Autonomous Biomedical Waste Segregation Robot

QURAT UL AIN
01-134221-066

Supervisor: Dr. Muhammad Asfand-e-Yar

Bachelor of Science in Computer Science

Department of Computer Science
Bahria University, Islamabad

December 2025

Abstract

The increasing volume of biomedical waste generated in hospitals and clinical environments presents serious challenges to environmental sustainability and occupational safety. Improper segregation and manual handling of hazardous waste, particularly infectious materials and sharp objects, significantly increase the risk of disease transmission and accidental injuries among healthcare personnel. These risks are further intensified in developing regions where biomedical waste management practices remain largely manual and insufficient. Consequently, there is a growing need for intelligent and automated waste segregation solutions that can improve safety and operational efficiency.

This research presents the design and implementation of an Autonomous Biomedical Waste Segregation Robot that integrates computer vision–based classification, robotic manipulation, ultraviolet sterilization, and cloud-enabled monitoring into a unified framework. The proposed system uses an ESP32 camera module to capture real-time visual data, which is processed using a YOLOv8 deep learning model trained to classify biomedical waste into infectious, sharp, and non-infectious categories. Based on the classification results, a simulated Franka Emika Panda robotic arm, implemented in the PyBullet environment, performs automated pick-and-place operations, including UV sterilization prior to disposal into category-specific bins.

An Internet of Things architecture connects the detection and segregation pipeline to a Firestore database and a Streamlit-based web dashboard that provides real-time system visualization, bin fill level monitoring, historical data analysis, automated alert generation, and predictive analytics. The prediction module estimates future bin fill levels using historical waste data, enabling proactive planning of waste collection activities and improving operational decision-making.

The results demonstrate the technical feasibility of integrating deep learning, robotic automation, ultraviolet sterilization, and cloud-based monitoring into a single autonomous biomedical waste management system. The proposed framework offers a scalable and cost-effective approach that can be adapted to healthcare facilities of varying sizes. By reducing manual intervention and enabling predictive monitoring, the system improves safety, supports regulatory compliance, and contributes to more intelligent and reliable biomedical waste management practices.

Acknowledgments

Starting with the name of Almighty Allah, who is the Most Benevolent and the Most Merciful, without whose help and blessings, I could never imagine having accomplished this project or indeed any achievement in my life. His mercy was the greatest driving force that got me through every phase and challenge.

I express my profound gratitude to my supervisor, Dr. Muhammad Asfand e Yar, for his trust and guidance throughout this journey. His confidence in me helped restore my own self-belief and motivated me to persevere.

I am eternally grateful to my parents for believing in me and giving me the confidence to achieve great things. Their presence itself is the biggest blessing for me, which inspires and motivates me to go forward.

Last but certainly not least, I extend my sincere thanks to my friends and classmates, who in the moments of self-doubt and fatigue reminded me of my capabilities. Thanks to them for cheering me on and providing me with the help and support whenever I needed it.

QURAT UL AIN
Bahria University
E-8, Islamabad, Pakistan

December 2025

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
BWS	Biomedical Waste Segregation
CNN	Convolutional Neural Network
CV	Computer Vision
FPS	Frames Per Second
FYP	Final Year Project
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
IP	Internet Protocol
LAN	Local Area Network
ML	Machine Learning
mAP	Mean Average Precision
NMS	Non-Maximum Suppression
PCB	Printed Circuit Board
RAM	Random Access Memory
REST	Representational State Transfer
RGB	Red Green Blue
RNN	Recurrent Neural Network
ROI	Region of Interest
RTOS	Real-Time Operating System
SD	Secure Digital
SSD	Single Shot Multibox Detector
TCP	Transmission Control Protocol
USB	Universal Serial Bus
UV	Ultraviolet
YOLO	You Only Look Once (Object Detection Algorithm)

Chapter 1

Introduction

1.1 Autonomous Biomedical Waste Segregation Robot

Biomedical waste refers to any waste that is generated during the diagnosis, treatment, or immunization of humans or animals, or during related research and production activities [1]. The management of such waste has become a global concern due to its hazardous composition and potential to cause infectious diseases and environmental hazards. The global expansion of healthcare services, coupled with the increasing complexity of medical procedures, has led to a proportional increase in the volume and complexity of biomedical waste, making efficient segregation and disposal an urgent necessity [2].

Conventional biomedical waste segregation practices rely heavily on manual processes, where workers physically handle and sort waste into different categories. This approach is not only labor-intensive and exposes personnel to risks such as needle prick injuries and pathogen exposure, but also suffers from inefficiencies, human error, and inconsistencies. These shortcomings can compromise both public health and environmental safety while violating the regulatory standards governing biomedical waste management.

Advancements in artificial intelligence (AI), computer vision, and robotics present an opportunity to address these limitations through automation. By integrating intelligent visual recognition with robotic manipulation, waste segregation can be performed in a manner that is faster, safer, and more accurate. Moreover, coupling the system with cloud connectivity and predictive analytics can enable real-time monitoring, proactive management, and compliance tracking for healthcare institutions.

In addition to reducing health risks and improving operational efficiency, automated biomedical waste segregation systems also contribute to better compliance with environmental and health regulations. Many countries have strict guidelines regarding the treatment and disposal of hazardous medical waste, and non-compliance can result in significant legal and financial penalties for healthcare institutions. An AI-powered robotic system can ensure consistent adherence to these standards by accurately identifying and categorizing waste in real time. Moreover, the system is equipped with data logging capabilities to provide traceability and analytics for waste management

audits. This not only supports transparency and accountability, but also enables continuous improvement in waste handling practices through data-driven decision-making.

1.2 Project Overview

This project presents the development of an Autonomous Biomedical Waste Segregation Robot, designed to automatically classify and segregate biomedical waste into three primary categories: infectious, sharp, and noninfectious. The intelligent system is designed to detect, classify, disinfect, and segregate biomedical waste with minimal human intervention. The solution leverages computer vision, robotic automation, cloud integration, and predictive analytics to create an end-to-end waste management framework.

1. An ESP32-based camera streams video frames over HTTP to the processing node.
2. A YOLOv8-based deep learning model performs object detection and classification of biomedical waste into infectious, sharp, and non-infectious categories.
3. A simulated Franka Panda robotic arm, implemented in PyBullet, executes pick-and-place operations, including a UV sterilization step before waste disposal.
4. A Firestore database logs detected items, segregation actions, predictions, and alerts for long-term analysis and traceability.
5. A Streamlit-based IoT dashboard visualizes bin status, waste distribution, alerts, and reports in real time.

Through this integration, the system performs continuous biomedical waste segregation, sterilization, and monitoring autonomously. The dashboard further supports proactive management by predicting bin fill levels and generating alerts before overflow occurs.

The system is designed to operate autonomously, with minimal human intervention, making it especially useful in high-risk or resource-constrained healthcare settings.

1.3 Problem Description

Biomedical waste management poses significant operational and safety challenges due to its hazardous nature and the necessity for strict handling protocols. Manual segregation processes face several issues:

- **Health Hazards:** Waste handlers are frequently exposed to infectious agents, sharps, and contaminated materials.
- **Human Error:** Misclassification can lead to improper disposal, cross-contamination and regulatory noncompliance.

- **Operational Inefficiency:** Manual segregation requires significant manpower, time, and cost.
- **Inconsistent Practices:** Especially in developing countries, waste segregation often lacks standardization and oversight.

An automated system capable of intelligent waste recognition, robotic handling, and real-time monitoring can mitigate these issues, ensuring safety, standardization, and traceability in waste management operations.

1.4 Project Objectives

The primary goal of this project is to design and develop a functional prototype that is capable of performing safe, efficient, and intelligent biomedical waste segregation. The specific objectives include:

- To develop an AI-based classification system capable of identifying infectious, sharp, and non-infectious biomedical waste using a camera and trained machine learning model.
- To simulate a robotic arm capable of picking, disinfecting, and placing waste items in designated bins.
- To integrate UV sterilization simulation for safe disinfection prior to disposal.
- To create a real-time monitoring dashboard using IoT technologies for visualizing system status, waste statistics, and performance logs.
- To implement a prediction and alert module that estimates bin fill levels and notifies users before reaching capacity.
- To ensure that the system design is modular, scalable, and potentially deployable in healthcare environments.

1.5 Project Scope

The system focuses on automating the complete cycle of biomedical waste segregation, from image acquisition to waste classification, robotic manipulation, disinfection, logging, prediction, and visualization. The implementation scope includes:

- **Included:**
  - Training of a computer vision model (YOLOv8) using a biomedical waste image dataset.
  - Integration of an AI model with a camera for real-time detection.

  - Robotic simulation of pick, sterilize, and drop operations.
  - Development of an IoT-based web dashboard for system monitoring.
  - Predictive alert generation for bin capacity management.

- **Excluded:**
  - Development of industrial-grade robotic hardware.
  - Compliance testing with international biomedical waste disposal standards.
  - Waste transportation and final disposal processes.

This project lays the groundwork for a potential real-world application, focusing on technological feasibility and proof-of-concept implementation rather than full-scale deployment.

1.5.1 Methodology

The proposed Autonomous Biomedical Waste Segregation Robot follows a modular and layered development methodology that integrates computer vision, robotic simulation, and cloud-based monitoring. Initially, a labelled biomedical waste image dataset was prepared and annotated, followed by training a YOLOv8 deep learning model for real-time waste classification. The trained model processes live visual input from an ESP32 camera module to identify waste categories such as infectious, sharp, and non-infectious waste. Based on the classification result, a simulated robotic arm executes automated pick-and-place operations through a predefined segregation workflow that includes ultraviolet sterilization prior to disposal. Detection results, bin status updates, and operational logs are transmitted to a cloud-based Firestore database, enabling real-time visualization, historical analysis, and predictive monitoring through a Streamlit-based web dashboard. System testing and evaluation are conducted to validate functional correctness, performance, reliability, and usability of the integrated system.

1.6 Significance of the Study

The proposed system contributes to the broader goal of improving healthcare waste management through automation and intelligence. By combining AI-driven perception, robotic control, and IoT-enabled monitoring, the project demonstrates how biomedical waste management can transition from manual, error-prone methods to autonomous, data-driven, and safe operations. The system enhances occupational safety, promotes regulatory compliance, and supports environmental sustainability by ensuring accurate segregation and sterilization of hazardous materials. The outcomes of this study are particularly beneficial for hospitals, diagnostic laboratories, and healthcare facilities operating in high-risk or resource-constrained environments.

1.7 Summary

This chapter introduced the growing concern of biomedical waste and the serious risks it poses to both human health and the environment. It highlighted how current manual segregation methods are not only risky and inefficient but also prone to error. To tackle these challenges, the idea of an AI-powered robotic system was proposed that can automatically identify and sort different types of biomedical waste. The chapter gave an overview of the project, its key goals, and what it aims to achieve within a realistic scope. Overall, it sets the foundation for building a smart, safer, and more efficient approach to biomedical waste management.

Chapter 2

Literature Review

A literature review provides a structured and critical examination of existing knowledge related to biomedical waste management, computer vision based classification, autonomous robotic handling, ultraviolet disinfection systems and Internet of Things enabled monitoring. This chapter evaluates previous research to identify strengths, limitations, and opportunities that inform the design of the Autonomous Biomedical Waste Segregation Robot. The review positions the project within the broader context of intelligent healthcare waste management and highlights the research gaps that the system aims to address.

2.1 Biomedical Waste Classification Using Deep Learning

Several studies have applied deep learning techniques for the automated identification of biomedical waste. Early work such as [3] utilized a ResNeXt architecture to classify eight categories of medical waste with approximately ninety seven percent accuracy. Although effective in visual recognition, this study did not extend to robotic manipulation or sterilization. Similarly, [4] employed EfficientNet with transfer learning for classifying masks, gloves, needles, and syringes, achieving high accuracy but limiting the contribution to classification alone.

More recent advances introduced hybrid architectures. For instance, [5] proposed a DenseNet plus capsule network approach that improved spatial feature representation and recognition robustness. Other work focused on general waste rather than biomedical categories. For example, [6] integrated CNN based classification with IoT sensing for general waste categorization but did not consider hazardous biomedical classes such as sharps or infectious materials.

Overall, the literature demonstrates substantial progress in waste classification, yet it seldom addresses end to end biomedical waste processing or integration with robotic actuation systems.

2.2 Waste Segregation Through Robotic Systems

A number of studies explored robotic handling for waste segregation. For instance, [7] developed a YOLO based robotic sorting system using an Arduino controlled arm combined with a UV C exposure step. While this work aligns closely with biomedical waste workflows, it relied on limited hardware and lacked cloud based reporting or predictive analytics.

Other works such as [8] demonstrated robotic pick and place operations for general dry waste, focusing on motion planning rather than biomedical applications. Additionally, [9] implemented a conveyor based waste sorter using classical image processing, but the system lacked generalization for irregular biomedical waste items.

Robotic sorting research generally concentrates on mechanical actuation and rarely incorporates physics based simulation environments. Very few studies employ realistic simulators such as PyBullet for controlled robotic manipulation, collision filtering, or object attachment. This represents a significant gap that the present system addresses. Simulation-based environments enable safe experimentation, repeatability, and controlled evaluation of robotic behaviors, which are challenging to achieve consistently in physical prototypes.

2.3 Ultraviolet Disinfection Systems

Ultraviolet disinfection has been widely studied in healthcare settings. A comprehensive review by [10] demonstrated the effectiveness of UV robots in neutralizing surface pathogens and examined operational factors such as exposure duration, lamp efficiency and geometrical constraints. Similarly, [11] reported on hospital deployment of UV C cleaning robots and identified practical challenges including shadow zones and safety concerns.

While UV disinfection is well established for environmental cleaning, its integration within automated waste handling systems remains limited. Existing works focus largely on room sanitization rather than item level sterilization, and there is limited research combining UV exposure with robotic waste handling workflows. Item-level UV sterilization is particularly relevant for biomedical waste due to the high risk of pathogen transmission during handling and disposal.

2.4 IoT Enabled Waste Monitoring and Analytics

IoT based monitoring systems have become increasingly prevalent in waste management. For example, [12] proposed a deep learning and IoT framework for tracking biomedical waste disposal, although the work lacked robotic manipulation or sterilization. Likewise, [13] developed a smart domestic waste segregation system combined with Android notifications, and [14] introduced an edge based classifier for smart bins to reduce latency, yet neither system incorporated predictive modeling or biomedical waste handling.

IoT literature primarily addresses sensing and monitoring but often ignores predictive analytics, autonomous robotics or safety critical sterilization processes.

2.5 Chronological Development of Relevant Research Domains

The development of biomedical and general waste automation technologies has progressed through distinct research phases. Each phase introduced specific innovations that contributed toward the capabilities required in modern autonomous systems. This section outlines these chronological developments and relates them to the studies reviewed earlier.

2.5.1 Early Computer Vision Based Waste Classification Era

Initial research focused on developing reliable computer vision models for waste classification. Studies such as [3] and [4] demonstrated that deep learning approaches could accurately identify biomedical items. These works established visual perception as the foundation of automated waste systems, although they did not integrate robotic handling or disinfection.

2.5.2 Transition to Hybrid Models and Advanced Feature Learning

Advancements in feature representation led to hybrid models such as the DenseNet plus capsule network approach of [5]. This era improved recognition robustness but remained limited to perception tasks without addressing the mechanical automation required for complete waste segregation workflows.

2.5.3 Expansion into Robotic Sorting and Mechanical Automation

Subsequent research extended classification to mechanical actuation. Works such as [8] and [9] implemented robotic and conveyor based sorting. Although these systems demonstrated automation potential, they primarily targeted general waste rather than biomedical waste. The study by [7] represented an important step towards biomedical applications, though constrained by limited hardware integration and lack of simulation.

2.5.4 Emergence of UV Based Disinfection Robotics

Parallel developments in healthcare robotics explored UV based disinfection. Studies such as [10] and [11] provided operational insights into UV C systems but focused on room or surface sanitization rather than automated waste handling. This research contributed essential sterilization knowledge relevant to biomedical waste systems.

2.5.5 Introduction of IoT Enabled Monitoring and Predictive Analytics

More recent trends emphasized IoT connectivity, cloud logging and remote monitoring, as seen in [12, 13, 14]. These systems enhanced situational awareness but lacked integration with robotics, sterilization or predictive analytics. This era highlighted the need to bridge sensing systems with physical automation for complete biomedical waste workflows.

After these chronological developments, a comparative summary of all reviewed studies is presented in Table 2.1.

Table 2.1: Comparison of Related Studies on Biomedical Waste Classification, Robotics and IoT Systems

Study	**Domain Focus**	**Key Contributions**	**Limitations**
Zhou et al. (2022)	Medical waste classification	High accuracy for eight biomedical waste classes	No robotic handling, no sterilization, no real-time workflow integration
Akkajit and Sukkuea (2025)	Biomedical item classification	High precision on masks, syringes, gloves and vials	Classification only, no segregation or IoT integration
Van den Broek et al. (2025)	Medical waste recognition	Improved F1 through capsule representations	Does not address robotic manipulation or sterilization
Rakshita et al. (2025)	Biomedical waste sorting prototype	Combines detection, hardware sorting and UV exposure	Hardware dependent, lacks simulation, cloud logging and prediction engine
Li et al. (2024)	General robotic waste sorting	Demonstrates pick and place automation	No biomedical focus, no UV sterilization, no cloud analytics
Ahmed et al. (2023)	General waste sorting	Operational mechanical sorting prototype	Weak generalisation, no AI detection, no biomedical classes
Mehta et al. (2022)	UV disinfection robots	Provides sterilization timing and protocol insights	No waste classification or robotic segregation
Diab El Schahawi et al. (2021)	Hospital UV cleaning robots	Real world UV usage challenges documented	Focus on room disinfection, not waste handling
Haritha et al. (2025)	IoT based biomedical waste monitoring	Tracking and monitoring framework	Lacks robotic handling, sterilization or prediction engine
Vimal et al. (2023)	Smart domestic waste management	Real-time notification and sorting aid	Not biomedical, lacks robotics, UV and prediction
Park et al. (2020)	Edge based smart bin classification	Optimised edge inference for bins	No robotic arm, no sterilization, no predictive analytics

These progressive developments collectively highlight the fragmented nature of existing solutions, where individual capabilities evolved independently rather than as unified autonomous systems.

2.6 Research Gaps and Motivation

A critical review of the literature highlights several gaps that motivate the development of the Autonomous Biomedical Waste Segregation Robot.

- Limited research focuses specifically on biomedical waste classes that include infectious waste, sharp waste and non-infectious waste. Most work targets general waste.
- Few systems combine computer vision based classification with robotic pick and place operations in a closed loop workflow.
- Integration of UV based sterilization as an intermediate stage in automated waste handling is rarely addressed.
- Cloud connected analytics, predictive modeling and timeline logging for waste segregation are not commonly integrated with robotic systems.
- Simulation based robotic manipulation using physics engines such as PyBullet is scarcely explored despite its safety and reproducibility advantages.
- End to end pipelines that include detection, actuation, sterilization, logging, prediction and dashboard visualization are nearly absent in the reviewed literature.

These gaps collectively motivate the design of an integrated system that unifies perception, manipulation, sterilization and intelligent monitoring within a single autonomous framework.

2.7 Summary

The literature demonstrates strong progress in isolated domains such as biomedical waste classification, basic robotic sorting and UV disinfection. However, there remains a lack of holistic systems that integrate these technologies into a unified autonomous workflow. The reviewed works seldom address both safety critical aspects such as sterilization and operational elements such as prediction and real-time monitoring. This project fills these gaps by combining computer vision, robotic simulation, ultraviolet sterilization, cloud analytics and interactive dashboards into a modular and autonomous biomedical waste segregation system. The next chapter defines the system requirements that guide this integrated design.

Chapter 3

Requirement Specifications

Biomedical waste segregation is a critical component of hospital safety and environmental hygiene. Traditional methods rely heavily on manual intervention, which introduces risks of infection, inefficiency, and human error. This chapter defines the requirements of the proposed system, including an analysis of the existing biomedical waste management systems, their drawbacks, and the improvements introduced by the Autonomous Biomedical Waste Segregation Robot. It further specifies the user and functional requirements, describes the system's operational behavior through use cases, and outlines the non-functional requirements and quality constraints that govern the development of the system.

3.1 Existing System

Biomedical waste management in healthcare facilities has traditionally relied on manual processes for segregation, collection, and disposal. The existing biomedical waste management system involves the following key steps:

- Manual identification and segregation of biomedical waste into color-coded bins.
- Heavy dependence on human decision-making and attention to protocol.
- Physical handling of potentially hazardous materials by staff.
- Lack of real-time data or waste monitoring systems.

3.1.1 Limitations of the current system:

The existing system presents several critical drawbacks:

- **High Human Error Rate:** Manual classification of biomedical waste is prone to mistakes, which can result in improper segregation and cross-contamination of waste categories.

- **Health Hazards:** Direct handling of biomedical waste exposes healthcare personnel to infectious materials and sharp objects, increasing the risk of infections and injuries.
- **Inconsistent Segregation:** Different individuals may interpret waste types differently, leading to inconsistency in segregation practices.
- **Lack of Real-Time Monitoring:** The current waste management process lacks real-time visibility of bin fill levels or the overall waste segregation progress within hospital facilities.
- **Reactive Waste Management:** Waste disposal decisions are often made reactively after bins are full, rather than proactively predicting fill levels.
- **Data Inaccessibility:** Manual recording of waste statistics and reports makes it difficult to perform trend analysis or maintain compliance documentation efficiently.

These challenges are widely reported in healthcare waste management literature and international guidelines, where improper segregation is linked to increased occupational risk and environmental contamination [1, 2].

The absence of automation, intelligent classification, and proactive monitoring results in inefficiencies, safety risks, and non-compliance with biomedical waste disposal regulations.

3.2 Proposed System

The Autonomous Biomedical Waste Segregation Robot is designed to overcome the limitations of the existing manual system by integrating computer vision, robotics, IoT, and cloud-based analytics into a unified smart system. The system automates biomedical waste segregation, sterilization, monitoring, and prediction processes using intelligent decision-making algorithms.

3.2.1 Key Features of proposed system:

The system functions autonomously through a sequence of well-coordinated operations:

- Biomedical waste is captured in real-time using an ESP32 camera connected to the system.
- A deep learning model (YOLOv8) classifies the waste into predefined categories: infectious, sharp, or non-infectious.
- A simulated robotic arm (Franka Panda) manipulates, sterilizes, and disposes of waste items into the correct bins through virtual robotic operations in PyBullet.
- Cloud-based storage (Firestore) logs each detection and segregation event, maintaining a timeline of actions and bin fill levels.
- Predictive analytics estimate future bin capacities and generate alerts when thresholds are likely to be exceeded.

- A Streamlit-based dashboard provides real-time visualization, monitoring, and reporting for hospital administrators.

3.2.2 Advantages of the proposed system:

- **Improved Safety:** Reduces the risk of exposure to hazardous materials by limiting human interaction with biomedical waste.
- **Consistent and Accurate Sorting:** Machine learning ensures consistent classification based on training data, minimizing errors due to human subjectivity.
- **Real-time Monitoring:** Logging of all activities to an IoT-enabled dashboard enables real-time monitoring and traceability.
- **Scalable to physical systems:** The architecture is designed to be extensible toward physical deployment; however, real-world integration would require additional hardware validation, safety interlocks, and compliance testing.

The proposed system offers a modern, intelligent, and safe alternative to the conventional method of biomedical waste segregation. By leveraging automation, artificial intelligence and Internet of Things (IoT) capabilities, the system aims to improve accuracy, efficiency, and safety in biomedical waste management within healthcare facilities. It also enables data-driven decision making for waste management and compliance monitoring.

3.3 Requirement Specifications

The primary users of the Autonomous Biomedical Waste Segregation Robot are hospital staff, waste management administrators, and system supervisors responsible for biomedical waste handling and monitoring. The users require a system that provides secure access and transparent visualization of waste segregation operations. Each user must be able to log into the dashboard using authorized credentials to monitor waste processing activities in real time. The system must ensure secure authentication through the existing Firebase Authentication mechanism, restricting access only to registered personnel. Users should also have the ability to log out securely to maintain data privacy and prevent unauthorized access. Since user registration is handled by system administrators through Firebase's backend configuration, the system itself only requires login and logout functionalities from the user interface perspective.

From the operational standpoint, users require an intelligent system capable of detecting, classifying, and segregating biomedical waste autonomously. The system must accurately identify waste items into three primary categories i.e. infectious, sharp, and non-infectious, through computer vision–based detection. Once detected, the robotic simulation must execute pick-and-place operations to transfer each classified item to its respective disposal bin, ensuring proper segregation according to biomedical standards. During this process, the waste item should undergo

UV sterilization to minimize contamination risk before disposal. The user should not need to intervene in these automated tasks, but they must be able to visualize each step and verify segregation accuracy through the monitoring dashboard.

Furthermore, users require comprehensive visualization and decision-support capabilities through the integrated web dashboard. The system should provide real-time bin fill levels, display the current distribution of waste categories, and generate alerts when bins approach capacity limits. Predictive analytics should enable users to anticipate when a bin will require emptying, facilitating proactive waste management. Users should also be able to access historical logs of waste segregation activities through timeline views, and export structured reports for auditing and compliance purposes. These functionalities ensure that the system not only automates waste handling but also empowers users to make informed operational decisions through data-driven insights.

3.3.1 Functional Requirements

The following functional requirements represent the user's needs and expected system behaviors. Each requirement is described in terms of its input, process, output, and a brief description.

- **FR1: User Login**
  Input: User credentials i.e. email and password
  Process: The system authenticates user credentials against records stored in Firebase Authentication.
  Output: Access granted to the dashboard upon successful authentication or error message on failure.
  Description: The system shall allow only authorized users to sign in and access the web dashboard. The user must already be registered in the Firebase Authentication system by the administrator.

- **FR2: User Logout**
  Input: Logout request initiated by the user.
  Process: The system terminates the active user session securely and clears authentication tokens.
  Output: User redirected to the login page.
  Description: The system shall allow the authenticated user to securely exit the system, ensuring session integrity and preventing unauthorized access.

- **FR3: Waste Detection and Classification**
  Input: Real-time video stream from the camera.
  Process: YOLOv8 model detects and classifies waste items.
  Output: Waste categorized as infectious, non-infectious, or sharp.
  Description: The system shall utilize a YOLOv8 computer vision model to detect and classify biomedical waste into predefined categories: infectious, non-infectious, and sharp.

- **FR4: Robotic Waste Segregation**
  Input: Classified waste category from the detection module.
  Process: Robotic arm executes movement to place waste in the correct bin.
  Output: Waste successfully placed into the appropriate bin.
  Description: The system shall simulate the movement of a robotic arm to pick and place classified waste items into their respective bins based on AI classification results.

- **FR5: UV Sterilization**
  Input: Picked item brought to disinfection unit by robotic arm.
  Process: UV light unit activates for a predefined duration for each waste item before disposal.
  Output: Robotic arm takes the waste item to the disposal bin after sterilization completes
  Description: The system shall simulate UV-based sterilization of each waste item before disposal, ensuring contamination reduction. The exposure duration shall vary according to the waste category.

- **FR6: Dashboard Visualization**
  Input: Live data streams from Firestore such as bin status.
  Process: The system retrieves and processes real-time data streams from Firebase Firestore for visualization.
  Output: The system shall display a real-time interactive dashboard showing waste category distribution, bin fill levels, alerts and system status indicators **Description:** The system shall provide an interactive dashboard that visualizes bin status, category distribution, recent detections, alerts, and system activity logs retrieved from Firestore.

- **FR7: Bin Fill Level Tracking**
  Input: Disposed waste logged to database.
  Process: System calculates bin capacity, updates the Firestore database, and compares against threshold values.
  Output: Updated bin fill level displayed on dashboard with color-coded indicators.
  Description: The system shall track, calculate, and update the fill level of each bin dynamically after every waste disposal operation, displaying the updated status on the dashboard with visual indicators.

- **FR8: Data Logging and Timeline Creation**
  Input: Event data from all modules i.e. detection results, bin updates, user actions.
  Process: System stores event details in the system database.
  Output: Historical timeline of system activities accessible via dashboard.
  Description: The system shall log every operational event, including detections, classifications, and user activities, to maintain a chronological timeline for monitoring and analysis.

- **FR9: Predictive Analytics and Alerts**
  Input: Historical bin fill data and timestamps.
  Process: The predictive model analyzes historical bin fill data to forecast capacity thresholds.

Output: Visual warnings and alerts are displayed to the user.
Description: The system shall predict future bin fill levels using historical data and generate alerts if any bin is nearing capacity.

- **FR10: Report Generation and Export**
  Input: User request specifying time range or category and report format, i.e. CSV or Excel format.
  Process: System retrieves relevant logs, detections, and analytics data from Firestore and compiles structured reports.
  Output: Report downloaded in selected format.
  Description: The system shall enable authorized users to generate waste management reports based on historical data and export them in standard formats such as CSV or Excel for auditing and compliance purposes.

3.3.2 Non-Functional Requirements

The non-functional requirements define the quality attributes and constraints under which the system must operate. These are the requirements that do not directly impact the core functionality, but are essential for the system's performance, reliability, and usability.

- **NFR1: Internet Connectivity**
  The system shall require a stable internet connection for real-time video streaming, dashboard synchronization, and alert notifications. In case of connectivity loss, the dashboard shall display cached data and show an offline warning.
- **NFR2: Environmental Compatibility**
  The system should be operable within hospital environmental conditions (temperature, humidity, etc.).
- **NFR3: Hardware Compatibility**
  The software should be compatible with embedded hardware (e.g., ESP32 camera module).
- **NFR4: Security Requirements:** All data transactions between the Firebase database and Streamlit dashboard shall be secured using HTTPS and Firebase authentication.
- **NFR5: Usability Requirements:** The dashboard interface shall provide an intuitive and user-friendly experience with clear visualization, color-coded indicators, and simplified navigation suitable for hospital staff.
- **NFR6: Scalability Requirements:** The system architecture shall be designed to support the integration of multiple waste bins and hardware nodes, enabling expansion for large-scale hospital deployments.

3.4 Assumptions and Constraints

This section outlines the assumptions and constraints considered during the analysis and design of the Autonomous Biomedical Waste Segregation Robot. These factors define the operational boundaries of the system and influence design decisions.

3.4.1 Assumptions

The following assumptions are made for the proposed system:

- A stable internet connection is available to support real-time communication between the system, Firebase cloud services, and the web dashboard.
- The biomedical waste images used for training are correctly labeled and representative of real-world hospital waste.
- The robotic arm operates within a simulated environment and follows predefined kinematic constraints.
- Authorized users possess basic technical literacy to interact with the web-based dashboard.

3.4.2 Constraints

The system is subject to the following constraints:

- The robotic arm implementation is limited to simulation and does not include physical hardware validation.
- Detection accuracy is constrained by dataset quality, lighting conditions, and camera resolution.
- Real-time performance depends on available computational resources and network latency.
- The system is designed as a prototype and is not certified for clinical deployment.

3.5 Use Cases

3.5.1 Complete System Use Case Diagram

The complete system use case diagram highlights all the functionalities supported by Autonomous Biomedical Waste Segregation Robot System. It covers core functional requirements, represented as use cases, along with their associated actors.

Figure 3.1 illustrates the interaction between the system's key actors and its major functionalities derived from the functional requirements. The primary actor, Hospital Staff, interacts with the system to perform waste monitoring and management tasks such as logging in, monitoring real-time

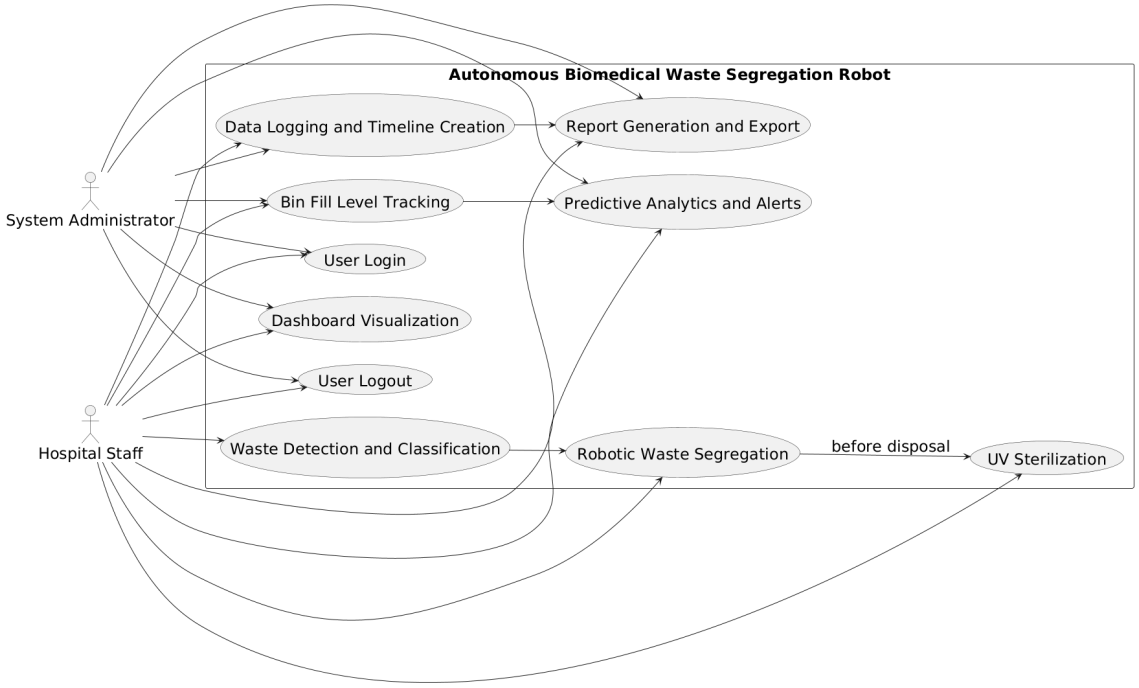


Figure 3.1: Full system Use Case Diagram of Autonomous Biomedical Waste Segregation Robot

waste detection and segregation, viewing dashboard analytics, tracking bin fill levels, generating reports, and receiving predictive alerts. Each of these actions corresponds to a defined functional requirement that supports biomedical waste segregation and monitoring through automation and computer vision.

The System Administrator serves as a secondary actor responsible for managing user access through Firebase Authentication and reviewing or exporting system-generated reports for compliance and auditing. Internally, the system executes automated processes such as waste detection using YOLOv8, robotic segregation based on classified categories, and UV sterilization before disposal. The dashboard aggregates all collected data through Firebase Firestore and provides visualization, predictive insights, and data traceability. Overall, this diagram provides a holistic view of how users and administrators interact with the system's intelligent modules to achieve autonomous, safe, and traceable biomedical waste management.

3.5.2 Use Case Modeling

The system use cases represent the interactions between users (hospital staff and administrators) and the system components. Each use case has been systematically derived from the corresponding functional requirements (FR1–FR10) to ensure traceability, completeness, and alignment with the system's intended behavior.

This section presents the detailed use cases corresponding to each functional requirement. For every functional requirement, a focused UML use case diagram fragment illustrates the relevant interaction between the actor(s) and the system. Each use case is introduced with a concise narrative description that elaborates on the system functionality and the involvement of the respective actors, and a structured table provides a detailed specification of each use case, outlining its actor(s), description, preconditions, postconditions, and flow of events.

3.5.2.1 UC01 - User Login

The *User Login* use case provides a secure authentication mechanism that ensures only authorized personnel can access the system's functionalities. Healthcare staff and system administrators must authenticate using pre-registered email and password credentials stored in Firebase Authentication. This use case acts as the entry point to the system, enforcing security and privacy of biomedical waste data. Upon successful authentication, the user gains access to all permitted dashboard components and operational modules, whereas invalid credentials prevent system entry. UC01 establishes the foundational trust boundary necessary for safe system operation. The above discussion is shown in Figure 3.2 and Table 3.1.

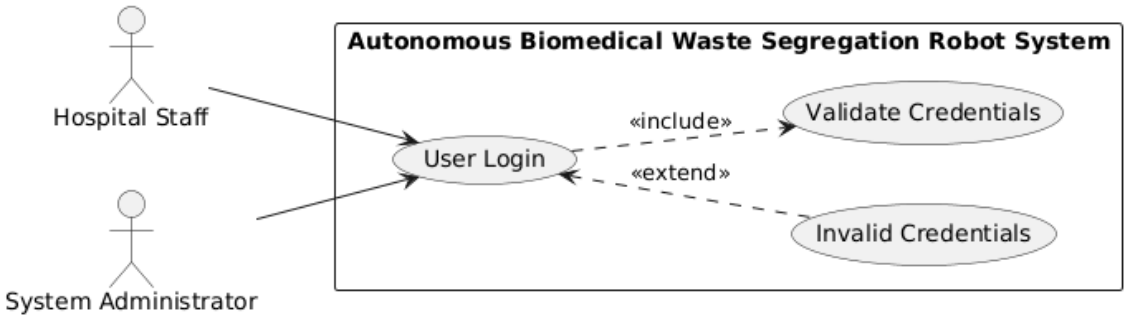


Figure 3.2: Use Case Diagram for User Login (UC01)

3.5.2.2 UC02 - User Logout

The *User Logout* use case ensures the safe termination of an authenticated session. Once the user completes system operations, this use case allows healthcare staff or administrators to explicitly sign out from the dashboard. Logging out prevents unauthorized access from unattended sessions and reinforces system security by clearing active tokens and associated session data. The logout process returns the interface to the login screen, ensuring that any subsequent access requires reauthentication. It maintains the integrity of user access control and minimizes security risks. By enforcing a proper logout flow, the system strengthens overall security and improves reliability for hospital operations. The above discussion is shown in Figure 3.3 and Table 3.2.

Table 3.1: Use Case Specification for UC01 - User Login

Use Case ID	UC01
Use Case Name	User Login
Actor(s)	Hospital Staff, System Administrator
Description	Allows authorized users to securely log into the system using Firebase Authentication.
Precondition	User must already be registered in Firebase Authentication.
Postcondition	User is granted access to the system dashboard upon successful authentication.
Trigger	User submits login credentials.
Main Flow	1. User opens the dashboard. 2. User enters email and password. 3. System verifies credentials via Firebase Authentication. 4. If valid, access is granted to the dashboard.
Alternative Flow	3a. If validation fails, system displays an error and prompts re-entry of credentials.
Exception Flow	1a. Internet connectivity failure prevents Firebase Authentication.

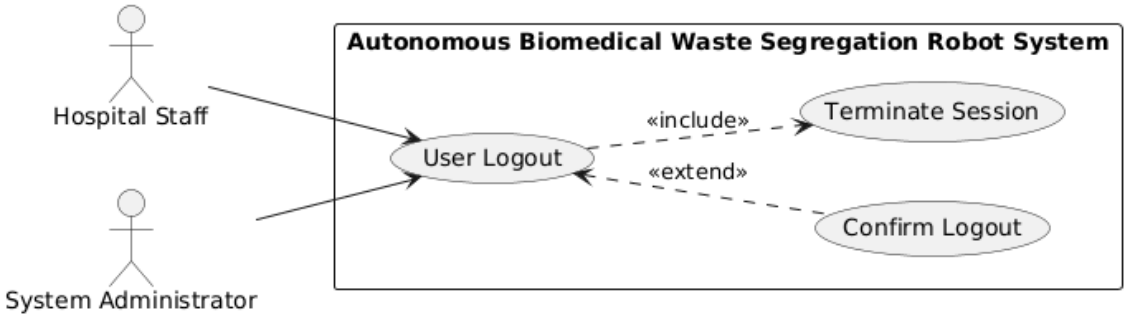


Figure 3.3: Use Case Diagram for User Logout (UC02)

3.5.2.3 UC03 - Waste Detection and Classification

The *Waste Detection and Classification* use case represents the core artificial intelligence component of the system. It utilizes the YOLOv8 computer vision model to detect biomedical waste and classify it into predefined categories such as infectious, non-infectious, and sharps. The process is initiated when the detection module receives a frame from the camera, after which the model performs inference and generates both class labels and confidence scores. The classification output serves as the primary decision-making input for subsequent use cases such as robotic segregation

Table 3.2: Use Case Specification for UC02 - User Logout

Use Case ID	UC02
Use Case Name	User Logout
Actor(s)	Hospital Staff, System Administrator
Description	Logs out the current user and ends the session securely.
Precondition	User must be logged in.
Postcondition	Session is terminated and user is redirected to the login page.
Trigger	User clicks the logout button.
Main Flow	1. User selects the logout option. 2. System clears session data. 3. User is redirected to the login page.
Alternative Flow	None.
Exception Flow	2a. Network interruption delays logout confirmation.

(UC04) and sterilization (UC05). It forms the analytical backbone of the Autonomous Biomedical Waste Segregation Robot, enabling consistent and automated waste identification. By automating classification, the system reduces human errors that may occur during manual segregation. This not only increases efficiency but also improves safety for healthcare workers and reduces risks of contamination. The above discussion is shown in Figure 3.4 and Table 3.3.

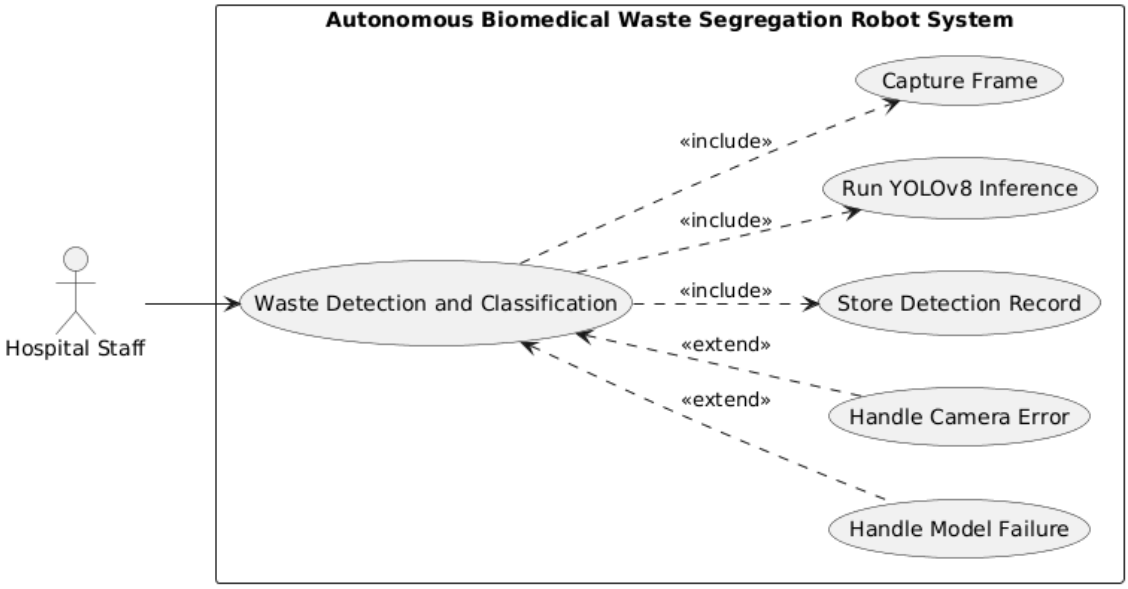


Figure 3.4: Use Case Diagram for Waste Detection and Classification (UC03)

Table 3.3: Use Case Specification for UC03 - Waste Detection and Classification

Use Case ID	UC03
Use Case Name	Waste Detection and Classification
Actor(s)	Hospital Staff
Description	Uses camera input and YOLOv8 model to detect and classify biomedical waste.
Precondition	ESP32 camera must be connected and operational.
Postcondition	Waste items are labeled by category and forwarded to the segregation module.
Trigger	System initiates detection upon execution.
Main Flow	1. Camera captures real-time frames. 2. YOLOv8 performs object detection and classification. 3. Classified results are stored in Firestore.
Alternative Flow	2a. If an object is unrecognized, it is labeled as "Unclassified Waste."
Exception Flow	1. Model is not loaded or fails during inference. 2. Camera loses feed.

3.5.2.4 UC04 - Robotic Waste Segregation

The *Robotic Waste Segregation* use case simulates the movement and decision-making capabilities of a robotic arm responsible for sorting biomedical waste. Once the classification module (UC03) identifies the waste type, the segregation routine is triggered automatically. The robotic arm simulation calculates and executes an appropriate movement trajectory to pick the detected item and place it into the correct bin. If model confidence is low, the item is instead directed to a designated Review Bin for manual evaluation. By streamlining the segregation workflow, it ensures accurate and repeatable handling of biomedical waste within the simulated environment. The above discussion is shown in Figure 3.5 and Table 3.4.

3.5.2.5 UC05 - UV Sterilization

The *UV Sterilization* use case simulates the ultraviolet sterilization process applied to biomedical waste before final disposal. When the robotic arm deposits an item into the sterilization unit, the UV chamber activates for a predefined exposure period to eliminate microorganisms and surface pathogens. The simulation tracks the sterilization cycle and ensures that only sterilized items are transferred to the final disposal bin. If the cycle is interrupted or prematurely terminated, the system automatically restarts the process or logs the error. It enhances safety by modeling an important

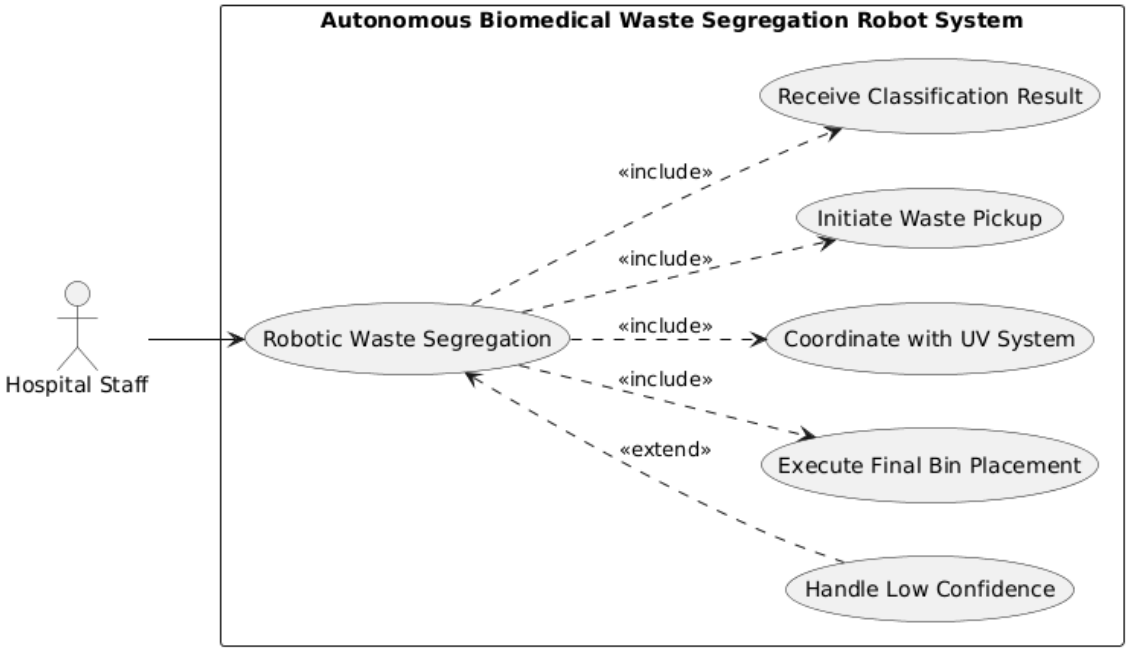


Figure 3.5: Use Case Diagram for Robotic Waste Segregation (UC04)

real-world disinfection step within the automated pipeline. The above discussion is shown in Figure 3.6 and Table 3.5.

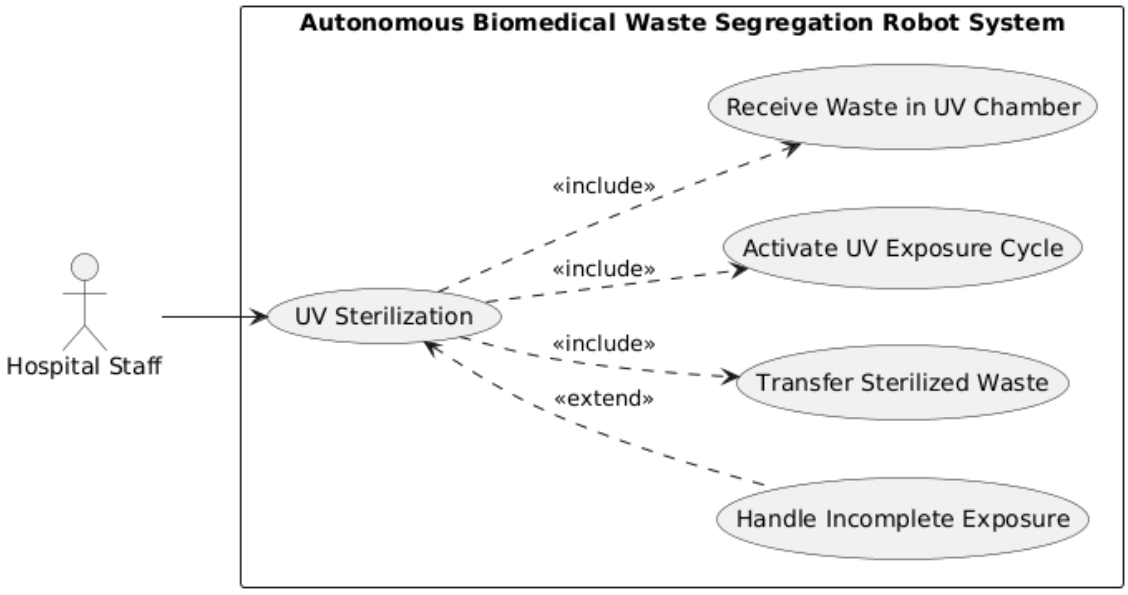


Figure 3.6: Use Case Diagram for UV Sterilization (UC05)

3.5.2.6 UC06 - Dashboard Visualization

The *Dashboard Visualization* use case enables real-time monitoring of the system through an interactive interface developed using Streamlit. Once the user accesses the dashboard, live data

Table 3.4: Use Case Specification for UC04 - Robotic Waste Segregation

Use Case ID	UC04
Use Case Name	Robotic Waste Segregation
Actor(s)	Hospital Staff
Description	Simulates robotic arm movement to segregate biomedical waste based on the classification result from the YOLOv8 model.
Precondition	Waste type must be classified and arm must be initialized.
Postcondition	Waste item is sorted into its appropriate bin.
Trigger	Classification result is received from the detection module.
Main Flow	1. Classification module sends waste type to robotic arm controller. 2. Robotic arm simulation executes movement to pick up waste item. 3. Waste item undergoes UV sterilization process. 4. Arm places the item in the appropriate final bin.
Alternative Flow	3a. If classification confidence is low, item is sent to the "Review Bin."
Exception Flow	2b. Simulation or control module failure interrupts segregation.

is fetched from Firestore and displayed through charts, tables, indicators, and status widgets. Relevant metrics such as waste counts, bin levels, classification events, and system activities are continuously updated to reflect the current operational state. The dashboard provides an intuitive overview for both healthcare staff and administrators, supporting situational awareness and informed decision-making. It acts as the system's primary visualization layer, offering insights into the waste management workflow. The above discussion is shown in Figure 3.7 and Table 3.6.

3.5.2.7 UC07 - Bin Fill Level Tracking

The *Bin Fill Level Tracking* use case is responsible for continuously monitoring the fill status of each biomedical waste bin using disposal event data stored in Firestore. Whenever a new disposal event is logged, the system retrieves the corresponding disposal records and computes the updated fill percentage for the affected bin. The new fill value is then written back to Firestore and immediately reflected on the dashboard. Each bin is visualized with both a numeric percentage and a color-coded status indicator, where typical thresholds classify bins as green (safe/low fill), yellow (moderate/approaching capacity), and red (high/critical fill). When the bin fill level crosses a defined threshold (e.g., 80%), the dashboard automatically highlights the bin in a warning state

Table 3.5: Use Case Specification for UC05 - UV Sterilization

Use Case ID	UC05
Use Case Name	UV Sterilization
Actor(s)	Hospital Staff
Description	Simulates sterilization of waste using UV light before final disposal.
Precondition	Waste must be delivered by robotic arm to sterilization unit.
Postcondition	Sterilized waste is transferred to disposal bin.
Trigger	Robotic arm delivers item to UV station.
Main Flow	1. Waste reaches the UV chamber. 2. UV light activates for preset exposure time. 3. Process completes and item is transferred to designated bin.
Alternative Flow	2a. System re-runs the exposure cycle if exposure is interrupted (simulated).
Exception Flow	2b. Power or simulation error halts UV process.

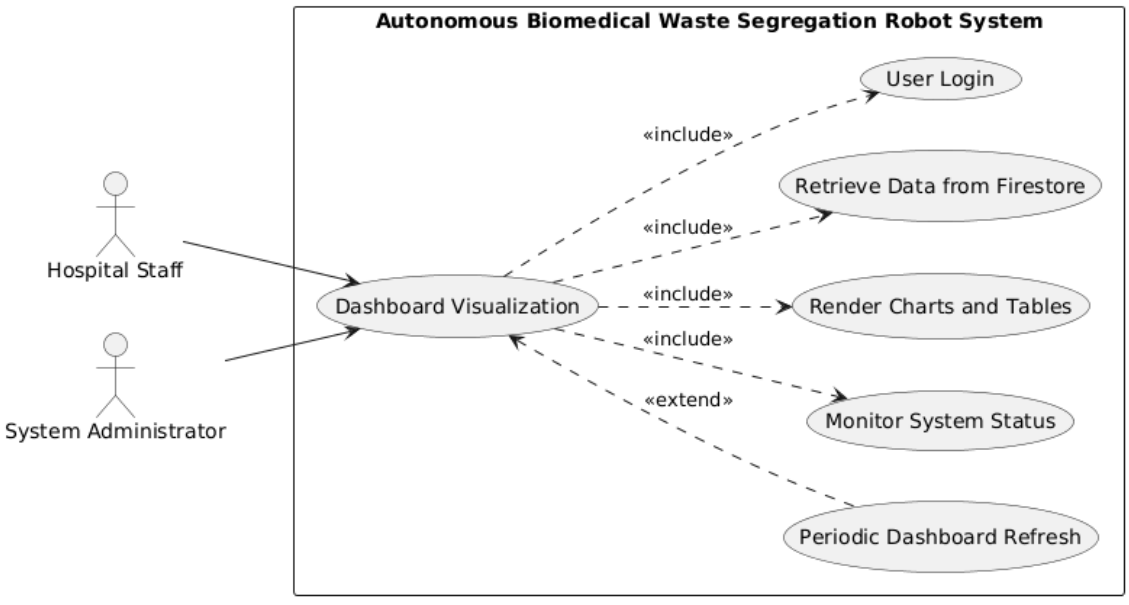


Figure 3.7: Use Case Diagram for Dashboard Visualization (UC06)

to draw the attention of hospital staff and administrators. This keeps track of bin utilization and fill levels in real-time using Firestore data rather than hardware sensors. This ensures timely intervention, prevents overflow, and supports safer biomedical waste handling practices. The above discussion is shown in Figure 3.8 and Table 3.7.

Table 3.6: Use Case Specification for UC06 - Dashboard Visualization

Use Case ID	UC06
Use Case Name	Dashboard Visualization
Actor(s)	Hospital Staff, System Administrator
Description	Displays real-time biomedical waste data and statistics via Streamlit dashboard.
Precondition	User must be logged in.
Postcondition	Dashboard updates automatically in real-time.
Trigger	User navigates to dashboard.
Main Flow	1. System retrieves live data from Firestore. 2. Data is visualized through Streamlit charts and tables. 3. User monitors system status.
Alternative Flow	2a. Dashboard refreshes periodically to reflect updated data.
Exception Flow	1a. Firestore connection failure causes temporary display lag.

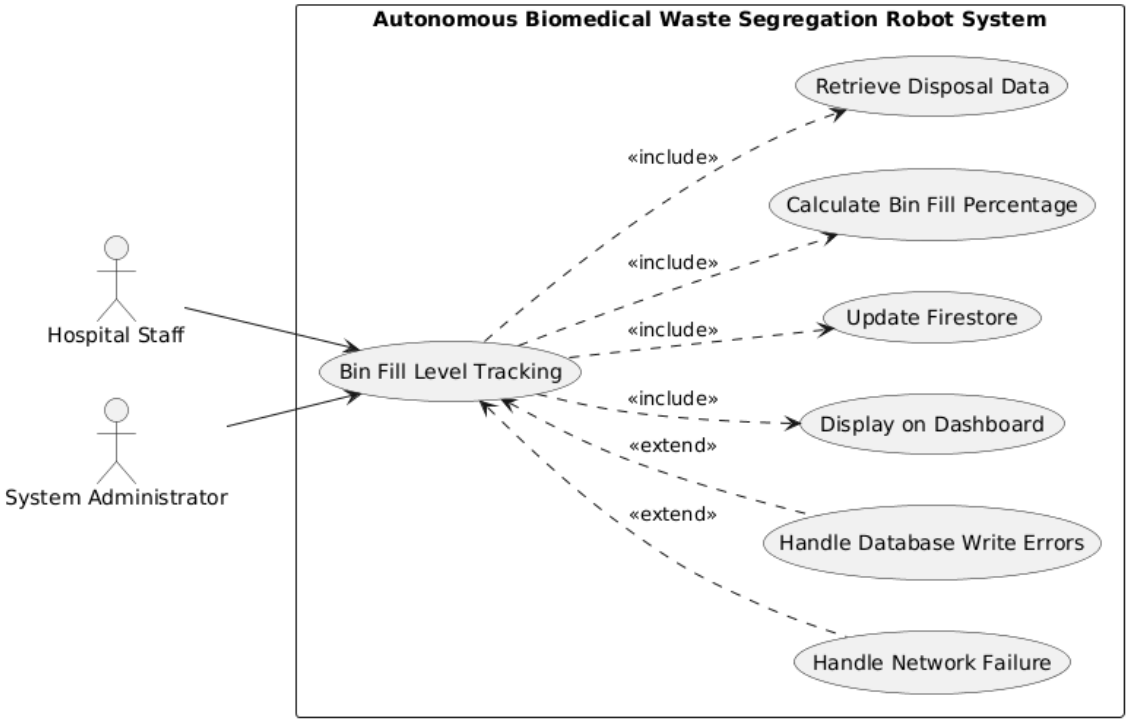


Figure 3.8: Use Case Diagram for Bin Fill Level Tracking (UC07)

Table 3.7: Use Case Specification for UC07 - Bin Fill Level Tracking

Use Case ID	UC07
Use Case Name	Bin Fill Level Tracking
Actor(s)	Hospital Staff, System Administrator
Description	Monitors and updates bin fill levels using disposal event data in Firestore.
Precondition	Disposal data must exist in Firestore.
Postcondition	Updated bin fill percentage shown on dashboard.
Trigger	A new disposal event is logged.
Main Flow	1. System retrieves disposal data. 2. Calculates bin fill percentage. 3. Updates Firestore record. 4. Dashboard displays the updated fill level. 5. Color indicators are automatically updated.
Alternative Flow	4a. Display cached data when network connection fails.
Exception Flow	3a. Data write failure delays dashboard update.

3.5.2.8 UC08 - Data Logging and Timeline Creation

The *Data Logging and Timeline Creation* use case is responsible for maintaining a complete and traceable history of system activity across all modules. Whenever a relevant event occurs such as waste detection, segregation, disposal, or bin status update, the corresponding module triggers the creation of a new log entry. The system captures key metadata (for example, timestamp, event type, waste category, and bin information) and writes this information into a dedicated Firestore collection. These records are stored in chronological order and used to generate a timeline view that can be reviewed by hospital staff and system administrators for monitoring, auditing, and analysis purposes. This use case strengthens system transparency, supports incident investigation, and contributes to compliance with biomedical waste handling guidelines. The above discussion is shown in Figure 3.9 and Table 3.8.

3.5.2.9 UC09 - Predictive Analytics and Alerts

The *Predictive Analytics and Alerts* use case enables the system to move from reactive monitoring to proactive decision support. At periodic intervals, the system retrieves historical bin fill data stored in Firestore and applies a prediction algorithm to estimate future fill levels for each bin. Based on these predictions, the system identifies bins that are expected to exceed a defined threshold

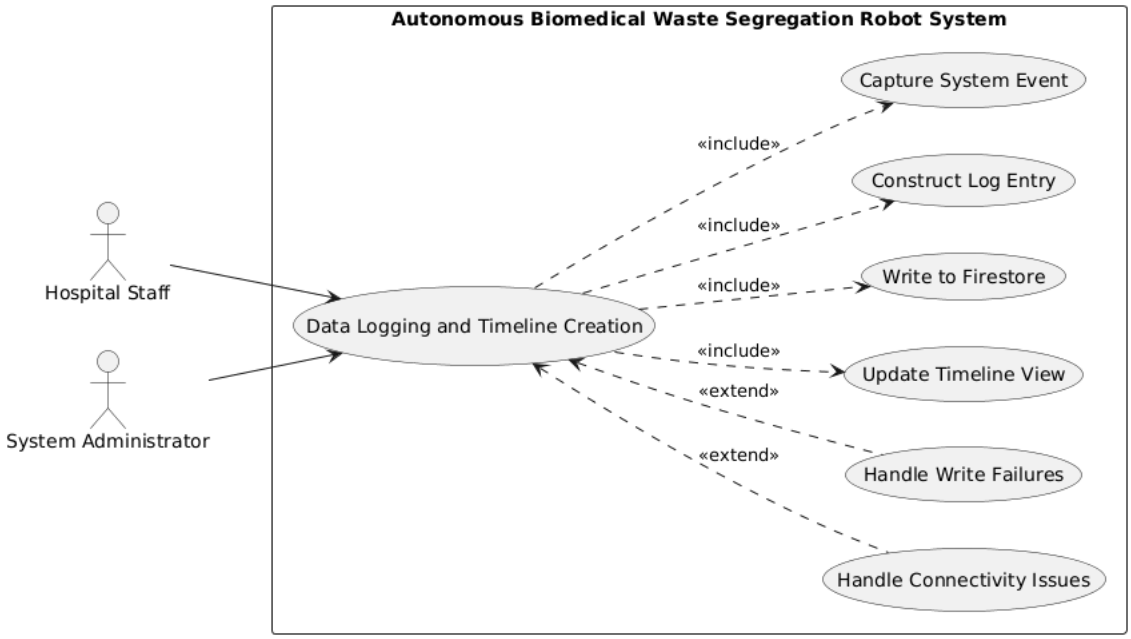


Figure 3.9: Use Case Diagram for Data Logging and Timeline Creation (UC08)

Table 3.8: Use Case Specification for UC08 - Data Logging and Timeline Creation

Use Case ID	UC08
Use Case Name	Data Logging and Timeline Creation
Actor(s)	Hospital Staff, System Administrator
Description	Logs all system operations into Firestore to maintain traceability and generate a timeline.
Precondition	System modules must be operational.
Postcondition	All events are stored chronologically in Firestore.
Trigger	Any system event (e.g., detection, disposal).
Main Flow	1. A system module generates a new event. 2. Constructs a log entry with relevant metadata. 3. Write entry to Firestore collection. 4. Updates timeline view.
Alternative Flow	2a. If connectivity is unstable, the system retries the Firestore write operation and reports logging failure to the dashboard.
Exception Flow	2b. Firestore write failure causes partial log loss.

within a given time window and generates corresponding alerts. These alerts are displayed on the

dashboard for both hospital staff and system administrators, helping them anticipate capacity issues before they occur. When insufficient or sparse historical data is available, prediction accuracy may be reduced, but the system can still provide approximate insights to support operational planning. The above discussion is shown in Figure 3.10 and Table 3.9.

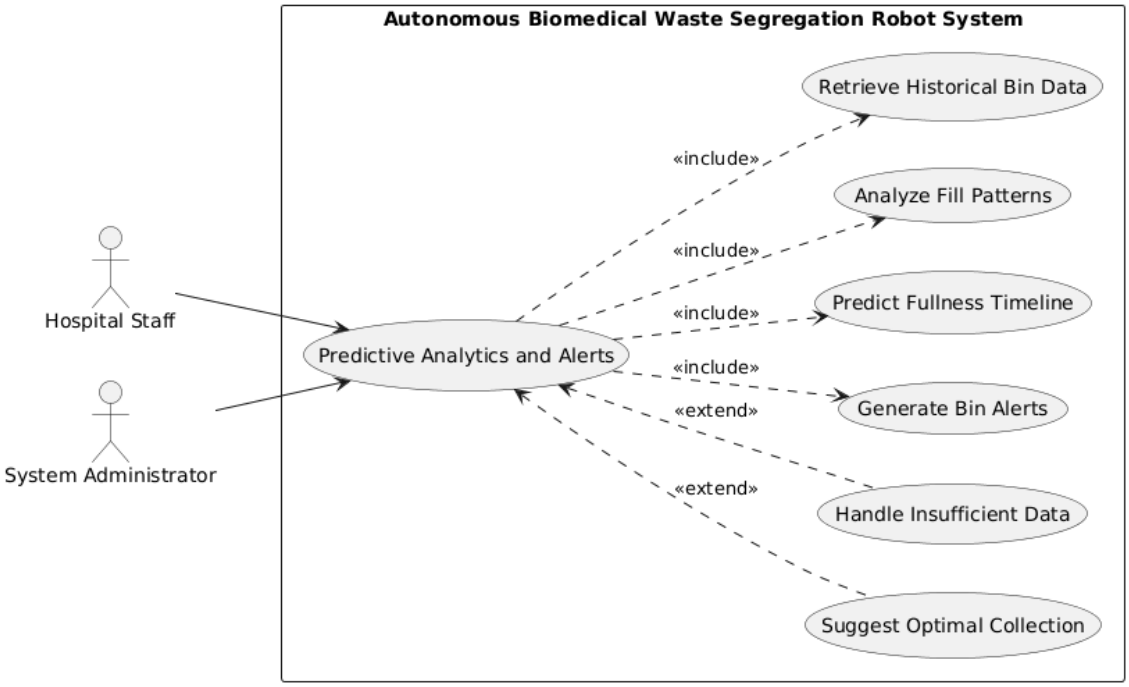


Figure 3.10: Use Case Diagram for Predictive Analysis and Alerts (UC09)

3.5.2.10 UC10 - Report Generation and Export

The *Report Generation and Export* use case enables hospital staff and system administrators to view historical waste records for a selected bin in tabular form and export the filtered data as a structured report for documentation and analysis. After logging into the system, the user initiates report creation by selecting the bin. The system then retrieves the corresponding records from Firestore and compiles them into a structured report view. Once generated, the report is made available for review and can be exported in a convenient format, such as CSV or Excel, for further analysis or archival purposes. This use case supports regulatory compliance, internal auditing, and performance monitoring by enabling evidence-based documentation of waste handling activities. The above discussion is shown in Figure 3.11 and Table 3.10.

Table 3.9: Use Case Specification for UC09 - Predictive Analytics and Alerts

Use Case ID	UC09
Use Case Name	Predictive Analytics and Alerts
Actor(s)	Hospital Staff, System Administrator
Description	Predicts bin fill trends and generates alerts when critical levels are expected.
Precondition	Historical bin data available in Firestore.
Postcondition	Alerts displayed on dashboard.
Trigger	Periodic data analysis cycle.
Main Flow	1. Retrieves historical bin fill level data. 2. Executes predictive analysis algorithm. 3. Generates alerts for bins predicted to exceed capacity threshold.
Alternative Flow	None
Exception Flow	1a. Insufficient historical data reduces prediction accuracy.

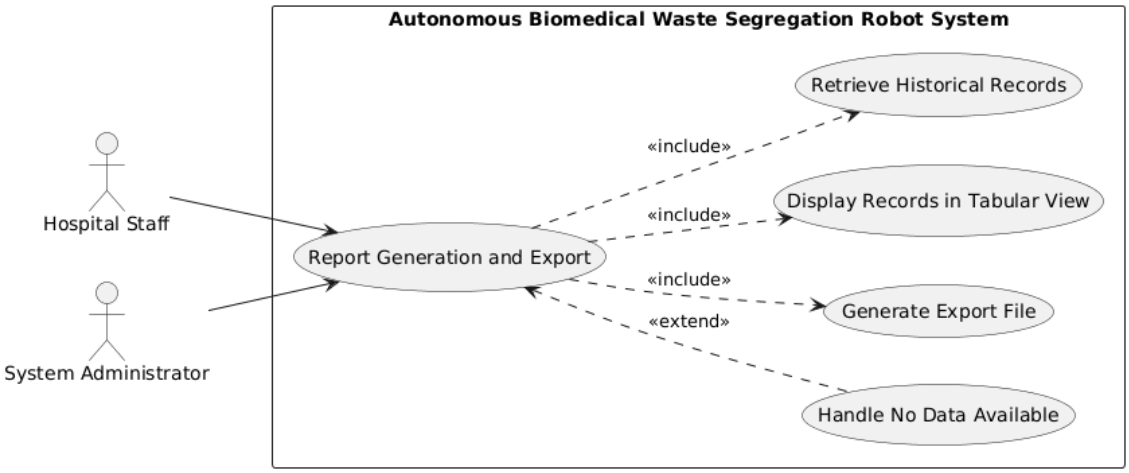


Figure 3.11: Use Case Diagram for Report Generation and Export (UC10)

Table 3.10: Use Case Specification for UC10 - Report Generation and Export

Use Case ID	UC10
Use Case Name	Report Generation and Export
Actor(s)	Hospital Staff, System Administrator
Description	Enables creation and export of structured waste management reports.
Precondition	User must be logged in and historical data must be available.
Postcondition	Report is generated and downloaded in selected format.
Trigger	User selects “Export Report” option.
Main Flow	1. User selects the bin in report section. 2. System retrieves relevant Firestore records. 3. System displays data in customizable table view. 4. User selects export format and triggers generation. 5. System generates and downloads report file.
Alternative Flow	3a. If no historical records exist for selected bin, system displays "No history available" message.
Exception Flow	2a. Data retrieval error prevents report generation.

3.6 Summary

This chapter presented the detailed requirement specifications for the Autonomous Biomedical Waste Segregation Robot. The system was analyzed in terms of its existing limitations, the proposed technological improvements, and comprehensive user-centered functional and non-functional requirements. The corresponding use cases defined user–system interactions ensuring traceability and alignment with the system objectives. This specification serves as the foundation for subsequent system design and implementation.

Chapter 4

Design

This chapter presents the architectural and detailed design of the *Autonomous Biomedical Waste Segregation Robot*. The design process focuses on transforming the approved requirements into a structured solution using standard software engineering practices. The system design formalizes how the hardware, software, and cloud components interact to achieve accurate biomedical waste detection, classification, segregation, logging, and monitoring. The primary goal of the design is to ensure reliability, scalability, low coupling, and high cohesion across system components.

4.1 System Architecture

The system architecture defines the structural organization of the Autonomous Biomedical Waste Segregation Robot. It describes the arrangement of the hardware subsystems, the software components, the data management infrastructure and the communication flow among these elements. The system integrates an embedded vision device, an artificial intelligence inference engine, a robotic handling simulation environment, a ultraviolet disinfection module, a cloud based data management service and a Streamlit based dashboard for monitoring and reporting.

The architecture enables real-time acquisition of image streams from the ESP32 camera, computer vision based detection of biomedical waste using YOLOv8, autonomous decision logic for category identification, robotic handling and ultraviolet disinfection, event logging into Firestore and user interaction through the dashboard. The information flow is managed in a controlled and sequential manner, ensuring transparency and consistency across the complete operational pipeline.

4.1.1 Framework Architecture

The framework architecture illustrates the complete technology stack that supports the implementation of the biomedical waste segregation system. Since the project is entirely Python based, all functional layers including the user interface, artificial intelligence inference, robotics simulation,

cloud connectivity and data export are developed within a consistent Python environment. This approach reduces platform dependencies and facilitates seamless interaction among components.

4.1.1.1 Framework Architecture Description

- **User Interface Layer** Implemented using the Streamlit framework. This layer provides user authentication, dashboard visualization, access to bin reports, viewing of alerts and downloading of generated reports.
- **Application Service Layer** Contains Python service modules that encapsulate system operations. This includes authentication, Firestore communication, waste prediction, reporting, alert processing and the pipeline management service.
- **Artificial Intelligence Layer** Implements the YOLOv8 model through the Ultralytics library. This layer performs detection of biomedical waste objects and maps detection classes to system categories.
- **Robotic Simulation Layer** Implemented using the PyBullet physics engine. This layer simulates the Franka Panda robotic arm, the gripper subsystem and the ultraviolet disinfection station. It executes pick and place operations based on the classification outputs.
- **Embedded Acquisition Layer** Consists of an ESP32 camera module that streams MJPEG images to the system. Frames are received through OpenCV for real-time analysis.
- **Cloud and Database Layer** Uses Google Cloud Firestore for storing structured data. Collections include users, bins, bin history, predictions, alerts and reports. The Firebase Authentication service manages user login credentials.
- **Report and Export Layer** Uses the Pandas library for generating comma separated value(CSV) and Excel files, which are stored in the exports directory for user download.

Figure 4.1 represents the framework architecture of Autonomous Biomedical Waste Segregation Robot.

4.1.2 High Level System Architecture

The high level architecture presents a top down view of the complete processing workflow. It illustrates the pathways through which data flows between the ESP32 camera, the artificial intelligence inference engine, the robotic simulation module, the Firestore database and the Streamlit dashboard. The system follows an Edge to Cloud architectural model in which image acquisition occurs at the embedded device, inference and actuation occur at the local processing layer and long term storage and reporting occur through the cloud database.

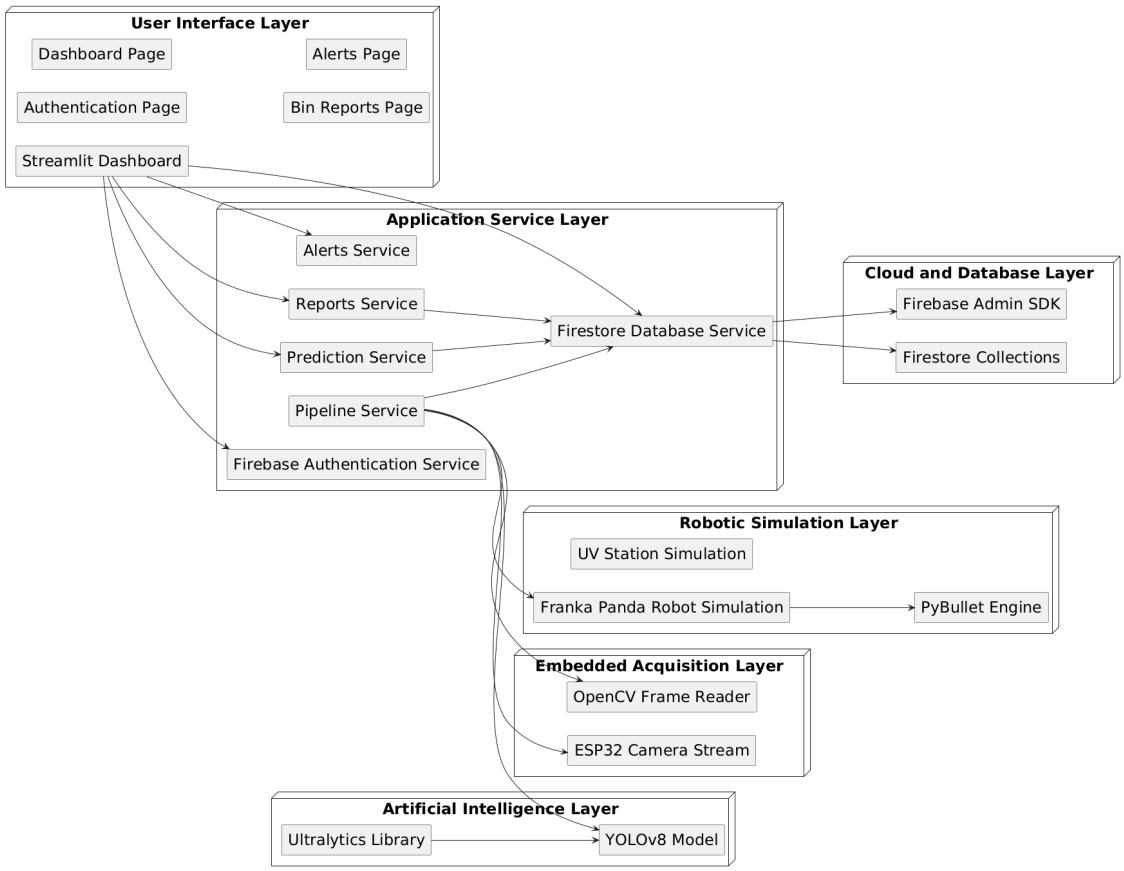


Figure 4.1: Framework Architecture

4.1.2.1 High Level Workflow Description

1. The ESP32 camera continuously streams biomedical waste images.
2. OpenCV retrieves frames from the stream and forwards them to the pipeline service.
3. The YOLOv8 model performs object detection and identifies biomedical waste items.
4. The classification module maps detected objects to predefined waste categories.
5. The pipeline manager issues commands to the PyBullet simulation for robotic actuation.
6. The robotic arm places detected waste through the ultraviolet station before final bin placement.
7. The system records each event into the Firestore database.
8. The Streamlit dashboard retrieves and displays system data to the user.

Figure 4.2 represents the high level system architecture of Autonomous Biomedical Waste Segregation Robot.

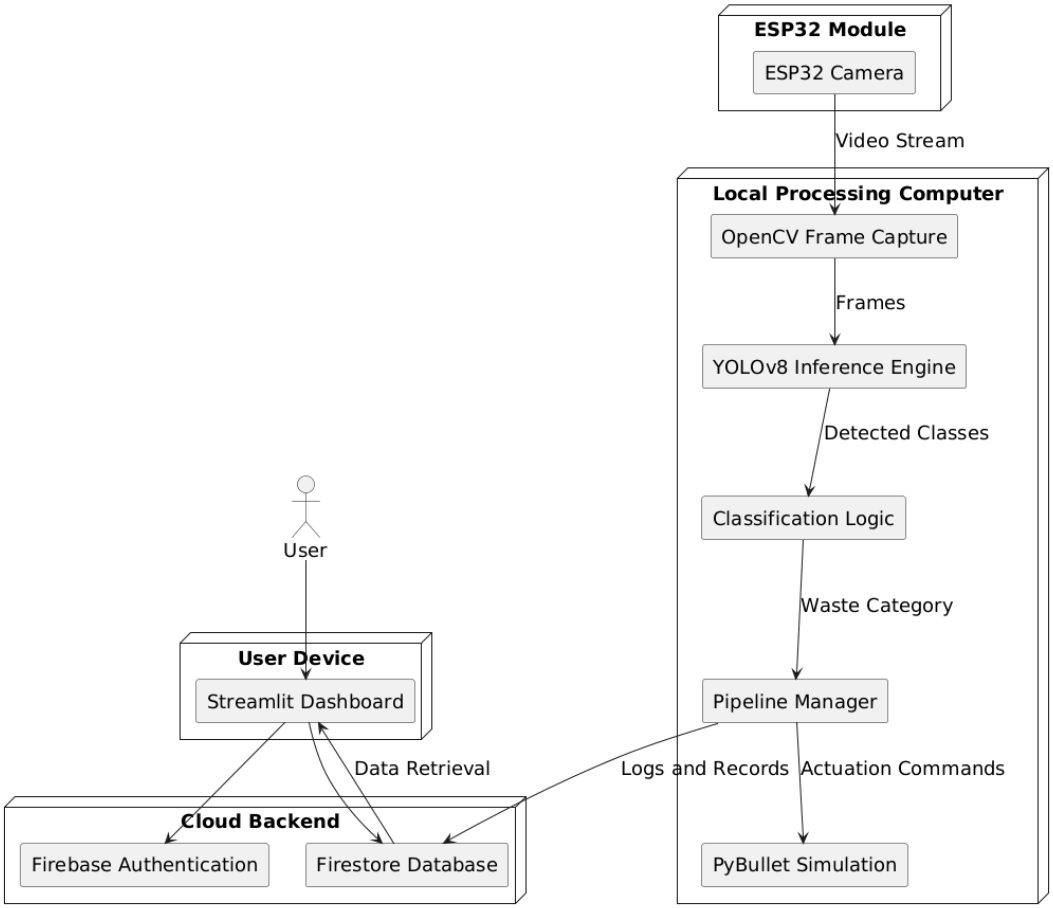


Figure 4.2: High Level System Architecture

4.1.2.2 Component Overview

This subsection provides a high-level description of the core system components involved in the Autonomous Biomedical Waste Segregation Robot. The components collaborate to enable waste detection, classification, segregation, and monitoring.

- **ESP32-CAM Module** The ESP32-CAM module serves as the image acquisition component of the system. It captures real-time visual data of biomedical waste items and streams image frames over a network connection. These frames act as input to the computer vision pipeline for waste detection and classification.
- **YOLOv8 Detection Model** The YOLOv8 deep learning model functions as the primary artificial intelligence component. It processes incoming image frames to detect and classify biomedical waste objects into predefined categories. The model outputs class labels and confidence scores, which are used for downstream decision-making.
- **Robotic Arm Simulation** The robotic arm is implemented using a physics-based simulation environment. Based on classification results, the simulated robot executes pick, transfer,

ultraviolet sterilization, and disposal operations. This component validates the segregation logic without requiring physical hardware.

- **Firebase Cloud Backend** Firebase Firestore acts as the cloud-based backend for the system. It stores detection records, bin status information, historical timelines, alerts, and prediction data. Firebase Authentication manages secure user access to the system.
- **Web-Based Dashboard** The Streamlit-based web dashboard provides the user interaction layer. It enables authorized users to monitor waste detection events, view bin fill levels, analyze historical data, receive alerts, and export reports in standard formats.

4.2 Software Architecture

The software architecture defines the structural decomposition of the system into functional layers, the software process model adopted for its development and the architectural style governing interaction between components. The Autonomous Biomedical Waste Segregation Robot integrates computer vision, robotic simulation, ultraviolet sterilization logic, cloud data management and a Streamlit based dashboard. The architecture therefore employs principles of modularity, abstraction and controlled data exchange to ensure reliability and traceability across the complete system lifecycle.

4.2.1 Software Process Model

The project follows an Iterative Incremental Hybrid Software Process Model. This model was selected due to the multidisciplinary nature of the system, which integrates computer vision, robotic handling, ultraviolet sterilization workflow, cloud-based storage, predictive analytics and a real-time monitoring dashboard. The hybrid model combines the evolutionary refinement of iterative development with the functional decomposition principles of incremental delivery.

Each major subsystem was developed as a separate increment while allowing iterative refinement within that increment. The first increment focused on perception and YOLOv8 based waste classification. The second increment integrated robotic simulation using the PyBullet environment together with ultraviolet sterilization logic. The third increment established cloud datastore integration and timeline logging through Firestore. The final increment introduced the Streamlit dashboard, the prediction engine and the alerting mechanism. Iterative cycles within each increment enabled continuous evaluation of model accuracy, robotic actuation reliability and user experience.

This software development approach ensured stable progress across interacting hardware and software components while enabling early detection of design inconsistencies. Furthermore, it supports future extensions, such as replacing the PyBullet robotic simulation with a physical robotic arm, without restructuring core subsystems.

Figure 4.3 presents the Iterative Incremental Hybrid Software Process Model used in the project.

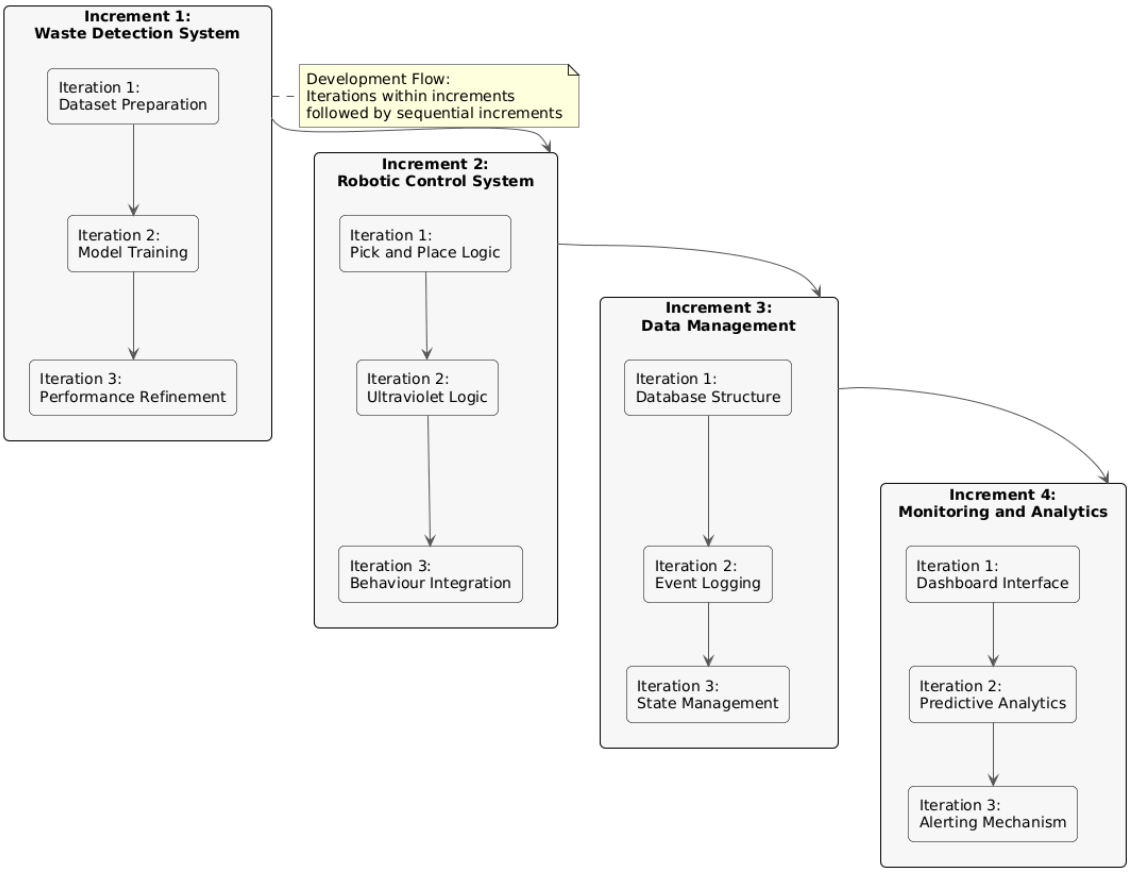


Figure 4.3: Iterative Incremental Hybrid Software Process Model

4.2.2 Architectural Model

The architectural model of the system follows a Multi Layer Edge AI Cloud Architecture. This architecture organizes the system into structured layers that separate data acquisition, inference, actuation, cloud communication and user interface responsibilities. The design ensures reliability, performance and traceability consistent with software engineering best practices.

- **Edge Layer** Includes the ESP32 camera module that streams real-time MJPEG frames. This layer performs no preprocessing and acts solely as a data acquisition point.
- **Artificial Intelligence Layer** Implements the YOLOv8 model using the Ultralytics framework. It performs object detection and category mapping for biomedical waste types.
- **Robotic Actuation Layer** Implemented in PyBullet with a Franka Panda robotic arm. It executes pick and place operations and simulates ultraviolet sterilization through predefined spatial coordinates.
- **Cloud Layer** Uses Firestore for structured data storage including waste logs, predictions, alerts, bin status and user accounts. The Firebase Authentication service manages login operations.

- **Application Service Layer** Contains Python services for database operations, authentication, prediction, reporting and the pipeline manager.
- **Presentation Layer** Streamlit based web interface providing dashboard visualization, report downloads, timeline exploration and alert management.

Figure 4.4 shows the Multi Layer Edge Artificial Intelligence Cloud Architectural Model.

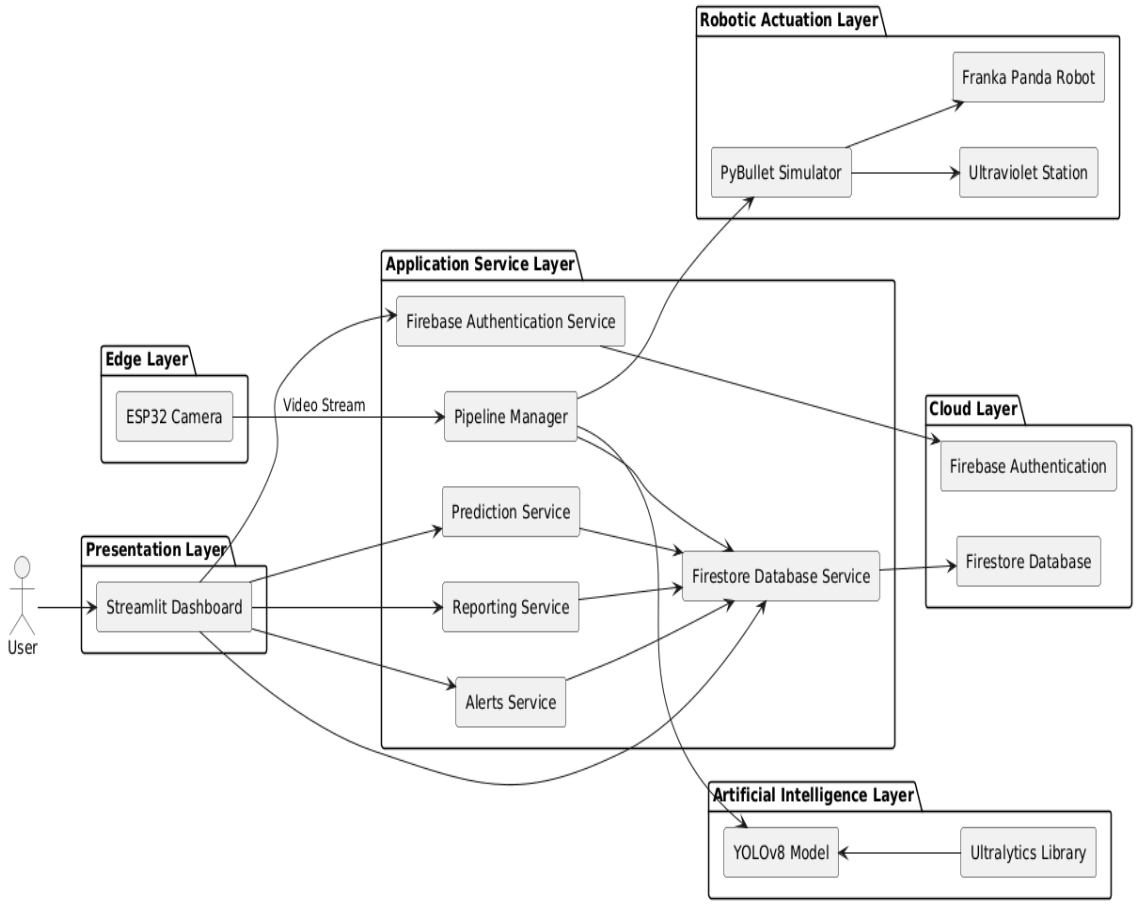


Figure 4.4: Multi Layer Edge AI Cloud Architectural Model

4.3 YOLOv8 Detection Workflow

This section describes the conceptual workflow of the YOLOv8 object detection model as applied in the Autonomous Biomedical Waste Segregation Robot. The objective is to explain how image data is transformed into classified biomedical waste categories without delving into mathematical formulations or low-level neural network equations.

4.3.1 Input Image Acquisition

The workflow begins with real-time image frames streamed from the ESP32 camera module. Each frame represents a potential biomedical waste instance and is forwarded to the inference pipeline without prior manual intervention.

4.3.2 Feature Extraction

The input image is processed by the YOLOv8 backbone network, which extracts hierarchical visual features such as edges, textures and object shapes. These features enable the model to distinguish biomedical waste objects from the background under varying lighting and orientation conditions.

4.3.3 Object Detection and Localization

The extracted features are passed to the detection head, where the model predicts bounding boxes around detected objects. Each bounding box represents a candidate biomedical waste item along with its spatial location in the image.

4.3.4 Class Assignment and Confidence Scoring

For each detected object, YOLOv8 assigns a class label corresponding to a biomedical waste category such as mask, glove, syringe or needle. A confidence score is also generated to represent the likelihood of correct classification. Only detections above a defined confidence threshold are considered valid.

4.3.5 Output Generation

The final output of the YOLOv8 workflow consists of bounding box coordinates, class labels and confidence scores. These outputs are forwarded to the pipeline manager, which maps the detected class to a system-defined waste category and triggers subsequent robotic handling and data logging operations.

Figure 4.5 illustrates the conceptual workflow of the YOLOv8 detection pipeline used in the system.

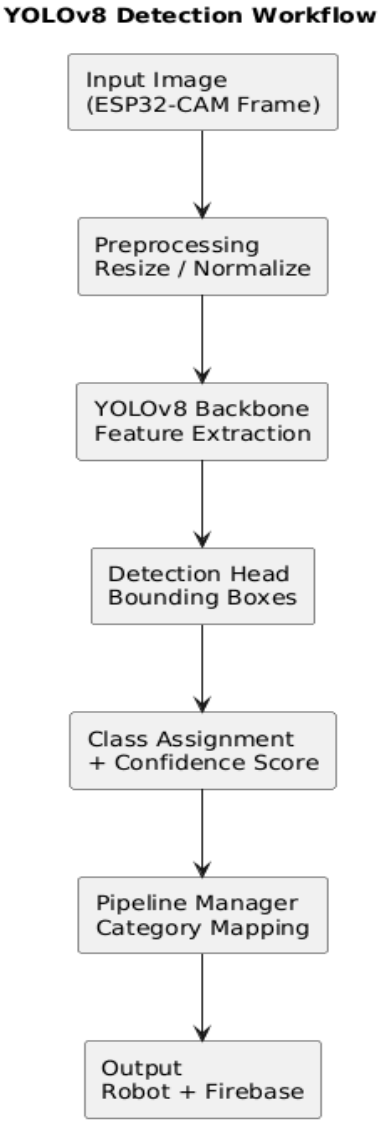


Figure 4.5: YOLOv8 Detection Workflow - Conceptual

4.4 System Flow Diagrams

The system flow diagrams describe the dynamic behavior of the Autonomous Biomedical Waste Segregation Robot. These diagrams illustrate the real-time operational logic, the interaction among subsystems and the sequence of activities performed from waste detection to final logging. The diagrams provide an integrated understanding of control flow, data flow and system responses under normal operating conditions. The system employs a unified processing pipeline in which image acquisition, artificial intelligence inference, robotic actuation, ultraviolet sterilization, data logging and dashboard visualization are connected as part of a continuous workflow.

4.4.1 Data Flow Diagrams

Data Flow Diagrams (DFDs) illustrate how data moves through the Autonomous Biomedical Waste Segregation Robot and how it is transformed at each processing stage. Unlike control-oriented diagrams, DFDs focus on data generation, processing, storage and visualization. This representation complements the architectural and behavioral diagrams by highlighting the system's information flow.

4.4.1.1 Level-0 Data Flow Diagram

The Level-0 DFD, also known as the context diagram, represents the system as a single high-level process and illustrates its interaction with external entities. The primary external entities include the ESP32 camera module, the Firebase cloud database and the system users accessing the Streamlit dashboard.

In this level, image data is captured by the ESP32 camera and forwarded to the core processing system. The system performs waste detection, classification and segregation, and stores the resulting data in Firestore. Authorized users retrieve processed information through the dashboard for monitoring, reporting and decision-making.

Figure 4.6 presents the Level-0 Data Flow Diagram of the system.

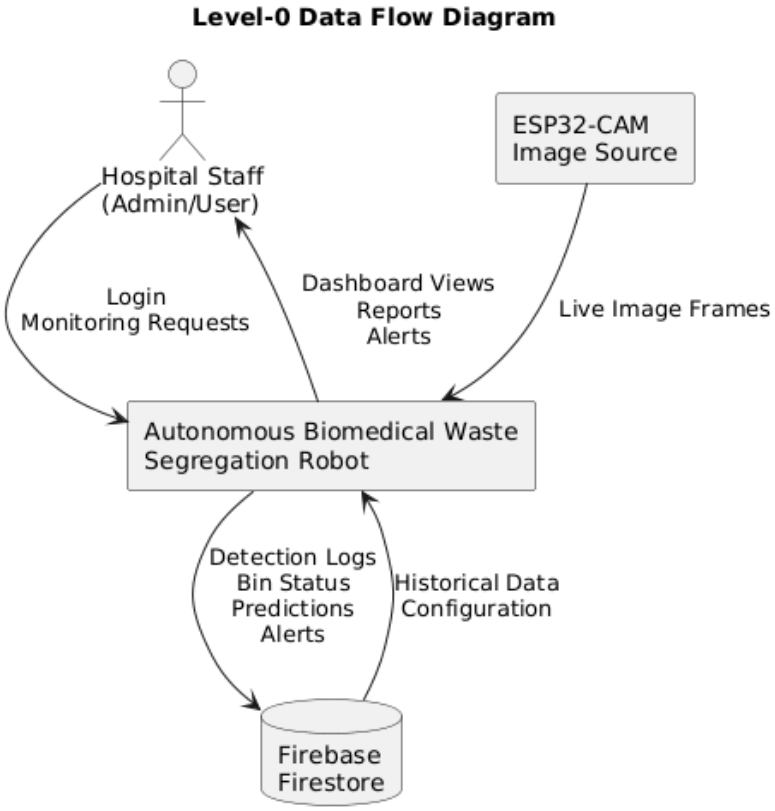


Figure 4.6: Level-0 Data Flow Diagram of the System

4.4.1.2 Data Flow Description

The internal data flow of the system follows a structured pipeline:

- Image frames captured by the ESP32 camera are streamed to the processing unit.
- The YOLOv8 inference engine analyzes the frames and generates detection results.
- Classification outputs are forwarded to the robotic simulation and logging modules.
- Event data, bin status and predictions are stored in the Firestore database.
- The Streamlit dashboard retrieves stored data and presents it to users in graphical and tabular formats.

This data-driven architecture ensures consistency, traceability and real-time visibility across all system components.

4.4.2 Activity Diagram

The activity diagram describes the detailed sequence of actions and decision points involved in the waste segregation process. It emphasizes the behavioral flow that begins with continuous image acquisition, progresses through classification and logging operations and subsequently triggers robotic actuation within the simulation environment. The diagram also captures parallel processes, such as simultaneous dashboard updates and robot motion execution, which occur after the detection and categorization phase. This diagram highlights the behavioral logic independent of implementation and effectively illustrates how sensing, computation, and actuation components interact throughout the system lifecycle.

Figure 4.7 presents the activity diagram of the system.

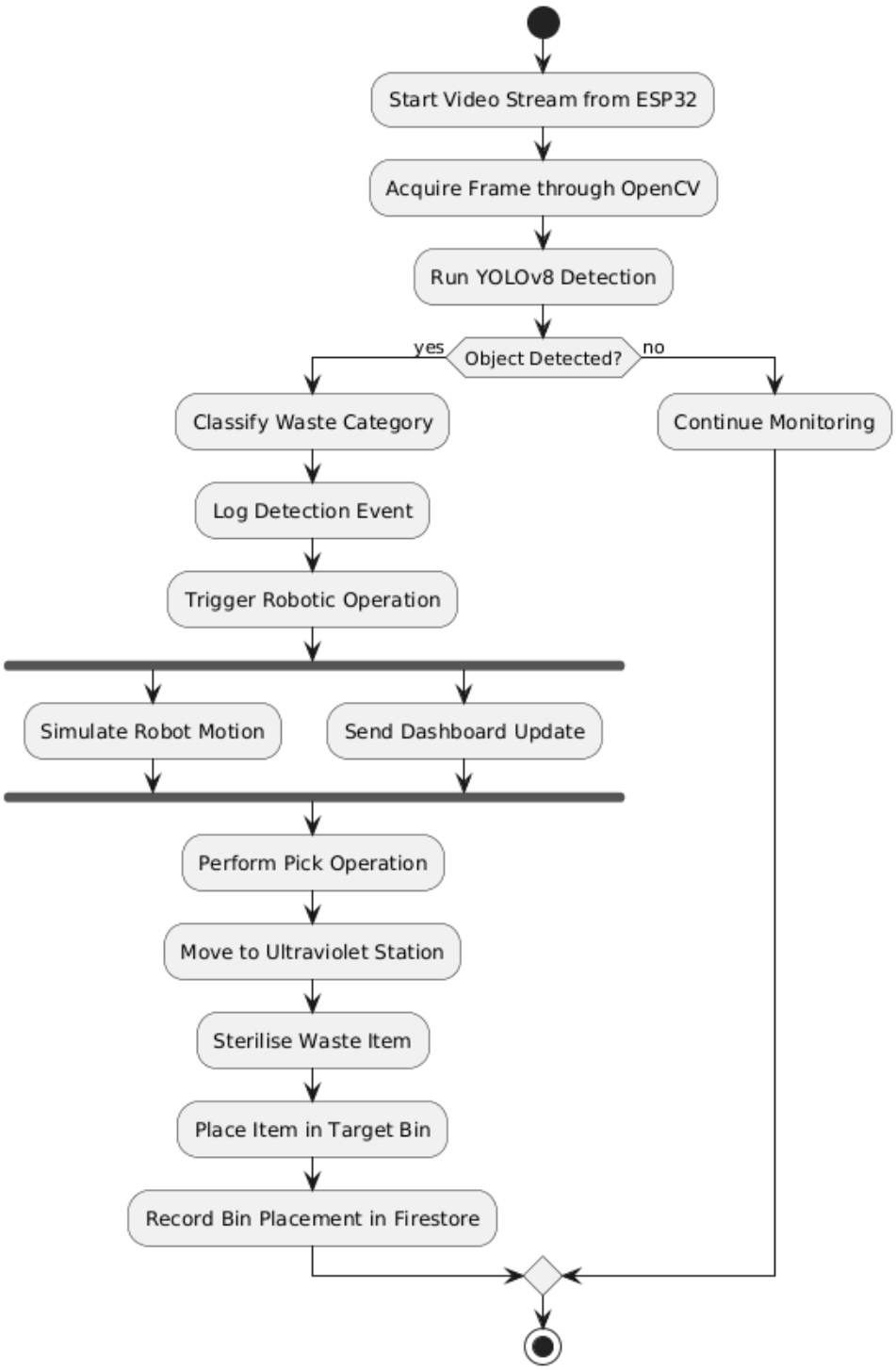


Figure 4.7: Activity Diagram of the System

4.4.3 Sequence Diagrams

The sequence diagrams describe message exchange and temporal ordering of operations among subsystems. Individual diagrams are provided for each major module to ensure clarity and modular

traceability. These diagrams present the runtime behavior as seen in the implemented Python services.

4.4.3.1 Sequence Diagram for Waste Detection Pipeline

This sequence diagram illustrates the runtime interaction among the embedded acquisition module, the computer vision inference engine, and the cloud logging service during biomedical waste detection. The process begins when the ESP32 camera continuously streams image frames over an HTTP-based MJPEG feed. These frames are captured by the OpenCV interface and forwarded to the pipeline service for processing. The pipeline service invokes the YOLOv8 inference engine to perform object detection and classification on each frame. Upon successful detection, the identified waste category and confidence information are returned to the pipeline manager. The system then records the detection event, along with relevant metadata such as timestamp and category, into the Firestore database. This sequence ensures that perception, classification, and data logging are executed in a coordinated and traceable manner, forming the foundation for subsequent robotic actuation and monitoring operations.

Figure 4.8 presents the sequence of operations for the ESP32 camera and YOLOv8 inference pipeline.

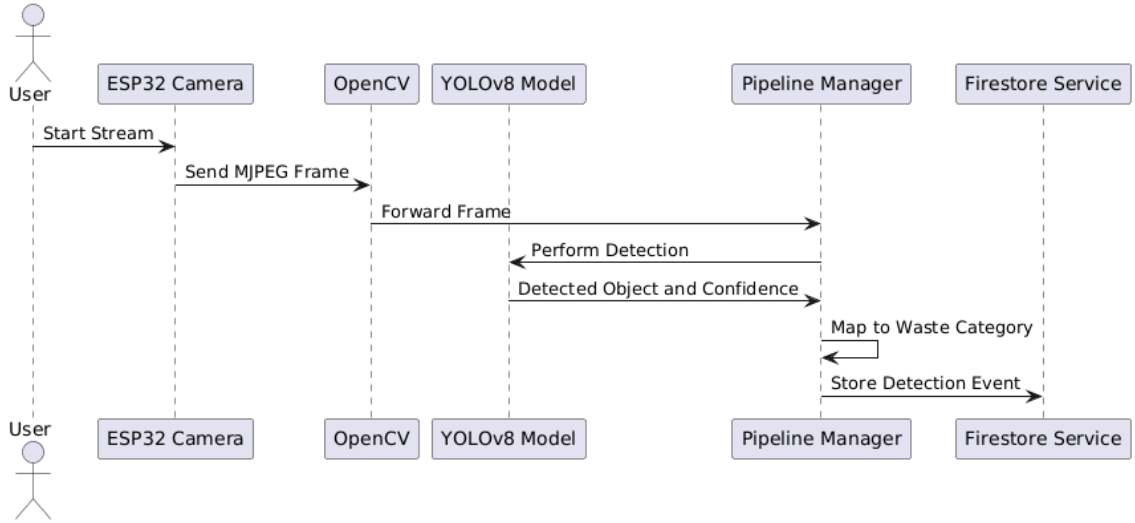


Figure 4.8: Sequence Diagram for Waste Detection Pipeline

4.4.3.2 Sequence Diagram for Robotic Pick and Place Operation

This sequence diagram represents the interaction flow responsible for robotic waste handling within the PyBullet simulation environment. Once the pipeline service receives the classified waste category from the detection module, it issues a command to the robotic control interface. The PyBullet simulation initializes the corresponding robotic action, where the Franka Panda robotic arm performs a pick operation on the detected waste item. After grasping the object, the arm moves the item through the ultraviolet sterilization station, where a simulated UV-C exposure is applied

for a predefined duration based on waste category. Upon completion of sterilization, the robotic arm places the waste item into the appropriate disposal bin. Following successful placement, the pipeline service updates the Firestore database with bin status and disposal event information. This sequence ensures deterministic and safe execution of waste segregation while maintaining synchronization between robotic actions and cloud-based logging.

Figure 4.9 represents the robotic actuation flow implemented in PyBullet.

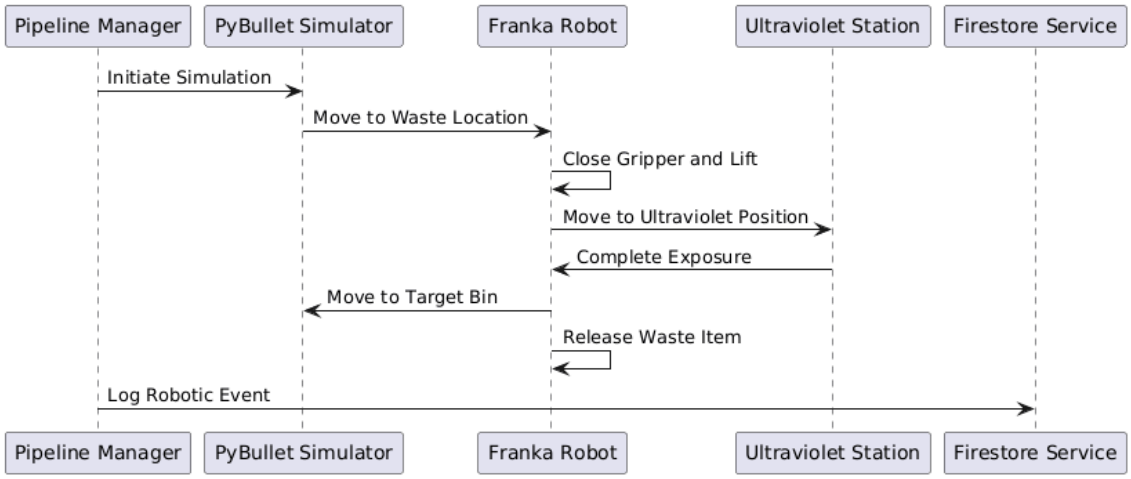


Figure 4.9: Sequence Diagram for Robotic Pick and Place Operation

4.4.3.3 Sequence Diagram for Dashboard Data Retrieval

This sequence diagram depicts the message flow involved in retrieving and presenting system data on the Streamlit-based dashboard. After a user successfully logs into the system, the dashboard initiates requests to the application service layer for updated system information. The service layer queries the Firestore database to retrieve current bin status, waste distribution statistics, alert records, and historical timeline data. The retrieved data is then processed and formatted by the dashboard logic into visual components such as charts, tables, and status indicators. In parallel, alert information and prediction results are fetched to inform users of potential capacity risks. This sequence highlights the read-only interaction pattern of the dashboard, where Firestore acts as the single source of truth, ensuring that all visualized information accurately reflects the most recent system state.

Figure 4.10 shows the message flow for updating the Streamlit dashboard.

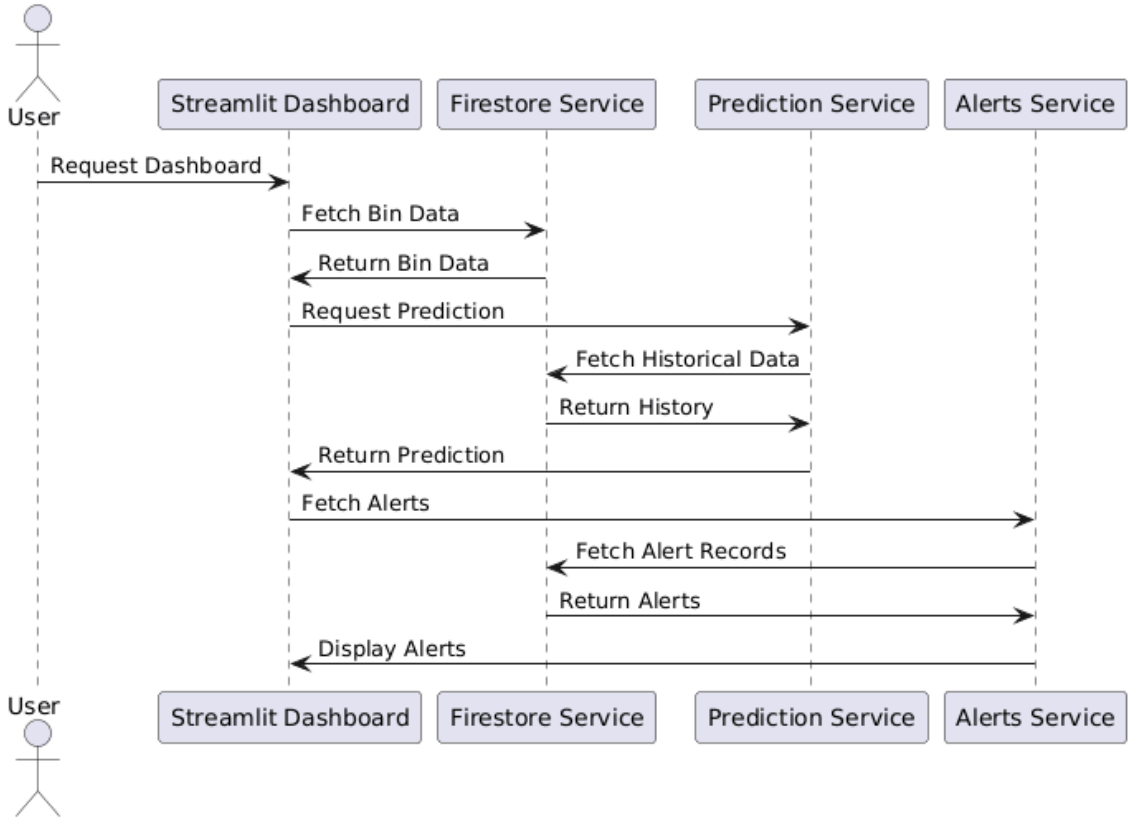


Figure 4.10: Sequence Diagram for Dashboard Data Retrieval

4.5 Entity Relationship Diagram

The Entity Relationship Diagram defines the logical data structure of the Autonomous Biomedical Waste Segregation Robot. The system uses Google Cloud Firestore as a NoSQL document database. Although Firestore does not employ relational tables, its collections and documents can be modelled through an enhanced entity relationship representation. This provides a conceptual view of entities, attributes and relationships that remain consistent with classical data modelling practice.

The database schema was designed to support real-time event logging, bin level monitoring, prediction analytics, report generation and user account administration. The primary collections in the system include users, bins, bin history, alerts, predictions and reports. Each collection contains documents that represent atomic data units, while relationships between them are represented through document references and shared keys.

The entity relationship diagram formalizes the interactions among these collections and illustrates how data flows across the cloud layer. Figure 4.11 presents the complete entity relationship model of the system.

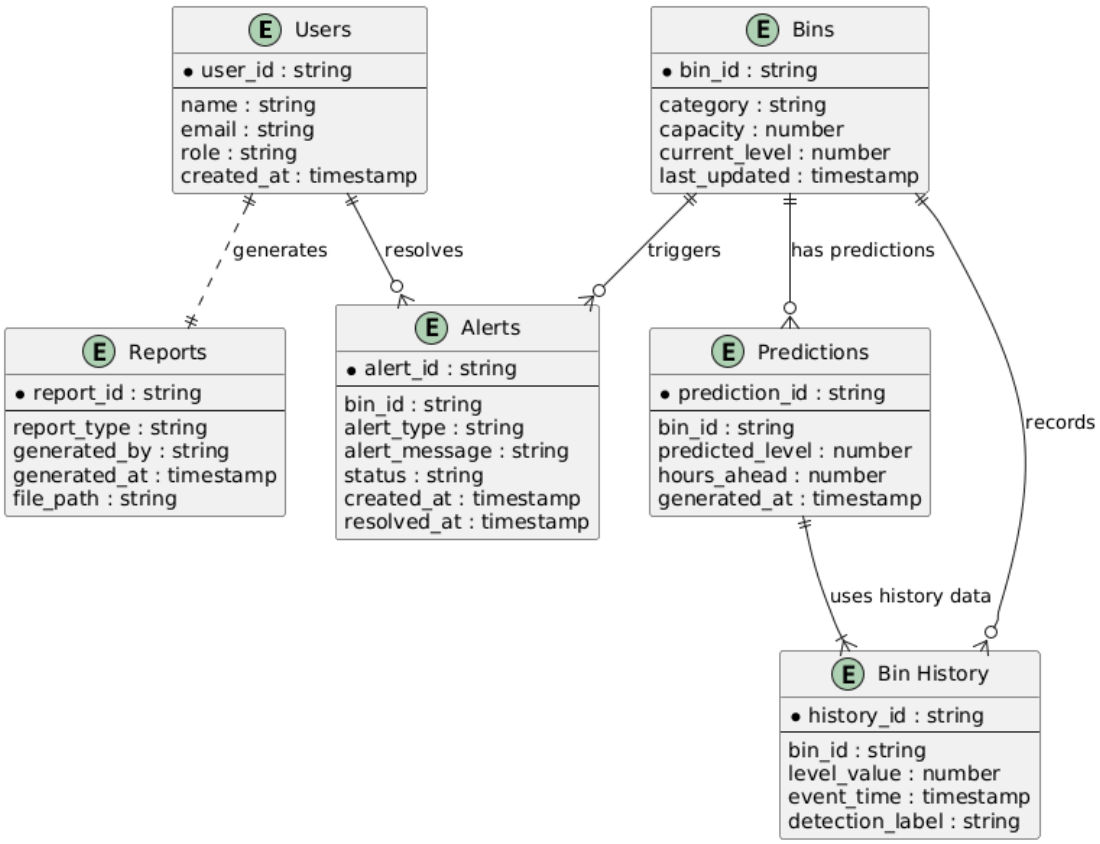


Figure 4.11: Entity Relationship Diagram for the System

4.6 Finite State Machine Diagram

The Finite State Machine describes the behavioral model of the autonomous robotic waste segregation system. It captures the sequence of operational states, the events that trigger transitions and the actions associated with each state. The FSM provides a formal specification of the dynamic behavior that is executed during the complete waste detection and handling cycle.

The finite state behavior is governed by the pipeline service which integrates computer vision, robotic simulation and ultraviolet exposure. The robot progresses deterministically from initial idle mode to detection, classification, pick and place operation, ultraviolet sterilization and final disposal. State transitions are triggered by internal events such as successful waste detection, classification output from the YOLOv8 model, completion of robotic movement or completion of ultraviolet exposure.

The FSM ensures that every waste item undergoes controlled processing and that the system returns to a stable idle condition after completion of each cycle. Figure 4.12 presents the finite state machine diagram for the robotic workflow.

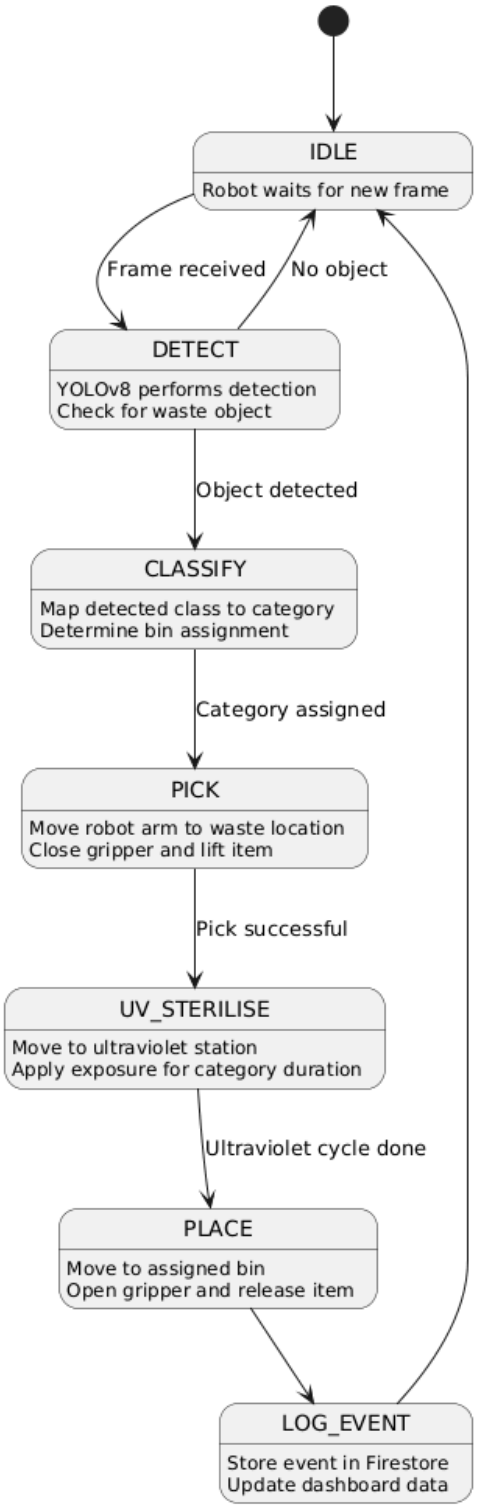


Figure 4.12: Finite State Machine for the Biomedical Waste Segregation Robot

4.7 Summary

This chapter presented the complete architectural and detailed design of the Autonomous Biomedical Waste Segregation Robot. The design formally translated the system requirements into a structured, modular and scalable solution. The framework architecture established the underlying technology stack, while the high-level system architecture demonstrated the coordinated interaction among the embedded acquisition layer, artificial intelligence inference engine, robotic actuation layer, cloud datastore and Streamlit based dashboard. The software architecture further detailed the process model and the layered design principles governing system behavior. The system flow diagrams, data flow diagrams and sequence diagrams provided a comprehensive view of the operational logic and data transformations from detection to disposal. The entity relationship diagram defined the cloud-based data organization, and the finite state machine model captured the dynamic behavior of the robotic workflow. Together, these design models provide a complete, coherent and robust blueprint for system implementation in the subsequent chapter.

Chapter 5

System Implementation

5.1 Introduction

This chapter presents the complete implementation of the Autonomous Biomedical Waste Segregation Robot, describing how the system architecture defined in Chapter 4 was transformed into a fully functional software and simulation environment. The system was implemented entirely in Python, integrating edge level image acquisition, deep learning based classification, robotic motion simulation, ultraviolet sterilization processes, cloud data handling, and a Streamlit based web dashboard for user interaction. Each subsystem was developed as an independent but interoperable module to ensure maintainability, scalability, and alignment with the design principles defined earlier. The chapter provides a detailed explanation of the development environment, system configuration, core modules, and integrated behavior of the deployed system.

5.2 Development Environment

The system was implemented using a Python centred development environment due to its extensive ecosystem of libraries for machine learning, computer vision, robotics and web application development. The complete implementation was developed and tested on a Windows machine using Visual Studio Code as the primary integrated development environment. Streamlit was used for delivering the web based dashboard, enabling rapid prototyping of data driven user interfaces.

Model training and experimentation were conducted on a Kaggle Notebook environment equipped with a NVIDIA Tesla P100 GPU, enabling accelerated YOLOv8 training. Firestore configuration and authentication mechanisms were configured through the Firebase Console, and the robotic simulation environment was implemented using the PyBullet physics engine with the Franka Emika Panda robotic arm model.

5.3 System Setup and Configuration

5.3.1 Firestore and Firebase Configuration

The system uses Google Cloud Firestore as the central NoSQL storage service. The Firebase Admin SDK was initialized using secure service account credentials, enabling authenticated read and write operations. The Firestore database was structured according to the schema defined in Chapter 4, including collections for users, bins, alerts, bin history, detections and prediction logs. The Firebase Authentication REST API was used to validate email and password credentials provided by authorized users at login. Configuration parameters, API keys and credential paths were stored in a dedicated configuration module to ensure separation of concerns. Firebase was selected due to its real-time synchronization capabilities, scalability, and seamless integration with IoT and web-based applications, making it suitable for healthcare monitoring systems.

5.3.2 Streamlit Configuration

Streamlit served as the presentation layer and was configured to support user authentication, session management and page level routing. Custom session state handlers were implemented to retain authentication tokens and dynamic system data throughout the user session. The Streamlit interface includes multiple pages, such as the dashboard, bin report viewer and alerts interface, each implemented using modular components for tables, cards and charts.

5.3.3 YOLOv8 Model Configuration

The YOLOv8 model trained on biomedical waste images was integrated into the system through the Ultralytics library. The best performing model weights, exported as `best.pt`, were loaded at runtime by the Pipeline Manager module. Frames were prepared using OpenCV, then forwarded to the YOLOv8 model for inference.

5.3.4 ESP32 Camera and OpenCV Setup

The ESP32-CAM module was configured to stream MJPEG video over an HTTP URL. OpenCV's `VideoCapture` object was used to retrieve frames in real time. Error checking routines ensured robust handling of intermittent network delays or dropped frames. Each captured frame was passed to the detection pipeline for inference.

5.3.5 PyBullet Simulation Setup

The PyBullet engine was configured to simulate a Franka Emika Panda robotic arm and a virtual workstation including the ultraviolet sterilization zone and bin positions. The Simulation Manager initialized the physics client, loaded the Panda URDF model, configured joint constraints and set environment parameters such as gravity. Kinematic functions were implemented to control end effector movements during picking, sterilization and placement operations.

5.4 AI Model Training and Evaluation

5.4.1 Dataset Description

The biomedical waste detection model was trained using the *Pharmaceutical and Biomedical Waste (PBW)* dataset obtained from Kaggle [15]. The original dataset consisted of 6,587 files, including 6,586 labeled images and one documentation file. After removing non-relevant categories, the dataset was reduced to 3,873 images across biomedical waste classes.

To enhance dataset diversity and improve generalization performance, data augmentation techniques were applied using the Roboflow platform. After augmentation, the dataset size increased to a total of 9,124 images. The dataset was automatically split using Roboflow into training, validation, and testing subsets, which were verified during model training.

Table 5.1 presents the statistics of dataset.

Table 5.1: Dataset Statistics

Description	**Value**
Original Images	6,586
Filtered Images	3,873
Augmented Images	9,124
Number of Classes	14
Training Images	7,983 (87%)
Validation Images	571 (6%)
Testing Images	570 (6%)

The dataset comprises 14 biomedical waste categories, including masks, gloves, syringes, needles, cotton, and related medical disposables.

5.4.2 Dataset Annotation and Preprocessing

Dataset annotation and preprocessing were performed using the Roboflow platform. Roboflow was utilized to standardize class labels, resize images to the required input resolution, and apply augmentation techniques such as rotation, scaling, and brightness variation. These preprocessing steps improved dataset robustness and reduced overfitting during training.

The processed dataset was exported in YOLO format, ensuring direct compatibility with the Ultralytics YOLOv8 training pipeline.

5.4.3 Model Training

The object detection model was trained using the Ultralytics YOLOv8 framework with the `yolov8s.pt` architecture. Transfer learning was employed by initializing the model with pre-trained weights to improve convergence speed and performance.

Training was conducted on the Kaggle cloud platform using an NVIDIA Tesla P100 GPU with 16 GB VRAM. The model was trained for 100 epochs with early stopping enabled to prevent overfitting. Key training hyperparameters are summarized in Table 5.2.

Table 5.2: YOLOv8 Training Hyperparameters

Parameter	**Value**
Model Architecture	YOLOv8-Small
Batch Size	32
Image Size	640 × 640
Optimizer	SGD
Initial Learning Rate	0.01
Momentum	0.9
Weight Decay	0.0005
Epochs	100
Mixed Precision (AMP)	Enabled
Workers	8

5.4.4 Model Evaluation Results

Model performance was evaluated using standard object detection metrics, including Precision, Recall, F1-score, and Mean Average Precision (mAP). The trained YOLOv8 model achieved strong detection accuracy across all biomedical waste categories.

The evaluation results obtained on the test dataset are summarized in Table 5.3

Table 5.3: Model Evaluation Results

Metrics	**Value**
Precision	0.9454
Recall	0.9463
F1 Score	0.9458
mAP@0.5	0.9726
mAP@0.5:0.95	0.7570

These results indicate high classification reliability and localization accuracy, validating the effectiveness of the proposed deep learning model for biomedical waste detection.

5.5 Module-Wise Implementation

5.5.1 User Interface Implementation

The Streamlit based user interface comprises multiple interactive pages that render real-time system data. The Authentication Page enables credential verification using the Firebase Authentication

Service. The Dashboard Page presents summary cards, graphical trends, bin status indicators and recent activity logs retrieved from Firestore. The Bin Reports Page provides historical records and supports sorting, filtering, and exporting. The Alerts Page displays system generated warnings such as overflow risks.

5.5.2 Authentication Service Implementation

The authentication subsystem was implemented using Firebase's REST authentication API. The service receives user credentials from the Streamlit interface and validates them through an API key secured within the project configuration. Once validated, authentication tokens are stored in Streamlit's session state and used to control page access and user specific features.

5.5.3 Firestore Database Service Implementation

The Firestore Database Service abstracts all communication between the application and Firestore. The service provides functions for creating, reading and updating entries across all collections. Each event is logged to the respective category in firestore. The dashboard retrieves summaries, statistics and time series data through optimized Firestore queries.

5.5.4 YOLOv8 Detection Pipeline Implementation

The detection pipeline is responsible for frame analysis, object detection and category mapping. Each frame captured through OpenCV is preprocessed before being forwarded to the YOLOv8 model. Detected objects are mapped to biomedical waste classes such as masks, gloves, syringes, needles and cotton. A mapping function converts these classes into system categories including infectious, non-infectious and sharp waste.

5.5.5 Pipeline Manager and Waste Handling Logic

The Pipeline Manager coordinates the decision making and task execution following a detection event. Once a waste category is identified, the Pipeline Manager triggers the simulation module, assigns the correct bin location based on category and logs the start of the handling operation. The manager ensures reliable sequencing of robot movement, UV exposure duration, and waste drop-off operations.

5.5.6 Robotic Simulation Implementation

The PyBullet based simulation implements the pick and place workflow for waste handling. The end effector is moved using joint position control commands. A grasp sequence is executed after aligning the gripper with the detected waste item. The robot then transports the item to the ultraviolet sterilization station, where a timed exposure process simulates microbial deactivation. Finally, the robot places the waste in the designated bin and resets its position for the next cycle.

5.5.7 Prediction Service Implementation

The Prediction Service computes future bin fill levels using historical time stamped data. A regression based approach was implemented using NumPy to estimate fill trends and generate forecasts for specified time horizon. Prediction results are logged in Firestore and displayed on the Dashboard. This enables proactive waste handling decisions and supports better resource allocation.

5.5.8 Reporting Service Implementation

The Reporting Service generates structured CSV and Excel files summarizing system activity. Firestore records are retrieved and tabulated using the Pandas library. Reports include waste category counts, bin utilization statistics, and time stamped operations. Generated files are stored in the `exports` directory for user download and audit purposes.

5.5.9 Alerts Service Implementation

The Alerts Service monitors system indicators and generates warnings based on defined thresholds. Alerts are triggered for bin overflow predictions, detection failures, or robotic errors. The Streamlit Alerts Page retrieves and displays these entries to authorized users, ensuring continuous system awareness and timely corrective action.

5.6 Integration of Subsystems

The entire system operates as a cohesive pipeline integrating sensing, machine learning, simulation and cloud based storage. Frames from the ESP32 camera are processed by the detection pipeline, which then passes classification results to the robotic simulation module. Robotic actions are executed in PyBullet and logged in Firestore. The dashboard retrieves real-time and historical data through Firestore, while prediction and alerts modules also operate on Firestore data streams. This fully integrated workflow ensures seamless operation across the edge, AI, robotics and cloud layers.

5.7 Graphical User Interface (GUI)

The system provides a user-friendly and interactive web-based interface developed using Streamlit framework. The GUI is designed to facilitate real-time monitoring, historical analysis, and administrative control of the biomedical waste segregation system. A modular, page-based architecture with intuitive navigation is adopted to accommodate hospital staff and waste management personnel with varying levels of technical expertise. Emphasis is placed on usability, responsiveness, and clarity to ensure effective operation in time-critical hospital environments.

5.7.1 Authentication Page

The Authentication Page enables authorized users to securely access the system using credentials validated through Firebase Authentication. The interface includes input fields for email and password, along with contextual error messages to guide users in case of invalid credentials. Upon successful authentication, users are automatically redirected to the Home Page, ensuring a seamless user experience while maintaining data confidentiality and access control.

Figure 5.1 illustrates the *Authentication Page* of the Streamlit web dashboard.

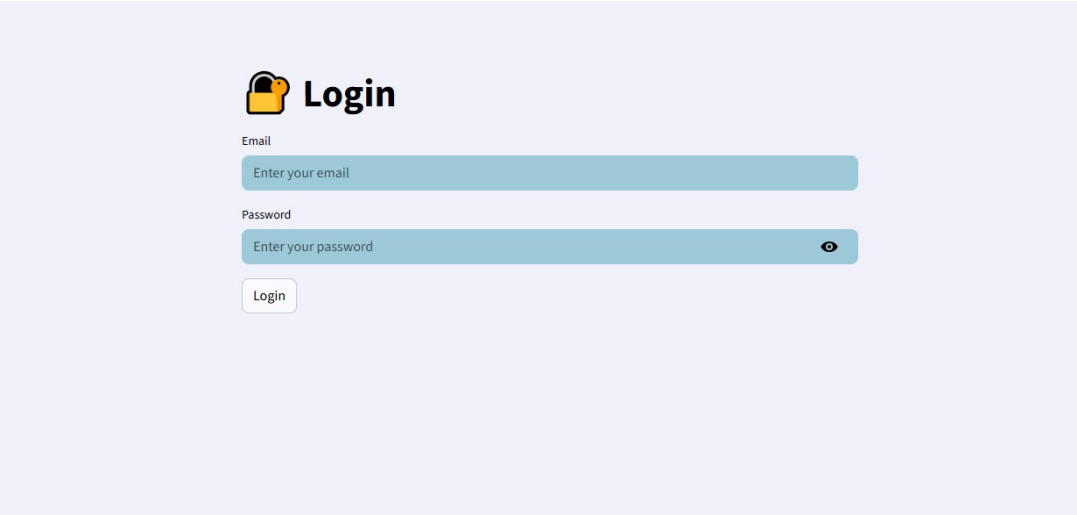


Figure 5.1: Authentication Page Interface

5.7.2 Home Page

The Home Page serves as the central navigation hub of the system after successful authentication. It presents a personalized greeting displaying the authenticated user's email address and provides quick access to all major system modules through clearly labeled, card-based action panels. Each card contains a brief description of the corresponding module along with direct navigation buttons. A prominently placed logout option ensures secure session termination. The Home Page employs a clean layout with adequate whitespace to reduce cognitive load and enhance accessibility in high-pressure clinical settings.

Figure 5.2 shows the *Home Page* of the Streamlit web dashboard.

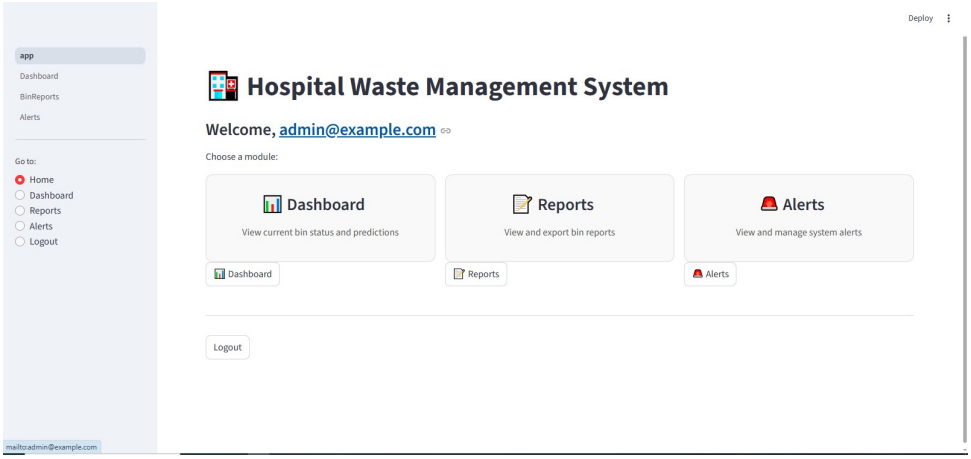


Figure 5.2: Home Page Interface

5.7.3 Dashboard Page

The Dashboard Page acts as the primary monitoring interface, providing comprehensive real-time insights into system operations. It presents an overview of current system activity, including bin status, waste category statistics, and predictive analytics. Streamlit widgets are utilized to render information cards and interactive charts that summarize ongoing waste handling metrics.

Figure 5.3 illustrates the *Dashboard Page* showing the current bin status.

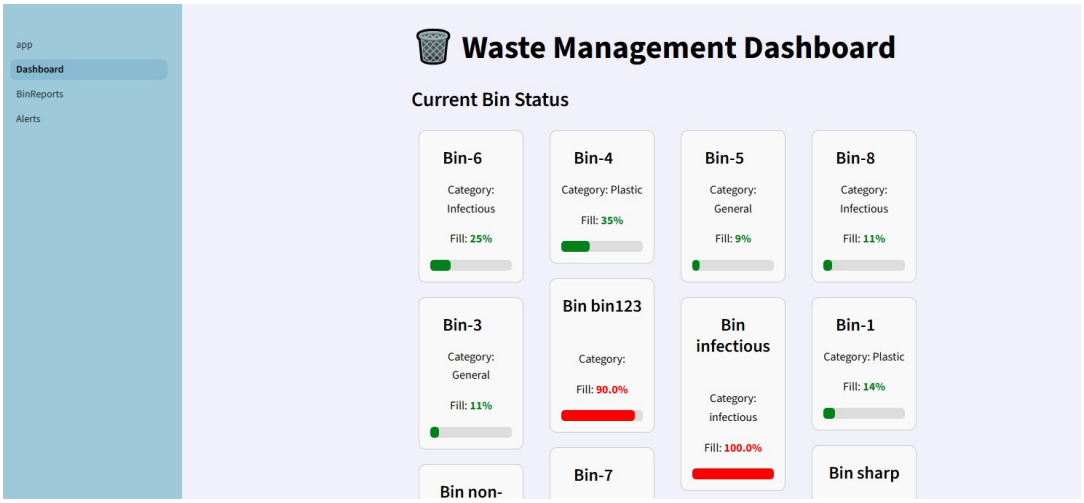


Figure 5.3: Dashboard Page Interface - Current Bin Status

To enhance situational awareness and support informed decision-making, the dashboard incorporates multiple graphical visualizations. A pie chart represents the proportional distribution of biomedical waste categories, including infectious, non-infectious, and sharp waste, providing an immediate overview of waste composition.

Figure 5.4 shows the waste distribution visualization.

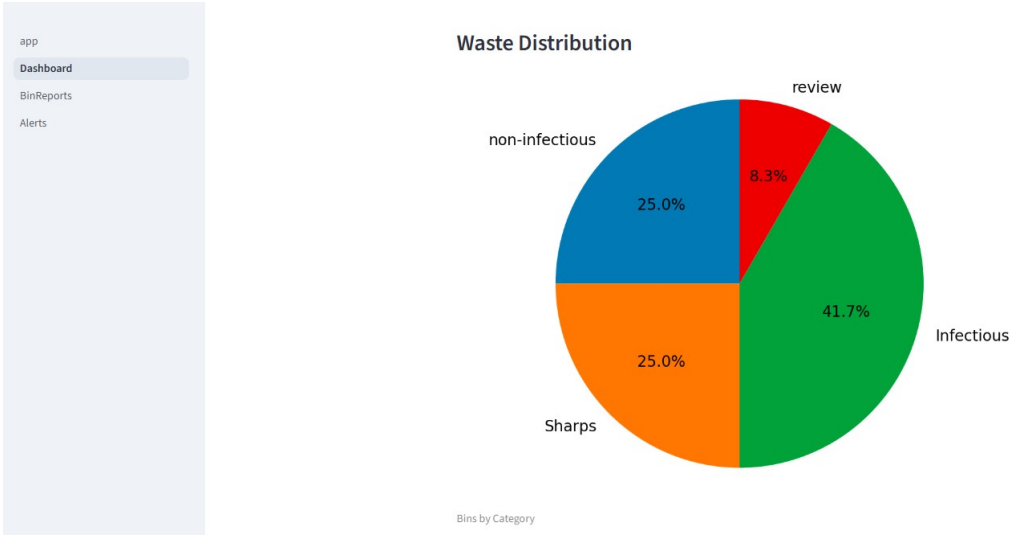


Figure 5.4: Dashboard Page Interface - Waste Distribution

In addition, a line chart displays predicted bin fill levels over a 24-hour horizon. This predictive visualization enables proactive monitoring of potential overflow conditions and supports timely waste collection and resource planning.

Figure 5.5 presents the predicted fill level analysis.

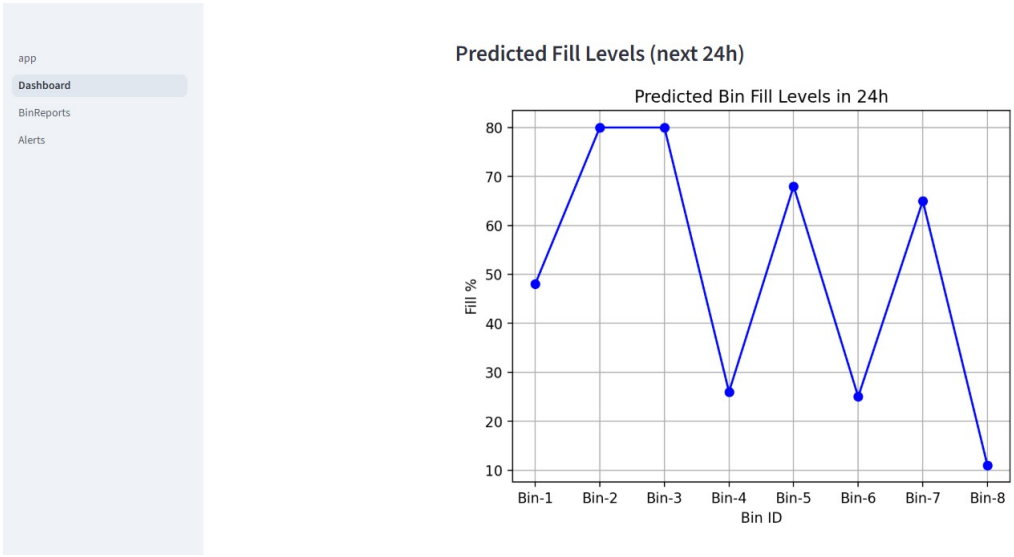


Figure 5.5: Dashboard Page Interface - Predicted Fill Levels

5.7.4 Bin Reports Page

The Bin Reports Page provides access to historical waste handling data for each bin in the system. Records are presented in a tabular format, allowing users to filter, sort, and analyze data based on time, bin type, or waste category. The interface also supports data export in CSV and Excel formats to facilitate offline analysis, reporting, and regulatory compliance.

Figure 5.6 shows the *Bin Reports Page* of the Streamlit web dashboard.

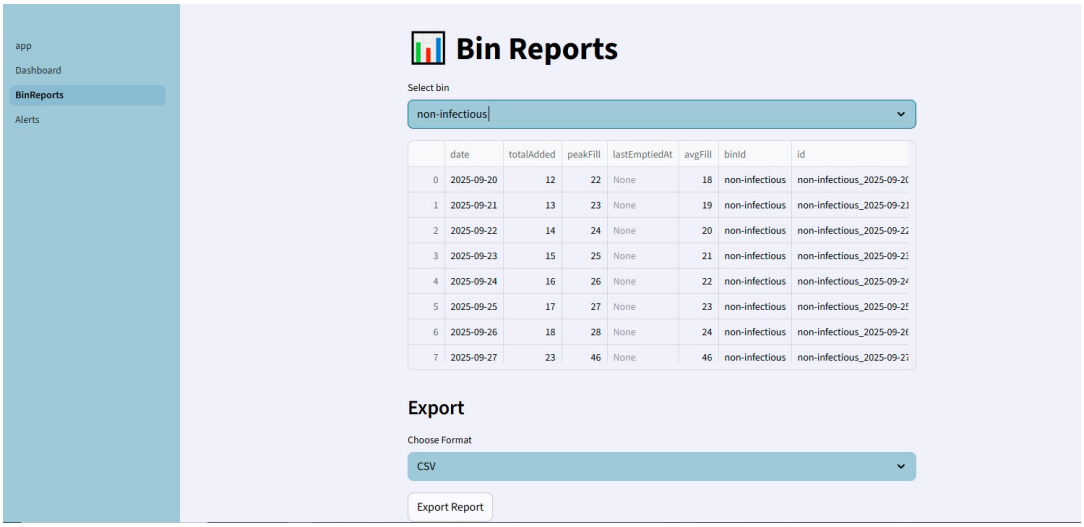


Figure 5.6: Bin Reports Page Interface

5.7.5 Alerts Page

The Alerts Page displays system-generated warnings related to critical operational conditions, such as predicted bin overflows, abnormal fill rate trends, or bins reaching critical thresholds. Alert data are retrieved from Firestore and presented in a structured list format, with visual indicators denoting alert severity. Each alert includes a *Resolve Alert* option, enabling authorized personnel to acknowledge and manage issues after appropriate corrective action.

Figure 5.7 illustrates the *Alerts Page* of the Streamlit web dashboard.

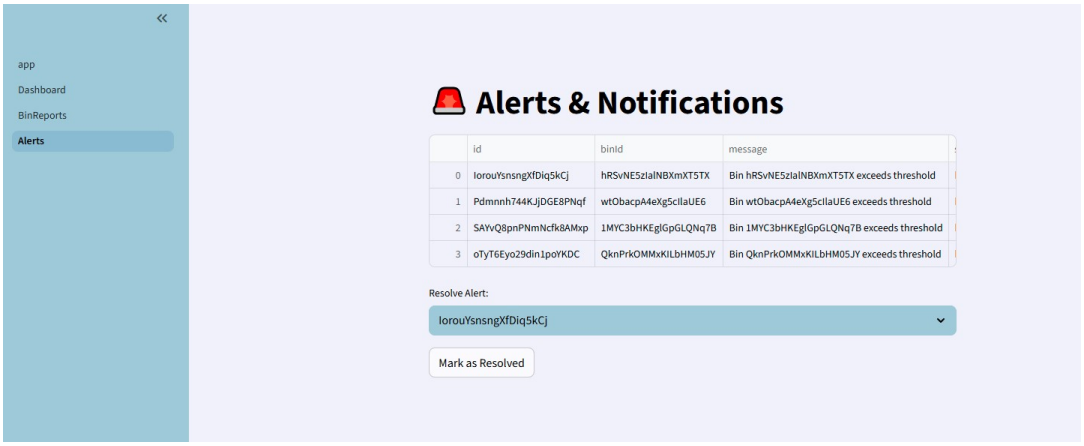


Figure 5.7: Alerts Page Interface

5.8 Summary

This chapter presents the complete implementation of the biomedical waste segregation system, detailing the configuration, programming logic, and integration of each subsystem. The implementation successfully realizes the design specifications through a Python centric environment incorporating deep learning based detection, robotic simulation, ultraviolet sterilization, and cloud enabled data management. The system functions as a modular, extensible, and reliable platform for automated biomedical waste handling in healthcare environments.

Chapter 6

System Testing and Evaluation

6.1 Introduction

System testing is the process of validating the complete and integrated Autonomous Biomedical Waste Segregation Robot against its specified functional and non-functional requirements. The objective is to determine whether the implemented system behaves correctly as a whole, interoperates with external services such as Firebase and the robotic arm simulator, and satisfies the expectations of its stakeholders.

This chapter provides the system level test cases for all primary functionalities, the overall traceability matrix, and the evaluation of performance, usability, reliability and robustness. The non-functional requirements that guide the evaluation include internet connectivity, environmental compatibility, hardware compatibility, security, usability, and scalability.

6.2 Testing Strategy

6.2.1 Testing Objectives

The main objectives of system testing for the Autonomous Biomedical Waste Segregation Robot are:

- To verify that all functional requirements are correctly implemented and that each use case executes as specified.
- To validate that the computer vision based classification pipeline, robotic segregation workflow, UV sterilization cycle and dashboard modules operate coherently as one integrated system.
- To evaluate the performance, usability, robustness and security characteristics of the prototype system.
- To identify residual defects, limitations and improvement opportunities before deployment in a real healthcare setting.

6.2.2 Test Levels and Types

Although unit and integration tests were performed during development, this chapter focuses on system testing. The following types of tests were planned at system level:

- Graphical user interface testing of the Streamlit-based dashboard.
- Functional testing based on FRs 3.3.1.
- Performance testing of the YOLOv8 inference pipeline and dashboard response time.
- Robustness and exception handling tests for camera failure, model loading failure and loss of network connectivity.
- Security testing of user authentication and access to administrative features.
- Installation and configuration testing to verify reproducible deployment on a new workstation.

6.3 Test Environment and Tools

System tests were executed on a typical development workstation equipped with Python, Ultralytics YOLOv8, Streamlit, Firebase client libraries and the robotic arm simulation environment. The ESP32 camera module streamed frames to the inference module through the configured local network. Firestore served as the central data store for detection results, bin fill levels, logs and reports.

The key tools and components used during system testing are summarised below.

- **YOLOv8 model**: Loaded using the Ultralytics framework for real-time waste classification.
- **Robotic arm simulator**: Receives classification results and simulates pick and place trajectories, including transfer through the UV chamber.
- **Firebase Firestore**: Stores detection records, bin status, timelines, alerts and report data.
- **Firebase Authentication**: Manages user accounts for hospital staff and system administrators.
- **Streamlit dashboard**: Provides the graphical user interface for monitoring and control.

All test data, such as sample waste images and log records, were separated into training, validation and testing subsets for the computer vision model so that evaluation results for the model were not biased by training data reuse.

6.4 Unit Testing

Unit level tests were conducted on isolated software functions including image preprocessing, Firebase write operations, and data parsing utilities. These tests validated correctness of individual modules before system integration. Only key functional units were tested since the project focuses primarily on integrated system behavior.

6.5 System Test Cases

The following tables present executed system-level test cases. Each test case includes objective, preconditions, steps, expected result, actual result, and status (Pass/Fail) based on execution outcomes.

6.5.1 STC01 User Login

This test case validates the authentication workflow defined in FR1 by confirming that only registered users can access the system. Successful login demonstrates correct integration between the Streamlit login interface and Firebase Authentication, and verifies that the dashboard is protected behind an authenticated session. The result also indicates that credential validation, token generation, and post-login routing operate correctly under normal conditions. The *User Login* test case specifications are documented in Table 6.1.

Table 6.1: STC01 User Login

Test Case ID	STC01
Objective	To verify that an authorized user can log in using Firebase Authentication and access the main dashboard.
Precondition	Registered user account exists in Firebase Authentication. Login page is displayed.
Steps	1. Enter a registered email address. 2. Enter the correct password. 3. Select the *Login* button.
Expected Result	User is authenticated through Firebase and redirected to the dashboard.
Actual Result	Authentication succeeded. Dashboard loaded with no errors.
Status	Pass

6.5.2 STC02 User Logout

This test case verifies compliance with FR2 by ensuring that an authenticated session can be terminated securely. The outcome confirms that session data and authentication tokens are cleared and that the system prevents unauthorized access after logout, including attempts to use browser navigation to return to protected pages. This demonstrates correct session control and supports

privacy requirements in shared hospital workstations. The *User Logout* test case specifications are documented in Table 6.2.

Table 6.2: STC02 User Logout

Test Case ID	STC02
Objective	To verify that the user session is terminated securely.
Precondition	User is logged in.
Steps	1. Select Logout option.
Expected Result	System clears session and displays login page.
Actual Result	Session was cleared. Login page displayed correctly. Browser back button prevented access.
Status	Pass

6.5.3 STC03 Waste Detection and Classification

This test case evaluates the end-to-end perception capability specified in FR3 by confirming that the system can process a live camera stream and produce correct classification output. Successful detection demonstrates correct integration of the ESP32 video feed, OpenCV frame acquisition, YOLOv8 inference, and category mapping logic. Logging of the detection record in Firestore further confirms traceability and real-time availability of results to downstream modules. The *Waste Detection and Classification* test case specifications are documented in Table 6.3.

Table 6.3: STC03 Waste Detection and Classification

Test Case ID	STC03
Objective	To validate real-time detection and classification from the ESP32 camera.
Precondition	YOLOv8 model is loaded. Camera feed is active.
Steps	1. Place mask in view. 2. Start detection cycle. 3. Observe classification on dashboard.
Expected Result	System detects and classifies as Infectious waste. Detection record stored in Firebase.
Actual Result	Object detected correctly with high confidence. Record stored in Firebase.
Status	Pass

6.5.4 STC04 Robotic Segregation

This test case validates FR4 by confirming that classification results are correctly translated into robotic handling actions within the simulation environment. The observed pick, transfer, and placement behavior demonstrates that the pipeline manager, robotic controller, and bin routing logic operate coherently. The successful completion of the motion sequence indicates that the system

can consistently execute the segregation workflow based on AI-generated waste categories. The *Robotic Segregation* test case specifications are documented in Table 6.4.

Table 6.4: STC04 Robotic Segregation

Test Case ID	STC04
Objective	To verify that the robotic simulator places waste in the correct bin.
Precondition	Simulator is connected. Classification result is available.
Steps	1. Trigger detection. 2. Allow classification to route waste. 3. Observe simulator path.
Expected Result	Item is transferred through UV chamber and placed in correct bin.
Actual Result	Simulator executed correct pickup, UV cycle, and placement.
Status	Pass

6.5.5 STC05 UV Sterilization

This test case verifies FR5 by confirming that the UV sterilization stage is executed as an intermediate step before disposal. The results demonstrate that the UV exposure routine activates, maintains the configured duration, and completes before the robot proceeds to final bin placement. This supports the system's safety objective by ensuring that each waste item undergoes a controlled disinfection step within the handling pipeline. The *UV Sterilization* test case specifications are documented in Table 6.5.

Table 6.5: STC05 UV Sterilization

Test Case ID	STC05
Objective	To ensure the system activates UV sterilization for the configured duration.
Precondition	Segregation process is active. UV chamber is initialised.
Steps	1. Trigger segregation. 2. Monitor UV indicator and timer.
Expected Result	UV turns on, remains active for set duration, and then turns off.
Actual Result	UV activated for full duration and turned off automatically.
Status	Pass

6.5.6 STC06 Dashboard Visualization

This test case validates FR6 by assessing whether the dashboard reflects system state changes in a timely and consistent manner. Successful updates after a new detection event confirm that Firestore operates as the system's single source of truth and that the dashboard retrieval logic correctly refreshes charts, tables, and status indicators. This demonstrates that users can reliably monitor system activity through the Streamlit interface during real-time operation. The *Dashboard Visualization* test case specifications are documented in Table 6.6.

Table 6.6: STC06 Dashboard Visualization

Test Case ID	STC06
Objective	To validate real-time visualization of system data.
Precondition	Data exists in Firebase. User is logged in.
Steps	1. Load dashboard. 2. Trigger new detection. 3. Observe updated charts.
Expected Result	Dashboard updates in real time.
Actual Result	Charts reflected new detection event immediately.
Status	Pass

6.5.7 STC07 Bin Fill Level Tracking

This test case evaluates FR7 by confirming that bin utilization metrics are updated correctly after each disposal event. The observed increase in fill percentage and correct threshold-based colour transitions demonstrate that the system's fill computation logic, Firestore update mechanism, and dashboard presentation are consistent. This supports proactive monitoring by ensuring that bin status reflects the most current operational data. The *Bin Fill Level Tracking* test case specifications are documented in Table 6.7.

Table 6.7: STC07 Bin Fill Level Tracking

Test Case ID	STC07
Objective	To verify that bin fill percentages update accurately after disposal events.
Precondition	Bin capacities are configured.
Steps	1. Note current fill percentage. 2. Execute disposal. 3. Observe fill update.
Expected Result	Percentage increases according to disposed volume. Indicators change at thresholds.
Actual Result	Percentage increased correctly and color indicator changed at threshold.
Status	Pass

6.5.8 STC08 Timeline and Data Logging

This test case validates FR8 by confirming that operational events are persistently logged and presented in chronological order. Successful creation of timeline entries demonstrates that key metadata such as timestamps, waste categories, and bin identifiers are recorded accurately. The correct ordering of events on the timeline view provides evidence of traceability and supports auditing and post-incident analysis requirements. The *Timeline and Data Logging* test case specifications are documented in Table 6.8.

Table 6.8: STC08 Timeline and Data Logging

Test Case ID	STC08
Objective	To verify that system events are logged and displayed chronologically.
Precondition	Logging module active.
Steps	1. Perform detection. 2. Perform segregation. 3. Check timeline view.
Expected Result	Events appear with correct timestamps in chronological order.
Actual Result	All events logged and displayed accurately with correct metadata.
Status	Pass

6.5.9 STC09 Predictive Analytics and Alerts

This test case verifies FR9 by evaluating whether the system can generate predictive insights and raise alerts based on forecasted bin fill trends. The results confirm that historical bin data is retrieved, processed by the prediction algorithm, and that alert conditions are triggered when forecasted values approach defined thresholds. This demonstrates that the system supports proactive decision-making rather than purely reactive monitoring. The *Predictive Analytics and Alerts* test case specifications are documented in Table 6.9.

Table 6.9: STC09 Predictive Analytics

Test Case ID	STC09
Objective	To verify alert generation based on predicted bin capacity levels.
Precondition	Historical bin data exists.
Steps	1. Trigger prediction module. 2. Observe alert panel.
Expected Result	System predicts fill time and issues alerts if capacity will be reached soon.
Actual Result	Forecast generated and alert displayed for bins reaching capacity soon.
Status	Pass

6.5.10 STC10 Report Generation and Export

This test case validates FR10 by confirming that authorized users can generate structured historical reports and export them in standard formats. Successful CSV and Excel export demonstrates that Firestore records are retrieved correctly, tabulated accurately, and converted into downloadable files without data loss. This capability supports compliance documentation, auditing workflows, and operational performance review. The *Report Generation and Export* test case specifications are documented in Table 6.10.

Table 6.10: STC10 Report Generation

Test Case ID	STC10
Objective	To verify that the user can generate and export historical records.
Precondition	Historical bin data is available.
Steps	1. Select bin. 2. View table. 3. Choose format. 4. Export report.
Expected Result	System downloads CSV or Excel file containing selected records.
Actual Result	CSV and Excel files generated successfully with complete records.
Status	Pass

6.5.11 STC11 Invalid Login Attempt

This test case validates the system's ability to handle authentication failures securely, ensuring that unauthorized users cannot access the dashboard. It evaluates the system behavior when incorrect login credentials are provided and confirms that access control mechanisms are correctly enforced, supporting the security requirements of the system. The *Invalid Login Attempt* test case specifications are documented in Table 6.11.

Table 6.11: STC11 Invalid Login Attempt

Test Case ID	STC11
Objective	To verify that the system denies access when incorrect login credentials are entered.
Precondition	Login page is displayed. User account exists in Firebase Authentication.
Steps	1. Enter a registered email address. 2. Enter an incorrect password. 3. Select the *Login* button.
Expected Result	System rejects credentials and displays an authentication error message. User is not redirected to the dashboard.
Actual Result	Login request was rejected. Error message displayed and dashboard access was denied.
Status	Pass

6.5.12 STC12 Camera Feed Failure

This test case evaluates the robustness of the system when the ESP32 camera feed becomes unavailable. It verifies that the detection pipeline handles the absence of input gracefully, displays appropriate error messages, and does not cause system crashes or inconsistent states. The *Camera Feed Failure* test case specifications are documented in Table 6.12.

Table 6.12: STC12 Camera Feed Failure

Test Case ID	STC12
Objective	To verify system behavior when the ESP32 camera feed is interrupted or unavailable.
Precondition	System is running and camera stream was previously active.
Steps	1. Disconnect or disable the ESP32 camera stream. 2. Observe system response on the dashboard.
Expected Result	System displays a camera error message and pauses detection without crashing.
Actual Result	Camera feed loss detected. Warning message displayed and detection resumed automatically when feed was restored.
Status	Pass

6.5.13 STC13 Firestore Write Failure

This test case evaluates system behavior when Firestore becomes temporarily unavailable during a write operation. It verifies that the system handles cloud storage failures gracefully without crashing and provides appropriate feedback to the user. The *Firestore Write Failure* test case specifications are documented in Table 6.13.

Table 6.13: STC13 Firestore Write Failure

Test Case ID	STC13
Objective	To verify system robustness when Firestore write operations fail.
Precondition	System is running. Firestore connection is temporarily unavailable.
Steps	1. Trigger a detection event. 2. Disable network or Firestore access. 3. Observe system response.
Expected Result	System displays a warning and retries or queues the operation without crashing.
Actual Result	Firestore write failure detected. Warning displayed and operation retried when connection restored.
Status	Pass

6.5.14 STC14 YOLOv8 Model Load Failure

This test case evaluates system behavior when the trained YOLOv8 model file is missing or corrupted. It ensures that the application fails gracefully and informs the user rather than terminating unexpectedly. The *YOLOv8 Model Load Failure* test case specifications are documented in Table 6.14.

Table 6.14: STC14 YOLOv8 Model Load Failure

Test Case ID	STC14
Objective	To verify system response when the YOLOv8 model fails to load.
Precondition	Model file is missing or corrupted.
Steps	1. Start system without valid model file. 2. Observe system startup behavior.
Expected Result	System displays error message and disables detection functionality.
Actual Result	Model loading error detected. Detection disabled and warning shown.
Status	Pass

6.5.15 Traceability Matrix

A traceability matrix is a document that maps and traces user requirements with test cases, ensuring that all requirements are covered by tests. It helps in tracking the progress and completeness of a project, ensuring that no requirements are missed during the testing phase. For completeness, Table 6.15 summarizes the traceability between FRs 3.3.1 and the corresponding system test cases.

Table 6.15: Traceability Summary

Test Objective	**Test Basis**	**Completion Criteria**
Verify system login process	FR1 User Login	User successfully accesses dashboard
Verify logout process	FR2 User Logout	Session ends and login page appears
Validate waste detection	FR3 Detection Pipeline	YOLOv8 detects and classifies correctly
Validate segregation workflow	FR4 Segregation Process	Waste placed in correct bin
Verify UV cycle	FR5 Sterilization Process	UV activates and completes duration
Verify dashboard updates	FR6 Dashboard View	Charts reflect real-time changes
Verify bin fill tracking	FR7 Bin Monitoring	Fill percentage updates correctly
Verify timeline logs	FR8 Logging Requirements	All events timestamped correctly
Verify predictive alerts	FR9 Analytics Module	Alerts generated before capacity
Verify report export	FR10 Reporting Process	CSV or Excel downloaded successfully

6.6 Non-Functional Testing and Evaluation

6.6.1 Performance Evaluation

The performance of the system was evaluated based on observed inference latency, dashboard responsiveness, and system stability during continuous operation. The YOLOv8 classification produced results with minimal perceptible delay and supported smooth frame processing from the ESP32 camera. The dashboard rendered updated values promptly following new detection or segregation events and maintained stable performance throughout testing.

The report generation process completed without noticeable delay. Performance evaluation verifies compliance with NFR1 Internet Connectivity, NFR3 Hardware Compatibility and NFR6 Scalability.

6.6.2 AI Model Evaluation

The performance of the YOLOv8-based biomedical waste detection model was quantitatively evaluated using standard object detection metrics. Evaluation was performed on the independent test subset of the dataset to ensure unbiased performance assessment.

A confusion matrix was generated to analyze class-wise prediction accuracy and misclassification patterns. The matrix demonstrates strong diagonal dominance, indicating that the majority of biomedical waste instances were correctly classified into their respective categories. Minor confusion was observed between visually similar classes, which is expected due to overlapping visual features.

Precision–Recall analysis further confirms the robustness of the trained model, with high precision indicating low false-positive rates and high recall indicating effective detection of relevant waste objects. These results validate the suitability of the YOLOv8 model for real-time biomedical waste segregation tasks.

6.6.3 Usability Evaluation

Usability testing can be performed by asking a small group of representative hospital staff or students to complete tasks that correspond to key use cases, such as monitoring bin levels, acknowledging alerts and exporting reports. Observations, completion times and subjective ratings on ease of use, clarity of visualization and perceived usefulness should be recorded.

A usability evaluation was conducted with one student acting as a representative user. The student attempted key tasks including logging in, monitoring bins, interpreting alerts and exporting a report. The observations were as follows.

- The dashboard layout was found intuitive with clear sections for detections, bin levels and logs.
- The color coded indicators helped the student quickly recognize critical states such as high bin fill levels.

- The reporting module was easy to locate and the export process required minimal effort.

Overall the student rated the system as easy to learn, visually clear and appropriate for a hospital environment which satisfies Usability.

6.6.4 Reliability and Robustness

Robustness testing involves intentionally triggering error conditions to verify system stability. The following tests were conducted.

- Temporary internet disconnection resulted in dashboard warnings. System recovered automatically when connection restored.
- Camera feed interruption generated an appropriate error message. Once the feed resumed the detection module recovered.
- Firestore write failures produced descriptive warnings without data corruption.

For each scenario, the system failed gracefully, displayed informative error messages and recovered when the fault condition was removed without corrupting stored data. These evaluations confirm that the system operates reliably under typical failure modes and returns to normal operation without manual intervention. Logging of such failures supports post incident analysis.

6.6.5 Security Considerations

Security testing concentrates on authentication and protection of clinical data. Tests include entering invalid credentials, attempting to bypass the login screen through direct URL access and verifying that unauthenticated users cannot see any detection or patient related information. Minimum password length and storage of API keys in configuration files rather than source code are also validated.

6.7 Installation and Configuration Testing

Installation testing was performed to verify correct system setup on a new workstation. The Firebase configuration, model loading, and dashboard dependencies were installed without error. The ESP32 camera was connected through the defined network configuration and streamed frames successfully. These results confirm that the installation procedure is reproducible and stable.

6.8 Critical Appraisal

The system testing and evaluation activities highlight both strengths and limitations of the current prototype.

- **Strengths**: The use case driven test design ensures coverage of all key user interactions, from detection to reporting. The modular architecture integrating YOLOv8, Firestore and Streamlit simplifies testing of individual components and supports clear traceability from requirements to tests. Automated logging provides a rich basis for audit trails and future analytics.
- **Limitations**: The robotic arm is currently evaluated in a simulated environment rather than on physical hardware, which limits conclusions about mechanical accuracy and real world timing constraints. Performance measurements are obtained on a specific workstation configuration; different hardware or network conditions may yield different results. Usability tests may involve a small sample size, which restricts generalization to large hospitals.

Future work should include hardware in the loop testing with an actual robotic arm, larger scale usability studies in partner hospitals, and more extensive stress tests under high event loads.

6.9 Summary

This chapter presented the system testing and evaluation of the Autonomous Biomedical Waste Segregation Robot. System testing demonstrated that the implemented prototype satisfies the major functional and non-functional requirements. Each test case confirmed the correct behavior of corresponding system functionalities. Non-functional testing addressed performance, usability, robustness and security. The critical appraisal clarified the strengths and current limitations of the prototype and identified directions for further enhancement.

Chapter 7

Conclusions

7.1 Summary of Work

This study presented the design and implementation of an Autonomous Biomedical Waste Segregation Robot that integrates computer vision, robotic actuation, ultraviolet sterilization, and IoT-enabled monitoring into a unified framework. The system employs a YOLOv8-based object detection model to classify biomedical waste into infectious, non-infectious, and sharp categories using real-time input from an ESP32 camera module. A Finite State Machine architecture governs the complete workflow, beginning with detection and classification and proceeding through robotic picking, ultraviolet exposure, and final placement into category-specific bins. A cloud-connected dashboard built with Streamlit and backed by Firestore supports real-time monitoring, historical data analysis, and bin-level tracking. The entire robotic manipulation process was validated in simulation using a Franka Panda model in the PyBullet environment, ensuring safe and repeatable evaluation before future deployment on physical hardware. Overall, the system demonstrates that automated biomedical waste segregation can be achieved through the coordinated integration of computer vision, robotic autonomy, and cloud-based monitoring, thereby addressing key safety and efficiency limitations of traditional manual waste handling practices.

7.2 Key Contributions

The thesis makes several contributions to both academic research and practical biomedical waste management applications.

- It demonstrates that low-cost computer vision hardware combined with a state-of-the-art deep learning model can achieve consistent classification performance under controlled conditions suitable for real-time biomedical waste segregation workflows.
- It provides a deterministic robotic control structure based on a Finite State Machine that coordinates perception, motion, and ultraviolet sterilization in alignment with safety-centred design principles.

- It introduces an Internet of Things monitoring architecture that supports traceability, data auditing, and remote supervision, which are essential for healthcare waste management compliance.
- It offers a complete end-to-end integration of detection, decision-making, robotic simulation, sterilization logic, and cloud monitoring, thereby serving as a reference implementation for future academic work in autonomous healthcare robotics.
- It contributes to practical applications by providing a scalable approach to reducing human contact with hazardous biomedical waste, ultimately improving occupational safety in clinical environments.

7.3 Challenges Faced

The development process encountered several technical and operational challenges.

- Real-time image capture using the ESP32 camera required optimization to ensure that latency did not degrade detection or robotic responsiveness.
- The YOLOv8 model required dataset balancing and iterative fine-tuning to achieve stable multi-class classification performance across varied waste types.
- Synchronizing perception and robotic actuation in simulation required precise calibration of coordinate frames and timing constraints.
- The ultraviolet sterilization logic introduced additional timing complexity because exposure duration varies according to waste category, requiring careful integration within the Finite State Machine.
- Network connectivity for the Internet of Things subsystem posed reliability concerns because real-time dashboard updates and alerts depend on uninterrupted communication.

7.4 Future Work

There are several directions in which this project can be extended to improve its utility and readiness for real-world deployment.

- Moving from simulation to a physical robotic arm with an enhanced gripper design capable of handling diverse waste geometries will significantly increase system applicability.
- Expanding the training dataset with real hospital waste images will improve model robustness under environmental variability, such as inconsistent lighting or partial occlusion.
- Integrating edge computing capabilities on an embedded platform will reduce dependence on continuous internet connectivity and enhance system resilience.

- Predictive analytics for bin fill level forecasting can be strengthened to support intelligent scheduling of waste collection operations.
- Additional sterilization verification methods, such as optical or thermal assessment, can be incorporated to further increase safety assurance.
- A complete role-based multi-user management system may be implemented to support larger clinical environments requiring controlled access to logs and analytics.

7.5 Final Thoughts

The project establishes a strong foundation for a next-generation biomedical waste management solution that combines intelligent perception, robotic autonomy, and cloud-based visibility. The integration of these technologies demonstrates that automated segregation is technically feasible and capable of reducing occupational hazards for healthcare personnel. While the current system serves as a functional prototype, the architecture and design decisions reflect industry-aligned engineering practices that can support future scaling and deployment. With continued refinement of both the mechanical and machine learning components, the system has the potential to contribute meaningfully to safer and more efficient biomedical waste handling within hospitals and research facilities. The outcomes of this work suggest that intelligent, autonomous waste handling systems can play a meaningful role in improving safety, efficiency, and accountability in future healthcare infrastructure.

Appendix A

User Manual

A.1 Introduction

This manual provides step-by-step instructions for operating the **Autonomous Biomedical Waste Segregation Robot** system via its web-based dashboard. The system is designed for hospital staff, waste management operators, and system administrators. No prior knowledge of robotics or AI is required to operate the system.

A.2 System Requirements

To operate the system effectively, the following requirements must be met:

- A computer, tablet, or smartphone with a modern web browser (Chrome, Firefox, Edge, etc.).
- Stable internet connection.
- Authorized user account registered in the system.
- Active connection to the camera and cloud database services.

A.3 User Roles

The system supports the following user role:

- **Operator**: Responsible for monitoring waste segregation activity, viewing bin status, checking alerts, and generating reports.

A.4 Accessing the System

A.4.1 Login

1. Open your web browser and navigate to the system's dashboard URL.

2. You will see the **Login Page** (Figure A.1).
3. Enter your registered **Email** and **Password**.
    - Use the eye icon to toggle password visibility.
4. Click the **Login** button.
5. Upon successful authentication, you will be redirected to the **Home Page**.

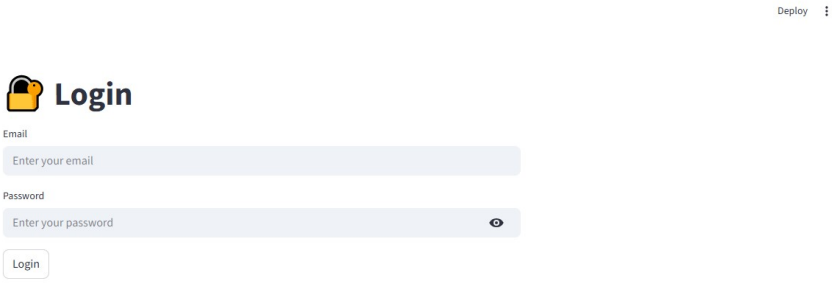


Figure A.1: Login Page

A.4.2 Home Page Navigation

After login, you will see the **Home Page** (Figure A.2) with the following options:

- **Dashboard** – Real-time monitoring and charts.
- **Reports** – View and export bin history.
- **Alerts** – Manage system notifications.
- **Logout** – End your session securely.

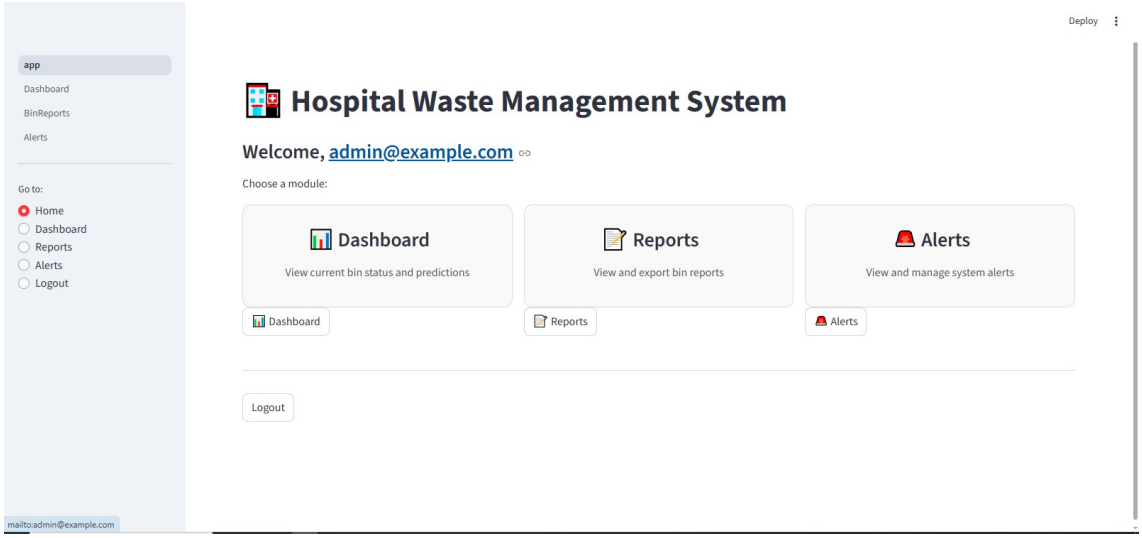


Figure A.2: Home Page

A.5 Dashboard Overview

A.5.1 Accessing the Dashboard

From the Home Page, click **Dashboard** or select it from the sidebar.

A.5.2 Dashboard Features

The Dashboard provides real-time insights into the waste segregation system.

A.5.2.1 Current Bin Status

Each bin is displayed with:

- **Category** (Infectious, Non-Infectious, Sharps, Review)
- **Fill Level** (%) with color indicators:
  - **Green**: Low fill (< 50%)
  - **Yellow**: Medium fill (50–79%)
  - **Red**: High fill ($\geq$ 80%)
- Example:
  - Bin-1: Infectious, Fill: 68% (Yellow)
  - Bin-3: Sharps, Fill: 13% (Green)

Figure A.3: Bin Status Section

A.5.2.2 Waste Distribution Pie Chart

A pie chart shows the **overall percentage** of waste in each category (Figure A.4).

- Hover over sections to see exact percentages.
- Options to toggle chart views are available in the chart menu.

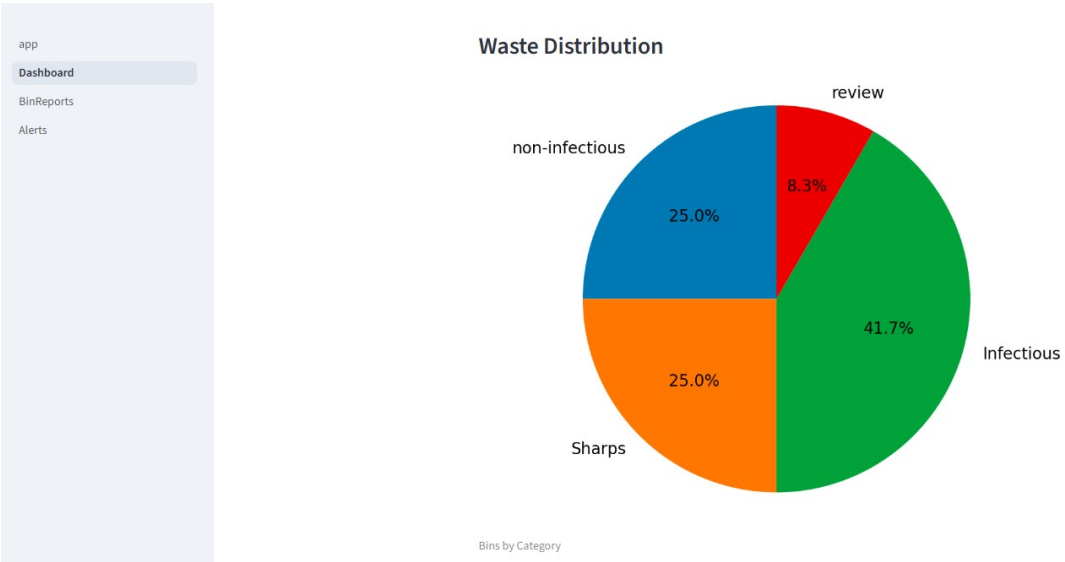


Figure A.4: Waste Distribution Pie Chart

A.5.2.3 Predicted Fill Levels (Next 24 Hours)

A line chart predicts bin fill levels for the next 24 hours (Figure A.5).

- Helps in proactive waste collection planning.

- Use chart controls to adjust the view or export the chart.

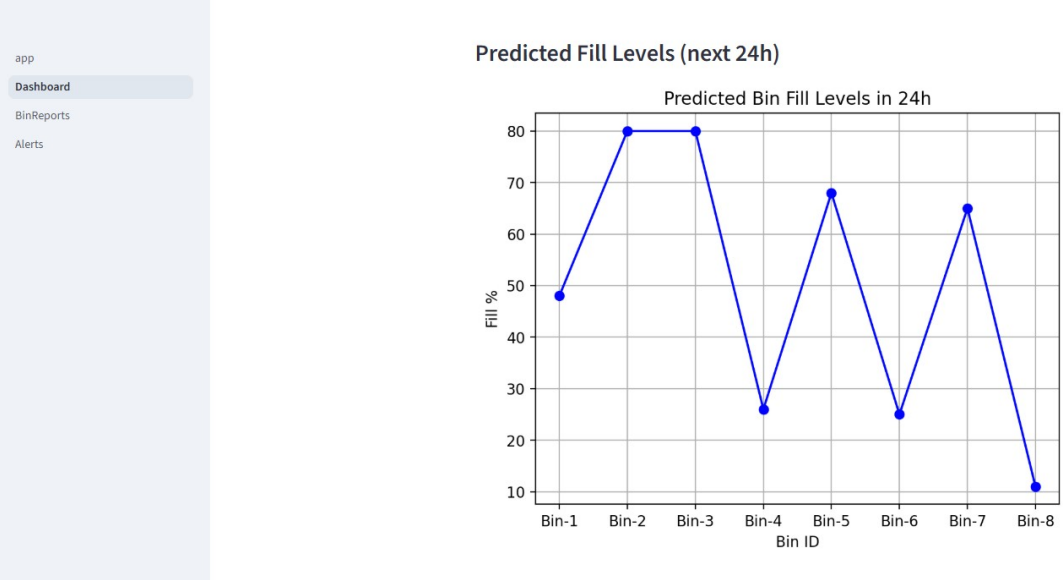


Figure A.5: Prediction Chart

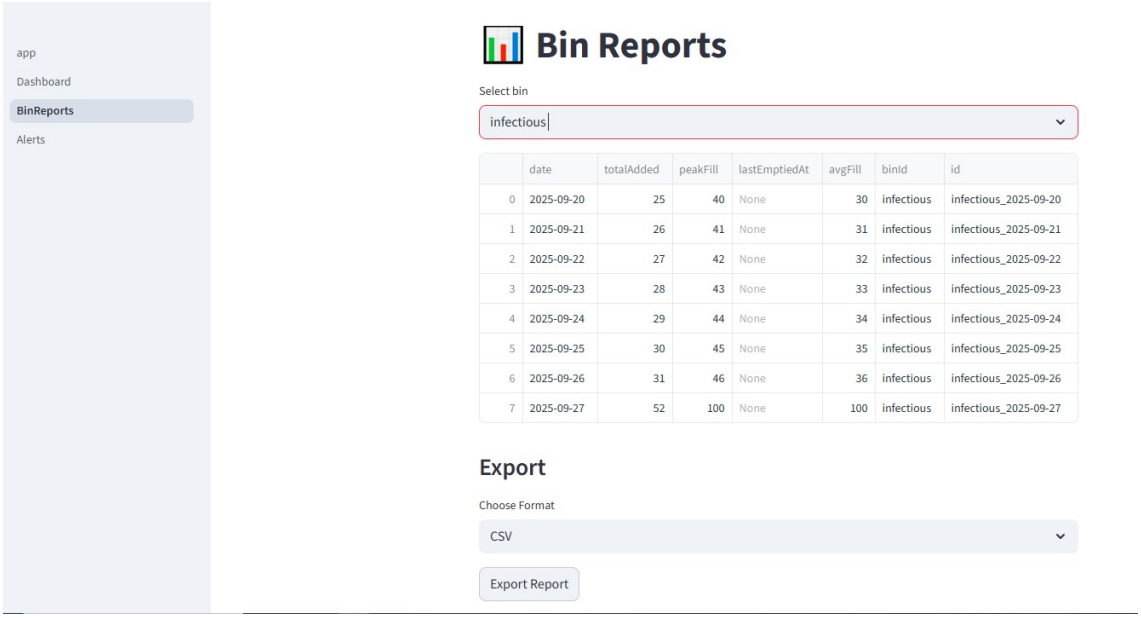
A.6 Generating Reports

A.6.1 Accessing Reports

From the Home Page or sidebar, click **Reports**.

A.6.2 Viewing Bin History

1. Use the **"Select bin"** dropdown to choose a specific bin (e.g., Infectious, Sharps, etc.).
2. The system displays a **table** with historical data, including:
    - Date
    - Total Added
    - Peak Fill
    - Last Emptied At
    - Average Fill
    - Bin ID



	date	totalAdded	peakFill	lastEmptiedAt	avgFill	binId	id
0	2025-09-20	25	40	None	30	infectious	infectious_2025-09-20
1	2025-09-21	26	41	None	31	infectious	infectious_2025-09-21
2	2025-09-22	27	42	None	32	infectious	infectious_2025-09-22
3	2025-09-23	28	43	None	33	infectious	infectious_2025-09-23
4	2025-09-24	29	44	None	34	infectious	infectious_2025-09-24
5	2025-09-25	30	45	None	35	infectious	infectious_2025-09-25
6	2025-09-26	31	46	None	36	infectious	infectious_2025-09-26
7	2025-09-27	52	100	None	100	infectious	infectious_2025-09-27

Figure A.6: Bin Report Table

A.6.3 Table Interaction

- **Sort**: Click column headers to sort ascending/descending.
- **Filter**: Use column filters to narrow data (Figure A.7).
- **Format**: Adjust column width, pin, or hide columns using the table menu.

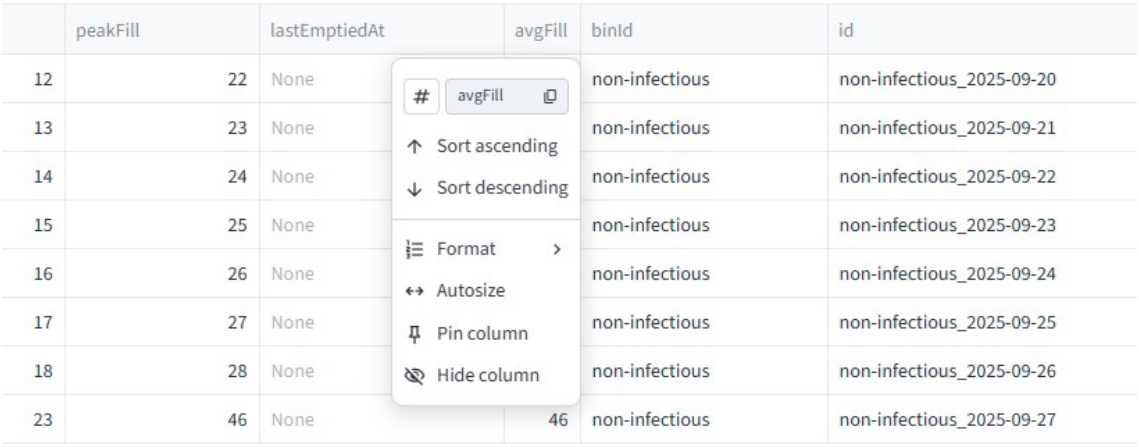


Figure A.7: Column Filter Options

A.6.4 Exporting Reports

1. Below the table, select the desired format: **CSV** or **Excel**.

2. Click **Export Report**.
3. The file will download to your device.

A.7 Managing Alerts

A.7.1 Accessing Alerts

From the Home Page or sidebar, click **Alerts**.

A.7.2 Viewing Active Alerts

A table lists all active alerts with:

- Alert ID
- Bin ID
- Message
- Severity (High, Medium, Low)
- Status (Active/Resolved)
- Created At (timestamp)

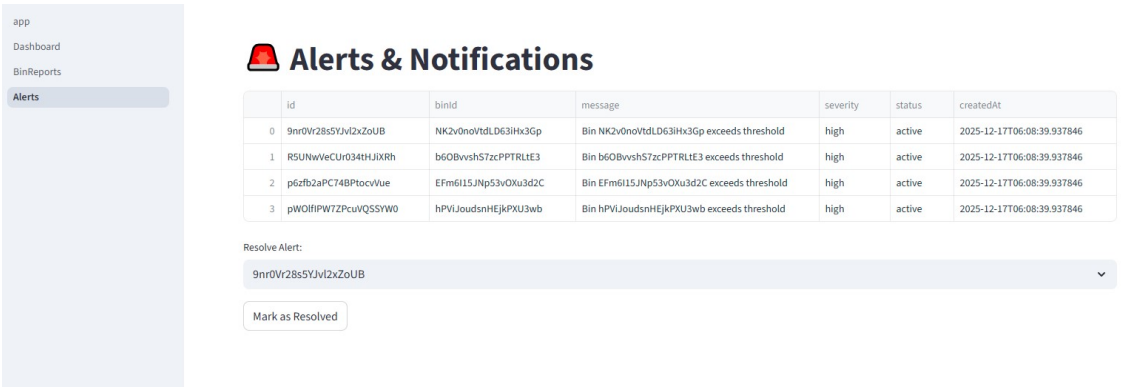


Figure A.8: Alerts Table

A.7.3 Resolving Alerts

1. In the **“Resolve Alert”** section, select an Alert ID from the dropdown.
2. Click **Mark as Resolved**.
3. The alert’s status will update to “Resolved” and disappear from the active list.

A.8 Logging Out

1. Return to the **Home Page** (via sidebar or back navigation).
2. Click the **Logout** button.
3. You will be redirected to the Login Page, and your session will end securely.

A.9 Basic Troubleshooting

Common issues and responses include:

Table A.1: Basic Troubleshooting Guide

Issue	**Possible Cause**	**Solution**
Login fails	Incorrect credentials / Network issue	Re-enter credentials; check internet.
Dashboard not updating	Camera feed lost / Cloud disconnect	Wait for auto-recovery; check network.
Export fails	Browser or format issue	Try another format; refresh page.
Alerts not clearing	System delay / Bug	Refresh page; contact admin.

For further assistance, contact the system administrator.

References

[1] World Health Organization. *Safe Management of Wastes from Health-Care Activities*. World Health Organization, Geneva, Switzerland, 2014. Cited on pp. 1 and 12.

[2] E. S. Windfeld and M. S.-L. Brooks. Healthcare waste management – a review. *Journal of Environmental Management*, 163:98–108, 2015. Cited on pp. 1 and 12.

[3] Y. Zhou, S. Li, and J. Wang. A deep learning approach for medical waste classification. *Scientific Reports*, 12(1):1–12, February 2022. Cited on pp. 6 and 8.

[4] P. Akkajit and P. Sukkuea. Medical waste classification using convolutional neural network. *Journal of Medical Systems*, 49(2):56–64, January 2025. Cited on pp. 6 and 8.

[5] S. van den Broek, L. Jansen, and M. Vermeer. Improving medical waste classification with hybrid capsule networks. *arXiv preprint arXiv:2503.10426*, pages 1–10, March 2025. Cited on pp. 6 and 8.

[6] M. Nahiduzzaman, T. Rahman, and M. Chowdhury. An automated waste classification system using deep learning and iot. *Knowledge-Based Systems*, 278:110987, January 2025. Cited on p. 6.

[7] A. Rakshita, P. Ramesh, and S. Devika. Biomedical waste sorting using yolo and uv c sterilization. In *Proceedings of the JETIR Conference on Robotics*, pages 112–118, Ahmedabad, India, May 2025. JETIR. Cited on pp. 7 and 8.

[8] X. Li, A. Kumar, and R. Desai. Multi class waste segregation using computer vision and robotic arm. *Automation in Construction*, 152:104897, 2024. Cited on pp. 7 and 8.

[9] S. Ahmed, F. Noor, and M. Hamza. A conveyor based automated waste sorter using vision techniques. *International Journal of Mechanical Engineering*, 18(1):23–31, 2023. Cited on pp. 7 and 8.

[10] V. Mehta, M. Joshi, and R. Pillai. Uv disinfection robots: A comprehensive review. *Sensors*, 22(20):7698, 2022. Cited on pp. 7 and 8.

[11] M. Diab-El Schahawi et al. Ultraviolet disinfection robots to improve hospital cleaning. *Antimicrobial Resistance and Infection Control*, 10(1):134, 2021. Cited on pp. 7 and 8.

[12] S. Haritha, S. Vadlamani, and K. Ram. Leveraging iot and deep learning for sustainable biomedical waste management. *International Journal of Innovative Applied Research*, 9(2):210–220, 2025. Cited on pp. 7 and 8.

[13] J. Vimal, P. Ravi, and A. George. Convowaste: Automatic waste segregation using deep learning. *arXiv preprint arXiv:2302.02976*, pages 1–12, February 2023. Cited on pp. 7 and 8.

[14] J. Park, S. Kim, and D. Lee. Wastenet: Waste classification at the edge for smart bins. *arXiv preprint arXiv:2006.05873*, pages 1–9, June 2020. Cited on pp. 7 and 8.

[15] Rapeepan Pitakaso, Surajet Khonjun, Thanatkij Srichok, Kanchana Sethanan, Natthapong Nanthasamroeng, Chawis Boonmee, Sarayut Gonwirat, and Peerawat Luesak. Pharmaceutical and biomedical waste, 2023. Cited on p. 52.