## **Table of Contents**

### **Chapter One: Introduction**

1.1 Background ................................................. 1

1.2 Problem Statement ...................................... 1

1.3 Proposed Solution ....................................... 2

1.4 Objectives .................................................. 2

1.4.1 General Objective .................................. 2

1.4.2 Specific Objectives ................................. 2

1.5 Scope ........................................................ 3

1.6 Project Justification ................................... 3

1.7 Research Methodology and System Methodology ...... 4

1.7.1 Research Methodology ........................... 4

1.7.2 System Development Methodology ........ 4

1.8 Project Requirements .................................. 5

1.8.1 Functional Requirements ....................... 5

1.8.2 Non-Functional Requirements ................ 6

### **Chapter Two: Literature Review**

2.1 Introduction ................................................ 7

2.2 Theoretical Review ...................................... 7

2.3 Research Area Application ........................... 8

2.3.1 Smart Waste Management Systems ...... 8

2.3.2 Camera-Based Waste Monitoring ........... 9

2.3.3 Optimization of Waste Collection ........... 9

2.4 Local Perspectives ........................................ 10

2.5 Integration and Architecture ......................... 10

### **Chapter Three: System Analysis and Design**

3.1 Introduction ................................................ 11

3.2 Feasibility Study .......................................... 11

3.2.1 Technical Feasibility ............................... 11

3.2.2 Economic Feasibility ............................... 12

3.2.3 Operational Feasibility ........................... 12

3.2.4 Legal Feasibility ...................................... 12

3.3 Requirements Analysis ................................ 13

3.3.1 Functional Requirements ....................... 13

3.3.2 Non-Functional Requirements ................ 13

3.4 Logical Design ............................................. 14

3.4.1 Use Case Diagram ................................ 14

3.4.2 System Architecture ............................... 14

3.4.3 Data Flow Diagrams ............................... 15

3.5 Physical Design ........................................... 15

3.5.1 Database Design .................................... 15

### **Chapter Four: Software Validation and Evolution**

4.1 Software Validation ..................................... 16

4.1.1 Unit Testing ........................................... 16

4.1.2 Integration Testing ................................. 16

4.1.3 User Acceptance Testing (UAT) ............. 16

4.2 Software Evolution ....................................... 17

4.2.1 Maintenance .......................................... 17

4.2.2 Future Enhancements ............................. 17

### **Chapter Five: Conclusion and Future Work**

5.1 Conclusion .................................................. 18

5.2 Future Work ............................................... 18

### **Glossary ............................................... 19**

### **Appendix**

1. Bin Status Detection Using OpenCV .................................... 19
2. Sending Notifications Using Django and REST API ................... 19

## **Chapter One: Introduction**

### **1.1 Background**

Waste management is a critical component of maintaining clean and healthy urban environments. With increasing urban populations, cities face growing challenges in managing the volume of waste produced. In many developing regions, the inefficiency of waste collection systems leads to irregular collection schedules, overflowing bins, and the accumulation of uncollected waste. These issues pose environmental and public health risks. This project aims to introduce a **Smart Waste Management System** that leverages cameras to monitor waste levels, thereby improving waste collection processes and reducing inefficiencies in the waste management cycle.

### **1.2 Problem Statement**

Currently, waste collection systems rely on static schedules rather than real-time data. As a result, bins often overflow before collection, contributing to unsanitary conditions, or are collected prematurely, wasting resources. There is a lack of efficient, real-time systems that can dynamically adjust collection schedules based on actual waste levels, particularly in settings where IoT hardware is not readily available. The challenge lies in creating a cost-effective system that addresses this issue using available technologies.

### **1.3 Proposed Solution**

This project proposes the development of a **camera-based Smart Waste Management System**. The system will use cameras installed in waste bins to monitor waste levels and send real-time data to a central management system. The system will then analyze this data to determine the optimal time for waste collection, ensuring that bins are neither overflowing nor emptied prematurely. By using cameras instead of IoT hardware, the project aims to provide a cost-effective solution that can be implemented in environments where IoT devices may not be accessible.

### **1.4 Objectives**

#### **1.4.1 General Objective**

The main objective of this project is to design and implement a Smart Waste Management System using camera technology to improve the efficiency of waste collection processes.

#### **1.4.2 Specific Objectives**

* To develop a waste monitoring system using camera technology to capture the fill levels of waste bins.
* To create a real-time waste level data transmission system that sends information to a central management platform.
* To optimize waste collection schedules based on real-time waste levels.
* To evaluate the performance of the system in reducing inefficiencies in the waste collection process.

### **1.5 Scope**

The scope of the Smart Waste Management System project is limited to the design, development, and implementation of a waste monitoring system using camera technology. The system will focus on detecting and recording the waste fill levels in bins and optimizing waste collection schedules based on this data. It will be implemented in urban areas where the lack of IoT hardware poses a challenge. However, this project will not cover the actual waste collection logistics, as it focuses on improving the decision-making process for waste collection teams.

### **1.6 Project Justification**

Efficient waste management is crucial for environmental sustainability and public health. Overflowing bins can result in unsanitary conditions, attracting pests and spreading diseases. Premature collection, on the other hand, leads to unnecessary use of resources such as fuel, labor, and time. This project is justified by its potential to provide a low-cost and accessible solution using existing camera technology to improve the efficiency and reliability of waste collection services. The system will benefit local governments and waste management companies by enabling better resource allocation, reduced operational costs, and improved urban sanitation.

### **1.7 Research Methodology and System Methodology**

#### **1.7.1 Research Methodology**

The research methodology for this project involves both qualitative and quantitative approaches:

* **Qualitative Research**: This will include interviews with stakeholders such as city planners, waste collection services, and urban residents to gather insights on existing waste management challenges and needs.
* **Quantitative Research**: Data will be collected from case studies, previous projects, and academic papers to quantify the impact of waste management inefficiencies and assess the potential benefits of a smart waste management system.

#### **1.7.2 System Development Methodology**

The system will be developed using the **Agile Software Development Methodology**. Agile allows for iterative development, which is essential for continuously improving the system based on real-time feedback from stakeholders. This methodology is ideal for projects like this, where user requirements may evolve during development. The system will undergo several iterations, including:

* **Planning**: Understanding project requirements and setting clear milestones.
* **Design**: Defining system architecture and technical design specifications.
* **Implementation**: Writing code for the camera monitoring system, data processing, and scheduling optimization.
* **Testing**: Conducting unit, integration, and user acceptance testing to ensure functionality and accuracy.
* **Deployment**: Releasing the system for live use and gathering feedback for further refinement.

### **1.8 Project Requirements**

#### **1.8.1 Functional Requirements**

* The system should monitor waste levels in bins using camera feeds.
* The system must analyze waste level data in real-time.
* The system should provide alerts or notifications when a bin reaches a predefined threshold.
* The system must allow waste collection schedules to be dynamically updated based on real-time data.
* The system should generate reports on waste collection efficiency.

#### **1.8.2 Non-Functional Requirements**

* **Reliability**: The system must provide accurate real-time data on waste levels.
* **Scalability**: The system should handle multiple camera feeds from different locations without performance degradation.
* **Usability**: The system interface should be user-friendly for operators managing the waste collection schedules.
* **Security**: Data collected by the system should be securely stored and transmitted.
* **Performance**: The system should be responsive, providing data and generating schedules without significant delays.

## **Chapter Two: Literature Review**

### **2.1 Introduction**

Waste management is a critical aspect of urban planning and environmental sustainability. Traditional waste management systems rely on pre-determined schedules and manual monitoring, often leading to inefficiencies such as premature waste collection or overflow of waste bins. Smart waste management systems leverage technology to optimize waste collection, reducing costs, improving resource utilization, and minimizing environmental impact. This chapter reviews relevant literature, focusing on the theoretical foundations of waste management, smart systems, the application of camera-based waste monitoring, and how these systems are applied globally and locally.

### **2.2 Theoretical Review**

The theoretical framework for smart waste management draws from several fields, including:

* **Systems Theory**: Waste management is viewed as a system where various inputs (waste production, transportation, and disposal) interact dynamically. Optimizing these interactions leads to better overall system performance.
* **Optimization Algorithms**: Efficient waste collection is closely linked to route optimization algorithms such as the **Vehicle Routing Problem (VRP)**, which minimizes travel distance and time while ensuring all bins are emptied.
* **Environmental Sustainability**: Smart waste management systems contribute to sustainable practices by reducing the carbon footprint associated with waste collection and promoting recycling.

### **2.3 Research Area Application**

#### **Smart Waste Management Systems**

Smart waste management systems are designed to monitor and control waste collection more efficiently than traditional systems. These systems often rely on sensors, GPS, and data analytics to provide insights into bin fill levels and optimize collection routes. However, due to the high costs of deploying IoT sensors, alternatives such as camera-based systems are gaining attention. Several studies highlight the potential of image recognition techniques, using cameras, to monitor waste bin levels, providing a cost-effective solution.

#### **Camera-Based Waste Monitoring**

Research into camera-based systems shows that advancements in **machine vision** and **image processing** have made it possible to extract accurate data from images. Techniques such as **image segmentation** and **pattern recognition** allow systems to identify waste levels by analyzing bin contents. Open-source libraries like **OpenCV** are often used for this purpose, making the implementation accessible to developers with limited resources.

#### **Optimization of Waste Collection**

Several studies focus on optimizing waste collection routes to save on fuel, reduce labor costs, and improve service reliability. Route optimization is typically done using algorithms like **Genetic Algorithms (GA)** and **Ant Colony Optimization (ACO)**, which have been successfully applied to minimize the distance and time for collection routes.

### **2.4 Local Perspectives**

In many developing countries, waste management remains a significant challenge due to limited infrastructure and financial constraints. In cities like Nairobi, waste collection is often sporadic, with insufficient bin monitoring leading to overflowing waste in public areas. Despite these challenges, there is growing interest in adopting technology-driven solutions. By using cameras instead of IoT sensors, smart waste management systems can be implemented with lower upfront costs, making them more feasible for cities with constrained budgets.

### **2.5 Integration and Architecture**

The integration of camera systems into waste management infrastructures requires a well-thought-out architecture. Previous works show that a combination of real-time image analysis, data analytics, and cloud-based platforms can significantly enhance waste management efficiency. The following components form the core of smart waste management systems:

* **Data Collection**: Cameras capture bin images at regular intervals, feeding into the system for analysis.
* **Data Processing**: Machine vision techniques process these images to assess the waste levels.
* **Decision Support**: Based on the processed data, the system generates optimal collection schedules and routes.
* **User Interface**: Operators monitor the system via dashboards, viewing bin statuses and adjusting schedules as needed.

The combination of these elements forms a robust solution for managing urban waste efficiently, especially in regions with limited IoT capabilities.

## **Chapter Three: System Analysis and Design**

### 3.1 Introduction

The system analysis and design phase is crucial in ensuring that the Smart Waste Management System (SWMS) is built with the right specifications to meet user and project requirements. This chapter provides a comprehensive look at how the system's feasibility was assessed, how the requirements were gathered, and the design processes that were applied to create both the logical and physical architecture of the SWMS. The system aims to address inefficiencies in waste collection through camera-based monitoring and notification systems.

### **3.2 Feasibility Study**

The feasibility study evaluates whether the Smart Waste Management System is viable in terms of technical, economic, operational, and legal aspects. This ensures that all important factors are considered before further development.

#### **3.2.1 Technical Feasibility**

The technical feasibility evaluates the technologies required to build the system, such as the software, hardware, and technical expertise. The primary challenge of this project, in the absence of IoT hardware, is integrating camera technology for bin level monitoring, coupled with the Django framework for backend processing and notifications.

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Technology/Tools Used** | **Expertise Required** | **Availability** |
| Frontend | Django templates | Intermediate | Available |
| Backend | Django, REST API | Intermediate | Available |
| Image Processing | OpenCV for camera input | High | Available |
| Database | PostgreSQL | Intermediate | Available |
| Camera | Phone cameras for image capture | Basic | Readily Available |

Since the technologies required are all accessible and within our technical capabilities, the project is deemed technically feasible.

#### **3.2.2 Economic Feasibility**

Economic feasibility involves evaluating the project's cost-effectiveness. Given that we are using open-source software (Django, PostgreSQL, OpenCV) and already available hardware (phone cameras), the project is expected to be financially viable.

|  |  |  |
| --- | --- | --- |
| **Cost Category** | **Estimated Cost** | **Justification** |
| Software Tools | Free (open-source) | Django, PostgreSQL, OpenCV |
| Hardware | N/A | Existing phone cameras |
| Server Hosting | Minimal (free tier services) | Services such as Render or Heroku |

By using free tools and avoiding the cost of IoT devices, we minimize costs and ensure that the project can be completed within a limited budget.

#### **3.2.3 Operational Feasibility**

Operational feasibility looks at how easily the SWMS can be integrated into existing waste management processes. The use of camera-based monitoring ensures that waste collection teams only collect full bins, optimizing the process and reducing operational costs.

|  |  |
| --- | --- |
| **Operational Area** | **Description** |
| User Training | Minimal training for personnel to use the notification system. |
| System Maintenance | Periodic updates to the software and occasional calibration of cameras. |
| Workflow Impact | Increased efficiency with real-time monitoring and optimized collection routes. |

Given that only basic training is required and the system improves current workflows, the operational feasibility is high.

#### **3.2.4 Legal Feasibility**

Legal feasibility assesses the compliance of the system with local and national regulations, particularly around data privacy and the use of cameras in public spaces.

|  |  |
| --- | --- |
| **Legal Aspect** | **Compliance Strategy** |
| Data Privacy | Ensure cameras focus only on bins and not public spaces |
| Data Retention | Store only necessary data and adhere to retention policies |
| Regulatory Compliance | Research and adhere to local data protection and privacy laws |

Ensuring compliance with data privacy regulations is critical. The cameras will be positioned to avoid capturing unnecessary personal information, ensuring legal feasibility.

### **3.3 Requirements Analysis**

This section identifies the functional and non-functional requirements for the Smart Waste Management System. The requirements provide a clear outline of what the system must accomplish and the standards it should meet.

#### **3.3.1 Functional Requirements**

Functional requirements define the core operations that the SWMS must perform to fulfill its objectives.

* **Image Capture:** The system should capture images of waste bins at regular intervals or when triggered.
* **Bin Status Detection:** Analyze the images to determine the fill levels of bins (e.g., full, half-full, or empty).
* **Notification System:** Notify waste management personnel when a bin is detected to be full.
* **Route Optimization:** Generate optimized routes based on which bins are full and need emptying.
* **Admin Interface:** Provide a user-friendly interface for system administrators to manage bins and monitor system status.

#### **3.3.2 Non-Functional Requirements**

Non-functional requirements describe how the system will perform and its quality attributes.

* **Usability:** The system must have an easy-to-navigate interface for both personnel and administrators.
* **Performance:** Real-time notifications should be sent within 5 seconds of detecting a full bin.
* **Scalability:** The system must be scalable to handle an increasing number of bins as required.
* **Security:** The system should be secure, with proper authentication for access and secure storage for collected data.
* **Reliability:** The system should operate reliably with minimal downtime.

#### **3.4 Logical Design**

The logical design of the system focuses on how different components of the SWMS interact with each other. This includes designing system architectures, data flows, and user interactions.

#### **3.4.1 Use Case Diagram**

A use case diagram demonstrates the interaction between users and the system. The key actors in this project include:

* **Waste Management Personnel**: View bin status and receive notifications.
* **System Administrator**: Manage bin data, view system reports, and adjust settings.

The key use cases are:

* View bin status
* Receive notifications
* Generate optimized routes
* Manage bins

(A visual representation of the use case diagram is provided in the appendix.)

#### **3.4.2 System Architecture**

The system architecture provides a high-level overview of how components of the SWMS work together. It consists of:

* **Frontend (Django templates)**: Displays bin statuses, notifications, and routes.
* **Backend (Django with REST API)**: Handles data processing, including image uploads, bin status analysis, and sending notifications.
* **Image Processing (OpenCV)**: Analyzes images captured by the camera to determine the bin fill level.
* **Database (PostgreSQL)**: Stores bin data, images, user information, and route information.

This architecture ensures a smooth flow of data from image capture to notification, enabling an efficient and scalable system.

#### **3.4.3 Data Flow Diagrams (DFD)**

The Data Flow Diagram (DFD) visually represents how data moves through the system. In SWMS, data flows as follows:

1. **Image Capture:** Camera captures an image of the bin.
2. **Image Processing:** OpenCV processes the image to assess the bin’s fill level.
3. **Data Storage:** Bin status is stored in the database.
4. **Notification:** If a bin is full, a notification is sent to waste management personnel.
5. **Route Optimization:** Routes are optimized based on the status of bins.

(A visual representation of the DFD is included in the appendix.)

### **3.5 Physical Design**

The physical design details the actual implementation of the system, including the database design and hardware components.

#### **3.5.1 Database Design**

The database design involves setting up the necessary tables to store system data. Key tables include:

* **Bins Table:** Stores data about each waste bin, such as bin ID, location, and status (full, half-full, or empty).
* **Images Table:** Stores the images captured by the camera along with timestamps.
* **Notifications Table:** Stores records of notifications sent to personnel, including the bin ID and time.
* **Routes Table:** Stores optimized waste collection routes.

Each table is structured to ensure that data is easily retrievable, allowing for efficient management and operation of the system.

## **Chapter Four: Software Validation and Evolution**

### **4.1 Software Validation**

Software validation ensures that the Smart Waste Management System (SWMS) meets all the specified requirements and functions as intended. It is a crucial step to ensure the system's reliability, security, and performance. Validation involves testing the system at various stages, from unit testing to system testing, and gathering feedback from users.

#### **4.1.1 Unit Testing**

Unit testing involves testing individual components or units of the system to ensure that each one functions correctly in isolation. In the case of the SWMS, unit testing would involve testing components such as:

* **Image Capture Module**: Verifying that images are captured at appropriate intervals and stored correctly.
* **Bin Status Detection Module**: Testing the accuracy of image analysis in determining the bin’s fill level.
* **Notification System**: Ensuring notifications are triggered and sent correctly when a bin is detected as full.

Unit testing will be conducted during development, ensuring that bugs are caught and corrected early.

#### **4.1.2 Integration Testing**

Integration testing checks how different modules work together when combined. For the SWMS, integration testing focuses on the interaction between the image capture, bin status detection, and notification modules. Tests include:

* **Data Flow Between Components**: Ensuring images are successfully passed from the camera module to the image processing module.
* **Real-Time Notification System**: Testing that a notification is sent to personnel when the bin status is full, and route optimization data is updated accordingly.

Integration testing ensures that the system functions seamlessly as a whole.

#### **4.1.3 User Acceptance Testing (UAT)**

User Acceptance Testing (UAT) involves testing the system in a real-world scenario with users. Waste management personnel will test the system to verify that it meets their needs, including:

* **Interface Usability**: Ensuring that the user interface is easy to navigate for both personnel and administrators.
* **Notification Timeliness**: Confirming that notifications are received promptly when bins are full.
* **Route Optimization**: Verifying that the system generates efficient routes based on bin statuses.

Any feedback from users during UAT will be used to make improvements to the system.

### **4.2 Software Evolution**

Software evolution refers to the process of updating and improving the system over time to meet changing needs or fix emerging issues. After the initial deployment, the SWMS will require periodic updates and maintenance.

#### **4.2.1 Maintenance**

System maintenance will involve regular updates and bug fixes to ensure smooth operation. Maintenance tasks include:

* **Bug Fixes**: Addressing any issues that arise after deployment.
* **System Updates**: Updating the software libraries, frameworks, or databases as needed to maintain compatibility and security.
* **Performance Tuning**: Monitoring the system to identify bottlenecks and improve performance as necessary.

#### **4.2.2 Future Enhancements**

Future enhancements will aim to improve the functionality of the system over time. Potential upgrades include:

* **IoT Integration**: Adding IoT sensors to automatically detect bin levels without relying on image processing, making the system more autonomous.
* **Mobile App Development**: Creating a mobile app for waste management personnel to access notifications and routes on the go.
* **AI and Machine Learning**: Implementing AI to improve image recognition accuracy and predict future waste levels based on historical data.

The evolution of the system will ensure that it remains relevant and continues to provide value as technology and user needs change.

## **Chapter Five: Conclusion and Future Work**

### **5.1 Conclusion**

The Smart Waste Management System provides a comprehensive solution for optimizing waste collection by using camera-based monitoring to detect bin statuses in real-time. By ensuring that waste management personnel are only notified when bins are full, the system reduces operational inefficiencies, minimizes collection costs, and promotes sustainability. Throughout this project, we have conducted a thorough feasibility study, analyzed system requirements, and designed a scalable solution that meets the needs of both the public and waste management personnel.

The SWMS has been designed with flexibility in mind, ensuring that it can be integrated into current waste management workflows with minimal disruption. Additionally, the use of open-source tools and existing hardware has made the system economically viable and accessible. Moving forward, the system is well-positioned to evolve through additional features, such as IoT integration and AI-based predictive analytics.

### **5.2 Future Work**

As we look to the future, several areas for enhancement and further development of the Smart Waste Management System have been identified:

* **IoT-Based Sensors**: Replacing the current camera-based monitoring system with IoT sensors that automatically detect bin levels would improve system efficiency and accuracy.
* **Mobile Applications**: Developing mobile applications for both waste management personnel and system administrators would allow for greater flexibility and accessibility.
* **Data Analytics**: Implementing advanced data analytics and AI algorithms to predict future waste levels, helping to further optimize collection routes and schedules.
* **Scalability**: As more cities adopt the system, scaling the infrastructure to support additional bins, personnel, and regions will be necessary to maintain system performance.

By continuously improving the system and incorporating new technologies, the SWMS will remain a valuable tool in creating cleaner and more efficient urban environments.

### **Glossary**

* **Bin Status**: The fill level of a waste bin, which is detected using cameras or sensors.
* **Camera-Based Monitoring**: A method of monitoring waste bin levels using images captured by cameras.
* **Data Flow Diagram (DFD)**: A graphical representation of how data flows through a system.
* **Functional Requirements**: The specific features and functions that the system must perform.
* **IoT (Internet of Things)**: A network of physical objects (in this case, waste bins) equipped with sensors, software, and other technologies that enable them to collect and exchange data.
* **Non-Functional Requirements**: The performance, security, and usability requirements of the system.
* **Route Optimization**: The process of generating the most efficient routes for waste collection based on bin statuses.
* **System Architecture**: The structure of the system, showing how its components interact and function together.
* **Use Case Diagram**: A diagram that shows the interaction between users and a system, representing different ways the system will be used.

### Appendix

***Questionnaire Used for Requirements Elicitation***

This questionnaire was created to gather valuable insights from key stakeholders, such as waste management personnel and local authorities, to guide the development of the Smart Waste Management System. The questions were designed to understand the challenges, needs, and potential improvements for waste collection processes.

1. What challenges do you currently face in waste collection and management?
2. How do you determine when bins are full and need to be emptied?
3. Would a real-time notification system for full bins improve your current workflow?
4. What factors contribute to inefficiencies in your current waste collection system?
5. Would route optimization based on bin status be useful to you? If yes, how?
6. How often are waste bins monitored or checked for their status?
7. What other features or functionalities would you find helpful in a waste management system?
8. Are there any regulatory requirements that the waste management system should comply with?
9. What security measures would be important for a system managing public waste data?
10. Do you foresee any challenges in integrating the system into your existing workflow?

This questionnaire provided a foundational understanding of user needs and system requirements, helping us to tailor the system effectively to its end users.

***Diagrams and Tables***

* Use Case Diagram
* Data Flow Diagram (DFD)
* System Architecture Diagram

***Code Snippets***

#### **1. Bin Status Detection Using OpenCV**

# Import necessary libraries

import cv2

import numpy as np

def detect\_bin\_status(image\_path):

"""

Function to detect bin status (full/empty) based on the image of the bin.

Args:

image\_path (str): Path to the bin image.

Returns:

str: Status of the bin ('Full' or 'Empty').

"""

# Load the image

image = cv2.imread(image\_path)

# Convert the image to grayscale

gray\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Apply threshold to detect bin fill level

\_, thresholded = cv2.threshold(gray\_image, 128, 255, cv2.THRESH\_BINARY)

# Calculate the number of white pixels (representing empty space)

white\_pixels = cv2.countNonZero(thresholded)

# Calculate the number of total pixels

total\_pixels = thresholded.size

# Calculate the fill percentage

fill\_percentage = (white\_pixels / total\_pixels) \* 100

# Set a threshold to determine if the bin is full or empty

if fill\_percentage < 30:

return "Full"

else:

return "Empty"

# Example usage

image\_path = "path/to/bin\_image.jpg"

status = detect\_bin\_status(image\_path)

print(f"The bin is {status}.")

#### **2. Sending Notifications Using Django and REST API**

# views.py

from django.http import JsonResponse

from django.core.mail import send\_mail

def send\_bin\_status\_notification(request, bin\_status):

"""

View to send notifications when the bin is full.

Args:

request (HttpRequest): The HTTP request object.

bin\_status (str): Status of the bin ('Full' or 'Empty').

Returns:

JsonResponse: Success message.

"""

if bin\_status == 'Full':

# Example: Send email notification

send\_mail(

'Waste Bin Full Notification',

'A waste bin is full and needs to be emptied.',

'[system@example.com](mailto:system@example.com)', # From email

['[recipient@example.com](mailto:recipient@example.com)'], # To email

fail\_silently=False,

)

# You can add additional notification methods here (e.g., SMS, push notifications)

return JsonResponse({'message': 'Notification sent successfully.'})

# urls.py

from django.urls import path

from . import views

urlpatterns = [

path('send-notification/<str:bin\_status>/', views.send\_bin\_status\_notification, name='send\_notification'),

]