Smart Waste Bin Network (Virtual IoT Design Challenge)

- *by Dhanshree Nagpure*

1. System Architecture

➢ System Description

The Smart Waste Bin Network is an IoT-enabled system designed to monitor waste bin fill levels across multiple urban zones and optimize garbage collection operations. Each waste bin functions as an intelligent edge node capable of sensing, local processing, and wireless communication. The collected data is transmitted to a centralized backend system where it is stored, analysed, and visualized for city authorities.

The system reduces manual inspection, prevents bin overflow, improves hygiene, and optimizes operational costs by enabling data-driven waste collection.

➢ Edge Node Hardware Architecture (Waste Bin Unit)

Each waste bin is equipped with a compact and energy-efficient edge device consisting of the following components:

a) Ultrasonic Sensor

- Measures the distance between the bin lid and waste surface
- Non-contact sensing avoids mechanical wear
- Distance values are converted into **fill level percentage**

b) ESP32 Microcontroller

- Acts as the main controller of the bin
- Interfaces with the ultrasonic sensor via GPIO
- Performs fill level calculation and threshold logic
- Constructs frame-based data packets
- Communicates with LoRa module via UART
- Supports low-power modes for energy efficiency

c) LoRa Communication Module (UART-based)

- Enables long-range wireless communication
- Operates without internet connectivity at the bin level
- Suitable for city-wide deployment
- Low power consumption compared to Wi-Fi or cellular

d) Solar-Powered Energy System

- Solar panel charges a rechargeable battery
- Charge controller regulates power flow
- Enables autonomous operation without grid power
- Ideal for outdoor smart city infrastructure

Communication Method:

Link	Technology Used	Reason
Sensor → ESP32	GPIO	Reliable and low latency
ESP32 → LoRa	UART	Simple, robust, and hardware-level control
Bin → Gateway	LoRa	Long range, low power, no internet needed
Gateway → Cloud	HTTP / MQTT	Lightweight and cloud-compatible

Edge Computing (At Bin Level)

- Sensor data acquisition
- Fill level calculation
- Threshold comparison
- Frame creation
- Low-power operation

Cloud Computing (AWS Backend)

- Data storage and management
- Historical analysis

- Dashboard visualization
- Route optimization logic

This separation reduces hardware complexity and improves scalability.

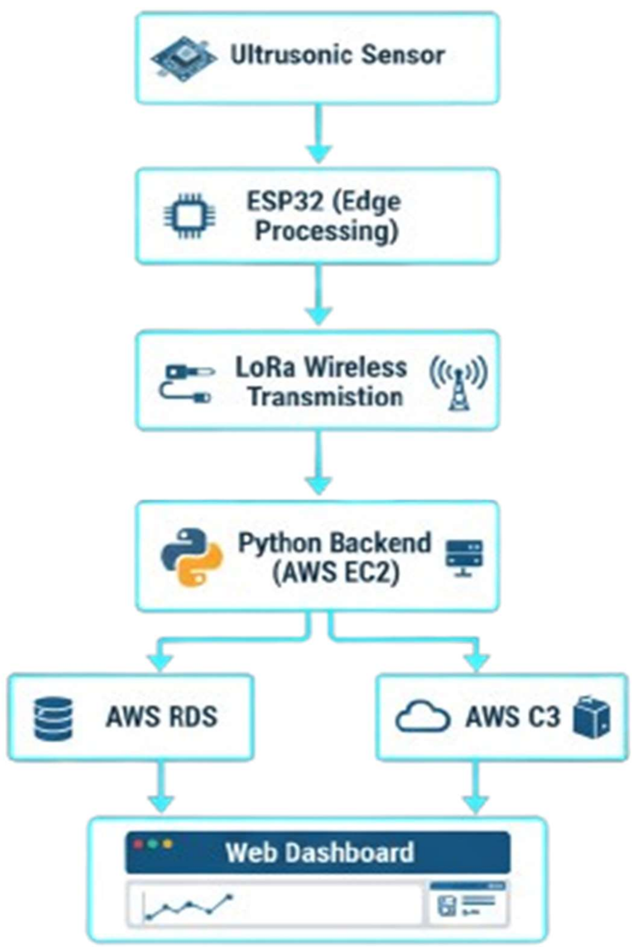
- **Dashboard and Visualization Concept**

A centralized dashboard is provided for city authorities to:

- View real-time bin fill levels
- Identify bins nearing overflow
- Monitor bin locations on a map
- View optimized garbage collection routes

The dashboard enables informed and timely decision-making.

2. Data Flow Design



Protocols Used and Justification

Protocol	Usage	Reason
UART	ESP32 ↔ LoRa	Reliable hardware communication
LoRa PHY	Bin → Gateway	Long range, low power
HTTP / MQTT	Backend → AWS	Lightweight, scalable

Data Frame Design

To ensure reliable transmission over LoRa, a compact frame-based structure is used.

| SOF | Node ID | Fill Level | Status | CRC | EOF |

Field	Size	Description
SOF	1 byte	Start of frame marker
Node ID	1 byte	Unique bin identifier
Fill Level	1 byte	0–100% fill
Status	1 byte	Normal / Threshold exceeded
CRC	1 byte	Error detection
EOF	1 byte	End of frame marker

This frame allows:

- Easy identification of bins
- Error detection
- Scalable multi-node communication

3. Route Optimization Strategy

➢ **Rule-Based Decision Logic**

- If fill level ≥ **80%**, mark bin for collection
- Bins with higher fill levels get higher priority

- **Algorithmic Strategy**

- Priority-based shortest path routing
- Uses bin location data stored in AWS
- Google Maps API used to compute optimal routes
- Minimizes travel distance and fuel consumption

- **Route Optimization Flow**

1. Identify bins above threshold
2. Fetch their locations
3. Apply shortest-path logic
4. Generate optimized route
5. Display route on dashboard

Data Flow Pseudocode:

START

Ultrasonic sensor measures waste level in bin

ESP32 reads sensor data

ESP32 calculates fill level percentage

ESP32 creates data frame:

  Node ID

  Fill Level

  Status

ESP32 sends data frame to LoRa module via UART

LoRa module transmits data wirelessly to gateway

Gateway forwards received data to backend server

Backend receives and decodes data frame

Backend stores data in cloud database (AWS)

Dashboard fetches latest data from database

Dashboard displays bin status and fill level

END

4. Power Management Plan

- Solar-powered edge nodes
- ESP32 operates in deep sleep between measurements
- Sensor activated only during readings
- LoRa transmits only when required
- Ensures long battery life and low maintenance

5. Reliability & Fault Handling

- Multiple sensor readings averaged
- Invalid frames rejected using CRC
- Abnormal readings logged

- Sensor recalibration supported
- Backend validation before storage

6. Scalability & Network Considerations

- **Topology:** Star topology
- Supports 100+ bins
- Each bin has unique Node ID
- Easy addition of new bins
- Cloud backend scales automatically

7. Cost & Feasibility Discussion

➢ **Approximate Cost per Bin (Conceptual)**

Component	**Cost (INR)**
ESP32	₹350 – ₹500
Ultrasonic Sensor	₹70 – ₹150
LoRa Module	₹500 – ₹600
Solar Power System (Panel + Battery + Charge Controller)	₹1500 – ₹2000
Miscellaneous (PCB, enclosure, wiring, connectors, mounting)	₹300 – ₹500
Software / Cloud Cost (Python, open-source tools, AWS Free Tier)	₹0 – ₹300
Total Cost per Bin	₹3,020 – ₹4,050