ASHESI UNIVERSITY

DESIGN AND IMPLEMENTATION OF A SMART GRAIN STORAGE MONITORING SYSTEM

CAPSTONE PROJECT

B.Sc. ELECTRICAL AND ELECTRONIC ENGINEERING

Wendinoda Chivambo and Rahinatu Mohammed

2022

ASHESI UNIVERSITY

DESIGN AND IMPLEMENTATION OF A SMART GRAIN STORAGE MONITORING SYSTEM

CAPSTONE PROJECT

Capstone Project submitted to the Department of Engineering, Ashesi University College in partial fulfilment of the requirements for the award of Bachelor of Science degree in Electrical and Electronic Engineering.

Wendinoda Chivambo and Rahinatu Mohammed

2022

DECLARATION

We hereby declare that this capstone is the result of our own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signatures:

Candidate's Names:

Date:

I hereby declare that preparation and presentations of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University College.

Supervisor's Signature:

Supervisor's Name:

Date:

Acknowledgments

Thank you to everyone who encouraged us and gave us academic guidance for the duration of the project. To begin, we would like to offer our profound gratitude to Mr. Kofi Adu-Labi, our project supervisor, for his advice and unwavering support throughout the entire project. His suggestions and feedback aided in the development of the project's scope. We would also like to express our gratitude to Dr. Nathan Amankwah and the engineering department faculty for giving constructive feedback and ideas during progress update presentations. The feedback and suggestions served as a guide in identifying areas to improve on. Finally, we would like to express our gratitude to our families and friends for their support and prayers.

Abstract

Grain post-harvest losses due to deterioration during storage remains a prevalent challenge. The changes in environmental conditions if not monitored, can cause fluctuations in grain storage bins' temperature and humidity, leading to decay and infestation. Even though automatic grain monitoring systems have been developed they are not affordable. This project seeks to design an efficient and low-cost smart grain monitoring system to reduce unnecessary grain losses. To monitor grain conditions, the system employs temperature, humidity, and carbon dioxide sensors. Furthermore, this system employs machine learning classification algorithms to predict grain quality status (good or bad) based on sensor readings. The project implements the system prototype with 2 sensor nodes that communicate to a gateway through HC-12 transceiver modules.

Table of Contents

Acknowledgments ..... **ii**
Abstract ..... **iii**
Chapter 1: Introduction **1**
1.2 Problem Definition 1
1.3 Objectives 2
1.4 Project Outcomes 3
1.5 Motivation 3
1.6 Scope of Project 4
1.7 Research Methodology 4
Chapter 2: Literature Review **5**
2.1 Introduction to Smart Grain Monitoring Systems 5
2.2 Existing Solutions 6
2.3 Related work 9
Chapter 3: Design **11**
3.1 Design Proposal 11
3.2 Design Assumptions 11
3.3 Design Requirements 11
3.3.1 User Requirements 11
3.3.2 Functional Requirements 12
3.4 Design Decisions 12
3.4.1 Sensor Decisions 12
3.4.2 Microcontroller Decision 13
3.4.3 Wireless Technology 14
3.4.4 Wireless Topology 15
3.5 System Design Architecture 16
3.5.1 IOT 16
3.5.1.1 Placement of Sensor Nodes 16
3.5.1.4 Overview HC12 Transceiver Module 19
3.6 Machine Learning 20
3.6.1 Model deployment 20
Chapter 4: Methodology **22**
4.1 Carbon dioxide Sensor Calibration 22
4.2 Experimental Setup 23
4.3 Experimental data analysis 25
4.3 Data Labelling 29

4.4 Classification Algorithms 30
4.4.1 Logistic Regression 30
4.4.2 Random Forest 31
4.4.3 K-Nearest Neighbour 32
4.4.4 Gradient Boosting 33
4.5 Metrics Considered 33
4.6 Cross Validation 34
4.7 Learning Curves 35
4.8 Circuit Designs 36
4.8.1 Receiver (Gateway) node 36
4.8.2 Sensor Nodes Circuit Designs 36
4.9 Range Test 39
4.10 Software Design of Sensor Nodes 39
4.11 Software Design for the Gateway 40
4.12 Network architecture 40
Chapter 5: Results 41
5.1 Results from the experiment 41
5.2 Grain Quality Classification 42
5.2.1 Feature correlation matrix 42
5.2.2 Results from testing the Logistic Regression Classification 43
5.2.3 Results from testing the Random Forest Classification 44
5.2.4 Results from testing the Gradient Boosting Classification 46
5.2.5 Results from testing the KNearest Neighbor Classification 47
5.3 Comparison of classifier accuracy scores 48
5.3. 1 The (ROC) Curve 48
Chapter 6: Conclusion 52
6.1 Discussion 52
6.2 Limitations 54
6.3 Future Work 55
References 57

List of Figures

Figure 2.1: Schematic diagram of SDWSN

Figure 3.1: Solid works design of a standard silo

Figure 3.2: Solid works design showing the placement of sensor nodes and gateway placement

Figure 3.3: Block diagram of sensor node

Figure 3.4: Block diagram for gateway

Figure 3.5: Block diagram for model deployment

Figure 4.1: Electrical circuit for MQ-135 sensor

Figure 4.2: Grain simple with added moisture

Figure 4.3: Control grain sample

Figure 4.4: Database used for experiment picture

Figure 4.5: Plot of Rs/Ro vs temperature based on experimental data (MQ135 Sensor)

Figure 4.6: MQ-135 Rs/Ro vs temperature characteristic behaviors

Figure 4.7: Distribution fit for MQ-135

Figure 4.8: Distribution fit for MG-811

Figure 4.9: Receiver (Gateway) node circuit diagram

Figure 4.10: Sensor node 2 circuit diagram

Figure 4.11: Perforated board implementation of sensor node 2

Figure 4.12: Sensor node 1 circuit diagram

Figure 4.13: Perforated board implementation of sensor node 1

Figure 5.1: Plot of Rs/Ro vs temperature based on experimental data

Figure 5.2: Plot of Rs/Ro vs temperature based on experimental data

Figure 5.3: Feature Correlation Matrix

Figure 5.4: Confusion Matrix for Logistic Regression

Figure 5.5: Learning Curve for Logistic Regression

Figure 5.6: Confusion Matrix for Random Forest

Figure 5.7: Learning Curve for Random Forest

Figure 5.8: Confusion Matrix for Gradient Boosting Classifier

Figure 5.9: Learning Curve for Gradient Boosting

Figure 5.10: Confusion Matrix for KNearest Neighbor

Figure 5.11: Learning Curve for KNearest Neighbor

Figure 5.12: Receiver Operating Characteristic Curve

Figure 5.13: Different classifier accuracy scores

Figure 5.14: Grain classification model deployment website interface

Figure 5.15: Model deployment result pages

Figure 5.16: Node data display at receiver end

List of Tables

Table 3.1: Functional requirements of the system

Table 3.2: Pugh matrix for microcontroller

Table 3.3: Pugh matrix for wireless technology

Table 3.4: Pugh matrix for network topology

Table 4.1: Represents the list of hardware and software components used for the experiment

Table 5.1: Logistic Regression Classification Report before cross validation

Table 5.2 Random Forest Classification Report before cross validation

Table 5.3 Gradient Boosting Classification Report before cross validation

Table 5.4: KNearest Neighbor Classification Report before cross validation

Table 5.6 showing comparison between classifier accuracy scores

Chapter 1: Introduction

The production of grain has been increasing due to the development in production technology and the increase in the population. However, large amounts of these grains are wasted. Reports indicate that food wastage amounts to about one-third of the food produced in many low income and developing countries. Poor storage conditions and lack of appropriate storage facilities contribute considerably to food wastage [1], [2]. The world population continues to grow significantly, and so does the food requirements [3]. Due to the limited land available for crop cultivation, the idea of scaling food production does not entirely ensure food security. Authors in [4] researched the different causes of post-harvest losses in Ghana. They concluded that grain deterioration and pest infestations caused by uncontrolled fluctuating environmental conditions in the storage houses; contribute largely to post-harvest losses. To reduce these losses that occur during storage, grain conditions such as temperature, carbon dioxide content, and humidity should be monitored and possibly controlled. Monitoring and controlling storage facilities have become imperative for farmers and governments because of the need to ensure food security and provide value addition. The safety and quality of grains directly impact the lives of people because grains are an essential resource in many countries. Taiwo and Bart-Plange state that agriculture does not only play in the overall economic performance of Ghana in terms of improving the country's GDP, but it plays a significant role in food security enhancement, poverty alleviation, and employment generation efforts of the country [4]. Despite the vital role that agriculture plays in Ghana's economy, the agriculture sector is still quite untouched by the Information Technology industry [5].

1.2 Problem Definition

The world's population is expected to grow by 9.1 billion by 2050[6]. With such an increase in the population, it is expected that about 70% of additional food would be required to feed this growing population [6], [7]. In order to ensure food security with these ever-

increasing numbers, post-harvest losses should be reduced. Globally about one-third of the world's foods produced are lost during post-harvest processes. In Africa, these losses are estimated to be between the ranges of 20% and 40% of the total food crops produced [8]. According to the world bank report, Sub-Saharan Africa alone loses food grain worth about 4 billion USD yearly to post-harvest losses [8].

Additionally, Ghana loses more than 318,500 tons of maize, rice, and millet annually, to post-harvest losses [9], [2]. Most of these losses occur during storage due to grain decay caused by fluctuating temperatures, moisture, and pest infestation. These losses have economic, social, and environmental impacts. Among the losses that occur during post-harvest processes, storage losses are considered the most critical in developing countries since there is little to no technology in managing losses that occur at this stage of the supply chain. If farmers are given an opportunity to technically and efficiently monitor their grain products in the post-harvest phase. In that case, they can be able to take appropriate steps to prevent probable losses that can be easily avoided. Thus, an intelligent grain monitoring system would prove beneficial in reducing the losses during storage. This capstone project seeks to provide a technical solution to post-harvest losses that occur in storage facilities due to uncontrolled environmental conditions by building a Smart Grain Monitoring System. The system detects environmental conditions in the storage house. The system design incorporates machine learning algorithms to analyse sensor readings and classify grain quality status based on past data gathered.

1.3 Objectives

This project aims to develop a smart grain monitoring system that will reduce unnecessary post-harvest losses that occur during storage. The system will detect changes in grain conditions and communicate with the monitoring centre. Furthermore, the system deploys machine learning algorithms to classify grain status as either good or bad and thus adding more meaning to sensor readings that are displayed to the farmer. The project objectives are

- Design a smart grain condition monitoring system for a granary using IOT
- Compare detected grain conditions against set threshold values
- To minimize food spoilage which leads to potential health risk
- To maintain optimum environmental conditions in grain storage
- To reduce losses that are associated with grain storage
- To be able to predict grain condition using machine learning algorithms

1.4 Project Outcomes

The outcome of the project is an implemented design prototype of the smart grain monitoring system that:

- Detects grain condition parameters
- Transmit the sensor data
- Receive sensor data and display on the system's dashboard
- Can classify grain quality using sensor readings
- Send notifications in form of text messages and alarms

1.5 Motivation

The world's population is estimated to double from 6 billion to 12 billion in about 50 years as of 2009 [11]. This implies that food consumption will increase too. In line with Sustainable Development Goal number 2, which aims to end hunger, achieve food security and improved nutrition, and promote sustainable agriculture [10]. One way of ensuring food security is tackling the issue of post-harvest losses. Maize is a staple crop in most African countries but often, farmers struggle to preserve and store grain after harvest [11] efficiently. In Ghana, small-scale maize producers living in the Ejura-Sekyedumane district of the Ashanti region of Ghana face the challenge of a lack of inefficient grain storage facilities [11]. In addition to that, maintaining food safety and quality is another motivation for this work. Food crops, particularly grain, are essential sources of nutrition in prepared foods.

For this reason, it is vital to store grains in environments with optimum conditions such that irrespective of the season or weather, grains would still be safe for use. As such, this project aims to provide farmers with an affordable grain monitoring system that helps them monitor temperature, carbon dioxide, and humidity levels in grain storage bins. The proposed system goes beyond displaying temperature, humidity, and carbon dioxide levels. It classifies the grain status, good or bad, through machine learning. The system would help retain grain quality while ensuring easy and reliable maintenance of storage barns.

1.6 Scope of Project

The scope of the work entails the mechanical design of granaries that shows optimal sensor node placements in SolidWorks, smart granary achieved by IOT, grain quality classifications through machine learning algorithms deployment, and a dashboard for the display of data at the monitoring centre. For real-world implementation, a series of grain storage barns would be used. We tested the functionalities of the system with a scaled-down version with two sensor nodes found in one silo.

1.7 Research Methodology

The main research technique that is performed is in the form of qualitative and quantitative. The qualitative part was done through literature review of existing designs. This gave an in-depth understanding to the current design trends and research into this area. The quantitative part is in the form of design and testing. With this, different aspects of the project were tested before final implementation.

Chapter 2: Literature Review

2.1 Introduction to Smart Grain Monitoring Systems

In past years, grain monitoring in grain storage was done manually by periodically taking samples of grains and observing their appearance. However, this method is labor intensive, inefficient and inaccurate. It is inaccurate because, during the periodic inspections, only grains at the top would be considered. For these reasons, researchers looked into how monitoring might be switched from manual to automation. With the success of automating grain monitoring operations, research is now underway to transition from wired to wireless automatic systems using relevant technology [12]. The wired systems' principal drawbacks are their limited anti-interference capacity and poor maintenance. These limitations of the wired systems have led many scholars in [12] to investigate the development of wireless sensor networks (WSN). The wireless sensor network is a specific network comprising of a large number of sensor nodes with self-organizing ways that can gather and transmit data through wireless technologies. There have been numerous wireless sensor networks developed and studied by different scholars. Most of these wireless sensor network systems are deployed in Asia, particularly in China, with few in Africa and even lesser in Sub-Saharan Africa. Sensor nodes play an important role in much of the activities relating to monitoring and managing grain storage through the WSN. Generally, the sensor nodes comprise temperature and humidity sensors while few investigations consider carbon dioxide sensors for measuring concentration of carbon dioxide in the silo. In most investigations, temperature and humidity are measured using the DHT11, SHT11, DS18B20, and LM series of sensors. MQ7 and MQ135 sensors are commonly used for CO2 sensing. Sensor nodes are a critical part of the system because they affect cost, detection capability and connectivity issues. The processing sub-system is part of the monitoring system that typically comprises of the microcontroller units and their peripherals. The MCU performs tasks, processes data, and controls other parts of the

system, particularly the sensor nodes. In most of the papers, different modules of the ATMEGA have been used. The standard modules include, ATmega 2560, ATmega 128L, and ATmega 32. In some cases, the Arduino Freedom board was used. Also, the STM32, PIC32, and the different modules of the Node MCU were used. Because these systems are used in storage barns, batteries are used to power the microcontrollers. For most applications, Zigbee communication protocol is used as the medium that connects between the sensor nodes and the cloud services where data is stored and processed. Some other designs used Bluetooth, IEEE 802.11b and the HC12 transceiver modules.

2.2 Existing Solutions

Purandare et al, [13] propose an end-to-end system for farmers and warehouse managers to reduce post-harvest losses. Their system consists of a notification-suggestion system, which provides current farm conditions, and offers suggestions about harvesting time and diseases that may affect the crop in its cultivation stages. The notification-suggestion system acquires data from environmental DHT sensors, communicates with the server for processing thereafter, the sensor readings are displayed on a dashboard. This system analyses previous data of storage losses, harvesting time, crop moisture content at the time of storage and meteorological data. This data is used to train machine learning models that predict grain's optimum temperature of storage and recommended sequence of stock dispatch/outlet. Prediction is done by using statistical and probability techniques such as classification, clustering and regression. The system's design is based on the major assumption that the farmer is unaware of the correct harvesting times which heavily impacts storage losses. The novelty of this system as described in the paper is the use of modern technologies like android, machine learning and internet of things (IOT) to fully automate the monitoring and analysis of preharvest activities up to the post-harvest storage stage. It is commendable that the design proposed is a thorough system that commences grain monitoring while it is still on the farm.

This ensures a highly accurate prediction of harvesting time and the optimum storage environmental conditions and the recommend grain stock dispatch. Moreover, this system establishes a multi-hop protocol for communication between various sensors, transmitters, and receivers in a sensor grid, as well as ensuring efficient power and bandwidth usage.

The ability to estimate the suggested stock dispatch is crucial since traditional farming implies that grain stored first is dispatched first, but this technology proves otherwise. Over the course of five months, each stock is monitored for factors such as warehouse temperature, humidity, deviation from the optimal harvesting time, and corresponding losses, and trustworthy grain dispatch predictions are generated using machine learning, specifically the Elastic Net Regression model. The system makes recommendations for possible actions a farmer or warehouse manager might make, such as grain dispatch sequence, harvesting time, and optimal storage temperatures. However, it would be more beneficial if the farmer/warehouse managers were notified of the repercussions of each action.

Mabrouk et al, [14] propose a new technique to tackle problems of massive grain losses that occur in Egypt due to theft and uncontrolled environmental conditions in storage houses using environmental sensors and level sensors. Sensor nodes collect data and communicate it to the main station via GSM module, where it is displayed on an LCD. The authors review an already existing technical design in place in the country which continuously measures the environmental conditions and grain content levels using satellite nourishes and other complex sensors. The advantages of these systems include high efficiency, but they are very expensive to acquire and require frequent maintenance thus Mabrouk et al propose a new design. According to the authors, grain theft incidents are prevalent in Egypt involving false quantity additions in some storage facilities, which have costed the country millions of dollars. The system suggested in this research would control silo level content by scanning the surface of grain using an ultrasonic sensor linked to a servo motor minimizing fraudulent grain additions

[14]. Environmental conditions are also monitored and compared to pre-set fixed values; if there is a significant divergence for an extended period of time, a heater or fan is turned on. This study has some limitations, such as inaccuracies that can occur due to incorrect sensor placement (for example, if the sensors are placed near the heater, the results may be inaccurate), and dust that can interfere with ultrasonic sensor readings, necessitating frequent cleaning. It is recommended that a radar sensor be used instead of an ultrasonic sensor.

According to Rekha and Singhai [15] temperature, humidity, CO_2 and O_2 are the most important factors that are to be monitored in order to maintain the quality of grains in the storage barns. They noted that temperature increases as the insects produce heat. However, they noted grain deterioration cannot be determined with just temperature because molds can still develop at very close range of temperatures. In addition to that, they also noted that high humidity is the main cause of insect infestation and fungal growth in the silos. Furthermore, they noted that the presence of CO_2 in the storage units indicates grain deterioration thus CO_2 can be used as a predictor in grain storage barns. They suggested other ways in which grain monitoring can be done, these include the use of acoustic sensing, environmental sensing, and odour sensing. In addition, they pointed out that different researchers have studied the effects of temperature, humidity, and CO_2 content on the grain individually. Still, very few of these studies analyzed the impact of simultaneous monitoring of these factors. This was one of the paper's strengths because they provided a comprehensive study on the sensing technologies that could be used to monitor the quality of stored grain. One downside of this study was that the study did not indicate how any of the technologies could be implemented.

Onibonoje et al [12], acknowledges that monitoring of grain storage conditions is necessary because of its economic importance. They also noted that temperature and humidity are the primary factors that affect the quality of grain. The main aim of this study was to design an application-based wireless sensor network capable of providing real time remote monitoring

and automated control of the environmental factors that can affect stored grains in granary systems. The system has a hardware unit, software and a network side. The hardware unit consist of the sensors, XBee transceivers, microcontroller unit, and battery unit. The sensor unit consist of coding the sensor nodes to acquire data from the sensors and networking these with the coordinator. The authors made several conclusions on the different temperature and humidity ranges that would be good for grain storage and the need to modify network protocols for effective connectivity. However, they did not indicate which network protocols will be modified.

2.3 Related work

Studies reviewed noted that even with the great success that has been achieved with the development of monitoring systems that make use of wireless sensor networks, losses associated with grain storage are yet to be reduced. They identified that the development of a WSN comes with its challenges, such as network management and bandwidth, computational capability, and data storage [12]. These challenges are associated with the different devices connected to the single network. Based on these identified challenges, they proposed a different approach that would be used to achieve the same goals while mitigating the challenges of the WSN [12]. The proposed system is a Software Defined Wireless Sensor Network (SDWSN) [12]. Its features include decoupling the data and control plane. Different machine learning algorithms are also deployed to perform grain classification based on sensor readings with this approach. Examples of the algorithms used include K-Nearest Neighbor, Random Forest, and Linear regression. However, the downside of this approach is the increase in costs because of the extra components used.

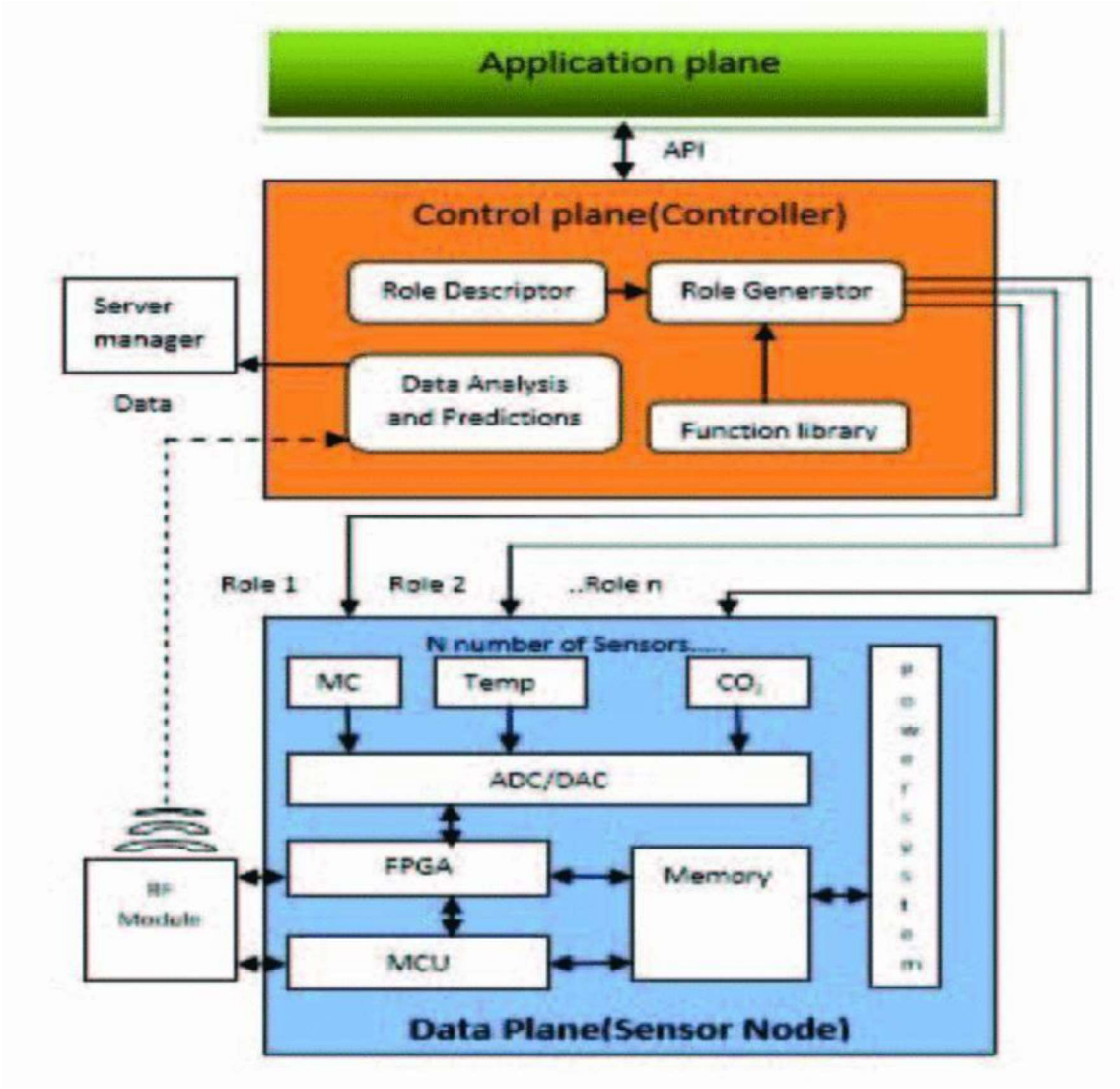

Figure 2.1: Schematic diagram of SDWSN

The following conclusions are drawn from a review of existing solutions in the literature

• The main conditions monitored are temperature and humidity

• The ATmega series of microcontrollers is preferred

• Zigbee communication module is widely used for monitoring purposes

• A battery power source is used implying low power consumption components should be selected.

Chapter 3: Design

3.1 Design Proposal

The proposed solution in this project is a smart grain monitoring system that is affordable and reliably monitors the temperature, humidity, carbon dioxide concentration, and grain level in a silo. This system further incorporates classification algorithms to classify grain quality. This chapter explores the system and user requirements, system specifications, and design decisions that were considered during the implementation of the project

3.2 Design Assumptions

Several assumptions were made after the review of the existing grain monitoring systems. The assumptions that were considered include:

- The full functional system would be complete within the time frame of the capstone session
- The overall design would have relatively low costs compared with the other systems out in the market
- All the components that would be used for the project are easily accessible

3.3 Design Requirements

The design requirements were set to ensure that the design meets the aim and objective of the project. These requirements guided the overall implementation of the project. The requirements are grouped into user/ non-technical requirements and engineering requirements.

3.3.1 User Requirements

- User friendly
- Reliability: The system should work as intended
- Affordable system
- Responsiveness: The system should be able to record changes quickly and accurately in the silo environment.

- Data integrity: The grain classification must be accurate.
- System should alert users of the grain quality status and silo surrounding conditions

3.3.2 Functional Requirements

Table 3.1: Functional requirements of the system

System Requirement	Engineering Requirements
It should be user friendly	• It should have a simple interface
Reliability	• It should intelligently sense data accurately • Interpretate data without error
Affordable	• Use low power consumption devices • Low-cost microcontrollers and sensor
Ability to classify	• System should be intelligent
Sensitivity	• The overall should be sensitive to changes of reading by sensors

3.4 Design Decisions

This section followed detailed processes for choosing components used in this project. The decision criteria were informed by the requirements and expected outcome of the proposed system.

3.4.1 Sensor Decisions

The choice of the sensor is of utmost importance for this system design and full implementation. For the selection of sensor modules, standard sensors were selected. The DHT22 sensor would be used to record the temperature and humidity in the system. This sensor was considered because it can record both temperature and humidity. The DH22 was preferred over the DHT11 because it has a higher temperature range, humidity range, temperature accuracy, and humidity accuracy. It is also relatively reliable and accurate. The ultrasonic sensor module would measure the level of grains in the storage barn. For the carbon dioxide

sensor, the MQ-135 sensor was selected. The MQ-135 sensor was however replaced with the MG811 sensor because of some inconsistences in its carbon dioxide readings. These sensors were selected because they can easily be integrated with the different microcontroller units.

3.4.2 Microcontroller Decision

Microcontrollers considered

- 8051
- AT Mega 328p
- STM32F103C8T6
- Arduino AT Mega 2560

The 8051 microcontrollers from Intel have a smaller memory size and processing speed when compared to ATmega328p-pu and STM32 using ATmega2560 as the baseline. The STM32 microcontroller has the best processing speed in comparison with the other microcontrollers. The selected microcontroller should be able to integrate well with the sensors and the other peripherals to be connected to it. Even though the overall score of the STM32 is the best ATmega328p-pu was selected. This decision was made considering power consumption and the ease of use of the two microcontrollers. The initialization of the STM 32 is cumbersome and time consuming as it involves the peripherals (GPIOs, UARTs and SPIs) and the cortex core itself. This initialization process is also prone to errors. ATmega328p-pu are widely used in Arduino development boards and it has a lower power consumption and high performance.

Table 3.2: Pugh matrix for microcontroller

Criteria	Baseline AT Mega 2560	Weight	A STM32 F103C8T6	B Intel MC3-51 (8051)	C AT Mega 328ppu
Power Consumption	0	5	+3	+1	+2
Cost	0	3	0	+3	+2
Processing speed	0	4	+2	-1	+1
Memory Size	0	4	-1	-2	-3
Total			4	1	2

3.4.3 Wireless Technology

Wireless technology considered

- Zigbee
- nRF24L01
- HC-12
- Wi-Fi • Bluetooth

Zigbee, Nrf24L01, Bluetooth and HC-12 were considered in choosing the wireless technology. Wifi is used as the baseline, and the criteria used for comparison are communication range, data rate, power consumption, cost, and number of devices supported. The power consumption and cost of the wireless technology is required to be the least possible. The data rate is not of much importance as we will be transmitting sensor readings at an interval. The number of devices supported, and communication range are of moderate importance; hence they are all assigned a weight of 3. This decision was necessary because the chosen technology should be able to communicate reliably between the sensor nodes and the gateway. The HC-12 module has the overall best score; hence it is selected. In addition to that, the HC-12 module can support up to 100 devices. In addition to that, it can work on three different modes, which can be used to conserve power

Table 3.3: Pugh matrix for wireless technology

Criteria	Baseline WIFI	Weight	A Zigbee	B nRF24L01	C Bluetooth	D HC-12
Communication Range	0	3	0	+2	-1	+3
Data rate	0	2	0	+1	+1	+2
Power Consumption	0	5	+2	-2	-1	-3
Cost	0	4	-1	+2	+3	+1
Number of devices supported	0	3	-1	-2	-3	0
Total			0	1	-1	3

3.4.4 Wireless Topology

Even though a decision was made for the type of network topology, the network topology depends more on the type of wireless communication. This is because each wireless technology has better performance with a particular network topology. The tree network topology should be used for our chosen wireless communication mode. The tree topology is chosen despite the mesh having a higher value. This was done partly because of the cost associated with implementing the full mesh topology and the choice of the wireless communication module.

Table 3.4: Pugh matrix for network topology

Criteria	Baseline Ring	Weight	A Tree	B Mesh	C Star
Cost	0	4	-1	-2	-3
Security	0	4	+1	+2	+4
Complexity	0	3	-1	-2	-3
Reliability	0	5	+3	+4	+4
Delay	0	3	0	+2	+3
Congestion control	0	2	+2	+1	-1
Total			4	5	4

3.5 System Design Architecture

Based on the system requirements of the system, the proposed solution design is divided into 2 main parts, an IOT system and a Machine learning part.

3.5.1 IOT

The IOT system includes 2 nodes installed, a gateway and a cloud database for storage. The sensor nodes take readings from the sensors attached and transmit it to the gateway using the HC12 communication module. The gateway then sends this data to the database which is displayed on the dashboard.

3.5.1.1 Placement of Sensor Nodes

Grain silos are usually designed as flat bottom silo, smooth wall design silos, and elevated silos. The materials traditionally used for these designs are primarily stainless steel, reinforced steel, steel silos, and Reinforced Cement Concrete. However, for our design we assumed that the silo is made of stainless steel and is the flat bottom design for this design. The geometrical limit for silos is that the ratio of the height to the diameter should be less than 10. With that in consideration, the height was made to be 20m height with a diameter of 15m. A standard silo with this dimension would have three sensor nodes placed at equal intervals around the walls

of the silo. Each sensor node would be placed in a plastic casing that would be bolted to the wall of the silo.

Figure 3.1: Solid works design of a standard silo

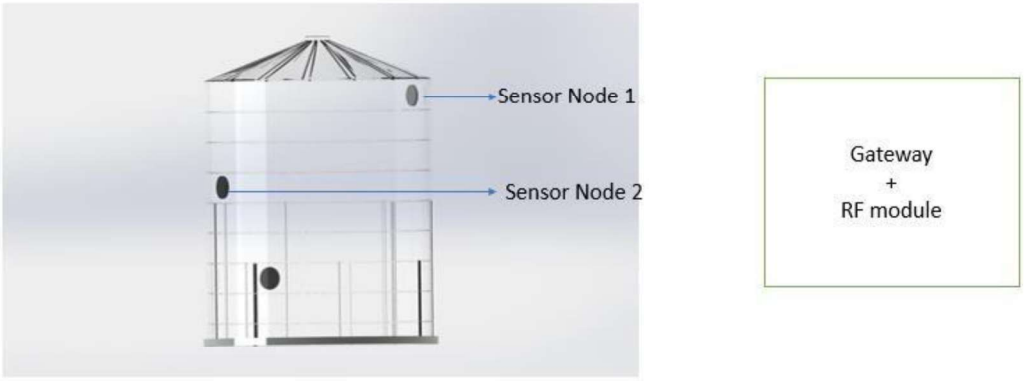

Figure 3.2: Solid works design showing the placement of sensor nodes and gateway placement

3.5.1.2 Sensor Node

The sensor node is the main part of the IOT system. The main job of the sensor node is to receive timely request from the gateway through the HC12 communication unit to collect grain environmental conditions and send the sensor readings to the gateway. The nodes are divided into two nodes, that is node one and node two. Node one comprises of a microcontroller, a temperature and humidity sensor, a carbon dioxide sensor and the level sensor which is powered by a battery. The second node comprises of the temperature and humidity sensor and a carbon dioxide sensor.

The microcontroller at each sensor node is the ATmega 328p. The recommended input voltage is 7-12V with an operating voltage of 5V. The DHT22 and the ultrasonic sensor are connected to digital pins on the board while the MG811 is connected to an analog pin in node one. For node two, the DHT22 and the MG811 are connected to digital pins.

The HC12 used for the communication in this project is operated in the FU3 mode which uses average power consumption when in working conditions [39]. All the HC12 modules at the nodes and gateway had the same transmission mode, wireless communication channel and the same over the air serial baud rate. All sensor nodes are powered by a rechargeable battery power supply and the LM7805 5 volts regulator ensures the power to the microcontroller does not exceed the required.

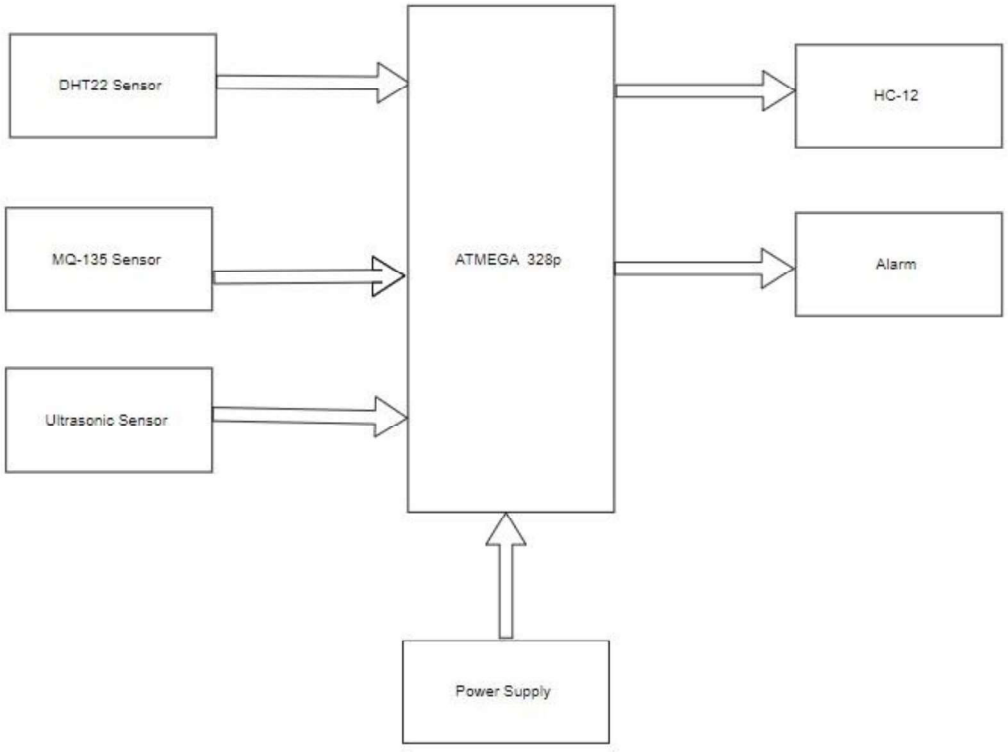

Figure 3.3: Block diagram of sensor node

3.5.1.3 Receiver

The receiver side serves as the gateway for all the sensor nodes. It connects all the nodes to one database system using the HC12 module and the esp32 microcontroller. The esp32 microcontroller is chosen for its WiFi module that can be used to send data to the database. The GSM SIM800L is used for alerting the user of the silo conditions and grain condition. The LCD is installed in the silo bin and displays sensor readings. Lastly the web application displays temperature, humidity and carbon dioxide readings, additionally, classification algorithms are deployed, and these are used for grain condition classification.

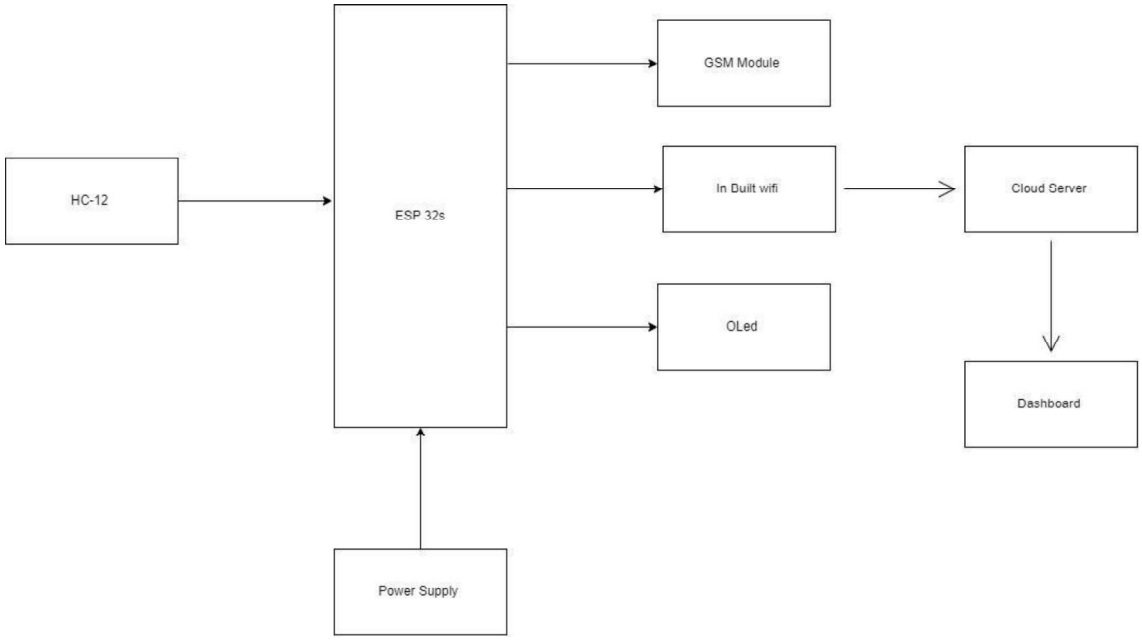

Figure 3.4: Block diagram for gateway

3.5.1.4 Overview HC12 Transceiver Module

The HC12 is a half-duplex wireless serial communication module that is used for long range applications. It has a working frequency band between 433MHz and 473MHz [39]. The transmitting power is between -1dBm(0.79mW) to 20dBm(100mW). Its receiving sensitivity is from -177dBm to -100dBM. This receiving sensitivity and transmitting power depend on the selected Serial and over the-the-air Baud rate. The operating voltage is between 3.2V to 5V

with the preferred voltage for most applications being the 3.3V rate. It has 100 channels and four radio modes. The default mode is the FU3 mode. In order to change any property of the module the AT commands is used. The HC12 module is simple to use because it is compatible with various microcontroller units and works with any digital pins. The module has five pins, the VCC, Ground, TX, RX and the set pin. The set pin is used to change the configurations of the module.

3.6 Machine Learning

The second part of the project involves using machine learning algorithms for the classification of grain quality (good/bad). The system uses supervised machine learning classification algorithms to achieve grain quality classification. The models considered are Random Forest, Logistic Regression, KNearest Neighbours and Gradient Boosting Classifier. The models are trained using data collected from the experiment. The grain features of interest are temperature, humidity and carbon dioxide concentration and the target feature that we wish to predict is grain quality class. The grain is classified as either normal or infested based on sensor readings. Various model performance measurement techniques like cross validation, learning curves and ROC AUC scores are implemented to ensure adequate comparison and choice of the good fit model. The model training is executed in Jupyter Notebook.

3.6.1 Model deployment

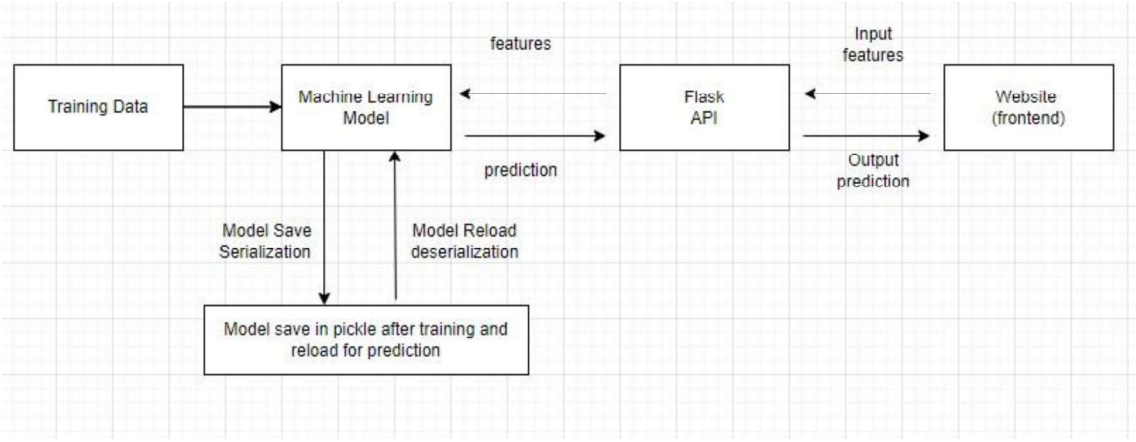

Figure 3.5: Block diagram for model deployment

Figure 3.5 shows the design and steps to deploy the model deployment for consumption. The good fit model that is chosen will be trained with cleaned experimental data. Flask is a web application framework written in Python [16]. The Flask API is a set of rules that allows you to send and receive data from a website [16]. The user interacts with the website by inputting features such as temperature, humidity, and carbon dioxide concentration, which the API then gets. The features will also be fed into the machine learning model, which will classify the grain. The API now receives the forecast, which it sends to the website and displayed for the user. Pickle is used for serializing after the model has been trained, which is the process of writing a Python object to memory in order for it to be deserialized (read) by a Python script object on a memory so that it can be later deserialized (read) back by python script.

Chapter 4: Methodology

In this section, the procedures used in the testing and development of the system are detailed. The calibration of the MQ135 gas sensor is discussed first. Second, the experiment used to collect data for grain quality classification is thoroughly described. Next, this section describes the circuit designs for the sensor (transmitter) and receiver nodes. Fourth, the HC12 transceivers range test and finally, the classification techniques used to classify grain quality are explained.

4.1 Carbon dioxide Sensor Calibration

The MQ-135 sensor is an air quality sensor that detects the presence of different gases like Ammonia (NH_3), Sulfur (S), Benzene (C_6H_6), CO_2 in the air. Despite the sensor's ability to measure different gases in air, it can be calibrated to measure carbon dioxide gas which is the gas of interest for this project. The MQ-135 consists of a surface covered with a thin layer of SnO_2 heater resistor which can raise the temperature of the sensor [17]. Prior to usage the sensor is preheated continuously using a regulated 5V DC power source for 24 hrs. The preheating is done to activate the tin oxide layer found on the surface of the sensor which is useful for gas sensing. Following the preheating process, the sensor was exposed to carbon dioxide gas using exhaled air collected in balloons as explained in [18]. Following that, the sensor was placed in a room with clean air. The value of Ro which is sensor resistance at a particular gas concentration was read and updated in the MQ135.h library. When the sensor is used, it is recommended to wait for approximately 2 minutes to allow the sensor to stabilize before it can read correct data [19].

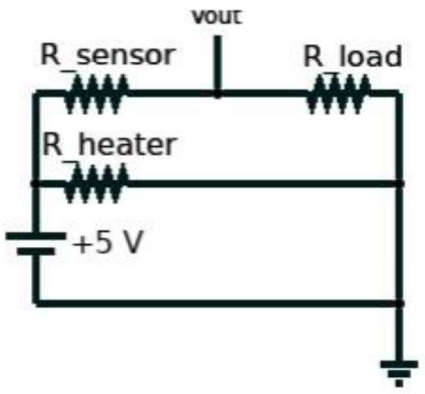

Figure 4.1: Electrical circuit for MQ-135 sensor

4.2 Experimental Setup

An experiment was conducted to obtain the dataset used for training the classification models. For the experiment, two grain samples were used: sample 1 served as the experiment's control, and sample 2 had water added to it to speed up the decaying process. The following were the experiment's parameters:

Mass of maize = 244g for each sample

Mass of water added to second sample = 34.6

Error margin of scale used = +- 0.01g

calculated moisture content = mass of moisture/ mass of maize x 100%

=34.6/244 x 100

= 14.19%

The aim of this experiment was to investigate the temperature, relative humidity and carbon dioxide concentration behaviours of decaying/decayed grain. Water was added to the grain sample 2 thus increasing grain moisture content. High moisture content speeds up the grain deterioration process thus enabling us to collect the necessary decaying grain characteristic data required to perform grain quality classification. Moisture content is the most important physical factor in grain storage because it influences mould growth, which infects all stored grain [20]. It is widely accepted that climatic conditions cause physical changes in stored

grain by increasing moisture, which causes deterioration [20]. The two samples were kept at room conditions. Each sample had 1 DHT22 and 1 calibrated MQ135 sensor connected as shown in Figure 4.2 and Figure 4.3 for measuring temperature, humidity and carbon dioxide concentration. The microcontroller used in the experiment is the esp32-wroom-32d and it was carefully chosen for its WIFI capability. Since this experiment's main aim is to collect data, we utilized the WIFI capability to establish a connection with MYSQL database on the local host where the recorded data is stored. Figure 4.4 below is a picture of the database where the data was stored. Each sample's data was stored in a different table of the database for distinguishing purposes. PHP through visual studio code was used to connect with the previously stated database. After three days, sample 2 with a moisture content of 14.19 percent began to visibly deteriorate; mold growth patches were observed. As the mold developed in the sample with added moisture, its temperature and the humidity levels started to increase. The recorded data from the experiment was exported from the database as an Excel csv file.

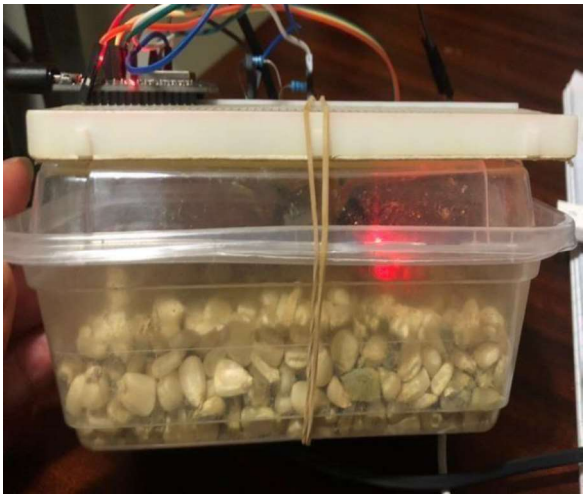

Figure 4.2: Grain simple with added moisture

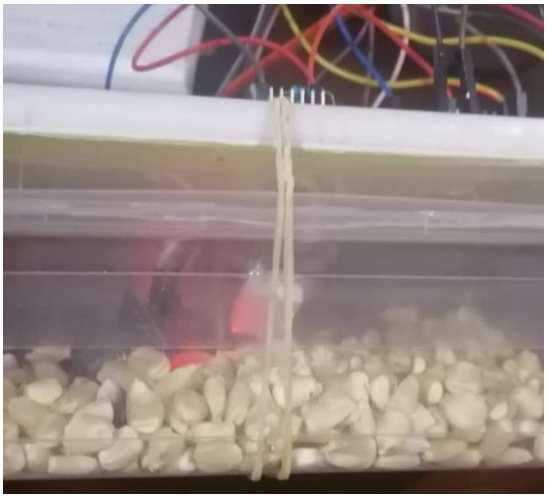

Figure 4.3: Control grain sample

Table 4.1: Represents the list of hardware and software components used for the experiment

Components	Application
Esp32 wroom	Employed its Wi-Fi module to communicate to the data base
Arduino Ide	It was used for taking sensor readings
Visual Studio Code	Connected the Arduino Ide to the database system
DHT22 Sensor	Used to take temperature and humidity readings
MQ-135 Sensor	Used to measure carbon dioxide within the samples
PhpMyAdmin database	Used to store data collected form the sensor nodes.
USB Port of Computer	The setup was powered from the PC through the USB port.

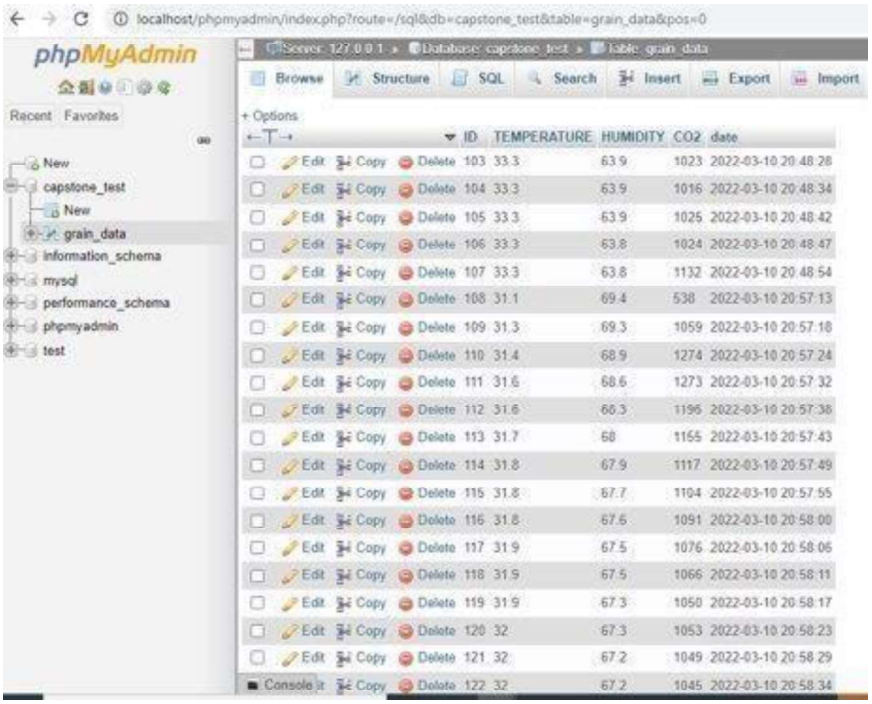

Figure 4.4: Database

4.3 Experimental data analysis

The MQ135 sensor is affected by temperature and humidity changes. Figure 4.6 from the sensor's datasheet shows a plot of Rs/Ro vs temperature for different relative humidity levels [21]. Rs is the sensor resistance at varied concentrations of a given gas, hence Rs/Ro is directly related to CO_2 concentration, as shown by equation 1 below.

$$R_0 = R_S * (1 / A * c)^{-1/B} \quad (1)$$

Where A and B are constants and c is the gas concentration in ppm [20].

As the temperature rises, Rs/ Ro drops, reflecting a decrease in CO_2. The curve for 85% humidity level indicates lower Rs/Ro values reflecting a decrease in CO_2 than the curve for 33% humidity. These curves suggest that higher humidity and temperature level will result in a decrease in the CO_2 sensor readings. As sample 2 started decaying, the temperature increased and the humidity level also increased. Figure 4.5 above shows the graph of Rs/Ro of the experimental data plotted in MATLAB. Rs/Ro increases steadily with temperature until about 34°C after which it starts decreasing and this explains why we recorded lower CO_2 values for the decaying value while we expected increased values.

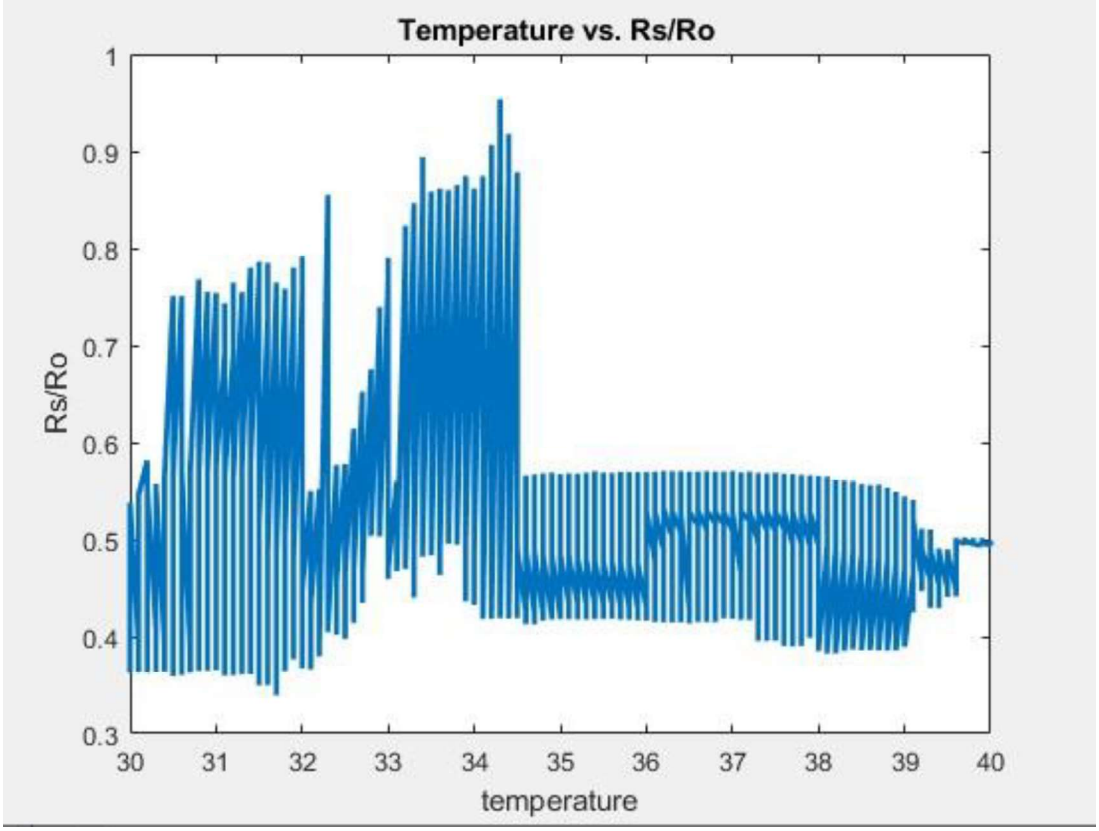

Figure 4.5: Plot of Rs/Ro vs temperature based on experimental data (MQ135 Sensor)

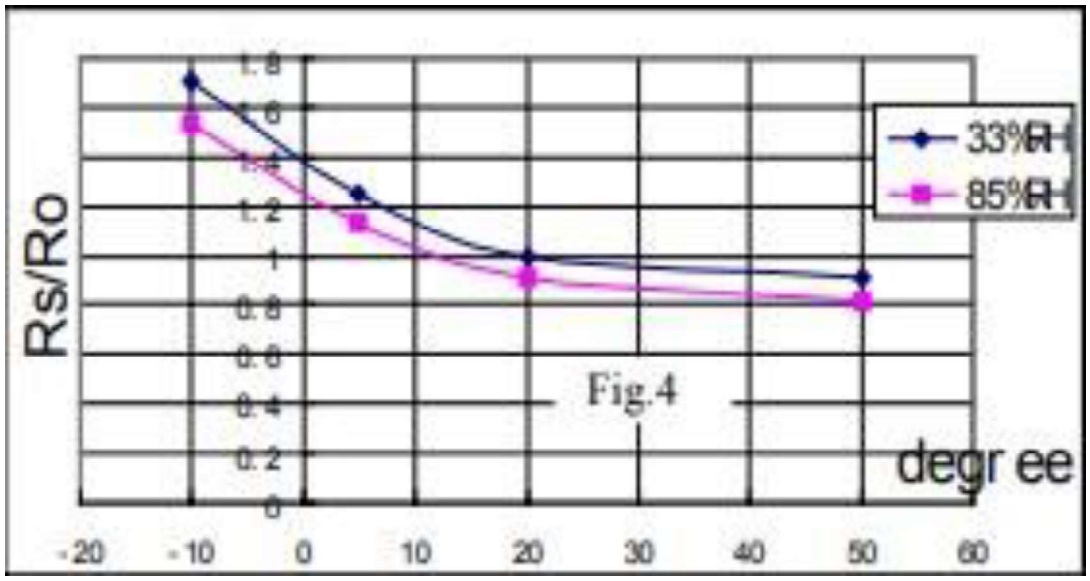

Figure 4.6: MQ-135 Rs/Ro vs temperature characteristic behaviours from the datasheet

In light of this, it was concluded that there is a need to acquire a carbon dioxide sensor that is not significantly affected by temperature and humidity. A new experiment was set up but this time with the MG-811 sensor. The MG-811 is highly sensitive to CO_2. The module's output voltage of the sensor module decreases as the concentration of the CO_2 increases. The MG-811's sensor's dependency on humidity and temperature is insignificant [21]. After getting the results from the MG-811 sensor, the plots of temperature vs CO_2 and humidity and CO_2 were generated, and the results are shown in Figures 5.1 and 5.2. The trend of the data is as perceived. Still, before concluding that MG-811 yields better results than MQ-135, we had to do a statistical analysis test to verify the difference in the datasets. First, we fit a normal distribution on the datasets using MatLab to determine whether a non-parametric or a parametric test would be suitable. The results shown in Figures 4.7 and 4.8 below imply that both datasets do not follow a normal distribution; hence non-parametric tests are better suited.

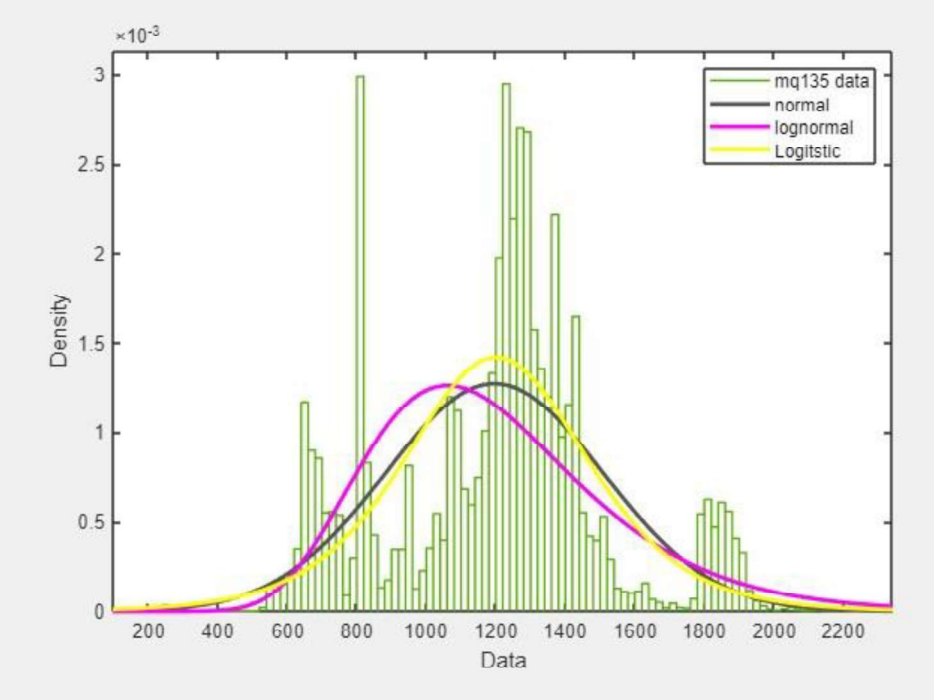

Figure 4.5: MQ-135 distribution fit for CO_2Figure 4.7: MQ-135 distribution fit for CO_2

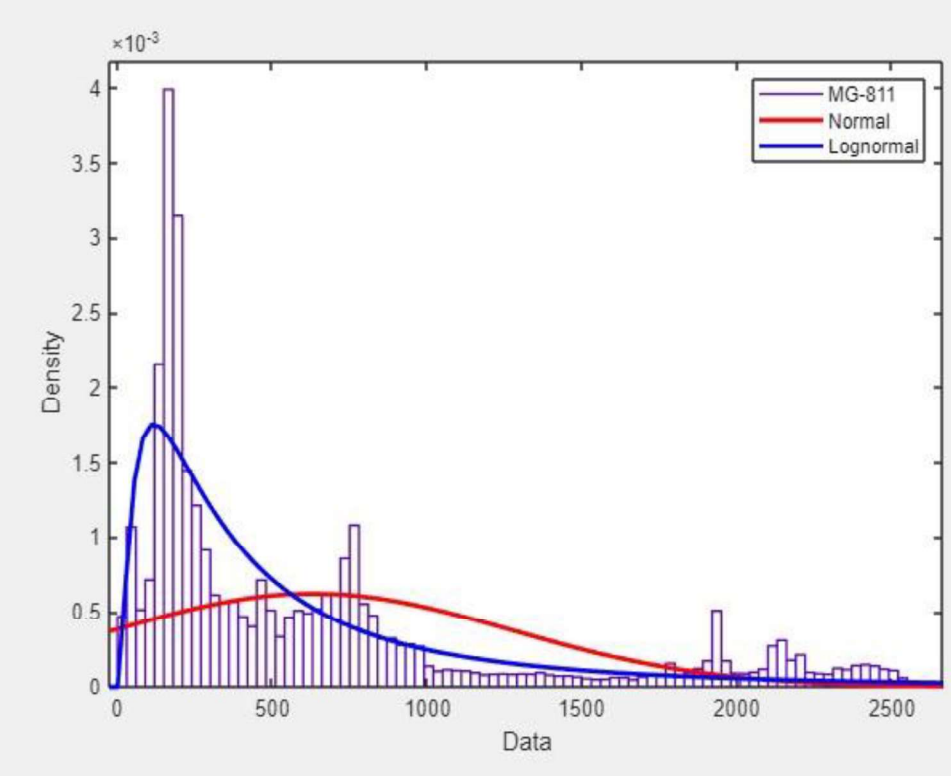

Figure 4.8: MG-811 distribution fit for CO_2

The type of data collected in this experiment is continuous and the assertion is that there is a difference in the MG-811 and MQ-135. Since we have two datasets which are independent, we had to consider the Mann Whitney test. The assumptions for Mann Whitney test are detailed in the table below [23]

Table 4.2: Represents the list of assumptions followed by Mann Whitney test

Assumption	How it is satisfied
Dependent variable should be continuous or ordinal	The variable of interest is measured CO_2 continuous
Two independent categorical groups	The two independent datasets that are analyzed are CO_2 concentration are from MG-811 and MQ-135 sensors which are independent of each other
Independence of observations	A new maize sample was used for MG-811 sensor than that used for MQ-135 sensor
Variables are not normally distributed	From Figure 4.5 and 4.6 the datasets are not normally distributed

For this test, the null hypothesis, H_0 says there is no difference between MG-811 and MQ-135. The alternative hypothesis, H_1 says there is a difference. A p-value of 0.05 is used and upon running the test in MatLab, the p-value result was 0.0223, implying that there is indeed a difference between MG-811 and MQ-135.

4.3 Data Labelling

The samples were inspected for mold growth every 2 hours, and the database was monitored for a significant rise in carbon dioxide, which indicates possible deterioration. As soon as mould growth infestation was noticed, the corresponding time stamp in the database

was recorded. This helped in the data labelling process. The system proposed in this project uses supervised learning, which is a predictive machine learning where the data comes with labels; the label is the target feature we are interested in predicting, and in our case, the target feature is grain quality status. The quality status of the grain is either good or bad, 0 or 1, respectively. As a result, data collected following the first observation of mold infestation was labelled as bad (1), while data collected prior to mold growth and the normal sample was labelled as good (0).

4.4 Classification Algorithms

This section of chapter 4 explains the classification models considered. The various processes involved in grain quality classification are also described. First, libraries such as numpy, matplotlib, classification models and sklearn.ensemble are imported from sklearn into Jupyter Notebook. The dataset was divided into two parts, 80% for training and 20% for testing using the train_test_split() from sklearn. Using the data gathered from the experiment, the following four classification models are trained using the train_set and then tested using the test_set.

4.4.1 Logistic Regression

Logistic regression is a supervised classification algorithm. In [24], logistic regression is defined as a multivariate analysis method used to study and predict the relationship between a variable, say x and a series of influencing factors. This algorithm develops a hypothesis that the likelihood that an input value belongs to a predefined class is p, where p is between 0 and 1. A threshold is defined to determine the range in which the probability will be classified as 1 or 0. This probability is usually set to 0.5 [25]. The data points are fitted to an S-shaped curve known as a sigmoid. We can determine the probability of a data observation matching into a class by computing the sigmoid function of (x). The Sigmoid function has the following formula [24]:

$$\textbf{Sigmoid}\ (\mathbf{x}) = \frac{1}{1+e^{-x}}$$

We calculate the weighted feature value given an input, which will be the input x of the Sigmoid function [24]. In this project, for a feature, say temperature, we first calculate the weight by

$$\boldsymbol{x = \theta . temperature + b}$$

The second step is calculating the probability that x belongs to a class using the sigmoid function above [23]. The cost function for logistic regression, whose role is to optimize, is given by J (θ), where J (θ) is expressed as [23]

$$J(\boldsymbol{\theta}) = -\frac{1}{m}\sum[y^{(i)} \log\left(h\theta\left(x(i)\right)\right) + (1 - y^{(i)})\log\left(-h\boldsymbol{\theta}\left(\boldsymbol{x}(\boldsymbol{i})\right)\right)]$$

Different techniques are used to optimize the cost function. The most commonly used are the gradient descent and the maximum likelihood [25]. The Logistic Regression is chosen for this project due to its binary nature since the classes into which the grain can fall are two; good or bad.

4.4.2 Random Forest

It is an expansion of a decision tree, also known as an ensemble algorithm. The operating concepts of a decision tree used to build the model are similar to conditional statements, and the result is a tree with decision nodes and leaf nodes. A decision node has two outgoing branches and one incoming branch [26]. The leaf node represents the class which the instances will be categorized into. This algorithm creates several decision trees from subsets of the overall dataset and then aggregates the votes from the different trees to arrive at the best output; this technique is called ensemble [26]. This is beneficial because individual trees are prone to errors, and this technique balances this out. Alternatively, the algorithm can work with weighting, where it will assign lower weight to algorithms that are prone to errors so that their influence

on the final decision is minimized [27]. Some critical features of the random forest include high stability through majority voting and diversity because not all attributes are considered while making each tree; hence every tree is different [27]. The following are some critical hyperparameters of the random forest algorithm that increase predictive power and speed:

1. n_estimators – number of trees the algorithm builds before averaging the predictions and for this project and this project [27]
2. max_features – maximum number of features random forest considers splitting a node [27]
3. min_sample_leaf – minimum number of leaves required to split an internal node [27]
4. random_state - controls randomness of sample and ensures that the results we have can be reproduced [28]

We defined a grid of hyperparameter ranges for this project and randomly sampled from it to conduct K-Fold CV with each combination of values, using Scikit-RandomizedSearchCV Learn's method to get the best hyperparameters for the model.

4.4.3 K-Nearest Neighbour

The K-Nearest Neighbour (KNN) algorithm is a relatively easy to implement supervised machine learning algorithm. The algorithm assumes that similar things exist nearby. The algorithm operates based on this assumption and uses math to calculate the distance between points. Usually, the Euclidean distance, also known as the straight-line distance, is used to calculate the distance between two points [29]. To find the right K value, it's recommended to repeatedly run the algorithm with different k values and choose the one that produces few errors. The algorithm's accuracy increases when the K value is greater than one, but this is true up to a point when the further increase will only result in more errors [29].

4.4.4 Gradient Boosting

It is also known as an ensemble technique, but more specifically, it is a boosting technique [30]. Gradient boosting classifiers are a group of machine learning techniques that merge multiple weak supervised learning methods to produce a robust predictive model [30]. Decision trees are frequently used in gradient boosting. The two most common boosting methods are adaptive and gradient. In gradient boosting, each predictor strives to improve on the errors of its predecessor by fitting on the residual errors rather than on data points at each iteration. To perform initial predictions on the data, the gradient boosting classifier calculates the log of the target feature's odds. This is the number of true values (1) divided by the number of falses (0). The log(odds) is then converted to a probability by using a logistic function in order to make predictions. To convert log(odds) into a probability, we do [31]:

(e*log(odds))/(1+e*log(odds))

For every instance in the training set, the classifier calculates residuals for each instance (true_value – predicted_value). In summary, two steps are executed. The first is getting the log(odds) prediction for each instance in the training set. The second is converting a log(odds) into a probability, and the formula for predicting is

Base_log_odds + (learning_rate*predicted_residual_value) [31]

4.5 Metrics Considered

The metrics that are also considered are F1-score, Precision and Recall. These metrics are also considered because accuracy only looks at the number of correct predictions, but precision looks at how many true positive predictions were made. In this project, true positive prediction implies grain that is classified as normal is indeed normal. The recall is the proportion of negative predictions, while the F1-score balances the concern of precision and recall in a single score. A good F1 score is interpreted as low false negatives and positives. A Receiver Operating Characteristics (ROC) curve for comparing the classifiers was generated.

The ROC curve is plotted for True Positive Rate (TPR) vs False Positive Rate (FPR). As the TPR increases, the FPR also increases. The ROC curve considers true negatives and true positives [15], which is an important consideration when the task at hand is interested in both classes. In this project, we need to classify the grain condition accurately, which is why the ROC curve is another essential matric that we will consider when evaluating the performance of the classifiers.

True positives (TP): Predicted positive and are actually positive [32].

False positives (FP): Predicted positive and are actually negative [32].

True negatives (TN): Predicted negative and are actually negative [32].

False negatives (FN): Predicted negative and are actually positive [32].

$$\text{True Positive Rate, } \mathbf{TPR} = \frac{TRUE\ POSITIVE\ (TP)}{TRUE\ POSITIVE(TP)+FALSE\ NEGATIVE(FN)} \quad (1)$$

$$\text{False Positive Rate, } \mathbf{FPR} = \frac{FALSE\ POSITIVE\ (FP)}{TRUE\ NEGATIVE(TN) + FALSE\ POSITIVE(FP)} \quad (2)$$

$$\text{Precision} = \frac{TRUE\ POSITIVE\ (TP)}{TRUE\ POSITIVE(TP) + FALSE\ POSITIVE(FP)} \quad (3)$$

$$\text{Recall} = \frac{FALSE\ POSITIVE\ (FP)}{TRUE\ NEGATIVE(TN) + FALSE\ POSITIVE(FP)} \quad (4)$$

$$\text{Accuracy} = \frac{TRUE\ POSITIVE\ (TP)+TRUE\ NEGATIVE\ (TN\_}{TRUE\ NEGATIVE(TN) + FALSE\ POSITIVE(FP) + TRUE\ POSITIVE\ (TP) + FALSE\ NEGATIVE(FN)} \quad (5)$$

$$\text{F1-score, FPR} = \frac{FALSE\ POSITIVE\ (FP)}{TRUE\ NEGATIVE(TN) + FALSE\ POSITIVE(FP)} \quad (6)$$

4.6 Cross Validation

Cross validation is a technique for ensuring that our trained model has a low level of bias and variance. It is the process of determining whether numerical results quantifying hypothesized relationships between variables are adequate as data descriptions (true reflection)

[33]. The train_test_split module divides the dataset into two parts: a training set from which the model learns and a test split from which the model is tested. The reserved data may contain important properties that we overlook when training the model, necessitating the use of cross validation, which provides a more precise assessment of accuracy. Furthermore, because each observation is used for both learning and testing, cross validation is a more "efficient" way of utilizing data. The method of cross validation used in this project is the k-fold. The k-fold method splits the dataset into k number of subsets and perform training on all the subsets leaving one (k-1) subset for testing. We iterate k times, each time reserving a different subset for evaluation. The k fold results can then be averaged (or otherwise combined) to generate a single estimate [34]. K-fold cross validation was implemented for each of the four algorithms and the resulting accuracy scores are tabulated in chapter 5.

4.7 Learning Curves

Learning curves can be used to detect models that are overfitting or underfitting. When a model is overtrained on data and learns its noise, it is said to overfit [35]. Since it learns the noise perfectly, an overfit model has a near-perfect training score but a bad test/validation score because it misclassifies unseen real-world instances [36]. Overfitting can occur when a complicated model is used to solve a basic problem, and the model picks the noise from all the data. When a model fails to understand the patterns in the data adequately and has a poor training and test score, it is said to be underfitting [36]. Underfitting can be caused by employing a simplistic model to solve a complex problem. It is critical to select a strong fit model that can provide dependable predictions for this project [37]. As a result, learning curves are utilized to distinguish between underfitting and overfitting. For each trained model, a learning curve is plotted. The optimal model will have properties that are similar to those of a good fit model [38]. The optimal model's learning curve does not suffer from high bias and variance. High bias is when a model's training score is far below the desired, which is one and

high variance occurs when the cross-validation score does not converge with the training score and there is large difference in their values [38].

4.8 Circuit Designs

4.8.1 Receiver (Gateway) node

The circuit design for the gateway comprises of the HC12 transceiver module for receiving sensor reading from the sensor nodes. The transceiver module is connected to the esp32; this microcontroller unit sends the data to the MYSQL database with the help of visual studio code.

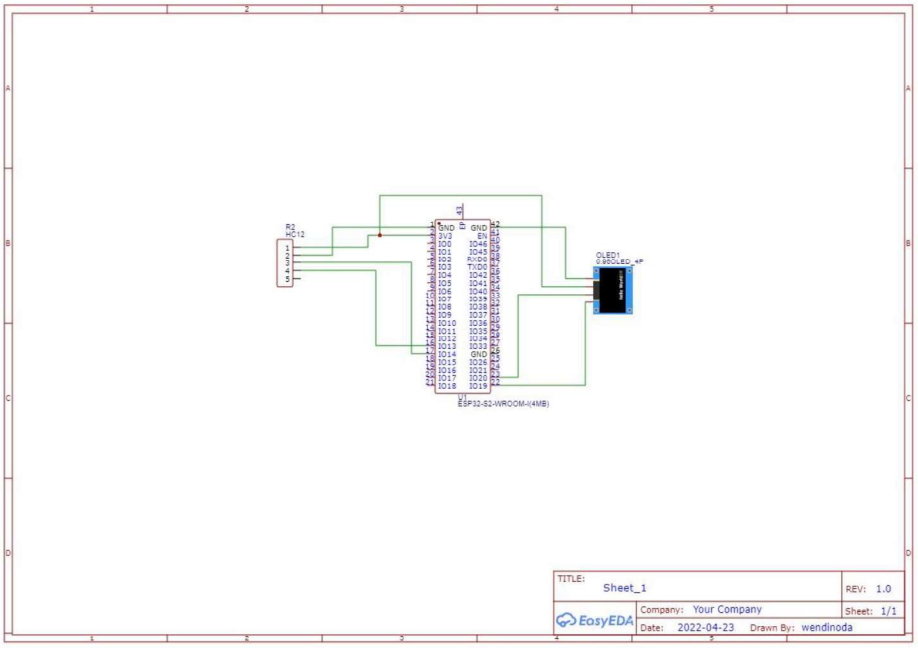

Figure 4.9: Receiver (Gateway) node circuit diagram

4.8.2 Sensor Nodes Circuit Designs

For the circuit of the sensor node, we had two designs for different sensor node. This is because the sensor nodes do not measure the same parameters. The first sensor node is the top most node in the silo, this node has the DHT22 sensor, the MG-811 sensor, and the HC-SR04 ultrasonic sensor. The ultrasonic sensor, in this case, is used to measure the level of the grain

in the silo. The sensor node two measures the same conditions that is, CO2, temperature and humidity and hence they have the same circuit design. All the sensor nodes make use of ATmega 328p and the HC12 transceiver module. Sensor nodes 1 and 2 include a battery power supply and the LM7805 5 volts regulator. The ceramic resonator/ quartz connected to XTAL1 and XTAL2 of the ATmega microcontroller chip makes the inbuilt clock work.

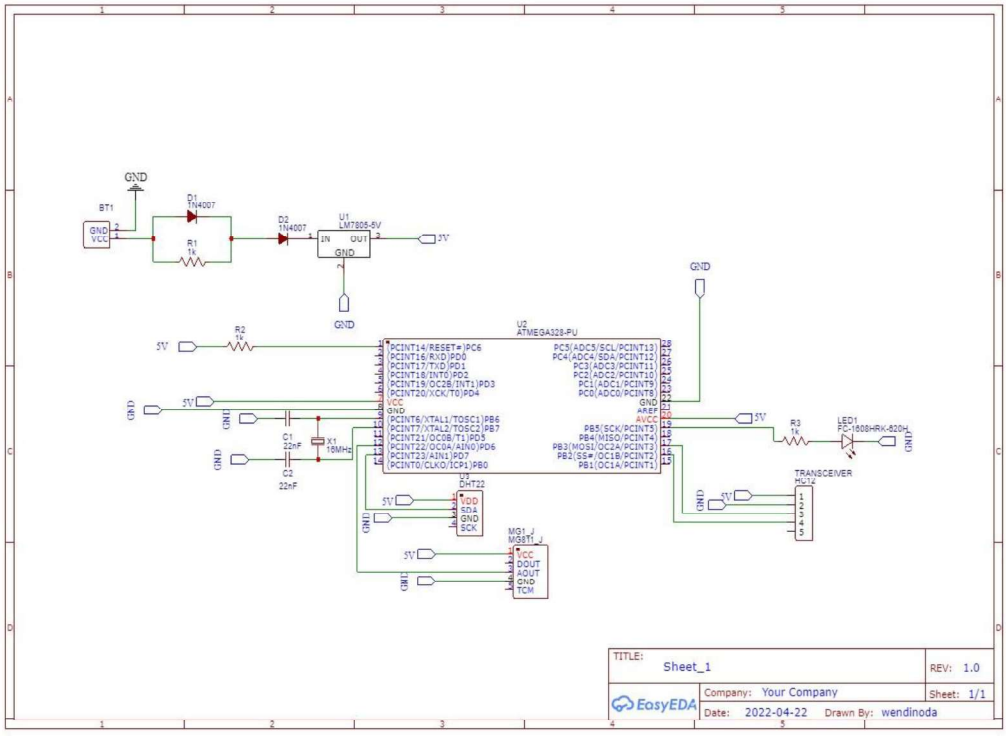

Figure 4.10: Sensor node 2 circuit diagram

Figure 4.11: Perforated board implementation of sensor node 2

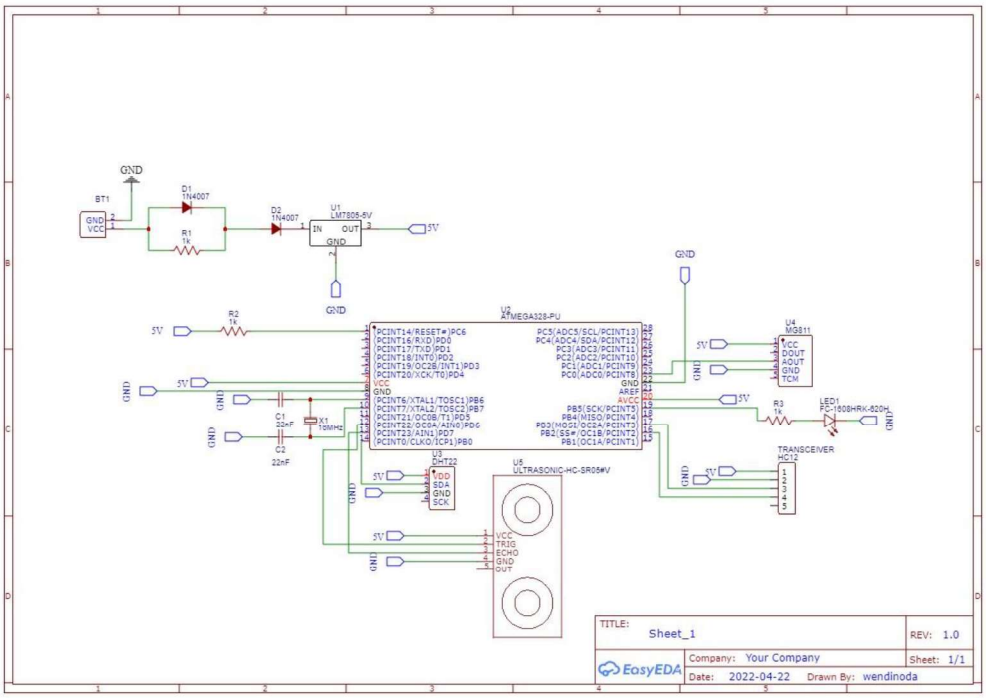

Figure 4.12: Sensor node 1 circuit diagram

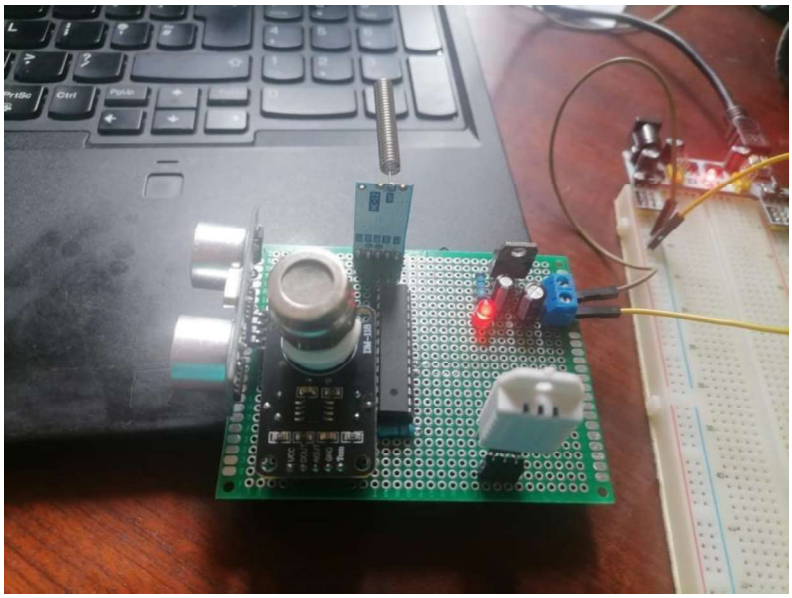

Figure 4.13: Perforated board implementation of sensor node 1

4.9 Range Test

In order to ensure that the HC12 transceiver modules would be able to reach the required distance, a range test was conducted with two of the modules. Each HC12 module was connected to one Arduino nano board. Pins 10 and 11 was used as the transmit and receive pins. one setup with the Arduino and the HC12 represented the transmitter which is our sensor node, and the other one represented the receiver side. A potentiometer which was connected to an analog pin at the transmitted side was used to control an LED at the receiver side. At distance of about 200m apart, the two devices were still able to communicate with each other.

4.10 Software Design of Sensor Nodes

The software design of the sensor nodes includes a setup function, a loop function, and a data reading function. The development and the compilation of the code was done on the Arduino ide using C++ programming language. In the set-up function, we declared and set up the input pins, and set up the serial and baud rate for the HC12 modules and also sensor declaration. The main loop function is configured to wait for an incoming request packet from

the gateway designated for that particular node. The sensors are triggered after the request packet is received to read and then subsequently forward readings to the gateway. The sensor node will only read when it receives request that is unique to its nodes. Each node is given a unique node ID that aids the receiver/ gateway to differentiate each data segment for the respective node.

4.11 Software Design for the Gateway

The software for the receiver or the gateway was designed based on the hardware design. The esp32 was used as the microcontroller in this node. This design includes a set up configuration of modules connected to it, sending request to different nodes, and finally storing data from the nodes. The development and the compilation of the code was done on the Arduino IDE using C++ programming language.

The setup of the receiver node includes setting the serial port and over the air baud rate for the HC12 module and the initialization process for the data base system. After each node request is sent, the data is stored with the aid of uniquely node IDs that are added to the data transmission from each node.

4.12 Network architecture

The network topology deployed is based on a tree topology. In this the gateway is the central mode that sends request to the sensor nodes. The sensor nodes depend on the higher node to be active. The gateway node generates the request, and broadcasts, each node checks the data segment and actions are triggered if the request is meant for that node. To prevent the gateway from being overwhelmed, unique IDs are given. For example, when the gateway sends a request for node one reading, node one sends its sensor reading with its node ID. The network logic for data transmission is implemented at the receiver side. When several data packages are being sent by the nodes, preference is given to the first node that started its transmission.

Chapter 5: Results

5.1 Results from the experiment

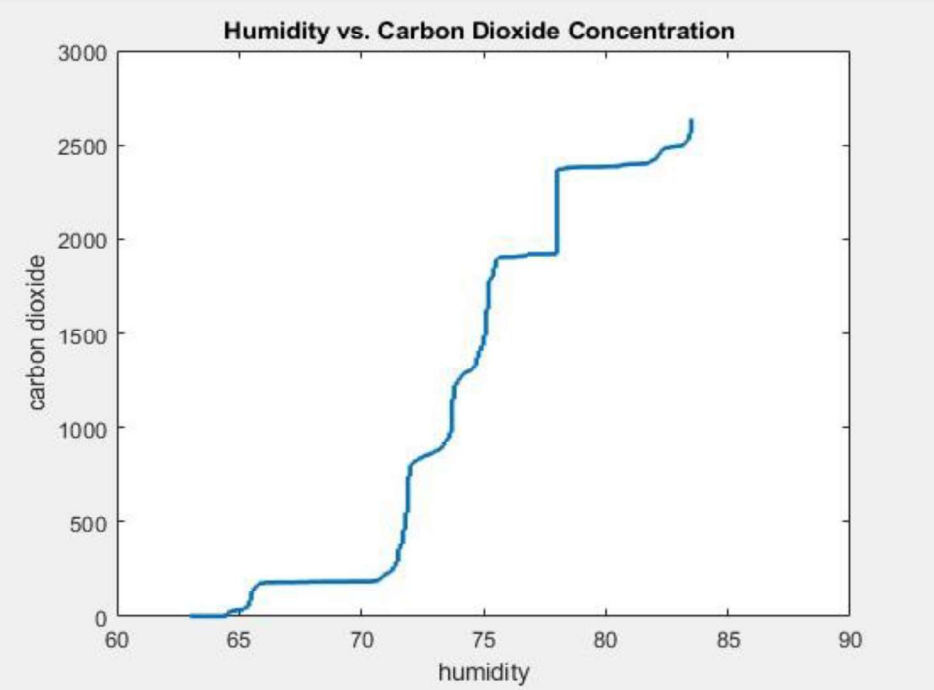

Figure 5.1: Plot of carbon dioxide vs temperature based on experimental data for the sample with added moisture

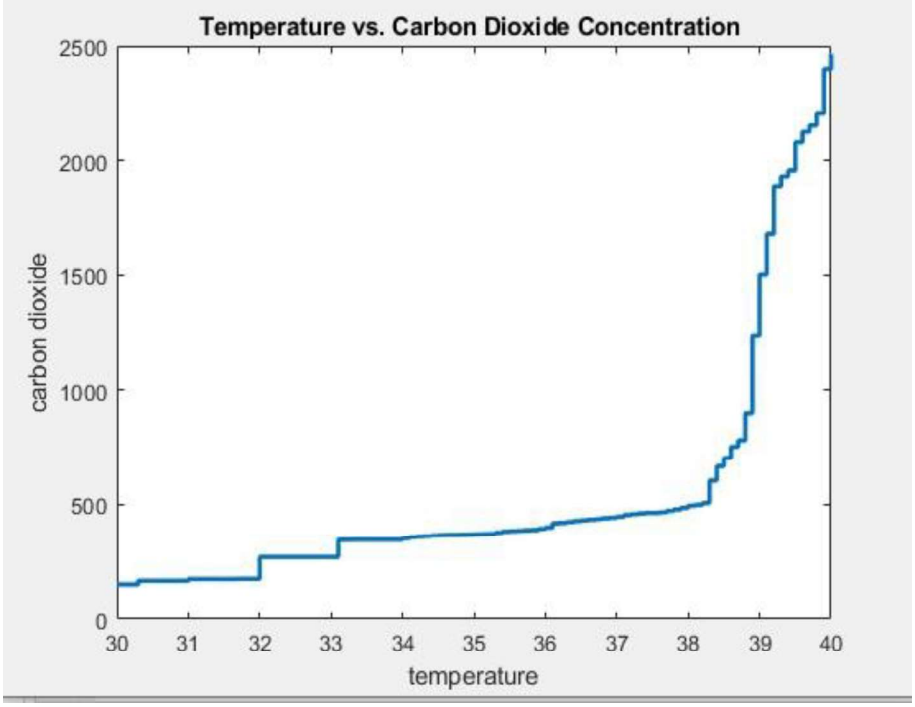

Figure 5.2: Plot of carbon dioxide vs temperature based on experimental data for the sample with added moisture

5.2 Grain Quality Classification

The train test split model from sklearn was used to split the dataset acquired throughout the data collection experiment. The dataset was divided into two parts: a train set with 80% of the data and a test set with 20% of the data. Random state parameter was initialized to 100; when random state is set to a fixed value, the same sequence of random integers is created each time the code is executed, verifying the data when the algorithm is run numerous times.

The learning curves of each trained model illustrating training and validation scores was plotted. Additionally, confusion matrices and classification reports for each model were also generated. A Receiver Operating Characteristics (ROC) curve and learning curves for comparing the classifiers was generated.

5.2.1 Feature correlation matrix

Figure 5.3: Feature correlation matrix

5.2.2 Results from testing the Logistic Regression Classification

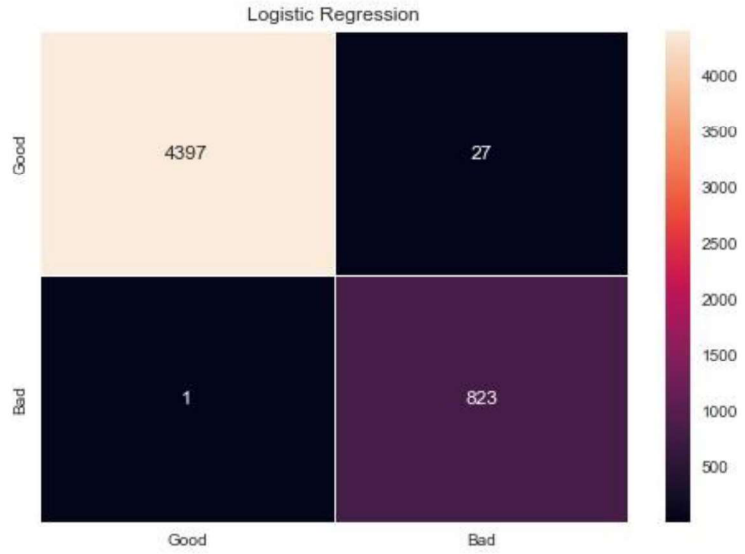

Fig 5.4: Logistic Regression confusion matrix

Table 5.1: Logistic Regression classification report before cross validation

	Precision	Recall	F1-score	support
0(normal)	0.9998	0.9939	0.9968	4424
1(infested)	0.9682	0.9988	0.9833	824
Accuracy			0.99	
Weighted avg	0.9948	0.9947	0.9947	5248

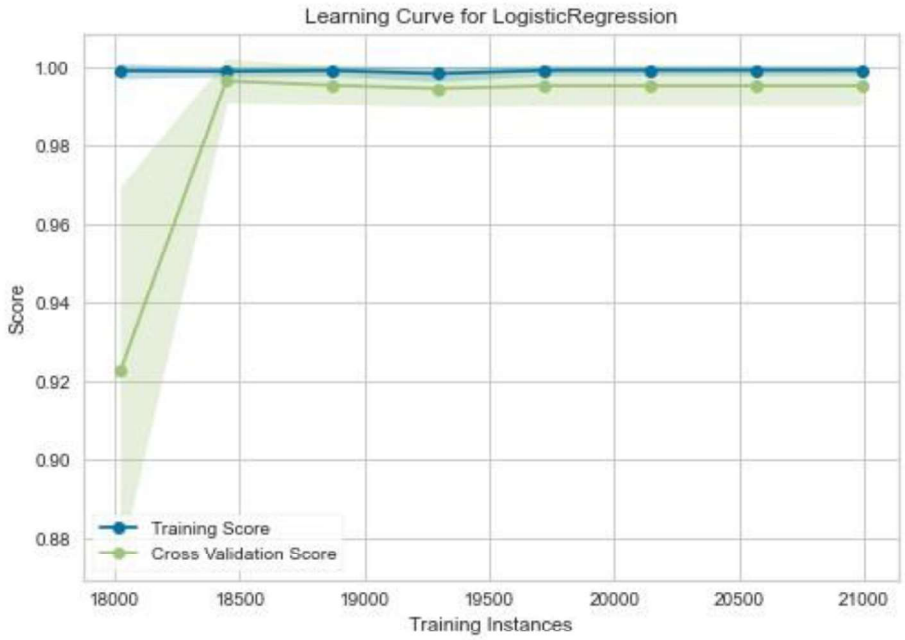

Figure 5.5: Learning curve for Logistic Regression

5.2.3 Results from testing the Random Forest Classification

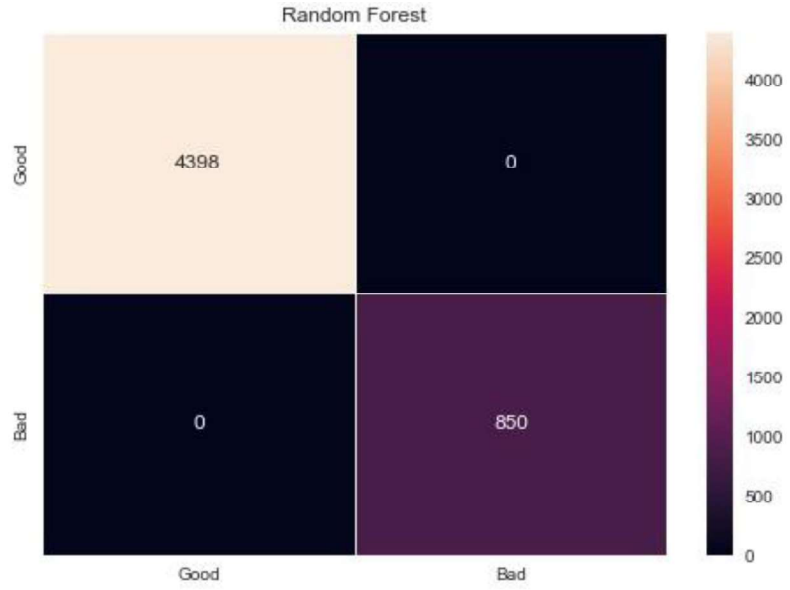

Figure 5.6: Confusion matrix for Random Forest

Table 5.2 Random Forest classification report before cross validation

	Precision	Recall	F1-score	support
0(normal)	1	1	1	4398
1(infested)	1	1	1	850
Accuracy			1	
Weighted avg	1	1	1	5248

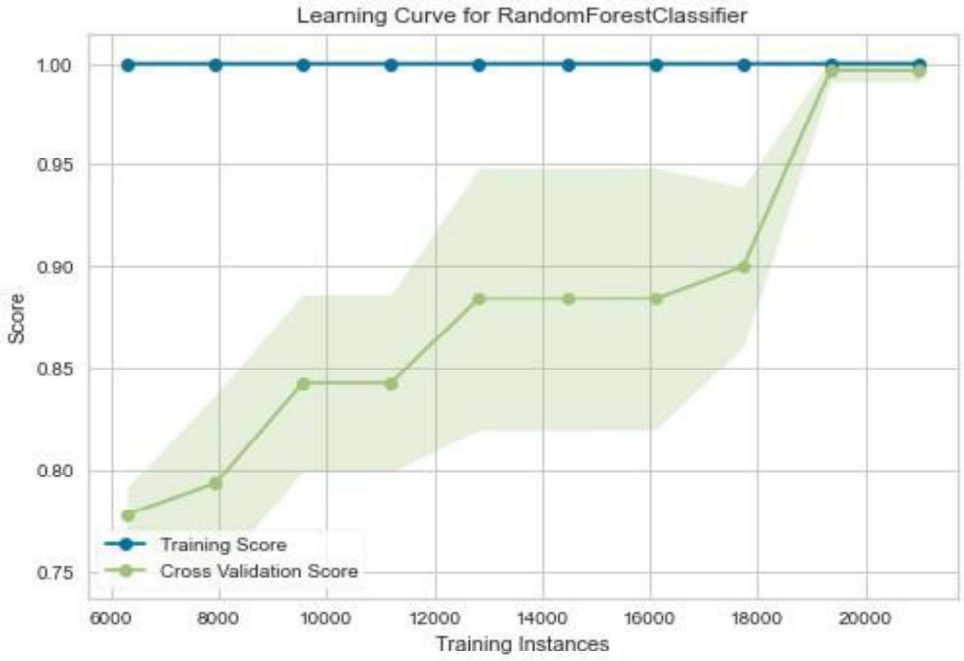

Figure 5.7: Learning curve for random forest

5.2.4 Results from testing the Gradient Boosting Classification

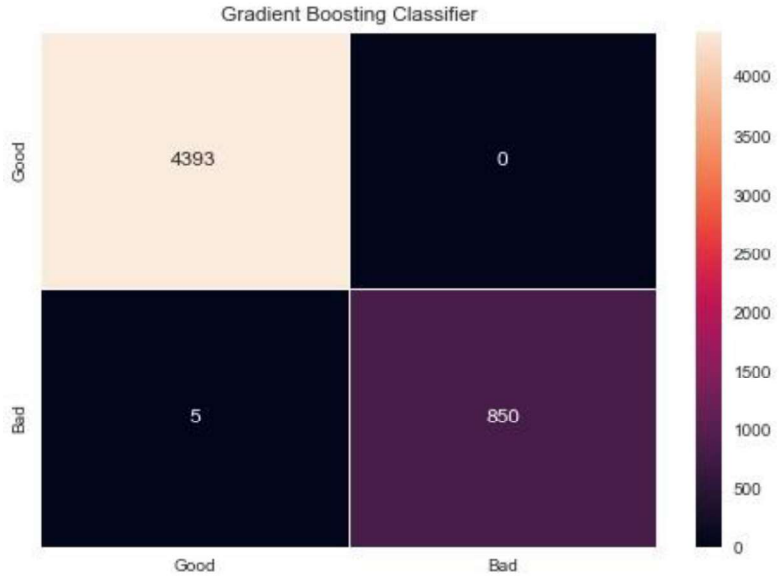

Figure 5.8: Confusion matrix for gradient boosting classifier

Table 5.3 Gradient boosting classification report before cross validation

	Precision	Recall	F1-score	support
0(normal)	0.9989	1	0.9994	4393
1(infested)	1	0.9942	0.9971	855
Accuracy			1	
Weighted avg	0.9990	0.9990	0.9990	5248

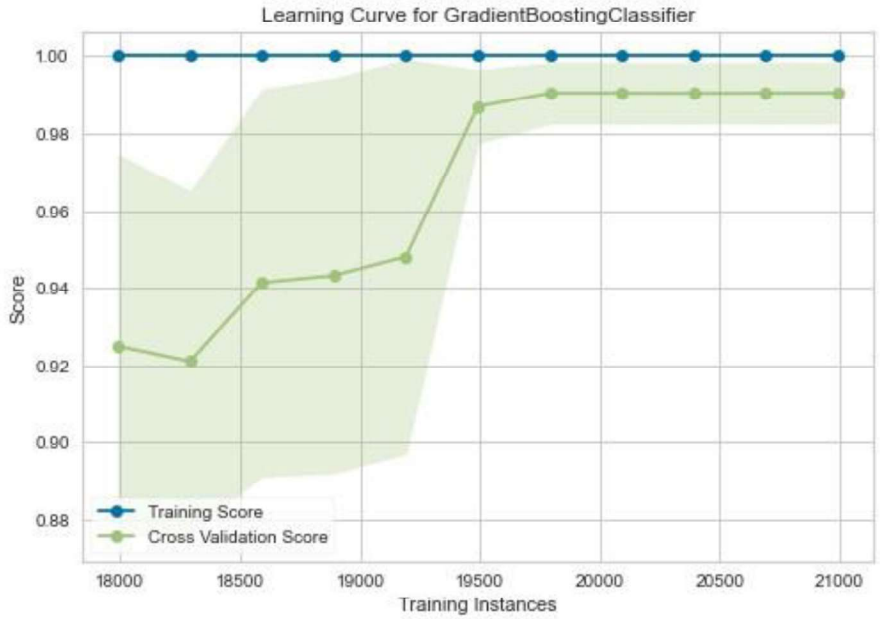

Figure 5.9: Learning curve for gradient boosting

5.2.5 Results from testing the KNearest Neighbor Classification

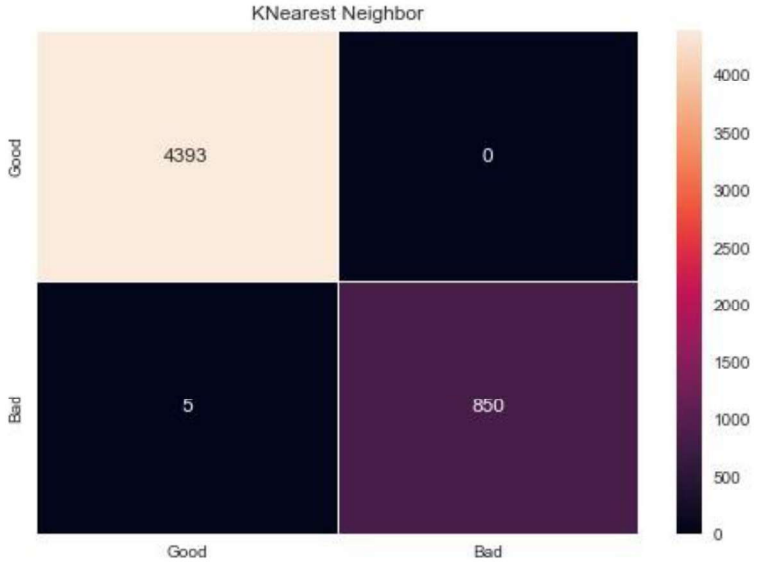

Figure 5.10: Confusion matrix for KNearest Neighbor

Table 5.4: KNearest Neighbor classification report before cross validation

	Precision	Recall	F1-score	support
0(normal)	0.9989	1	0.9994	4393
1(infested)	1	0.9942	0.9971	855
Accuracy			1	
Weighted avg	0.9990	0.9990	0.9990	5248

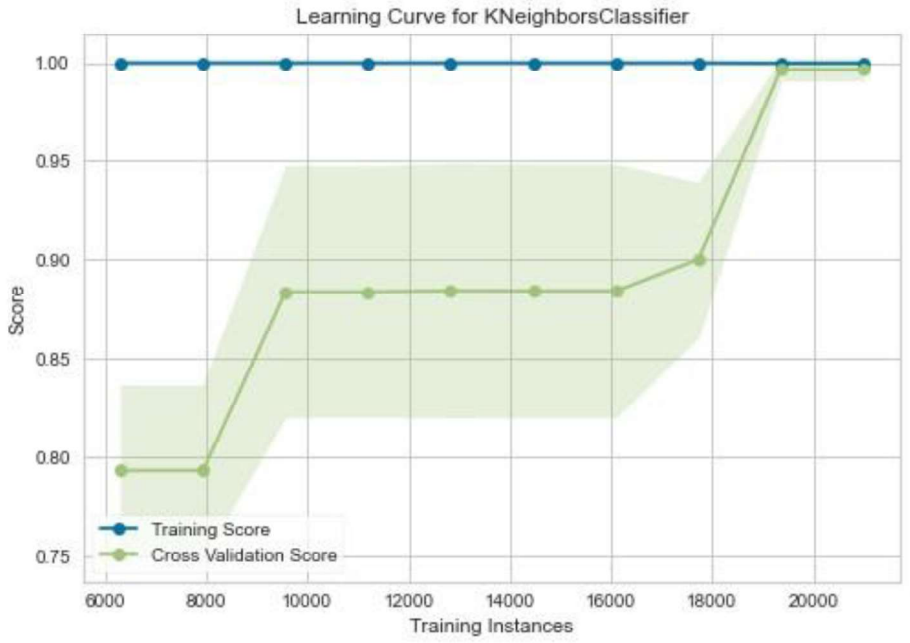

Figure 5.11: Learning curve for KNearest Neighbor

5.3 Comparison of classifier accuracy scores

5.3. 1 The (ROC) Curve

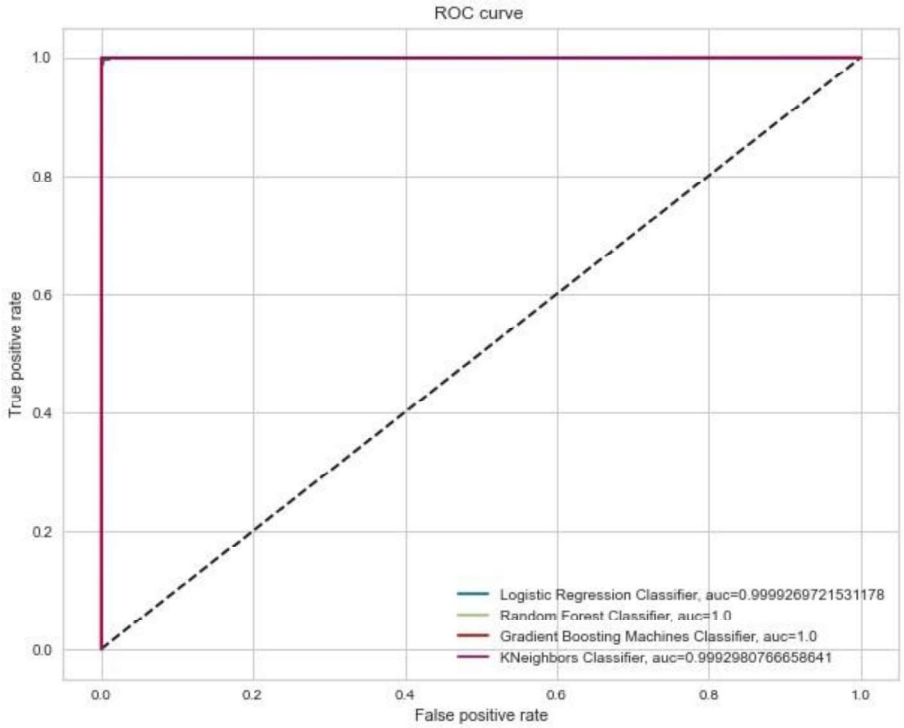

Figure 5.12: Receiver Operating Characteristic curve

Table 5.6 showing comparison between classifier accuracy scores

	AUC-ROC score	Train_test_split Accurary	Cross Validation Accuracy
Logistic Regression	0.99999	0.99	0.9485
Random Forest	1	1	0.9058
Gradient Boosting	1	1	0.8899
KNearest Neighbors	0.99929	1	0.9410

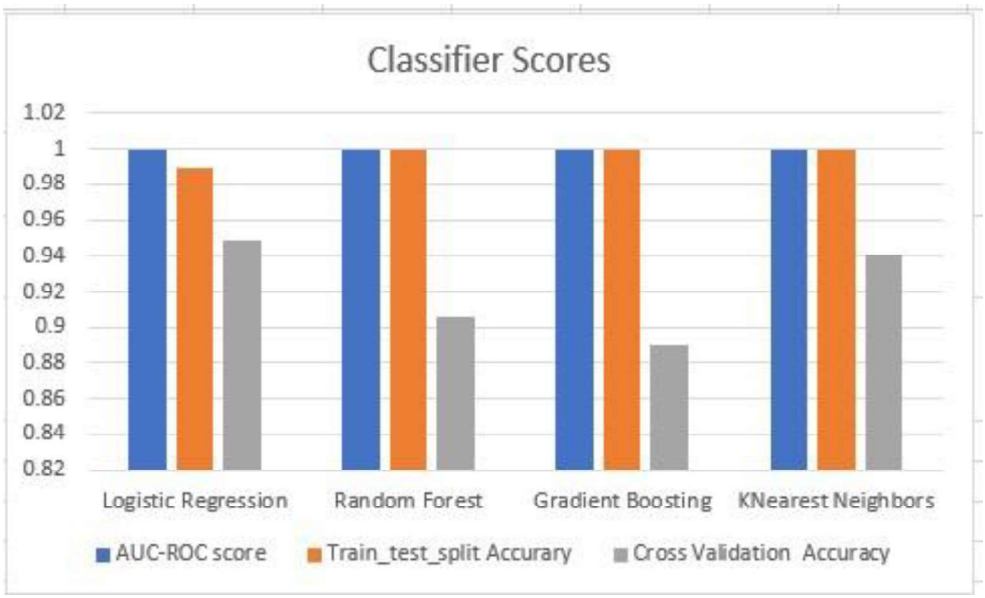

Figure 5.13: Different classifier accuracy scores

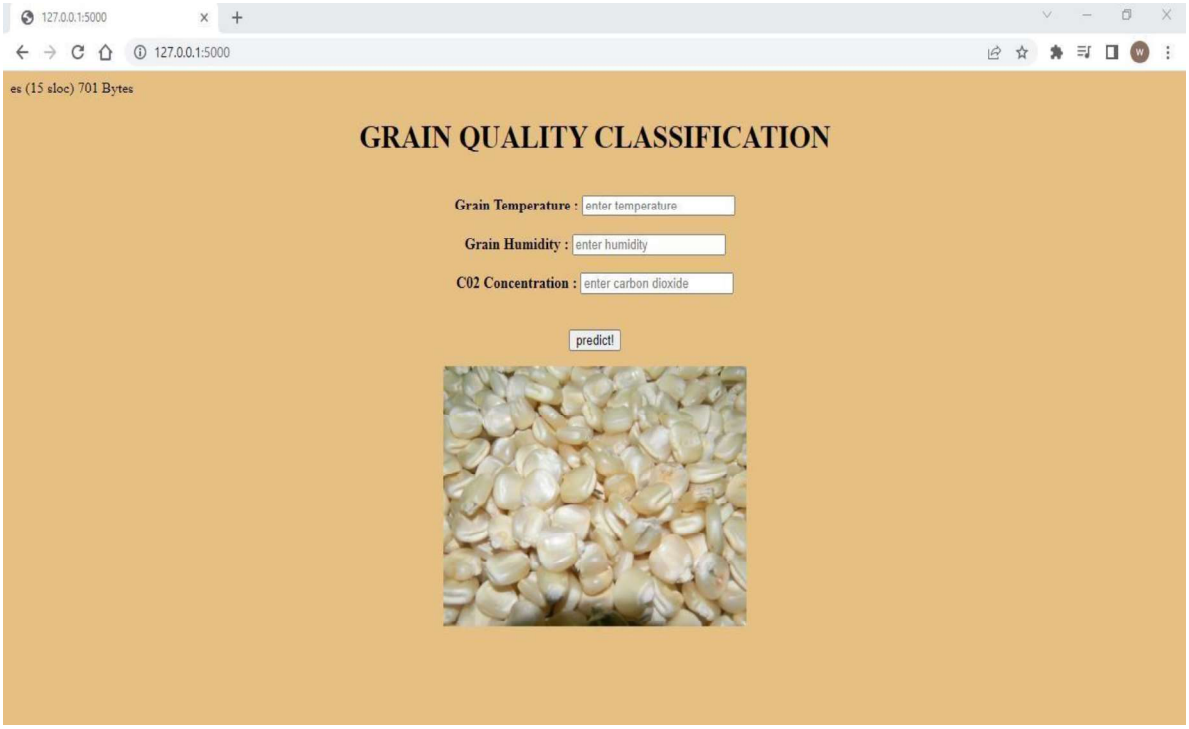

Figure 5.14: Model deployment website homepage

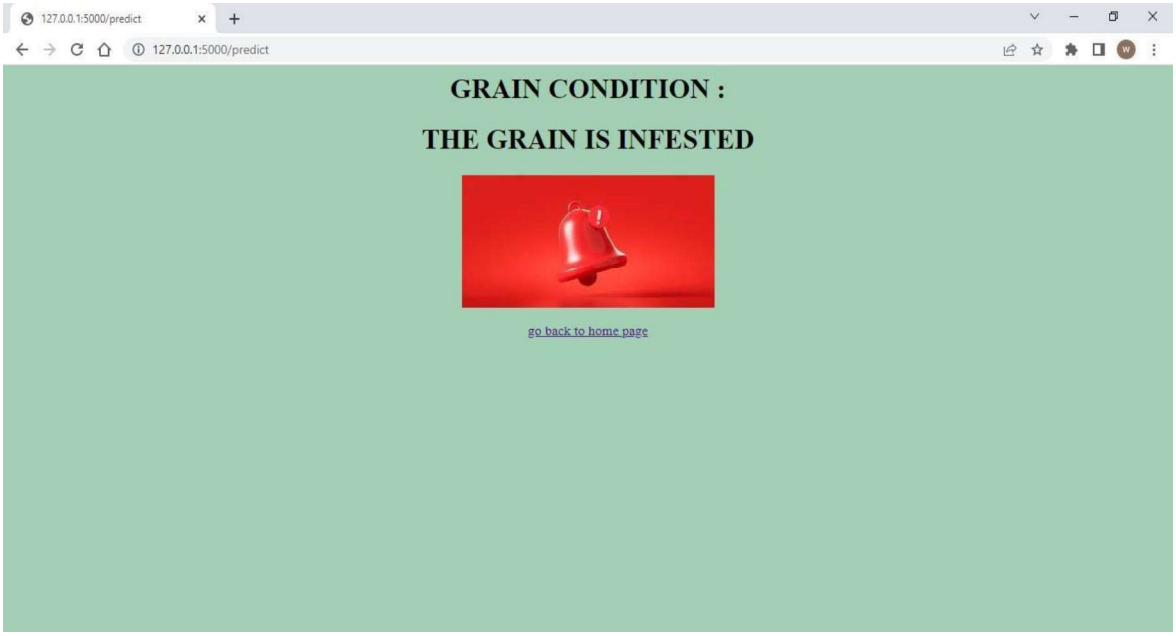

Figure 5.15: Model deployment result pages

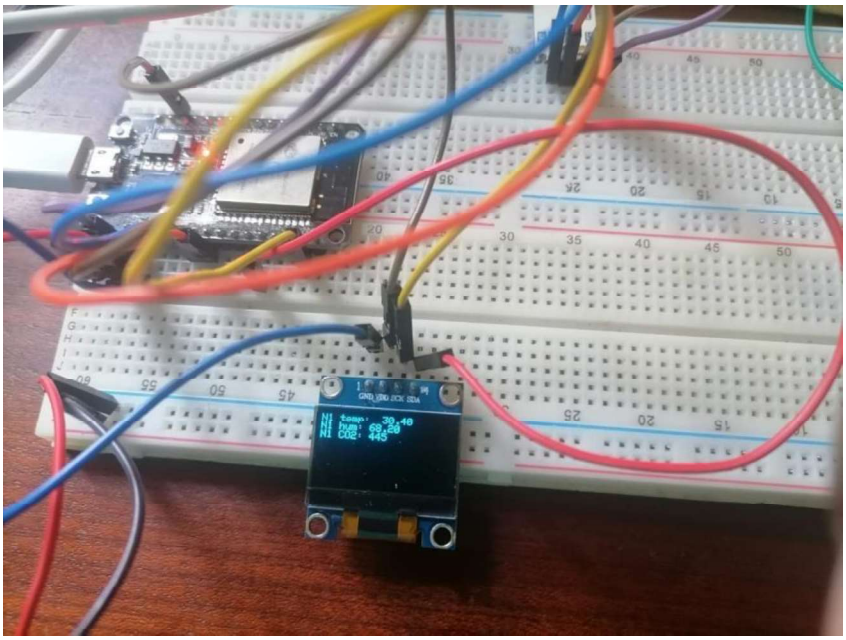

Figure 5.16: Node data display at receiver end

Chapter 6: Conclusion

6.1 Discussion

Figures 5.1 and 5.2 show the plot of the experimental data gathered from the experiment using MG-811 carbon dioxide sensor. Temperature, humidity and carbon dioxide values readings increased when the grain started developing molds. The carbon dioxide concentration was plotted against temperature and humidity and from both graphs it is evident that as the temperature and humidity increased, the carbon dioxide concentration also increased.

Figure 5.3 illustrates the feature correlation matrix, which is important in detailing the relationship between the features of the dataset and the target class we wish to predict. For this project, the features of the dataset are humidity, temperature and carbon dioxide, while the target class is the grain condition status. CO_2 has a higher correlation coefficient of 0.85, meaning a much clearer distinction between the CO_2 of normal grain and that of decaying grain, followed by temperature with 0.56 and humidity with 0.48, which means that all features are distinguishing factors. They are useful for classification purposes.

After splitting the dataset into train and test set, the different classification models were fit to the train data set. After fitting each model to the train data, it was evaluated using the test set. The accuracy score was calculated using the .score() function. Gradient Boosting classifier, KNearest Neighbours and Random Forest classifier have a perfect score of 1, while Logistic Regression have accuracy scores of 0.99.

The most popular measure for measuring a model's performance is accuracy, which is not always a reliable indicator [12]. Even though the train_test_split accuracy of all the classifiers are perfect other implications of other metrics discussed in chapter 4 and shown in chapter 5 are to be considered. Gradient Boosting classifier and KNearest Neighbours have the same weighted average precision, recall and F1-score values, 0.9989, 1 and 0.9994,

respectively while Random Forest has all 1's. Lastly, Logistic Regression recorded the precision, recall and F1-score of 0.9948, 0.9947 and 0.9947 respectively. Considering the confusion matrices of the models, we can deduce the misclassification rate. In the case of the Gradient Boosting classifier and Random Forest, 0 instances out of 5258 are misclassified, giving a misclassification rate of 0%. In the case of KNearest Neighbours, 5 cases out of 5248 instances are misclassified, giving a misclassification rate of 0%. Additionally, in the case of the Logistic Regression classifier, 28 instances out of 5248 are misclassified, which gives a misclassification rate of 0.8739%.

Up until roughly 18,300 instances, the Logistic Regression learning curve exhibits considerably low test score. Beyond that, the model begins to converge on an F1 score of around 0.998. We can observe that the training and test scores have not yet converged, implying that more training data could enhance this model, but it has low variance and bias making it ideal for this project. The learning curves for Random Forest and KNearest Neighbours are similar; the test score starts comparatively lower than the training score until about 19,000 instances. They both start converging with the training score around 0.99. The learning curve of the Gradient Boosting Classifier shows that the cross-validation score starts low and starts to converge after about 19,570 instances at an F1 score of approximately 0.99. Logistic regression converges first, which suggests that this model learns the data better than the others and is a good fit. Furthermore, the other three models' cross-validation scores for test data are still variable relative to training data scores at the point when Logistic Regression starts to converge (18,300 instances), implying that these models may be overfitting. The AUC-ROC Score informs us on how well a model is capable of differentiating between classes [13]. From figure 5.8 and table 5.6 the AUC-ROC scores of Logistic Regression, Random Forest, Gradient Boosting and KNearest Neighbours classifiers are 0.9999, 1, 1 and 0.9929, respectively. Furthermore, cross-validation is implemented to ensure that the models are not overfitting.

From table 5.8 above the cross-validated accuracy scores for Logistic Regression, Random Forest, Gradient Boosting and KNearest Neighbours are 0.9485, 0.9058, 0.8899 and 0.9410, respectively.

Judging from the metrics and results from chapter 5, the Logistic Regression classifier outperforms the other classification models when considering all the evaluation metrics hence it is chosen for the project.

In addition to that, the overall system design was able to meet its main objective. First, we correctly measured grain environmental conditions at the different sensor nodes and sent the readings to the gateway using the HC12 transceiver modules. At the gateway side, the farm manager will see the grain condition at different nodes at different intervals. This was achieved by placing an OLED display at the receiver side. The smart grain monitoring system was able to process sensor readings accurately by classing the grain-based on the recorded grain conditions.

6.2 Limitations

The smart grain monitoring system was designed and implemented to decrease postharvest losses in grain silos during storage. Some limitations hampered the system's final prototype design. The following are the restrictions:

- There is a scarcity of information on stored grain conditions. Existing data would have been critical to spotting patterns in grain environmental conditions in the grain silos. Hence, this was a severe design flaw. Furthermore, having previous data would have aided us in improving the machine learning part of the project, which predicts how long the grain will remain based on sensor readings.

- The slow responsiveness of some of the electronic components was another drawback. During testing, for example, components, such as the HC12 transceiver module ceased operating at certain times, resulting in data loss.
- We had to develop an experiment to collect data due to a lack of existing data on grain conditions. The problem was that the project's scheduling schedule would not allow us to let the grain to decay naturally, so we added moisture to one sample to speed up the rate of decay in that sample. This was a limitation because grains that decay naturally might have a different range of environmental condition.

6.3 Future Work

In executing the project, areas that need more research and improvement were identified. Therefore, the following future works are proposed to improve the overall system's efficiency

- The system design should be able to trigger regulatory actions based on the readings of the sensors; if readings exceed predefined thresholds, the system can automatically turn on/off a fan or open/close windows
- Acquire past data of harvesting time, storage conditions (temperature, humidity, etc.) and duration of storage vs storage losses that will allow us to predict how long grain can stay in storage before decaying (grain dispatch sequence). Meteorological data can also be taken into consideration
- Use advanced sensors like the radar for grain level monitoring. Ultrasonic sensors are affected by dust, and a silo environment is dusty; hence they are not suitable for this application. Temperature cables are recommended because they run through the grain thus, they can capture the temperature of the grain better than DHT22. Grain infestation starts occurring in localized areas called hotspots which causes a higher temperature in those specific areas, and the DHT22 may fail to capture this.
- Test other transceiver modules because the HC-12 would go on and off during the

testing of this project

- In addition to that, to prevent data packets from colliding, future work would implement data polling on the receiver side.

References

[1] I. Sugri, M. Abubakari, R. K. Owusu, and J. K. Bidzakin, "Post-harvest losses and mitigating technologies: evidence from Upper East Region of Ghana," *Sustainable Futures,* vol. 3, p. 100048, Jan. 2021, doi: 10.1016/j.sftr.2021.100048.

[2] "Ghana loses 318,514 tons of maize annually to post-harvest losses – Study," GhanaWeb, Nov. 16, 2017. https://www.ghanaweb.com/GhanaHomePage/business/Ghana-loses-318-514-tons-of-maizeannually-to-post-harvest-losses-Study-601182 (accessed Oct. 10, 2021).

[3] S. Parvin, A. Gawanmeh, and S. Venkatraman, "Optimised Sensor Based Smart System for Efficient Monitoring of Grain Storage," *IEEE Xplore*, May 01, 2018.

[4] A. Taiwo and A. Bart-Plange, "Factors Responsible For Post-Harvest Losses And Their Effects On Rice Producing Farmers: A Case Study Of Afife And Aveyime Rice Projectsin The Volta Region Of Ghana," *International Research Journal of Engineering and Technology (IRJET)*, vol. 3, pp. 1014–1022, Apr. 2016.

[5] P. K. Awua, "Upstream processing of agricultural produce: prospects and challenges for Ghana's economic development," *Journal of the Ghana Science Association*, vol. 2, no. 3, Mar. 2000.

[6] H. C. J. Godfray et al., "Food Security: The Challenge of Feeding 9 Billion People," *Science*, vol. 327, no. 5967, pp. 812–818, Feb. 2010, doi: 10.1126/science.1185383.

[7] R. J. Hodges, J. C. Buzby, and B. Bennett, "Post-harvest losses and waste in developed and less developed countries: opportunities to improve resource use," *The Journal of Agricultural Science*, vol. 149, no. S1, pp. 37–45, 2011.

[8] D. Kumar and P. Kalita, "Reducing Postharvest Losses during Storage of Grain Crops to Strengthen Food Security in Developing Countries," *Foods*, vol. 6, no. 1, p. 8, Jan. 2017, doi: 10.3390/foods6010008.

[9] "Post-harvest losses In Ghana; The Way Forward," *Sesi Technologies,* Apr. 14, 2021. https://sesitechnologies.com/post-harvest-losses-in-ghana-the-way-forward/

[10] United Nations, "Goal 2: Zero Hunger - United Nations Sustainable Development," *United Nations Sustainable Development*.

[11] A. Nsafoah, "Centralized Grain Storage in Ghana: A Feasibility Analysis," Dec. 2012.

[12] M. O. Onibonoje and T. O. Olowu, "Real-time remote monitoring and automated control of granary environmental factors using wireless sensor network," in 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Sep. 2017, pp. 113–118.

[13] H. Purandare, N. Ketkar, S. Pansare, P. Padhye, and A. Ghotkar, "Analysis of post-harvest losses: An Internet of Things and machine learning approach," in *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, Pune, India, Sep. 2016, pp. 222–226.

[14] Mabrouk, S., Abdelmonsef, A., & Toman, A. (2017). Smart grain storage monitor and control. American Academic Scientific Research Journal for Engineering, Technology, and Sciences, 31(1), 156-162.

[15] R. Kaushik and J. Singhai, "Classification of Quality of Granary using Machine Learning based on Software-Defined Wireless Sensor Network," in *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*, Aug. 2021, pp. 652–656.

[16] M. Hughes, "Understanding and Implementing the HC-12 Wireless Transceiver Module - Projects," Nov. 2016. https://www.allaboutcircuits.com/projects/understanding-andimplementing-the-hc-12-wireless-transceiver-module/

[17] Y. Shimizu, "SnO2 Gas Sensor," in Encyclopedia of Applied Electrochemistry, G. Kreysa, K. Ota, and R. F. Savinell, Eds. New York, NY: Springer, 2014, pp. 1974–1982.

[18] B. Alexander, "Measuring CO2 with MQ135," Pocket Science Lab, Mar. 04, 2021. https://pslab.io/blog/measuring-co2-with-mq135/

[19] T. K. Hareendran, "How To Use MQ-135 Gas Sensor," Codrey Electronics, Jan. 16, 2020. https://www.codrey.com/electronic-circuits/how-to-use-mq-135-gas-sensor/

[20] pslabio, "Measuring CO2 with MQ135," *Pocket Science Lab*, Mar. 04, 2021. https://pslab.io/blog/measuring-co2-with-mq135/ (accessed Apr. 24, 2022).

[21] "MG811 pdf, MG811 Description, MG811 Datasheet, MG811 view::: ALLDATASHEET ::":, *pdf1.alldatasheet.com*.https://pdf1.alldatasheet.com/datasheet-pdf/view/172004/ETC2/MG811.html (accessed Apr. 23, 2022).

[22] R. Kaushik and S. Jyoti, "Sensing Technologies used for Monitoring and Detecting Insect Infestation in Stored Grain," International Journal of Engineering and Technology (UAE), vol. 7, no. 4, pp. 169–173, Sep. 2018, doi: 10.14419/ijet.v7i4.6.20456.

[23] Laerd Statistics, "Mann-Whitney U Test in SPSS Statistics | Setup, Procedure & Interpretation | Laerd Statistics," *Laerd.com*, 2013. https://statistics.laerd.com/spss-tutorials/mann-whitney-u-test-using-spss-statistics.php

[24] "Logistic Regression 3-class Classifier," *scikit-learn*.

[25] S. Li, "Building A Logistic Regression in Python, Step by Step," *Towards Data Science*, Sep. 29, 2017.

[26] T. Yiu, "Understanding Random Forest," *Medium*, Jun. 12, 2019.

[27] S. E R, "Random Forest | Introduction to Random Forest Algorithm," *Analytics Vidhya*, Jun. 17, 2021. https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

[28] W. Koehrsen, "Hyperparameter Tuning the Random Forest in Python," *Medium*, Jan. 10, 2018. https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

[29] O. Harrison, "Machine Learning Basics with the K-Nearest Neighbors Algorithm," *Medium*, Sep. 10, 2018. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

[30] V. Aliyev, "Gradient Boosting Classification explained through Python," *Medium*, Oct. 07, 2020. https://towardsdatascience.com/gradient-boosting-classification-explained-through-python-60cc980eeb3d

[31] Analytics Vidhya, "Guide to Hyperparameter Tuning in Gradient Boosting (GBM) in Python," *AnalyticsVidhya*,Aug.02,2019. https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/

[32] K. Nighania, "Various ways to evaluate a machine learning models performance," Medium, Jan. 30, 2019.

[33] A. Bose, "Cross Validation — Why & How," *Medium*, Sep. 20, 2019. https://towardsdatascience.com/cross-validation-430d9a5fee22#:~:text=Cross%20validation%20is%20a%20technique

[34] Abhigyan, "Cross-Validation Techniques," Geek Culture, Aug. 31, 2021.)

[35] K. S. V. Muralidhar, "Learning Curve to identify Overfitting and Underfitting in Machine Learning," *Medium*, Feb. 22, 2021. https://towardsdatascience.com/learning-curve-to-identify-overfitting-underfitting-problems-133177f38df5

[36]A. Olteanu, "Tutorial: Learning Curves for Machine Learning in Python for Data Science," *Dataquest*, Jan. 03, 2018. https://www.dataquest.io/blog/learning-curves-machine-learning/

[37]"Comparison of Calibration of Classifiers," *scikit-learn*. https://scikit-learn.org/stable/auto_examples/calibration/plot_compare_calibration.html?highlight=learning%20curve (accessed Apr. 23, 2022).

[38] "Learning Curve — Yellowbrick v1.4 documentation." https://www.scikit-yb.org/en/latest/api/model_selection/learning_curve.html (accessed Apr. 24, 2022).

[39] M. Hughes, "Understanding and Implementing the HC-12 Wireless Transceiver Module - Projects," Nov. 2016. https://www.allaboutcircuits.com/projects/understanding-andimplementing-the-hc-12-wireless-transceiver-module/ (accessed Apr. 04, 2022).

Appendix

Codes can be found: https://github.com/Geeky-Wendy/Wendi_Rahinatu.git