**CEBU INSTITUTE OF TECHNOLOGY**

**UNIVERSITY**

COLLEGE OF COMPUTER STUDIES

Software Requirements Specifications

for

**Garbage Management System**

**Change History**

**⦿** Added

⦿ OnProgress

**⦿** Removed

| **Revision No.** | **Changes** | **Date** | **Changed By** |
| --- | --- | --- | --- |
| 1 | **⦿ Functional Requirements** | 21/2025 | Paraiso, Gella, Largo |
| 2 | **⦿ Purpose**  **⦿ Scope**  **⦿ Definition**  **⦿ References**  **⦿ Product Perspective  ⦿ User Characteristics**  **⦿ Constraints**  **⦿ Assumptions & Dependecies  ⦿ External Interface**  **⦿ Non-Functional Requirements** | 2/28/2025 | Paraiso, Gella, Largo |
| 3 | **⦿ Functional Requirements and Nonfunctional Requirements table** | 3/11/2025 | Paraiso, Gella, Largo |
| 4 | **⦿ Purpose**  **⦿ Scope**  **⦿ Definition**  **⦿ References**  **⦿ Product Perspective  ⦿ User Characteristics**  **⦿ Constraints**  **⦿ Assumptions & Dependecies  ⦿ External Interface** | 3/12/2025 | Paraiso, Gella, Largo |

**Table of Contents**

**Change History 2**

**Table of Contents 3**

**1.** **Introduction 4**

1.1. Purpose 4

1.2. Scope 4

1.3. Definitions, Acronyms and Abbreviations 4

1.4. References 4

**2.** **Overall Description 5**

2.1. Product perspective 5

2.2. User characteristics 5

2.4. Constraints 5

2.5. Assumptions and dependencies 6

**3.** **Specific Requirements 7**

3.1. External interface requirements 7

*3.1.1.* *Hardware interfaces 7*

*3.1.2.* *Software interfaces 7*

*3.1.3.* *Communications interfaces 7*

3.2. Functional requirements 7

*Module 1 7*

*Module 2 8*

3.4 Non-functional requirements 8

*Performance 8*

*Security 8*

*Reliability 8*

# 1. Introduction

## 1.1. Purpose

This Software Requirements Specification (SRS) outlines the detailed requirements for the **Garbage Management System (GMS)**, a solution designed to streamline waste collection, provide real-time tracking, send notifications, and offer waste reduction tips.

This document serves as a reference for **developers, project managers, and stakeholders** to ensure that the final product aligns with operational and community needs. It also provides guidance for system development, testing, and maintenance.

**Intended Audience:**

* **Project Managers** – Understand the scope, technical requirements, and development timeline.
* **System Developers** – Use this as a guide for building the required system functionalities.
* **Stakeholders** – Local government units, municipalities, and waste management organizations ensuring alignment with existing operations.
* **End Users** – Residents, municipal staff, and street cleanup organizations using the system to manage trash pickup schedules, report issues, and receive notifications.

## 1.2. Scope

#### **Software Products**

The **Garbage Management System (GMS)** will consist of:

* **Web-based Administrative Interface** – Allows municipalities to manage garbage collection schedules and monitor reports.
* **Mobile Application** – Provides tracking and reporting functionality for garbage truck drivers and residents.
* **Backend Server** – Manages user data, schedules, and communication.

#### **What the Software Will Do:**

* The system **shall** allow users to view the garbage pickup schedule online.
* The system **shall** display a map showing the locations of reported garbage piles.
* The system **shall** send push notifications and alerts for garbage collection updates.
* The system **shall** provide a feedback and complaint feature for users to report issues.
* The system **shall** offer waste reduction tips to educate users on proper waste management.
* The system **shall** send trash pickup reminders to users based on their location and schedule.
* The system **shall** allow users to register and log in to access personalized features.
* The system **shall** maintain a history of garbage collection for tracking and reference.
* The system **shall** include a dashboard displaying key waste management statistics.
* The system **shall** enable users to report missed garbage collections.

#### **What the Software Will Not Do:**

* The system **will not** physically manage waste collection operations.
* The system **will not** handle other community services unrelated to waste management.

#### **Application and Benefits:**

* **Application:** Used by municipalities, garbage truck drivers, and residents to manage waste collection efficiently.
* **Benefits:**
  + Improved efficiency in garbage collection and tracking.
  + Enhanced public participation in waste management through reporting and feedback.
  + Real-time updates and notifications to reduce uncollected trash.

#### **Objectives and Goals:**

* **Reduce** response time to waste pile reports.
* **Encourage** cleaner streets through community participation.
* **Ensure** a user-friendly experience for both residents and administrators.

## 1.3 Definitions, Acronyms and Abbreviations

* **GMS**: Garbage Management System
* **API**: Application Programming Interface
* **CRUD**: Create, Read, Update, Delete
* **RDS**: Relational Database Service

## 1.4 References

* Google Maps API Documentation
* Java Spring Boot Documentation
* Android Kotlin Documentation

# 2. Overall Description

## 2.1. Product perspective

The **Garbage Management System (GMS)** follows a **client-server architecture**:

* **Backend:** Java Spring Boot
* **Web Frontend:** ReactJS
* **Mobile App:** Android Kotlin
* **Database:** MySQL (AWS RDS) for backend, Firebase for mobile data synchronization
* **External Services:**
  + **Google Maps API** – For tracking garbage trucks and waste pile locations.

## 2.2. User characteristics

* **Administrators:** Manage schedules, monitor garbage collection, and handle complaints.
* **Garbage Truck Drivers:** Use the mobile app for live tracking and reporting.
* **Residents:** View pickup schedules, report waste pile locations, and receive notifications.

## 2.4. Constraints

* The system shall support up to 100 concurrent users without performance issues.
* The system shall be hosted on a cloud platform (e.g., AWS, Azure).
* The mobile app shall be compatible with devices running Android 8.0 or higher.
* Reliable internet connectivity is required for real-time tracking and communication.
* Google Maps must be functional for key system features.

## 2.5. Assumptions and dependencies

* **Reliable Internet Access:** The system assumes users will have a stable internet connection for real-time tracking, notifications, and data synchronization.
* **Availability of Google Maps API:** The system depends on Google Maps API for accurate location tracking of garbage trucks and reported waste piles. Any downtime or limitations in the API may affect functionality.
* **Device Compatibility:** The mobile application assumes users have Android devices running **Android 8.0 (Oreo) or higher** to ensure compatibility.
* **Cloud Hosting Services:** The backend and database assume availability of cloud hosting (e.g., **AWS RDS for MySQL, Firebase for mobile**) for data storage and management.
* **Government and Municipality Collaboration:** The system assumes cooperation from local government agencies and waste management organizations for implementing garbage collection schedules and response to reports.
* **User Adoption:** The effectiveness of the system relies on active participation from residents, garbage truck drivers, and administrators in reporting and addressing waste issues.
* **Automated System Maintenance:** Regular system maintenance and software updates will be require to ensure uptime and security, depending on third-party hosting providers.

# 3. Specific Requirements

## 3.1. External interface requirements

### 3.1.1. Hardware interfaces

* The system shall be accessible via desktop and mobile devices.
* Android smartphones shall support the mobile application.

### 3.1.2. Software interfaces

* Backend: Java Spring Boot
* Web Frontend: ReactJS
* Mobile App: Android Kotlin
* Database: MySQL hosted on AWS RDS, Firebase for Mobile

### 3.1.3. Communications interfaces

* REST APIs for data exchange between web, mobile, and backend.
* Google Maps API for location services.

## 3.2. Functional requirements

| Requirement Type | Requirement | Justification |
| --- | --- | --- |
| Functional | The system shall allow users to view the garbage pickup schedule online. | Users can access an interactive calendar or schedule view showing the exact times and dates when garbage collection trucks will pick up trash in their area. This allows them to plan when to put out their garbage and avoid missed pickups |
| Functional | The system shall display a map showing the locations of reported garbage piles. | The web app displays a map with markers indicating where the garbage piles are located within a specified area. This helps waste collectors easily identify locations and optimize their routes. Users can also report or update the status of their garbage pile on the map. |
| Functional | The system shall send push notifications and alerts for garbage collection updates. | The mobile app sends real-time push notifications directly to users' smartphones to remind them of upcoming pickup times. It can also notify users of delays or if the truck is nearby, ensuring that users are well-informed and don't miss their scheduled garbage collection. |
| Functional | The system shall provide a feedback and complaint feature for users to report issues. | Users can submit feedback or complaints regarding garbage collection services through the app, ensuring that service providers can address issues and improve overall user satisfaction. |
| Functional | The system shall offer waste reduction tips to educate users on proper waste management. | The web app can display a section with practical tips to help users reduce their household waste. Tips could cover areas such as reducing plastic use, composting, and upcycling materials. The website could also feature a "Tip of the Day" on the homepage, offering fresh ideas daily to encourage users to adopt more sustainable practices. |
| Functional | The system shall send trash pickup reminders to users based on their location and schedule. | The mobile app allows users to input their garbage collection schedule, including their specific pickup day and area. The app sends push notifications to remind users when to put out their trash, ensuring they don't miss the scheduled pickup. Users can set recurring reminders based on their collection days and update their schedule if there are changes, such as holiday pickups or service delays. |
| Functional | The system shall allow users to register and log in to access personalized features. | Allows users to create an account and log in securely, ensuring that personalized features like schedule reminders, feedback, and map views are accessible only to registered users. This also helps track user activity and preferences. |
| Functional | The system shall maintain a history of garbage collection for tracking and reference. | Provides users with a log of their past garbage pickups, including dates and times. This helps them track whether their waste has been consistently collected and identify any missed pickups. |
| Functional | The system shall include a dashboard displaying key waste management statistics. | A dashboard for garbage collection providers to update schedules, manage user feedback, and monitor reported garbage pile locations. This helps them respond quickly and efficiently to issues reported by users. |
| Functional | The system shall enable users to report missed garbage collections. | Allows users to quickly report if their garbage wasn’t collected on the scheduled day. This notifies the service provider, helping them address missed pickups efficiently and improve service reliability. |

## 3.4. Non-functional requirements

| Requirement Type | Requirement | Justification |
| --- | --- | --- |
| Non-Functional | The system shall support up to 100 concurrent users without performance issues. | Ensures the system’s performance and scalability, allowing multiple users to access and use the system without slowdowns or crashes. |
| Non-Functional | The system shall encrypt all stored and transmitted data to ensure security. | Protects sensitive information from unauthorized access and ensures data privacy and security. |
| Non-Functional | The system shall securely store user data and shall not share it with third parties. | Maintains confidentiality and compliance with data protection regulations, reducing the risk of data breaches. |
| Non-Functional | The system shall maintain an uptime of at least 99.9%. | Guarantees high availability and reliability of the system, minimizing downtime and ensuring users can access services as needed. |
| Non-Functional | The system shall perform automated database backups every 24 hours. | Prevents data loss by maintaining regular backups, allowing quick recovery in case of system failure or corruption. |