

SMART COOP MONITORING SYSTEM FOR
POULTRY FARMS BASED ON INTERNET OF
THINGS

WILFRED CHEAH SENG WEI

Bachelor Of Computer Science (Software
Engineering) With Honors

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : WILFRED CHEAH SENG WEI :

Date of Birth

Title : SMART COOP MONITORING SYSTEM FOR POULTRY FARMS BASED ON INTERNET OF THINGS

Academic Session : 2022/2023 SEMESTER 2

I declare that this thesis is classified as:

- CONFIDENTIAL (Contains confidential information under the Official Secret Act 1997)*
- RESTRICTED (Contains restricted information as specified by the organization where research was done)*
- OPEN ACCESS I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

(Student's Signature)

(Supervisor's Signature)

DR. MOHD ZAMRI BIN OSMAN

Name of Supervisor

Date: 22/02/2023

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.



SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Bachelor of Computer Science (Software Engineering)

A handwritten signature in black ink, appearing to read 'Zamri', is written over a horizontal line.

(Supervisor's Signature)

Full Name : DR MOHD ZAMRI BIN OSMAN
Position : SENIOR LECTURER
Date : 22/02/2023



STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

A handwritten signature in black ink, appearing to read "Wilfred Cheah Seng Wei", is written above a horizontal line.

(Student's Signature)

Full Name : WILFRED CHEAH SENG WEI

ID Number : CB20028

Date : 15 January 2023

SMART COOP MONITORING SYSTEM FOR POULTRY FARMS
BASED ON INTERNET OF THINGS

WILFRED CHEAH SENG WEI

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Bachelor's Degree In Computer Science
(Software Engineering) With Honors

Faculty of Electrical & Electronics Engineering
UNIVERSITI MALAYSIA PAHANG

JANUARY 2023

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Ts. Dr. Mohd Zamri Bin Osman, for his unwavering support, guidance, and encouragement throughout the completion of this thesis. I would like to extend my appreciation to my family and friends for their love and support. Lastly, I would like to acknowledge the resources and facilities provided by Universiti Malaysia Pahang that helped in the completion of this research.

ABSTRAK

Industri penggemukan ayam memainkan peranan penting dalam memberi makan kepada jutaan orang di seluruh dunia. Walau bagaimanapun, kejayaan industri ini bergantung sepenuhnya kepada kesihatan dan produktiviti ayam. Faktor persekitaran seperti suhu, kelembapan dan kualiti udara boleh mempunyai kesan yang signifikan terhadap kesihatan dan produktiviti ayam. Salah satu masalah utama yang dihadapi oleh penggemuk ayam adalah kadar kematian yang tinggi pada tahap pertumbuhan awal. Ini boleh menyebabkan kerugian kewangan yang signifikan bagi ladang dan juga mempengaruhi produktiviti keseluruhan ladang. Sistem pemantauan kandang pintar adalah penyelesaian yang berkesan dan mudah digunakan yang boleh membantu penggemuk ayam untuk mengurangkan kadar kematian ayam pada tahap pertumbuhan awal. Ia boleh memantau keadaan persekitaran dalam kandang dalam masa nyata, termasuk suhu, kelembapan, tahap air, keintenan cahaya dan berat broiler. Selain itu, sistem ini dilengkapi dengan ramalan AI untuk berat broiler, yang boleh membantu penggemuk ayam untuk meramal pertumbuhan broiler dan mengambil tindakan selaras. Sistem ini dirancang untuk memberikan amaran kepada penggemuk ayam mengenai keadaan yang tidak normal dan memberikan data sejarah untuk analisis dan optimisasi persekitaran kandang. Dengan memantau persekitaran dalam kandang, penggemuk ayam boleh memastikan keadaan pertumbuhan yang optimum untuk ayam mereka, mengesan dan mengatasi keadaan yang tidak normal dengan cepat, dan akhirnya, meningkatkan kesihatan dan produktiviti ladang mereka. Tesis ini menyajikan reka bentuk dan pelaksanaan sistem pemantauan kandang pintar untuk ladang penggemukan ayam dan manfaat potensialnya dalam mengurangkan kadar kematian ayam pada tahap pertumbuhan awal.

ABSTRACT

The poultry farming industry plays a vital role in providing food for millions of people worldwide. However, the success of this industry depends heavily on the health and productivity of the poultry. Environmental factors such as temperature, humidity, and air quality can have a significant impact on the health and productivity of the poultry. One of the major issues that poultry farmers face is the high mortality rate of chicken during early stages of growth. This can lead to significant financial losses for the farm and negatively impact the overall productivity of the farm. A smart coop monitoring system is a cost-effective and easy-to-use solution that can help farmers to decrease the mortality rate of chicken during early stages of growth. It can track the environmental conditions inside the coop in real-time, including temperature, humidity, water level, light intensity, and the weight of broiler. Additionally, the system is equipped with AI prediction for weight of broiler, which can help the farmers to predict the growth of broiler and take actions accordingly. The system is designed to alert farmers to any abnormal conditions and provide historical data for analysis and optimization of the coop environment. By monitoring the environment inside the coops, farmers can ensure the optimal growth conditions for their poultry, detect, and address any abnormal conditions in a timely manner, and ultimately, improve the health and productivity of their farm. This thesis presents the design and implementation of a smart coop monitoring system for a poultry farm and its potential benefits to decrease the mortality rate of chicken during early stages of growth.

TABLE OF CONTENT

DECLARATION	
TITLE PAGE	
ACKNOWLEDGEMENTS	ii
ABSTRAK	iii
ABSTRACT	iv
TABLE OF CONTENT	v
LIST OF TABLES	vii
LIST OF FIGURES	viii
LIST OF ABBREVEATIONS	x
CHAPTER 1 INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	2
1.3 Objective	3
1.4 Scope	3
1.5 Thesis Organization	4
CHAPTER 2 LITERATURE REVIEW	5
2.1 Introduction to Smart Coop Monitoring System	5
2.2 Existing Systems	5
2.2.1 The Broiler Chicken Coop Temperature Monitoring Use Fuzzy Logic and LoRaWAN	5
2.2.2 Internet of Things and Machine Learning Techniques In Poultry Health and Welfare Management	7
2.2.3 A Robust Internet of Things-Based Aquarium Control System Using Decision Tree Regression Algorithm	8
2.3 Comparison of Existing Systems	10
2.4 Summary	12
CHAPTER 3 METHODOLOGY	13
3.1 Introduction	13
3.2 Methodology	13
3.3 Project Requirements	14
3.3.1 System Requirements	14
3.3.2 Functional Requirements	19
3.3.3 Non-functional Requirements	19

3.3 Project Design	19
3.4.1 System Sketch	20
3.4.2 System Operation	21
3.4.3 IoT Architecture	27
3.4.4 Circuit Diagram	32
3.4.5 Dialogue Diagram	32
3.4.6 Database Design	33
3.4.7 Proposed AI Technique	35
3.4.8 UI/UX Design	37
3.4 Gantt Chart	38
CHAPTER 4 IMPLEMENTATION, RESULT AND DISCUSSION	
4.1 Introduction	40
4.2 Implementation Process and Results	40
4.2.1 Data Collection	40
4.2.2 Database Implementation	42
4.2.3 Hardware Development	43
4.2.4 Web Application Development	45
4.2.5 Web Server Hosting	45
4.2.6 MQTT Broker	46
4.2.7 AI Server	47
4.2.8 WebSocket Server	48
4.2.9 AI Server	48
4.2.10 Dashboard Implementation	51
4.2.10.1 Charts	52
4.2.10.2 Alerts	52
4.2.10.3 Controls	54
4.2.10.4 Analytics	55
4.3 Discussion	56
4.3 Summary	56
CHAPTER 5 CONCLUSION	57
5.1 Introduction	57
5.2 Research Constraint	57
5.3 Future Works	57
REFERENCES	59

LIST OF TABLES

Table 2.1	Comparison of Existing System	10
Table 3.1	Hardware Requirements	14
Table 3.2	Software Requirements	19
Table 3.3	Description of Dialogue Diagram	33
Table 3.4	Temperature Table Scheme	34
Table 3.5	Light Intensity Table Scheme	34
Table 3.6	Water Level Table Scheme	34
Table 3.7	Food Weight Table Scheme	34
Table 3.8	Chicken Weight Table Scheme	34
Table 3.9	Alert Table Scheme	35

LIST OF FIGURES

Figure 2.1	Chicken Design Diagram	6
Figure 2.2	Chicken Coop Fuzzy Inference System “Mamdani”	6
Figure 2.3	Overview of IoT And ML in Poultry Health Management	7
Figure 2.4	Smart Poultry Health and Welfare Management Framework	8
Figure 2.5	Aquaculture Sensors: (a) Waterproof temperature sensor (b) TDS sensor (c) Dissolved oxygen sensor	8
Figure 2.6	Aquaculture Actuators: (a) Water heater (b) Fan (c) Automatic fish feeder	9
Figure 2.7	System Block Diagram	9
Figure 2.8	Mobile App User Interface. (a) Setting. (b) Monitoring	10
Figure 3.1	Rapid Application Development (RAD)	13
Figure 3.2	SCMS Block Diagram	20
Figure 3.3	SCMS System Sketch	21
Figure 3.4	Flowchart of Welcome Page	22
Figure 3.5	Flowchart of Login Page	22
Figure 3.6	Flowchart of Register Page	23
Figure 3.7	Flowchart of Analytics Page	23
Figure 3.8	Flowchart of Dashboard Page	24
Figure 3.9	Flowchart of Arduino Uno reading sensor data	25
Figure 3.10	Flowchart of Raspberry Pi publish data using MQTT and controlling actuator	25
Figure 3.11	Flowchart of MQTT Subscriber insert data into database	26
Figure 3.12	Flowchart of AI Prediction using One-Step Ahead Technique	26
Figure 3.13	SCM System Architecture	27
Figure 3.14	Establish Connection Arduino to Raspberry Pi	28
Figure 3.15	Establish Connection to MQTT broker	29
Figure 3.16	Publish Data to Topic	29
Figure 3.17	Establish Connection to Subscribe	30
Figure 3.18	Subscribe to Topic	30
Figure 3.19	Establish Connection with Pusher	30
Figure 3.20	Establish Connection with Pusher Client Side	31
Figure 3.21	Pusher Client Trigger	31
Figure 3.22	Circuit Diagram of SCMS	32

Figure 3.23	Dialogue Diagram of SCMS	33
Figure 3.24	Example data of NARX inputs and output	36
Figure 3.25	Dashboard UI	37
Figure 3.26	Control panel and Alarm logs	38
Figure 3.27	Analytics UI	38
Figure 3.28	User Profile UI	38
Figure 4.1	Publish Collected Data Snippet	42
Figure 4.2	Database for Data Collection	42
Figure 4.3	SCMS Database	43
Figure 4.4	Insert to DB Snippet	43
Figure 4.5	Database Table Structure	43
Figure 4.6	Hardware Setup	44
Figure 4.7	Read from Serial	44
Figure 4.8	Subscribe MQTT Topic	44
Figure 4.9	WebSocket Connection	45
Figure 4.10	Web Hosting Setup	45
Figure 4.11	FileZilla Interface	46
Figure 4.12	MQTT Broker	47
Figure 4.13	AI Server	47
Figure 4.14	WebSocket Server	48
Figure 4.15	Training Parameters	49
Figure 4.16	Training Set	49
Figure 4.17	Comparison Set	50
Figure 4.18	Residual Plot	50
Figure 4.19	Temperature Chart Setup	51
Figure 4.20	Temperature Chart Setup	52
Figure 4.21	Connection Time Check	52
Figure 4.22	Alarm History	53
Figure 4.23	Discord Webhook	53
Figure 4.24	Discord Webhook Implementation	53
Figure 4.25	Discord Notification	54
Figure 4.26	Control Hardware	54
Figure 4.27	Raspberry Pi Control Side	55
Figure 4.28	Raspberry Pi Control Snippet	55
Figure 4.29	Prediction Interface	55

LIST OF ABBREVIATIONS

SCMS	Smart Coop Monitoring System
IoT	Internet of Things
AT	Attention
MQTT	Message Queuing Telemetry Transport
OneNET	Open Network for Easier Network Service
GSM	Global System for Mobile Communications
AI	Artificial Inteligence
ML	Machine Learning
CNN	Convolutional Neural Networks
RNN	Reccurent Neural Network
GAN/AE	Generative Adversarial Networks/Auto Encoders
RDL	Reinforcement Deep Learning
NARX	Nonlinear Autoregressive Network with Exogenous Inputs

CHAPTER 1

INTRODUCTION

1.1 Introduction

The poultry farming industry plays a vital role in providing food for millions of people worldwide. However, the success of this industry depends heavily on the health and productivity of the poultry. Environmental factors such as temperature, humidity, and air quality can have a significant impact on the health and productivity of the poultry [10]. Additionally, monitoring the growth of broilers is essential for farmers to ensure their productivity and health. One of the major issues that poultry farmers face is the high mortality rate of chicken during early stages of growth [7]. This can lead to significant financial losses for the farm and also negatively impact the overall productivity of the farm.

A smart coop monitoring system is a cost-effective and easy-to-use solution that can help farmers to decrease the mortality rate of chicken during early stages of growth. It can track the environmental conditions inside the coop in real-time, including temperature, humidity, water level, light intensity, and the weight of the broiler. The system is designed to alert farmers to any abnormal conditions and provide historical data for analysis and optimization of the coop environment. By monitoring the environment inside the coops, farmers can ensure the optimal growth conditions for their poultry, detect and address any abnormal conditions in a timely manner, and ultimately, improve the health and productivity of their farm.

The system is equipped with sensors such as DHT11 for temperature and humidity, Water Level sensor for monitoring water level, Photocell for measuring light intensity and Load Cell for monitoring the weight of the broiler. These sensors are placed inside the coop to collect data on the environmental conditions and the weight of the broiler. The collected data is then transmitted to the database, where it is analyzed by the AI prediction model and made available to farmers in real-time through an IoT dashboard. This feature allows farmers to predict the growth of broilers and take necessary actions to improve the health and productivity of the poultry.

One of the key advantages of a smart coop monitoring system is that it can be configured to maintain the optimal temperature for different age of broiler. At an age 0-7 days, with optimal temperature 33°C, at 7-14 days old, with optimum temperature 30°C, at an age above 14 days, with optimum temperature 27°C (Y. A. Liani et al, 2021)[1]. This feature ensures that the broilers are always in an optimal temperature range for their age which in turn decreases the mortality rate of chicken during early stages of growth.

1.2 Problem Statement

The problem is that poultry farmers currently lack an effective and efficient way to monitor the environment inside the coops [9], ensure the optimal growth conditions for their poultry, and detect and address abnormal conditions in a timely manner. This leads to poor conditions that negatively impact the health and productivity of the poultry, leading to high mortality rate of chicken during early stages of growth, and ultimately, the profitability of the farm. This problem is significant as it results in financial losses for the farm and also negatively impacts the overall productivity of the farm. The proposed solution is to design and implement a smart coop monitoring system that can track the temperature, humidity, water level, light intensity, and weight of broilers in real-time and maintain the optimal temperature range for different age of broiler (0-7 days, 7-14 days, above 14 days) in order to decrease the mortality rate of chicken during early stages of growth. The system should be able to alert farmers to any abnormal conditions, provide historical data for analysis and optimization of the coop environment, be cost-effective, easy to install and maintain, and able to integrate with existing farm management software [8]. The proposed solution will help farmers to optimize the coop environment, improve the health and productivity of their poultry, decrease the mortality rate of chicken during early stages of growth, and ultimately, increase their profitability.

1.3 Objectives

Based on the problem statements, the objectives of the project are:

- i. To study the current challenges faced by poultry farmers in monitoring the environment inside the coops and the growth of broilers.
- ii. To develop a smart coop monitoring system that can track the temperature, humidity, water level, light intensity, and weight of broilers in real-time, and maintain the optimal temperature range for different ages of broilers.
- iii. To evaluate the effectiveness of the developed smart coop monitoring system in improving the health and productivity of the poultry, decreasing the mortality rate of chicken during early stages of growth, and increasing the profitability of the farm.
- iv. To incorporate an artificial intelligence technique to predict the weight of broilers in advance and provide valuable insights to the farmers for decision making.

1.4 Scope

User Scope:

- i. Poultry farmers
- ii. Research institutions

System Scope:

- i. The smart coop monitoring system will be developed for monitoring the temperature, humidity, water level, light intensity, and weight of broilers in real-time.
- ii. The system will be designed to maintain the optimal temperature range for different age of broilers.
- iii. The system will be equipped with an AI prediction model for the weight of broilers, which can help the farmers to predict the growth of broilers and take actions accordingly.

Development Scope:

- i. The smart coop monitoring system will be developed using sensors such as DHT11, Water Level sensor, Photocell and Load Cell.

- ii. The system will be designed to be cost-effective, easy to install and maintain.
- iii. The system will be configured to send alerts and notifications to farmers in case of any abnormal conditions.
- iv. The system will be able to provide historical data for analysis and optimization of the coop environment.
- v. The system will be able to optimize the coop environment and make data-driven decisions to improve the operations of poultry farms.

1.5 Thesis Organization

Chapter 1 of this research provides an overview of the poultry farming industry and highlights the importance of monitoring the growth of broilers, as well as the problem of high mortality rates in chicken during early stages of growth. To address this issue, a smart coop monitoring system is proposed as a solution. Chapter 2 presents an overview of the existing literature on the topic, including the current state of the art and the gaps in the literature that the proposed research aims to fill. In Chapter 3, the research design and methods used in the study are detailed, including the selection of participants, data collection and analysis techniques, and the limitations of the study. The findings of the research are presented in Chapter 4 and are discussed in relation to the research questions and objectives. Lastly, Chapter 5 provides a summary of the main findings of the research and their implications for the poultry farming industry, along with recommendations for future research.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, the review on different existing smart coop monitoring systems that related to this project will be discussed. Three of the existing systems will be chosen as the research, and this literature review will illustrate the description, advantages, disadvantages, and the comparisons of the existing systems based on their features, functions, monitoring capabilities and so forth.

2.2 Existing Systems

In this study there are three related existing systems for monitoring and management in agriculture, including The Broiler Chicken Coop Temperature Monitoring Use Fuzzy Logic and LoRAWAN, Internet of Things and Machine Learning Techniques In Poultry Health and Welfare Management, and A Robust Internet of Things-Based Aquarium Control System Using Decision Tree Regression Algorithm.

2.2.1 The Broiler Chicken Coop Temperature Monitoring Use Fuzzy Logic and LoRaWAN

This system uses LoRa and Fuzzy Logic to monitor temperature and humidity in broiler chicken coops and determine ideal conditions for growth. The system uses sensors, microcontroller and LoRa technology for real-time monitoring and analysis to increase production and improve quality of chickens. [2].

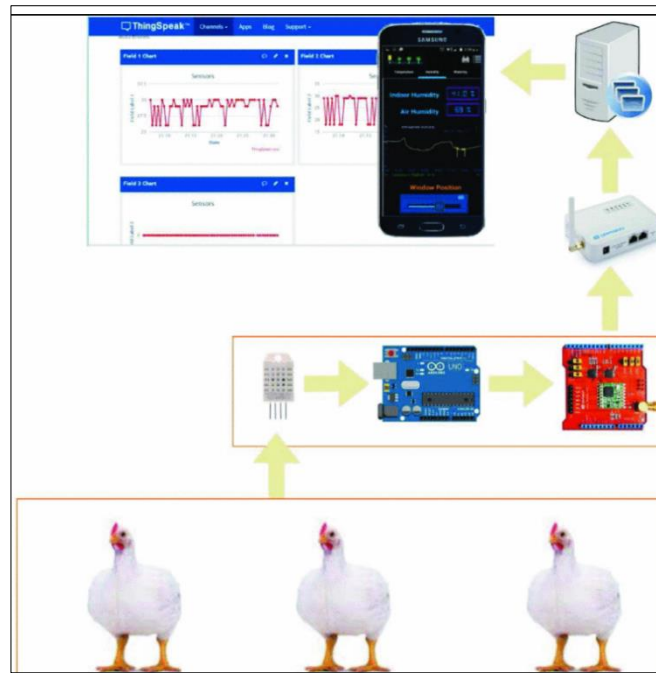


Figure 2.1 Chicken Design Diagram

The study uses LoRaWAN (Long Range Wide Area Network) protocol to transmit temperature and humidity sensor data from the broiler chicken coops to a remote location for monitoring and analysis. LoRaWAN uses a star-of-stars topology where gateways act as intermediaries between the end devices (sensors) and the network server.

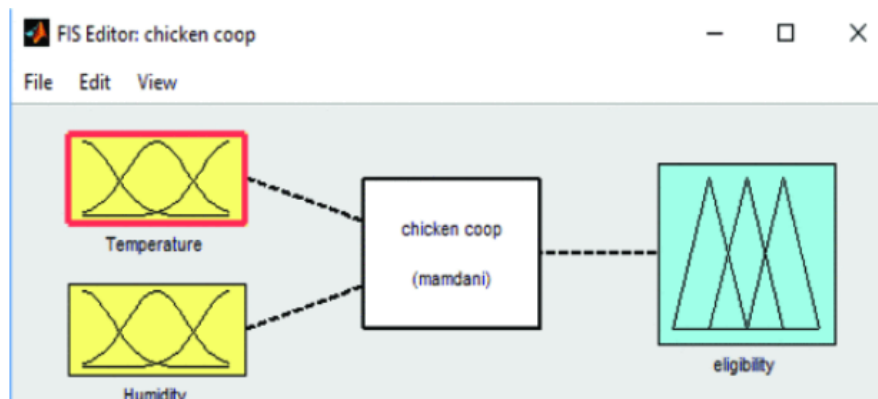


Figure 2.2 Chicken Coop Fuzzy Inference System "Mamdani"

In this study, the Mamdani Fuzzy Logic technique is used to determine the feasibility of temperature and humidity conditions in the broiler chicken coops. The input variables are temperature and humidity, and the output variable is the feasibility of the conditions. The study uses fuzzy sets to describe the ideal conditions for temperature and humidity, and fuzzy rules to determine the feasibility of the conditions based on the input variables.

2.2.2 Internet of Things and Machine Learning Techniques In Poultry Health and Welfare Management

IoT and Machine Learning (ML) techniques have been used in poultry health and welfare management to improve efficiency, monitor, and control environmental conditions, and detect and diagnose diseases in animals. Environmental sensors such as temperature and humidity sensors, as well as contaminant gas sensors, are used to monitor conditions in poultry buildings. Acoustic sensors are used to measure the characteristics of sounds emitted by hens, which can serve as indicators of stress and disease. IoT and ML techniques have been used to improve nutrient utilization, optimize feed for improved production and respond to sudden disease outbreaks [3].

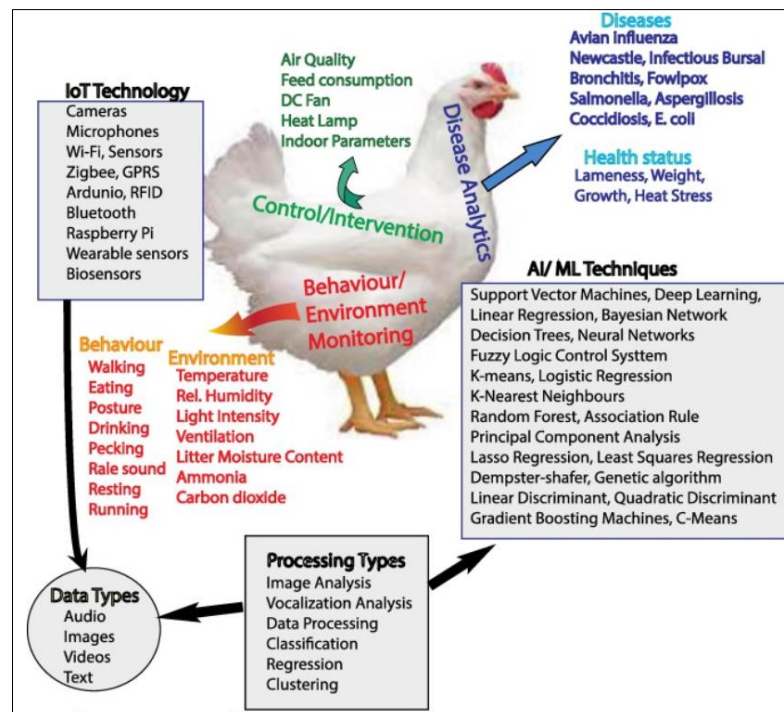


Figure 2.3 Overview of IoT and ML in poultry health management.

AI and IoT techniques are being increasingly used to improve efficiency and productivity. These techniques include the use of deep learning algorithms such as CNN for image and audio classification, RNN for regression operations, GAN/AE for data analysis and feature learning, and RDL for controlling poultry KPI parameters through actuators. These algorithms are used to monitor environmental conditions, detect, and diagnose diseases, track and analyze animal behavior, and optimize feeding and production.

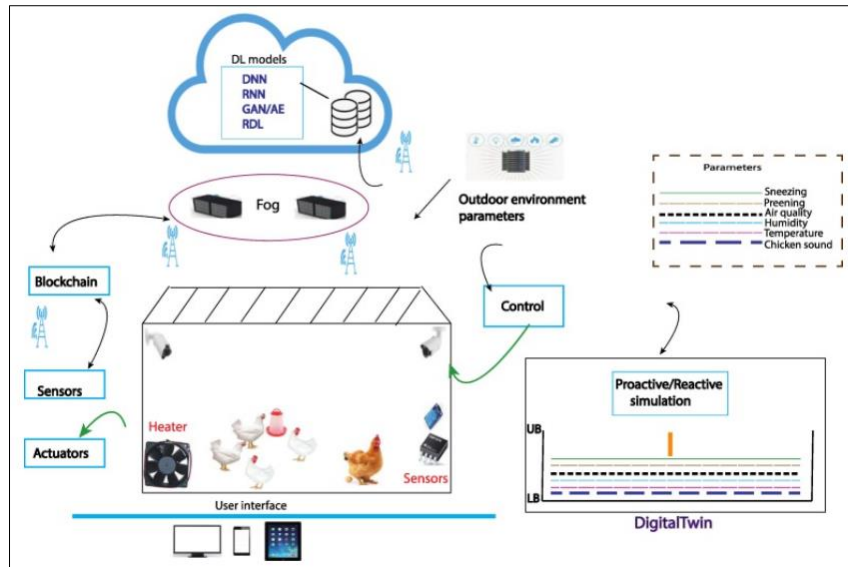


Figure 2.4 Smart poultry health and welfare management framework.

The communication module transmits data and signals in the network using wireless and mobile communication. Additionally, blockchain techniques were proposed to be used to secure the poultry welfare network platform due to the multitude of devices, sensors, and services involved in data collection and transmission.

2.2.3 A Robust Internet of Things-Based Aquarium Control System Using Decision Tree Regression Algorithm

This research proposes a novel Internet of Things (IoT) system for controlling the environment of an aquarium using a decision tree regression (DTR) algorithm for prediction. The system includes sensors for measuring water temperature, TDS, and dissolved oxygen, as well as actuators such as a water heater, fan, relay, and RO filter [4].

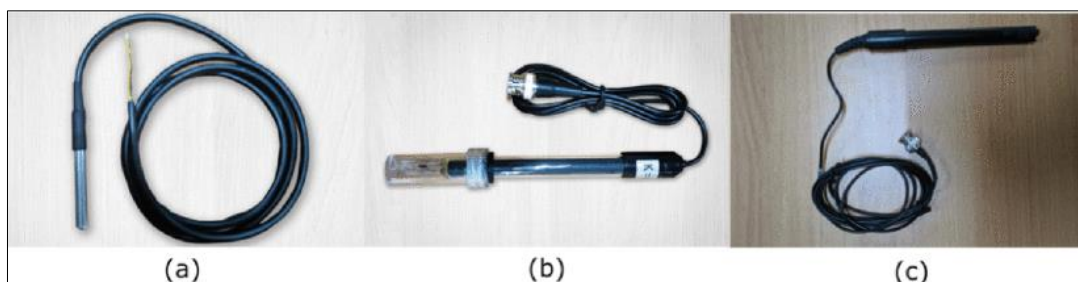


Figure 2.5 Aquaculture sensors: (a) Waterproof temperature sensor (b) TDS sensor (c) Dissolved oxygen sensor.

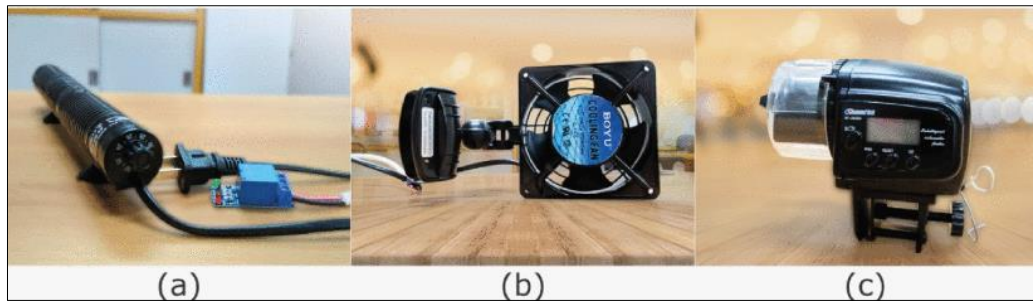


Figure 2.6 Aquaculture actuators: (a) Water heater (b) Fan (c) Automatic fish feeder.

The system includes an array of sensors, actuators, and a Python server, all connected through an android-based application and MQTT communication protocol. It is designed to be delay-tolerant with an analysis model that ensures proper functioning even in high network delay conditions. The system is tested and shown to have a high level of accuracy and can actively provide necessary actions and push information to the user when conditions are near critical.

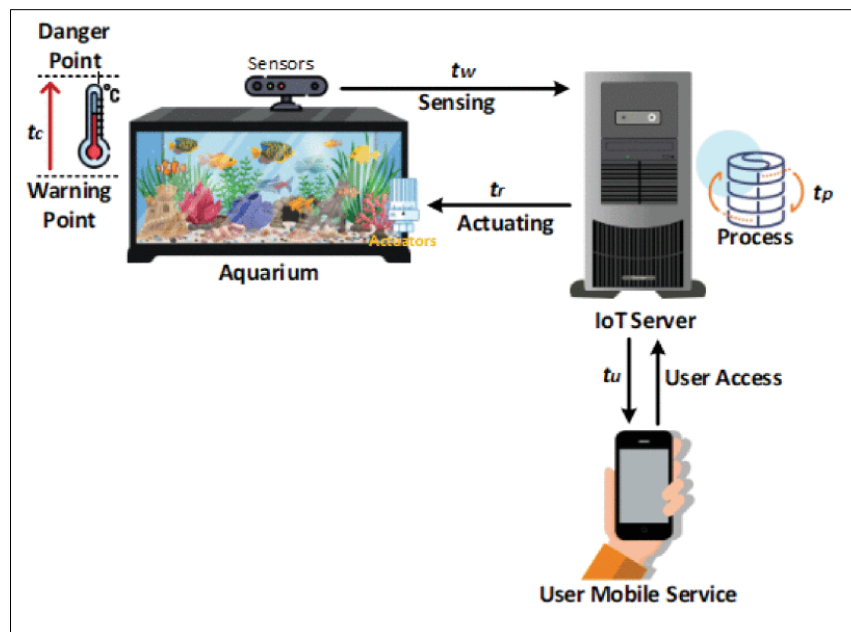


Figure 2.7 System block diagram.

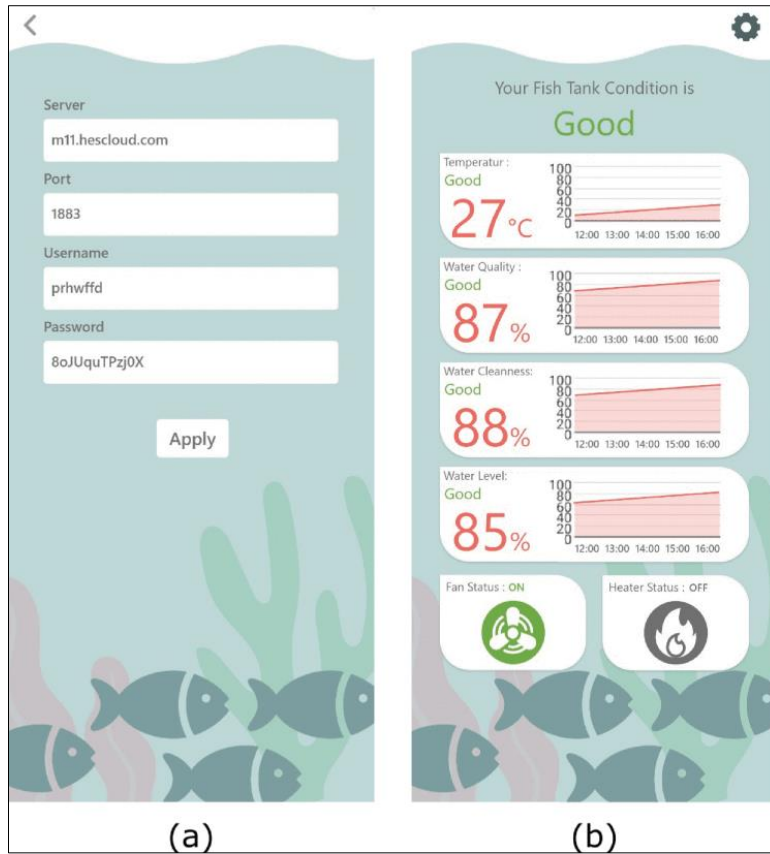


Figure 2.8 Mobile app user interface. (a) Setting. (b) Monitoring.

2.3 Comparison of Existing Systems

System	The Broiler Chicken Coop Temperature Monitoring Use Fuzzy Logic and LoRAWAN	Internet of Things and Machine Learning Techniques In Poultry Health and Welfare Management	A Robust Internet of Things-Based Aquarium Control System Using Decision Tree Regression Algorithm
Features			
Platform	ThinkSpeak & Mobile Application	Mobile & Website Application	Mobile Application
IoT Protocol	LoRaWAN	Not Specified	MQTT

AI Technique	Mamdani Fuzzy Logic	CNN, RNN, GAN/AE, RDL	Decision Tree Regression (DTR)
Database	No database	Not Specified	Not Specified
Features	<ul style="list-style-type: none"> - The system uses LoRaWAN technology to transmit sensor data over long distances. - The system uses Thingspeak as a LoRa Application Server which allows the data to be viewed in real-time on a web interface. - The system uses Mamdani Fuzzy Inference System to determine the feasibility of temperature and humidity conditions in the coops. 	<ul style="list-style-type: none"> - Cloud-fog-based computing for efficient data management and reduced latency. - Blockchain techniques for secure data collection and transmission. - Deep learning techniques such as CNN, RNN, GAN/AE, and RDL for classification, regression, data analysis, feature learning, and control of poultry KPI parameters through actuators. 	<ul style="list-style-type: none"> - Uses a Decision Tree Regression (DTR) algorithm for prediction. - Has a high level of accuracy in prediction.

<p>Limitations</p>	<ul style="list-style-type: none"> - There is limited data storage if Thingspeak is used as a LoRa Application Server, it can only store data for a limited time period. 	<ul style="list-style-type: none"> - Lack of interpretability of deep learning models, which makes it difficult to understand why a model performs differently on different datasets. - Conventional controllers' reliance on user knowledge of the system and complicated rules. 	<ul style="list-style-type: none"> - Relies on a stable and reliable internet connection for communication between the sensors, server, and actuators.
--------------------	---	---	---

Table 2.1 Comparison of Existing Systems

2.4 Summary

The first system, "The Broiler Chicken Coop Temperature Monitoring Use Fuzzy Logic and LoRAWAN", uses a combination of Fuzzy Logic and LoRAWAN protocol for monitoring the temperature in a broiler chicken coop. The second system, "Internet of Things and Machine Learning Techniques In Poultry Health and Welfare Management", uses IoT and machine learning techniques for monitoring and improving the health and welfare of poultry. The third system, "A Robust Internet of Things-Based Aquarium Control System Using Decision Tree Regression Algorithm", uses a decision tree regression algorithm and IoT for controlling the environment in an aquarium. All these systems use a combination of IoT technology, sensors, and actuators to collect and act on data in real-time. They also use machine learning techniques such as fuzzy logic, decision tree regression, and quantile regression forest to make predictions and decisions.

Overall, my system is relevant and significant as it utilizes similar technology and techniques as the three systems discussed, but with a specific focus on individual chicken growth. The comparison highlights the importance and potential of using IoT and AI in animal management, specifically in monitoring and predicting growth and health.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In this chapter, the methodology used in this project will be discussed in detail. It will include a description of the research design, data collection and analysis methods, as well as the procedures and techniques used to gather and analyze data. The methodology will also include information on the tools and equipment used in the project, as well as the limitations and potential sources of error.

3.2 Methodology

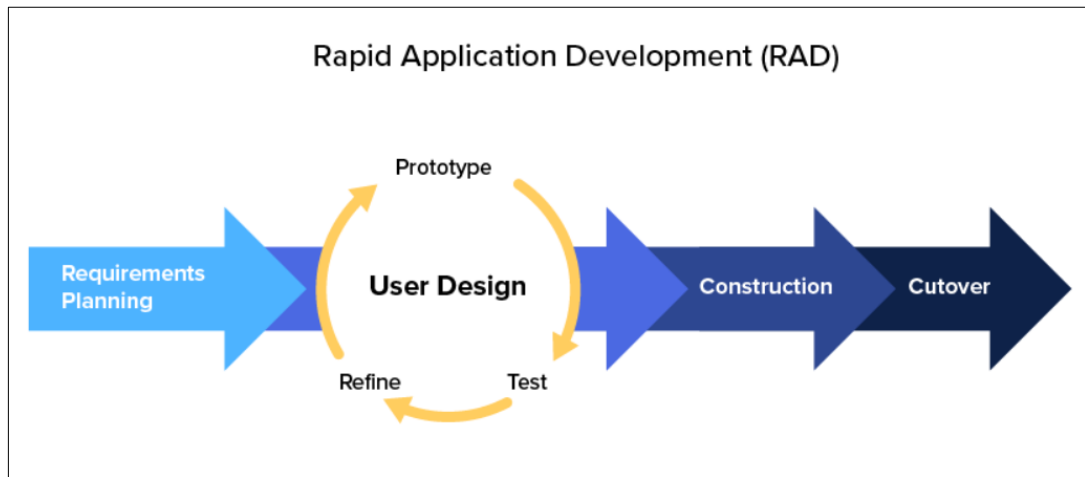


Figure 3.1 Rapid Application Development (RAD)

In this project, the Rapid Application Development (RAD) methodology is employed to guide the development process. The first phase of RAD, I identified and gathered the functional and non-functional requirements for the system and determined the objectives and goals for the project based on a comparison of similar existing systems which has been explained in Chapter 1.4 and Chapter 2. Following this, the iterative User Design phase is implemented, which consists of three sub-phases, creating a prototype, testing it, and refining

it. This will be further elaborated in Chapter 3.4 Proposed Design. Once satisfied with the prototype, the Construction phase begins, where the SCMS is being developed. Finally, the Cutover phase is executed, during which the system is deployed and maintained as a fully developed system.

3.3 Project Requirements

This section explains the project requirements that need to be met for the project to be successful. These requirements are identified based on the project scope and user needs and are used to guide the development process.

3.3.1 System Requirements

Table 3.1 and table 3.2 depicts the hardware and software required for the development.

No.	Hardware	Specifications	Price	Purpose
1	Illegear Raven SE	<ul style="list-style-type: none"> • Windows 10 • Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz • 16.0 GB RAM • 64-bit operating system • NVIDIA GeForce GTX 1650 	RM3900	Used for development of the system.
2	Raspberry Pi 3B+	<ul style="list-style-type: none"> • Broadcom BCM2837B0, 64-bit ARM Cortex-A53 Quad Core Processor SoC running @ 1.4GHz, with metal body for better heat dissipation. • 1GB LPDDR2 SDRAM • 4 x USB2.0 Ports with up to 1.2A output • Expanded 40-pin GPIO Header 	RM156	Used to control actuator and receive data from Arduino Uno to send to MQTT Broker.

		<ul style="list-style-type: none"> • Video/Audio Out via 4-pole 3.5mm connector, HDMI, CSI camera, or Raw LCD (DSI) • Storage: MicroSD • Gigabit Ethernet over USB 2.0 (maximum throughput of 300 Mbps) • Power-over-Ethernet (PoE) support (requires separate PoE HAT) • 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE • Low-Level Peripherals: <ul style="list-style-type: none"> - 27 x GPIO - UART - I2C bus - SPI bus with two chip selects - +3.3V - +5V - Ground • Power Requirement, 5V/2.5A via Micro B USB connector. • Supports Raspbian, Windows 10 IoT Core, OpenELEC, OSMC, Pidora, Arch Linux, RISC OS, and More 		
--	--	---	--	--

		<ul style="list-style-type: none"> • Dimensions: 85mm x 56mm x 17mm 		
3	Arduino Uno Rev 3	<ul style="list-style-type: none"> • Microcontroller ATmega328P • Operating Voltage 5V • Input Voltage (recommended)7-12V • Input Voltage (limits) 6-20V • Digital I/O Pins 14 (of which 6 provide PWM output) • Analog Input Pins 6 • DC Current per I/O Pin 40 mA • DC Current for 3.3V Pin 50 mA • Flash Memory 32 KB (ATmega328) of which 0.5 KB used by the bootloader • SRAM 2 KB (ATmega328) • EEPROM 1 KB (ATmega328) • Clock Speed 16 MHz 	RM115	Used to collect sensor data.
4	Temperature Sensor	<ul style="list-style-type: none"> • DHT11 • Supply Voltage: +5 V • Temperature range: 0-50 °C error of ± 2 °C 	RM4.90	Used to measure the environment temperature and humidity

		<ul style="list-style-type: none"> • Humidity: 20-90% RH \pm 5% RH error • Interface: Digital • Size: 30*20mm 		
5	Water Level Sensor	<ul style="list-style-type: none"> • Operating voltage: DC5V • Operating current: < 20mA • Sensor type: Analog • Detection area: 40mm x16mm • Working Temperature: 10°C-30°C • Working Humidity: 10%-90% without condensation • Weight: 3g • Dimensions: 65mm x 20mm x 8mm 	RM4.60	Used to measure the amount of water in the tank
6	Load Cell	<ul style="list-style-type: none"> • Material: Aluminum • Dimensions: 75×12.7×12.7mm (L x W x H) • Range: 10kg • Rated output 1.0 \pm 0.1mV/V • Nonlinear \pm 0.03% F.S • Hysteresis 0.03% F.S • Repeatability 0.03% F.S • Zero Balance \pm 0.1 mV / V • Input impedance 1066 \pm 10% Ω • Output Impedance 1000 \pm 10% Ω 	RM21.90	Used to measure the weight of the feed

		<ul style="list-style-type: none"> • Insulation resistance: 2000 MΩ • Weight:25g 		
7	Photocell	<ul style="list-style-type: none"> • Interface type: analog • Working voltage: 5V • Size: 30*20mm • Weight: 3g 	RM14.80	Used to measure the surrounding light intensity
8	Container	<ul style="list-style-type: none"> • ABS Material • 	RM41.90	Used as a makeshift coop
9	Heating Lamp	<ul style="list-style-type: none"> • ABS Material • 14cm Size • 100-300W • 220V • 60Hz 	RM28.60	Used to regulate temperature of coop
10	DC Fan	<ul style="list-style-type: none"> • 5V • 2400 rpm • 8cm x 8cm x 2.5cm 	RM13.50	Used to regulate temperature of coop
11	Food & Water Dispenser	<ul style="list-style-type: none"> • Polyethylene • 16cm x 14cm x 7cm 	RM5.60 x2	Used to dispense food and water automatically
12	Server	<ul style="list-style-type: none"> • Websocket Server • MQTT Broker • AI Server 	RM1900	Used to host web application, managing data and run AI models
		Total Price:	RM6212.40	

Table 3.1 Hardware Requirements

No.	Software	Purpose
1	Microsoft Visual Studio Code	Used to program and develop the SCMS
2	Laravel	A PHP web application framework for SCMS web application.
3	XAMPP - phpMyAdmin	Used to manage and interact with MySQL databases through a web interface.
4	Eclipse Mosquitto	Allows the communication between different devices and software using the MQTT protocol.
5	Google Chrome	A web browser for web application implementation.

Table 3.2 Software Requirements

3.3.2 Functional Requirements

1. The system should be able to monitor the environment of the coop in real-time.
2. The system should be able to send alerts and notifications to the users in case of any abnormal conditions.
3. The system should allow users to remotely control the coop's environment, such as adjusting temperature or lighting.
4. The system should be able to collect data from various sensors and store it for future analysis.

3.3.3 Non-functional Requirements

1. The system should be able to handle a large number of sensors and data without compromising performance.
2. The system should be secure to prevent unauthorized access and protect data from cyber threats.
3. The system should be reliable and able to operate continuously without any major interruptions.

3.4 Project Design

This section provides an overview of the project design, including the system architecture, technologies used, and overall design approach.

3.4.1 System Sketch

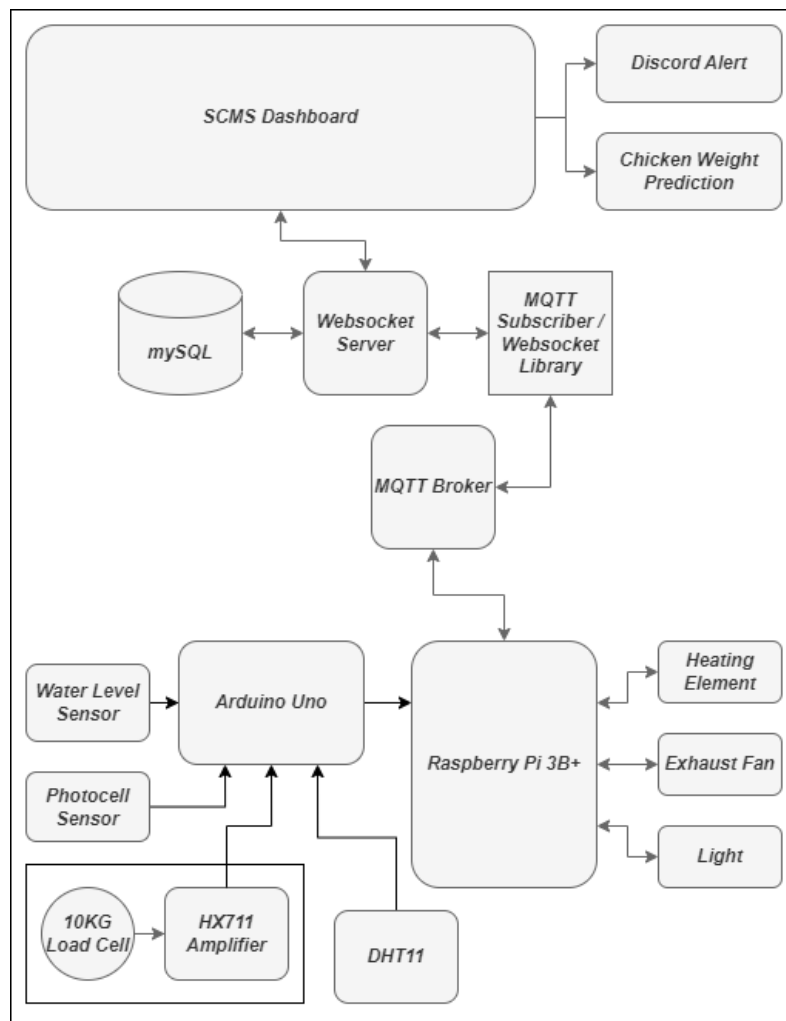


Figure 3.2 SCMS Block Diagram

The block diagram, illustrated in Figure 3.2, provides a visual representation of the components and their relationships within the Smart Coop Monitoring System (SCMS). The diagram shows how the different elements of the system interact with each other to achieve the desired outcome.

MQTT is used as the messaging protocol for transmitting data between the various sensors and devices in the coop. WebSocket is used to establish a connection between the system and the internet, allowing for real-time data transfer and remote monitoring and control. MySQL is used as the database management system, storing, and organizing the data collected by the sensors. Laravel is the framework used for developing the system, providing a structured and efficient way to build and maintain the system's functions and features.

Overall, the use of MQTT, WebSocket, MySQL, and Laravel in the Smart Coop Monitoring System allows for seamless connectivity and development, ensuring that the system is able to accurately and efficiently collect, transmit, and store data, and can be easily updated and maintained as needed.

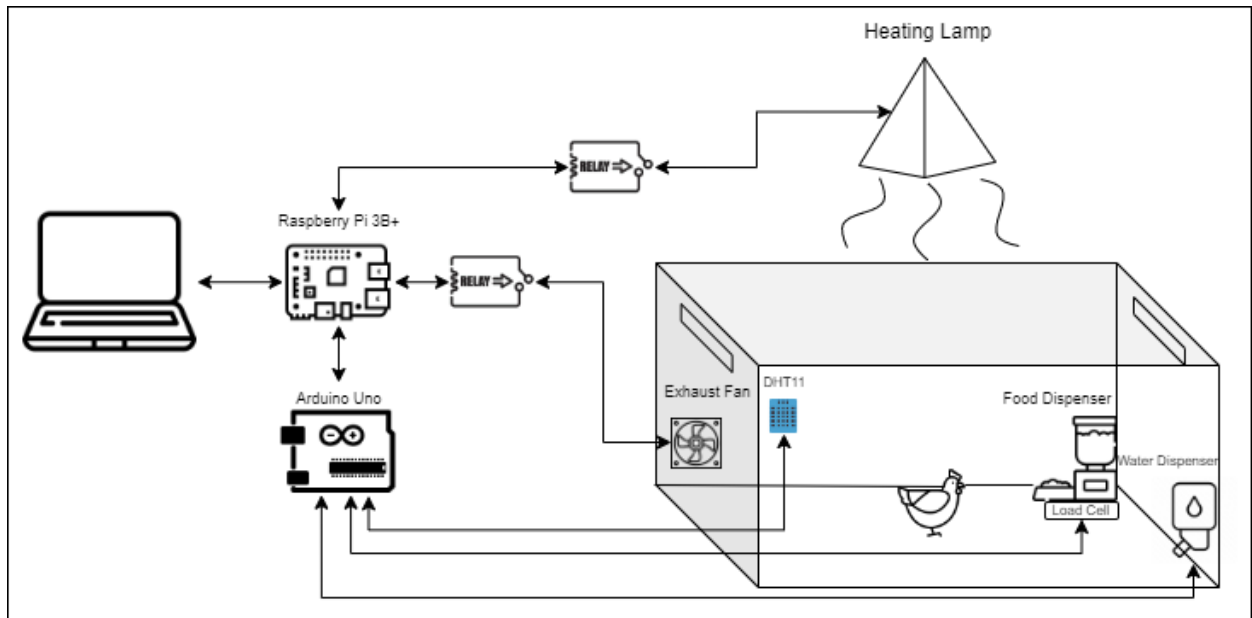


Figure 3.3 SCMS System Sketch

3.4.2 System Operation

This section will describe the system operation using a flowchart. The flowchart will provide a visual representation of the processes and steps involved in the operation of Smart Coop Monitoring System (SCMS).

The following are the flowchart of the system operation:

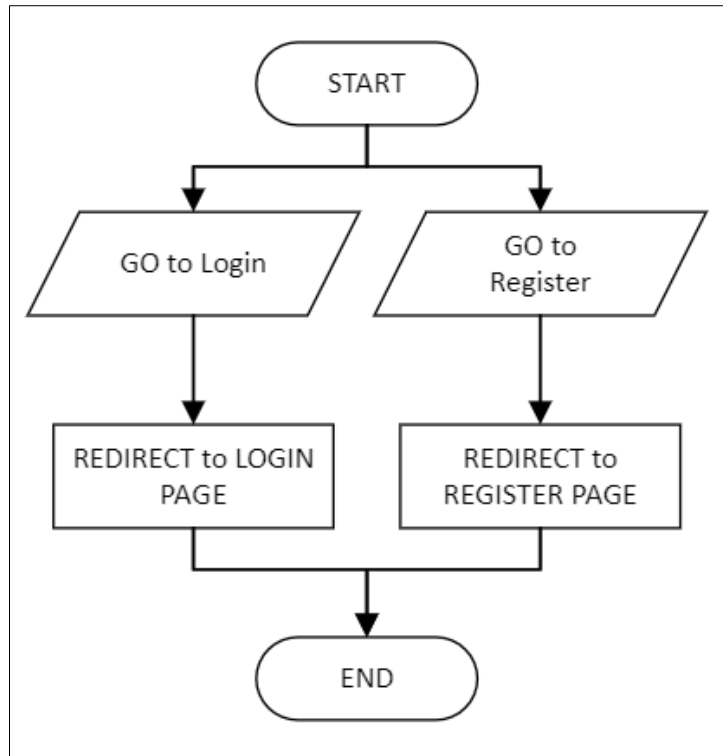


Figure 3.4 Flowchart of Welcome Page

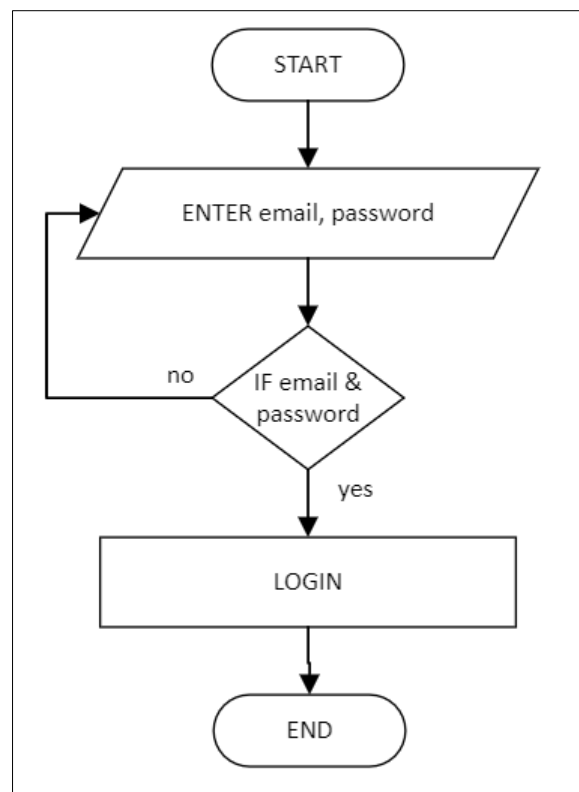


Figure 3.5 Flowchart of Login Page

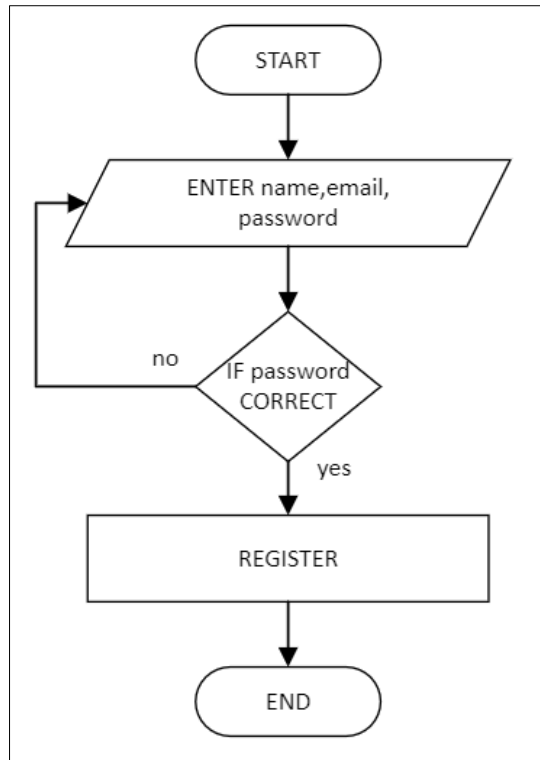


Figure 3.6 Flowchart of Register Page

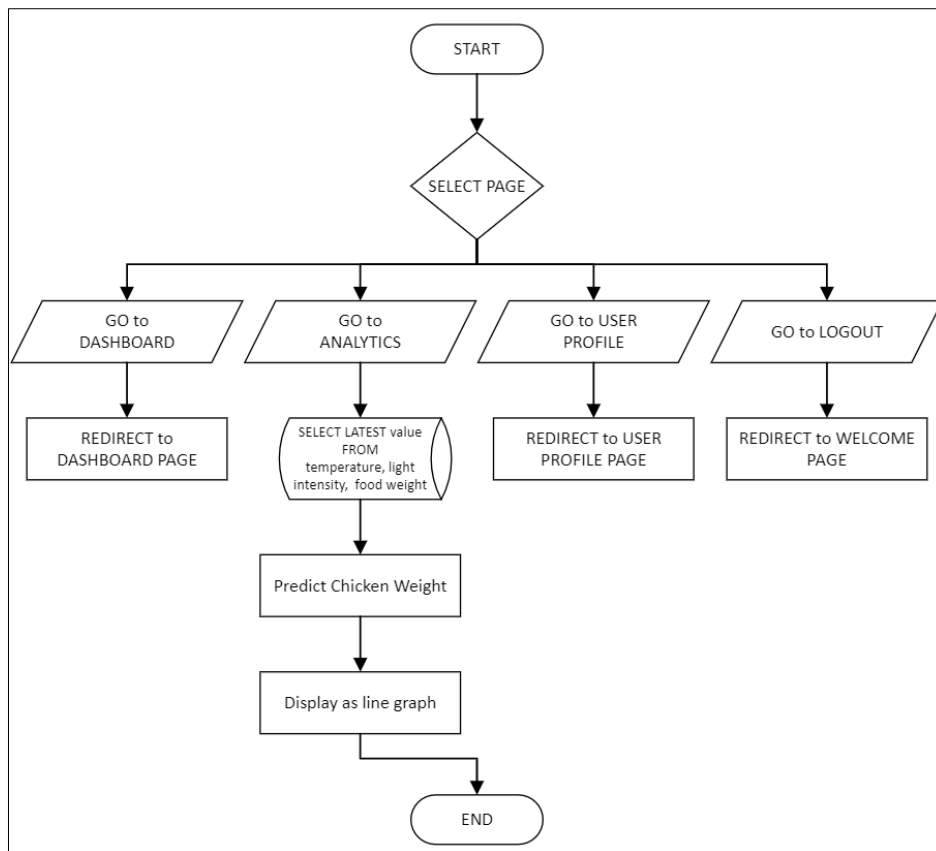


Figure 3.7 Flowchart of Analytics Page

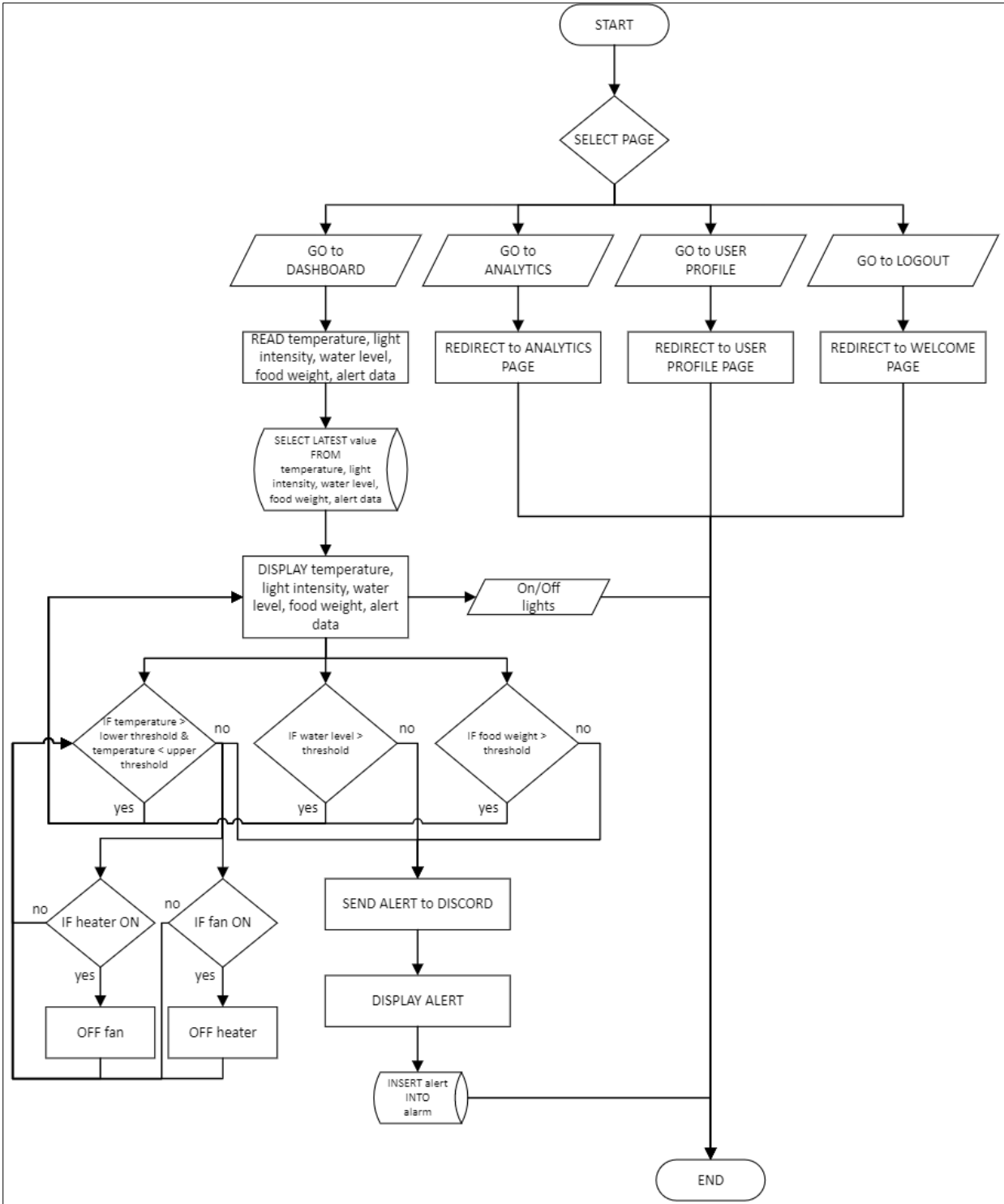


Figure 3.8 Flowchart of Dashboard Page

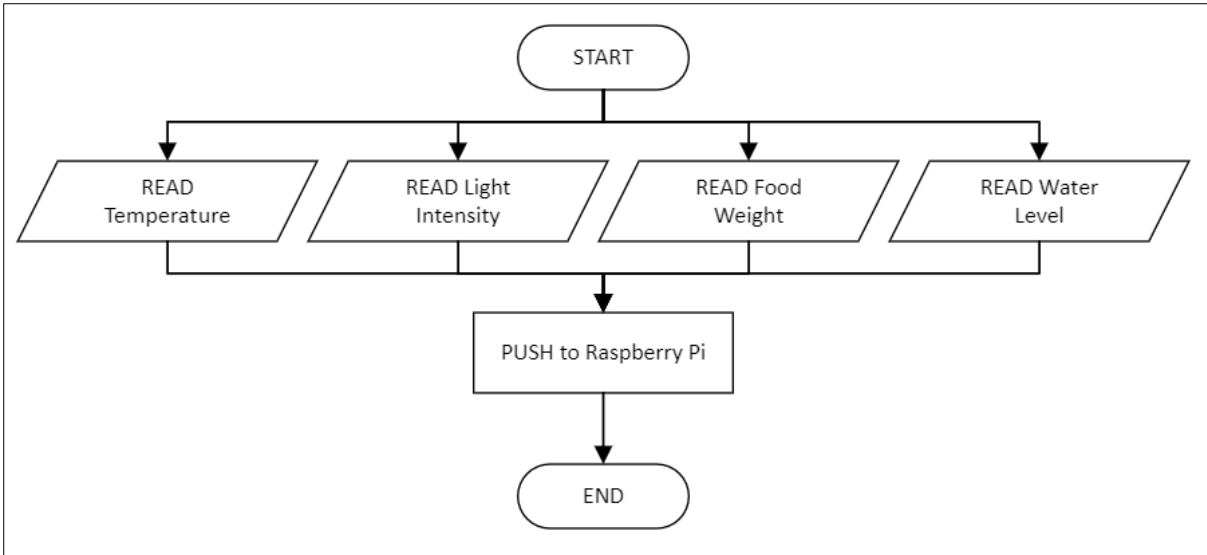


Figure 3.9 Flowchart of Arduino Uno reading sensor data

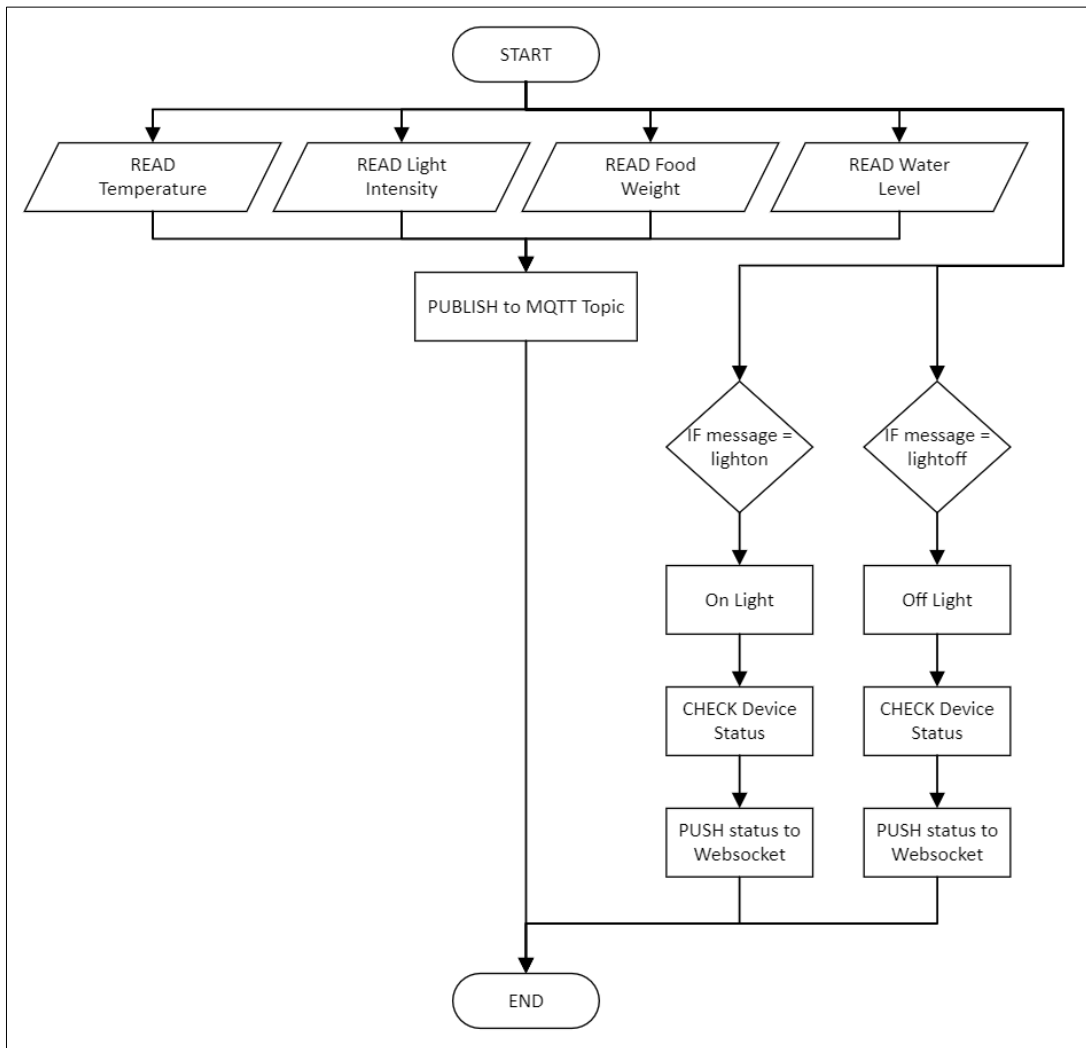


Figure 3.10 Flowchart of Raspberry Pi publish data using MQTT and controlling actuator

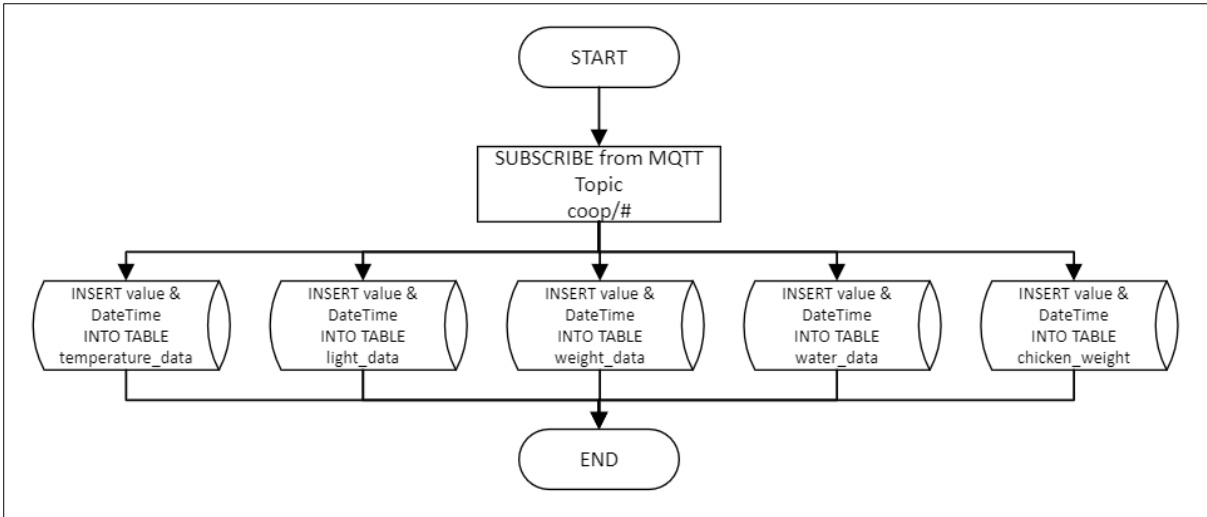


Figure 3.11 Flowchart of MQTT Subscriber insert data into database

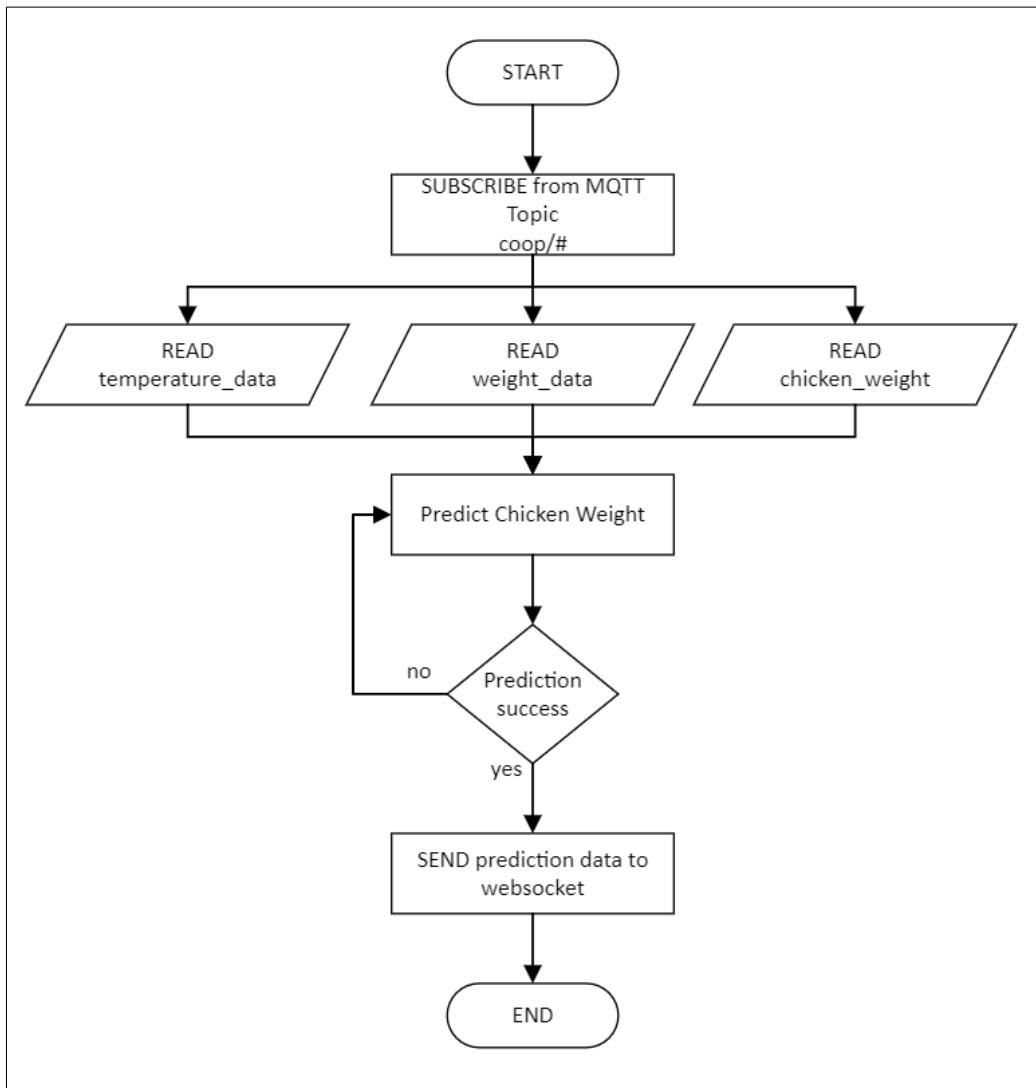


Figure 3.12 Flowchart of AI Prediction using One-Step Ahead Technique

3.4.3 IoT Architecture

This section provides an overview of the IoT architecture for the Smart Coop Monitoring System (SCMS). The architecture outlines the components, interfaces, and relationships between the components that make up the system, and the overall structure of the system and how it will operate.

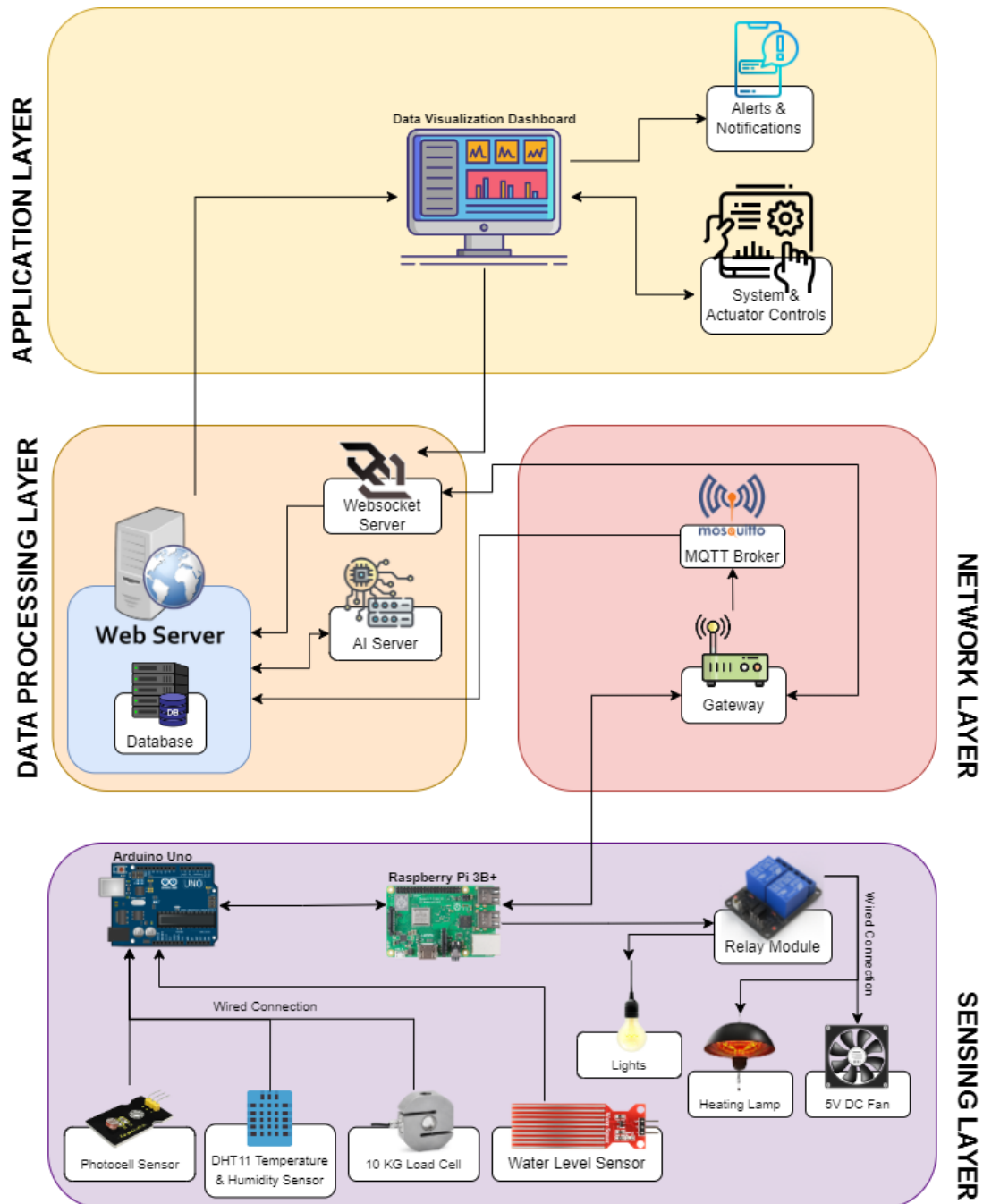


Figure 3.13 SCM System Architecture

The Sensing Layer is a critical component of the Smart Coop Monitoring System as it is responsible for collecting environmental data and sending it to the Network Layer. In the Sensing Layer, various sensors such as DHT11, 10kg Load Cell, Water Level Sensor, and Photocell Sensor are connected to an Arduino Uno microcontroller. The DHT11 sensor is used to measure temperature and humidity in the coop, which are important environmental factors that affect the well-being of the chickens. The 10kg Load Cell sensor is used to measure the weight of the food, which is important for predicting the growth and health of the chickens. The Water Level Sensor is used to monitor the water levels in the coop, ensuring that the chickens have access to enough water. The Photocell Sensor is used to measure the amount of light in the coop, which is important for regulating the lighting in the coop. In addition to the sensors, the Sensing Layer also includes a Relay Module that is connected to the heating lamp and a 5V DC fan. The Relay Module allows the Raspberry Pi to control the heating lamp, fan, and lights ensuring that the environment in the coop is comfortable for the chickens. The connection from Arduino to Pi is established as shown in Figure 3.14.

```
#import all libraries
from operator import truediv
import paho.mqtt.client as mqtt
from random import randrange, uniform
import time
import serial

mkrwifi1010 = serial.Serial('/dev/ttyUSB0', 9600, timeout=.1)
print("Established serial connection to mkrwifi1010")
```

Figure 3.14 Establish Connection Arduino to Raspberry Pi

The network layer is responsible for transmitting data from the sensing layer to the data processing layer and vice versa. In the network layer, the main component is the gateway which acts as a bridge between the Raspberry Pi and the MQTT broker. The Raspberry Pi collects sensor readings from the Arduino Uno and sends it to the gateway. The gateway then sends this data to the MQTT broker. The MQTT broker receives data from the gateway and acts as a middleman, allowing the data to be transmitted to the data processing layer. The MQTT broker uses a publish/subscribe pattern, where the Raspberry Pi acts as a publisher and the data processing layer acts as a subscriber. This pattern allows for easy communication between the different layers and enables the data processing layer to receive updated data in real-time. The MQTT broker also ensures that the data is transmitted securely and efficiently, which is essential in ensuring that the system operates correctly. Additionally, the network layer also

includes the WebSocket server, which acts as a listener for changes made to the dashboard in the application layer. The WebSocket server communicates with the gateway, which then pushes data to the dashboard. This communication between the WebSocket server and the gateway is critical in ensuring that the system provides real-time updates to the user.

```
#provide port and ip address of the broker
port = 1883
mqttBroker = "10.26.30.33"

#'Suhu_Dalam' is the client name
client = mqtt.Client("Chicken Coop")

#provide uname and pwd
client.username_pw_set(username="umpfk", password="u4h#w1Tr12")

# Assign event callbacks
client.on_connect = on_connect
client.on_publish = on_publish
client.on_subscribe = on_subscribe
client.on_message = on_message

# create mqtt connection to broker
client.connect(mqttBroker,port)
```

Figure 3.15 Establish Connection to MQTT broker

```
rc = client.loop()
tuple =client.publish("coop/weight_data",sonic, qos=2, retain=False)
# print the rc code. 0 means ok. 5 means not.
print("rc: " + str(rc) + " publish " + str(sonic) + " to topic usonic")
print ("tuple" + str(tuple))

tuple =client.publish("coop/water_data",watah, qos=2, retain=False)
# print the rc code. 0 means ok. 5 means not.
print("rc: " + str(rc) + " publish " + str(watah) + " to topic water")
print ("tuple" + str(tuple))
```

Figure 3.16 Publish Data to Topic

The data processing layer of the Smart Coop Monitoring System is a critical component of the system as it handles all the incoming data from the sensing layer and processes it to be displayed on the application layer. The web server is responsible for storing the data in the database, which is located on the web server. It also receives the data from the MQTT broker, which it uses to update the database. In addition, the web server communicates with the AI server to run predictions based on the incoming data. The WebSocket server listens to the web server and receives updates from the dashboard on the application layer.

```

import paho.mqtt.client as mqtt

MQTT_Broker = "10.26.30.33"
MQTT_Port = 1883
Keep_Alive_Interval = 45
MQTT_Topic = "coop/#"

```

Figure 3.17 Establish Connection to Subscribe

```

def sensor_Data_Handler(Topic, jsonData):
    if Topic == "coop/temperature_data":
        goToDBtemperature(jsonData)
    if Topic == "coop/weight_data":
        goToDBweight(jsonData)
    if Topic == "coop/water_data":
        goToDBwaterlevel(jsonData)
    if Topic == "coop/light_data":
        goToDBlight(jsonData)
    if Topic == "coop/weight_predict":
        goToDBlweightpred(jsonData)
    if Topic == "coop/temperature_predict":
        goToDBltemppred(jsonData)
    if Topic == "coop/chicken_weight":
        goToDBlchickenpred(jsonData)

```

Figure 3.18 Subscribe to Topic

```

import sys
import time
import pysher

PUSHER_APP_ID='1'
PUSHER_APP_KEY='umpfkpusher'
PUSHER_APP_SECRET='u%M15z2h%3A'
PUSHER_APP_CLUSTER='mt1'
PUSHER_APP_HOST= '103.53.35.134'
PUSHER_APP_PORT = 6001

```

Figure 3.19 Establish Connection with Pusher

```

PUSHER_APP_ID='1'
PUSHER_APP_KEY='umpfkipusher'
PUSHER_APP_SECRET='u%M15z2h%3A'
PUSHER_APP_CLUSTER='mt1'
PUSHER_APP_HOST= '10.26.30.32'
PUSHER_APP_PORT = 6001
pusher_client = pusher.Pusher(host = PUSHER_APP_HOST,port = PUSHER_APP_PORT,app_id=PUSHER_APP_ID,
                               key=PUSHER_APP_KEY, secret=PUSHER_APP_SECRET,
                               ssl = False, cluster=PUSHER_APP_CLUSTER)

```

Figure 3.20 Establish Connection with Pusher Client Side

```

pusher_client.trigger('controlstats', u'App\\Events\\ChickenCoopEvent',
                    [{u'heaterstat': GPIO.input(7), u'coolerstat': GPIO.input(8), u'motorstat': GPIO.input(10)}])

```

Figure 3.21 Pusher Client Trigger

The AI server is responsible for making predictions based on the incoming data. It runs algorithms that process the data and provide predictions for future events. The predictions are then sent back to the web server for storage in the database and display on the dashboard. The data processing layer acts as a mediator between the sensing layer and the application layer. It receives data from the sensing layer, processes it, and then sends it to the application layer for display. This layer is crucial for ensuring that the system operates effectively and efficiently, providing accurate and relevant information to the user.

The application layer in the system architecture is responsible for presenting the data collected from the sensing layer to the end-user in a visual and intuitive format. It allows the user to interact with the system, make changes to the parameters, and receive notifications and alerts. The application layer consists of a dashboard that is connected to the WebSocket server and web server in the data processing layer. The dashboard displays information such as the current temperature, water level, light intensity, and food weight readings. It also provides a graphical representation of the data, making it easier to understand the trends and patterns. The user can control the system through the dashboard by turning on or off the heating lamp, lights, and fan. The user can also receive notifications and alerts if the temperature or water level falls below a certain threshold, for example.

3.4.4 Circuit Diagram

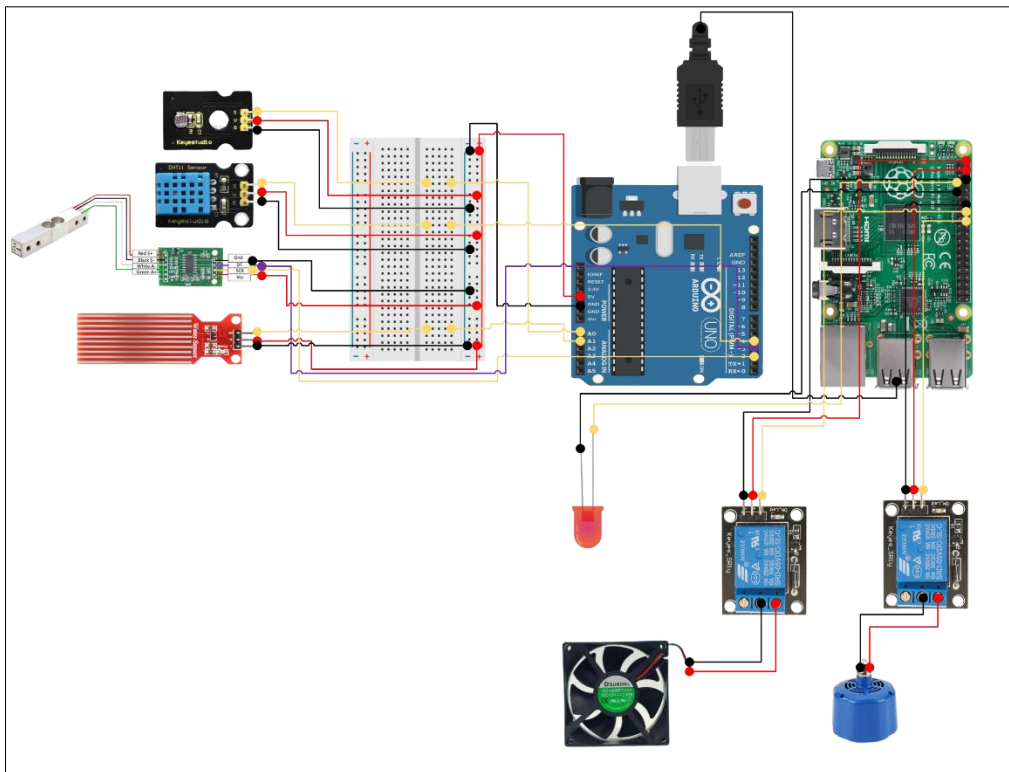


Figure 3.22 Circuit Diagram of SCMS

The circuit diagram is an illustration of the electrical connections between all the components in the system. It is a visual representation of the wiring, connections, and other electrical details in the system. The circuit diagram provides important information about the functioning of the system and helps in understanding the electrical flow between the different components.

3.4.5 Dialogue Diagram

This section of the report is about the dialogue diagram for the system. The diagram showcases the different pages that the user will encounter while using the system. The dialogue diagram consists of three main pages, which are the welcome page, dashboard, analytics, and user profile.

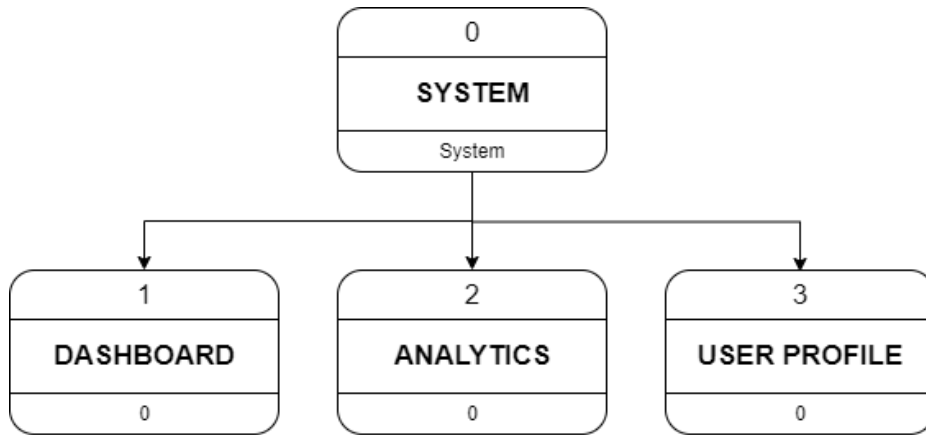


Figure 3.23 Dialogue Diagram of SCMS

Page Notation	Activity	Description
0	WELCOME	<ul style="list-style-type: none"> A welcome page to greet the users
1	DASHBOARD	<ul style="list-style-type: none"> IoT Dashboard of SCMS Displays real-time information about the coop. Provides the user with controls to adjust the heating lamp and fan.
2	ANALYTICS	<ul style="list-style-type: none"> Displays predicted weight of chicken.
3	USER PROFILE	<ul style="list-style-type: none"> Let's users to change their profile details.

Table 3.3 Description of Dialogue Diagram

3.4.6 Database Design

This section describes the database design for the system. The database is designed to store the data collected from the sensors and to keep track of the user's interactions with the system. The data collected from the sensors includes the readings from the DHT11, 10kg load cell, water level sensor, and photocell sensor. Furthermore, the weight of chicken and alert log is also recorded.

DATA FIELD	id	date_time	value
TYPE	int	varchar (50)	varchar (50)
DESCRIPTION	Row ID	Data and time of data logged	Environmental temperature reading

Table 3.4 Temperature Table Scheme

DATA FIELD	id	date_time	value
TYPE	int	varchar (50)	varchar (50)
DESCRIPTION	Row ID	Data and time of data logged	Environmental light intensity reading

Table 3.5 Light Intensity Table Scheme

DATA FIELD	Id	date_time	value
TYPE	Int	varchar (50)	varchar (50)
DESCRIPTION	Row ID	Data and time of data logged	Water level reading

Table 3.6 Water Level Table Scheme

DATA FIELD	Id	date_time	value
TYPE	Int	varchar (50)	varchar (50)
DESCRIPTION	Row ID	Data and time of data logged	Food weight reading

Table 3.7 Food Weight Table Scheme

DATA FIELD	Id	date_time	value
TYPE	Int	varchar (50)	varchar (50)
DESCRIPTION	Row ID	Data and time of data logged	Chicken weight reading

Table 3.8 Chicken Weight Table Scheme

DATA FIELD	id	date	deviceID
TYPE	int	varchar (50)	varchar (50)
DESCRIPTION	Row ID	Data and time of data logged	Alert event log

Table 3.4 Alert Table Scheme

3.4.7 Proposed AI Technique

The objective of this section is to provide a comprehensive explanation of the proposed AI technique for the Smart Coop Monitoring System. The AI technique plays a crucial role in the system as it helps in predicting the weight of the broilers in real-time. The weight prediction is an essential factor in determining the health and productivity of the poultry. AI techniques have been widely used in various industries to improve efficiency, accuracy, and productivity. In the agriculture industry, AI is used to improve the growth and health of crops and animals. The poultry industry is no exception, and AI is used to monitor the environment and growth of broilers in real-time.

The proposed AI technique for the Smart Coop Monitoring System is a Nonlinear Autoregressive Network with Exogenous Inputs (NARX). NARX is a type of Artificial Neural Network (ANN) that is used to model the dynamic relationship between inputs and outputs. It is used to predict future values based on past values and current inputs [11]. NARX consists of two parts: the Autoregressive (AR) part and the Exogenous (X) part. The AR part models the relationship between the past values of the output and the current state of the system. The X part models the relationship between the current inputs and the future values of the output.

In the Smart Coop Monitoring System, the NARX will be trained with historical data collected from the sensing layer. The inputs to the NARX will be the temperature, light intensity, and weight of feed. The output will be the weight of the broilers. The NARX will use the inputs and past values of the output to predict the future weight of the broilers [12].

u1	u2	u3	y1
13.25	22	931	40
13.25	22	964	45
13.25	22	965	50
13.25	23	921	55
13.75	23	982	60
13.75	23	911	64
13.75	22	951	69
13.75	22	970	73
14.25	22	931	77
14.25	22	923	85
14.25	22	924	93
14.25	21	976	101
14.5	21	910	109
14.5	23	981	123
14.5	23	985	137
14.5	23	980	151
13.25	22	967	165
13.25	22	966	175
13.25	22	938	185
13.25	22	923	195
14	22	940	205
14	21	906	218
14	21	930	231
14	22	944	244
13.75	22	945	257
13.75	22	931	269
13.75	22	967	281
13.75	23	999	293
22	23	967	305
22	23	972	318

Figure 3.24 Example data of NARX inputs and output

The proposed AI technique of one-step ahead prediction using nonlinear autoregressive network with exogenous inputs has the potential to provide accurate and reliable predictions of the future values of the monitored parameters in the Smart Coop Monitoring System. The implementation of this model will improve the effectiveness of the system in monitoring the environment and growth of the broilers, and ultimately increase the profitability of the poultry farm.

3.4.8 UI/UX Design

In the design of the user interface (UI) for the Smart Coop Monitoring System, the proposed approach is to follow the heuristics established by experts in the field of human-computer interaction. The goal of the design is to create an intuitive and user-friendly interface that will allow users to interact with the system easily and effectively.

This section will provide a detailed overview of the proposed UI/UX design, including the following key elements:

- **Dashboard:** The dashboard will be the primary interface for the system and will provide users with an overview of all relevant information, including real-time sensor data and system controls.
- **Analytics Page:** The analytics page will allow users to view a graph that shows the prediction of chicken weight, providing users with a visual representation of the system's performance.
- **User Profile:** The user profile will allow users to view and manage their personal information and settings, including their preferences and notification settings.

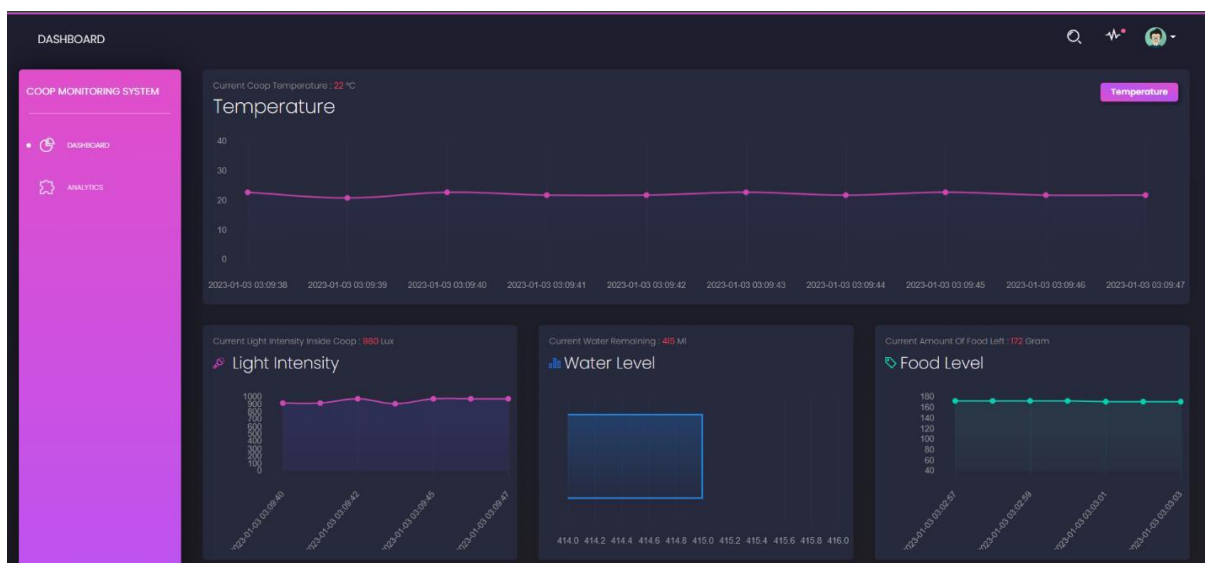


Figure 3.25 Dashboard UI

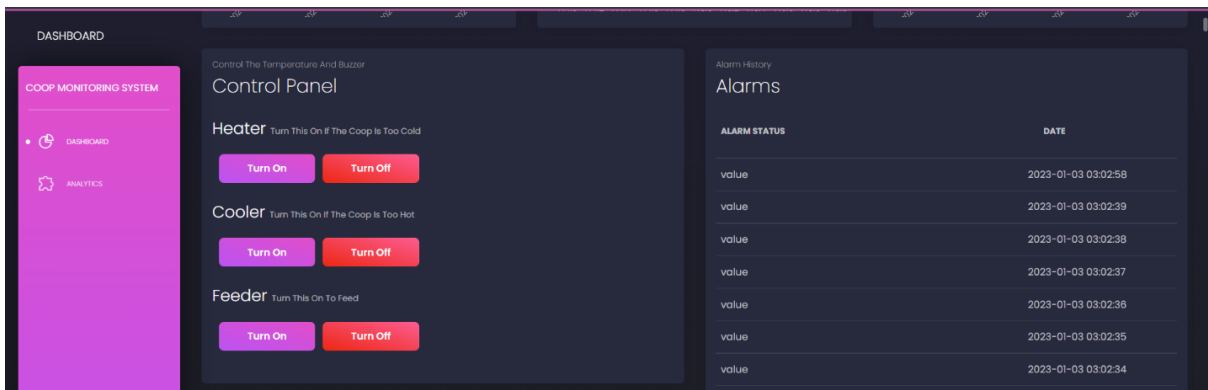


Figure 3.26 Control panel and Alarm logs

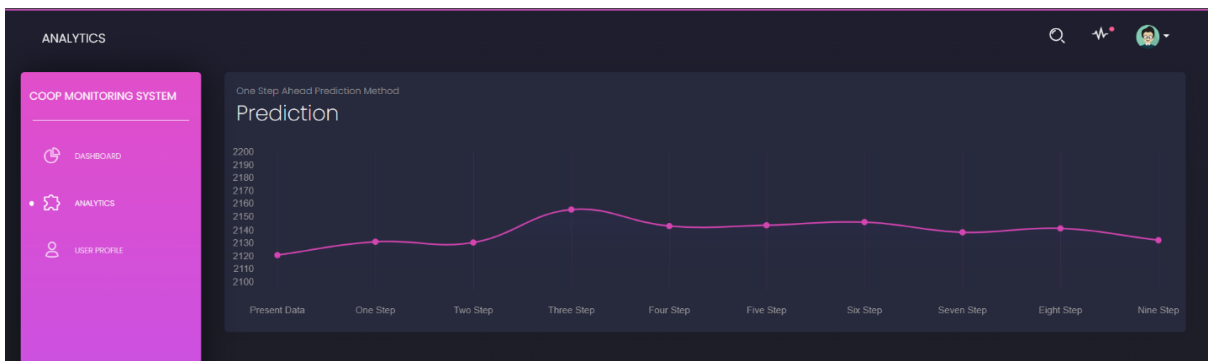


Figure 3.27 Analytics UI

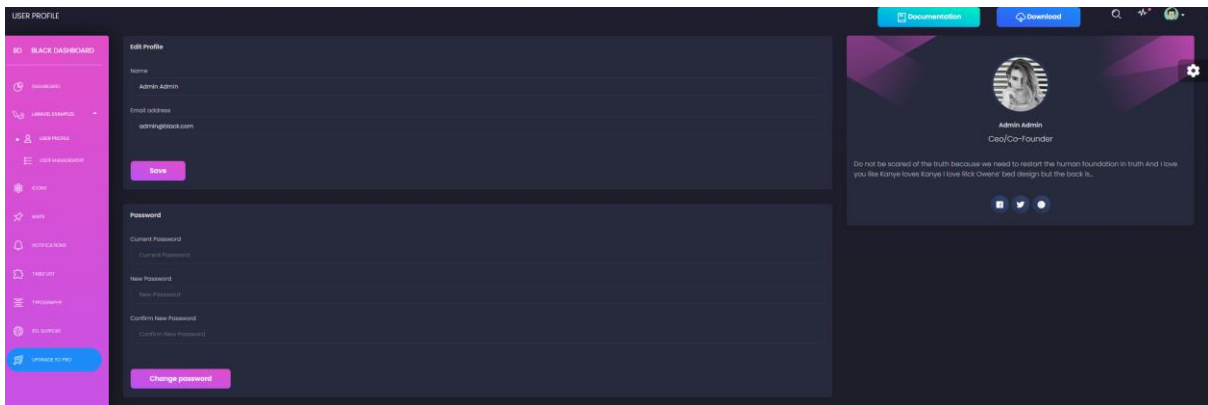


Figure 3.28 User Profile UI

3.5 Gantt Chart

This section discusses the Gantt chart for the Rapid Application Development (RAD) model applied in the development of the Smart Coop Monitoring System. The Gantt chart helps to keep track of the progress of the project, ensuring that tasks are completed within the set time frame, and resources are used effectively.

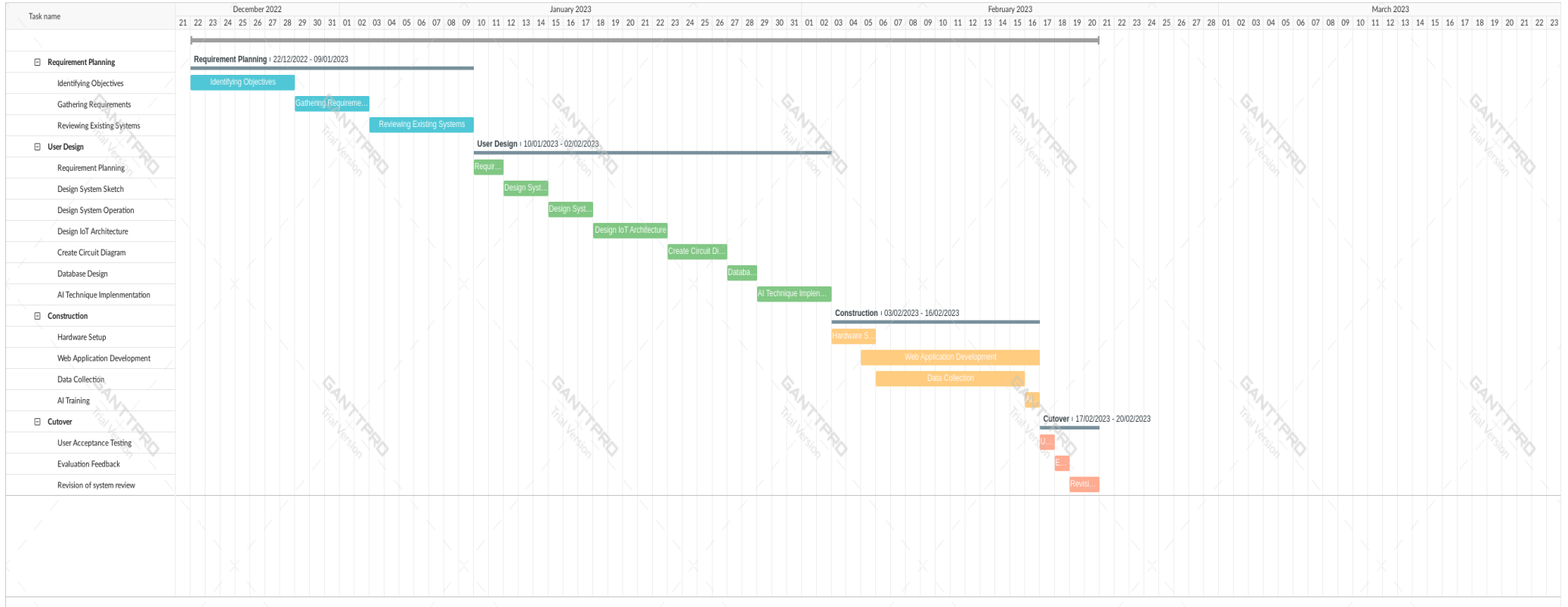


Figure 3.29 Gantt Chart of SCMS

CHAPTER 4

IMPLEMENTATION, RESULT AND DISCUSSION

4.1 Introduction

This chapter focuses on the practical aspects of the project, including the development of the hardware and software components, the testing process, and the analysis of the collected data. The results and discussion section provides an evaluation of the system's effectiveness in achieving the stated objectives, including the accuracy of AI predictions, and offers insights into the future development of the system.

4.2 Implementation Process and Results

In this section, we will discuss in detail the implementation process of the smart coop monitoring system.

4.2.1 Data Collection

For this project, the data collection process involved gathering information on various environmental factors inside the poultry coop such as temperature, water level, feed weight, amount of feed eaten, weight of chicks, and light intensity. In addition, the weight of the broilers was also recorded at various time intervals to monitor their growth.

To collect the data, several sensors were installed inside the coop, including temperature sensor, water level sensor, two load cells, and photocell sensor. The weight of the broilers was measured using a digital scale.

The data was then stored in a database and used as inputs for the machine learning algorithm. The collected data was cleaned, preprocessed, and normalized to ensure the accuracy of the algorithm's predictions.

```

if time.time() - last_publish_time >= 5:
    rc = client.loop()
    tuple =client.publish("coop/temp_data",temperature, qos=2, retain=False)
    # print the rc code. 0 means ok. 5 means not.
    print("rc: " + str(rc) + " publish " + str(temperature) + " to topic
temperature")
    print ("tuple" + str(tuple))

    tuple =client.publish("coop/eweight_data",eatenweight, qos=2,
retain=False)
    # print the rc code. 0 means ok. 5 means not.
    print("rc: " + str(rc) + " publish " + str(eatenweight) + " to topic
eatenweight")
    print ("tuple" + str(tuple))

    tuple =client.publish("coop/light_data",light, qos=2, retain=False)
    # print the rc code. 0 means ok. 5 means not.
    print("rc: " + str(rc) + " publish " + str(light) + " to topic light")
    print ("tuple" + str(tuple))

    tuple =client.publish("coop/water_data",water, qos=2, retain=False)
    # print the rc code. 0 means ok. 5 means not.
    print("rc: " + str(rc) + " publish " + str(water) + " to topic water")
    print ("tuple" + str(tuple))

    tuple =client.publish("coop/chick_weight",chickweight, qos=2,
retain=False)
    # print the rc code. 0 means ok. 5 means not.
    print("rc: " + str(rc) + " publish " + str(chickweight) + " to topic chick
weight")
    print ("tuple" + str(tuple))

    tuple =client.publish("coop/fweight_data",feedweight, qos=2, retain=False)
    # print the rc code. 0 means ok. 5 means not.
    print("rc: " + str(rc) + " publish " + str(feedweight) + " to topic
feedweight")
    print ("tuple" + str(tuple))

    # Update last_publish_time
    last_publish_time = time.time()

```

Figure 4.1 Publish Collected Data Snippet

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> chicken_weight	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> light_data	★ Browse Structure Search Insert Empty Drop	489	InnoDB	utf8mb4_general_ci	48.0 KiB	-
<input type="checkbox"/> temperature_data	★ Browse Structure Search Insert Empty Drop	489	InnoDB	utf8mb4_general_ci	48.0 KiB	-
<input type="checkbox"/> water_data	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> weight_data	★ Browse Structure Search Insert Empty Drop	489	InnoDB	utf8mb4_general_ci	48.0 KiB	-
5 tables	Sum	1,467	InnoDB	utf8mb4_general_ci	176.0 KiB	0 B

Figure 4.2 Database for Data Collection

4.2.2 Database Implementation

For this project, MySQL is chosen, a popular open-source relational database management system. The first step in the implementation process was to design the database schema, which involves identifying the tables needed for the project, determining the relationships between them, and specifying the attributes and data types of each table.

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> alarm_stats	★ Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> chick_weight	★ Browse Structure Search Insert Empty Drop	3,807	InnoDB	utf8mb4_general_ci	208.0 KiB	-
<input type="checkbox"/> eweight_data	★ Browse Structure Search Insert Empty Drop	3,808	InnoDB	utf8mb4_general_ci	208.0 KiB	-
<input type="checkbox"/> failed_jobs	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
<input type="checkbox"/> fweight_data	★ Browse Structure Search Insert Empty Drop	3,807	InnoDB	utf8mb4_general_ci	224.0 KiB	-
<input type="checkbox"/> light_data	★ Browse Structure Search Insert Empty Drop	3,808	InnoDB	utf8mb4_general_ci	208.0 KiB	-
<input type="checkbox"/> migrations	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> password_resets	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
<input type="checkbox"/> personal_access_tokens	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
<input type="checkbox"/> temperature_data	★ Browse Structure Search Insert Empty Drop	3,807	InnoDB	utf8mb4_general_ci	208.0 KiB	-
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
<input type="checkbox"/> water_data	★ Browse Structure Search Insert Empty Drop	3,808	InnoDB	utf8mb4_general_ci	208.0 KiB	-
<input type="checkbox"/> websockets_statistics_entries	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
13 tables	Sum	22,860	InnoDB	utf8mb4_general_ci	1.4 MiB	0 B

Figure 4.3 SCMS Database

The data collected from the microcontroller is sent to the database through MQTT.


```

# Function to save temperature to DB Table
def goToDBtemperature(jsonData):

    Value = float(json.loads(jsonData))
    # print(Value)

    sql = "INSERT INTO temperature_data (id, value) VALUES (%s,%s)"
    val = ('',Value)

    mycursor.execute(sql, val)
    mydb.commit()
    current_time = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print("Temperature Data Store at " + current_time)

```

Figure 4.4 Insert to DB Snippet

The table structure for all tables is:


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	value	varchar(50)	utf8mb4_general_ci		No	None		
3	date_time	varchar(50)	utf8mb4_general_ci		No	current_timestamp()		

Figure 4.5 Database Table Structure

4.2.3 Hardware Development

In the hardware development phase of the project, various components were used to develop the smart coop monitoring system. These components included a DHT11 sensor for temperature monitoring, two load cells for weight measurement, a water level sensor, an exhaust fan, a heating lamp, a light bulb, an Arduino Uno microcontroller, and a Raspberry Pi 3B+.

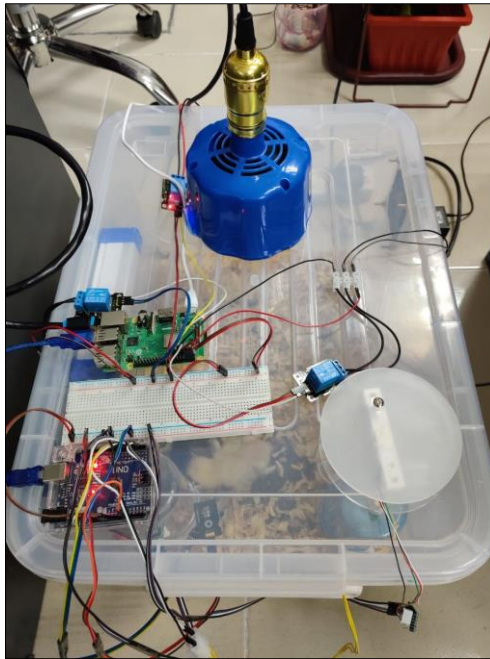


Figure 4.6 Hardware Setup

The Arduino is responsible for collecting data. The Raspberry Pi will read the data from Arduino as shown in Figure 4.6. The Pi will then publish the data to MQTT topic and data will be subscribed by `mqttodb.py` file. The data subscribed will be sent to MySQL database.

```
arduinouno = serial.Serial('/dev/ttyUSB0', 9600, timeout=.1)
print("Established serial connection to arduinouno")
```

Figure 4.7 Read from Serial

```
def sensor_data_handler(topic, jsonData):
    if topic == "coop/temp_data":
        go_to_db_temperature(jsonData)
    if topic == "coop/eweight_data":
        go_to_db_eweight(jsonData)
    if topic == "coop/water_data":
        go_to_db_water_level(jsonData)
    if topic == "coop/light_data":
        go_to_db_light(jsonData)
    if topic == "coop/chick_weight":
        go_to_db_lchick_weight(jsonData)
    if topic == "coop/fweight_data":
        go_to_db_lfweight(jsonData)
```

Figure 4.8 Subscribe MQTT Topic

4.2.4 Web Application Development

In this section, we will discuss in detail the web application development process that was undertaken to build the front-end interface for the smart coop monitoring system. The aim of the web application is to provide an easy-to-use interface that can be accessed from anywhere, and that displays real-time information on the coop environment and the broiler growth progress.

For the development of the web application, we chose the Laravel PHP framework. Additionally, we also utilized the WebSocket technology to enable real-time data transmission between the web application and the backend server.

```
Echo.channel('alarm-history').listen('AlarmStatus', (e) => {  
    var table = document.getElementById("myTable");  
    var row = '<tr><td>' + e['deviceID'] + '</td><td>' + new Date().toLocale  
    table.innerHTML = row + table.innerHTML;  
    var row = table.insertRow(0);  
    var cell1 = row.insertCell(0);  
    var cell2 = row.insertCell(1);  
    cell1.innerHTML = e['deviceID'];  
    cell2.innerHTML = new Date().toLocaleString('en-CA', {hour12: false,});  
    // document.getElementById("myAlert").innerHTML = 'Alarm Trigger!!';  
    // document.getElementById("myAlert" ).style.color="red";  
});
```

Figure 4.9 WebSocket Connection

4.2.5 Web Server Hosting

After the completion of the web application, the next step is to host it online so that it can be accessed from anywhere. For this purpose, a web hosting service is required, which allows the website to be published on the internet. In this project, the web application was hosted on a shared web hosting server. The server was provided by Universiti Malaysia Pahang (UMP).

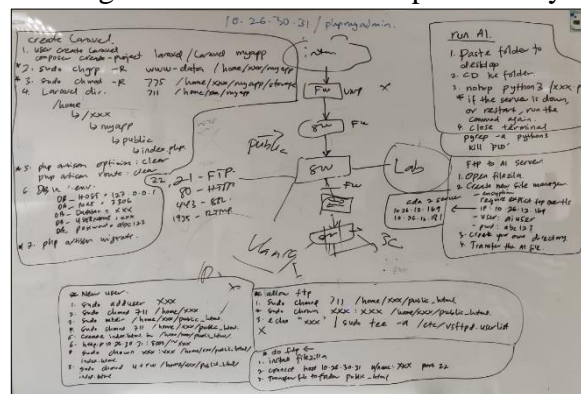


Figure 4.10 Web Hosting Setup

To upload the web application to the hosting server, the files were transferred using FileZilla. The database was also imported using the phpMyAdmin tool.

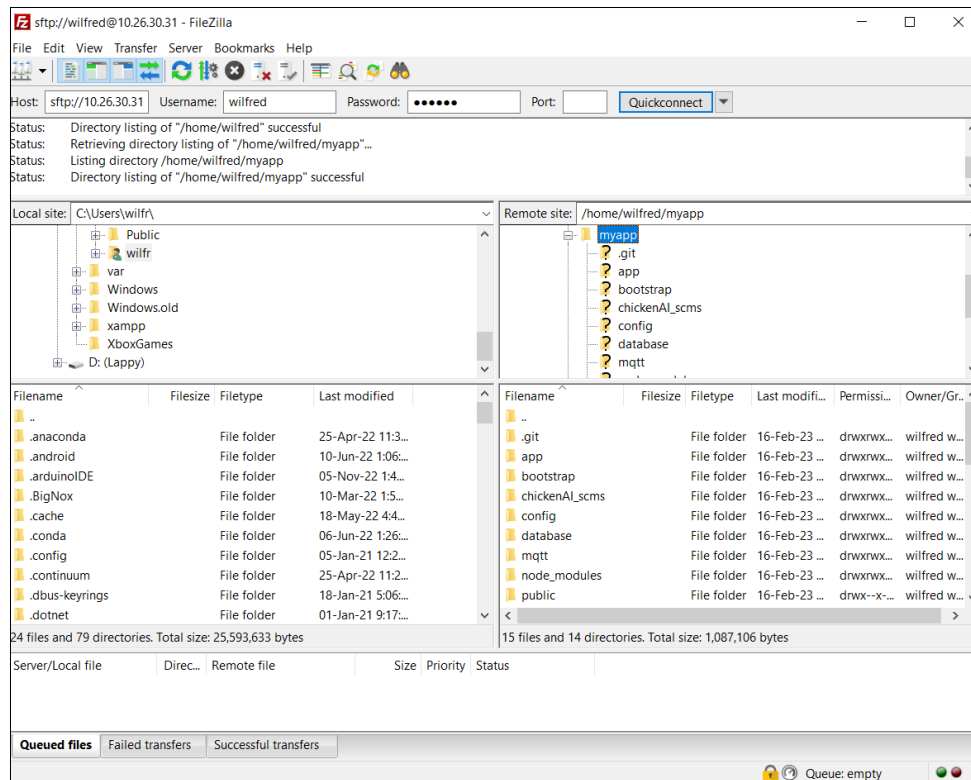


Figure 4.11 FileZilla Interface

Once the web application was uploaded and the database was connected, the website was live and could be accessed from anywhere with an internet connection. This allowed the poultry farmers to remotely monitor the environment inside their coops and receive alerts in case of any abnormal conditions.

4.2.6 MQTT Broker

In the context of this project, the microcontroller sends data collected from the sensors to the MQTT server, which then forwards the data to the database. MQTT allows for low bandwidth usage, making it ideal for IoT applications where network resources are limited. Furthermore, it has low overhead and supports various Quality of Service (QoS) levels, ensuring that messages are delivered reliably and efficiently. The chosen MQTT server for this project is Mosquitto.

```

1676559933: Received PUBREL from Chicken Coop (Mid: 6932)
1676559933: Sending PUBCOMP to Chicken Coop (m6932)
1676559933: Received PUBREL from Chicken Coop (Mid: 6933)
1676559933: Sending PUBCOMP to Chicken Coop (m6933)
1676559933: Received PUBREL from Chicken Coop (Mid: 6934)
1676559933: Sending PUBCOMP to Chicken Coop (m6934)
1676559933: Received PUBREL from Chicken Coop (Mid: 6935)
1676559933: Sending PUBCOMP to Chicken Coop (m6935)
1676559933: Received PUBREL from Chicken Coop (Mid: 6936)
1676559933: Sending PUBCOMP to Chicken Coop (m6936)
1676559933: Received PUBREL from Chicken Coop (Mid: 6937)
1676559933: Sending PUBCOMP to Chicken Coop (m6937)
1676559933: Received PUBLISH from Chicken Coop (d0, q2, r0, m6938, 'coop/temp_data', ... (4 bytes))
1676559933: Sending PUBREC to Chicken Coop (m6938, rc0)
1676559933: Received PUBLISH from Chicken Coop (d0, q2, r0, m6939, 'coop/eweight_data', ... (6 bytes))
1676559933: Sending PUBREC to Chicken Coop (m6939, rc0)
1676559933: Received PUBLISH from Chicken Coop (d0, q2, r0, m6940, 'coop/light_data', ... (3 bytes))
1676559933: Sending PUBREC to Chicken Coop (m6940, rc0)
1676559933: Received PUBLISH from Chicken Coop (d0, q2, r0, m6941, 'coop/water_data', ... (1 bytes))
1676559933: Sending PUBREC to Chicken Coop (m6941, rc0)
1676559933: Received PUBLISH from Chicken Coop (d0, q2, r0, m6942, 'coop/chick_weight', ... (4 bytes))
1676559933: Sending PUBREC to Chicken Coop (m6942, rc0)
1676559933: Received PUBLISH from Chicken Coop (d0, q2, r0, m6943, 'coop/fwight_data', ... (6 bytes))
1676559933: Sending PUBREC to Chicken Coop (m6943, rc0)
1676559933: Received PUBLISH from Syahmi (d0, q0, r0, m0, 'CA19146/TEMPERATUREDATA', ... (5 bytes))
1676559933: Received PUBLISH from Syahmi (d0, q0, r0, m0, 'CA19146/AIR_QUALITYDATA', ... (2 bytes))
1676559933: Received PUBLISH from syahmiad11 (d0, q0, r0, m0, 'CA19146-2/PULSE_RATEDATA', ... (3 bytes))
1676559934: Received PINGREQ from auto-E0C5C700-BB32-1631-87EB-68A1D2E032D3
1676559934: Sending PINGRESP to auto-E0C5C700-BB32-1631-87EB-68A1D2E032D3
1676559934: Received PUBLISH from Syahmi (d0, q0, r0, m0, 'CA19146/TEMPERATUREDATA', ... (5 bytes))
1676559934: Received PUBLISH from Syahmi (d0, q0, r0, m0, 'CA19146/AIR_QUALITYDATA', ... (2 bytes))
1676559934: Received PINGREQ from auto-ACEB9914-5019-0699-4A22-B251C4E1A1FE
1676559934: Sending PINGRESP to auto-ACEB9914-5019-0699-4A22-B251C4E1A1FE
1676559934: Received PUBLISH from Syahmi (d0, q0, r0, m0, 'CA19146/TEMPERATUREDATA', ... (5 bytes))
1676559934: Received PUBLISH from Syahmi (d0, q0, r0, m0, 'CA19146/AIR_QUALITYDATA', ... (2 bytes))
1676559934: Received PUBLISH from syahmiad11 (d0, q0, r0, m0, 'CA19146-2/PULSE_RATEDATA', ... (3 bytes))
1676559935: Received PUBLISH from Raspi (d0, q1, r0, m4782, 'SAS/temperature', ... (5 bytes))
1676559935: Sending PUBREL to auto-D063788-8533-1053-9088-BE73A95980 (d0, q0, r0, m0, 'SAS/temperature', ... (5 bytes))

```

Figure 4.12 MQTT Broker

4.2.7 AI Server

An AI server was set up to receive the data collected by the sensors through the MQTT broker and perform predictive analysis using a Nonlinear Autoregressive network with eXogenous inputs (NARX) model.

To ensure that the system operates efficiently and provides accurate predictions, the AI server was deployed on a separate computer system from the database and the web application. This ensures that the AI server can handle the computational requirements of the predictive analysis without affecting the performance of the web application.

```

aluser@fkiotumpal2-PowerEdge-R740: ~/ftp/files/cb20028_ai
warnings.warn(
1/1 [=====] - 0s 42ns/step
1/1 [=====] - 0s 17ns/step
1/1 [=====] - 0s 107ns/step
1/1 [=====] - 0s 17ns/step
1/1 [=====] - 0s 46ns/step
1/1 [=====] - 0s 10ns/step
1/1 [=====] - 0s 43ns/step
1/1 [=====] - 0s 18ns/step
1/1 [=====] - 0s 47ns/step
1/1 [=====] - 0s 17ns/step
1/1 [=====] - 0s 45ns/step
1/1 [=====] - 0s 18ns/step
1/1 [=====] - 0s 44ns/step
1/1 [=====] - 0s 18ns/step
1/1 [=====] - 0s 38ns/step
1/1 [=====] - 0s 19ns/step
1/1 [=====] - 0s 49ns/step
1/1 [=====] - 0s 26ns/step
1/1 [=====] - 0s 45ns/step
1/1 [=====] - 0s 19ns/step
publish to socket
-1.72

```

Figure 4.13 AI Server

4.2.8 WebSocket Server

The WebSocket server was used to communicate with the web application and push real-time data to the client. The server listened to events sent from the client and broadcasted data to the appropriate channel. The server was implemented as a PHP script that ran continuously in the background and listened for incoming connections.

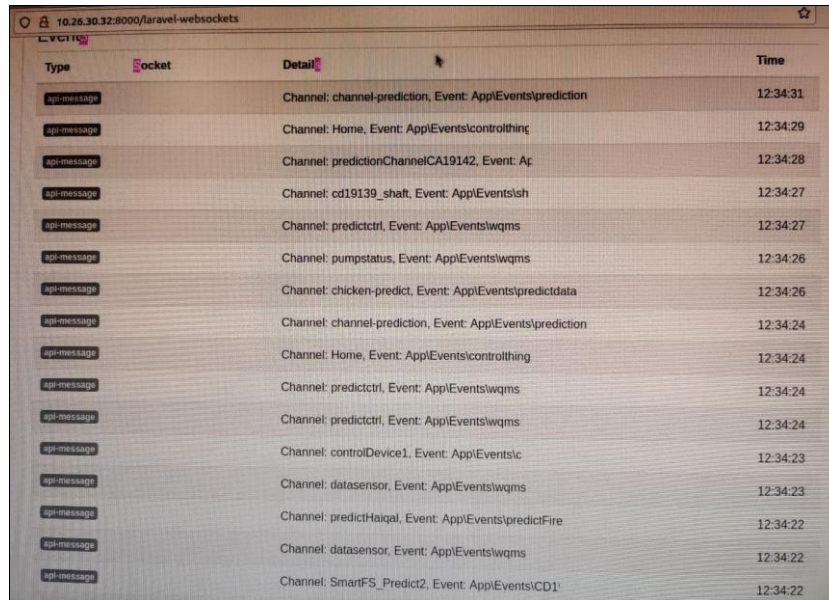


Figure 4.14 WebSocket Server

4.2.9 AI Implementation

NARX model has been implemented for one step ahead prediction of broiler weight. The model uses the historical data of the broiler weight along with the data such as temperature, amount of food eaten, and light intensity. The input data is preprocessed and fed to the NARX model which is then trained using backpropagation algorithm. A total of 489 data was collected for each sensor. The data is trained with these parameters: -

```
# parameters
tf.compat.v1.disable_eager_execution()
learningRate = 1e-3
weightDecay = 1e-3 / 100
maxEpochs = 10000
miniBatchSize = 512
lag = 3
# hidden layer sizes
h1 = 10
h2 = 12
h3 = 1
```

Figure 4.15 Training Parameters

The results produced after training the dataset is as follows: -

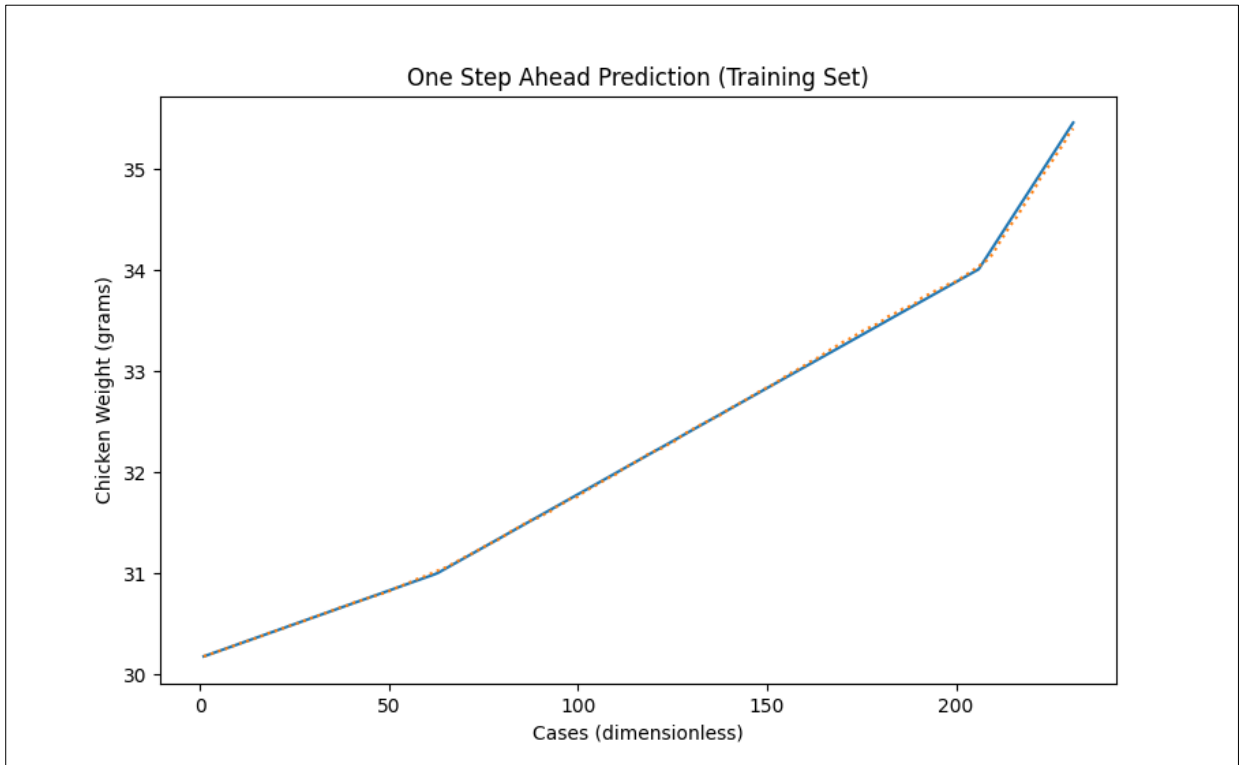


Figure 4.16 Training Set

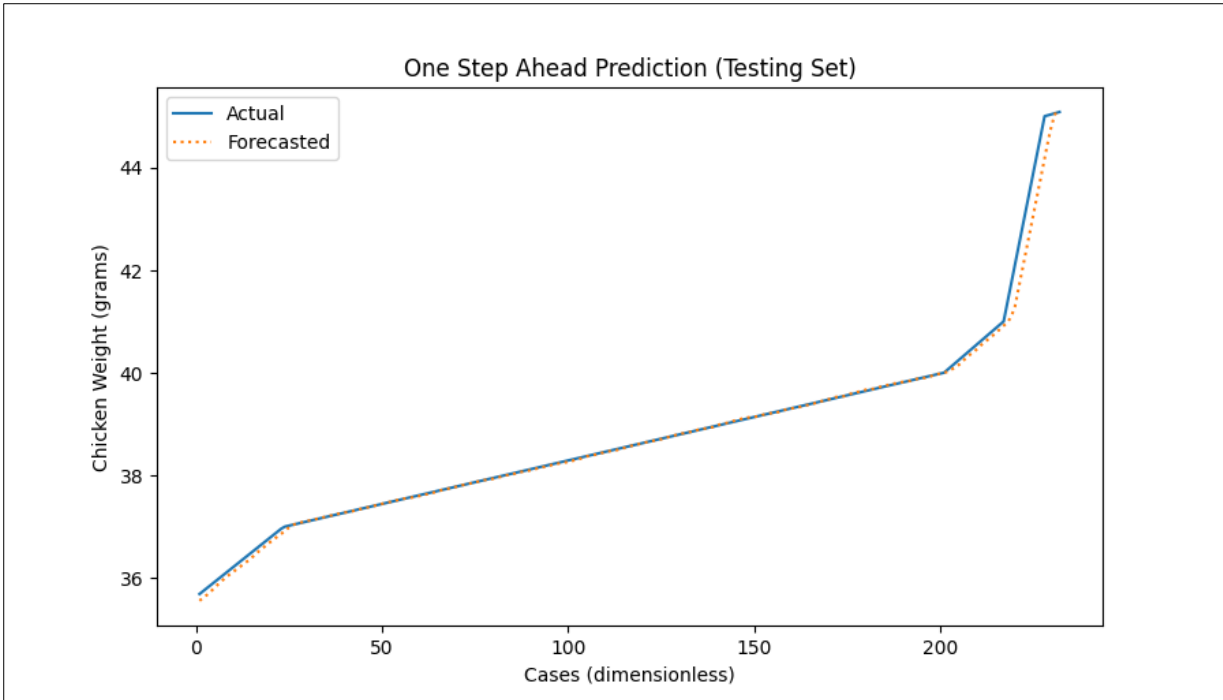


Figure 4.17 Comparison Set

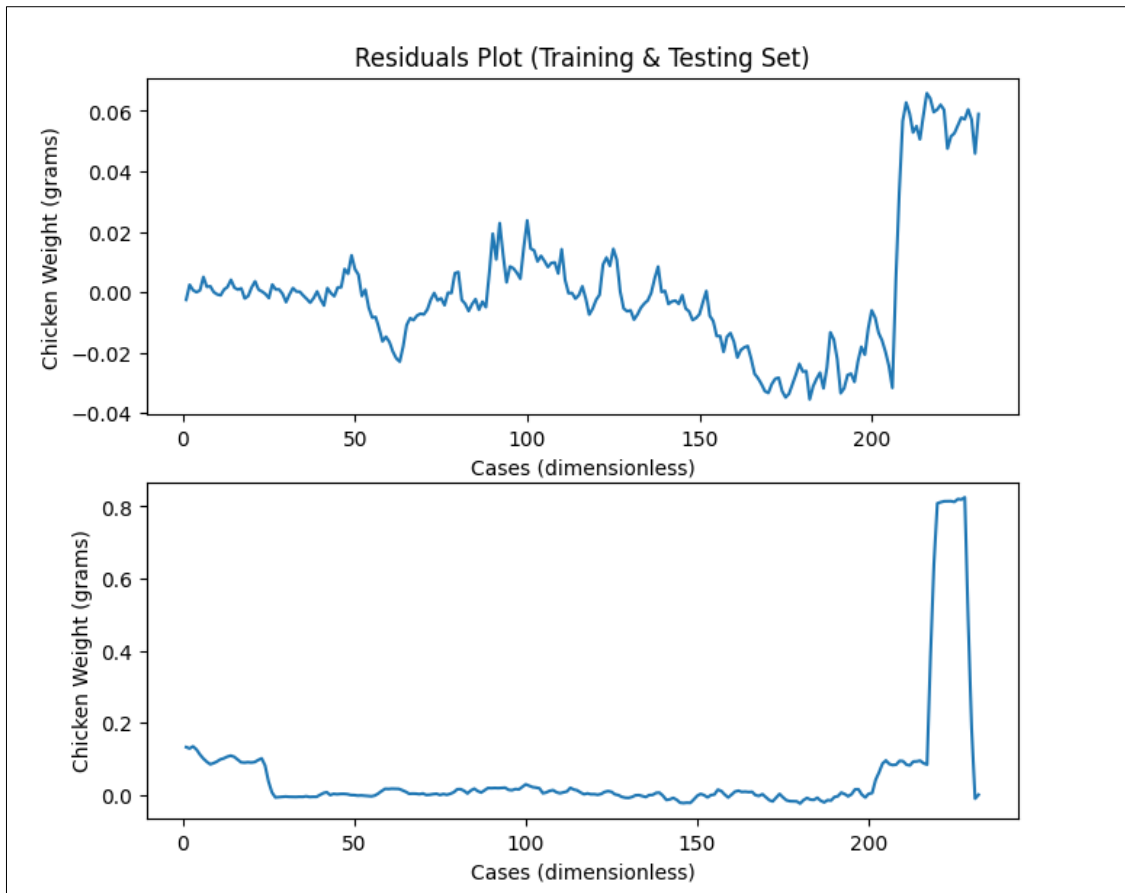


Figure 4.18 Residual Plot

4.2.10 Dashboard Implementation

The dashboard is an essential part of the smart coop monitoring system as it allows farmers to view and analyze the real-time data collected by the sensors in a user-friendly interface. In this section, we will discuss the implementation of the dashboard.

4.2.10.1 Charts

The dashboard was implemented using HTML, CSS, and JavaScript. It is designed to be responsive, allowing farmers to view the dashboard on different devices, such as desktop computers, tablets, and mobile phones. To display the real-time data collected from the sensors, the dashboard is integrated with the database using Laravel framework. The real-time data is collected from the sensors and sent to the database through MQTT. The dashboard retrieves the data from the database and displays it in the form of graphs and charts. To provide real-time data updates to the dashboard, the WebSocket protocol was used, allowing the server to push updates to the client whenever new data is available.

```
$.getJSON('/route/temperatureroute', function(blocksall){
  var datas = blocksall.blocks.map(Number);
  datas = datas.reverse().slice(-11, -1);
  console.log(datas)
  var datasx = blocksall.blocks2.map(String);
  datasx = datasx.reverse().slice(-11, -1);

  document.getElementById("temptxt").innerHTML = datas[datas.length-1];

  //Timecheck
  first = new Date(datasx[datasx.length-1]);
  second = new Date();
  diff = second.getTime() - first.getTime();

  if (diff > 15000){
    document.getElementById("temptxt").style.color = "#F6465B";
  }
  else{
    document.getElementById("temptxt").style.color = "#5ced73";
  }

  var ctx1 = document.getElementById("chartBig1").getContext('2d');
```

Figure 4.19 Temperature Chart Setup

The design of the dashboard was kept simple and intuitive, with easy-to-read graphs and charts displaying the key parameters, such as temperature, humidity, water level, and weight of broilers.

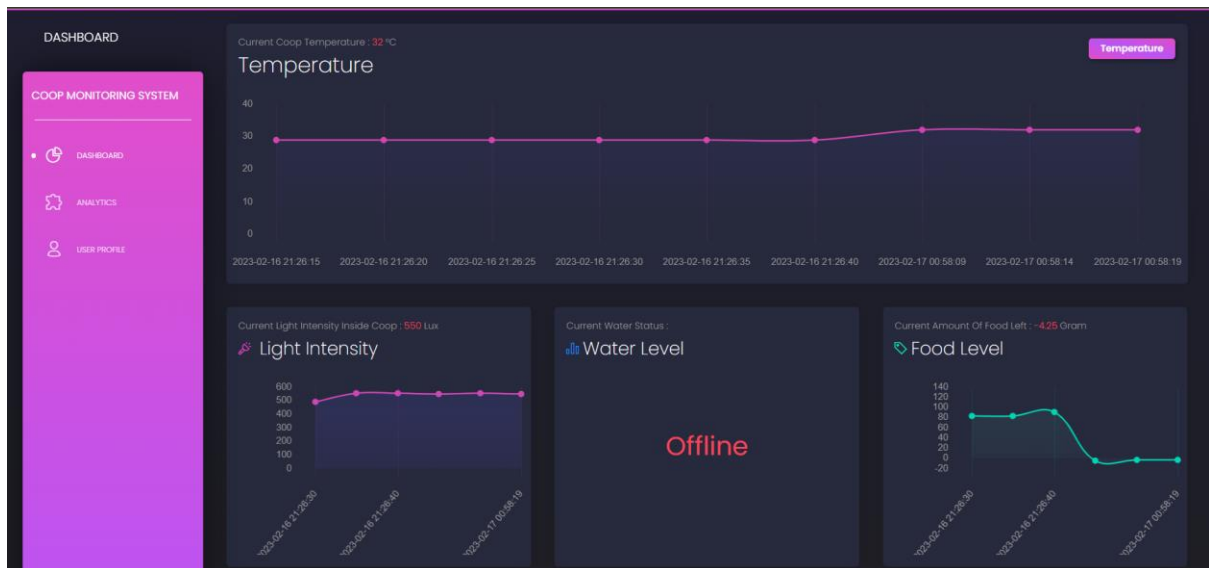


Figure 4.20 Temperature Chart Setup

The graphs are updated every 5 seconds by using the Ajax Call function. We also implemented a time check for every 15 seconds between each new data to visualize the connection of the system.

```
//Timecheck
first = new Date(datasx[datasx.length-1]);
second = new Date();
diff = second.getTime() - first.getTime();

if (diff > 15000){
  document.getElementById("temptxt").style.color = "#F6465B";
}
else{
  document.getElementById("temptxt").style.color = "#5ced73";
}
```

Figure 4.21 Connection Time Check

4.2.10.2 Alerts

The dashboard also includes an alert system that sends notifications to farmers if any of the parameters fall outside the normal range, allowing them to act immediately. Notifications will be sent to Discord app.

Alarm History

Alarms

ALARM STATUS	DATE
Chicken Food	2023-02-16 21:24:53
Water Level	2023-02-16 21:24:52
Chicken Food	2023-02-16 21:24:47
Water Level	2023-02-16 21:24:46
Chicken Food	2023-02-16 13:37:10
Chicken Food	2023-02-16 13:37:07

Figure 4.22 Alarm History

To connect to Discord, we just need to take the webhooks and implement it in code.

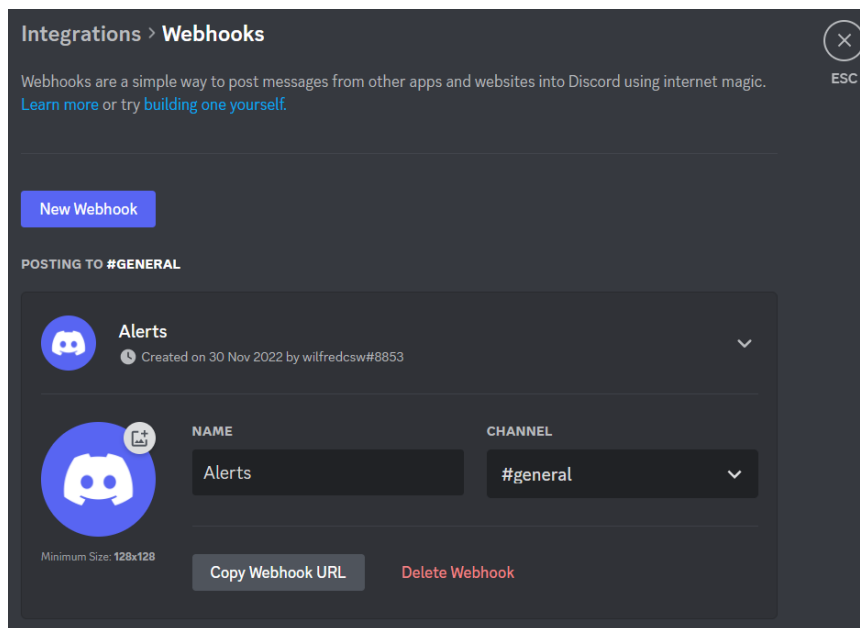


Figure 4.23 Discord Webhook

```
webhook_url = 'https://discord.com/api/webhooks/1047195620005384252/lmEMLl-jdfpOU2Hk51MDjQVInj1B6b58N1sY_qpurXr3WuBQKQ-8P_0ThUngPB1Y4giz'
pusher_client = pusher.Pusher(app_id='1', key='umpfkpusher', secret='u%M15z2h%3A', cluster='mt1', ssl=False, host='10.26.30.32', port=443)
```

Figure 4.24 Discord Webhook Implementation

If one of the sensors reaches their set threshold, it will notify the farmers for any abnormalities within the coop.

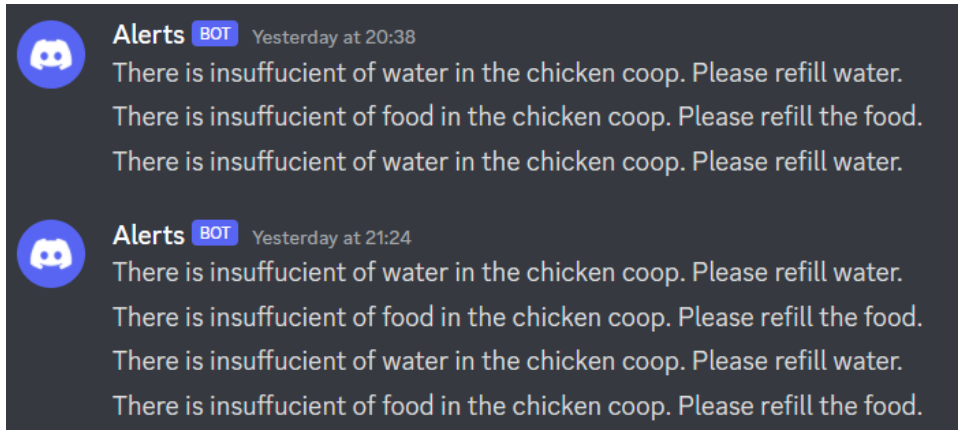


Figure 4.25 Discord Notification

4.2.10.3 Controls

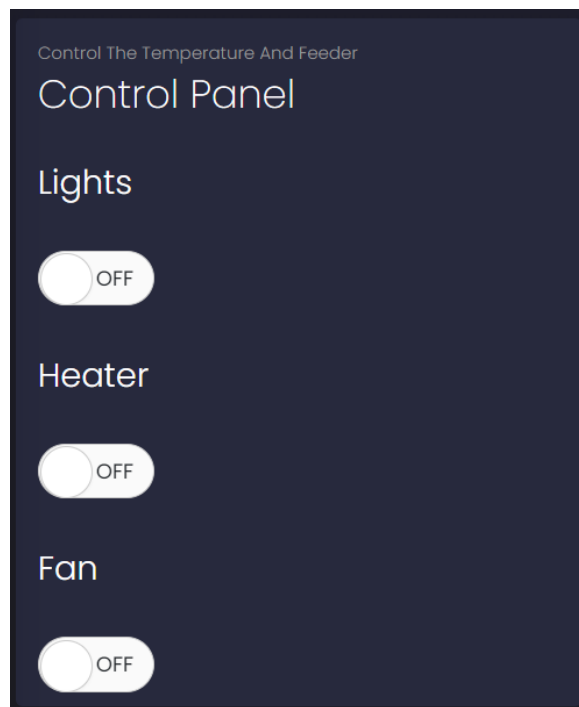


Figure 4.26 Control Hardware

The control panel is a web-based interface that allows the user to control the output devices such as lights, fan, and heater using MQTT protocol. The MQTT protocol is used to communicate between the control panel and the output devices.

```
pi@raspberrypi:~/Desktop/mqtt $ python mqtt.py
Connected to MQTT broker
lights on
lights on
lights off
lights off
lights on
lights on
lights off
lights off
Heater on
Heater on
Heater off
Heater off
Fan on
Fan on
Fan off
Fan off
```

Figure 4.27 Raspberry Pi Control Side

```
if msg.topic == "coop/trigger/lights":
    if string == "ON":
        GPIO.output(13, GPIO.HIGH)
        print("lights on")
    elif string == "OFF":
        GPIO.output(13, GPIO.LOW)
        print("lights off")

if msg.topic == "coop/trigger/heater":
    if string == "ON":
        GPIO.output(11, GPIO.HIGH)
        print("Heater on")
    elif string == "OFF":
        GPIO.output(11, GPIO.LOW)
        print("Heater off")
```

Figure 4.28 Raspberry Pi Control Snippet

4.2.10.4 Analytics



Figure 4.29 Prediction Interface

In order to improve the system's accuracy, machine learning algorithms were implemented to predict the conditions inside the coop. The predicted data was then displayed on an analytics page, providing insight into the system's performance and accuracy.

The predicted data is based on the historical data collected from the sensors in the coop. This data is fed into the NARX one-step-ahead neural network model for training, which then provides predictions for the current and future conditions inside the coop.

4.3 Discussion

The results of this project demonstrate that the proposed smart coop monitoring system is effective in improving the environment conditions of the coops and reducing the mortality rate of chicken during the early stages of growth. The system provides real-time monitoring and control of temperature, amount of food eaten, water level, light intensity, and weight of broilers, and alerts farmers to any abnormal conditions in the coops. The data collected from the system is stored in a database and analyzed using an NARX one-step-ahead model to provide accurate predictions of the environment conditions in the coops.

The system provides real-time monitoring and control of the coop environment, accurate predictions of the environment conditions, and a dashboard for real-time monitoring and control. However, further research is needed to address the limitations of the system and improve its overall effectiveness and efficiency.

4.4 Summary

Chapter 4 focuses on the implementation of the proposed system, including the data collection process, database implementation, hardware development, web application development, MQTT and WebSocket server implementation, AI implementation, dashboard implementation, control panel, and analytics. The chapter details the hardware and software components used in the system, as well as the specific steps taken to implement each component. The chapter concludes that the proposed system can significantly improve the health and productivity of poultry, decrease mortality rates during early stages of growth, and increase the overall profitability of the farm.

CHAPTER 5

CONCLUSION

5.1 Introduction

Chapter 5 serves as a conclusion to the project/research conducted. It includes a summary of the project's main findings, the extent to which they meet the project objectives, and a review of the methodology and implementation of the project. Additionally, the chapter offers suggestions for future improvements and enhancements to the project or research.

5.2 Research Constraint

In this project, there were a few constraints that affected the implementation and outcomes. One of the main constraints was limited time. Due to the timeline of the project, there were limitations on the amount of data that could be collected, as well as the depth of analysis that could be performed. This made it challenging to fully explore the potential of the system and the AI predictions.

Chickens are susceptible to a range of health problems, including diseases, parasites, and infections. These can affect the productivity and profitability of the farm. Chickens require a balanced diet to thrive. Limited availability or poor-quality feed can impact their growth and productivity.

5.3 Future Works

Currently, the system is only using a few sensors, such as temperature and humidity sensors, water level sensors, and load cells. Other sensors, such as carbon dioxide sensors and ammonia sensors, can be integrated to provide a more comprehensive analysis of the environment inside the chicken coop.

The system can be further improved by integrating with automatic feeders and waterers, which can dispense feed and water based on the needs of the chickens. This would ensure that the chickens are well-fed and hydrated at all times.

The current system can be further enhanced by developing a mobile application that will enable farmers to access real-time information about their poultry farm from their mobile devices. This will provide farmers with the flexibility to monitor and control their farm while on the go.

REFERENCES

1. M. Munadi, M. N. Setiawan, A. H. Fauzi, A. Goni, I. Haryanto and N. Iskandar, "Study of Sensor Placement and Temperature Control Reference Value Based on Computational Fluid Dynamics in Closed-Type Broiler Chicken Cage," 2022 9th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), Semarang, Indonesia, 2022, pp. 90-95, doi: 10.1109/ICITACEE55701.2022.9923960.
2. Y. A. Liani et al., "The Broiler Chicken Coop Temperature Monitoring Use Fuzzy Logic and LoRAWAN," 2021 3rd International Conference on Electronics Representation and Algorithm (ICERA), Yogyakarta, Indonesia, 2021, pp. 161-166, doi: 10.1109/ICERA53111.2021.9538771.
3. Rasheed O. Ojo, Anuoluwapo O. Ajayi, Hakeem A. Owolabi, Lukumon O. Oyedele, Lukman A. Akanbi, Internet of Things and Machine Learning techniques in poultry health and welfare management: A systematic literature review, *Computers and Electronics in Agriculture*, Volume 200, 2022, 107266, ISSN 0168-1699, <https://doi.org/10.1016/j.compag.2022.107266>.
4. M. Abdurohman, A. G. Putrada and M. M. Deris, "A Robust Internet of Things-Based Aquarium Control System Using Decision Tree Regression Algorithm," in *IEEE Access*, vol. 10, pp. 56937-56951, 2022, doi: 10.1109/ACCESS.2022.3177225.
5. S. Waluyo and M. Efendi, *Beternak Ayam Broiler Tanpa Bau Tanpa Vaksin*, Jakarta:Agromedia Pustaka, pp. 27-28, 2016.
6. E. P. Wibowo, A. Wibisono, S. Nawangsari and A. Suritalita, "Prototype Of Feeding Devices, Temperatures And Humidity Monitoring At Broiler Chickens Breeders With The Internet Of Things Concept," 2018 Third International Conference on Informatics and Computing (ICIC), Palembang, Indonesia, 2018, pp. 1-5, doi: 10.1109/IAC.2018.8780448.
7. A. Faroqi, A. N. Utama, M. A. Ramdhani and E. Mulyana, "Design Of a Cage Temperature Monitoring System and Microcontroller Base On Automatic Chicken Feeder," 2020 6th International Conference on Wireless and Telematics (ICWT), Yogyakarta, Indonesia, 2020, pp. 1-5, doi: 10.1109/ICWT50448.2020.9243636.

8. Abd Karim, N. S. (2019). Urbanized Chicken Coop Monitoring System using IoT. *International Journal of Advanced Trends in Computer Science and Engineering*, 8(1.6), 450–454. <https://doi.org/10.30534/ijatcse/2019/6581.62019>.
9. Aamir Nawab, Fahar Ibtisham, Guanghui Li, Barbara Kieser, Jiang Wu, Wenchao Liu, Yi Zhao, Yasir Nawab, Kongquan Li, Mei Xiao, Lilong An, Heat stress in poultry production: Mitigation strategies to overcome the future challenges facing the global poultry industry, *Journal of Thermal Biology*, Volume 78, 2018, Pages 131-139, ISSN 0306-4565, <https://doi.org/10.1016/j.jtherbio.2018.08.010>.
10. Alemneh, & Getabalew. (2019). Factors Influencing the Growth and Development of Meat Animals. *Factors Influencing the Growth and Development of Meat Animals*. <https://doi.org/10.1201/9781439833094.ch2>.
11. Guzman, S.M., Paz, J.O. & Tagert, M.L.M. The Use of NARX Neural Networks to Forecast Daily Groundwater Levels. *Water Resour Manage* 31, 1591–1603 (2017). <https://doi.org/10.1007/s11269-017-1598-5>.
12. V. K. Saini, R. Kumar, A. Mathur and A. Saxena, "Short term forecasting based on hourly wind speed data using deep learning algorithms," 2020 3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE), Jaipur, India, 2020, pp. 1-6, doi: 10.1109/ICETCE48199.2020.9091757.
13. Muhammad, Maryam & L.U, Muhammad & Ambali, Abdul-Ganiyu & Mani, Aliyu. (2010). A Survey of Early Chick Mortality on Small-Scale Poultry Farms in Jos, Central Nigeria. *International Journal of Poultry Science*. 9. 10.3923/ijps.2010.446.449.