensors including a temperature sensor for thermal monitoring, a voltage sensor for electrical health tracking, an ultrasonic sensor to measure oil levels in storage tanks, and a vibration sensor for detecting abnormal equipment behavior. These sensors provide critical real-time data which is processed locally and sent to a cloud platform for machine learning-based analysis and decision-making.

The ML model processes this data to identify patterns, detect anomalies, and make predictive adjustments to the system. In the event of abnormal conditions such as overheating, excessive vibration, low oil levels, or voltage drops an SMS alert is automatically sent to the operator. Simultaneously, a buzzer is activated on-site to provide immediate audible alerts to nearby personnel. The system also includes a web-based dashboard, which offers real-time monitoring and visualization of sensor data and system status, enabling remote supervision and control.This integrated approach ensures a smart, responsive, and scalable solution for industrial process control, leveraging IoT and machine learning technologies to enhance operational efficiency, safety, and preventive maintenance.

3.2.1. Interview

The interview method in this research is used to collect practical, experience based insights from industrial air compressor operators and maintenance technicians to identify common challenges and fault conditions encountered during daily operations. Through semi structured interviews, participants are asked targeted questions about frequent compressor failures such as overheating, vibration, voltage fluctuations, and oil level issues, their current methods of detection and response, and their opinions on the effectiveness of traditional maintenance practices. This qualitative approach allows for a deeper understanding of real world problems, operator needs, and system gaps, providing essential input for designing a machine learning-based monitoring and control system. The feedback gathered will help ensure that the proposed intelligent solution is realistic, relevant, and user friendly, while also enhancing reliability, safety, and operational efficiency in industrial environments [19].

3.2.2. Data Collection

Data is gathered from a suite of integrated sensors specifically selected to monitor the operational health and efficiency of an industrial air compressor. These include:

- ✓ **Temperature sensor** to track internal heat levels and prevent overheating.
- ✓ **Voltage sensor** to monitor power input and detect fluctuations that may signal electrical instability.
- ✓ **Ultrasonic sensor** to measure oil levels, ensuring proper lubrication and avoiding damage due to low oil conditions.
- ✓ **Vibration sensor** to detect abnormal mechanical movements such as imbalance, misalignment, or bearing wear.

The sensor data is processed by a NodeMCU (ESP8266) microcontroller, which transmits real-time readings to a cloud-based platform for machine learning-driven analysis. This enables the system to identify trends, detect anomalies, and trigger appropriate responses. In the event of abnormal conditions, the system automatically activates a relay module to switch off the air compressor, preventing further damage. Simultaneously, a buzzer sounds on-site to alert nearby personnel, and an SMS notification is sent to the operator. Additionally, a web-based dashboard provides remote monitoring and visualization of key operational metrics, supporting informed decision-making and proactive maintenance in industrial settings.

3.2.3.Data Analysis

Data analysis in the integrated industrial process control system is performed using machine learning models embedded within the platform. These models continuously process real-time sensor data such as temperature, voltage, oil level, and vibration collected from the industrial equipment and stored in the cloud. By analyzing this data, the system can detect anomalies, predict equipment failures, and optimize operational parameters to enhance process efficiency and safety. The machine learning algorithms also enable predictive maintenance by identifying long-term trends and patterns in equipment performance. This allows the system to generate actionable insights and alerts for operators, helping to prevent unplanned downtime and reduce maintenance costs. Additionally, comprehensive reports summarizing system health and performance metrics are generated regularly, providing industrial engineers and decision-makers with a clear overview of the process status and enabling data-driven optimization of industrial workflows [19].

3.3. Machine Learning Process

The machine learning process for the integrated industrial process control system begins with data acquisition from key sensors installed on the industrial air compressor. These include a temperature sensor, voltage sensor, ultrasonic sensor for oil level detection, and a vibration sensor. These sensors generate continuous real-time data reflecting the operational state and health of the compressor. This data is transmitted to a cloud-based platform, where it undergoes preprocessing including cleaning, normalization, and handling of missing values to ensure data quality and consistency. Once the data is prepared, it is used to train a Random Forest Classifier, a robust ensemble learning algorithm known for its high accuracy and ability to handle both categorical and continuous input variables [19].

The Random Forest model is trained to classify the system's operational state into categories such as normal, warning, or fault, based on sensor input patterns. Using historical data from simulated or real industrial conditions, the model learns how specific sensor combinations correlate with different machine states. Through cross-validation and hyperparameter tuning, the model's accuracy is enhanced and overfitting is minimized. After deployment, the trained model processes real-time sensor data to detect anomalies, predict potential failures, and trigger safety mechanisms. When abnormal conditions are identified, the system can automatically activate a relay to shut down the compressor, trigger a buzzer on-site, and send an SMS alert to the operator. A web-based dashboard also displays live system status and predictions, enabling remote monitoring and informed decision-making.

3.3.1. Random Forest Algorithm Definition

Random Forest is a powerful machine learning algorithm that uses an ensemble of decision trees to make predictions [17]. And serves as a highly effective classification tool. Random Forest is an ensemble learning method that builds multiple decision trees during training, with each tree trained on a randomly selected subset of the data and features. This randomness increases the model's generalization ability and reduces the risk of overfitting, which is particularly valuable in complex industrial environments with noisy or fluctuating sensor data.

For this project, Random Forest is used to classify the operational status of industrial equipment such as air compressors into categories like normal**,** warning**,**or fault [19]**.** The model takes input from various sensors, including temperature, voltage, oil level via ultrasonic sensor, and vibration. By analyzing patterns across these multiple variables, the algorithm can make reliable predictions about equipment health. Its ability to handle both numerical and categorical data, combined with high accuracy and robustness, makes Random Forest a suitable and scalable solution for real-time anomaly detection and predictive maintenance in industrial process control systems.

3.3.2. Application of Random Forest in Developing a Machine Learning (ML) Model for an Integrated Industrial Process Control System

The application of the Random Forest algorithm in developing a machine learning model for an integrated industrial process control system focuses on improving the reliability and safety of industrial air compressors. By using real-time data from multiple sensors such as temperature, voltage, oil level via ultrasonic sensor, and vibration the system continuously monitors the compressor's operational status [9]. This data, collected through a NodeMCU (ESP8266) and transmitted to the cloud, undergoes preprocessing to ensure accuracy and consistency. A Random Forest Classifier is then trained to classify the system's state as either

"Normal" or "Abnormal," offering robust performance even under variable and noisy conditions [16].

Once deployed, the model enables real-time decision-making. In the event of an abnormal condition, it automatically shuts down the compressor via a relay, triggers an on-site buzzer, sends an SMS alert to the operator, and updates a web-based dashboard. The system benefits from high accuracy, resistance to data noise, and the ability to identify which sensor readings are most critical to predicting faults. This intelligent control approach enhances operational efficiency, supports predictive maintenance, and demonstrates how machine learning can transform industrial process monitoring and automation.

CHAPTER 4: SYSTEM DESIGN AND DEVELOPMENT

4.1. Introduction

This chapter presents the system architecture and development process for the proposed Machine Learning-based Integrated Industrial Process Control System, with a focus on monitoring and controlling an industrial air compressor**.** The system is designed to enhance real-time fault detection, predictive maintenance, and autonomous control using sensor data and machine learning techniques. The design integrates multiple sensors including temperature, voltage, vibration, and ultrasonic oil level detectors with a NodeMCU (ESP8266) microcontroller for real-time data acquisition and transmission. This industrial application focuses on machine health monitoring. Data collected from the air compressor is processed and analyzed using a Random Forest classifier, which detects abnormal patterns and predicts system failures. Upon identifying a fault, the system automatically takes action by shutting down the compressor via a relay**,** alerting operators via SMS, and activating a local buzzer, while updating a web-based dashboard for visualization and decision support. This section elaborates on the hardware and software design, sensor integration, communication flow, and the implementation of the ML model for real-time industrial process optimization.

4.2. System architecture

4.2.1. Block diagram of the system

To guarantee effective air compressor monitoring and control, the Smart Industrial Machine Control and Monitoring System's hardware architecture combines sensors, a microprocessor, and actuators. Important sensors include a temperature sensor (LM35) to avoid overheating, a vibration sensor to identify unusual mechanical activity, a voltage sensor to track power variations, and an ultrasonic sensor to check cooling oil levels. The NodeMCU microcontroller, the central processor of the system, receives real-time data from these sensors.

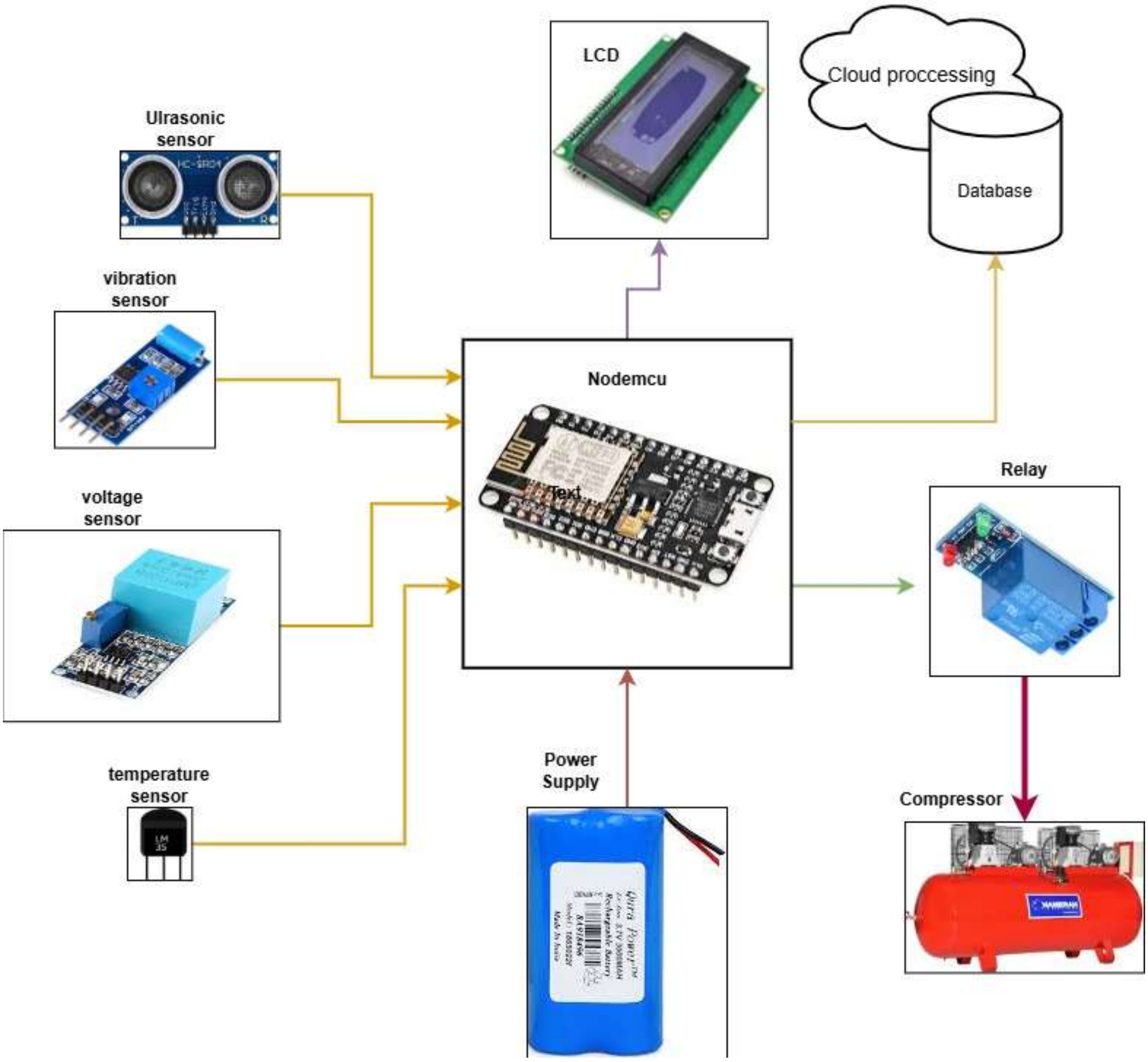

Figure 1: Block diagram of the system

Operators can keep an eye on crucial parameters on-site thanks to the NodeMCU's processing of the data and local display of it on an LCD module. In parallel, it sends the data to a cloud-based database for remote storage and sophisticated analytics, allowing machine learning-based predictive maintenance. As a safety precaution, a relay module immediately turns off the compressor when it detects any severe situation, like overheating or extreme vibration. A controlled power source powers the entire system, guaranteeing dependable functioning. This design combines local control, remote data processing, and real-time monitoring to increase compressor lifespan, decrease downtime, and improve safety.

4.2.2. Hardware architecture

The NodeMCU microcontroller, which serves as the industrial machine's central processing unit for monitoring and control, is at the center of the system's hardware architecture. It incorporates four key sensors: a vibration sensor to identify mechanical failures through unusual vibrations, a temperature sensor to track and avoid overheating, a voltage sensor to monitor power levels and identify anomalies, and an oil level sensor to guarantee proper lubrication. The NodeMCU processes the data from these sensors and shows it on an LCD screen so the user may see it in real time.

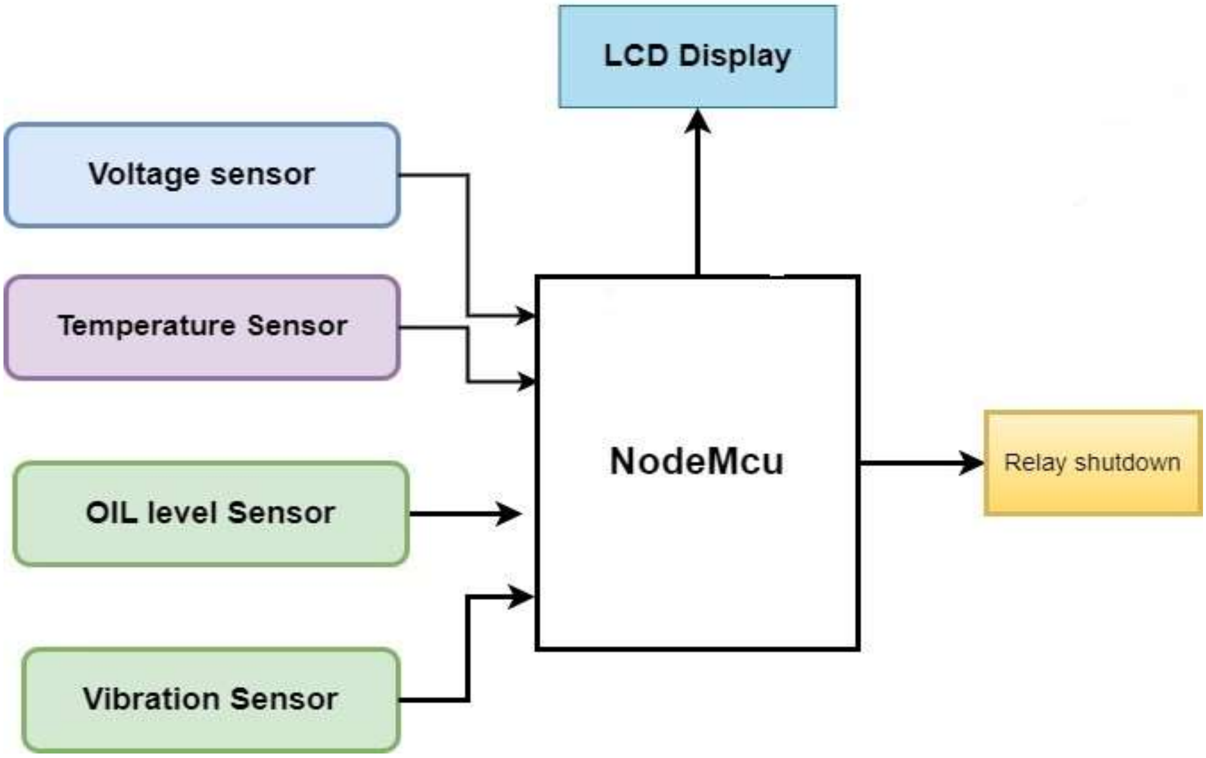

Figure 2: Hardware architecture

The NodeMCU activates a relay module to turn off the machine in the case that any sensor detects unusual activity, averting possible harm or safety risks. This architecture improves operating efficiency and safety by offering a dependable and effective way to monitor important machine characteristics.

4.2.3. Flow Chart diagram of the system

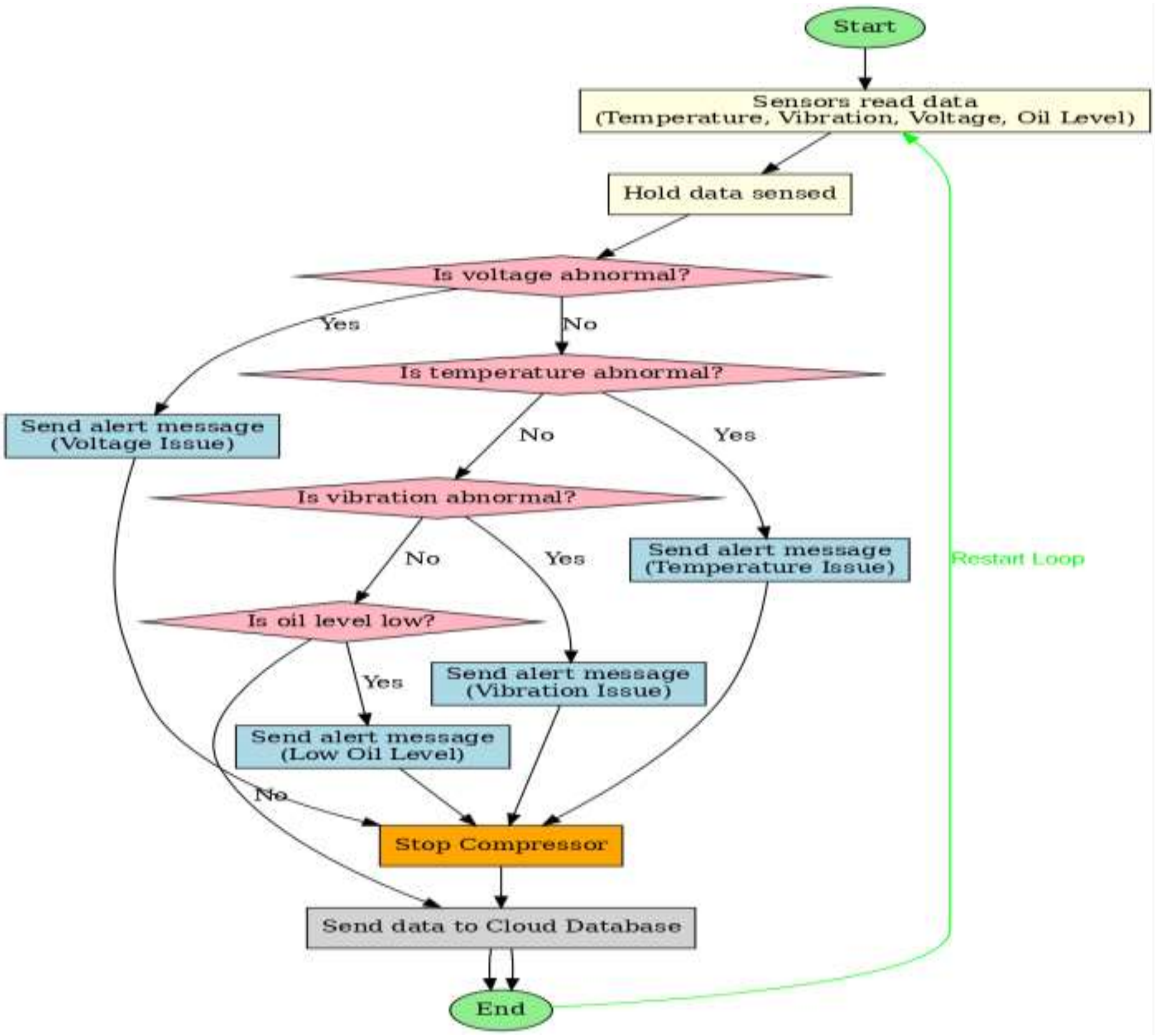

Figure 3: Flow chart of the system

In order to continuously monitor the machine's working characteristics, the IoT-based machine monitoring system with integrated machine learning and alerting capabilities first gathers data from a variety of sensors, including as temperature, vibration, voltage, and ultrasonic sensors. Initial processing of this real-time data takes place on a NodeMCU microcontroller. After processing, the data is sent to a cloud-hosted machine learning model. In order to identify irregularities, forecast maintenance requirements, and guarantee effective functioning, the machine learning model examines the data.

Decisions about whether to continue with regular operations or to sound an alarm in the event of abnormal conditions are made based on the analysis. Stakeholders receive notifications such as emails or SMS about important problems or planned repair. For local monitoring, the system also shows real-time statistics and alerts on an LCD. While the notification system guarantees prompt action, the incorporation of machine learning improves predictive maintenance, which in turn improves machine efficiency and decreases downtime.

4.3. Tools and component used

4.3.1.Hardware requirements

4.3.1.1. NodeMCU ESP8266 wi-fi internet development board

In this research, the design utilizes NodeMCU Microcontroller. As the name suggests, a microcontroller is an integrated system provided to control the operation of embedded systems. It includes a processor, a memory and peripherals to help its communication with environment. NodeMCU ESP8266 wi-fi internet development board module is an open-source firmware and
Development kit that helps to prototype an IoT product within Arduino codes.

Figure 4: NodeMCU ESP8266 WIFI Internet Development Board

The best way to develop quickly an IoT application with less Integrated circuits to add is to choose this circuit "NodeMCU". Today,we will give a detailed Introduction on NodeMCU V3. It is an open-source firmware and development kit that plays a vital role in designing a proper IoT product using a few script lines. The module is mainly based on ESP8266 that is a low-cost Wi-Fi microchip incorporating both a full TCP/IP stack and microcontroller capability. It is introduced by manufacturer Espressif Systems. The ESP8266 NodeMcu is a complex device, which combines some features of the ordinary Arduino board with the possibility of connecting to the internet. Arduino Modules and Microcontrollers have always been a great choice to incorporate automation into the relevant project. But these modules come with a little drawback as they don't feature a built-in WiFi capability, subsequently, we need to add external WiFi protocol into these devices to make them compatible with the internet channel. This is the famous NodeMCU which is based on ESP8266 WiFi SoC. This is version 3 and it is based on ESP-12E (An ESP8266 based WiFi module). NodeMCU is also an open-source firmware and development kit that helps you to prototype your IOT product within a few LUA script lines, and of course you can always program it with Arduino IDE. In this article, we will try present useful details related to this WiFi Development Kit, its main features, pinout and everything we need to know about this module and the application domain.

4.3.1.2. Parts of NodeMCU

Multiple GPIO pins on the board allow us to connect the board with other peripherals and are capable of generating PWM, I2C, SPI, and UART serial communications. The interface of the module is mainly divided into two parts including both Firmware and Hardware where former runs on the ESP8266 Wi- Fi SoC and later is based on the ESP-12 module. The firmware is based on Lua A scripting language that is easy to learn, giving a simple programming environment layered with a fast scripting language that connects you with a well-known developer community

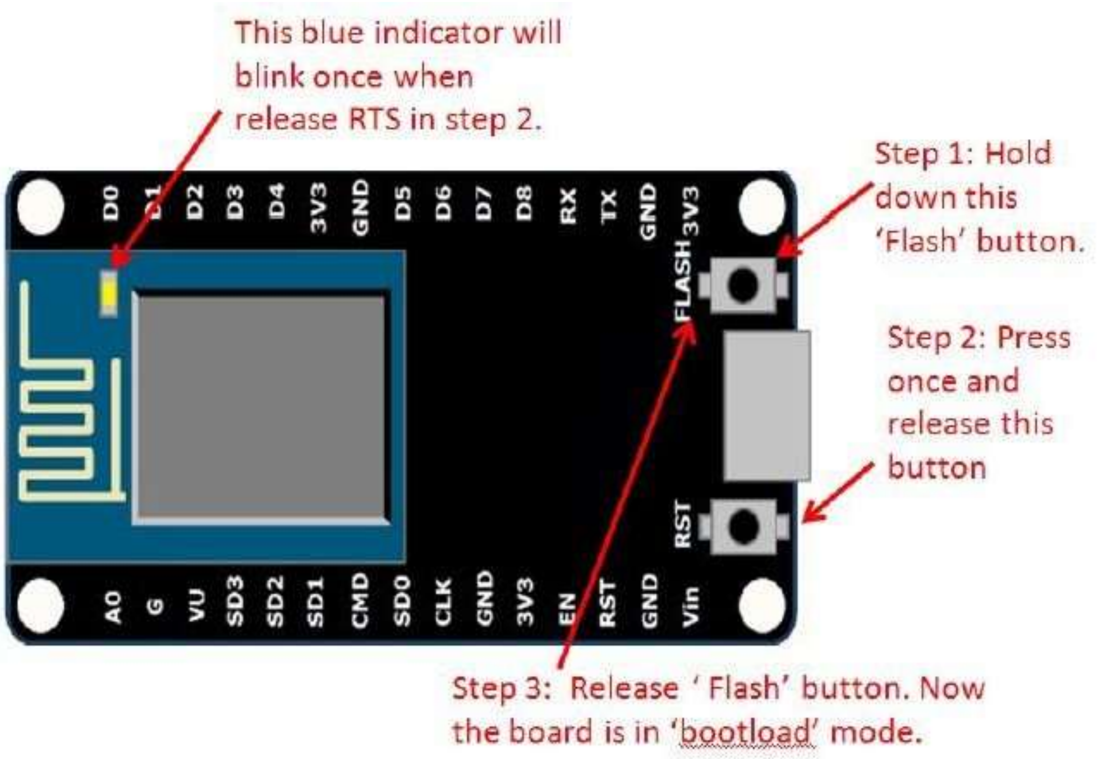

Figure 5: Main parts of NodeMCU

And open source firmware gives you the flexibility to edit, modify and rebuilt the existing module and keep changing the entire interface until you succeed in optimizing the module as per your requirements. USB to UART converter is added on the module that helps in converting USB data to UART data which mainly understands the language of serial communication. Instead of the regular USB port, MicroUSB port is included in the module that connects it with the computer for dual purposes: programming and powering up the board. The board incorporates status LED that blinks and turns off immediately, giving you the current status of the module if it is running properly when connected with the computer.

The ability of module to establish a flawless WiFi connection between two channels makes it an ideal choice for incorporating it with other embedded devices like Raspberry Pi

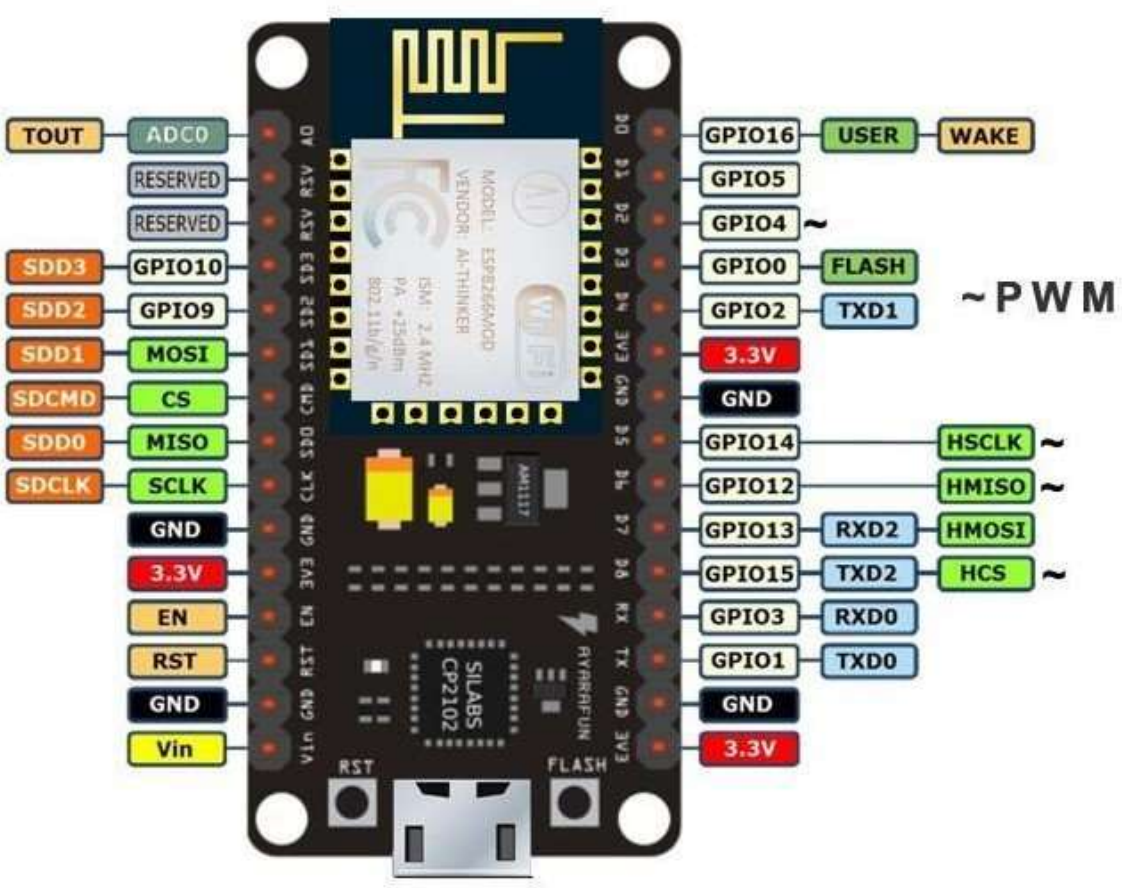

Figure 6: pins of NodeMCU

Figure 3: pins of NodeMCU

NodeMCU V3 Pinout NodeMCU V3 comes with a number of GPIO Pins. Following figure shows the Pinout of the board. There is a candid difference between Vin and VU where former is the regulated voltage that may stand somewhere between 7 to 12 V while later is the power voltage for USB that must be kept around 5 V.

4.3.1.3.Ultrasonic sensor

The ultrasonic sensor works on the principle of sonar and radar system which is used to determine the distance to an object. An ultrasonic sensor generates high-frequency sound (ultrasound) waves. When this ultrasound hits the object, it reflects as echo which is sensed by the receiver as shown in below figure

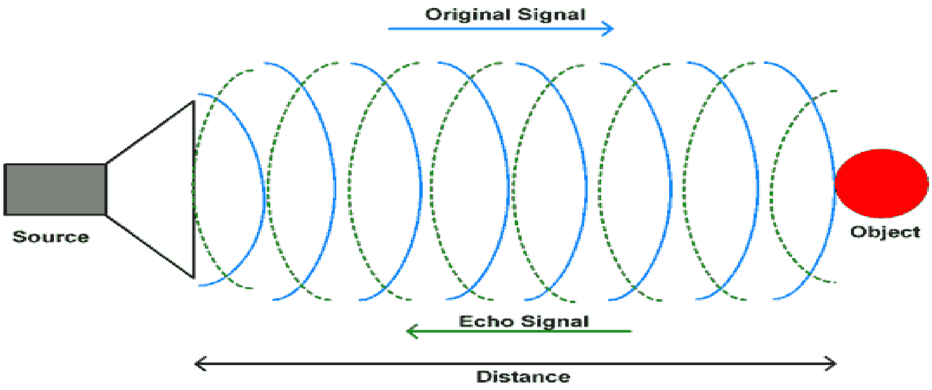

Figure 7: Ultrasonic Sensor Working Principle

By measuring the time required for the echo to reach to the receiver, we can calculate the distance. This is the basic working principle of Ultrasonic module to measure distance.

Figure 8: Ultrasonic Module

In the ultrasonic module HCSR04, we have to give trigger pulse, so that it will generate ultrasound of frequency 40 kHz. After generating ultrasound i.e. 8 pulses of 40 kHz, it makes echo pin high. Echo pin remains high until it does not get the echo sound back. So the width of echo pin will be the time for sound to travel to the object and return back. Once we get the time we can calculate distance, as we know the speed of sound.

Figure 9: HC-SR04 Pin Diagram

Figure 6: HC-SR04 Pin Diagram

VCC: +5 V supply

TRIG: Trigger input of sensor. Microcontroller applies 10 us trigger pulse to the HC-SR04 ultrasonic module.

ECHO: Echo output of sensor. Microcontroller reads/monitors this pin to detect the obstacle or to find the distance.

GND: Ground

The ultrasonic sensor in this project is used to measure oil levels by emitting high-frequency sound waves. It calculates the distance to the oil surface based on the time it takes for the sound waves to bounce back. This non-contact method provides accurate real-time oil level data, ensuring the air compressor has acceptable lubrication, which is critical for efficient operation and maintenance.

Use Ultrasonic Sensors

Ultrasonic sensors can detect a variety of materials, regardless of shape, transparency, or color. The only requirement for ultrasonic sensing is that the target material is a solid or liquid. This enables contactless detection of metal, plastic, glass, wood, rocks, sand,oil, water and other hard, non-sound absorbent materials

These materials are able to reflect sound back towards the sensor through the air. Certain objects can be more difficult to detect, like angled surfaces that direct the echo away from the sensor, or permeable targets like sponge, foam, and soft clothing. These absorb more reflected ultrasonic energy.

4.3.1.4. Liquid Crystal Display (LCD)

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over segments and other multi segmented. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special &even custom characters (unlike in seven segments), animations and so on. A 16x2 LCD means it can display 16 characters per line and there are two such lines. This LCD has two registers, namely, Command and Data. The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD

Figure 10: Liquid Crystall Display

An LCD (Liquid Crystal Display) serves as the primary interface for communicating critical information to the driver. The LCD can display real-time feedback on the driver's alcohol level, indicating whether it is safe to drive or if the vehicle is immobilized due to alcohol detection. It can also show alert messages, such as warnings to stop the vehicle, instructions to retry the breathalyzer test, or system status updates for example an Alcohol Detected or Engine Disable. The clear visual output provided by the LCD enhances user interaction, making the system more intuitive and informative for the drive.

The LCD in this project provides real-time display of key data like oil level, temperature, and system status, ensuring easy monitoring, quick alerts, and user-friendly control for efficient machine operation.

4.3.1.5. AC Output Voltage Sensor

Active output single-phase AC voltage transformer module has an onboard precision op amp circuit, the signal for precise sampling and appropriate compensation function, it used a measure the 250V AC voltage corresponding to the analog output that can be adjusted.

Figure 11: Voltage sensor

ZMPT101B Voltage Module Pinout, the ZMPT101B module has 4 pins: **VCC**: Module power supply: 5 V

GND: Ground

OUT: Module output which is analog

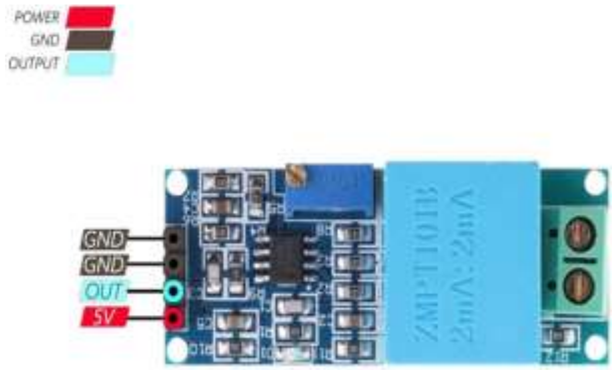

Figure 12: Voltage sensor pins

The AC Output Voltage Sensor monitors the voltage supplied to the air compressor, ensuring it operates within safe limits. It detects irregularities, helping prevent damage from overvoltage or under voltage conditions.

4.3.1.6. Vibration Sensor

This module features a highly sensitive vibration sensor and an on-board voltage comparator for producing a digital output signal. When the vibration switch is under closed conduction state, the output signal is low and the green light is on. When the vibration switch is disconnected, the output signal is high and the green light is turned off. It can be directly connected to a microcontroller for reading the output level and to easily determine the sensor state. The output of the sensor indicates whether vibration was detected in the environment.

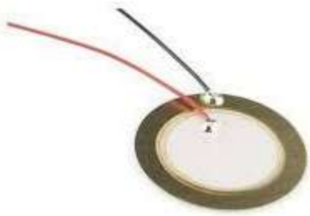

Figure 13: Vibration pins

The vibration sensor monitors the air compressor's vibration levels, detecting abnormal patterns that indicate potential mechanical issues. It helps in early fault detection, improving maintenance and preventing equipment failure.

4.3.1.7. Temperature Sensor

LM35 is a temperature measuring device having an analog output voltage proportional to the temperature.

It provides output voltage in Centigrade (Celsius). It does not require any external calibration circuitry. The sensitivity of LM35 is 10 mV/degree Celsius. As temperature increases, output voltage also increases.

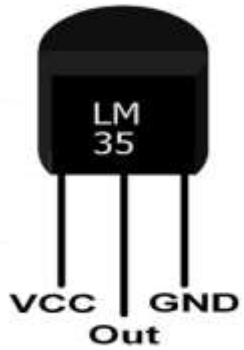

Figure 14: Temperature Sensor

It is a 3-terminal sensor used to measure surrounding temperature ranging from -55 °C to 150 °C. LM35 gives temperature output which is more precise than thermistor output. The temperature sensor monitors the air compressor's temperature in real-time, detecting overheating or abnormal fluctuations. This data helps prevent potential damage, ensures safe operation, and contributes to predictive maintenance by identifying issues early, improving the compressor's overall performance and longevity.

4.3.1.7. Buzzer

Buzzers are audio signaling devices that, when activated, emit sound. In Internet of Things applications and embedded devices, they are commonly used for alerts or warnings. It typically operates on electrical signals and comes in two varieties: active and passive. An active buzzer has an oscillating circuit built into it that, when energized, produces a continuous sound, whereas a passive buzzer requires an additional signal to make noise. When specific events occur, such as successful transactions, system failures, or attempts by unauthorized individuals to gain access, the buzzer will be used in my project to sound a warning. By ensuring that operators or users receive a response right away, this enhances the system's responsiveness and usability. Attached to the buzzer, the NodeMCU microcontroller controls its operation based on predetermined settings.

Figure 15: Buzzer

4.3.1.8. PCB

The PCB (Printed Circuit Board), an essential element of hardware in the Internet of Things, connects and supports all hardware components, including the NodeMCU microcontroller, temperature sensor, vibration sensor, LCD, buzzer, and battery. It provides a well-organized layout for the electrical connections between different components, guaranteeing appropriate power and signal flow. The PCB also simplifies wiring, improves system stability, and reduces errors. It provides the structure for a small and orderly system assembly, which makes it easier to integrate components for efficient operation.

Figure 16: PCB

4.3.2. Experimental data

To gather data for this study, the researcher uses ultrasonic, vibration, temperature, and AC output voltage sensors on industrial air compressors in different industries across Rwanda. These sensors measure critical parameters such as oil level, machine vibration, temperature, and voltage supply to monitor the compressor's performance. The system, enhanced with machine learning algorithms, analyzes the data to predict maintenance needs and optimize the air compressor's operation. Based on sensor data, the system provides insights to prevent breakdowns and improve operational efficiency, ensuring the safe and efficient functioning of the compressors.

4.3.3. Software requirements

Language libraries, Code editors, test monitor, Programming tools.

4.3.4. Cloud platform

The cloud platform provides distant data storage, analysis, and processing as well as remote synchronism of hardware and software system. The platform used here receives data from IoT sensors, stores them, process them and analyses them. In this case sensor data is sent using NodeMCU microcontroller to the dashboard through WiFi network.

4.3.Circuit Diagram

The hardware architecture of a smart industrial machine control system is depicted in the circuit diagram. Connecting to numerous sensors and peripherals for monitoring and control, the NodeMCU functions as the main microcontroller. While a vibration sensor keeps an eye out for unusual vibrations, a temperature sensor is connected to detect the machine's temperature. A voltage sensor guarantees steady power supply monitoring, and an ultrasonic sensor is integrated for proximity detection or liquid level monitoring.

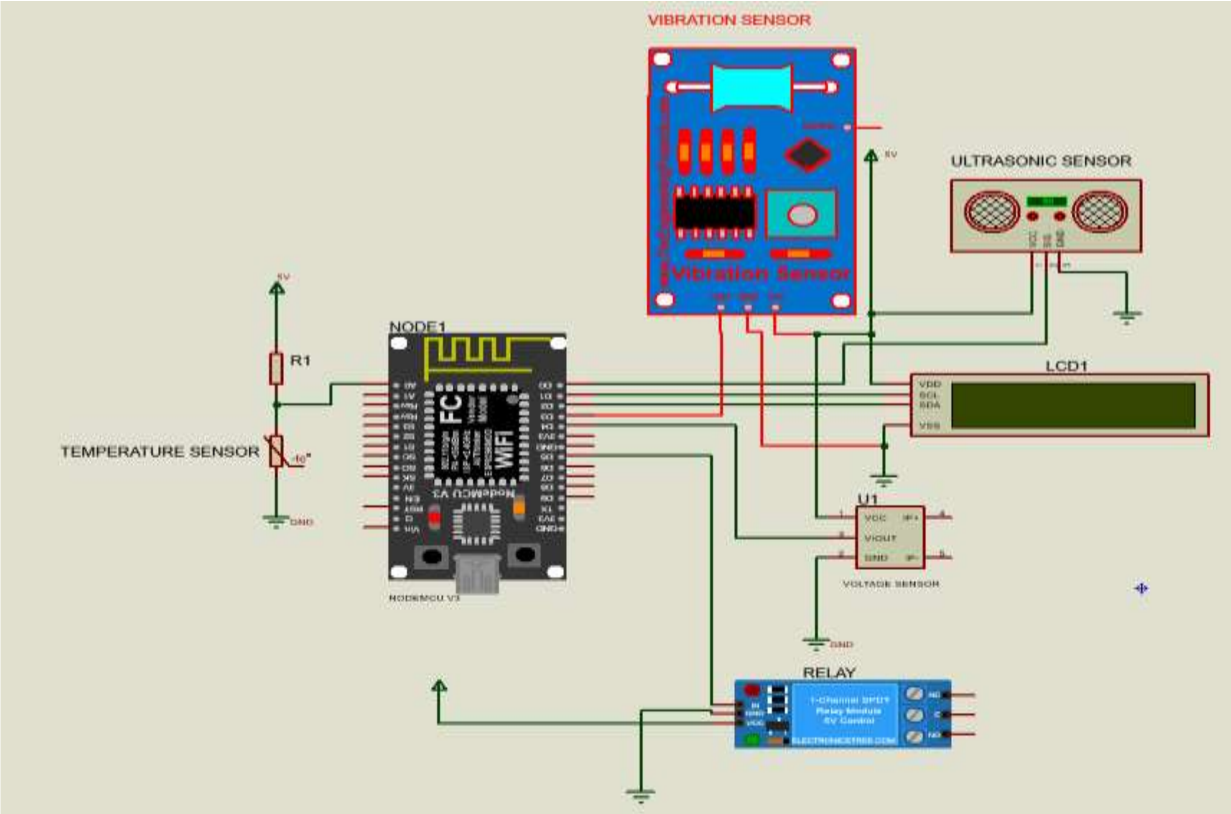

Figure 17: Circuit Diagram

An LCD is used by the system to visualize data in real time, displaying variables including voltage levels, temperature, and vibration status. In order to enable automatic shutdown in dangerous situations, a relay module is integrated to operate external devices such as a compressor or other machinery. All of the sensor outputs are connected to the GPIO pins of the NodeMCU, and the components are powered by a 5V supply. I2C facilitates communication between the microcontroller and the LCD, guaranteeing effective pin utilization. The robust design for real-time monitoring and control shown in this graphic improves operational efficiency and safety in industrial settings.

4.4.Software architecture

In order to handle sensor data, interface with a cloud-based dashboard, and apply safety control logic, the software architecture of the smart industrial machine control and monitoring system was created. The NodeMCU microcontroller is configured to read and interpret data from vibration, temperature, oil level, and voltage sensors using the Arduino IDE. The measurements are then compared to predetermined safety criteria. The NodeMCU sends the processed data to a cloud-based server via Wi-Fi connectivity, where it is shown on a dashboard for both historical analysis and real-time monitoring. In order to help detect any faults before they happen, the dashboard integrates machine learning algorithms for predictive maintenance. In order to prevent damage and guarantee safety, the control logic makes sure that when abnormal conditions are recognized, the program activates a relay to shut down the machine. To improve the dependability, effectiveness, and safety of industrial processes, this design integrates automatic response mechanisms, cloud-based analysis, and local processing.

CHAPTER 5: SYSTEM RESULTS AND ANALYSIS

5.1 Introduction

This chapter describes the compressor monitoring system prototype's implementation, presenting the findings and performance analysis using graphical representations. Three crucial compressor parameters voltage, temperature, and vibration are automatically monitored by the system for anomalies or variations. The system isolates compressor operation within 2 seconds and sends real-time data to the cloud over Wi-Fi if any of these parameters depart from typical operating ranges. A buzzer gives an on-site alert, and a vibration sensor also picks up on odd activities, including possible equipment tampering. In order to enable preventative maintenance and minimize downtime, machine learning algorithms are incorporated into the system to evaluate sensor data and forecast probable failures based on past trends. The system keeps monitoring without sending more data to the cloud if no anomalies are found. The NodeMCU microcontroller processes and gathers data from the voltage, temperature, and vibration sensors used in the system's architecture. Using PHP, HTML, CSS, and JavaScript, a web-based application offers a user-friendly interface for visualizing data in real time. The machine learning model is trained and improved for better failure detection and prediction using both historical and real-time data stored in the MySQL database. Through predictive insights, this technology improves dependability and guarantees effective compressor monitoring.

5.2 System coding and testing

The NodeMCU microcontroller, the central component of the system, is integrated with sensors, actuators, and gateway devices. The Arduino IDE software was used to create the programming codes and upload them to the NodeMCU hardware. This microcontroller makes it easier to collect data from sensors, processes it, and uploads it to the cloud for analysis and monitoring. The technology ensures smooth integration with cloud storage and web-based apps by utilizing Wi-Fi connectivity for real-time data transfer.

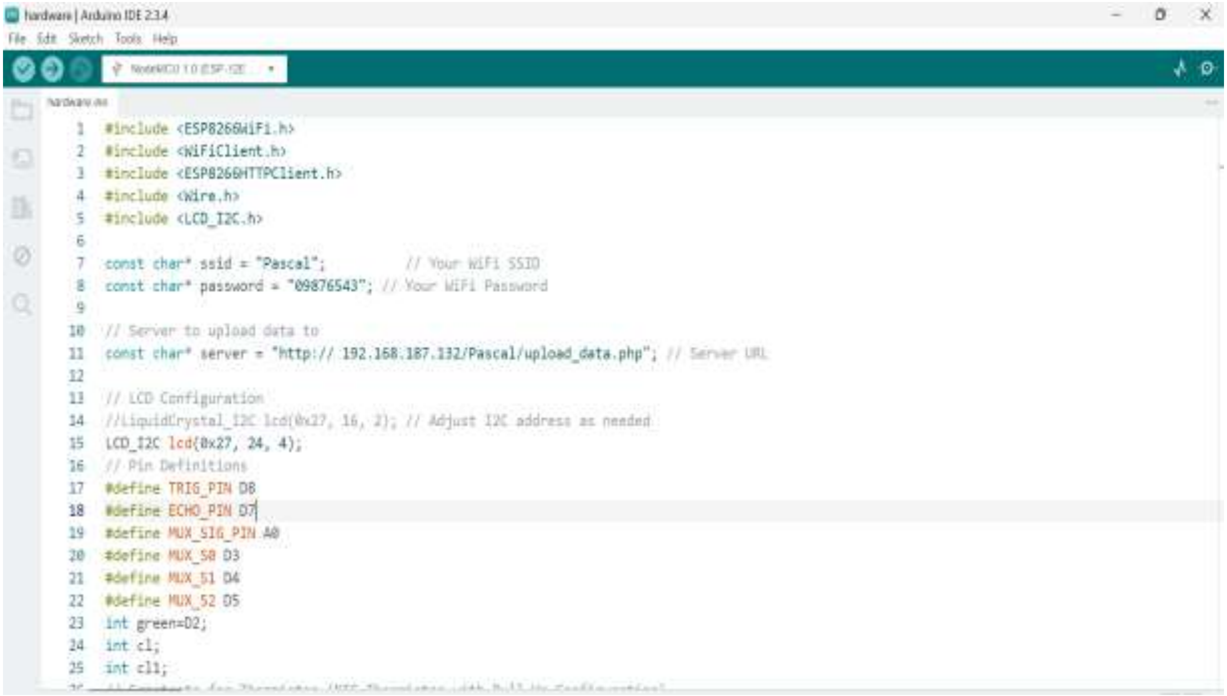

Figure 18: Uploading code in Nodemcu using Arduino IDE.

This hardware setup is a prototype for an integrated industrial process control system designed to monitor and optimize the performance of an industrial air compressor. It includes a microcontroller (Nodemcu) as the central processing unit, an ultrasonic sensor for measuring distance or fluid levels, a piezoelectric sensor for detecting vibrations, and a voltage sensor for monitoring electrical input or system voltage, which is crucial for identifying power anomalies. A 16x2 LCD display provides real-time feedback on parameters such as voltage, vibration, and system status, while the perfboard organizes the components for prototyping. The sensors gather operational data, which the microcontroller processes and displays, enabling real-time monitoring, fault detection, and predictive maintenance to enhance the compressor's reliability and efficiency.

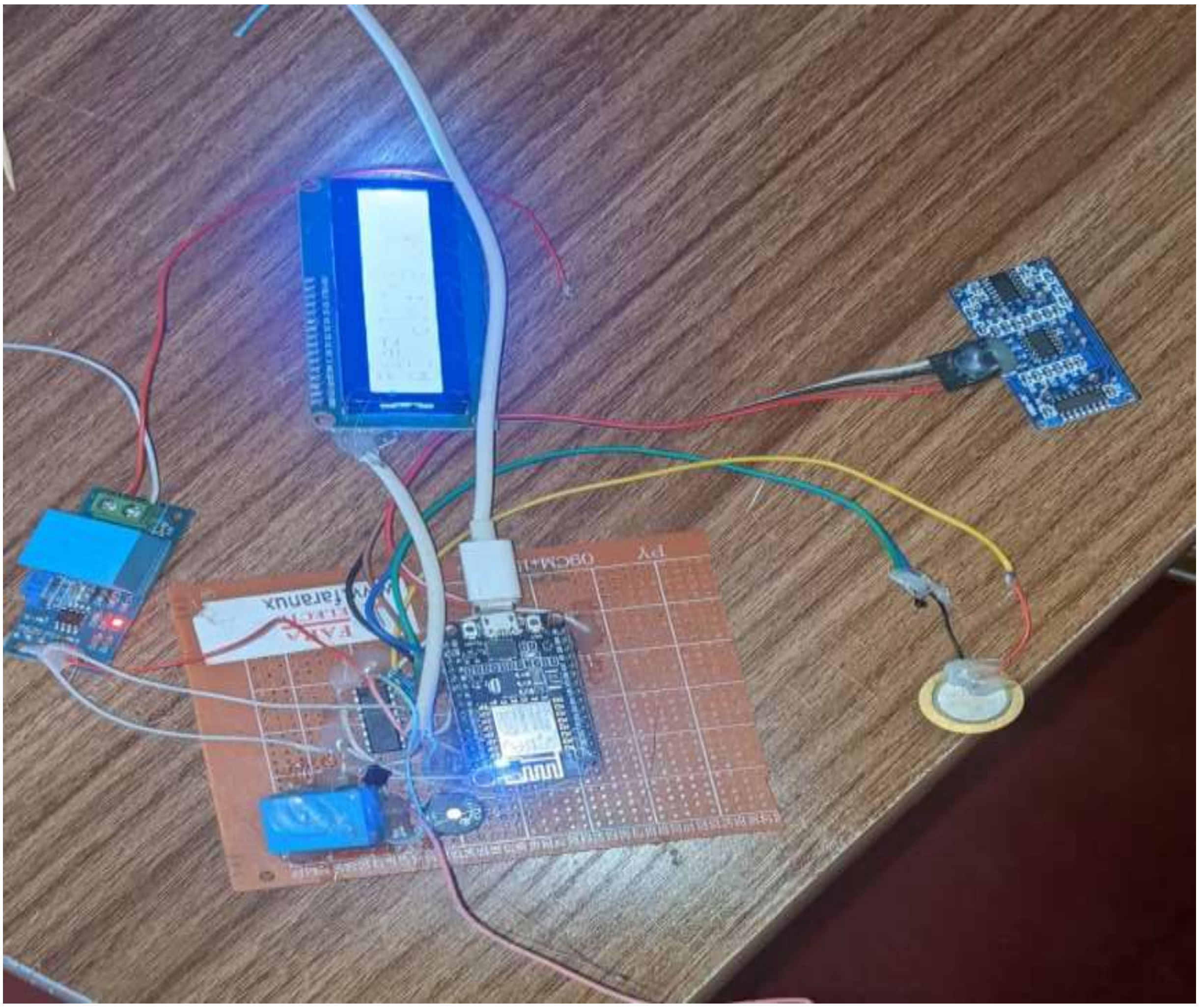

Figure 19: Hardware implementation

This hardware shows a system using a NodeMCU microcontroller, an ultrasonic sensor, voltage sensor, vibration sensor, temperature sensor and an LCD module. The display indicates vibration status ("Inactive") and monitored voltage (203V). The system processes sensor data and can transmit it to the cloud for remote monitoring. It's a prototype designed for industrial applications like fault detection and parameter monitoring.

Figure 20: Lcd displaying data

This hardware prototype integrates an ESP32 microcontroller, sensors (voltage, ultrasonic, temperature, and vibration), an LCD for real-time data display, and a relay module for control. Mounted on a perfboard, it monitors key parameters like voltage, distance, and vibrations, enabling predictive maintenance and efficient control of industrial systems like air compressors.

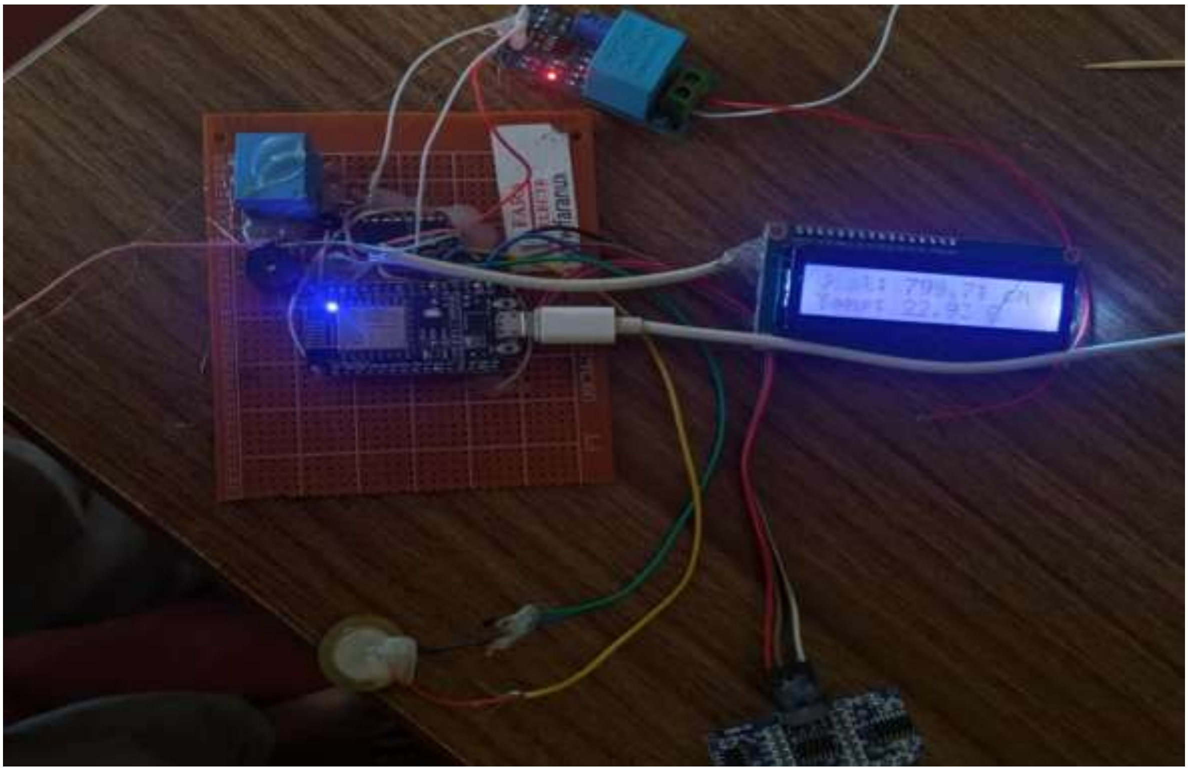

Figure 21: Lcd displaying data

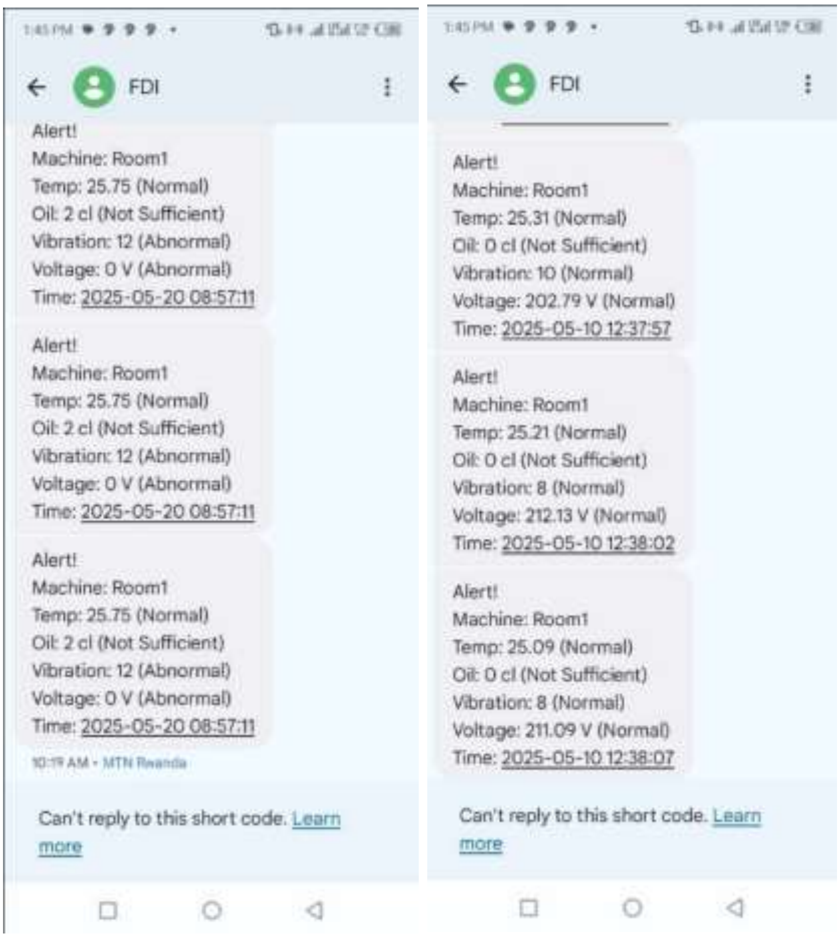

Figure 22: The message alerts to the operator

5.3 System Dashboard

The IoT Sensor Data Report provides a detailed log of key parameters for monitoring an industrial system over time. It includes temperature values in °C, oil levels in centiliters, system vibration status (active or inactive), and voltage readings in volts, all recorded with precise timestamps. This structured data enables operators to track system performance, identify trends, and address potential issues proactively. The "Back to Dashboard" button allows easy navigation to the main dashboard for an overview. By offering real-time and historical data, the report enhances decision-making, supports predictive maintenance, and ensures optimal system functionality, making it an essential tool for effective industrial management.

Figure 23: Data on Dashboard.

This IoT Sensor Data Report dashboard provides real-time monitoring and historical data for an industrial system, likely an air compressor, displaying key parameters such as temperature, oil level, vibration status, voltage, and timestamps. Each parameter includes a status indicator (e.g., Normal, Abnormal, Sufficient, Not Sufficient) to highlight operational conditions and potential issues. For instance, abnormal vibration and voltage statuses, coupled with insufficient oil levels, point to possible mechanical or electrical faults requiring immediate attention. The timestamped entries allow for precise tracking and troubleshooting of anomalies, making this dashboard a critical tool for proactive maintenance and ensuring the system's reliability and efficiency.

IoT Sensor Data Report

Back to Dashboard

Temperature (°C)	Status	Oil Level (Cl)	Status	Vibration Status	Status	Voltage (V)	Status	Timestamp
34.5	Normal	140	Sufficient	Abnormal	Abnormal	198	Abnormal	2024-12-11 11:04:52
34.5	Normal	140	Sufficient	Normal	Normal	198	Abnormal	2024-12-11 11:04:02
34.5	Normal	140	Sufficient	Abnormal	Abnormal	198	Abnormal	2024-12-11 11:03:38
40	Normal	79	Not Sufficient	Normal	Normal	224	Normal	2024-12-11 10:49:46
40	Normal	79	Not Sufficient	Abnormal	Abnormal	198	Abnormal	2024-12-11 10:48:16
42	Abnormal	79	Not Sufficient	Abnormal	Abnormal	198	Abnormal	2024-12-11 10:47:50
42	Abnormal	79	Not Sufficient	Abnormal	Abnormal	198	Abnormal	2024-12-11 10:43:43
36.5	Normal	136	Sufficient	Abnormal	Abnormal	227	Normal	2024-12-10 17:33:03
36.5	Normal	136	Sufficient	Abnormal	Abnormal	227	Normal	2024-12-10 17:32:52
36.5	Normal	134	Sufficient	Abnormal	Abnormal	228	Normal	2024-12-10 17:32:06
35.5	Normal	130	Sufficient	Normal	Normal	228	Normal	2024-12-10 17:31:26
35.5	Normal	130	Sufficient	Normal	Normal	228	Normal	2024-12-10 17:30:59
35.5	Normal	130	Sufficient	Normal	Normal	228	Normal	2024-12-10 17:30:52
34.5	Normal	140	Sufficient	Abnormal	Abnormal	223	Normal	2024-12-10 17:30:11

Figure 24: Data on report form.

CHAPTER 6: CONCLUSION AND RECOMMENDATION

6.1. Conclusion

The project model was effectively created and put into use to track and evaluate the three main components of an industrial compressor system: vibration, temperature, and voltage. The solution guarantees dependable and effective operation by combining machine learning algorithms for predictive maintenance, Wi-Fi for smooth communication, and IoT-enabled sensors for real-time data collection. In order to prevent damage, the technology immediately shuts down the compressor within two seconds after detecting abnormal conditions including voltage fluctuations, overheating, or extreme vibration. By analyzing past data, finding trends, anticipating possible malfunctions, and optimizing maintenance plans, machine learning enables the system to drastically cut down on maintenance expenses and downtime.

Adopting such intelligent monitoring systems is strongly advised for Rwanda's industrial sector in order to improve the dependability, security, and effectiveness of vital machinery. Compressors and other heavy machinery used in manufacturing, mining, construction, and agriculture are frequently subject to routine inspection and maintenance. Common issues like unplanned equipment breakdowns, resource waste, and expensive maintenance can be resolved by integrating this IoT-based system. Predictive analysis powered by machine learning may help industries even more by guaranteeing prompt actions, lowering operational hazards, and extending the life of equipment.

In order to make this system even more adaptable and suitable for a range of industrial requirements, further study could broaden its reach to monitor other factors like oil level, acoustic analysis, or energy use. Pilot projects in Rwandan industry would confirm the system's efficacy and offer insightful input for future improvement.

6.2. Recommendations

Based on the findings and challenges encountered during the implementation of this project, several recommendations are proposed for both future research efforts and the university's academic support systems:

6.2.1. Recommendations for future research

Future studies should consider expanding the scope of intelligent industrial monitoring systems to include additional parameters such as acoustic analysis, oil quality monitoring, energy consumption, and compressor pressure levels. This will enhance system accuracy and cover a broader range of fault types.

- ✓ Researchers are encouraged to explore the integration of edge computing to reduce dependence on cloud services and improve real-time responsiveness, especially in remote or bandwidth-limited industrial sites.
- ✓ More advanced machine learning models such as deep learning e.g., LSTM or CNN could be investigated for complex pattern recognition and long-term trend analysis in compressor behavior.
- ✓ Collaboration with industrial partners for pilot testing and data collection in live environments is essential to validate and optimize the developed models under real-world conditions.

6.2.2. Recommendations for the University

- ✓ The university should invest in more practical facilities, equipment, and modern lab resources to support applied research in IoT, embedded systems, and machine learning. The lack of sufficient sensors, development boards, and real world testing platforms limited the depth and realism of this study.
- ✓ It is recommended that practical workshops and hands on sessions be integrated more deeply into the curriculum to help students bridge the gap between theory and real world applications.
- ✓ Establishing partnerships with local industries would enable students to access real data, gain practical experience, and better align their projects with current technological needs.
- ✓ The university should also support interdisciplinary collaboration by encouraging students from ICT, electronics, mechanical engineering, and data science to work together on industrial automation and intelligent systems.

REFERENCES

[1] Frank Lamb, "Industrial Automation," 2021.

[2] J. S. G. S. K. K. Kumar and P. K. Gupta, "Industrial IoT," 2021.

[3] S. R. Mohanty and A. Kumar, "Predictive Maintenance in Smart Factory," 2022.

[4] Khan, M. A., Machine Learning for IoT, 2021.

[5] Mohanty, S. R., & Kumar, A., Predictive Maintenance in Smart Factory, 2022.

[6] Lamb, F, Industrial Automation, 2021.

[7] Khan, M. A., Machine Learning for IoT, 2021.

[8] Gohil, P., Hands-On Industrial Internet of Things, 2021.

[9] IoT and Edge Computing for Architects, Lea, P., 2020.

[10] Schwab, K., Responding to the Impact of Artificial Intelligence on Business and Society, 2021.

[11] Mohanty, S. R., & Kumar, A, Predictive Maintenance in Smart Factory, 2022.

[12] Chhabra, J. K, Data Science for IoT and Industrial Application, 2023.

[13] Kumar, R, Artificial Intelligence for Industry 4.0, 2021.

[14] Michael Johnson, David Anderson, "Predictive Maintenance in Industrial Systems: IoT Applications and Case Studies," pp. 150-168, 2019.

[15] Sarah Smith, John Doe, "Advanced Industrial Automation: IoT and Machine Learning Approaches," 2020.

[16] Luis Garcia, Maria Thompson, "Industrial IoT: Automation and Data Analytics," 2018.

[17] Minh Nguyen, Susan Clark, "IoT in Industrial Health Monitoring: Techniques and Applications," pp. 112-128, 2021.

[**18**] Robert Lee, Emily White, "Emerging Technologies and Applications," Industrial Security and IoT, 2020.

[19] Anil Patel, Kevin Brown, "IoT and Machine Learning for Industry," Smart Maintenance, 2019.

APPENDIXES

1. Hardware code

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266HTTPClient.h>
#include <Wire.h>
#include <LCD_I2C.h>
const char* ssid = "Pascal";          // Your WiFi SSID
const char* password = "09876543"; // Your WiFi Password
// Server to upload data to
const char* server = "http://192.168.235.132/Pascal/upload_data.php"; // Server URL
// LCD Configuration
//LiquidCrystal_I2C lcd(0x27, 16, 2); // Adjust I2C address as needed
LCD_I2C lcd(0x27, 24, 4);
// Pin Definitions
#define TRIG_PIN D8
#define ECHO_PIN D7
#define MUX_SIG_PIN A0
#define MUX_S0 D3
#define MUX_S1 D4
#define MUX_S2 D5
int green=D2;
// Constants for Thermistor (NTC Thermistor with Pull-Up Configuration)
// Constants for Thermistor (NTC Thermistor with Pull-Up Configuration)
int ThermistorChannel = 2;
float R_PULLUP = 10000.0;
float c1 = 1.009249522e-03, c2 = 2.378405444e-04, c3 = 2.019202697e-07;
// Constants for AC Voltage Sensor
int ACVoltageChannel = 0;    // AC voltage sensor on channel 4
float VREF = 5.0;           // Reference voltage of ADC (NodeMCU operates at 5V)
```

```
// Constants for Piezoelectric Sensor
int PiezoChannel = 1;       // Piezoelectric sensor on channel 1
int piezoValues[100];
float Veff = 0; // Calculated AC Voltage
// Variables
long duration;
float distance;
float temperature;
float piezoValue;
float outputValue;
int buzzer= D6;
int relay=D0;
// Multiplexer Control
void selectMuxChannel(int channel) {
  digitalWrite(MUX_S0, channel & 1);
  digitalWrite(MUX_S1, (channel >> 1) & 1);
  digitalWrite(MUX_S2, (channel >> 2) & 1);}
void setup() {
  // Initialize Serial Monitor
  Serial.begin(115200);
pinMode(buzzer,OUTPUT);
pinMode(relay,OUTPUT);
pinMode(green,OUTPUT);
  // Initialize Pins
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(MUX_S0, OUTPUT);
  pinMode(MUX_S1, OUTPUT);
  pinMode(MUX_S2, OUTPUT);
digitalWrite(relay,HIGH);
  // Initialize LCD
  lcd.begin();
  lcd.backlight();
  lcd.clear();
```

```
  // Connect to WiFi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");}
  Serial.println("Connected to WiFi");}
void loop() {
  // Read Ultrasonic Sensor
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  duration = pulseIn(ECHO_PIN, HIGH);
  distance = duration * 0.034 / 2;
  // Read Piezoelectric Sensor
  selectMuxChannel(PiezoChannel);
  piezoValue = analogRead(MUX_SIG_PIN);
  // Read Thermistor
  selectMuxChannel(ThermistorChannel);
  int analogValue = analogRead(MUX_SIG_PIN);
float Vout = analogValue * (5.0 / 1023.0);
  float R_thermistor = R_PULLUP * ((5.0 / Vout) - 1.0);
//  float logR = log(R_thermistor);
//  temperature = (1.0 / (c1 + c2 * logR + c3 * logR * logR * logR)) - 273.15;
// Calculate temperature for PTC thermistor using adjusted Steinhart-Hart equation
//float logR = log(R_thermistor);
//temperature = (1.0 / (c1 - c2 * logR + c3 * logR * logR * logR)) - 273.15;
// Calculate temperature for PTC thermistor (linear approximation)
//float temperature = (R_thermistor - 1000) / 50.0; // Adjust based on thermistor specs
float temperature = (R_thermistor - 10000) / 2000 + 25;
  // Read AC Voltage Sensor
  selectMuxChannel(ACVoltageChannel);
  int maxReading = 0;
```

```
  for (int i = 0; i < 100; i++) {
    int reading = analogRead(MUX_SIG_PIN);
    piezoValues[i] = reading > 511 ? reading : 0;
    if (piezoValues[i] > maxReading) {
      maxReading = piezoValues[i]; }
    delay(1); // Stabilization delay }
  if (maxReading != 0) {
    float Vmax = maxReading * (VREF / 1023.0);
    Veff = Vmax / sqrt(2);
  } else {
    Veff = 0;  }
  outputValue = (Veff - 0.5) * 100; // Adjust calibration as needed
if(outputValue>220){
  outputValue=outputValue-70; }
  else{outputValue=0;}
  // Display data on LCD
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Dist: ");
  lcd.print(distance);
  lcd.print(" cm");
  lcd.setCursor(0, 1);
  lcd.print("Temp: ");
  lcd.print(temperature);
  lcd.print(" C");
  delay(2000);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Piezo: ");
  lcd.print(piezoValue);
  lcd.setCursor(0, 1);
  lcd.print("AC Volt: ");
  lcd.print(outputValue);
  delay(2000);
```

```
if(outputValue>240||outputValue<190||temperature>40){
  digitalWrite(buzzer,HIGH);
  digitalWrite(relay,LOW);
    digitalWrite(green,HIGH);}
else{ digitalWrite(buzzer,LOW);
 digitalWrite(green,LOW);
digitalWrite(relay,HIGH);}
  // Prepare data to send
  String postData = "device_id=Compressor&";
  postData += "location=Room1&";
  postData += "distance=" + String(distance) + "&";
  postData += "temperature=" + String(temperature) + "&";
  postData += "vibration_status=" + String(piezoValue) + "&";
  postData += "voltage=" + String(outputValue);
  // Send data to the server
  if (WiFi.status() == WL_CONNECTED) {
    WiFiClient client;
    HTTPClient http;
    http.begin(client, server);
    http.addHeader("Content-Type", "application/x-www-form-urlencoded");
    int httpResponseCode = http.POST(postData);
    if (httpResponseCode > 0) {
      Serial.println("Data sent successfully!");
      Serial.println(httpResponseCode);
    } else {
      Serial.println("Failed to send data!");
      Serial.println(httpResponseCode);  }
    http.end();
  } else {
    Serial.println("WiFi Disconnected"); }
  // Delay for next loop iteration
  delay(600); // Send data every 60 seconds}
```

2. VS code for report

```php
<?php
include 'db.php';

// Fetch the first 100 records from the sensor_readings table, ordered by the latest time
$sql = "SELECT * FROM sensor_readings ORDER BY created_at DESC LIMIT 50";
$result = $conn->query($sql);

// Function to get AI model predictions
function getPrediction($data) {
    $api_url = 'http://127.0.0.1:5000/predict'; // Update with your API endpoint

    $ch = curl_init($api_url);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_POST, true);
    curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($data));
    curl_setopt($ch, CURLOPT_HTTPHEADER, ['Content-Type: application/json']);

    $response = curl_exec($ch);

    if (curl_errno($ch)) {
        echo 'Curl error: ' . curl_error($ch);
        return ['prediction' => 'Error']; // Handle API failure
    }

    curl_close($ch);

    // Debugging: Log the response to check its structure
    file_put_contents('debug_log.txt', print_r($response, true), FILE_APPEND);

    $decoded_response = json_decode($response, true);
    return $decoded_response ?? ['prediction' => 'Unknown'];
}
```

```
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>IoT Sensor Data Report</title>
    <style>
        body { font-family: Arial, sans-serif; background-color: #f4f4f9; margin: 0; padding: 0; }
        header { background: #35424a; color: white; text-align: center; padding: 20px; font-size:
24px; }
        .container { width: 80%; margin: auto; padding: 20px; }
        table { width: 100%; border-collapse: collapse; margin: 20px 0; font-size: 18px; }
        table th, table td { padding: 12px; border: 1px solid #ddd; text-align: left; }
        table th { background: #35424a; color: white; }
          a { text-decoration: none; color: white; background: #4CAF50; padding: 10px 15px;
border-radius: 5px; display: inline-block; }
        .status-normal { color: green; font-weight: bold; }
        .status-abnormal { color: red; font-weight: bold; }
        .status-sufficient { color: green; font-weight: bold; }
        .status-insufficient { color: red; font-weight: bold; }
        .status-unknown { color: grey; font-weight: normal; }
         .back-button { background-color:  #4CAF50; color: white; padding: 10px 20px; border-
radius: 5px; text-decoration: none; display: inline-block; margin-bottom: 20px; }
        .back-button:hover { background-color: #4CAF50; }
        .logout-button {
        float: right;
        background-color: red;
        color: white;
        padding: 10px 15px;
        border-radius: 5px;
        text-decoration: none;
        margin-bottom: 10px;
```

```
    }
    .logout-button:hover {
        background-color: darkred;
    }

    </style>
</head>
<body>
<header>
    IoT Sensor Data Report
    <a href="logout.php" class="logout-button">Logout</a>
</header>

    <div class="container">
    <a href="dashboard.php" class="back-button">Back to Dashboard</a>
        <table>
            <thead>
                <tr>
                    <th>Temperature (°C)</th>
                    <th>Status</th>
                    <th>Oil Level (Cl)</th>
                    <th>Status</th>
                    <th>Vibration Status</th>
                    <th>Status</th>
                    <th>Voltage (V)</th>
                    <th>Status</th>
                    <th>Timestamp</th>
                </tr>
            </thead>
            <tbody>
                <?php while ($row = $result->fetch_assoc()) {
                    // Prepare data for AI prediction
                    $sensor_data = [
                        'temperature' => $row['temperature'],
```

```
        'distance' => $row['distance'],
        'vibration_status' => $row['vibration_status'],
        'voltage' => $row['voltage']
    ];

    $prediction_response = getPrediction($sensor_data);
    $prediction = $prediction_response['prediction'] ?? 'Unknown';

    // Determine the temperature status
    if ($row['temperature'] >= 20 && $row['temperature'] <= 30) {
        $temperature_status = 'Normal';
        $temperature_class = 'status-normal';
    } else {
        $temperature_status = 'Abnormal';
        $temperature_class = 'status-abnormal'; }
    // Determine the oil level status
    if ($row['distance'] < 100) {
        $oil_level_status = 'Sufficient';
        $oil_level_class = 'status-sufficient';
    } else {
        $oil_level_status = 'Insufficient';
        $oil_level_class = 'status-insufficient';    }
    // Determine vibration status
    $vibration_status = $row['vibration_status'] >6 ? 'Abnormal' : 'Normal';
   $vibration_class = ($row['vibration_status'] >6) ? 'status-abnormal' : 'status-normal';
    // Determine voltage status
    $voltage_status = $row['voltage'] < 1.5 ? 'Abnormal' : 'Normal';
    $voltage_class = ($row['voltage'] < 1.5) ? 'status-abnormal' : 'status-normal'; ?>
    <tr>
        <td><?php echo $row['temperature']; ?></td>
        <td class="<?php echo $temperature_class; ?>">
            <?php echo $temperature_status; ?>
        </td>
        <td><?php echo $row['distance']; ?></td>
```

```
                <td class="<?php echo $oil_level_class; ?>">
                    <?php echo $oil_level_status; ?>
                </td>
                <td><?php echo $vibration_status; ?></td>
                <td class="<?php echo $vibration_class; ?>">
                    <?php echo $vibration_status; ?>
                </td>
                <td><?php echo $row['voltage']; ?></td>
                <td class="<?php echo $voltage_class; ?>">
                    <?php echo $voltage_status; ?>
                </td>
                <td><?php echo $row['created_at']; ?></td>
            </tr>
        <?php } ?>
        </tbody>
    </table>
</div>
</body>
</html>
```

3. VS code for model training

```
import pandas as pd
from sqlalchemy import create_engine
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
import joblib
# Database connection using SQLAlchemy
engine = create_engine("mysql+mysqlconnector://root:@localhost/iot_data")
# Load data from database
query = "SELECT temperature, distance, vibration_status, voltage FROM sensor_readings"
df = pd.read_sql(query, engine)
```

```python
# Define abnormal conditions
conditions = (
    (df['temperature'] > 40) |
    (df['distance'] > 100) |
    (df['vibration_status'] == 1) |
    (df['voltage'] < 200) | (df['voltage'] > 230))
df['status'] = conditions.astype(int)  # 1 for Abnormal, 0 for Normal
# Prepare features and labels
X = df[['temperature', 'distance', 'vibration_status', 'voltage']]
y = df['status']
# Normalize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
# Train model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
# Save model and scaler
joblib.dump(model, 'status_model.pkl')
joblib.dump(scaler, 'scaler.pkl')
print("Model training completed and saved as 'status_model.pkl'")
```

4. VS code for Dashboard

```php
<?php
// session_start();
// if (!isset($_SESSION['user_id'])) {
//     header("Location: index.php");
//     exit();
include'db.php';
// Fetch the latest sensor data
$sql = "SELECT * FROM sensor_readings ORDER BY created_at DESC LIMIT 1";
$result = $conn->query($sql);
$data = $result->fetch_assoc();
```

```
// Set default values if no data is found
$temperature = $data['temperature'] ?? 0;
$distance = $data['distance'] ?? 0;
$vibration_status = $data['vibration_status'] ?? 0;
$voltage = $data['voltage'] ?? 0;
$timestamp = $data['created_at'] ?? 'N/A';
// Conditions for abnormality
$temp_condition = ($temperature > 40) ? 'Abnormal' : 'Normal';
$oil_condition = ($distance < 100) ? 'Not Sufficient' : 'Sufficient';
$vibration_condition = ($vibration_status> 6) ? 'Abnormal' : 'Normal';
$voltage_condition = ($voltage >= 200 && $voltage <= 230) ? 'Normal' : 'Abnormal';
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Industrial Air Compressor Dashboard</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/chartjs-plugin-datalabels"></script>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f9; }
        header {
            background: #35424a;
            color: white;
            text-align: center;
            padding: 20px;
            font-size: 24px;
            position: relative;}
        .logout-button {
```

```css
    position: absolute;
    top: 15px;
    right: 20px;
    background: red;
    color: white;
    padding: 10px 15px;
    border-radius: 5px;
    text-decoration: none; }
.container {
    width: 80%;
    margin: auto;
    padding: 20px; }
.gauge-container {
    display: flex;
    justify-content: space-around;
    flex-wrap: wrap; }
.gauge {
    width: 300px;
    height: 200px;
    margin: 10px;    }
.table-container {
    margin-top: 14px;}
table {
    width: 100%;
    border-collapse: collapse;
    margin: 20px 0;
    font-size: 18px;
    text-align: left; }
table th, table td {
    padding: 12px;
    border: 1px solid #ddd; }
table th {
    background: #35424a;
    color: white;}
```

```
        .status-normal {
            color: green;
            font-weight: bold; }
        .status-abnormal {
            color: red;
            font-weight: bold; }    a {
            text-decoration: none;
            color: white;
            background: #4CAF50;
            padding: 10px 15px;
            border-radius: 5px; }
    </style>
</head>
<body>
    <header>
        Industrial Air Compressor Dashboard
        <a href="logout.php" class="logout-button">Logout</a>
    </header>
    <div class="container">
        <div class="gauge-container">
            <div class="gauge">
                <canvas id="temperatureGauge"></canvas>
            </div>
            <div class="gauge">
                <canvas id="oilLevelGauge"></canvas>
            </div>
            <div class="gauge">
                <canvas id="voltageGauge"></canvas>
            </div>
        </div>
        <div class="table-container">
            <a href="report.php">View Full Report</a>
            <h2>Latest Sensor Data</h2>
            <table>
```

```
            <tr>
                <th>Device ID</th>
                <td><?php echo $data['device_id'] ?? 'N/A'; ?></td>
            </tr>
            <tr>
                <th>Machine</th>
                <td><?php echo $data['location'] ?? 'N/A'; ?></td>
            </tr>
            <tr>
                <th>Temperature (°C)</th>
                <td><?php echo $temperature; ?></td>
                <td class="<?php echo ($temp_condition === 'Normal') ? 'status-normal' : 'status-abnormal'; ?>">
                    <?php echo $temp_condition; ?>
                </td>
            </tr>
            <tr>
                <th>Oil Level (cl)</th>
                <td><?php echo $distance; ?></td>
                <td class="<?php echo ($oil_condition === 'Sufficient') ? 'status-normal' : 'status-abnormal'; ?>">
                    <?php echo $oil_condition; ?>
                </td>
            </tr>
            <tr>
                <th>Vibration Status</th>
                <td><?php echo $vibration_status; ?></td>
                  <td class="<?php echo ($vibration_condition === 'Normal') ? 'status-normal' : 'status-abnormal'; ?>">
                    <?php echo $vibration_condition; ?>
                </td>
            </tr>
            <tr>
                <th>Voltage (V)</th>
```

```
                <td><?php echo $voltage; ?></td>
                   <td class="<?php echo ($voltage_condition === 'Normal') ? 'status-normal' :
'status-abnormal'; ?>">
                   <?php echo $voltage_condition; ?>
                </td>
            </tr>
            <tr>
                <th>Timestamp</th>
                <td colspan="2"><?php echo $timestamp; ?></td>
            </tr>
        </table>
    </div>
  </div>

  <script>
    const gaugeConfig = (label, max, value, title) => ({
        type: 'doughnut',
        data: {
            labels: [label, 'Remaining'],
            datasets: [{
                data: [value, max - value],
                backgroundColor: ['#4CAF50', '#E0E0E0'],
                borderWidth: 0
            }]
        },
        options: {
            circumference: 180,
            rotation: 270,
            cutout: '80%',
            plugins: {
                legend: { display: false },
                tooltip: { enabled: false },
                datalabels: {
                    display: true,
```

```
                color: '#000',
                font: { size: 30, weight: 'bold' },
                formatter: (value, context) => context.chart.data.datasets[0].data[0]
              },
              title: {
                display: true,
                text: title,
                font: { size: 16 }
              }
            }
          }
        });

        new Chart(document.getElementById('temperatureGauge'), gaugeConfig('Temperature',
100, <?php echo $temperature; ?>, 'Temperature (°C)'));
        new Chart(document.getElementById('oilLevelGauge'), gaugeConfig('Oil Level', 400,
<?php echo $distance; ?>, 'Oil Level (Cl)'));
        new Chart(document.getElementById('voltageGauge'), gaugeConfig('Voltage', 240,
<?php echo $voltage; ?>, 'Voltage (V)'));
    </script>
</body>
</html>
```