Smart Waste Bin Monitoring and Collection Optimization System

1. System Architecture:

The system follows an Edge-to-Cloud architecture optimized for real-time response and low maintenance. ESP32 is used as the primary controller due to its dual-core processing and power efficiency. Fill-level detection is performed using a waterproof HC-SR04 ultrasonic sensor. An automatic lid mechanism using an MG995 servo motor and proximity sensor enables touchless operation. LoRaWAN is selected for long-range, low-power communication. Azure IoT Hub handles device management, Azure Functions process logic, and Azure SQL stores historical data. A centralized dashboard (Power BI or React-based) provides live maps, hotspot analytics, and truck tracking.

2. Data Flow & Communication Design

The data pipeline begins with ultrasonic sensing, followed by edge filtering and local fill-percentage calculation on the ESP32. The processed data is sent to Azure IoT Hub using MQTT, where Azure Functions store and analyze the data before updating the dashboard in real time.

Fill Level Condition	System Action
< 75% (Green)	Monitoring only; no action required
75–84% (Yellow)	Flagged as potential pickup
85–90% (Orange)	Automatically scheduled for collection
98–100% (Red)	Lid locks and emergency alert triggered

3. Dynamic Collection & Route Optimization

Objective: Transition from fixed-schedule to demand-driven collection to optimize fuel and manpower.

Collection Decision Logic (Traffic Light System)

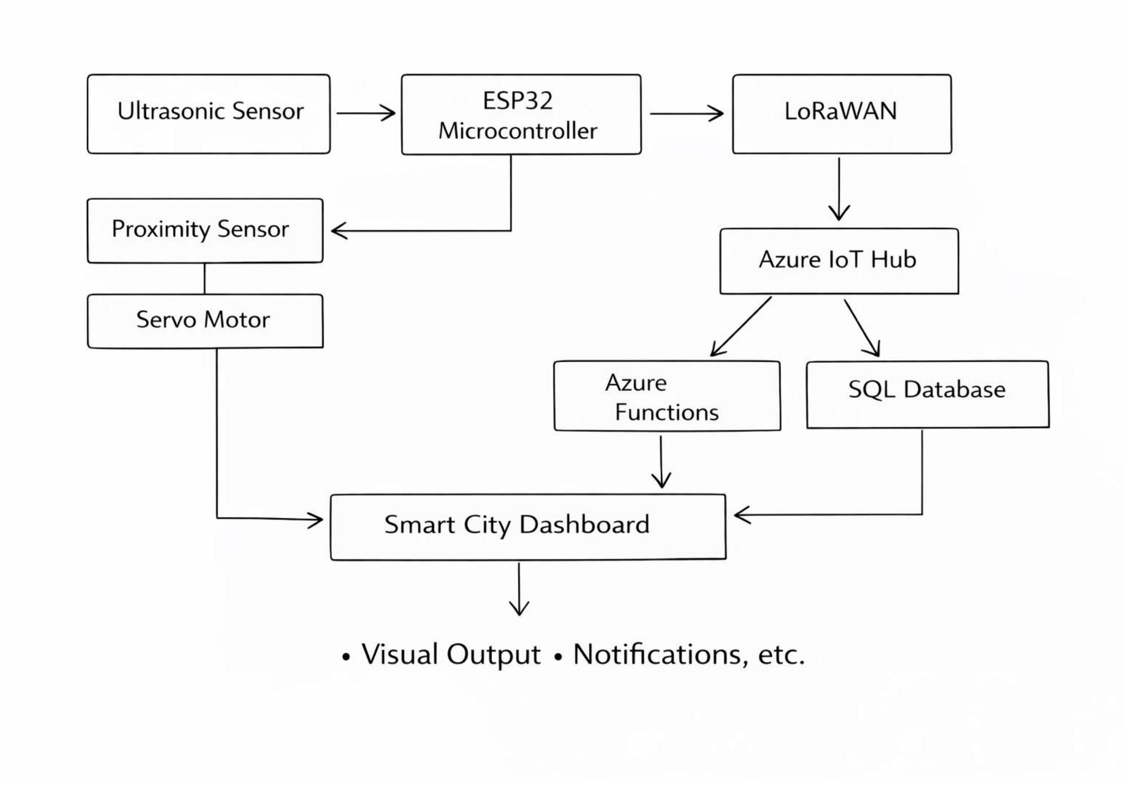
- **< 75% (Green):** Monitoring state; energy conservation mode.
- **75% – 84% (Yellow):** *Monitoring Stage*; flagged on dashboard as a potential pickup within 24 hours.
- **85% – 90% (Orange):** *Scheduling Stage*; automatically added to the active shift route.
- **98% – 100% (Red):** *Immediate Action Stage*; automated lid locks via servo to prevent overflow; emergency alert triggered for nearest personnel.

Algorithmic Routing Approach A hybrid of **Geographic Clustering** and **Shortest-Path Logic** to minimize city crisscrossing.

Optimization Process Flow:

1. **Ingestion:** Receive real-time telemetry from Azure.
2. **Filtration:** Exclude bins below the 85% (Orange) threshold.
3. **Weighting:** Assign "Red" bins as high-priority, first-stop locations.
4. **Clustering:** Group active bins by geographic zone (e.g., Zone A, Zone B).
5. **Pathing:** Apply shortest-path logic per zone for a linear driver sequence.
6. **Dispatch:** Push finalized route to the driver’s mobile application.
7. **Re-routing:** Real-time updates if a nearby bin hits 98% (Red) mid-route.

Flowchart:



4. Power Management Plan

- ESP32 deep sleep is used as the default state to minimize power usage. Fill-level sensing occurs every 30 minutes.
- The servo motor is electronically detached when idle and activated only upon proximity detection.
- MQTT transmissions are limited to significant fill-level changes, enabling months-long battery or solar-powered operation.

5. Reliability & Fault Handling

Common Sensor Blockage Scenarios. Urban waste presents several challenges that can trigger "false full" readings:

- Suspended Debris: Plastic carry bags or light wrappers often hang near the sensor opening, reflecting waves prematurely. Surface Interference: Thin plastic sheets or liners can create uneven surfaces that distort ultrasonic wave returns.
- Path Obstruction: Folded bin liners or large cardboard pieces may partially cover the sensor path, leading to inconsistent data.

❖ **Mitigation & Calibration Techniques**

To handle these scenarios, I will implement the following redundancy and calibration mechanisms:

- Median Filtering & Multi-Sample Averaging: Instead of relying on a single data point, the ESP32 will take a burst of 10 readings. I will program it to discard the highest and lowest values (outliers) and average the remainder to find the true fill level.
- Sudden Spike Rejection: I will implement logic that ignores "instant" changes. For example, if a bin jumps from 10% to 90% in one second, the system flags it as a potential blockage rather than a full state until the reading remains stable for several minutes
- Periodic Cloud-Based Recalibration: I will design the system to allow remote recalibration via the Azure IoT Hub. This enables me to adjust the sensor's sensitivity or "empty" baseline without physically visiting the bin.
- Historical Data Validation: We will use the cloud backend to cross-reference current readings with historical filling trends. If a bin reports a "Red" status at a time when it is historically empty, the system will flag the unit for a manual check-up.

6. Scalability & Network Considerations

The system is designed to scale from pilot deployment to city-wide coverage without performance degradation.

Aspect	Design Approach
Network Topology	Star topology using LoRaWAN gateways
Fault Isolation	Independent bins with no interdependency
Device Identity	Unique X.509 certificates per bin
Zone Management	Azure IoT Device Twins with zone metadata
Scalability	Azure IoT Hub auto-scales for 100+ bins
Dashboard Control	Filtering by zone, priority, and maintenance status

7.Cost & Feasibility Discussion

I have selected components that offer the best "performance-to-price" ratio. The approximate cost breakdown for one smart bin unit is as follows:

- **ESP32 Microcontroller**: ₹350
- **Ultrasonic Sensor** (Waterproof): ₹120
- **Servo Motor** (MG995 High Torque): ₹180
- **LEDs, Wiring, and Miscellaneous**: ₹100
- ✓ Total Estimated Cost: ₹750 – ₹800 per unit

❖ Trade-Off Analysis

Engineering a city-wide system requires balancing competing priorities. I have made the following strategic choices:

- Accuracy vs. Cost: I have chosen Ultrasonic sensors because they provide sufficient accuracy (within 1–2 cm) for waste monitoring at a very low cost. While laser-based LiDAR sensors would offer millimeter precision, they would quadruple the cost per bin, making a city-wide rollout financially impossible.

- <u>Cost vs. Scalability</u>: By utilizing low-cost hardware and cloud-native services (Azure), I have ensured that the system is economically viable. The initial hardware savings allow us to allocate more budget toward robust data analytics and a better user interface for city authorities.
- <u>Scalability vs. Complexity</u>: I am using a cloud-centric architecture to keep the on-field hardware as simple as possible. By moving the "intelligence" (routing and ML) to the cloud, we minimize the complexity of the physical bins, which is crucial for managing 100+ units across multiple zones.

8. Future Scope & Machine Learning Capabilities

In future phases, **Machine Learning models** can be integrated using Azure ML to analyze historical fill data.

ML Capabilities:

- Predict when each bin will get full
- Identify high-usage vs low-usage locations
- Track **filling frequency ratios**
  - Example:
    - Area A → fills every 6–8 hours
    - Area B → fills once in 2 days

Smart Guidance Feature:

- ML-based system can **directly guide pickup personnel** to bins that are most critical
- Optimized navigation with real-time priority updates

Repository link & project submission details:

- **Project Title:** Smart Waste Bin Monitoring and Collection Optimization System
- **Developer:** Kadambari K Patil
- **College:** SVERI’s College of Engineering, Pandharpur
- **Submission Date:** December 26, 2025
- **GitHub Repository:** https://github.com/Kadambari11/WM-Kadambari11-SVERI-COE