# AI Based Smart Waste System for Organizations

MUHAMMAD CHARAGH HAIDER YOUSAF KHAN
**01-134221-044**

KASHAF AMEEN
**01-134221-030**

## Bachelor of Science in Computer Science

Supervisor: Ms. Afrah Naeem

Department of Computer Science
Bahria University, Islamabad

December 2025

# Certificate

We accept the work contained in the report titled "AI Based Smart Waste System for Organizations," written by Muhammad Charagh Haider Yousaf Khan and Kashaf Ameen, as meeting the standards for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by:

Supervisor: Ms. Afrah (Lecturer)

______________________________

Internal Examiner:

______________________________

External Examiner:

______________________________

Project Coordinator: Dr. Asim Ali (Sr. Assistant Professor)

______________________________

Head of the Department: Dr. Muhammad Khurram Ehsan (Sr. Associate Professor)

______________________________

December, 2025

# Abstract

The AI-Based Smart Waste System for Organizations (AISWO) is a waste management tool created to meet the demands of contemporary organizations, such, as corporate offices, colleges, medical centers and public institutions. AISWO utilizes Artificial Intelligence (AI) utilizing the Internet of Things (IoT) to track waste containers live forecast their fill levels and alert users when collection becomes necessary.

The system utilizes bins fitted with weight Microcontrollers that send real-time data to the cloud. AI analysis handles this information to predict the time each bin will become completely filled allowing for anticipatory waste handling.

Companies frequently encounter issues, like bins being overfilled, pickups and unwarranted disposal of bins that are only partly filled. Such problems may result in sanitation concerns, health risks, and operational inefficiencies, while also increasing labor costs and environmental waste.

AISWO addresses these challenges by providing a centralized dashboard for facility managers to monitor bin statuses, optimize collection schedules, assign staff efficiently, and receive instant notifications. Staff members can engage with the platform, via web applications and a chatbot, enabling them to check bin conditions, report issues, and receive timely updates.

In developing countries like Pakistan, numerous organizations do not possess waste management systems infrastructure. AISWO provides a economical option. By enhancing collections management, optimizing timetables and enhancing oversight the system can lower labor expenses. Avoid overflows, Uphold tidier, safer work environments. Research shows that smart waste management systems have the ability to decrease inefficiencies by 30 to 40 percent while enhancing the work environment hygiene and sustainability.

# Acknowledgment

In the name of Allah, the most beneficent and the most merciful, who enabled us to take note of this fantastic opportunity to expand our knowledge and provided us with the devotion and wisdom to work and plan with great passion. We owe all our academic work and struggles to our dear parents and respected teachers; without them, we are nothing. We would like to convey our heartfelt gratitude to our project supervisor "Ms. Afrah Naeem", for their helpful assistance, comments, and recommendations during the project. Lastly, we would like to commend our friends who make study challenges and hard times full of excitement.

Muhammad Charagh Haider Yousaf Khan
Kashaf Ameen
Bahria University
Islamabad, Pakistan

December 2025

*“Technology, like art, is a soaring exercise of the human imagination”*

— Daniel Bell

# Contents

# List of Figures

# List of Tables

# Acronyms and Abbreviations

| | |
|---|---|
| API | Application Programming Interface |
| API Key | Authentication Token for External API Services |
| CRUD | Create, Read, Update, Delete |
| CSV | Comma-Separated Values |
| CV | Computer Vision |
| DB | Database |
| ESP32 | Wi-Fi and Bluetooth Enabled Microcontroller |
| ETA | Estimated Time of Arrival |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Secure Hypertext Transfer Protocol |
| JSON | JavaScript Object Notation |
| JWT | JSON Web Token |
| KPI | Key Performance Indicator |
| LLM | Large Language Model |
| ML | Machine Learning |
| NoSQL | Non-Relational Database Model |
| NTP | Network Time Protocol |
| PWM | Pulse Width Modulation |
| REST | Representational State Transfer |
| RTDB | Firebase Realtime Database |
| SDK | Software Development Kit |
| SMS | Short Message Service |
| SPA | Single Page Application (React) |
| TS | Time Series |
| UI | User Interface |
| UI/UX | User Interface / User Experience |
| URL | Uniform Resource Locator |
| UX | User Experience |
| Wi-Fi | Wireless Fidelity |

# Chapter 1

# Introduction

The fast urbanization and population growth has contributed a lot to the increase in the quantity of waste produced in an organizational and industrial setting which imposes more burden on the already available collection and disposal systems [1]. Still in most places of work, waste management is done in the traditional way that involves manual check up and predetermined collection schedules. When these processes are not matched with the real rate of waste production bins will either be underutilized or overfilled before they are emptied. This leaves organizations with hygiene problems that reoccur, delays in operations, and wastage of man power and transport. In order to surpass these limitations, there is an increasing trend at moving towards smart systems that comprise of sensing, connectivity, and automated decision-making. This means that such systems can facilitate more efficient and proactive handling of waste by constantly tracking bin status and facilitating action based on data [2]. In this context, the AI-Based Smart Waste System for Organizations (AISWO) will be suggested as a specific solution in the organizational setting of academic institutions, corporate offices, and healthcare facilities. AISWO uses real-time data gathering, AI-driven prediction and human-friendly interfaces to enhance the timing of waste collection, minimize manual labor and promote cleaner and more efficient working conditions.

AISWO integrates smart bins powered by IoT, cloud-based analytics, and fore-casting based on AI. They have the weight sensor in each bin which is connected to ESP32 microcontroller which constantly sends the fill-level data to a real-time cloud database (Firebase). The system uses machine learning algorithms to forecast the time the bins will be close to the capacity thresholds to allow the collection staff to respond before the bins fill up. A web-based dashboard allow administrators and workers to have a role-based access to monitor the live position of the bins and generate analytical report and an alert when bins are filled past the pre-defined limit. Also, a chatbot assistant will be included to enable the interaction and enable the employees to track the status of the bins, provide feedback, or report to the problem via natural language questions.

In addition to surveillance, AISWO serves as a decision-support tool by analyzing historical waste data to identify trends by location, day, and time. This enables optimized collection planning, improved staff allocation, and targeted placement of bins or awareness initiatives, transforming waste management into an efficient and data-driven process.

## 1.1 Background

The old system of waste management within organizations like hospitals, universities, factories and corporate offices has been more of a reactive system whereby waste is collected manually and through predetermined collection schedules. Trip routes and time plans of staff or janitorial teams usually visit bins regardless of whether or not they are full or empty. This strategy usually causes overfilling of bins in places of heavy usage and unnecessary visits to half-filled bins causing unproductive utilization of resources and poor hygiene.

Recent research on smart waste systems demonstrates that smart waste systems the combination of IoT sensors, communication and control platforms have the potential to enhance the accuracy of monitoring, decrease the occurrence of overflow, and enable the collection vehicles to route more effectively [3]. The IoT bins will be able to constantly send fill-level data to a cloud-based system where it can be stored, processed, and visualized. Over this data, smart algorithms may be used to approximate the time left to the fullness of a bin, or to prioritize on where to be served first. These systems prove the idea that the integration of sensing and analytics can minimise the number of manual interventions, enhance the quality of services, and facilitate more environmentally-friendly waste management.

In addition to real-time surveillance, AI and computational intelligence methods have been used to improve the predictive and optimization features of waste management systems. As an illustration, hybrid optimization techniques which include Genetic Algorithm and Fuzzy Inference Systems (GA-FIS) have been investigated in demand forecasting, scheduling and decision making activities involving waste collection [4]. These techniques aim to represent the unpredictable and dynamic members of the waste generation with the system being able to adjust to different usage profiles in different locations and time periods. Using these methods to process the data of IoT-based waste will allow the organizations to shift to the process of predictive and preventive controls.

The AISWO system is an expansion of such advancements, integrating real time monitoring and automatic notification into a single platform that is specific to an organizational environment. With the integration of IoT-based sensing with cloud storage and a central dashboard, AISWO will help administrators and personnel to see real-time bin reports, monitor fill rates, and react to possible overflow scenarios in advance. This combined approach will allow AISWO to offer a solution to waste management in organizations that will be data driven, connected, and user friendly to achieve increased operational efficiency, hygiene, and sustainability.

## 1.2 Problem Description

Although certain facilities may have access to simple forms of digital tools, most organizations rely on the static waste collection schedules that fail to indicate the real state of the waste bins. Mostly, collection activities will be conducted at fixed interval whether the bins were full or were almost empty. This causes premature and delayed collections. Early collections cause them to make unnecessary trips, use more fuel, and spend more on labor and late collections make their bins overflow and become health hazards, cause poor smell, and create poor perception to the employees, students, patients and visitors.

The lack of a centralized monitoring system also makes it more difficult to manage the waste in large organizations where the bins can be located in a large number of floors, departments, or buildings. The supervisors do not have a common platform where they can see the status of bins and therefore, they find it challenging to give priority to the locations during peak hours or address complaints effectively. In addition, a majority of the conventional systems fail to utilize past data in the determination of pattern of increase in waste during events, office hours or visiting hours in hospitals. Consequently, there is a lack of proactive waste management that is widely reliant on manual reports by individuals.

These difficulties are magnified in high theoretical settings like universities and hospitals where there is a great difference in waste production during the day. Devoid of automation and real time monitoring, the staff can be either underserved or overworked, and the collection routes might be poorly laid out. AISWO solves these problems by combining sensor based monitoring, cloud data storage, and notifications to make sure that the bins are picked at the appropriate time. The system offers a centralized cloud based dashboard to have real time visibility and is able to make sound decision making using historical data trends. The AISWO enhance cleaner facilities, better use of resources and uniform hygiene standards throughout the organization by making it less reliant on manual dependency and more responsive.

## 1.3 Objectives

The primary goal of the AI Based Smart Waste System for Organizations is to develop a smart and scalable framework for efficient waste monitoring and management using sensor based and cloud enabled technologies. The specific objectives of the project are as follows:

- **Real time waste monitoring**

  Deploy IoT based smart bins equipped with weight and ultrasonic sensors and ESP32 microcontrollers to continuously measure and transmit waste level data to a cloud database.

- **Cloud based data management**

  Store real time and historical bin data in a centralized cloud platform to enable easy access, monitoring, and record keeping.

- **Automated notifications**

  Send alerts to assigned collection staff when bins reach predefined thresholds to ensure timely collection and prevent overflow.

- **Role based dashboards**

  Provide separate web interfaces for administrators and employees allowing administrators to view all bins and employees to view only assigned bins.

- **Chatbot integration**

  Enable users to interact with the system using a chatbot for checking bin status viewing assigned bins and reporting issues through simple text commands.

- **Data visualization and reporting**

  Display bin levels and historical records in graphical format to help administrators observe usage trends and improve collection planning.

Collectively, these objectives ensure the development of a connected and efficient waste management system that improves operational response reduces manual monitoring and enhances cleanliness within organizational environments. The system is designed to support future upgrades such as intelligent forecasting or route optimization without requiring major architectural changes.

## 1.4 Project Scope

AISWO is designed to be adaptable across diverse organizational environments, particularly in facilities where multiple buildings and user groups share common waste infrastructure. In corporate offices, the system supports hygiene by monitoring departmental waste levels and identifying areas with unusually high or low usage. In hospitals and clinics, timely disposal of general and non-critical waste helps maintain infection control and reduces contamination risks in waiting areas, corridors, and staff zones.

Educational institutions represent a key application area where smart waste systems can enhance both operational efficiency and environmental awareness. Campus-wide deployments of IoT-based waste monitoring have demonstrated that combining sensor data with machine learning can improve waste collection and resource planning in university environments [5]. AISWO aligns with this approach by enabling administrators to monitor distributed bins, analyze waste patterns across departments, and integrate insights into sustainability initiatives.

The scalable and modular design of AISWO allows deployment in both small organizations with limited bins and large campuses with multiple buildings. Core components such as monitoring, notifications, and dashboards can be customized according to infrastructure and operational needs. Smaller offices may prioritize basic monitoring and alerts, while larger facilities can integrate advanced forecasting models and role-specific dashboards.

Future enhancements include GPS-based smart routing for optimized collection paths, waste-type classification using computer vision, and carbon-emission tracking to assess environmental impact. Additional extensions such as a mobile application, AI-based forecasting, and gamification features to encourage eco-friendly behavior can further strengthen AISWO as a comprehensive and sustainable waste management solution.

## 1.5 Solution Application Areas

- **Corporate and Institutional Facilities**

  Department and Floor to enable centralised monitoring to bins which helps facility managers to maintain workplace clean and optimize bins pickup time.

- **Hospitals and Healthcare Centers**

  Helps to contain general and non-infectious waste more efficiently, making overflowing bins in critical areas (such as corridors, waiting rooms and staff room) less likely.

- **Educational Campuses**

  Allows to observe the full bins in the campus, also it can schedule the collection rounds, and can be incorporated into sustainability initiatives.

In conclusion, AISWO provides an integrated framework for intelligent waste management using AI and IoT technologies. By combining real-time monitoring, predictive analytics, and user interactivity, the system offers a practical solution to modern organizational waste challenges while contributing to cleaner and more sustainable environments.

# Chapter 2

# Literature Review

The recent focus on environmental sustainability along with the rising amount of waste produced by organizations have boosted the pace of innovations in the sphere of smart waste management. The entry of the Internet of Things (IoT) has allowed the organizations to leave the traditional method of waste management, designing systems that can track, foresee and optimize the process of waste collection with minimal human effort. These inventions have streamlined waste management to be more efficient, environmental friendly and economical. The AI-Based Smart Waste System to Organizations (AISWO) is an extension of these innovations as it offers a single platform, over which smart bins, cloud-based data visualization, and operational dashboards connect, and thus, improves the decision-making process and the overall level of waste-handling efficiency.

The chapter is a review of the main developments and technological trends in smart waste management. It considers the development of IoT-based waste systems, discusses the issues related to their realization, and indicates the opportunities of the data integration to enhance the performance of waste management operations. Besides, it determines the current gaps in research and presents the role of AISWO in enhancing intelligent waste management. The discussion highlights solutions that are specifically applicable to large organizational setups, including universities, hospitals, industrial facilities, and corporate campuses, where the distributed sources of waste and multi-layered activities cannot be handled with simplistic means of information, such as the traditional sensor-based alerts.

## 2.1 Overview

The classical waste collection systems may be based on the fixed schedules and manual monitoring. This causes inefficiencies like slowness in pickups, full bins and unnecessary use of fuel. IoT-based systems can be used to address these challenges by filling in the bin fill level and making an automatic collection decision based on sensor data in real-time. Researcher highlighted that IoT sensors and AI-powered analytics can be used to control the waste proactively due to the ability to predict the level of filled and minimize the frequency of collections, thus increasing the efficiency of operations and their environmental friendliness [6]. Their results formed a basis to create intelligent waste systems such as AISWO that applies predictive algorithm to control waste in a dynamic manner.

Smart waste management solutions have evolved over the years to complex multi-layered systems that integrate sensing, communication, data storage, analytics and user interfaces as opposed to basic threshold-driven alert systems. The initial systems primarily discussed here included the ability to receive a notification when a bin was full and forward this to a central server. Newer techniques include the analysis of historical data, merging with geographic information systems and setting up business rules which are easily configurable enabling operators to set responses to various locations or bin types. This development is seen as a transition between operational monitoring only and the strategic decision support where the garbage data is considered an important resource in the case of planning and policy making in the organizations.

## 2.2 Existing Systems and Technological Background

Various researchers have suggested IoT-based models that will track the amount of waste with the help of sensors and forward the information to a centralized server via a wireless network. Such systems usually use ultrasonic, infrared, or weight sensors to check the level of fills. Nevertheless, most of them do not have predictive intelligence or scalability. To achieve reliability in waste-handling, scientist developed an IoT-based model that allows automation of data collection with the help of microcontrollers and cloud connectivity [7]. Their research showed that the combination of sensors and communication to the cloud can greatly decrease the number of manual labor and make organizations manage waste in real time. AISWO builds on this idea by including user-friendliness on the interface of the administrator and the employees. Technologically speaking, the layered architecture is similar to most smart waste systems. Sensors of perception layer include parameters like fill level, weight, temperature, or even gas concentration. The network layer is based on such technologies as Wi-Fi, LoRaWAN, cellular networks, or NB-IoT to send the readings to a backend. The application layer will most commonly be cloud databases, analytics engines, and dashboards or web apps. The design decisions at every level affect the system performance: low-power wide-area communication can be used to increase battery life but restrict bandwidth whereas high-resolution sensing can be used to predict better at the cost of higher energy use and data volume. The awareness of these trade-offs is essential to the creation of systems that are reliable and maintainable at an organizational scale.

## 2.3 Challenges in Current Waste Management Systems

Although there is technological advancement, the majority of the current waste management systems have a number of shortcomings. Most of them are made to be implemented small-scale and are incapable of large, distributed networks like campuses or industrial estates. The widespread adoption is often hampered by network instability, loss of data and complexity in system maintenance. In addition, there are no real-time communication systems that allow scheduling collection in advance. AISWO eliminates these challenges through the provision of a single cloud platform, which combines the round-the-clock monitoring and multi-user accessibility in the form of dashboards to ensure efficient cooperation and quick reactions.

Interoperability is also another challenge. Sensors, smart bins, and platforms of other vendors typically have proprietary protocols or data formats, and it is hard to make them work in one management system. This has fragmentation that may compel organizations to have different tools in regard to various buildings or waste streams, which raise overheads in operations. Security and confidentiality are also significant issues, particularly when a system retrieves the information of a healthcare or a company setting. To be adopted in the long-run, it is therefore crucial to make sure that it is secure in communication, access control, and auditability.

## 2.4 IoT and AI Integration in Smart Waste Management

The amalgamation of IoT and AI will be the main theme of the future generation of waste management systems. IoT sensors help amass real-time data, whereas AI algorithms help chatbot make predictions based on past and current data streams. This integration does not only best optimize the waste collection routes but also minimizes the environmental impact. Scientist emphasized that predictive modeling, together with the integration of IoT, can be used to estimate the trend of waste generation with high accuracy, guarantee the optimal route planning, and reduce the operational costs to a minimum [4]. AISWO applies this technique by using the feature of weight sensors which can enable organizations to take action before the overflow has taken place.

## 2.5 Smart Monitoring

The smart dashboards are used to offer real-time information on waste levels, trends, and performance indicators, enabling the administrators to make informed decisions. This incorporation makes collection of waste effective as well as eco-friendly. The analysis of the long-term trends could show the departments or periods of time when the generation of the waste is higher than usual, which means that the processes could be optimized or the arrangement of the bins could be changed. Detection of anomalies can be used to alert about abnormal sensor behavior or spikes in the amount of waste, which can be an indication of abuse, or a fault in the equipment.

## 2.6 Research Gaps and AISWO's Contribution

The current smart waste solutions are usually not scalable, holistically monitored, and cannot interactively communicate. Most of them pay attention to only the simplest monitoring and do not combine data visualization and interaction with users into one system. AISWO fills these weaknesses by combining the IoT and cloud computing into a single solution. Its dashboards enable easy human system interaction. The innovations through AISWO offer a holistic and scalable solution of intelligent waste management that adds to cleaner, smarter and more sustainable organizational environments.

Moreover, much previous literature also tends to consider the municipal or city-level-deployments, and there is relatively lesser discussion of intra-organizational context where there is the division of roles and responsibilities among the facility managers, departmental coordinators as well as cleaning employees. AISWO is directly aimed at this situation, providing access to individual bins with fine-grained visibility and role-based access and communication tools in line with organizational workflows. float

## 2.7 Existing Examples / Solutions

In this part, a number of available smart waste management solutions are examined and compared to the offered AISWO system.

Table 2.1: Comparison of Existing Smart Waste Solutions with AISWO

| **Application Name** | **Key Features** | **Weakness / Limitations** | **AISWO Advantage** |
|---|---|---|---|
| Enevo Smart Waste Platform [8] | • Wireless fill-level sensors<br>• Cloud-based analytics<br>• Machine-learning forecasting | • Targeted mainly at cities and haulers<br>• Focus on fleet logistics<br>• Limited role-based access | • Designed for organizational departments<br>• Role-based access for admins and employees<br>• Weight-sensor monitoring |
| Bigbelly Smart Waste System [9] | • Solar-powered smart bins<br>• Compaction technology<br>• Real-time fullness alerts | • High deployment cost<br>• Requires physical installation of smart bins | • Uses existing waste bins<br>• Lower implementation cost |
| Ecube Labs CleanCity Platform [10] | • IoT sensors<br>• Collection route optimization<br>• Data analytics dashboard | • Focuses mainly on municipalities<br>• Limited customization for internal organizations | • Customizable to departmental needs<br>• Integrated employee access system |

# Chapter 3

# Requirement Specifications

This chapter defines the functional requirements, non-functional requirements, the user interactions, and system behaviors that will be important in the successful implementation of the AI-Based Smart Waste System for Organizations (AISWO). Being an IoT-based smart waste monitoring platform, AISWO combines smart bins on ESP32, ultrasonic and weight sensors, Firebase cloud storage, a Node.js server, and a React-based dashboard. These requirement specifications would be used as a baseline blueprint of the system architecture and would be used to guide development of hardware components, backend services, cloud infrastructure, and user interface features. AISWO is expected to make a shift in the traditional waste handling and turn it into a responsive, automated and data-driven workflow of operation. The system tracks the amount of trash left in the bins in real time, notifies operators, forecasts the amount of trash that will be left in the future based on previous data, and also allows administrators to see the analytics via dashboards. The system also includes a chatbot interface which enables employees to query bin status, report problems, and otherwise interact with the system in a natural manner. The dashboard also shows the weather of the current day with an inbuilt Weather API, which assists the operators and administrators predict the environmental conditions that could be affecting the waste generation and collection trends. The specifications make sure that AISWO fulfills the requirement of performance, reliability, and usability of large organizations including universities, hospitals, and corporate campuses.

## 3.1 Functional Requirements

The fundamental feature of AISWO is based on real-time sensor awareness, intelligent bin control, predictive analytics, operator alerts, and intelligent user interface. Func- tional requirements describe how the system would act, the interaction of the hardware software and the tasks that the system needed.

Table 3.1: Functional Requirements of AISWO

| S. No | Functional Requirement | Type | Status |
|---|---|---|---|
| 1. | User (Admin) shall be able to register an account in AISWO and Admin is capable to register new employee. | Core | Done |
| 2. | User shall be able to log in securely using Firebase Authentication and access features according to assigned role. | Core | Done |
| 3. | System shall support smart bins equipped with ESP32 microcontrollers, weight sensors, and ultrasonic sensors for real-time fill-level detection | Core | Done |
| 4. | System shall continuously transmit sensor readings from ESP32 devices to Firebase Realtime Database or backend APIs for processing. | Core | Done |
| 5. | AI module shall use historical waste data to display graph trents on bin dashboard to predict future fill levels | Core | Done |
| 6. | System shall retrieve current weather information from a Weather API and display it on the dashboard. | Intermediate | Done |
| 7. | System shall send automated alerts (email and push notifications) when bin fill levels exceed predefined thresholds or when predicted overflow is imminent. | Core | Done |
| 8. | Dashboard shall display live bin status, fill percentages, historical trends, weather information, and alert history for administrators and operators. | Core | Done |
| 9. | Users shall be able to interact with a chatbot to query bin status, report issues, and request assistance using natural language. | Intermediate | Done |
| 10. | Administrator shall be able to create and manage operator accounts and assign specific bins or locations to each operator. | Core | Done |
| 11. | System shall store and synchronize sensor data, predictions, operator assignments, and notification logs across all connected clients. | Core | Done |

## 3.2 Non-Functional Requirements

To achieve high performance, scalability and reliability, AISWO complies with the following non-functional requirements:

Table 3.2: Non-Functional Requirements of AISWO

| S. No | Non-Functional Requirement | Category | Status |
|---|---|---|---|
| 1. | Dashboard shall reflect updated bin readings with minimal latency between sensor transmission and user display. | Performance | Done |
| 2. | ESP32 devices shall maintain reliable wireless connectivity and automatically recover from temporary network failures. | Reliability | Done |
| 3. | Firebase and Node.js backend shall scale to support an increasing number of bins, locations, and concurrent users without significant performance degradation. | Scalability | Done |
| 4. | All communication between devices, backend, and clients shall use secure protocols (HTTPS/TLS) and enforce role-based access control. | Security | Done |
| 5. | Web dashboard and chatbot interfaces shall be responsive and usable on both desktop and mobile browsers with clear navigation and feedback. | Usability | Done |
| 6. | Database shall support concurrent reads and writes while preserving data consistency across all clients. | Reliability | Done |
| 7. | System components (backend services, sensor firmware, frontend modules) shall follow modular design to support maintenance, feature extension, and updates. | Maintainability | Done |

## 3.3 Use Case Analysis

Use cases describe the primary interactions between users and the AISWO system. The system users include administrators, operators, employees, IoT devices, and automated backend services.

### 3.3.1 Primary Use Cases

- **User Registration and Login:** Enables users to register and log in with appropriate access roles.
- **Bin Registration and Configuration:** Administrators configure new bins, assign capacity values, and map sensors.
- **Real-Time Monitoring:** Sensor data from ESP32 devices is continuously uploaded and displayed on the dashboard.
- **Sensor Data Processing:** The backend processes ultrasonic and weight readings to compute fill percentages.
- **AI-Based Fill Prediction:** The AI module predicts waste accumulation trends and time-to-full.
- **Weather Information Display:** The system retrieves real-time weather data through the Weather API and shows it on the dashboard for operator awareness.
- **Alerts and Notifications:** Operators receive threshold-based notifications via email and FCM.
- **Chatbot Interaction:** Users interact with the chatbot for information retrieval or issue reporting.
- **Operator Assignment:** Administrators manage operator profiles and assign bins.
- **View Reports and Analytics:** Users access reports on bin history, alerts, predictions, weather context, and performance insights.

### 3.3.2 Use Case Diagram

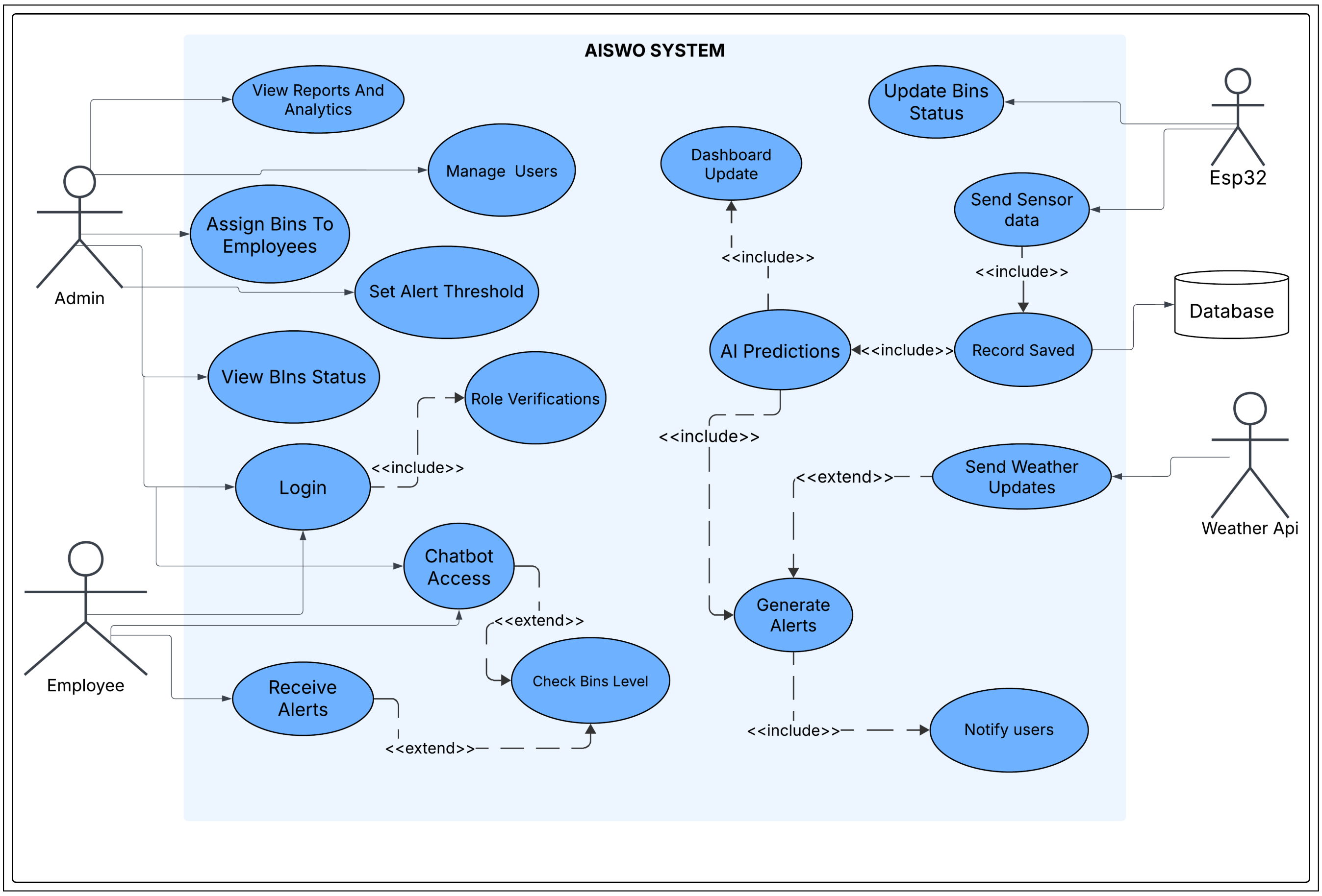



Figure 3.1: Main Use Case Diagram for AISWO

### 3.3.3 Use Case Tables

Table 3.3: Textual Description of Use Case UC-01 for User Login

| | | |
|---|---|---|
| **Use Case ID** | UC-01 | |
| **Use Case Name** | User Login | |
| **Actors** | Admin, Employee | |
| **Description** | This use case allows a registered user to log in to AISWO and access the system according to their role. | |
| **Trigger** | User enters credentials and clicks on the "Login" button. | |
| **Preconditions** | • User is registered in the system.<br>• Application and internet connection are available. | |
| **Postconditions** | • User is authenticated.<br>• User is redirected to the corresponding role-based dashboard. | |
| **Normal Flow** | **User** | **System** |
| | 1. Opens AISWO login page.<br>3. Enters email and password.<br>5. Clicks "Login". | 2. Displays login form.<br>4. Validates input format.<br>6. Authenticates credentials with Firebase.<br>7. Loads appropriate dashboard based on user role. |
| **Alternative Flows** | • User clicks on "Forgot Password" to reset credentials. | |
| **Exceptions** | • Invalid email or password entered.<br>• User account disabled or not verified.<br>• Network error while authenticating. | |
| **Assumptions** | • User remembers registered email and password. | |

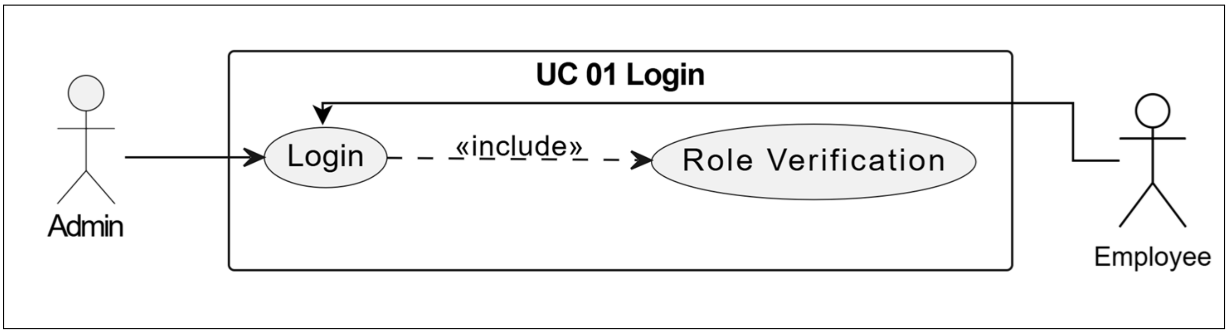



Figure 3.2: Use Case Diagram for User Login

The User Login use case enables secure and authenticated access to the AISWO system for registered users. Users are required to enter valid credentials, which are verified through Firebase Authentication to ensure data security and identity validation. Upon successful authentication, the system determines the user's role and redirects them to the appropriate role-based dashboard. This process ensures controlled access, protects system resources, and provides a personalized and efficient user experience based on assigned permissions.

Table 3.4: Textual Description of Use Case UC-02 for Chatbot Access

<table>
<tr><td>Use Case ID</td><td>UC-02</td></tr>
<tr><td>Use Case Name</td><td>Chatbot Access</td></tr>
<tr><td>Actors</td><td>Employee, Admin</td></tr>
<tr><td>Description</td><td>This use case allows an employee and Admin to query bin status or report waste-related issues using a chatbot interface.</td></tr>
<tr><td>Trigger</td><td>User opens chatbot window and submits a message.</td></tr>
<tr><td>Preconditions</td><td><ul><li>User is logged into AISWO.</li><li>Chatbot service and AI API are available.</li></ul></td></tr>
<tr><td>Postconditions</td><td><ul><li>User receives a response to the query.</li><li>Any reported issue is stored for follow-up.</li></ul></td></tr>
<tr><td>Normal Flow</td><td><table>
<tr><th>User</th><th>System</th></tr>
<tr><td>1. Opens chatbot from dashboard.</td><td>2. Displays chat interface.</td></tr>
<tr><td>3. Types question (e.g., bin status, report issue).</td><td>4. Processes query using Gemini API and internal data.</td></tr>
<tr><td>5. Sends message.</td><td>6. Returns appropriate response and displays it in chat.</td></tr>
<tr><td></td><td>7. Logs issue if user reports a problem.</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><ul><li>Chatbot suggests contacting an collector for complex issues.</li></ul></td></tr>
<tr><td>Exceptions</td><td><ul><li>AI service unavailable or times out.</li><li>User enters unsupported query type.</li></ul></td></tr>
<tr><td>Assumptions</td><td><ul><li>Chatbot model is trained/configured for AISWO domain.</li></ul></td></tr>
</table>

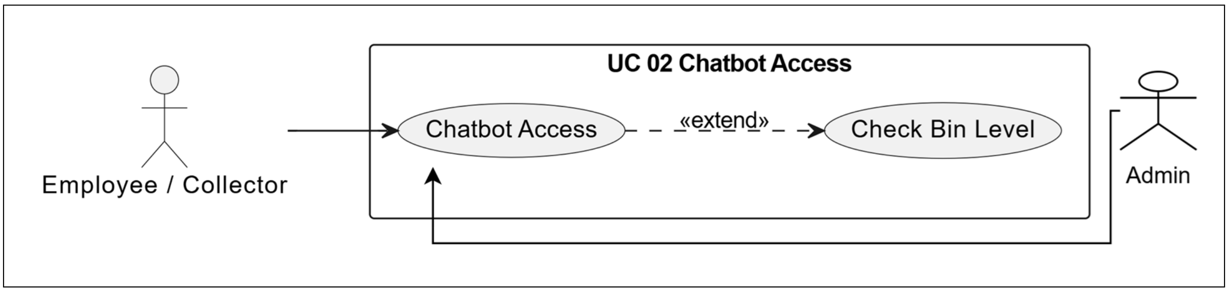



Figure 3.3: Use Case Diagram for ChatBot Access

The Chatbot Access use case enables users to communicate with the AISWO system through an interactive interface to obtain real-time information and assistance. Users can ask queries related to smart bin status, waste level predictions, weather conditions, or general system instructions. The chatbot retrieves relevant data from the system database, analyzes the request, and provides accurate and immediate responses. This functionality helps users make informed decisions, minimizes response time, and reduces dependency on manual system navigation. By offering automated support and instant access to critical information, the chatbot significantly enhances system efficiency, usability, and overall user satisfaction.

Table 3.5: Textual Description of Use Case UC-03 for View Reports and Analytics

| | |
|---|---|
| **Use Case ID** | UC-03 |
| **Use Case Name** | View Reports and Analytics |
| **Actors** | Admin |
| **Description** | This use case allows users to view historical bin readings, alerts, AI predictions, and weather context through charts and reports. |
| **Trigger** | User selects the “Reports/Analytics” section from the dashboard. |
| **Preconditions** | • System has stored historical sensor and notification data. |
| **Postconditions** | • Selected analytics data is displayed.<br>• User may export reports if required. |
| **Normal Flow** | **User** / **System**<br>1. Opens analytics section. / 2. Retrieves aggregated historical data.<br>3. Selects filters (date range, location, bin). / 4. Applies filters and generates charts/tables.<br>5. Reviews trends, predictions, and alert history. / 6. Provides option to export or download report. |
| **Alternative Flows** | • User switches between different visualization types (bar, line, table). |
| **Exceptions** | • No data available for selected filters. |
| **Assumptions** | • Historical data has been collected for a sufficient period. |

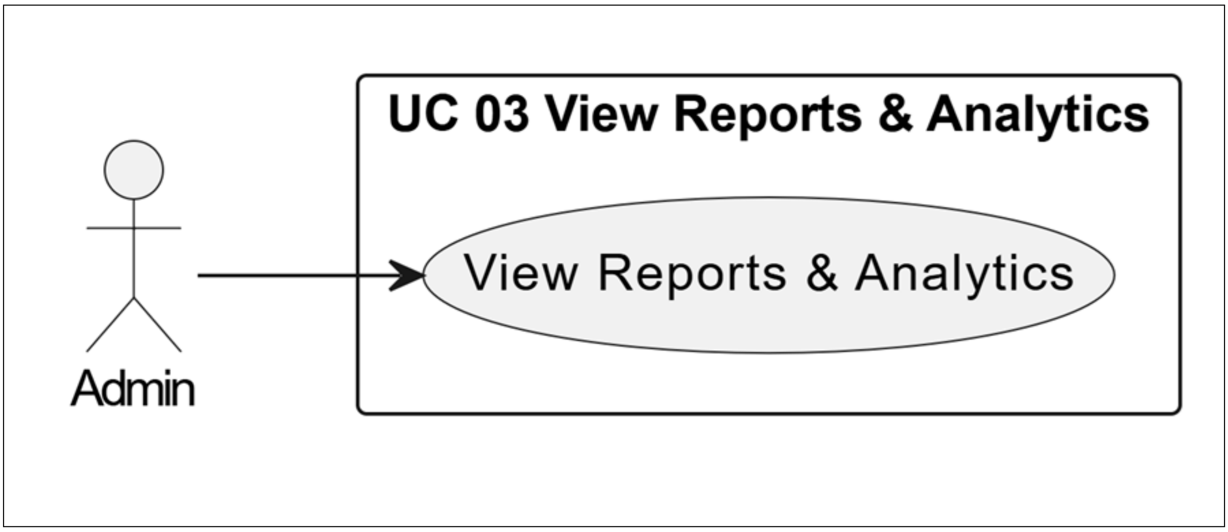



Figure 3.4: Use Case Diagram for View Reports And Analytics

The Reports and Analytics use case delivers comprehensive visual summaries and analytical insights related to the performance of the AISWO system. It presents detailed information on bin fill level trends, waste collection patterns, prediction accuracy, operator performance, and system alert statistics through charts, graphs, and dashboards. These analytical tools enable administrators and decision-makers to evaluate system efficiency, identify operational issues, and monitor overall performance. By supporting data-driven decision-making, this use case assists in optimizing resource allocation, improving operational strategies, and facilitating effective long-term planning for smart waste management.

Table 3.6: Textual Description of Use Case UC-04 for Assign Bins to Employees

<table>
<tr><td>Use Case ID</td><td colspan="2">UC-04</td></tr>
<tr><td>Use Case Name</td><td colspan="2">Assign Bins to Employees</td></tr>
<tr><td>Actors</td><td colspan="2">Admin, Employee</td></tr>
<tr><td>Description</td><td colspan="2">This use case describes how the Admin assigns specific waste bins to employees for monitoring and maintenance. The system ensures that each bin is linked to a responsible employee, enabling accountability and efficient operations.</td></tr>
<tr><td>Trigger</td><td colspan="2">Admin decides to allocate or reallocate bins to employees.</td></tr>
<tr><td>Preconditions</td><td colspan="2">• Employees are registered in the system.<br>• Bins are available and identifiable in the system.</td></tr>
<tr><td>Postconditions</td><td colspan="2">• Assigned bins are linked to employees in the database.<br>• Employees receive notifications about their assigned bins.</td></tr>
<tr><td rowspan="2">Normal Flow</td><td>Admin</td><td>System</td></tr>
<tr><td>1. Logs into the AISWO SYSTEM.<br>3. Selects an employee from the list.<br>5. Assigns one or more bins to the employee.</td><td>2. Displays list of employees and bins.<br>4. Shows available bins for assignment.<br>6. Updates database with assignment details.<br>7. Sends notification to the employee.</td></tr>
<tr><td>Alternative Flows</td><td colspan="2">• Admin reassigns bins if an employee is unavailable.<br>• Admin filters bins by location, type, or status before assignment.</td></tr>
<tr><td>Exceptions</td><td colspan="2">• Employee not found in the system.<br>• Bin already assigned or unavailable.<br>• Database update failure.</td></tr>
<tr><td>Assumptions</td><td colspan="2">• Admin has the necessary permissions to manage assignments.<br>• System maintains accurate records of bins and employees.</td></tr>
</table>

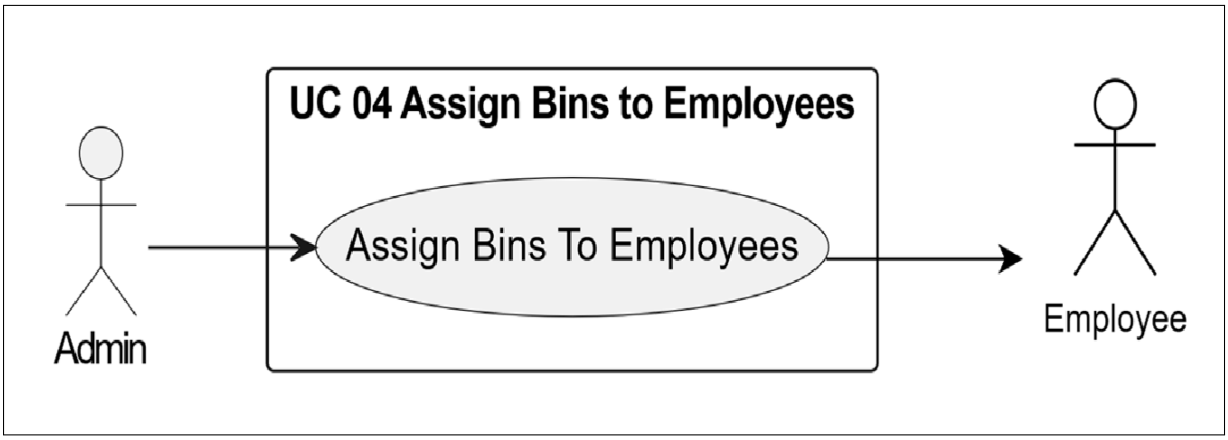



Figure 3.5: Use Case Diagram for Assign Bins To Employees

The Assign Bins to Employees use case allows the Admin to allocate specific waste bins to individual employees for monitoring and maintenance. After logging into the AISWO SYSTEM, the Admin selects an employee from the user list and assigns one or more bins to that employee. The system records the assignment in the database and ensures that each bin is linked to a responsible employee. Once the assignment is complete, the employee receives a notification about their designated bins. This process ensures accountability, efficient tracking, and proper distribution of workload, while also allowing the Admin to reassign bins if an employee becomes unavailable.

Table 3.7: Textual Description of Use Case UC-05 for Set Alert Threshold

<table>
<tr><td>Use Case ID</td><td colspan="2">UC-05</td></tr>
<tr><td>Use Case Name</td><td colspan="2">Set Alert Threshold</td></tr>
<tr><td>Actors</td><td colspan="2">Admin</td></tr>
<tr><td>Description</td><td colspan="2">This use case describes how the Admin configures threshold levels for bin fill status to trigger alerts. The system uses these thresholds to notify employees when bins approach or exceed critical levels.</td></tr>
<tr><td>Trigger</td><td colspan="2">Admin initiates threshold configuration for one or more bins.</td></tr>
<tr><td>Preconditions</td><td colspan="2">• Admin is logged into the AISWO SYSTEM.<br>• Bins are registered and available in the system.</td></tr>
<tr><td>Postconditions</td><td colspan="2">• Threshold values are stored in the system.<br>• Alerts are triggered based on configured thresholds.</td></tr>
<tr><td rowspan="2">Normal Flow</td><td>Admin</td><td>System</td></tr>
<tr><td>1. Logs into the AISWO SYSTEM.<br>3. Selects a bin to configure.<br>5. Enters new threshold value.</td><td>2. Displays list of bins and current thresholds.<br>4. Prompts for threshold input.<br>6. Validates and saves the threshold.<br>7. Applies threshold to alert logic.</td></tr>
<tr><td>Alternative Flows</td><td colspan="2">• Admin updates thresholds for multiple bins in batch mode.<br>• Admin resets thresholds to default values.</td></tr>
<tr><td>Exceptions</td><td colspan="2">• Invalid threshold value entered.<br>• Bin not found or inaccessible.<br>• Database update failure.</td></tr>
<tr><td>Assumptions</td><td colspan="2">• Admin has permission to configure thresholds.<br>• System supports real-time alerting based on thresholds.</td></tr>
</table>

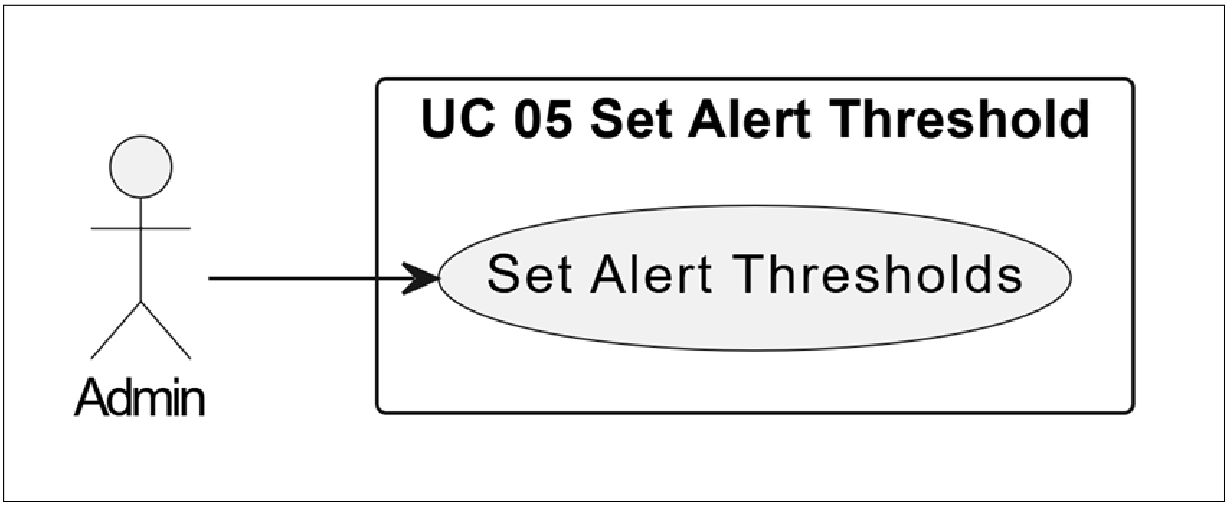



Figure 3.6: Use Case Diagram for Set Alert Threshold

The Set Alert Threshold use case enables the Admin to configure specific fill-level thresholds for waste bins within the AISWO SYSTEM. These thresholds determine when alerts should be triggered, helping ensure timely maintenance and preventing overflow. After logging into the system, the Admin selects one or more bins and inputs the desired threshold values. The system validates and stores these settings, integrating them into its alert logic. Once configured, the system automatically monitors bin levels and notifies relevant users when thresholds are exceeded. This functionality supports proactive waste management and improves operational efficiency.

Table 3.8: Textual Description of Use Case UC-06 for View Bins Status

<table>
<tr><td>Use Case ID</td><td>UC-06</td></tr>
<tr><td>Use Case Name</td><td>View Bins Status</td></tr>
<tr><td>Actors</td><td>Admin</td></tr>
<tr><td>Description</td><td>This use case describes how the Admin accesses real-time status information of all registered bins in the AISWO SYSTEM. The system displays current fill levels, location, and alert status to support monitoring and decision-making.</td></tr>
<tr><td>Trigger</td><td>Admin requests bins status overview from the dashboard.</td></tr>
<tr><td>Preconditions</td><td>• Admin is logged into the AISWO SYSTEM.<br>• Bins are registered and actively reporting data.</td></tr>
<tr><td>Postconditions</td><td>• Bins status data is displayed to the Admin.<br>• Alerts are visible for bins exceeding threshold levels.</td></tr>
<tr><td>Normal Flow</td><td><table><tr><td>Admin</td><td>System</td></tr><tr><td>1. Logs into the AISWO SYSTEM.</td><td>2. Retrieves latest bins data.</td></tr><tr><td>3. Navigates to bins status dashboard.</td><td>4. Displays fill level, location, and alert status.</td></tr><tr><td></td><td>5. Highlights bins exceeding alert thresholds.</td></tr></table></td></tr>
<tr><td>Alternative Flows</td><td>• Admin filters bins by location, type, or alert status.<br>• Admin exports bins status data for reporting.</td></tr>
<tr><td>Exceptions</td><td>• Bins data unavailable due to sensor failure.<br>• System unable to retrieve real-time data.</td></tr>
<tr><td>Assumptions</td><td>• Sensors are functioning and transmitting data.<br>• Admin has access rights to view bins status.</td></tr>
</table>

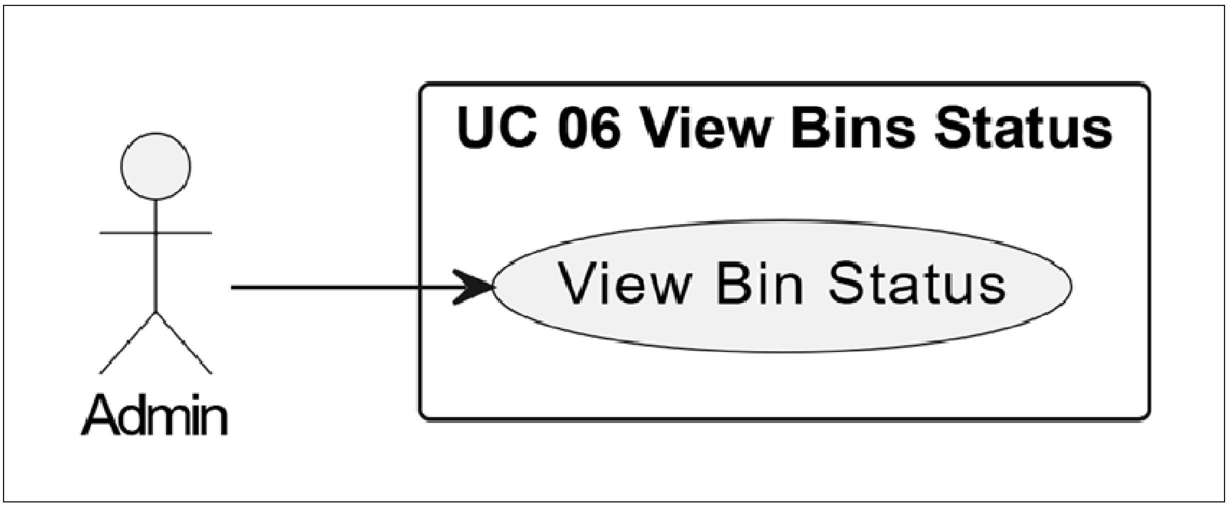



Figure 3.7: Use Case Diagram for View Bins Status

The View Bins Status use case allows the Admin to monitor the current condition of all bins registered in the AISWO SYSTEM. Upon logging in, the Admin accesses a dashboard that displays real-time data including fill levels, bin locations, and alert statuses. The system highlights bins that have exceeded configured thresholds, enabling timely intervention. Admins can also filter the view by location or bin type and export the data for reporting purposes. This functionality supports efficient waste management and ensures that critical bins are addressed promptly.

Table 3.9: Textual Description of Use Case UC-07 for Manage Users

<table>
<tr><td>Use Case ID</td><td colspan="2">UC-07</td></tr>
<tr><td>Use Case Name</td><td colspan="2">Manage Users</td></tr>
<tr><td>Actors</td><td colspan="2">Admin, Employee</td></tr>
<tr><td>Description</td><td colspan="2">This use case describes how the Admin manages user accounts within the AISWO SYSTEM. Admin can create, update, or deactivate user profiles. Upon changes, the system may notify Employees or update their profile information accordingly.</td></tr>
<tr><td>Trigger</td><td colspan="2">Admin initiates user management actions such as adding, editing, or disabling a user.</td></tr>
<tr><td>Preconditions</td><td colspan="2">• Admin is authenticated and authorized to manage users.<br>• Employee accounts exist or are pending creation.</td></tr>
<tr><td>Postconditions</td><td colspan="2">• User profiles are updated in the system.<br>• Employees receive notifications or see updated profile details.</td></tr>
<tr><td rowspan="5">Normal Flow</td><td>Admin</td><td>System</td></tr>
<tr><td>1. Logs into the AISWO SYSTEM.</td><td>2. Displays user management dashboard.</td></tr>
<tr><td>3. Selects a user to manage.</td><td>4. Prompts for user details or action.</td></tr>
<tr><td>5. Updates or deactivates user profile.</td><td>6. Saves changes and sends notification.</td></tr>
<tr><td></td><td>7. Updates Employee profile view if applicable.</td></tr>
<tr><td>Alternative Flows</td><td colspan="2">• Admin bulk edits multiple user accounts.<br>• Admin restores a previously deactivated user.</td></tr>
<tr><td>Exceptions</td><td colspan="2">• Invalid user data entered.<br>• Employee not found in the system.<br>• Database update failure.</td></tr>
<tr><td>Assumptions</td><td colspan="2">• Admin has full access to user management features.<br>• System maintains accurate and secure user records.</td></tr>
</table>

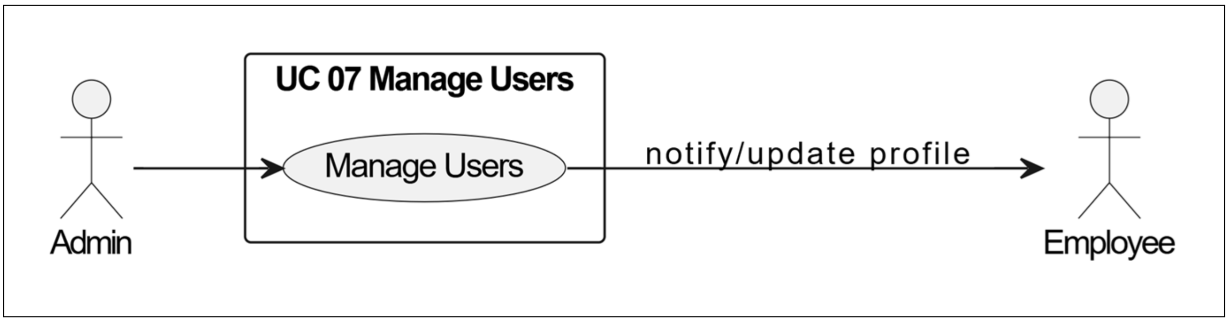



Figure 3.8: Use Case Diagram for Manage Users

The Manage Users use case enables the Admin to control user accounts within the AISWO SYSTEM. Admins can create new profiles, update existing user details, or deactivate accounts as needed. When changes are made, the system may notify Employees or reflect updates in their profile view. This ensures that user records remain accurate and that Employees are informed of any relevant changes. The functionality supports secure and efficient user administration, contributing to overall system integrity and role-based access control.

Table 3.10: Textual Description of Use Case UC-08 for Receive Alerts

<table>
<tr><td>Use Case ID</td><td colspan="2">UC-08</td></tr>
<tr><td>Use Case Name</td><td colspan="2">Receive Alerts</td></tr>
<tr><td>Actors</td><td colspan="2">Employee / Collector</td></tr>
<tr><td>Description</td><td colspan="2">This use case describes how the Employee or Collector receives system-generated alerts when bins approach or exceed configured fill-level thresholds. The alert may prompt the user to check the bin level for further action.</td></tr>
<tr><td>Trigger</td><td colspan="2">Bin fill level crosses the defined alert threshold.</td></tr>
<tr><td>Preconditions</td><td colspan="2">• Alert thresholds are configured by Admin.<br>• Bins are actively reporting fill-level data.</td></tr>
<tr><td>Postconditions</td><td colspan="2">• Employee receives alert notification.<br>• Employee may check bin level for confirmation or action.</td></tr>
<tr><td rowspan="3">Normal Flow</td><td>System</td><td>Employee / Collector</td></tr>
<tr><td>1. Monitors bin fill levels continuously.</td><td>2. Receives alert notification.</td></tr>
<tr><td>3. Detects threshold breach.</td><td>4. Opens alert details.<br>5. Chooses to check bin level.</td></tr>
<tr><td>Alternative Flows</td><td colspan="2">• Employee filters alerts by location or urgency.<br>• Employee forwards alert to another team member.</td></tr>
<tr><td>Exceptions</td><td colspan="2">• Alert not delivered due to network failure.<br>• Bin data inaccurate or delayed.</td></tr>
<tr><td>Assumptions</td><td colspan="2">• Employees have access to alert dashboard or mobile notifications.<br>• System is configured to extend alert flow to bin-level checking.</td></tr>
</table>

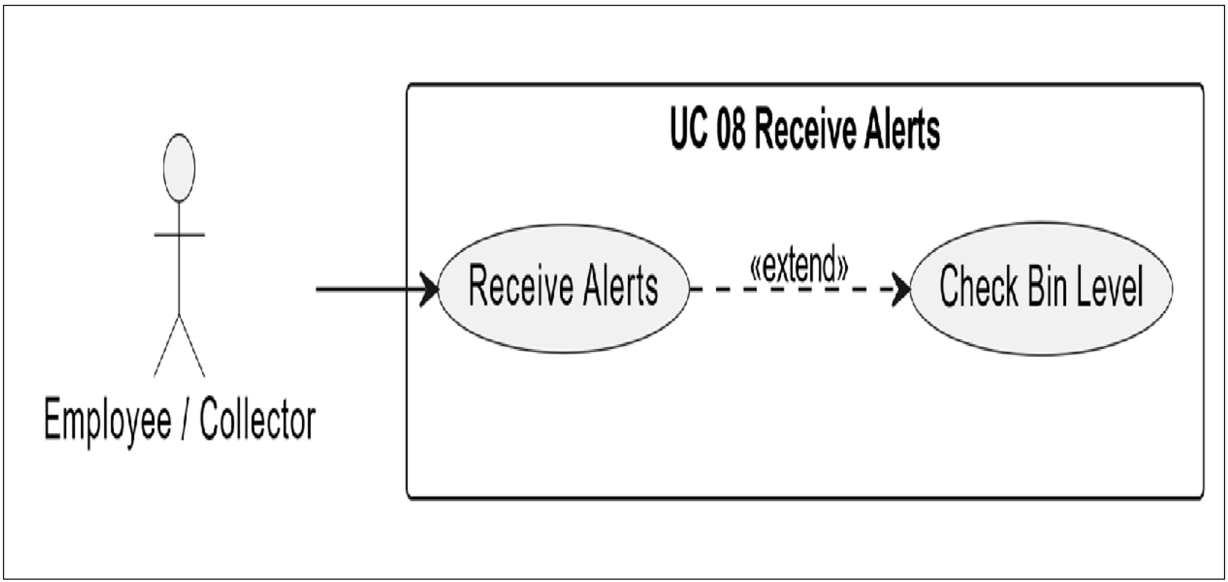



Figure 3.9: Use Case Diagram for Receive Alerts

The Receive Alerts use case explains how the Employee or Collector interacts with the AISWO SYSTEM to stay informed about critical bin conditions. When a bin's fill level crosses a predefined threshold configured earlier by the Admin, the system automatically generates an alert. This alert is delivered to the Employee or Collector through the dashboard or mobile notification system, ensuring timely awareness of bins that require attention. Upon receiving the alert, the user may choose to view additional details such as the bin's location, type, and urgency level. The use case includes an «extend» relationship to Check Bin Level, allowing the user to verify the bin's current status before taking action. This extension ensures that alerts are not only received but also contextualized with real-time data, supporting informed decision-making. Employees may also filter alerts by location or severity, forward them to other team members, or prioritize bins based on operational needs. This functionality plays a vital role in proactive waste management, minimizing overflow incidents, and optimizing collection routes. By integrating alert logic with bin-level monitoring, the system empowers field personnel to respond efficiently and maintain service quality.

Table 3.11: Textual Description of Use Case UC-09 for Update Bins Status

<table>
<tr><td>Use Case ID</td><td colspan="2">UC-09</td></tr>
<tr><td>Use Case Name</td><td colspan="2">Update Bins Status</td></tr>
<tr><td>Actors</td><td colspan="2">ESP32</td></tr>
<tr><td>Description</td><td colspan="2">This use case describes how the ESP32 device updates the status of bins in the AISWO SYSTEM by transmitting sensor data that reflects current bin conditions. The system uses this data to refresh bin records and support real-time monitoring.</td></tr>
<tr><td>Trigger</td><td colspan="2">ESP32 detects a change in bin condition or completes a scheduled sensing cycle.</td></tr>
<tr><td>Preconditions</td><td colspan="2"><ul><li>ESP32 is connected and authenticated with the AISWO SYSTEM.</li><li>Sensors are properly installed and calibrated.</li></ul></td></tr>
<tr><td>Postconditions</td><td colspan="2"><ul><li>Bin status is updated in the system database.</li><li>The updated status is visible to Admins and Employees via the dashboard.</li></ul></td></tr>
<tr><td rowspan="2">Normal Flow</td><td>ESP32</td><td>System</td></tr>
<tr><td>1. Reads sensor data from the bin.<br>3. Packages and transmits the data.</td><td>2. Receives and parses the data packet.<br>3a. Updates bin status in the database.<br>4. Refreshes dashboard with latest bin status.</td></tr>
<tr><td>Alternative Flows</td><td colspan="2"><ul><li>ESP32 sends batched updates for multiple bins.</li><li>System applies AI predictions to enrich bin status.</li></ul></td></tr>
<tr><td>Exceptions</td><td colspan="2"><ul><li>Sensor failure or invalid data.</li><li>Network transmission error.</li><li>Database update failure.</li></ul></td></tr>
<tr><td>Assumptions</td><td colspan="2"><ul><li>ESP32 has stable connectivity and sufficient power.</li><li>System supports real-time bin status updates.</li></ul></td></tr>
</table>

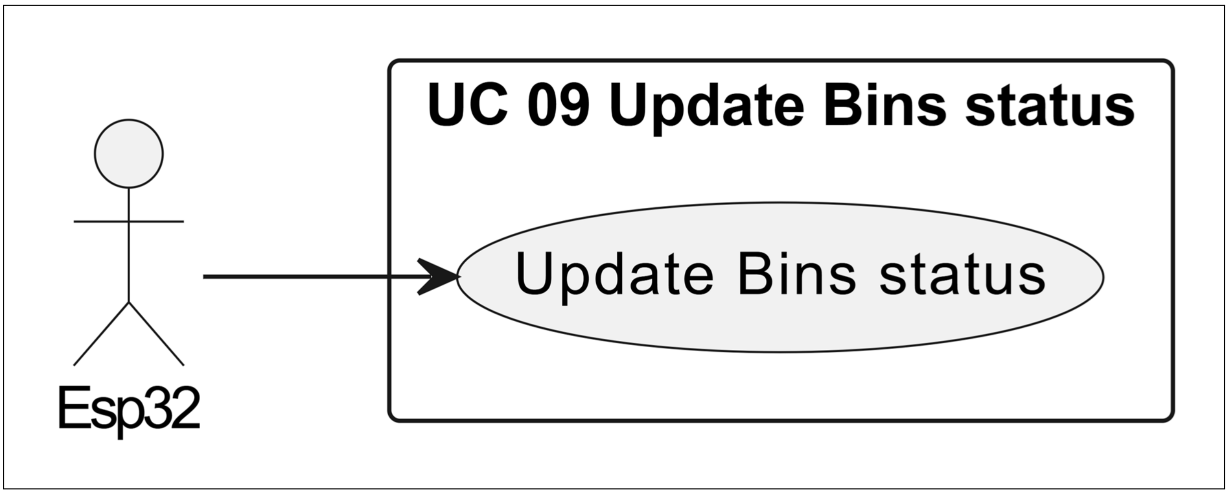



Figure 3.10: Use Case Diagram for Update Bins Status

The Update Bins Status use case defines how the ESP32 device maintains accurate and timely bin condition records within the AISWO SYSTEM. The ESP32 periodically reads sensor data from each bin, capturing metrics such as fill level, temperature, or other environmental indicators. Once collected, the device transmits this data to the system's backend, where it is parsed and used to update the bin's status in the central database. This updated information is then reflected on the dashboard, allowing Admins and Employees to monitor bin conditions in real time. The system may also apply AI predictions to enrich the status with future projections or anomaly detection. In some cases, ESP32 may batch updates for multiple bins to optimize transmission. This use case ensures operational visibility, supports timely waste collection decisions, and contributes to the overall efficiency of smart waste management.

Table 3.12: Textual Description of Use Case UC-10 for AI Prediction Based Upon Sensor Data

<table>
<tr><td>Use Case ID</td><td colspan="2">UC-10</td></tr>
<tr><td>Use Case Name</td><td colspan="2">AI Prediction Based Upon Sensor Data</td></tr>
<tr><td>Actors</td><td colspan="2">ESP32</td></tr>
<tr><td>Description</td><td colspan="2">This use case describes how the AISWO SYSTEM utilizes sensor data sent by ESP32 to perform AI-based predictions. These predictions support decision-making by generating alerts, updating dashboards, and notifying users based on real-time and historical bin conditions.</td></tr>
<tr><td>Trigger</td><td colspan="2">ESP32 sends sensor data to the system.</td></tr>
<tr><td>Preconditions</td><td colspan="2"><ul><li>ESP32 is actively transmitting sensor data.</li><li>System has access to historical bin records.</li><li>AI prediction models are trained and deployed.</li></ul></td></tr>
<tr><td>Postconditions</td><td colspan="2"><ul><li>Sensor data is saved and processed.</li><li>AI predictions are generated and acted upon.</li><li>Dashboards are updated and users are notified.</li></ul></td></tr>
<tr><td rowspan="2">Normal Flow</td><td>ESP32</td><td>System</td></tr>
<tr><td>1. Sends sensor data.</td><td>2. Saves data to bin records.<br>3. Applies AI prediction models.<br>4a. Updates dashboard with predicted insights.<br>4b. Generates alerts based on prediction.<br>5. Notifies users if alert conditions are met.</td></tr>
<tr><td>Alternative Flows</td><td colspan="2"><ul><li>System enriches predictions using external data sources.</li><li>Alerts are prioritized based on prediction confidence.</li></ul></td></tr>
<tr><td>Exceptions</td><td colspan="2"><ul><li>Sensor data is incomplete or corrupted.</li><li>AI model fails to generate prediction.</li><li>Notification delivery fails.</li></ul></td></tr>
<tr><td>Assumptions</td><td colspan="2"><ul><li>ESP32 provides consistent and timely data.</li><li>AI models are regularly updated and validated.</li><li>System supports automated alerting and dashboard refresh.</li></ul></td></tr>
</table>

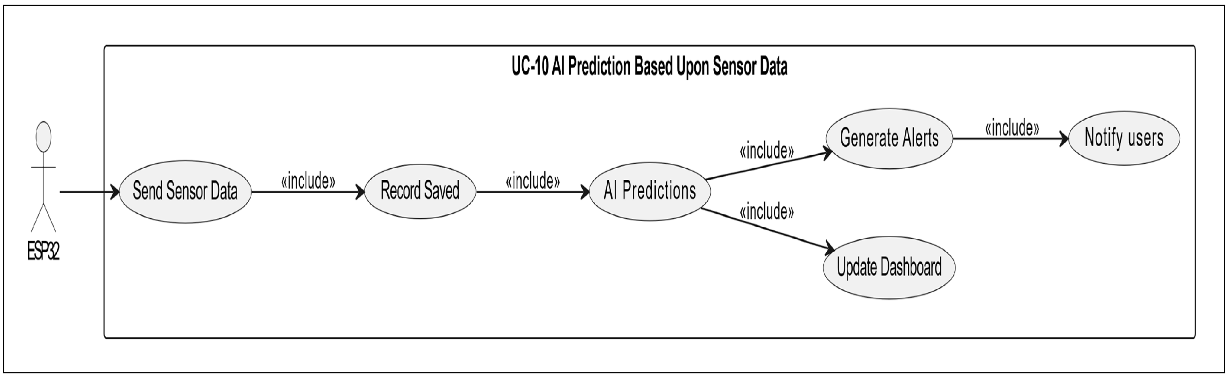



Figure 3.11: Use Case Diagram for AI Prediction Based Upon Sensor Data

The AI Prediction Based Upon Sensor Data use case illustrates how the AISWO SYSTEM transforms raw sensor input from ESP32 into actionable intelligence. When ESP32 sends sensor data, the system first saves it into structured bin records. These records are then processed by AI models trained to detect patterns, forecast fill levels, and identify anomalies. Based on these predictions, the system may update the dashboard with visual insights, generate alerts for bins nearing capacity or showing irregular behavior, and notify users such as Admins or Employees. This predictive capability enhances operational planning, reduces overflow incidents, and supports data-driven decision-making. In some cases, predictions may be enriched using external data sources, and alerts may be prioritized based on confidence scores. This use case demonstrates the integration of IoT sensing with AI analytics to create a responsive and intelligent waste management system.

Table 3.13: Textual Description of Use Case UC-11 for Weather Based Notification

| | | |
|---|---|---|
| **Use Case ID** | UC-11 | |
| **Use Case Name** | Weather Based Notification | |
| **Actors** | WeatherAPI, Employee / Collector | |
| **Description** | This use case describes how the AISWO SYSTEM receives weather updates from an external WeatherAPI and uses this data to generate alerts that are relevant to bin conditions or operational planning. These alerts are then delivered to Employees or Collectors to support timely and weather-aware decision-making. | |
| **Trigger** | WeatherAPI sends updated weather data to the system. | |
| **Preconditions** | • WeatherAPI is integrated and authorized to send updates.<br>• System is configured to interpret weather data for alert generation. | |
| **Postconditions** | • Weather data is received and processed.<br>• Alerts are generated based on weather conditions.<br>• Employees or Collectors are notified of relevant alerts. | |
| **Normal Flow** | **WeatherAPI** | **System and Employee / Collector** |
| | 1. Sends weather update to system. | 2. Parses and evaluates weather data.<br>3. Generates alerts based on weather impact.<br>4. Notifies Employee / Collector with alert details.<br>5. Employee reviews alert and takes action if needed. |
| **Alternative Flows** | • Alerts are filtered by location or severity before delivery.<br>• Employee forwards alert to another team member. | |
| **Exceptions** | • WeatherAPI fails to deliver update.<br>• Weather data is malformed or incomplete.<br>• Notification fails to reach Employee / Collector. | |
| **Assumptions** | • WeatherAPI provides timely and accurate data.<br>• System is capable of mapping weather data to operational alerts.<br>• Employees have access to alert dashboard or mobile notifications. | |

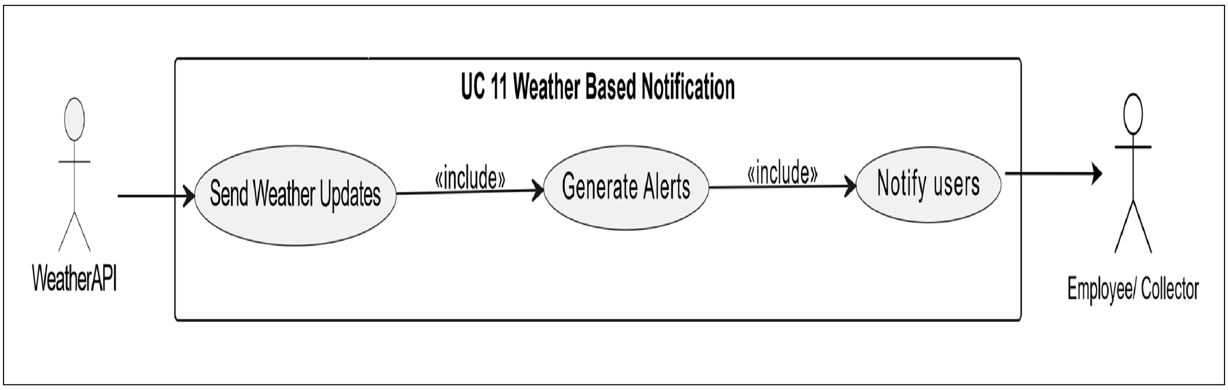



Figure 3.12: Use Case Diagram for Weather Based Notification

The Weather Based Notification use case outlines how the AISWO SYSTEM integrates external weather data to enhance operational awareness and responsiveness. When the WeatherAPI sends updates, the system parses the incoming data and evaluates its impact on bin conditions, collection schedules, or field operations. Based on this evaluation, the system generates alerts that may indicate weather-related risks such as flooding, heat stress, or collection delays. These alerts are then delivered to Employees or Collectors through the notification system, enabling them to take proactive measures. The use case includes an «include» relationship from Send Weather Updates to Generate Alerts, and another from Generate Alerts to Notify users, ensuring a structured flow of information. Employees may review the alert details, filter them by urgency or location, and respond accordingly. This functionality supports smarter field coordination, reduces service disruptions, and ensures that waste management operations remain resilient under changing weather conditions.

# Chapter 4

# Design

The design phase breaks down the requirements of the AI-Based Smart Waste System for Organizational ( AISWO ) requirements into a thought-through and systematic blueprint to carry out. In this chapter, the software process model, the overall system architecture, component-level design, and the interaction workflow are outlined to make sure that AISWO is scalable, modular, and maintainable. The focus is on the creation of autonomous and reusable modules that are capable of operating the IoT sensors ingestion, real-time monitoring, predictive analytics, weather-sensitive notifications, and chatbot communication. Arts like architecture diagrams, workflow sequences, and Ui mockups are used to help the development team in the development of the system into a complete waste management platform.

## 4.1 Software Process Model

The AISWO development is based on the Incremental Software Development Model. The model is applicable in a project whereby the general requirements are clear at the outset, yet the system may be built and delivered in a series of functional segments. ASWSO has well-organized modules sensor ingestion, alert engine, prediction pipeline, operator manage- ment, weather integration and chatbot processing that would make it best to be implemented in staged steps.

The Incremental Model enables one to develop, test and validate each subsystem separately and incorporate them in the entire solution. Given AISWO concerns the IoT hardware, backend services, cloud infrastructure, and predictive analytics, this model provides the stability of the system growth and necessary system control. It also enables tweaking or refinements to subsequent increments without impacting on the already delivered components.

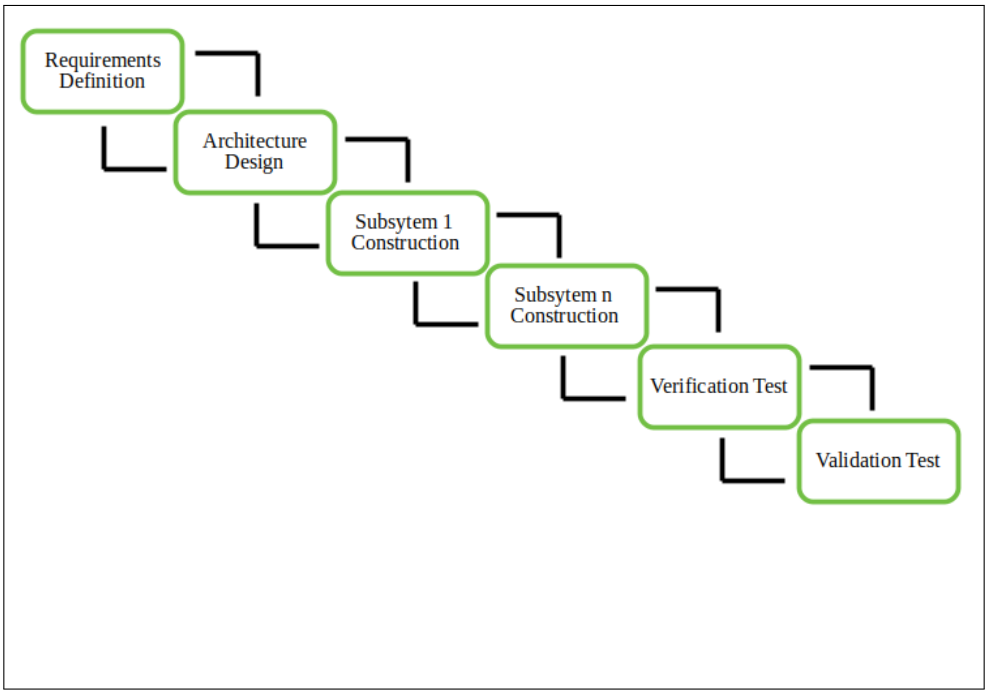



Figure 4.1: Incremental Software Development Model

## 4.2 Design Phase

The design stage would make sure that the architecture of AISWO meets functional and non-functional needs. This involves the definition of the structure of:

- IoT-based sensor data processing node based on Node.js.
- Web dashboard A React based administrator and operator dashboard.
- A Firebase real-time cloud database of data ingestion and synchronization.
- Waste accumulation AI prediction service.
- An interaction layer chatbot based on the API.
- Weather API to access and show the daily weather.

This step also incorporates the preparation of UI wireframes, backend route logic, API architecture, routes of data flow, and communication specifications that will guarantee a smooth integration of hardware devices, cloud services, and user interfaces.

## 4.3 System Architecture

AISWO is a multi-layer system that consists of IoT, back-end services, AI models, stored data and user interfaces. The architecture provides a clean separation between the data acquisition, data processing, prediction, and user interaction. Also, the system incorporates. a Weather API to show the weather conditions of the day on the dashboard, which would give the operators an environmental background that might cause the change in patterns of wasted or outdoor collection paths. The architecture consists of the following three layers:

- **Presentation Layer:** Being implemented with the help of React, this layer offers administrators, operators, and employees dashboards. It shows the level of bins, alert history, predictions, weather data, chatbot interface, and operator assignments. It sends messages to the services in the back-end server through secure REST APIs and Firebase SDK.

- **Application Layer:** This layer, which was built with the help of Node.js and Express, handles sensor data the ingestion, bin processing logic, the alert engine, weather API retrieval, operator management, and chatbot endpoints.

- **Data Layer:** Firebase Realtime Database and Firestore contains raw sensor data, processed data, history of bins, forecasts, operator information, weather, and user activity information to be used in further analytics.

## 4.4 AI-Based Prediction and Monitoring Pipeline

The prediction subsystem will supplement the monitoring process of waste generation through the analysis of the historical sensor data, which will be used to determine trends in the fill. This subsystem takes real time measurements of weight sensors and ultrasonic sensors and smooths them as needed and also incorporates prediction outcomes into the dashboard. The pipeline includes:

- **Input Interface:** Receives the data on the raw weight, ultrasonic, and timestamp of ESP32 devices.

- **Processing Layer:**Imposes calculations to calculate the percentage fill, anomaly detection, and it combines several sensor readings as long as both are present.

- **Alert Engine:** It uses the results of predictions to send notifications. The weather data can affect the severity of alerts in case of extreme weather which will raise the waste and limit the prompt collection.

- **Chatbot Module:** Retrieves bin information, forecasts, and summary of statuses on Firebase and uses it to answer questions within the context of the Gemini API.

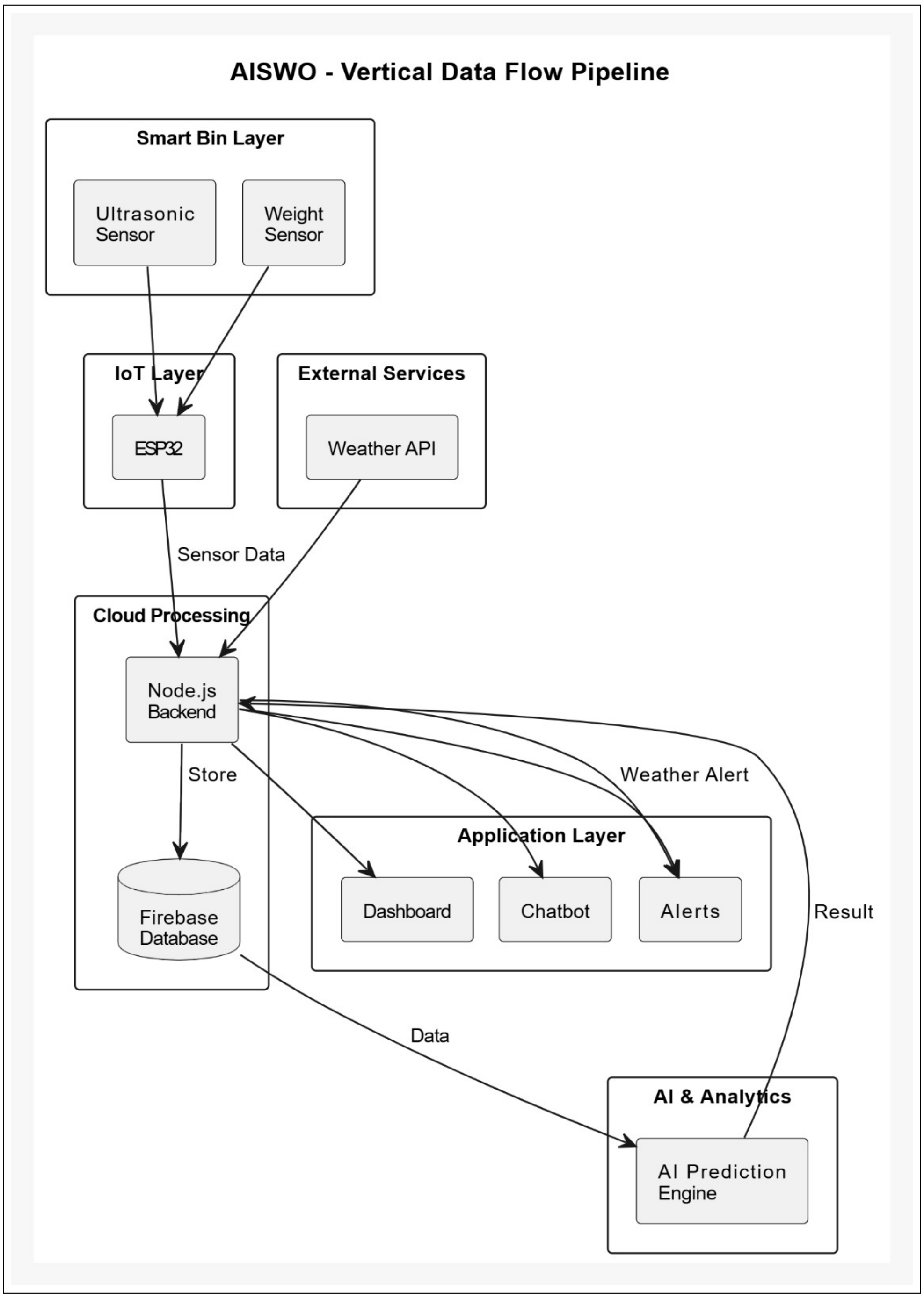



Figure 4.2: AI-Powered Monitoring and Prediction Pipeline

## 4.5 Sequence Diagram

The sequence diagrams are the time order interaction among AISWO components. The flow of the Real-Time Monitoring and Prediction is as shown in the following sequence:

1. ESP32 records the weight and ultrasonic measurements.
2. The sensor payload is transferred to Firebase through device.
3. Backend retrieves the data, works with it and changes bin status.
4. Weather API is consulted to obtain daily weather.
5. AI model is fed with new historical data and it produces predictions.
6. Dashboard gets up-to-date readings, forecasts and weather data.
7. User views real-time status through the React interface.

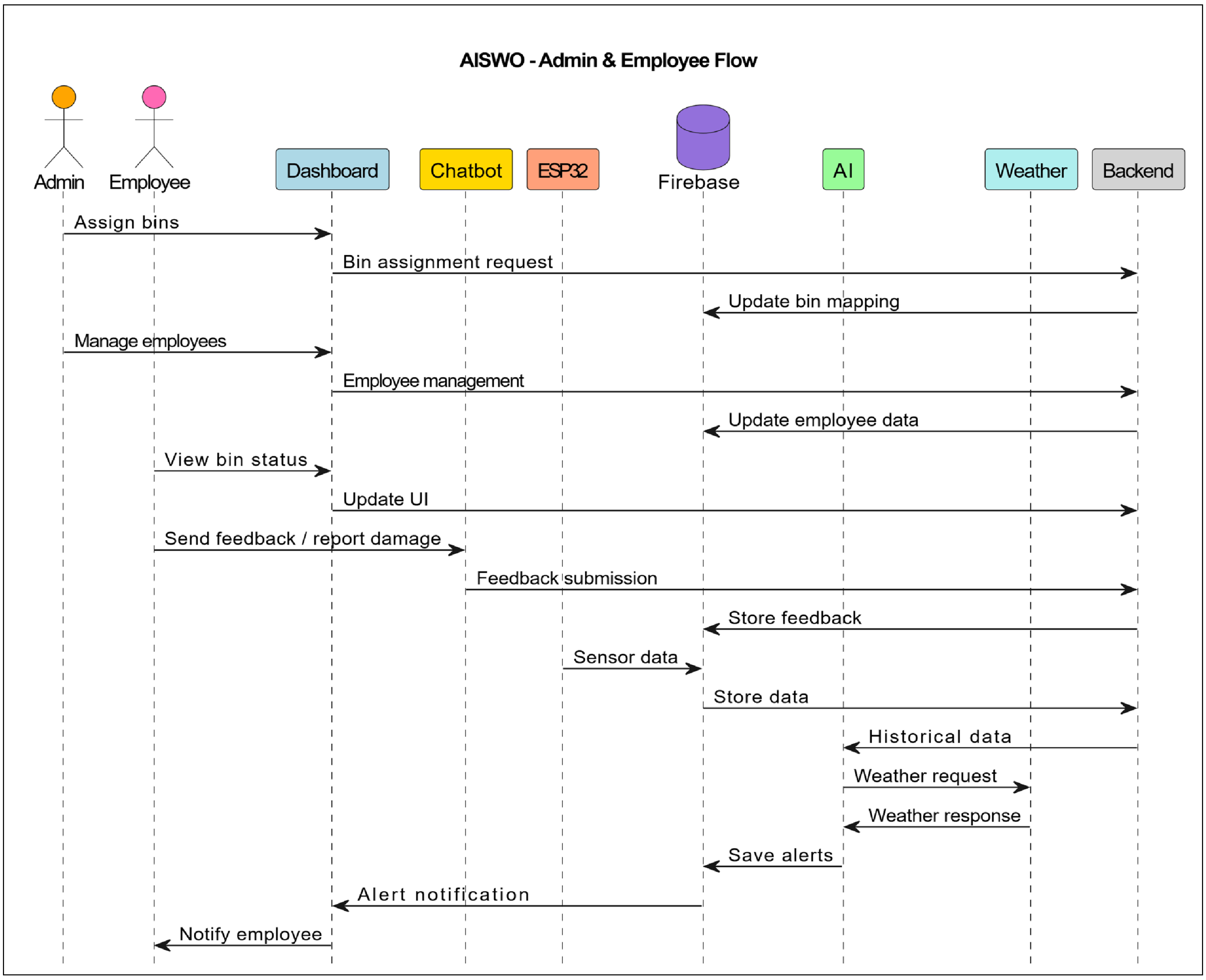



Figure 4.3: Sequence Diagram for AISWO Monitoring Process

## 4.6 Activity Diagram

The activity diagram provides the key workflow of AISWO, such as sensor processing, prediction, weather fetching, and dashboard visualization.

- **User Login:** The user, either Administrator or Operator, logs into the AISWO system with valid credentials. The system checks the login using Firebase and gives access to the dashboard based on role privileges.
- **Bin Status Retrieval:** After logging in, the dashboard gets the current status of all registered bins, including fill levels, battery status, and timestamps of the last updates. This helps the user quickly spot bins that need attention.
- **Sensor Data Upload:** Smart bins with weight or ultrasonic sensors send their readings to the AISWO backend through the ESP32 module. This happens regularly or when there's a significant change in a bin's fill level.
- **Data Processing:** The backend processes raw sensor readings, removes noise or faulty values, updates the database, and checks that the readings are within normal operating limits. Any issues are flagged for further inspection.
- **Weather API Fetch:** The system gets real-time weather information, such as temperature, rainfall, and humidity, from an external API. Weather data is relevant because environmental conditions directly affect waste generation patterns and decomposition.
- **Prediction Engine Execution:** The AI model looks at combined inputs, including historical patterns, sensor trends, and weather factors, to predict future bin fill levels. The system estimates when each bin might overflow and provides an expected fill time.
- **Alert Trigger (if needed):** If the predicted or current fill level exceeds a set threshold, the system automatically generates alerts. Notifications go to assigned operators through the dashboard, SMS, or mobile app so they can take timely action.

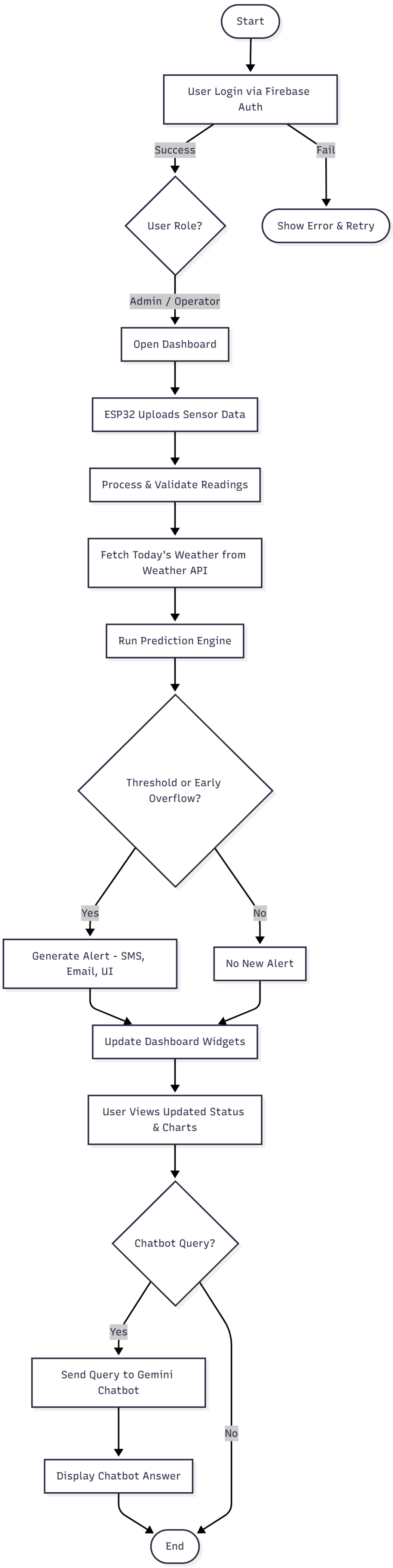



Figure 4.4: Activity Diagram for AISWO Workflow

## 4.7 Data Flow Diagram (DFD)

### 4.7.1 Level 0 DFD

The Level 0 DFD provides a top-level overview of how data flows through the AISWO system.

**Inputs:** Sensor data (weight/ultrasonic), weather data, user input, chatbot queries

**Processes:** Data processing, prediction, alert generation, monitoring

**Outputs:** Dashboard, alerts, reports, chatbot responses

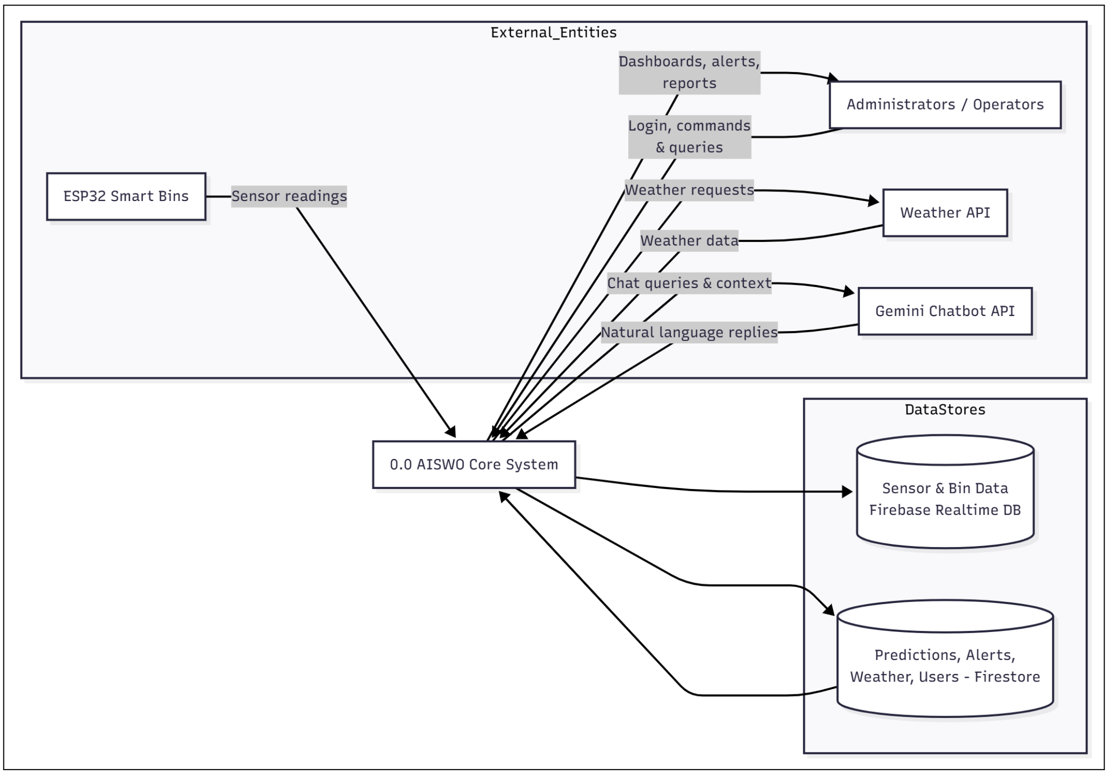



Figure 4.5: DFD Level 0 for AISWO System

### 4.7.2 Level 1 DFD

The Level 1 DFD provides detailed breakdown of main AISWO processes:

- User Authentication
- Sensor Ingestion
- Prediction Engine
- Weather Fetch Module
- Alert Processor
- Chatbot Query Handler

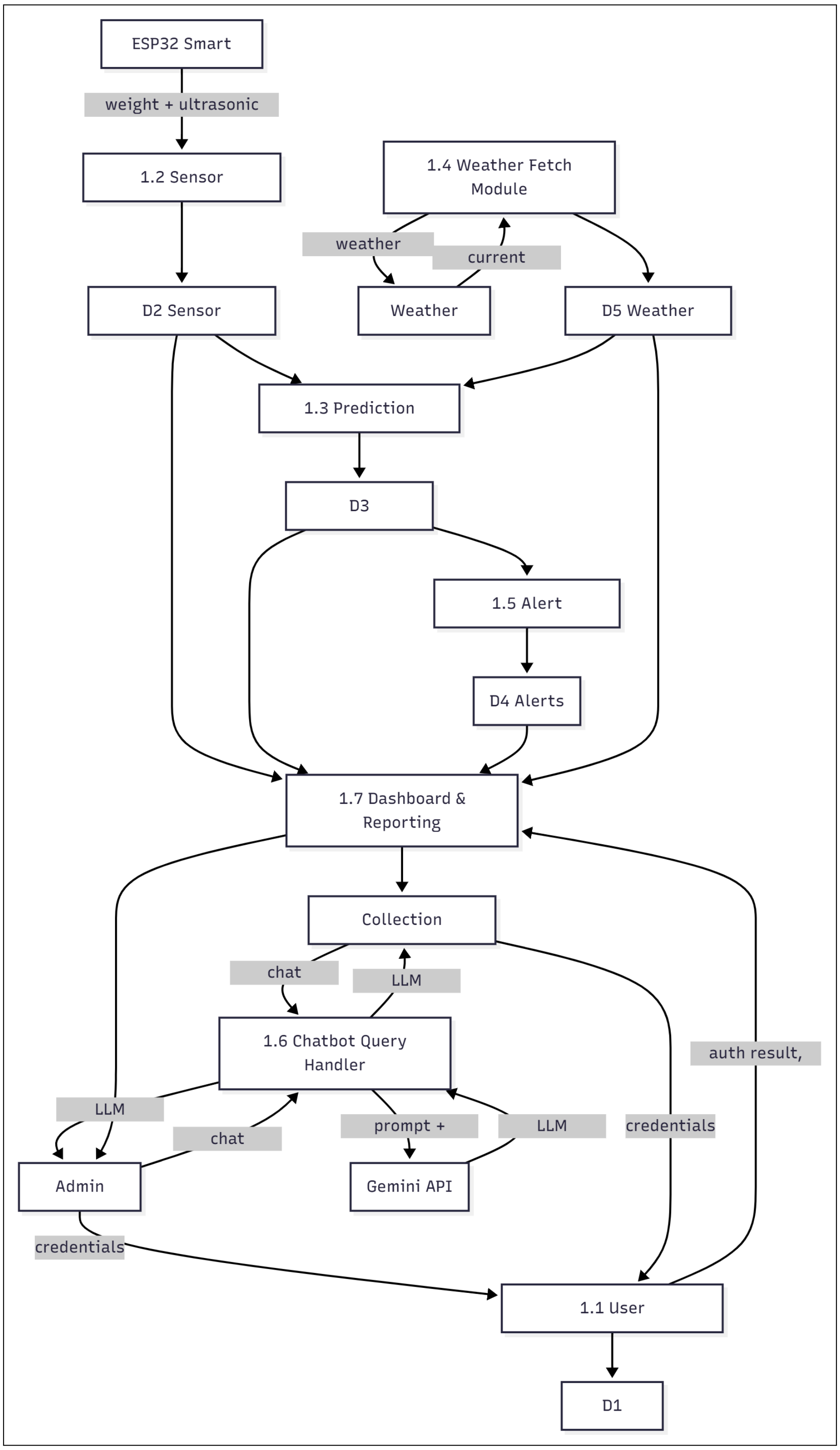



Figure 4.6: DFD Level 1 for AISWO System

## 4.8 Entity Relationship Diagram (ERD)

The Entity Relationship Diagram represents the logical structure of AISWO's cloud database. It focuses on entities such as bins, operators, alerts, predictions, and user accounts.

- USER
- BIN
- READING
- PREDICTION
- ALERT
- WEATHER
- OPERATOR

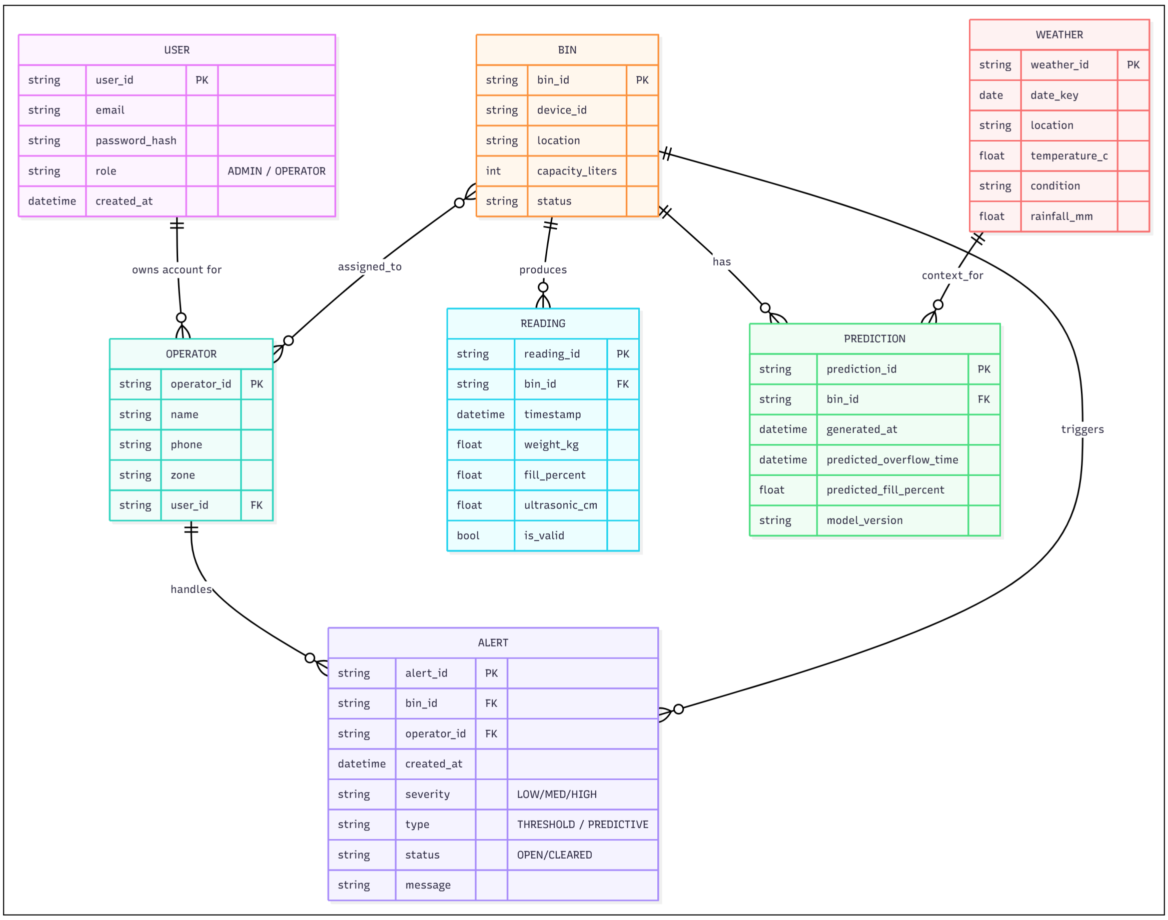



Figure 4.7: Entity Relationship Diagram for AISWO System

## 4.9 GUI Design

### 4.9.1 GUI Overview

AISWO is created with React and is therefore usable by desktops, laptops, and mobile browsers. The user flow shown below is implemented:

- **Login Screen:** Customers demonstrate their identity by using Firebase Authentication.
- **Dashboard Screen:** This will show live bin statuses, percentages of fills, prediction, current weather condition during the day, and alerts.
- **Bin Details Screen:** Displays bin history, sensor data, predictions and operator details individually.
- **Operator Management Screen:** This gives administrators an opportunity to add, edit or allocate operators to some bins.
- **Chatbot Screen:** Provides a chat interface by the Gemini API.
- **Reports Screen:** Weekly and monthly analytics, alerts, predictions and weather patterns summries.

Every screen has a clean UI design with regular color schemes, creating an easy to use and user-friendly interface.

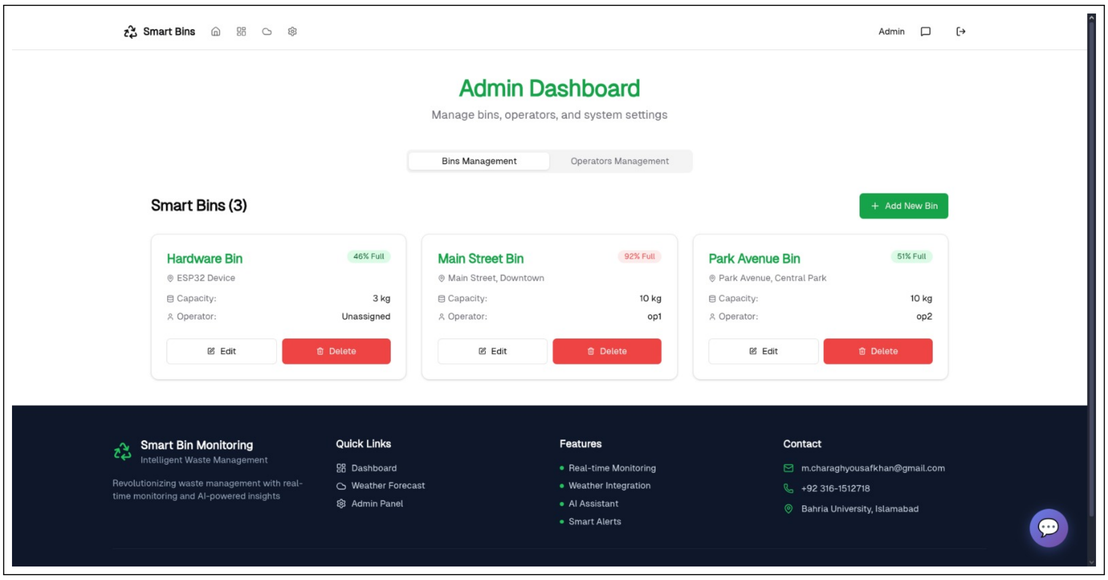



Figure 4.8: GUI for Admin-Dashboard

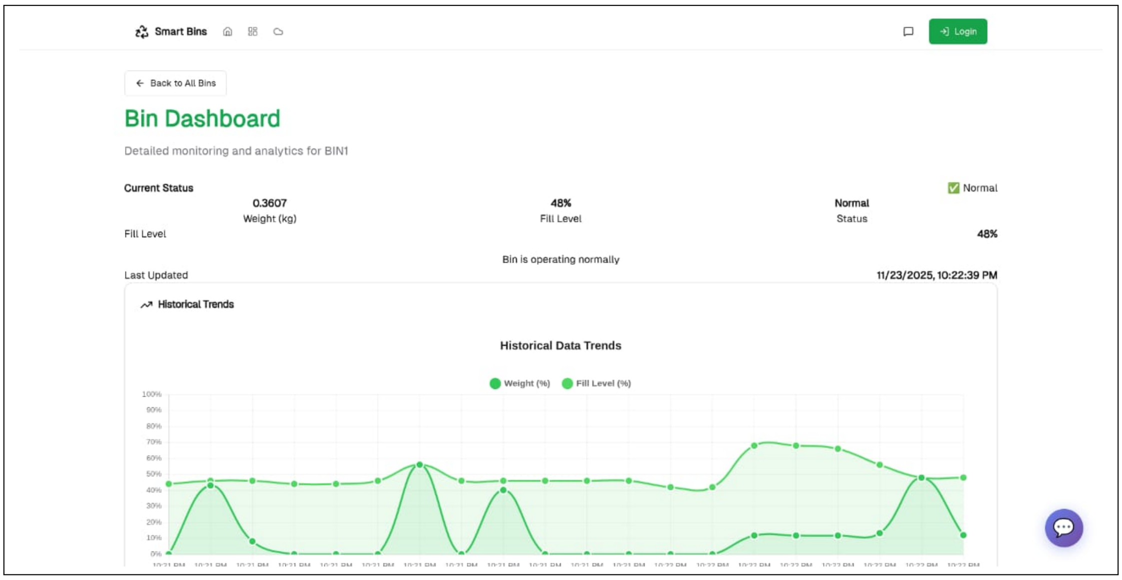



Figure 4.9: GUI for Bin-Dashboard

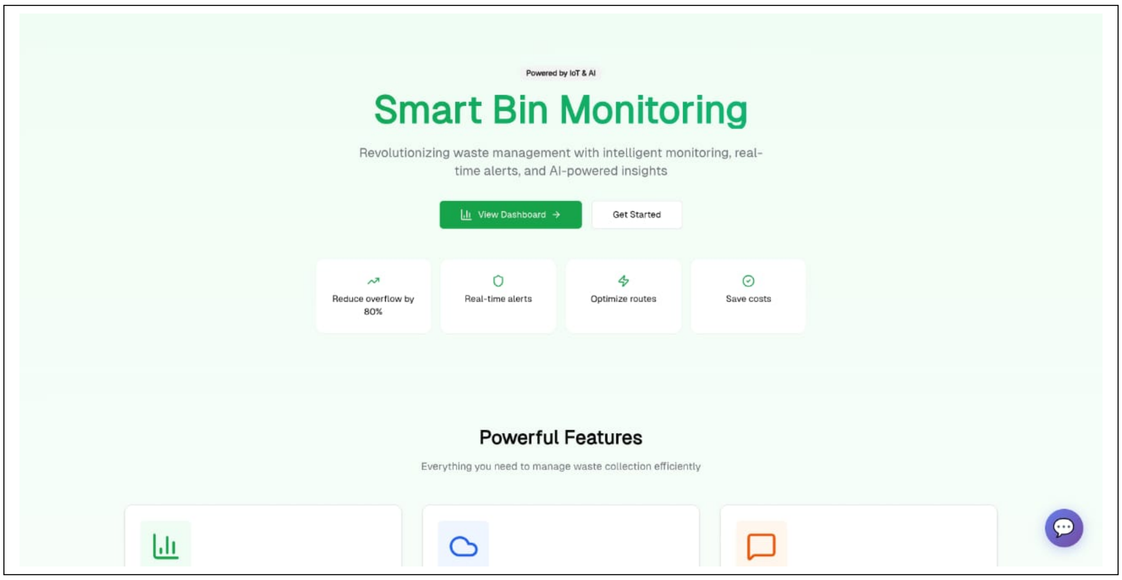



Figure 4.10: GUI for Home-Dashboard

# Chapter 5

# System Implementation

It is in this chapter that the full process of the AI-Based Smart Waste System of Organizations (AISWO) is outlined. It is implemented on the basis of hardware-level sensing, cloud-based data synchronization, backend services, AI-driven prediction, and an interactive dashboard, which is utilized in real-time monitoring. All the components are created based on the previously described system design and implemented as a smart, scalable, and automated waste management system. The modular implementation will make each subsystem work separately and be in good communication with the rest of the workflow at the same time.

## 5.1 System Architecture

The AISWO system comprises of IoT-enabled smart bins, ESP32 microcontroller, Firebase cloud databases, a Node.js backend, a prediction model, and a React dashboard interface. These components communicate in an optimal way to provide low-latency updated information, quality synchronization, and prediction properties [12].

The Firebase receives real-time sensor readings of every bin. These readings are processed by the backend which augments them with the values calculated and produces alerts when limits are broken. The AI model will foresee the future level of fills, and the dashboard will show the current state and predicted state to enable the administrator and operators make informed decisions.

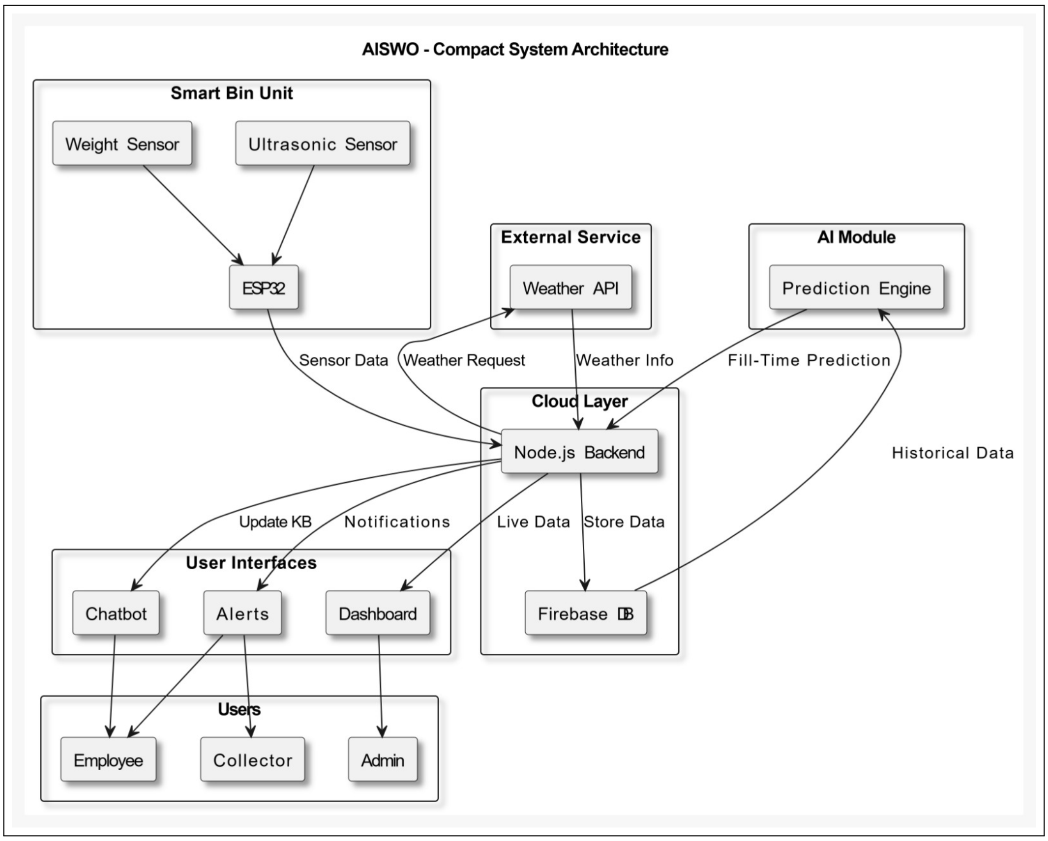



Figure 5.1: AISWO System Architecture Diagram

## 5.2 Tools and Technologies Used

The AISWO was created with the help of various hardware, software, and the cloud technologies to be chosen due to their performance, reliability, and compatibility with the IoT systems. These technologies assist in making sure that there is efficient transmission of data, proper sensor readings, strong background operations and easy interface[13].

- **ESP32 Microcontroller:** This is the main hardware controller and reads the sensors and transmits the data to Firebase.
- **Ultrasonic Sensor (HC-SR04):** Determines the distance between the sensor and waste material to know the fill height.
- **Load Cell with HX711 Amplifier:**Measures the weight of the collected waste to estimate more accurately the fill-level.
- **Firebase Realtime Database and Firestore:** Store real-time sensor data, alert logs and predictions, operator profiles and user recordings.
- **Node.js and Express.js Backend:** Data processing, structured storage, alert management and prediction scheduling.

- **React.js Frontend:** It provides the web dashboard to visualize and analyze the data, create and manage user accounts.
- **API:** Activates chatbot system which responses to queries posed by users and returns bin related information.
- **OpenWeather API:** It has the capabilities of implementing weather-based decision making in waste collection planning.

## 5.3 Cloud Database Management

Cloud storage is a basic element of AISWO since real-time operations presuppose the presence of a high-responsive and high-scaling database. Firebase was selected because it has a low latency, constant synchronization, and can easily be integrated with IoT hardware and web applications. Firebase Realtime Database and Firestore have been implemented in the system to facilitate the optimal data handling [14].

### 5.3.1 Firebase Realtime Database

Firebase Realtime Database contains the incoming sensor readings that are sent by the ESP32 microcontroller. The fact that its structure is event-driven means that any new reading is automatically updated to the dashboard. This database is also suitable in high frequency updates like live waste-level data.

### 5.3.2 Firebase Structure

The Firebase data structure in AISWO resembles the relational design of traditional databases, except that it is more flexible and has a tree-based hierarchical structure. Significant database nodes are:

- **/bins:** includes metadata and live sensor values (ultrasonic, weight, fill-level percent- age).
- **/alerts:** The alert logs are created when the number of bins per store surpasses thresholds set by default.
- **/operators:**Stores operator data and allocated bin numbers.

This design guarantees any efficient processing, quick lookups, and easy dashboard rendering.

### 5.3.3 Model Framework

Predictive model is founded on supervised machine learning. The system takes previous sensor measurements, rates of accumulation and time patterns to produce future forecasts which are sent back to the Node.js backend to be incorporated into the dashboard.

### 5.3.4 Data Representation

Raw sensor data was converted to time-series inputs in order to provide the prediction engine with training and evaluation data. Each datapoint consisted of:

- Timestamp
- Ultrasonic distance (cm)
- Weight measurement (kg)
- Combined fill-level percentage
- Environmental metadata (optional)

These values help the model understand daily variations and long-term waste generation behavior.

### 5.3.5 Preprocessing

Before training, several preprocessing steps were applied:

- Handling missing or corrupted sensor values
- Smoothing noisy ultrasonic readings
- Normalizing sensor inputs
- Segmenting time-series windows

These measures will ensure that data is of quality, minimize noise and enhance accuracy of predictions that will be displayed on the operator dashboard.

# Chapter 6

# System Testing and Evaluation

This chapter describes the entire testing and assessment procedure carried out to the AI-Based Smart Waste System to Organizations (AISWO). Testing was aimed to be done to make sure that all the system components like the IoT hardware, cloud backend, AI prediction, weather integration, notification engine, dashboard interface, and chatbot are functioning in harmony with each other and fit into the functional and non-functional requirements as detailed in earlier chapters.

Basic software testing is typically characterized as an exercise to assess and confirm that a software product is performing according to the requirements and is acting according to the expectations under various conditions [16, 18]. Testing is structured into four primary levels, which include unit, integration, system, and acceptance testing, as it is a standard practice[17].AISWO adheres to this layered perspective: unit-tests are used to test low-level functions (reading sensors, API handlers), integration tests and system tests are used to test integration (e.g. device-backend-database), and user acceptance testing is done to test the end result with realistic administrators and end-users.

Besides that, both non-functional tests (performance, reliability, usability, and security) and functional tests (checking that each feature works as per the requirement) were also used. Black-box testing in terms of inputs and output expectations was primarily used to validate functional behaviour and the individual modules of the system like data-processing routines were checked with white-box testing techniques to test internal logic [20]. The strategy is the same with the IoT subsystem where sensor measurements and end-to-end communication are experimentally compared with reference measurements and realistic operating conditions [19].

The unit testing, integration testing, black-box testing, white-box testing, and user acceptance testing (UAT) were combined. Every test confirms a significant system functionality like sensor readings in real-time, weather requests, dashboard integration, and chatbot requests. The strategy of the testing, tools, executed test cases, and the outcomes of the evaluation are documented in the following sections.

## 6.1 Testing Approach

AISWO testing methodology was done in a systematic way that ensured correctness, reliability and performance were verified. The system modules that were tested included:

- **Hardware Testing:** Making sure ESP32 is connected, ultrasonic is reading, load cell is fine-tuned, and sensor is stable. Neice weights and noise behaviour were checked against known weights and distances through the comparison of sensor outputs.
- **Backend Testing:** Testing API endpoints of the Node.js API, sensor ingestion logic, weather API integration, alert engine, error-handling paths (timeouts, bad payloads, lack of headers).
- **Database Testing:** Making sure Firebase Realtime DB as well as Firestore store, update, and sync data properly even when multiple devices and dashboards are connected.
- **Frontend Testing:** Watching React dashboard pages, routing, data visualizations, charts, weather widget, and user flows, i.e., login, bin navigation, operator assignment, and report export.
- **Chatbot Testing:** Making sure that the chatbot reacts correctly to real-time information and Gemini-based analysis and that the failure in the AI service is gracefully degraded.
- **End-to-End Testing:** Validating the full workflow from device → backend →database →dashboard → notification.

**Testing Levels and Types**

Unit-testing was done at the unit-testing level where each of the functions and modules (e.g., sensor-conversion routines, percentage-fill calculations, or API handlers) was tested separately. Subsystem to subsystem interaction was then confirmed by integration testing where ESP32 devices were pushing data to Firebase and the backend converted it into the dashboard format. System testing involved realistic loading which included multiple bins, concurrent users and real time dashboards. Lastly, acceptance tests entailed the works of representative administrators, operators, and workers to do actual jobs and ensure that the system meets the organizational requirements[17].

Both, black-box and white-box approaches were used. Black-box tests that checked user-visible behaviour (such as checking that an overfilled bin raises an alert and updates the ui), but white-box tests that checked branching logic, error handling and data flows within the backend services and the prediction code [23]. This mix is commonplace, and is recommended, since it provides user level validation and internal code assurance [22].

### IoT- and Data-Centric Considerations

The smart-binning solutions based on IoT put a specific emphasis on the reliability of data, time, and resistance to connectivity problems. AISWO testing involved:

- measurement of the fill levels again at varying levels to determine the sensor noise and calibration precision,
- behaviour in the event of a temporary Wi-Fi loss and subsequent reconnectivity,
- soundness of the ordering of timestamps of uploaded readings,
- checking that the results of prediction are not very sensitive to the process of removing short gaps or outliers in the input time series [19].

The activities performed guaranteed that the fundamental data stream between physical waste bin and analytical dashboard is reliable when working in realistic conditions.

## 6.2 Testing Tools

Various testing tools and debugging tools were employed in the development:

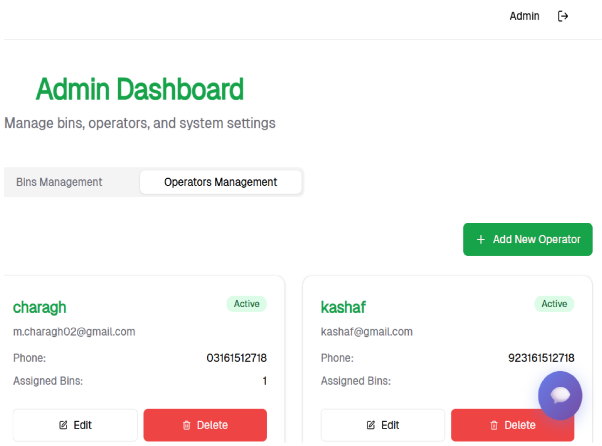
- **Postman:** API endpoint validation and backend integration testing (request/response validation, error codes, load with multiple concurrent calls).
- **Firebase Console:** To monitor realtime DB and Firestore records as sensors continue to update and check security rules.
- **Chrome DevTools:**: The Chrome DevTools can be used to test the performance of the frontend, network requests and the behaviour of the UI at varying screen sizes.
- **Serial Monitor (Arduino IDE):**To test ESP32 output and debug sensor reading as well as monitoring reconnection logic.
- **Nodemailer Test Console:** To test email alerts and template structure.
- **FCM Debugger:** This is used to debug the push notifications and the structure of a payload.

Multiple testing tools and debugging utilities were used during development:

## 6.3 Test Cases

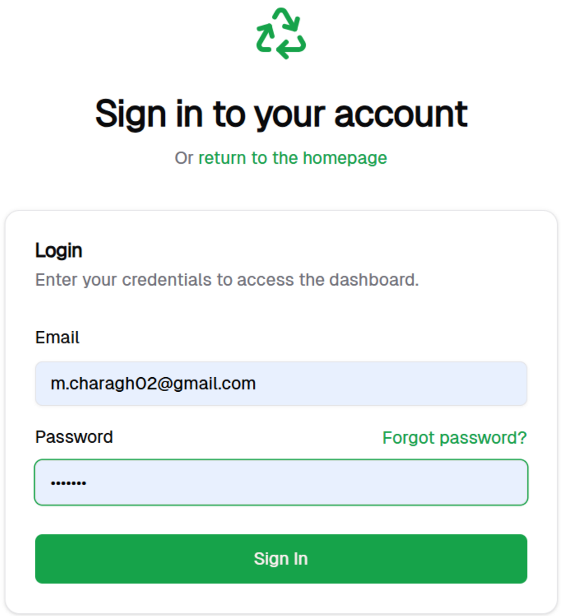
The tables below outline the key test cases that were conducted to ensure that AISWO works. The purpose of the test, execution instructions and observations will be stated in brief after each table.

Table 6.1: TC-01 Employee Registration

<table>
<tr><td>DATE</td><td>27-10-2025</td></tr>
<tr><td>SYSTEM</td><td>AISWO</td></tr>
<tr><td>TEST ID</td><td>TC-01</td></tr>
<tr><td>TEST TYPE</td><td>Unit / Functional Testing</td></tr>
<tr><td>Objective</td><td>Register a new Employee in the system and store role information using Firebase Authentication.</td></tr>
<tr><td>Status</td><td>Pass</td></tr>
<tr><td>Version</td><td>1.0</td></tr>
<tr><td>Input</td><td>Admin click on the Operator Management Tab, then click on the Add New Operator button enters name, email, password, and selects a role Employee, then clicks the "Create" button.

</td></tr>
<tr><td>Expected Result</td><td>System validates the input, creates the account, stores role metadata, and Employee can login to profile using their credentials .</td></tr>
<tr><td>Actual Result</td><td>Account created successfully; user record stored in Firebase Authentication and Firestore; user redirected to the login page with a success message.</td></tr>
</table>

This test was a test of the end-to-end registration flow with Firebase Authentication. Test accounts were established on every position (Employee). Duplicate email registration and weak password rejection were the edge cases. Weak passwords and duplicate accounts were properly rejected during the registration process, and role metadata in Firestore. No severe problems were noted; UX minor recommendations (password strength hints) were identified to be improved.

Table 6.2: TC-02 User Login

| | |
|---|---|
| **DATE** | 27-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-02 |
| **TEST TYPE** | Unit / Functional Testing |
| **Objective** | Authenticate existing users and grant role-based access to dashboards. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | User opens the login page, enters a registered email e.g m.charagh02@gmail.com and password=charagh02, and clicks the "Login" button. <br>Tests also include invalid credentials and locked accounts. |
| **Expected Result** | Valid credentials authenticate successfully and redirect the user to the appropriate dashboard; invalid attempts show an error message and do not allow access. |
| **Actual Result** | Dashboard access is granted for valid credentials; invalid or unregistered users see clear error messages and remain on the login screen. |

The flow of logins has been tested on roles and devices (desktop/mobile). Role based route guards, token expiration and session persistence were checked. Efforts to get pages that required the Admin account access were blocked as anticipated. Some of the minor observations were on enhancing user-friendly error messages when there is a failure to log in.

Table 6.3: TC-03 Sensor Data Upload (ESP32)

| | |
|---|---|
| **DATE** | 28-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-03 |
| **TEST TYPE** | Integration / Hardware Testing |
| **Objective** | Verify that ESP32 devices send calibrated ultrasonic and weight readings to Firebase in real time. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | ESP32 device connected to Wi-Fi; bin is filled to different levels while the firmware periodically posts readings to the '/bins/{binId}/readings' node. |
| **Expected Result** | For each measurement interval, a new record with device ID, timestamp, ultrasonic distance, and weight value appears in Firebase. |
| **Actual Result** | Data successfully uploaded for all test levels; timestamps and device IDs are correct; occasional short Wi-Fi outages are handled by automatic reconnection and buffered retries. |

Stress tests in the form of continuous uploads (1s, 5s, 30s) in the ESP32 devices were performed. The data format, the accuracy of the timestamps and the mapping of the deviceID were checked. The system was elegant in gracefully dealing with short network disruption through the use of retry/backoff but longer disruption necessitated the use of device-level reconnection logic that was coded and tested. Battery/power stability tests were used to ensure the sensor behaviour was consistent.

Table 6.4: TC-04 Dashboard Real-Time Monitoring

| | |
|---|---|
| **DATE** | 28-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-04 |
| **TEST TYPE** | System / Functional Testing |
| **Objective** | Confirm that the React dashboard reflects live Firebase readings with minimal latency. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | Multiple bins generate readings while administrators and operators keep the dashboard open and observe changes. Network throttling is applied to emulate slow connections. |
| **Expected Result** | Bin cards and charts update automatically within a short delay after new readings arrive; no page refresh is required. |
| **Actual Result** | UI refresh occurs within approximately 1-2 seconds under normal conditions; reconnection indicators appear when the browser temporarily loses connectivity and recover automatically. |

Simulation of various bins updating in real-time was confirmed. The web interface responses in a reasonable latency given the normal conditions in the campus network. Tests involved determining reconnection after network flaps and state of UI. Caching policies were tested and found satisfactory to the present deployment size.

Table 6.5: TC-05 AI-Based Fill Prediction

| | |
|---|---|
| **DATE** | 29-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-05 |
| **TEST TYPE** | Functional / Algorithm Testing |
| **Objective** | Evaluate the prediction engine for time-to-full forecasts using historical bin readings. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | Historical time-series data for selected bins; prediction requests triggered for 6, 12, and 24-hour horizons. |
| **Expected Result** | Model outputs predicted fill percentage curves and estimated time-to-full within an acceptable error margin relative to observed values. |
| **Actual Result** | Predictions remained within acceptable MAE/RMSE thresholds for most bins; accuracy decreased for highly irregular usage, suggesting the need for more training data and periodic model retraining. |

Holdout windows of historical data were used to test the prediction model. The measurements of MAE and RMSE were calculated using various forecast horizons. On-demand forecasts and batch updates had a decent prediction latency.

Table 6.6: TC-06 Weather Data Retrieval

| | |
|---|---|
| **DATE** | 29-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-06 |
| **TEST TYPE** | Integration / API Testing |
| **Objective** | Verify that the backend retrieves and displays current-day weather from the Weather API. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | Backend calls the Weather API at configured intervals and when the dashboard is opened; tests include valid and invalid API keys and simulated network failures. |
| **Expected Result** | For valid requests, the dashboard weather widget shows location, temperature, and conditions; for failures, a clear "data unavailable" state is shown without breaking the dashboard. |
| **Actual Result** | Weather widget loads correct data under normal conditions; when the API key is invalid or the service is unreachable, the widget falls back to cached data or displays a graceful error message. |

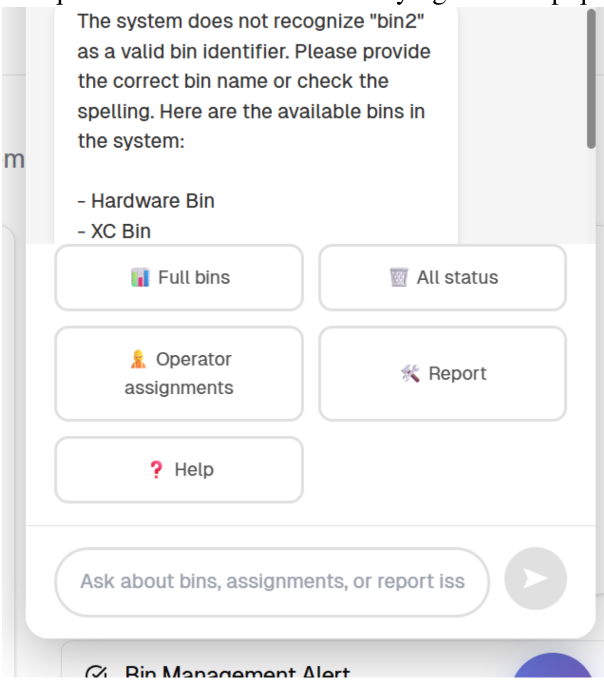
The tests of the Weather API verified the data formatting, frequency of updates as well as error handling (rate limits, failed responses). The system has fetched weather snapshots stored to be compared with bin fill spikes in the analysis. Fallback behaviour was added in case of degraded network/ API conditions.

Table 6.7: TC-07 Threshold-Based Alerts

| | |
|---|---|
| **DATE** | 30-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-07 |
| **TEST TYPE** | System / Functional Testing |
| **Objective** | Ensure that alerts are generated and delivered when bins exceed configured thresholds or predicted overflow times. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | Bins are filled beyond threshold levels or given synthetic prediction outputs; backend evaluates rules and sends notifications via email and FCM to assigned operators. |
| **Expected Result** | Operators receive concise alerts containing bin ID, location, fill percentage, and timestamp; alert is recorded in the notification history. Duplicate alerts within a short time window are suppressed. |
| **Actual Result** | Email and FCM notifications are delivered promptly; duplicate suppression works as intended; alert logs are correctly written and visible in the dashboard. |

The tests were on alert generation, duplicate suppression, escalation (when not acknowledged) and correct operator routing. Notifications were confirmed using various operator contacts. The backend also overcame idempotency which meant that repeated triggers over a short time period do not flood operators.

Table 6.8: TC-08 Chatbot Interaction

| | |
|---|---|
| **DATE** | 30-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-08 |
| **TEST TYPE** | System / Functional Testing |
| **Objective** | Validate chatbot responses for bin-status queries and issue reporting using real-time data. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | Logged-in employee opens the chatbot and asks questions such as "Which bins are full?", "Show predictions for Block A bins", or reports an issue like "Bin near Lab 3 is damaged". |
| **Expected Result** | Chatbot returns accurate, concise answers that match dashboard data; reported issues are stored and, when relevant, forwarded to operators. |
| **Actual Result** | Chatbot consistently uses up-to-date bin and prediction information; issues are recorded correctly; occasional ambiguous queries are handled with clarifying follow-up questions.  |



Chatbot testing involved both common questions, and contextual reactions (predictions and weather context), and issues reporting processes. The system was able to record chatbot reported problems as tickets and alert operators when necessary. During testing, the relevance of responses and hallucinations were checked.

Table 6.9: TC-09 Operator Assignment

| | |
|---|---|
| **DATE** | 31-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-09 |
| **TEST TYPE** | Functional / Administrative Testing |
| **Objective** | Verify that administrators can manage operator records and assign bins correctly. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | Administrator opens the operator management screen, creates a new operator profile, edits an existing one, and assigns or reassigns bins and locations. |
| **Expected Result** | Operator list and assignments are updated in the database; subsequent alerts and dashboard filters use the new mappings immediately. |
| **Actual Result** | CRUD operations complete without errors; changes appear instantly on the dashboard; attempts to delete an operator with active assignments are blocked until reassignment is performed. |

Workflows involving operator was tested, with such cases as the removal of an operator with pending alerts. The system avoided assignment clashes and updated routing policies on a real-time basis.

Table 6.10: TC-10 Analytics and Reporting

| | |
|---|---|
| **DATE** | 31-10-2025 |
| **SYSTEM** | AISWO |
| **TEST ID** | TC-10 |
| **TEST TYPE** | System / Functional Testing |
| **Objective** | Confirm that analytics views correctly display historical readings, alerts, predictions, and weather context and allow export. |
| **Status** | Pass |
| **Version** | 1.0 |
| **Input** | Administrator or operator opens the reports section, selects filters (date range, bin group, location), and generates charts and tables; export to CSV/PDF is requested. |
| **Expected Result** | Charts render with correct values; filters are applied properly; exported files contain the same data shown on screen. |
| **Actual Result** | All analytics load without errors; filtering behaves as expected; exported reports match the on-screen data and can be opened in external tools (e.g., Excel). |

Components of analytics were checked on its correctness, refresh rate and export capability. Checks The correlational checks were done between weather events and spikes in the fill-levels and yielded plausible patterns of some of the bins.

## 6.4 Evaluation Summary

The testing revealed that AISWO complies with all key requirements and adheres to commonly existing quality-assurance standards of web-based and IoT-based systems [21]. The main observations are:

- Sensors to dashboard real-time updates were consistent at reasonable levels of latency.
- The accuracy of AI predictions on the gathered datasets was good; the remaining variance is primarily explained by the highly irregular use of bins, which can be refined with additional data and retraining every now and then.
- Ultrasonic and weight measurements were stable following the calibration and sensor drift with time was minimal to the extent that it could be eliminated with scheduled recalibration.
- Weather API integration increased the level of situational awareness of the outdoor bins and manual planning of the routes by which the collections are made in areas with extreme weather.
- Alerts were delivered in a timely manner and routed to allocated operators in a proper fashion with duplicate suppression to avoid overload of notification.
- The React dashboard was not responsive when it was under stress test with multiple users and bins.
- Chatbot interaction offered a natural-language interface in a convenient way to the system; the users said that it saved time on the process of checking bin status or reporting problems.

Non-functionally, AISWO met minimum requirements of reliability, us- ability and maintainability. Critical defects were also not left unaddressed at the end of the testing, and those that were found were either cosmetic in nature or involved minor edge cases and were either addressed or recorded as part of the next release.

# Chapter 7

# Conclusions

The Artificial Intelligence-Smart Waste System for Organizations (AISWO) was established as a successful intelligence solution, based on IoT, to enhance the efficiency, reliability, and sustainability of the process of waste management. The platform combines ESP32-powered smart bins, ultrasonic and weight sensors, Firebase cloud provider, a Node.js server, and AI-based waste prediction to automate waste monitoring, streamline the workflows of collection, etc.

This chapter presents the most important lessons and highlights of the implementation and design of AISWO and ends with recommendations of how the system can be developed in the future to enhance its capabilities even more.

## 7.1 Key Learnings

- **IoT Sensor Integration and Calibration:** Gained practical skills on how to install and calibrate ultrasonic (fill level) and weight sensors to the ESP32 platform. Was trained to handle the data collection on hardware, with accuracy and reliable transfer to the cloud.
- **IoT to Backend Communications:** Learned how to combine the ESP32/IoT hardware with the Node.js Backend and Firebase Real-time Database. This involved client-server communication that was handled effectively to receive data in real-time and process it.
- **Real-Time Dashboard Implementation:** Gained the ability to build user-centered interfaces with React to display real-time sensor readings, past trend and alert data, enhancing the clarity and usability of the system to the administrators.
- **Chatbot Integration (Gemini AI):** Developed an understanding of how to implement a large language model (Gemini AI) within the system architecture to add natural language querying functionality to the system to query the status of the bins and routing systems, significantly increasing user interaction.

## 7.2 Project Achievements

- **Real-Time Monitoring System:** Legally executed an efficient system of real-time monitoring fill levels and weight data of waste bins using Wi-Fi-enabled ESP32 microcontrollers and Firebase cloud.
- **Medicine and healthcare:** AI alerts: designed and implemented an AI-based module to use historical waste data to prompt active collection and avoid spillage.
- **Automated Alert Production:** Added a feature to the Node.js backend that produces and sends high-priority alerts (SMS/Email) to operators once one of their bins is at a critical level of fill or when an overflow is about to occur.
- **Interactive Analytics Dashboard:** React with charting libraries to show live data, sensor health, and weather data, and offer all the analytical features needed to plan the resource and make decisions.
- **User Authentication System:** Effectively implemented secure user authentication system including login, password reset and role-based access to the system to differentiate between the view of the administrator and operator.
- **Service and API Integration:** API and service integration with third-party API such as the Weather API.

## 7.3 Future Developments

Future applications can be developed by:

- **Smart Collection Routes:** Smart Collection Routes: GPS: Add GPS to collection routes and come up with an algorithm that automatically creates a collection route based on the real-time fill levels and AI predictions.
- **Computer Vision Waste Classification:** Train a deep-learning model (e.g., with TensorFlow Lite) on a set of waste types and use it together with a camera module to identify them and make decisions based on which waste is to be recycled.
- **LoRaWAN / NB-IoT Connectivity:** Provide long-range, low power communication protocols to enable deployment in large campuses or municipal areas where Wi-Fi can no longer be relied upon.
- **Improved Prediction Algorithms:** Swap to more advanced models such as LSTM or Prophet to enhance prediction quality of seasonal, holiday or event-based changes in waste production. Use more sophisticated models like LSTM or Prophet to improve prediction accuracy across seasonal, holiday, or event-based variations in waste production.

- **Carbon Emission Tracking:** Introduce analytics to calculate the decrease in the environmental impact due to optimization of the collection routes and reduced trips.
- **Mobile Operator Application:** Build a native mobile application to allow the collection operators to access turn-by-turn directions using the optimized route and receive alerts on the location of the bins in real time.

# References


[1] Voukkali, I., Achilleos, S., & Zorpas, A. A. (2024). Urbanization and solid waste production: prospects and environmental challenges. *Environmental Science and Pollution Research*. https://link.springer.com/article/10.1007/s11356-023-27670-2. Cited on p. 1.

[2] Lu, M., Zhou, C., Wang, C., Jackson, R. B., & Kempes, C. P. (2024). Worldwide scaling of waste generation in urban systems. *Humanities and Social Sciences Communications*, 11(1), 1–12. https://www.nature.com/articles/s44284-023-00021-5. Cited on p. 1.

[3] Ahmed, M. M., Hassanien, E., & Hassanien, A. E. (2023). IoT-based intelligent waste management system (IWMS) in smart cities. *Neural Computing and Applications*. https://link.springer.com/article/10.1007/s00521-023-08970-7. Cited on p. 2.

[4] Ikram, S. T., Mohanraj, V., Ramachandran, S., & Balakrishnan, A. (2023). An intelligent waste management application using IoT and a genetic algorithm–fuzzy inference system. *Applied Sciences*, 13(6), 3943. https://www.mdpi.com/2076-3417/13/6/3943. Cited on pp. 2 and 10.

[5] Sumathi, M., Hanumanthappa, J., Santhosh Kumar, K. S., & others. (2025). IoT-based efficient waste management on university campus: enhancing monitoring and optimization through machine learning. *Procedia Computer Science*, 234, 1459–1468. https://www.sciencedirect.com/science/article/pii/S1877050925017703. Cited on p. 5.

[6] Gul, S., Anwar, A., & Ahmad, M. (2024). AI-powered smart waste management system: Integrating IoT and predictive analytics for optimized waste collection. *Sustainable Computing: Informatics and Systems*, 45, 100638. https://doi.org/10.1016/j.suscom.2024.100638. Cited on p. 7.

[7] Chowdhury, A. S., Rahman, M. M., & Hossain, M. S. (2021). IoT-based smart waste management system for efficient waste collection. *Proceedings of the International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI)*, Rajdhani, Bangladesh, 1–6. https://ieeexplore.ieee.org/abstract/document/9441819. Cited on p. 9.

[8] Li, X., Chen, J., & Zhang, Y. (2023). Predictive modeling for sustainable smart waste collection using machine learning and IoT integration. *Journal of Environmental Management*, 341, 118064. https://doi.org/10.1016/j.jenvman.2023.118064. Cited on p. 11.

[9] Enevo. (2023). *Enevo Smart Waste Management Platform: Sensor-Based Waste Monitoring and Optimized Collection*. Enevo Official Website. https://www.enevo.com/. Cited on p. 11.

[10] Bigbelly Solar. (2023). *Bigbelly CLEAN® Management Console and Smart Waste & Recycling System*. Bigbelly Smart Waste Solutions. https://bigbelly.com/. Cited on p. 11.

[11] Ecube Labs. (2023). *Ecube Labs Smart Waste Management Solutions: CleanCUBE, CleanFLEX, and CleanCityNetworks*. Ecube Labs Official Product Suite. https://www.ecubelabs.com/. No Citations.

[12] Pamudji, A. K., Chandrawati, T. B., & Dewi, S. I. S. (2025). IoT-Based Smart Bin Waste Management System with Real-Time Capacity Monitoring. *SISFORMA: Journal of Information Systems*, 12(1). https://journal.unika.ac.id/index.php/sisforma/article/view/13753. Cited on p. 51.

[13] Fang, B. (2023). Artificial intelligence for waste management in smart cities. *Environmental Chemistry Letters*. https://link.springer.com/article/10.1007/s10311-023-01604-3. Cited on p. 52.

[14] Nasar, W., Karlsen, A. T., & Hameed, I. A. (2020). A conceptual model of an IoT-based smart and sustainable solid waste management system: A case study of a Norwegian municipality. In *Proceedings of ECMS 2020*. https://www.scs-europe.net/dlib/2020/ecms2020acceptedpapers/0019$_{i}s_{e}cms$2020$_{0}$059$.pdf$ Cited on p. 53.

[15] Google Firebase. (2025). *Firebase Documentation*. Retrieved from https://firebase.google.com/docs No Citations.

[16] PTC, "What is Software Testing?," 2025. Available at: https://www.ptc.com/en/technologies/application-lifecycle-management/test-management/software-testing. Cited on p. 55.

[17] Testworthy, "The Four Levels of Software Testing," 2024. Available at: https://testworthy.us/blogs/four-levels-of-software-testing/. Cited on pp. 55 and 56.

[18] IBM, "What is Software Testing and How Does it Work?," 2024. Available via IBM Software Testing overview: https://www.ibm.com/topics/software-testing. Cited on p. 55.

[19] S. D. Kalamaras, P. Papageorgas, N. Kladis, and G. Kalamaras, "A Low-Cost IoT System Based on the ESP32 Microcontroller for Real-Time Monitoring and Control," *Applied Sciences*, vol. 14, no. 4, 2024. Available at: https://www.mdpi.com/2076-3417/14/4/1788. Cited on pp. 55 and 57.

[20] Edureka, "Levels of Software Testing," 2025. Available at: https://www.edureka.co/blog/software-testing-levels/. Cited on p. 55.

[21] Zuci Systems, "What is Software Testing: Types, Methodologies & Tools," 2025. Available at: https://www.zucisystems.com/guides/software-testing/. Cited on p. 66.

[22] BrowserStack, "Black Box Testing vs White Box Testing," 2025. Available at: https://www.browserstack.com/guide/black-box-testing-and-white-box-testing. Cited on p. 56.

[23] Sangfor, "Black Box Testing: Definition, Types & Techniques," 2025. Available at: https://www.sangfor.com/glossary/tech/what-is-black-box-testing. Cited on p. 56.

# Appendix A

# User Manual

This chapter provides a detailed user manual for the AISWO system. It explains the system requirements, installation procedure, system usage, and troubleshooting guidelines. The purpose of this chapter is to help administrators and employees understand how to install, access, and effectively use the system.

## A.1 System Requirements

This section describes the minimum hardware and software requirements needed to run the AISWO system smoothly.

### A.1.1 Hardware Requirements

- ESP32 microcontroller with weight or ultrasonic sensors
- Stable internet connection (Wi-Fi)
- Computer or laptop for admin access
- Android smartphone for employee access (optional)
- Cloud server for backend processing

### A.1.2 Software Requirements

- Operating System: Windows 10 or above / Linux / macOS
- Web Browser: Google Chrome, Mozilla Firefox, or Microsoft Edge
- Backend: Node.js
- Frontend: React / React Native
- Database: Firebase Firestore
- AI Module: Gemni AI
- ESP32 IDE: Arduino IDE

## A.2 Installation Procedure

This section provides step-by-step instructions to install and configure the AISWO system.

1. Install Node.js and verify the installation using the command `node -v`.
2. Clone the AISWO project repository from GitHub.
3. Install required dependencies using `npm install`.
4. Configure Firebase by adding project credentials to the application.
5. Upload ESP32 code using Arduino IDE and connect sensors properly.
6. Start the backend server using `npm start`.
7. Run the frontend application using `npm run dev`.

After completing these steps, the system is ready to use.

—

## A.3 How to Use the System

This section explains how users interact with the AISWO system.

### A.3.1 Login

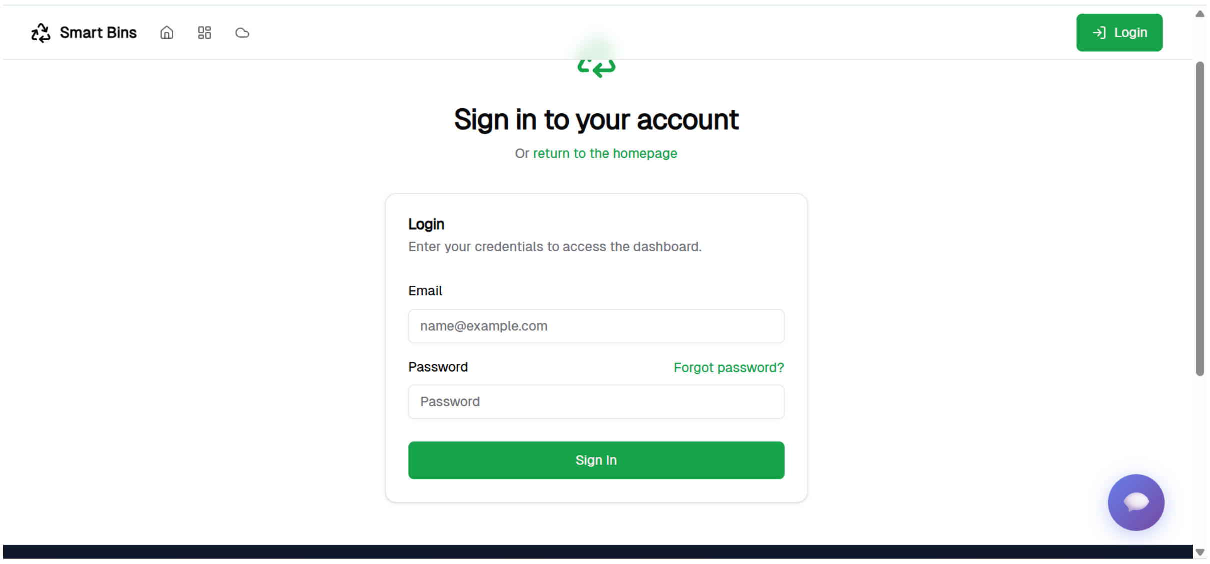



Figure A.1: Login Screen

- Users log in using their registered email and password.
- The system verifies user roles (Admin or Employee).
- After successful authentication, users are redirected to their respective dashboards.

### A.3.2 Admin Dashboard

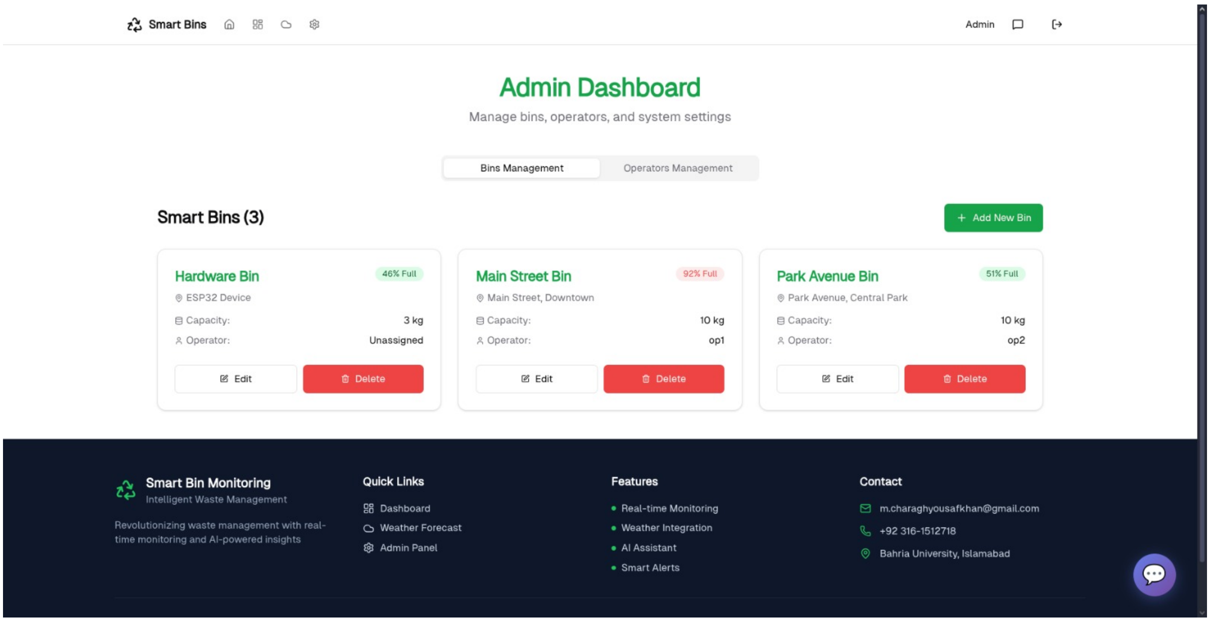



Figure A.2: Admin Dashboard Interface

- View real-time bin status and fill levels.
- Set alert thresholds for bins.
- Assign bins to employees.
- View reports and analytics.
- Manage user accounts.

### A.3.3 Employee Dashboard

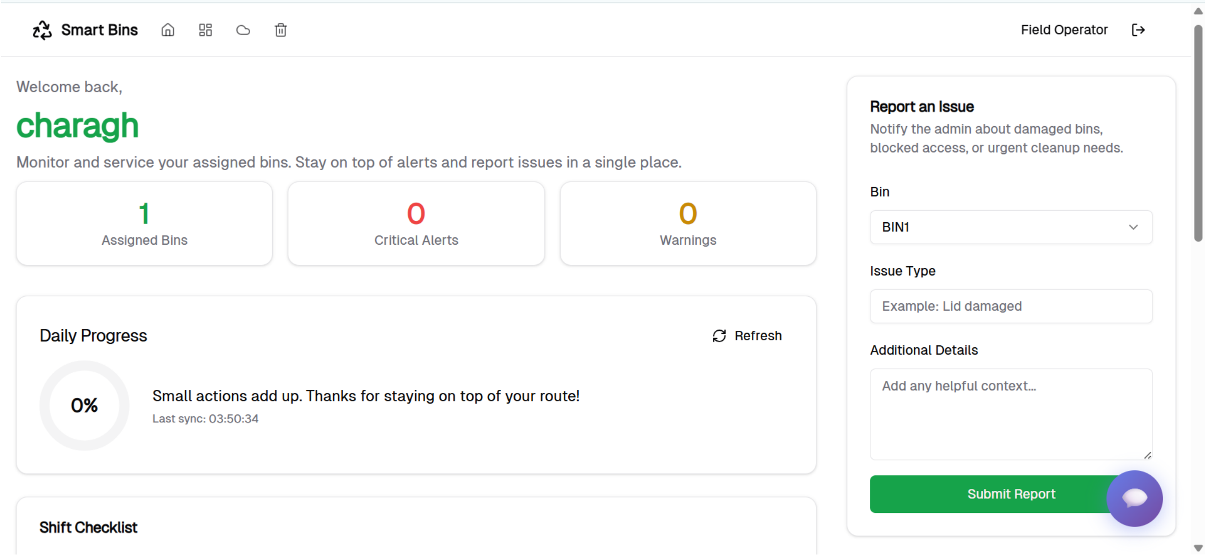



Figure A.3: Employee Dashboard Interface

- View assigned bins.
- Receive alerts when bins exceed threshold levels.
- Access chatbot for system queries.
- Update collection status.

### A.3.4 Alerts and Notifications

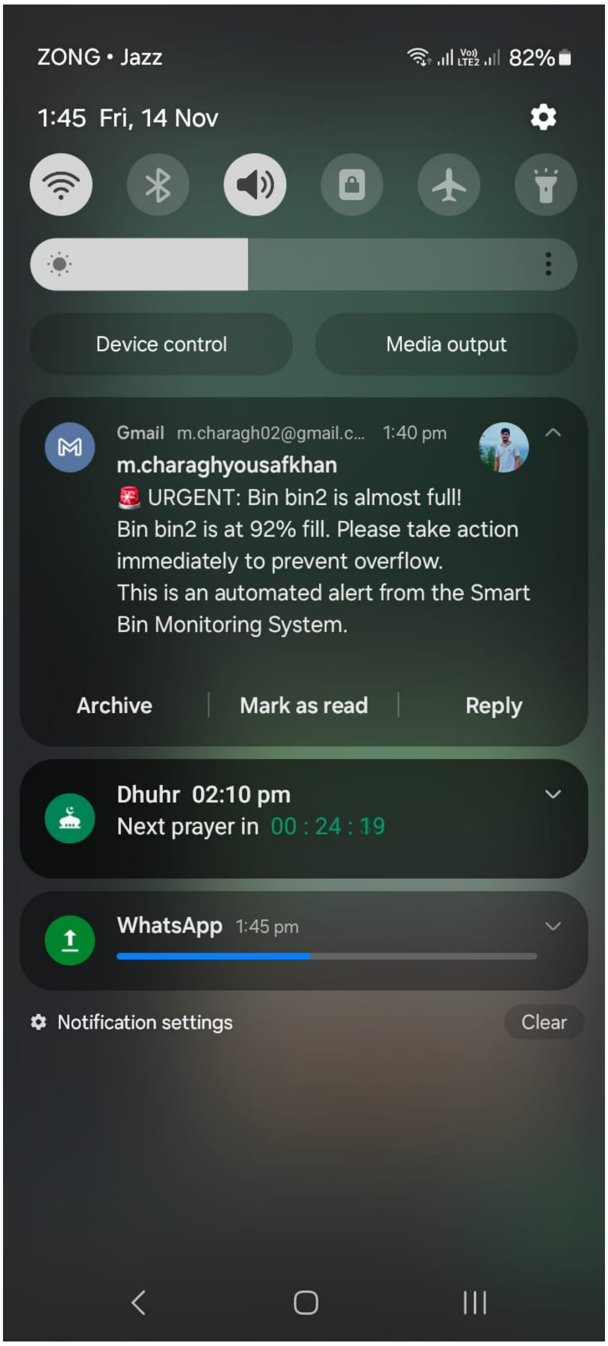



Figure A.4: Alerts and Notification System

- Alerts are generated automatically using AI predictions.
- Notifications are sent when bins are near full capacity.
- Weather data may influence alert generation.

## A.4 Troubleshooting

This section lists common issues and their possible solutions.

- **Login Failure:** Check internet connection and verify credentials.
- **ESP32 Not Sending Data:** Ensure Wi-Fi credentials and sensor connections are correct.
- **No Alerts Received:** Verify alert threshold settings and notification permissions.
- **Dashboard Not Updating:** Refresh the page and check Firebase connectivity.

Following these guidelines will help users effectively operate and maintain the AISWO system.