ANALYTICAL MODULE DEVELOPMENT FOR WASTE COLLECTION SYSTEM

RAJA ZAREEF FIRDAUS BIN RAJA AZMAN NAHAR

17207394/2

WIA3003 ACADEMIC PROJECT II

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

UNIVERSITY OF MALAYA

KUALA LUMPUR

SEMESTER 1 (2023/2024)

SUPERVISOR:

TS. DR. ISMAIL BIN AHMEDY

TABLE OF CONTENT

ABSTRACT

By 2050, there is a pressing need for more efficient garbage collection systems. The objective of this project is to analyze historical waste data to make prediction of waste production rates, to design and implement an analytical dashboard within the ThingsBoard platform as well as integrating it into a mobile application. Findings from the literature review indicates that the use of a prediction algorithm that can handle time series data is required. As a result, a waste sensor prototype is built to detect waste levels to be sent to Thingsboard Cloud. Then the data is aggregated & visualized into an interactive dashboard, which can be accessed via desktop or on mobile. Users can view waste forecast via a forecasting web application that uses the Prophet forecasting algorithm hosted on Streamlit Cloud. The development of this analytical module successfully answered the objective of this project & further improvements can be made to tailor the needs according to Malaysia's current situation.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

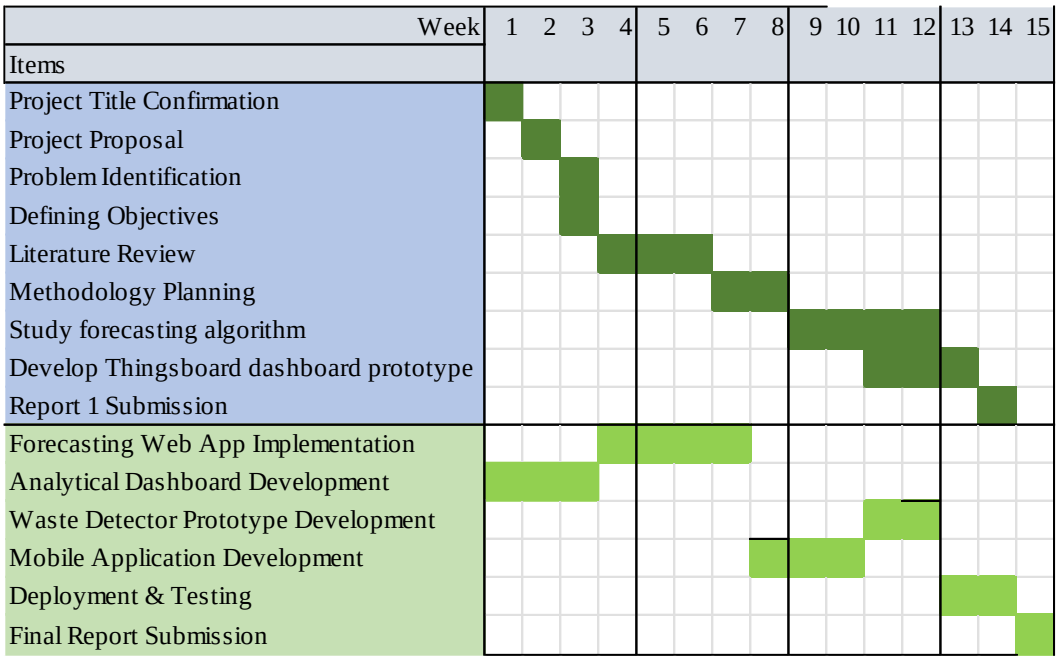
Waste management plays a vital role in contemporary society, ensuring the proper disposal and treatment of waste materials. However, with the increasing human activities that generate a substantial amount of waste, estimated to reach 3.40 billion tons by 2050 (World Bank, 2018), there is a pressing need for more efficient garbage collection systems. While existing research studies have proposed various solutions, a literature gap exists in the domain of dedicated analytical modules that provide comprehensive insights into waste generation and disposal patterns, specifically addressing waste level detection and forecasting.

To fill this gap, our project aims to develop an analytical module specifically designed for waste management systems, focusing on accurate waste level detection, and forecasting capabilities. By leveraging historical data on waste generation and collection, we employ machine learning models to make precise predictions about future waste generation patterns. This enables proactive planning of collection operations and efficient allocation of resources. The development of this analytical module involves designing and implementing a software component that utilizes machine learning algorithms to predict waste levels. We collect the necessary historical data from strategically placed Arduino sensors at disposal sites.

In addition to waste level detection and forecasting, our analytical module will provide a user-friendly interface for data exploration, visualization, and decision support. It will seamlessly integrate with existing waste management systems, allowing stakeholders to gain valuable insights into waste accumulation trends, optimize collection routes and schedules, and make informed decisions regarding resource allocation. By developing this module within the Thingsboard platform, we ensure compatibility and scalability.

The project timeline is divided into two semesters with 15 weeks each – FYP 1 & 2. FYP 1 focuses on project planning, proposal development & idea generation. This phase will also involve in-depth research on existing systems and relevant research papers to gain a comprehensive understanding of the problem statement. The subsequent 15 weeks, corresponding to the Final Year Project 2 timeframe, will be dedicated to developing the prototype, mobile application & testing.

Table 1:Project Schedule



CHAPTER 2: PROJECT OBJECTIVE

The SMART research objectives for this project are as follows:

- To analyze waste data through predictive analysis to identify trends and make prediction of waste production rates.
- To design and implement an analytical dashboard within the ThingsBoard platform to facilitate efficient data analysis.
- To integrate the analytical dashboard into a mobile application on ThingsBoard.

CHAPTER 3: LITERATURE REVIEW

3.1 Paper 1: Prediction of Municipal Waste Generation in Poland Using Neural Network Modeling. (2020)

The establishment of relationships between factors determining waste generation and the forecasting of waste management needs is widely recognized as fundamental in the development of effective planning strategies and the implementation of sustainable development (Kulisz & Kujawska, 2020). In this regard, proposed models incorporating selected explanatory indices have been utilized to reflect the influence of social, demographic, and economic factors on the amount of waste generated. Notably, the application of Artificial Neural Network (ANN) models has proven to be an effective and cost-efficient approach to planning integrated waste management systems, as observed in various studies (Kulisz & Kujawska, 2020; Azarmi et al., 2018; Azadi & Karimiashni, 2018). These models consider socioeconomic and demographic factors, enabling the training and verification of predictions using limited available waste data. The significance of creating an appropriate model for waste management and prediction is emphasized, recognizing the dependence of waste quantity and morphological composition on location, community financial status, consumption levels, and seasonal variations (Kulisz & Kujawska, 2020). The comparison of different predictive models, such as Multiple Linear Regression (MLR), Canonical Correlation Analysis (CCD), and ANN, has been conducted in specific contexts, such as predicting waste generation rates from hotels in North Cyprus (Azarmi et al., 2018) and seasonal municipal solid waste generation in Fars province, Iran (Azadi & Karimiashni, 2018). These studies collectively underline the importance of considering various factors and employing ANN models in waste management planning to achieve sustainable and efficient waste management practices.

3.2 Paper 2: Predictive analysis of urban waste generation for the city of Bogotá, Colombia, through the implementation of decision trees-based machine learning, support vector machines and artificial neural networks. (2019)

The use of machine learning algorithms, specifically the decision tree and support vector machine, emerges as a promising approach for analysing waste management data and accurately forecasting waste generation patterns. The inclusion of factors such as distribution by collection zone, socio-economic stratification, population, and waste quantity in the analysis enhances the accuracy of waste generation forecasts. Notably, artificial intelligence models, particularly support vector machines, demonstrate their efficacy in predicting monthly waste generation by leveraging waste generation time series data. The importance of conducting a comparative analysis of predictive models to identify the most suitable tool for waste generation forecasting is highlighted. The application of information and communication technologies (ICTs) and operations research (OR) methods in the field of solid waste management emphasizes the potential for integrating analytical modules into waste management systems. Furthermore, the creation of information files and data aggregation based on collection areas and operating companies provides valuable insights for informed decision-making in waste management. These findings collectively serve as a foundation for the development of the analytical module, emphasizing the significance of machine learning algorithms, comprehensive data analysis, and the incorporation of relevant factors to optimize waste management strategies and enhance forecasting capabilities.

3.3 Paper 3: Forecasting solid waste generation in Negeri Sembilan and Melaka (2021)

The study discussed the importance of forecasting solid waste generation rates for effective waste management strategies. Quantitative forecasting techniques, such as time series analysis and causal variable forecasting, were employed to predict future waste generation based on historical data. The ARIMA (Autoregressive Integrated Moving Average) time series model emerged as a reliable method for estimating and forecasting solid waste generation. The ARIMA models (e.g., ARIMA (2,1,2) and ARIMA (1,1,1)) demonstrated satisfactory performance in predicting waste production, considering factors like urbanization rates and population growth. Accurate waste generation forecasts are crucial for adequate waste management system planning, including landfill capacity estimation. The evaluation of forecast accuracy, through the separation of data into training and testing sets, was emphasized to ensure the suitability and reliability of the forecasting models. The use of ARIMA models for solid waste forecasting has been widely recognized, and their capability to handle different types of waste data makes them applicable in various scenarios. The findings highlight the need for an effective waste management system, informed decision-making, and accurate forecasting to address the growing waste generation challenges. In summary, the studies demonstrate the significance of the ARIMA time series model in predicting and forecasting solid waste generation, providing valuable insights and methodologies that can be leveraged in the development of the analytical module for waste management systems.

3.4 Findings from the Reviewed Literature

Based on the findings from the literature review, it is evident that various approaches exist for forecasting and predicting waste generation. However, for the specific focus of this project, the approach presented in paper 3 appears to be the most suitable. The study in paper 3 utilizes historical data of waste generation exclusively to train their forecasting model. This aligns well with the objectives of this project and enables the implementation of statistical methods like ARIMA within the Thingsboard platform without excessive computational requirements or dependencies on additional factors. Although papers 1 and 2 emphasize the advantages of using neural networks for prediction, the incorporation of factors such as population and socio-economic aspects into their models will not be applicable in the scope of this project. Therefore, the chosen approach aligns with the project's goals and resource constraints, ensuring a focused and efficient implementation within the waste management system.

Paper	Algorithm	Data Set	Technique	Pros	Cons
Paper 1	Artificial Neural Network (ANN)	Socioeconomic and demographic factors Waste Quantity	Training and verification using limited data	- Effective and cost-efficient approach for planning waste management systems. - Capable of incorporating various factors for accurate predictions. - Flexible and adaptable for different contexts.	- Requires enough data for training and verification. - Complex implementation and interpretation compared to traditional statistical models. - May require domain expertise to select relevant explanatory indices.
Paper 2	Recurrent Neural Network (RNN), Support Vector Machine (SVM)	Distribution by collection zone, population, etc. Daily municipal waste.	Comparative analysis of predictive models	- Accurate forecasting of waste generation patterns. - Integration of analytical modules into waste management systems. - Capability to handle complex data with multiple variables.	- May be computationally intensive for large datasets. - Interpretability of the models may be challenging.
Paper 3	ARIMA (Autoregressive Integrated Moving Average)	Historical waste generation data	Time series analysis and causal variable forecasting	- Reliable method for estimating and forecasting solid waste generation. - Applicable in various scenarios. - Handles different types of waste data.	- Requires proper identification of appropriate ARIMA parameters. - Assumptions of linear relationships and stationarity may not hold in all cases.

Table: Literature Review Comparison

CHAPTER 4: PROBLEM STATEMENTS

Waste management systems play a crucial role in achieving sustainable development, preserving the environment, and safeguarding public health. However, the increasing population and evolving consumption patterns have posed significant challenges in effectively managing waste accumulation. The resulting burden on waste management systems has led to adverse consequences, including pollution, health risks, and financial strains. Therefore, there is an urgent need to develop a predictive module that can accurately forecast waste accumulation and optimize waste management processes.

The objective of this project is to develop an analytical module for a waste management system that utilizes machine learning algorithms and data analytics techniques, in this case by implementing time series forecasting techniques. By leveraging historical data and employing predictive modelling, the module aims to forecast the locations that will reach maximum waste capacity first. This information will empower stakeholders such as academics, policymakers, government organizations, and municipalities to make informed decisions and formulate sustainable waste management strategies in Malaysia. The insights provided by the analytical module will facilitate proactive planning, resource allocation, and efficient waste collection, thereby contributing to effective waste management practices and environmental preservation.

In the ideal situation, the analytical module will enable stakeholders to anticipate and address waste accumulation challenges before they become critical. By accurately predicting the locations at risk of reaching maximum waste capacity, timely interventions can be implemented, such as redistributing waste collection resources or implementing alternative waste disposal methods. This proactive approach will alleviate the strain on overloaded waste management systems, mitigate pollution, minimize health hazards, and optimize financial resources.

Through the development of the analytical module, this project seeks to bridge the existing gap in waste management systems by providing a reliable and data-driven tool for forecasting waste accumulation. The module's integration of machine learning algorithms and data analytics techniques serves as a proof-of-concept that will enable stakeholders to make evidence-based decisions and develop sustainable waste management strategies tailored to the specific needs of Malaysia. By optimizing waste collection, resource allocation, and planning processes, the project aims to contribute to effective waste management practices and promote environmental preservation in the country.

CHAPTER 5: METHODOLOGY

To ensure that the system design complies with the project expectations and meets all the project objectives, the following methodology/technique/approach was adopted:

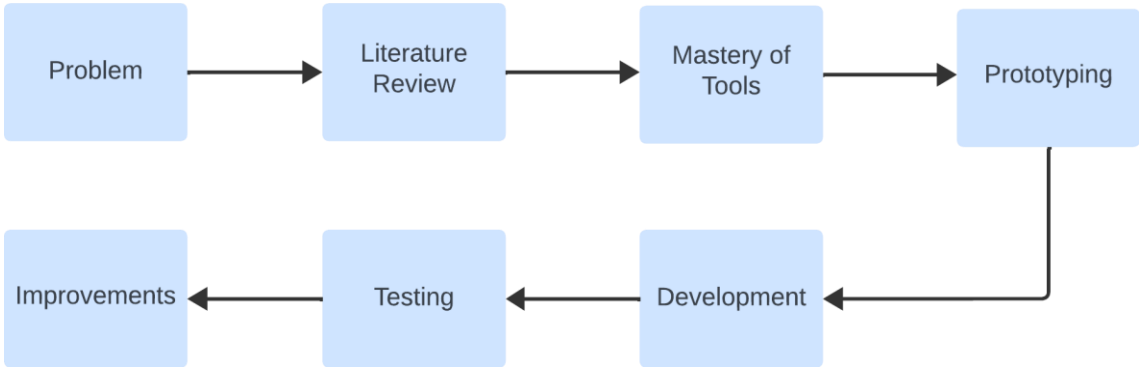


Figure 5.1: Project Methodology

5.1 Problem Statement

Before beginning to develop the system, the first phase of the project is to study the existing problem, which is the problem faced by the waste collection companies in collecting solid waste. In this phase, I conducted research and studies to find out they face increasing challenges due to population growth and consumption patterns which causes problems faced by the waste collection companies when assigning their resources to collect the solid waste from area to area. After gathering all the problems faced by the companies, the requirements are stated out and are discussed with the supervisor of the project to seek approval. This project's objective is to develop an analytical module using machine learning algorithms and data analytics techniques to predict waste accumulation and optimize waste management processes.

5.2 Literature Review

Under the literature review phase, I have better understanding towards the problems faced by the waste collection companies and how other researchers proposed different solutions in predicting the accumulation of waste. I also discovered different tools that can were used by other researchers in running their predictions for waste collection. The paper I

have chosen includes the general ideas on the issues in waste management, the approach and tools used such as the implementation of ARIMA models in time-series forecasting.

5.3 Master the Tools

In this phase, I learnt the basic concept of Thingsboard dashboard, Flutter SDK, various programming languages & protocols such as Dart, JavaScript, C & Python as well as HTTP requests, APIs & MQTT via online resources and documentations. The system proposed is developed using these tools. I need to understand how to use Flutter SDK to build the mobile application which implements the Thingsboard dashboard into it. The Thingsboard cloud will function as the database for the proposed system. Mastering the basic concept of these tools will help in designing and developing the mobile application which uses the Thingsboard dashboard and connects to the Thingsboard Cloud. The mastery of Python is also important as it is the most common language used to build a machine learning model & web application that can communicate with Thingsboard Cloud via HTTP requests using APIs, whereas C & MQTT is used to code the logic & connect the sensor devices (Arduino UNO) to Thingsboard respectively.

5.4 Prototyping

Lastly, a prototype has been developed to simulate the final product. It is a quick model to show and explain the features of the proposed system. The prototype is constantly developed, built, tested, and improved to get the best possible for the system.

5.5 Development

For the development phase, Thingsboard and Flutter are configured, and data is inputted into the database through Thingsboard dashboard. Then JavaScript is implemented within the dashboard to redirect the user to the forecasting web application. After inputting the data into the database, we can fetch data from Thingsboard cloud using HTTP request to obtain all the historical data or telemetry to be fed into our forecasting model. Python

programming language is used to fetch & modify the data and store into variables so that it can be used to train the forecasting model & present the predictions to the user. After the system is done developing, it will be evaluated according to various criteria. If any problems or limitations are found, improvement will be made to the proposed system.

CHAPTER 6: SYSTEM ANALYSIS & DESIGN

6.1 System Architecture

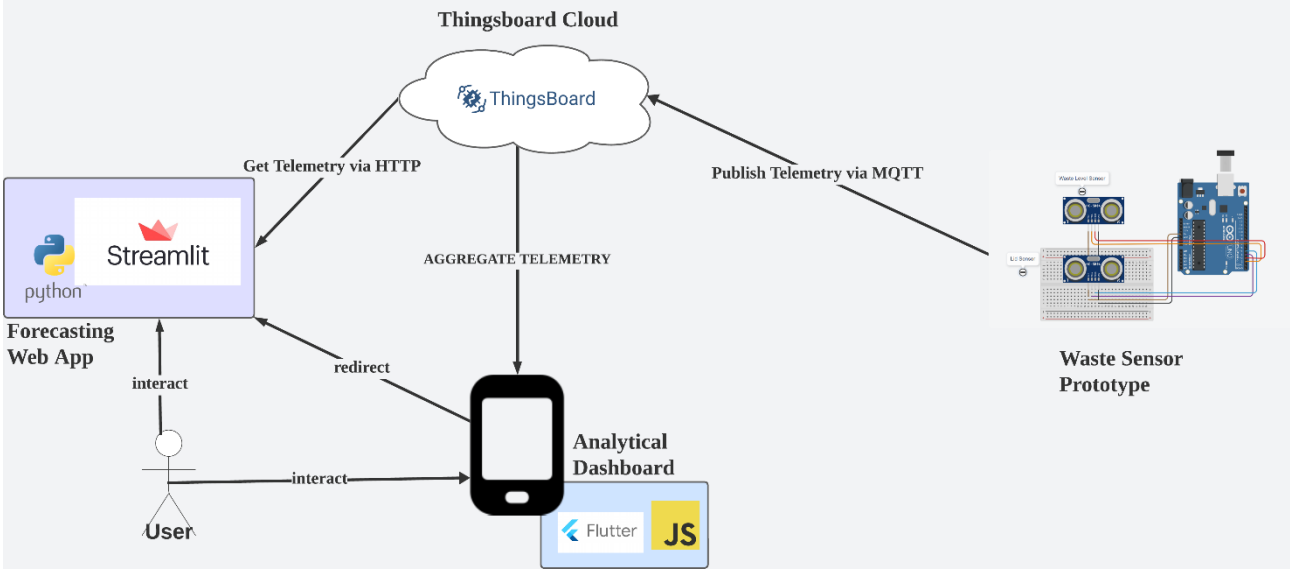


Figure 6.1: System Architecture

Based on Figure 2, the system architecture consists of the following components:

- Sensor devices: These are the Arduino UNO devices that are installed at waste collection bins and other strategic locations to collect data on waste levels, location, and other relevant parameters.
- Thingsboard Cloud: This is an open-source IoT platform that provides a central hub for collecting, storing, and managing data from the sensor devices. In this architecture, it serves as the main database for the telemetry data.
- Forecasting web app: This is a web app that uses machine learning algorithm to forecast waste accumulation levels at different locations by retrieving all the historical data of a device from Thingsboard Cloud via HTTP request with an access token. In this case, Python is used to build the model by implementing Prophet as the forecasting algorithm & is hosted on Streamlit Community Cloud.
- Analytical dashboard: This is a dashboard within Thingsboard that aggregates the data from various devices retrieved from Thingsboard Cloud & visualizes the data to provide stakeholders with insights into the waste accumulation data. The

dashboard redirects the user to the forecasting web application when triggered. It is supported by Flutter & JavaScript for the frontend & backend aspect respectively.

6.2 System Design

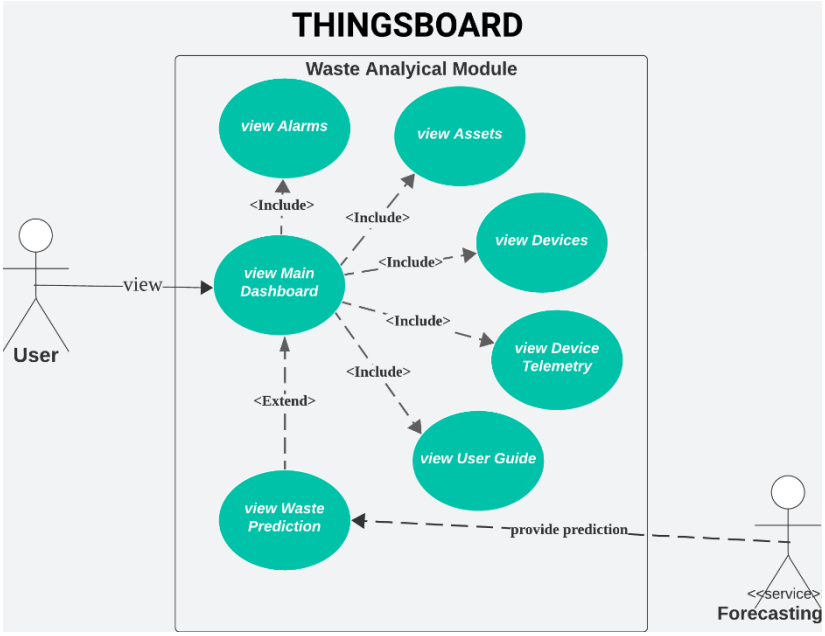


Figure 6.2: System Use Case Diagram

The use case diagram for the waste analytical module in Figure 3 shows the following functionality:

- View alarms: The system generates an alarm when the waste accumulation level at an asset reaches a predefined threshold. The user can view the alarm and take appropriate action, such as dispatching a waste collection crew.
- View assets: The user can view a list of all assets monitored by the system. The user can click on an asset to view more information, such as its location and the sensor device that is assigned to that location.
- View devices: The user can view a list of all devices installed at an asset. The user can click on a device to view more information, such as the history of data it collects and the frequency of data collection.

- View devices telemetry: The user can view real-time data collected by the sensor devices. The user can use this data to track trends in waste accumulation and identify potential problems.
- View main dashboard: The user can view an overview of the waste accumulation data for all assets. The dashboard shows the current waste level, the current number of active alarms, total active devices, and a user guide. The user can use this information to identify assets that are at risk of reaching maximum waste capacity.
- View waste prediction: The user can view the forecasting results for waste accumulation levels at different locations. The user can use this information to plan waste collection routes and allocate resources more efficiently.

The use case diagram also shows the following actors:

- User: This actor represents any user of the waste analytical module, such as academics, policymakers, government organizations, and municipalities.
- Forecasting web app: This actor represents the web app that uses machine learning algorithms to forecast waste accumulation levels.

The simplified overview of this project is as follows:

1. Waste sensor prototype sends raw telemetry to ThingsBoard Cloud via MQTT.
2. The telemetry is aggregated to be visualized on the analytical dashboard - mobile & desktop view.
3. The waste level of a certain location for the next 7 days can be predicted using the Forecasting Web App hosted on Streamlit.

6.3 System Requirements

The system requirements section aims to outline the necessary functional and non-functional requirements for the development and implementation of an effective analytical module in a waste management system. By defining these requirements, we establish a clear understanding of the system's expected functionalities and desired qualities. This section will be divided into two parts: functional requirements and non-functional requirements.

6.3.1 Functional Requirements

1. Capable of utilizing historical data to predict early insights of potential waste capacity issues.
2. The analytical dashboard should be implemented within ThingsBoard.
3. The dashboard should allow relevant stakeholders to visualize and analyse waste accumulation data effectively.

6.3.2 Non-functional Requirements

Non-functional requirements are criteria that define the overall behaviours, quality, and constraints of a system rather than its specific functionalities. In the context of this project, the following non-functional requirements can be considered:

1. Performance: The system should be capable of processing and analysing large volumes of data efficiently and provide real-time or near-real-time results. It should be able to handle simultaneous user interactions without significant delays or performance degradation.
2. Reliability: The system should be always reliable and available for use. It should have mechanisms in place to handle system failures, minimize downtime, and ensure data integrity and accuracy. Additionally, it should be able to recover from unexpected errors or crashes without data loss.

3. Scalability: The system should be designed to accommodate future growth and an increasing number of users and waste management sites. It should be scalable enough to handle additional sensors, data sources, and users without compromising performance or functionality.

CHAPTER 7: SYSTEM DEVELOPMENT & TECHNICAL IMPLEMENTATION

7.1 Interactive Analytical Dashboard

This dashboard is implemented within Thingsboard itself. Firstly, to implement the dashboard, the data needs to be aggregated & stored within Thingsboard. This can be done by configuring the rule chain as follows:

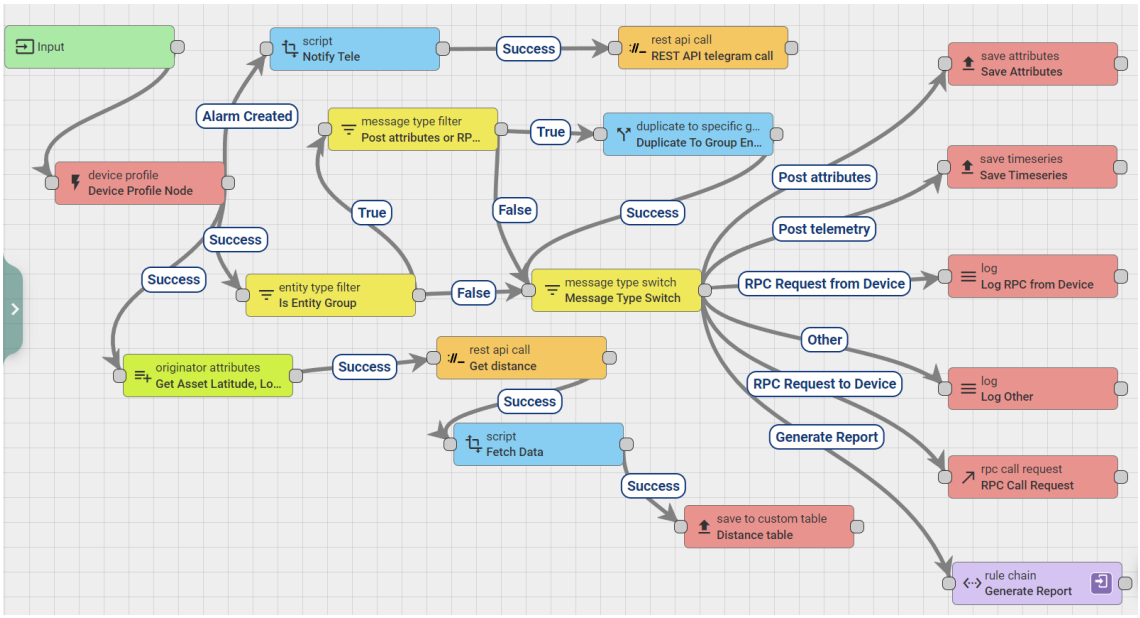


Figure 7.1: Level Sensor Rule Chain

After that, relevant devices & assets can be connected to the dashboard, then the dashboard is built by including different types of interactive widgets that allows users to navigate to different sub-dashboards, this step can be done by referring to the Dashboard Thingsboard Development Guide (https://www.youtube.com/watch?v=0io1YnQjIwA) on Thingsboard's official YouTube channel. The way that the interactive analytical dashboard for this project is structured as such:

Figure 7.2: Analytical Dashboard Structure

As mentioned previously, the users can navigate to different dashboards, each dashboard serves a specific purpose. These are the dashboards implemented.

- **Main Dashboard**: Essentially the home page for the dashboard where the overall functionality is visualized, from here the users can drill into certain dashboards through the widgets.

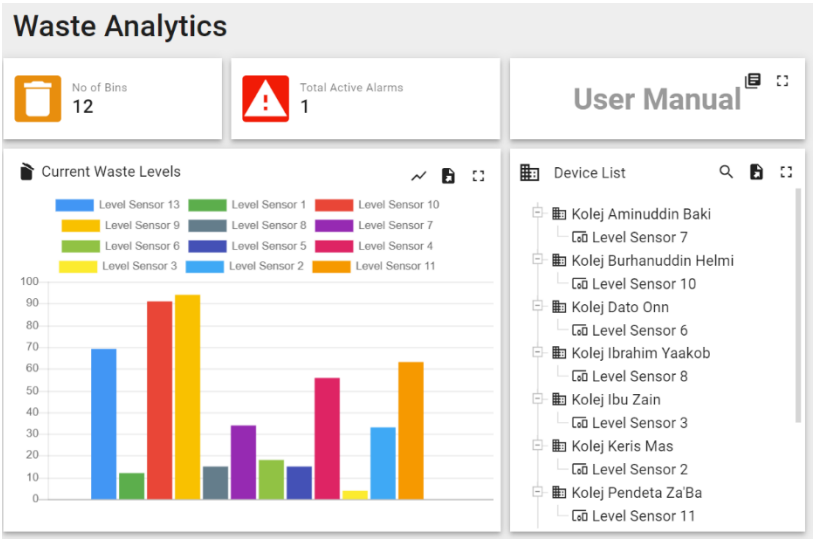


Figure 7.3: Main Dashboard

- **User Manual**: Manual for first-time users to review the functionalities available.

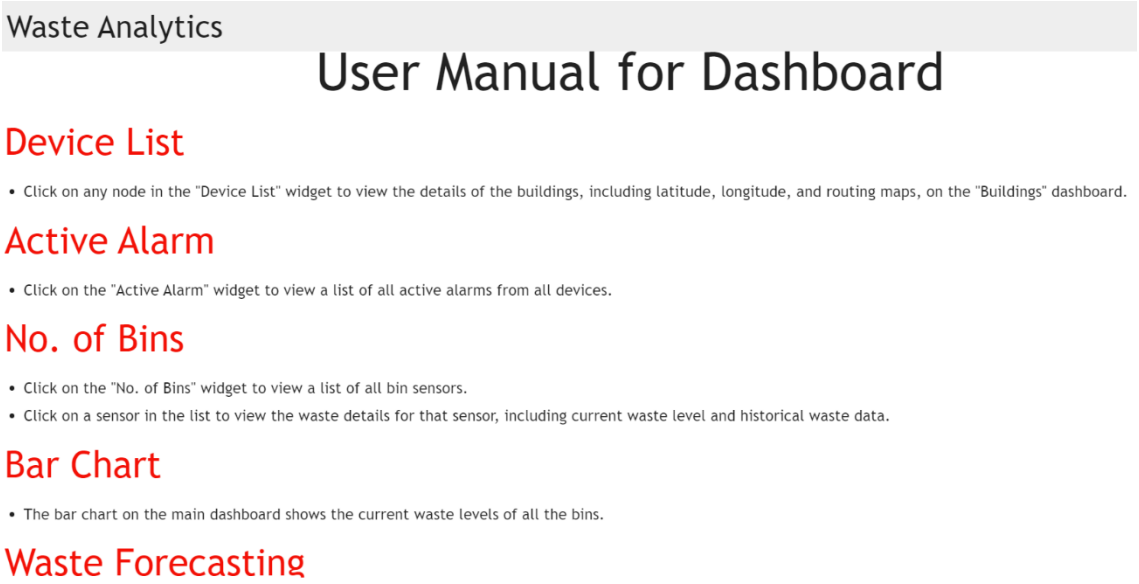


Figure 7.4: User Manual

- **Alarm Dashboard**: Lists out all the details of the alarms which includes the alarm type, originator, severity & status.

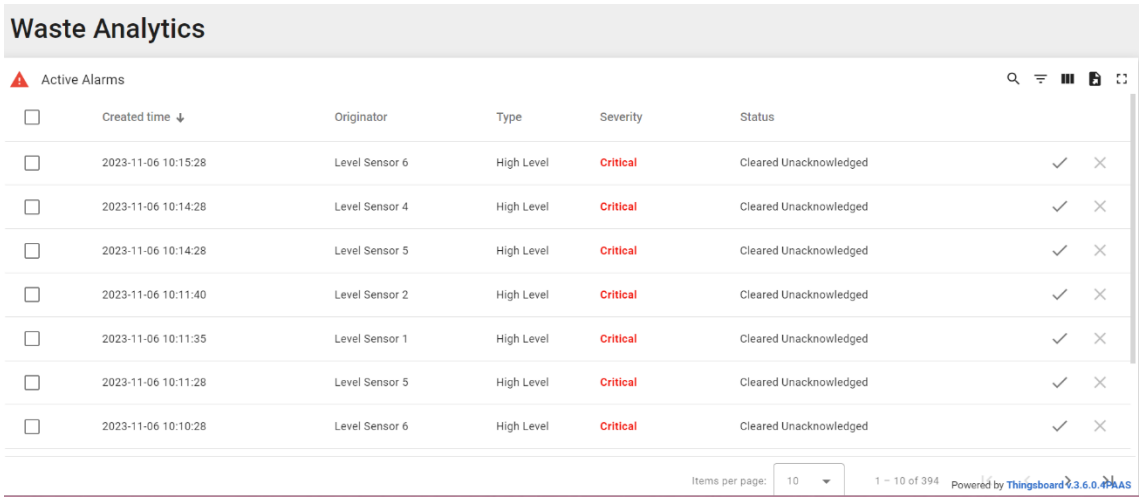


Figure 7.5: Alarm Dashboard

- **Asset Dashboard**: Shows the assets (buildings) associated with each device as well as the location of these assets.

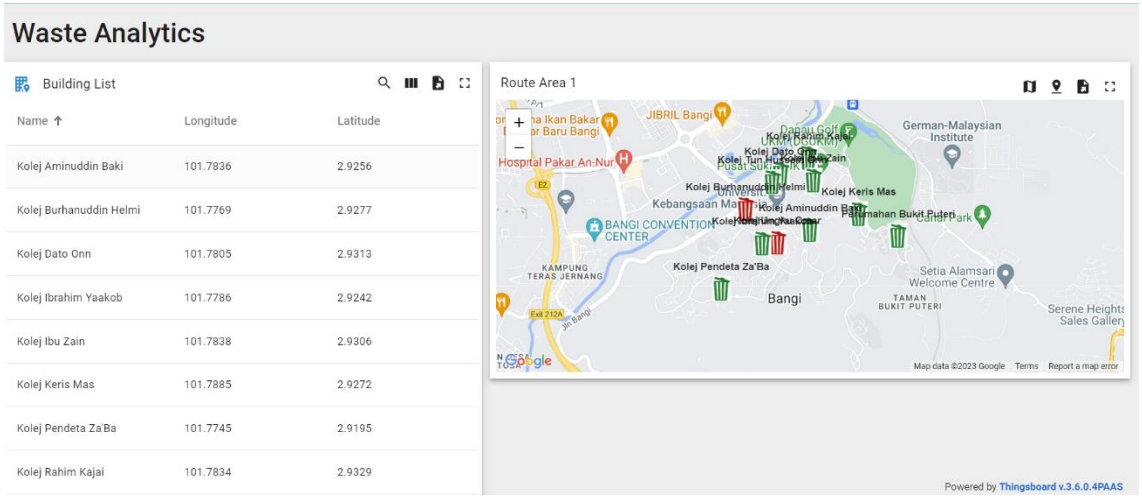


Figure 7.6: Assets Dashboard

- **Device Sensor List**: Lists out all the devices along with the latest telemetry.

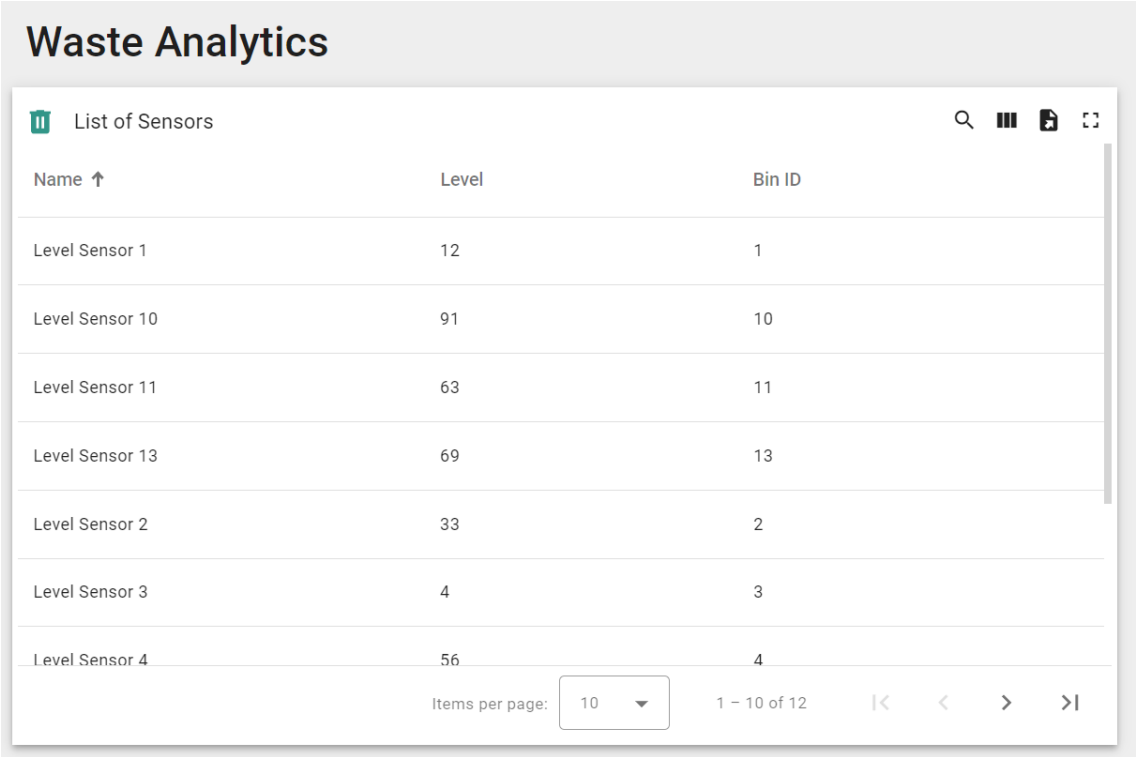


Figure 7.7: Sensor List

- **Sensor Telemetry**: This shows a deeper insight into the telemetry collected by the device, it includes the alarm, historical data & the latest telemetry captured by that device.

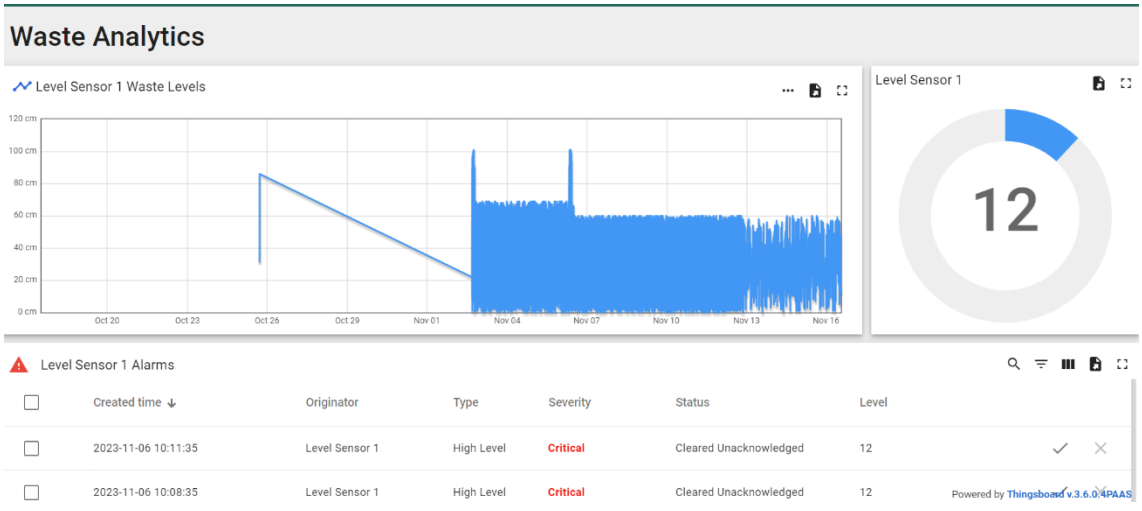


Figure 7.8: Sensor Telemetry

This interactive analytical dashboard fulfils objective 2 & requirements 2 & 3 where we can design and implement an analytical dashboard within the ThingsBoard platform

to facilitate efficient data analysis. Furthermore, the analytical dashboard is implemented within ThingsBoard and allows relevant stakeholders to visualize and analyze waste accumulation data effectively.

7.2 Forecasting Web Application

For the forecasting web application, it is built using Python & hosted on Streamlit Cloud. The prediction algorithm implemented for this forecasting module is Prophet. Prophet (formerly known as FBProphet) is a forecasting algorithm developed by Facebook's data science team in 2017. The algorithm is designed to be scalable, fast, and accurate, making it suitable for a wide range of applications, from predicting sales in e-commerce to forecasting weather patterns. Prophet employs a Fourier series to model seasonality in the data. It decomposes the time series into components for yearly, weekly, and daily seasonality. The Fourier series is a mathematical representation that allows the model to capture repetitive patterns at different time scales. It uses a piecewise linear or logistic growth curve to capture changes in trends at different points in time. This enables the model to adapt to varying patterns in waste generation.

Since this forecasting module cannot be integrated into the Thingsboard platform, the alternative is to allow the users to be redirected to the web application. This can be done by utilizing the custom widget header button within Thingsboard & configure the action as such:

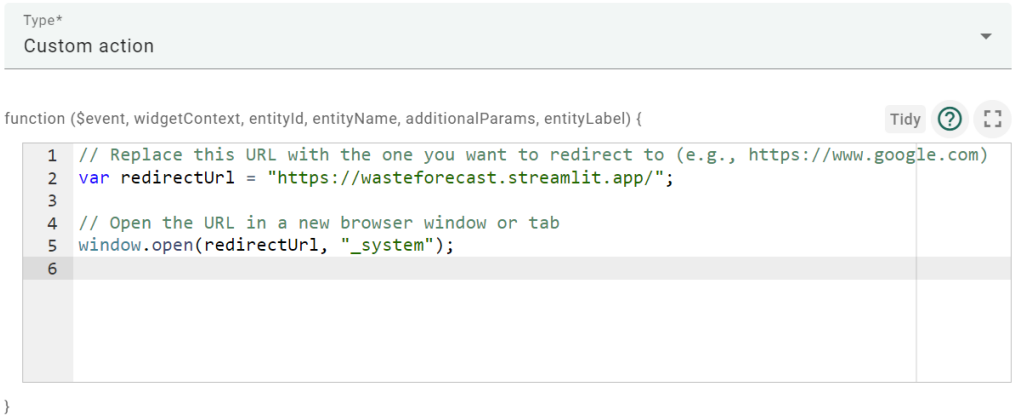


Figure 7.9: Custom Widget Action

Once the user is redirected to the web application, the flow of the process are as follows:

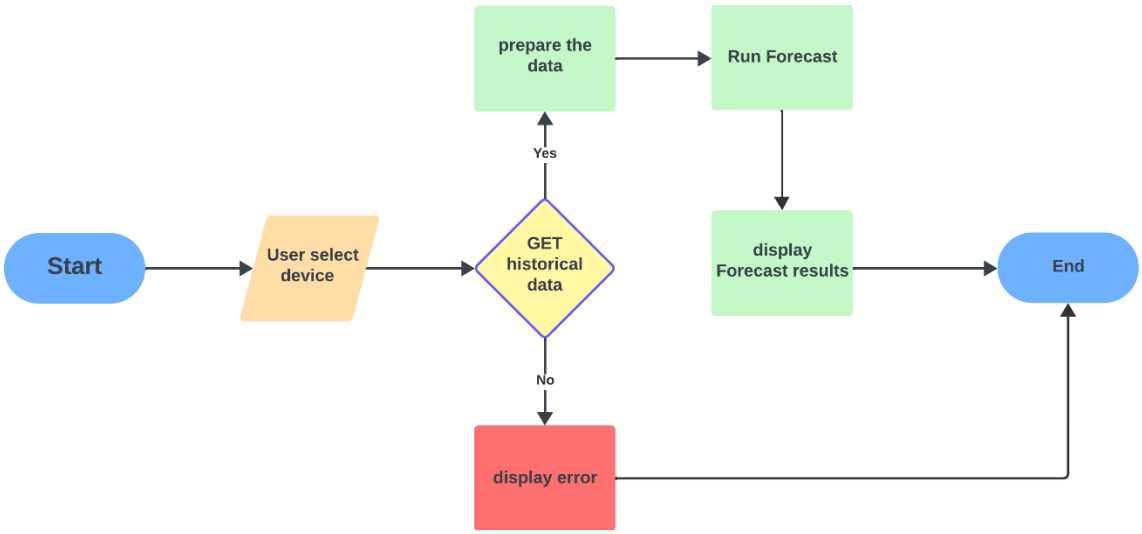


Figure 7.10: Forecasting Flowchart

1. Raw telemetry is retrieved from ThingsBoard Cloud via HTTP GET Request. The request is done by using the Thingsboard API retrieved from https://thingsboard.cloud/swagger-ui/#/telemetry-controller/getTimeseriesUsingGET
2. The data will be prepared & modified according to a suitable JSON format.
3. The modified data will be given to the Prophet model to be trained & run the prediction.

4. The prediction results will be displayed to the user, including interactive line graphs to show trends & seasonality.

The full code for this can be found on my Github page:https://github.com/zareefrj/Waste_Forecasting_FYP

For the hosting part, the selected platform is Streamlit Cloud, as it is free. The requirements.txt file is needed for the Streamlit Cloud to install all the dependencies needed to run the forecasting module smoothly. The exact demonstration on the deployment steps can be found on YouTube. After deploying, you can still edit the code anytime by changing it on your GitHub repository that is linked to the Streamlit Cloud dashboard, you can also change your web application's URL to your needs after deployment. This flexibility is the main reason why Streamlit Cloud is chosen for the hosting.

To run the GET requests successfully, you would need to get the access token of the Thingsboard user to be plugged into the forecasting web app. As a safety feature, you can paste the access token directly into the Streamlit Cloud dashboard instead of manually changing the code. This can be done at the "secrets" tab:

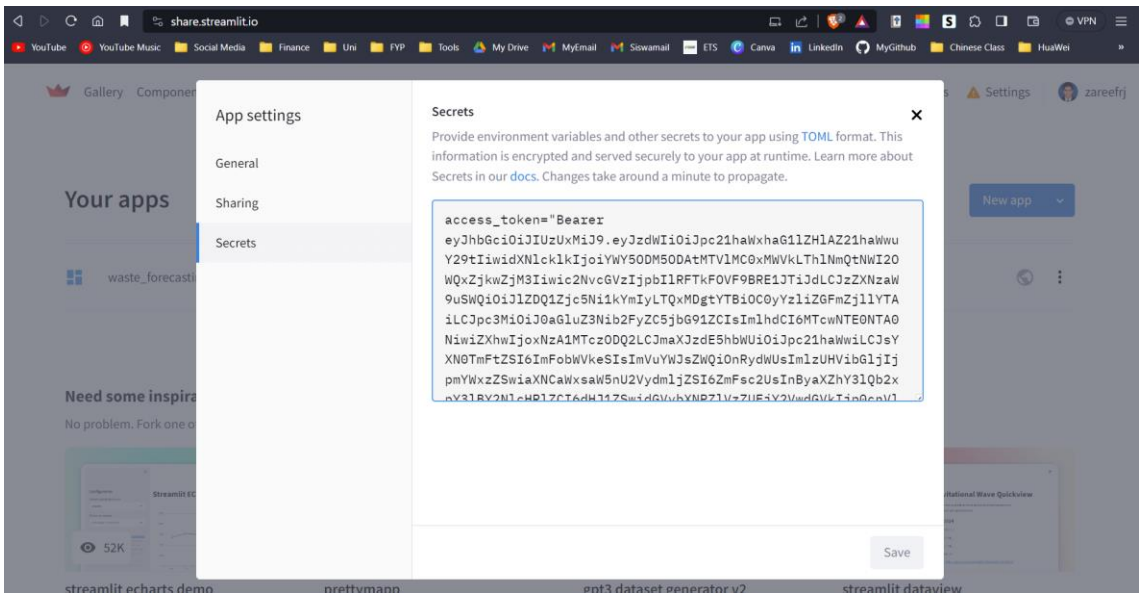


Figure 7.11: Secrets Tab

The user's access token can be generated from the account profile on Thingsboard by clicking the "Copy JWT Token", then paste it into the secrets tab shown above:

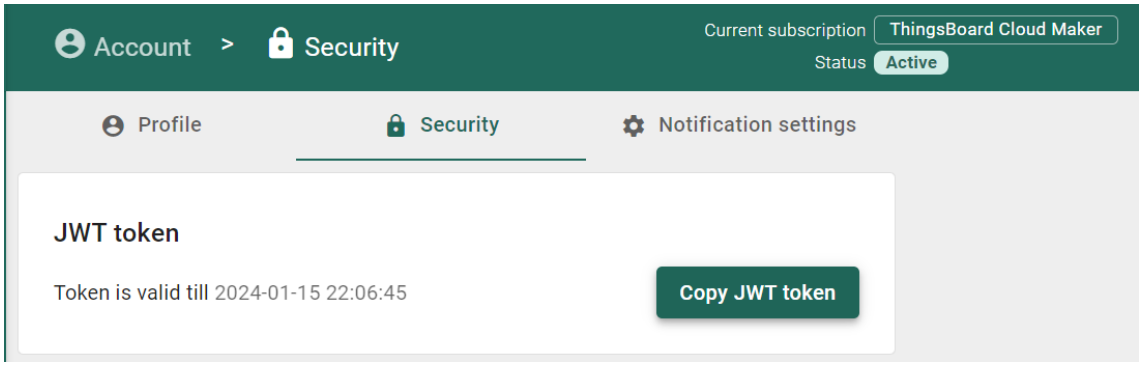


Figure 7.12: Getting Access Token

This forecasting web application fulfils objective 1 & requirement 1 where it can analyze waste data through predictive analysis to identify trends and make prediction of waste production rates and capable of utilizing historical data to predict early insights of potential waste capacity issues respectively.

7.3 Thingsboard Mobile Application

The mobile application version of Thingsboard is developed by cloning the Flutter ThingsBoard PE Mobile Application package from their official github page. The version used in this project is version 1.0.8. The steps on how to prepare your environment & configure the mobile application is on this page: https://thingsboard.io/docs/pe/mobile/getting-started/ . The tools used for this will be the Flutter SDK. On any IDE of your choice, after completing the configuration as instructed, run the "flutter run" command to build the app, for this part you must run it in sync with Android Studio to run an Android emulator to mimic the Android environment as shown below:

Once everything is running smoothly, you can release the app as an apk file so that you can install the mobile app on your actual Android devices. You can release the app by running this command: "flutter build apk –release".

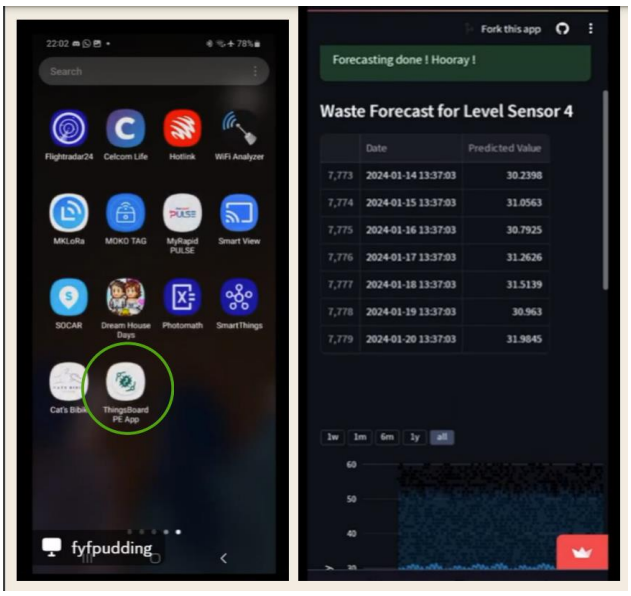


Figure 7.13: Thingsboard PE App

This mobile application fulfils objective 2 & 3 and requirements 2 & 3 where we can design and implement an analytical dashboard within the ThingsBoard platform to

facilitate efficient data analysis and integrate the analytical dashboard into a mobile application on ThingsBoard. Furthermore, the mobile application is implemented within ThingsBoard and allows relevant stakeholders to visualize and analyze waste accumulation data effectively.

7.4 Sensor Device

The waste sensor device prototype is built to detect waste levels of a bin & sends the telemetry to Thingsboard Cloud. This prototype is built using an Arduino UNO WiFi Rev 2 & two ultrasonic sensors. The circuit diagram & schematics for this prototype are as follows:

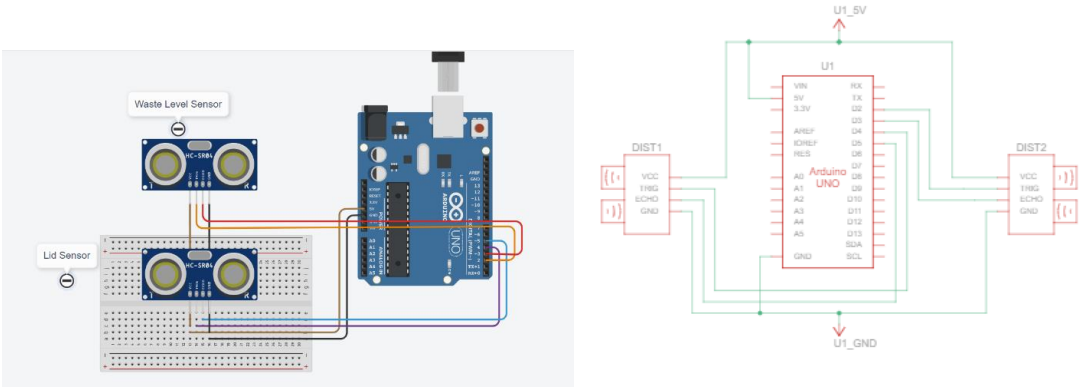


Figure 7.14: Prototype Circuit Diagram & Schematics

The code is compiled in C using Arduino IDE. The logic behind the functionalities of this prototype are as follows:

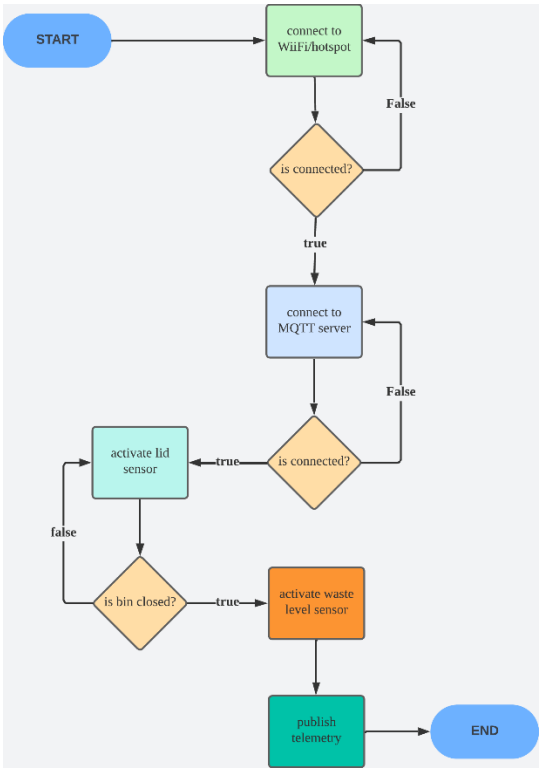


Figure 7.15: Prototype Flowchart

1. Connect to Wi-Fi or hotspot.
2. Connect to ThingsBoard MQTT server.
3. Check whether bin is closed, only detect waste level when bin is closed.
4. Publish telemetry to ThingsBoard via MQTT.

The MQTT client ID of the device can be retrieved from the device profile on Thingsboard:

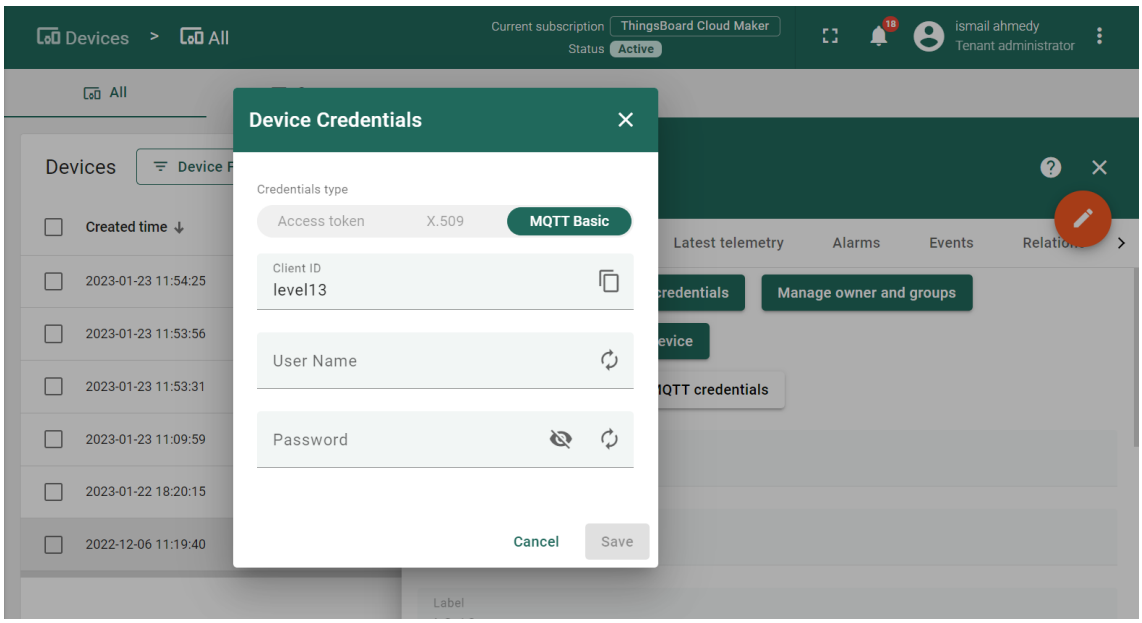


Figure 7.16: MQTT Device Credentials

Here is the code for the prototype:

```
#include <WiFiNINA.h>
#include <PubSubClient.h>

//***********Connectivity*************************
#define ssid "Raja Zareef"
#define pass "matstopa"

const char* mqttServer = "mqtt.thingsboard.cloud"; // ThingsBoard MQTT
broker URL
const char* deviceId = "level13"; // mqtt client ID

WiFiClient wifiClient;
PubSubClient client(wifiClient);

//*************Ultrasonic sensor******************
//sensor 1 to check waste level
const int sensor1TrigPin = 2;    // Ultrasonic sensor 1 trigger pin
const int sensor1EchoPin = 3;    // Ultrasonic sensor 1 echo pin
//sensor 2 to check whether bin is opened or closed
const int sensor2TrigPin = 4;    // Ultrasonic sensor 2 trigger pin
const int sensor2EchoPin = 5;    // Ultrasonic sensor 2 echo pin
const int thresholdDistance = 5; // Threshold distance for sensor 2 in
centimeters
float duration, distance, level;

void setup() {
  Serial.begin(9600);
  connectWiFi(); //connect to hotspot
```

```
  client.setServer(mqttServer, 1883); // MQTT broker port
  pinMode(sensor1TrigPin, OUTPUT);  // Set sensor 1 trigger pin as
output
  pinMode(sensor1EchoPin, INPUT);   // Set sensor 1 echo pin as input
  pinMode(sensor2TrigPin, OUTPUT);  // Set sensor 2 trigger pin as
output
  pinMode(sensor2EchoPin, INPUT);   // Set sensor 2 echo pin as input
}

void loop() {
  //connect to server first
  if (!client.connected()) {
    reconnect();
  }
  getAndSendLevel();
  delay(30000); //delay 30 seconds before sending the next telemetry
}

//Wifi connection
void connectWiFi() {
  WiFi.begin(ssid, pass);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connected to Wi-Fi");
}

//MQTT connection
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    if (client.connect(deviceId)) {
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}

void getAndSendLevel() {
  if (IsBinClosed()) {
    // Sensor 2 detected a distance less than the threshold, indicating
the bin is closed
    Serial.println("Bin is closed.");

    // Delay to avoid interference from sensor 2
```

```
    delay(1000);

    // Proceed to measure distance from sensor 1 for waste level
    digitalWrite(sensor1TrigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(sensor1TrigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(sensor1TrigPin, LOW);
    duration = pulseIn(sensor1EchoPin, HIGH);
    distance = duration * 0.034 / 2;
    level = 35 - distance;

    if(level<=0){
      level = 0;
    }

    // Print the measured distance from sensor 1 to the Serial Monitor
    Serial.print("Sensor 1 Distance: ");
    Serial.print(level);
    Serial.println(" cm");

    // Check if any reads failed and exit early (to try again).
    if (isnan(distance)) {
      Serial.println("Failed to read from Level sensor!");
      return;
    }

    // Publish telemetry data to ThingsBoard using MQTT
    String payload = "{\"Level\": " + String(level, 2) + "}";
    char buffer[payload.length() + 1]; // Add space for null terminator
    payload.toCharArray(buffer, payload.length() + 1); // Convert
String to C-style string
    client.publish("v1/devices/me/telemetry", buffer); // Publish the
C-style string payload
    Serial.println(buffer);

  } else {
    Serial.println("Bin is open. Please wait for it to close");
  }

}

bool IsBinClosed() {
  // Measure distance from sensor 2 to determine whether bin is closed
or opened
  digitalWrite(sensor2TrigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(sensor2TrigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(sensor2TrigPin, LOW);
  duration = pulseIn(sensor2EchoPin, HIGH);
```

```
  distance = duration * 0.034 / 2;
  Serial.print("Sensor 2 Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  return (distance <= thresholdDistance);
}
```

Here's the breakdown of the code:

1. **Constants:**
    - SSID: WiFi network name.
    - PASSWORD: WiFi network password.
    - MQTT_SERVER: ThingsBoard MQTT broker URL.
    - DEVICE_ID: MQTT client ID.
    - SENSOR1_TRIG_PIN, SENSOR1_ECHO_PIN: Pins for ultrasonic sensor 1 (waste level).
    - SENSOR2_TRIG_PIN, SENSOR2_ECHO_PIN: Pins for ultrasonic sensor 2 (bin open/close).
    - THRESHOLD_DISTANCE: Distance threshold for sensor 2 to determine if the bin is closed.
2. **Variables:**
    - duration, distance: Variables for measuring time and distance with ultrasonic sensors.
    - level: Variable to represent the calculated waste level.
3. **Setup Function:**
    - Initializes serial communication and connects to the WiFi network.
    - Sets up the MQTT server and port.
    - Configures pins for ultrasonic sensors.
4. **Main Loop:**
    - Checks if the MQTT client is connected. If not, attempts to reconnect.
    - Calls the **getAndSendLevel** function.

- Delays execution for 30 seconds before sending the next telemetry.

5. **Connect WiFi Function:**
    - Connects to the specified WiFi network, printing progress dots until connected.
6. **Reconnect Function:**
    - Attempts to reconnect to the MQTT server.
    - If successful, prints "connected"; otherwise, prints the connection state and retries after a 5-second delay.
7. **Get and Send Level Function:**
    - Checks if the bin is closed using ultrasonic sensor 2.
    - If closed, delays to avoid interference and then measures waste level with ultrasonic sensor 1.
    - Publishes the calculated waste level to the ThingsBoard server via MQTT.
    - Prints the telemetry data to the Serial Monitor.
8. **Is Bin Closed Function:**
    - Measures distance using ultrasonic sensor 2 to determine if the bin is closed.
    - Returns **true** if the distance is less than or equal to the threshold, indicating the bin is closed; otherwise, returns **false**.

This program essentially monitors the status of a bin by using two ultrasonic sensors. If the bin is closed, it calculates the waste level and sends this information to ThingsBoard using MQTT. It periodically repeats this process with a delay of 30 seconds between each telemetry update. The Serial Monitor is used for debugging and displaying sensor readings.

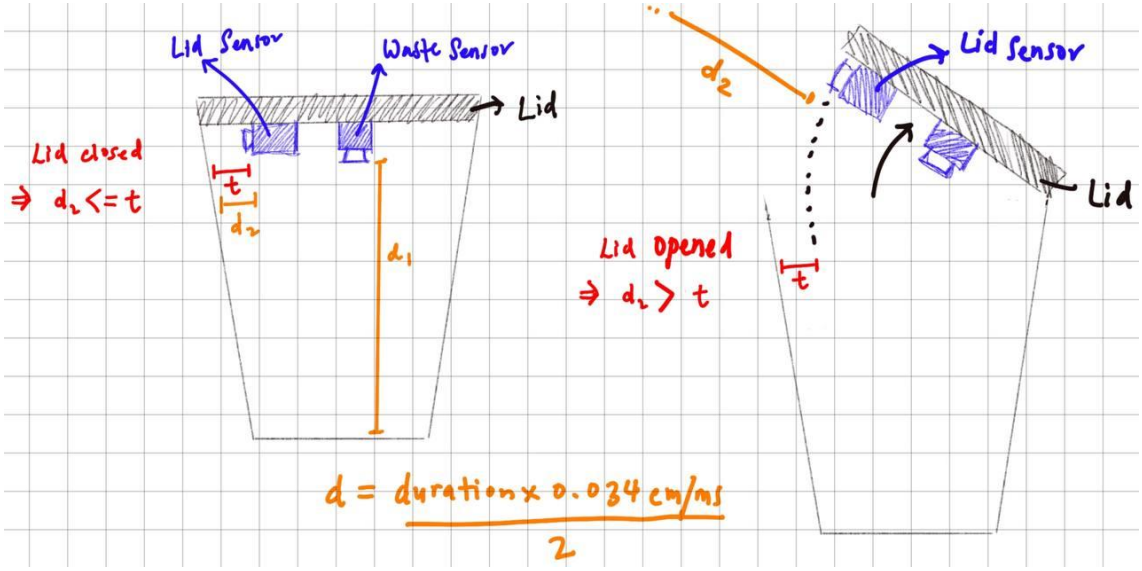


Figure 7.17: Prototype Design

1. **Triggering the Ultrasonic Sensor (Waste Sensor):**
    - **digitalWrite(sensor2TrigPin, LOW);**: Initially, the trigger pin of the ultrasonic sensor is set LOW to ensure it's in a stable state.
    - **delayMicroseconds(2);**: A small delay is introduced to ensure stability before triggering the sensor.
2. **Sending the Ultrasonic Pulse:**
    - **digitalWrite(sensor2TrigPin, HIGH);**: The trigger pin is then set to HIGH to send an ultrasonic pulse.
    - **delayMicroseconds(10);**: The HIGH signal is maintained for a brief period (10 microseconds in this case) to generate the ultrasonic wave.
    - **digitalWrite(sensor2TrigPin, LOW);**: The trigger pin is set back to LOW, concluding the pulse transmission.
3. **Measuring the Echo Duration:**
    - **duration = pulseIn(sensor2EchoPin, HIGH);**: This function **pulseIn()** measures the duration for which the echo pin remains HIGH after sending the ultrasonic pulse. It captures this duration in microseconds.
4. **Calculating Distance:**

- **distance = duration * 0.034 / 2**: The duration of the echo (in microseconds) is converted to distance in centimeters. The speed of sound in air is around 343 meters per second or 0.0343 centimeters per microsecond (approximated to 0.034 in the formula). Since the pulse travels to the object and back, the division by 2 accounts for the round trip.

Figure 7.18: Prototype, Prototype (Side-view), Prototype (Top-view)

CHAPTER 8: SYSTEM EVALUATION

8.1 Testing Techniques

The usability of the system is tested by running the apk file on the user's Android device. The successful installation of the mobile application is indicating the testing is a success. In terms of the forecasting module, the performance is tested by using performance matrix such as the Mean Squared Error & Root Mean Squared Error. If the values for both errors are less than 20%, then the forecasting module is adequate. However, it must be noted that in an ideal situation, a percentage of less than 5% is the most desirable.

8.2 Error Handling

For the forecasting web application, the error handling can be conducted according to the status code given by the error message, this can aid in further troubleshooting.

8.3 System Strengths & Limitations

The system architecture has several advantages:

- It is scalable and can be easily expanded to accommodate more sensor devices and more data.
- It is cloud-based, which means that it can be accessed from anywhere with an internet connection.
- It is open source, which means that it is free to use and modify.
- It uses machine learning algorithms to forecast waste accumulation levels, which can help to improve the efficiency and effectiveness of waste management operations.

Overall, the system architecture is a well-designed and comprehensive solution for waste management. It is scalable, cloud-based, open source, and uses machine learning

algorithms to forecast waste accumulation levels. The system can help stakeholders to make informed decisions, plan proactively, and implement timely interventions to optimize waste management practices and promote environmental preservation.

In terms of limitations, this project is limited by the amount of storage allocated by Streamlit Cloud due to its free-tier service. Future considerations can include using platforms such as AWS or Azure to host the forecasting model with more storage & processing power allocated. Another limitation to mention is the waste level sensor, since there is only one sensor facing directly downwards, so if the trash is concentrated on the sides, then the level sensed will probably be less accurate. In response to that, future implementations can include more sensors detecting the waste scattered all over the lid of the bin. From this various measurements, the average waste levels can be taken & account for any inaccuracy.

CHAPTER 9: CONCLUSION

To conclude, the analytical module for this waste collection system has been developed accordingly & was able to hit all the objectives & requirements set for this waste collection system. The system was able to use historical data to make predictions about the waste production rates which aids stakeholders in decision making. In addition, the interactive analytical dashboard fulfils the objective of visualizing the data efficiently to the stakeholders. Finally, the analytical module was successfully implemented into a mobile application which enables users to access the system on-the-go. All of these serves as a proof-of-concept that will enable stakeholders to make evidence-based decisions and develop sustainable waste management strategies tailored to the specific needs of Malaysia. By optimizing waste collection, resource allocation, and planning processes, the project is able to contribute to effective waste management practices and promote environmental preservation in the country.

REFERENCES

Azarmi, S. L., Akeem Adeyemi Oladipo, Vaziri, R., & Alipour, H. (2018). *Comparative Modelling and Artificial Neural Network Inspired Prediction of Waste Generation Rates of Hospitality Industry: The Case of North Cyprus*. *10*(9), 2965–2965. https://doi.org/10.3390/su10092965

Kulisz, M., & Kujawska, J. (2020). Prediction of Municipal Waste Generation in Poland Using Neural Network Modeling. *Sustainability*, *12*(23), 10088. https://doi.org/10.3390/su122310088

Nasir, N., Faridah Zulkipli, Nor, Nurfarahin Mohamad Ghadafy, & Nur Hazieqah Azman. (2021). Forecasting solid waste generation in Negeri Sembilan and Melaka - UKM Journal Article Repository. *Journalarticle.ukm.my*. http://journalarticle.ukm.my/17831/1/jqma-17-1-paper5.pdf

O. Mudannayake, D. Rathnayake, J. D. Herath, D. K. Fernando and M. Fernando, "Exploring Machine Learning and Deep Learning Approaches for Multi-Step Forecasting in Municipal Solid Waste Generation," in IEEE Access, vol. 10, pp. 122570-122585, 2022, doi: 10.1109/ACCESS.2022.3221941.

Prabhakaran, S. (2021, August 22). *ARIMA Model*. Machinelearningplus.com. https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/

Solano Meza, J. K., Orjuela Yepes, D., Rodrigo-Ilarri, J., & Cassiraga, E. (2019). Predictive analysis of urban waste generation for the city of Bogotá, Colombia, through the implementation of decision trees-based machine learning, support vector machines and artificial neural networks. *Heliyon*, *5*(11), e02810. https://doi.org/10.1016/j.heliyon.2019.e02810

World Bank. (2018). *The World Bank Annual Report 2018*. World Bank. https://documents.worldbank.org/en/publication/documents-reports/documentdetail/630671538158537244/the-world-bank-annual-report-2018