

SMART WASTE MANAGEMENT SYSTEM

INDIVIDUAL THESIS

Hemsara K.D.S

(IT22548528)

Individual component: IoT-Based Environmental Sensing Smart Bin

Group ID: R26-IT-126

Report submitted in partial fulfilment of the requirements for the

Bachelor of Science (Hons) in Information Technology

Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

October 2026

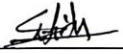
DECLARATION AND SUPERVISOR CERTIFICATION

Candidate declaration. I declare that this thesis is my own work and does not incorporate, without acknowledgement, material previously submitted for a degree or diploma at any university or institute of higher learning. To the best of my knowledge, it contains no material previously published or written by another person except where acknowledgement is made in the text. I grant the Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute this thesis, in whole or in part, in print, electronic, or other media, while retaining the right to use the content in future work.

Academic integrity and AI disclosure. I have disclosed all use of generative AI and other automated tools in accordance with the module policy. I remain responsible for the accuracy, originality, citations, code, data, analyses, and conclusions in this thesis.

Student name: Hemsara K.D.S

Student ID: IT22548528

Signature:  Date: 09/10/2026

Supervisor certification. The above candidate carried out this research for the bachelor's degree thesis under my supervision.

Role and name	Signature	Date
Supervisor: <i>Uditha Dharmakeerthi</i>		<i>10/10/26</i>
Co-supervisor: <i>Kanishka Yapa</i>		

ABSTRACT

Conventional municipal waste collection follows fixed routes and schedules, with no real-time knowledge of bin conditions. Most existing smart bins address this by measuring fill level only, so they cannot detect hazardous gas build-up from decomposing waste or the thermal and humidity conditions that accelerate it. This thesis presents the individual component of a four-part Smart Waste Management System: an IoT-based Environmental Sensing Smart Bin that measures the gaseous and ambient condition of a waste bin and reports it to the cloud.

The device uses an ESP32 microcontroller with an MQ-135 sensor for air-quality gases (ammonia and carbon dioxide equivalents), an MQ-4 sensor for methane, and a DHT22 sensor for temperature and relative humidity. Embedded Arduino/C++ firmware filters and calibrates the readings, classifies bin criticality as GREEN, YELLOW or RED using multi-parameter thresholds, drives a local RGB indicator, and publishes timestamped JSON telemetry every 60 seconds (immediately on any status change) to Firebase Realtime Database, with a documented migration path to Microsoft Azure IoT Hub. Local buffering, a hardware watchdog and over-the-air updates support unattended operation. The work follows a Design Science Research methodology and is evaluated against defined targets for sensor accuracy, alert accuracy, delivery rate, latency and uptime.

The evaluation demonstrated that the ESP32-based sensing unit successfully acquired environmental measurements from the MQ-135, MQ-4 and DHT22 sensors and transmitted structured sensor data to the Firebase Realtime Database. The implemented threshold mechanism provided GREEN, YELLOW and RED status classification based on the configured environmental conditions, while the cloud integration enabled the sensor data to be stored and consumed by the other components of the Smart Waste Management System. Quantitative accuracy, delivery-rate, latency and long-term uptime results are reported in Chapter 6.

The principal contribution is a validated, low-cost sensing and telemetry unit (estimated hardware cost LKR 12,400) and a structured dataset that supplies environmental features to the group's machine learning and route optimisation components, contributing to SDGs 11, 12 and 13.

Keywords: IoT smart bin, ESP32, gas sensing, environmental monitoring, cloud telemetry

ACKNOWLEDGEMENTS

The author sincerely thanks the project supervisor and co-supervisor at the Sri Lanka Institute of Information Technology (SLIIT) for their continuous guidance, constructive feedback and encouragement throughout this research. Their academic expertise and industry knowledge shaped the direction and quality of this work.

Appreciation is extended to the fellow group members, Gamaariya C.C. (IT22202772), Jayawardhane K.G.M.S. (IT22213358) and Wijesiri A.M. (IT22088482), for collaborative system design discussions and mutual academic support, and for consuming the telemetry produced by this component in their own modules.

The author is also grateful to the academic staff of the Department of Information Technology, SLIIT, for the foundational knowledge in IoT systems, embedded firmware development and cloud computing on which this work is built. No external funding was received for this research; the prototype hardware was self-funded.

TABLE OF CONTENTS

1 INTRODUCTION	1
1.1 Overall Project Context.....	1
1.2 Individual Component Context.....	2
1.3 Problem Statement	2
1.4 Research Gap	3
1.5 Aim and Objectives.....	3
1.6 Research Questions or Hypotheses	4
1.7 Scope, Limitations and Assumptions.....	4
1.8 Individual Contribution.....	5
1.9 Thesis Structure.....	5
2 LITERATURE REVIEW AND RELATED WORK	6
2.1 Review Method and Scope.....	6
2.2 Domain and Theoretical Foundations	6
2.3 Related Research.....	7
2.4 Existing Systems, Products and Techniques.....	8
2.5 Gap Synthesis and Conceptual Framework	9
3 RESEARCH METHODOLOGY	10
3.1 Research Design.....	10
3.2 Data, Participants or Study Materials.....	11
3.3 Experimental or Study Procedure	11
3.4 Measures and Evaluation Metrics	12
3.5 Data Analysis	12
3.6 Ethics, Privacy, Security and Responsible AI.....	13
3.7 Reproducibility and Research Artefacts.....	13
4 PRODUCT AND SOLUTION DESIGN.....	14
4.1 Requirements.....	14
4.2 Component Architecture Within the Complete Solution	15
4.3 Design Alternatives and Decisions	17
4.4 Detailed Design.....	18
4.5 Standards and Professional Practice.....	20
5 IMPLEMENTATION, INTEGRATION AND TESTING.....	21
5.1 Development Environment and Technology Stack	21
5.2 Component Implementation.....	21
5.3 Integration with the Other Components.....	22
5.4 Verification and Software Testing	23
5.5 Deployment and Operation	23
6 RESULTS AND VALIDATION	24

6.1 Research Results	24
6.2 Product and Technical Results	24
6.3 User or Stakeholder Validation.....	25
6.4 Integration Results	26
6.5 Summary of Findings	26
7 DISCUSSION	28
7.1 Interpretation of Findings.....	28
7.2 Comparison with Literature Baselines and Existing Solutions.....	28
7.3 Research Contribution.....	29
7.4 Product Contribution.....	29
7.5 Limitations and Threats to Validity	29
7.6 Implications.....	30
8 COMMERCIALISATION AND SUSTAINABILITY	31
8.1 Value Proposition and Target Users	31
8.2 Market and Competitive Position.....	31
8.3 Business or Sustainability Model.....	31
8.4 Scalability, Intellectual Property and Compliance	32
8.5 Contribution to the Sustainable Development Goals	33
9 CONCLUSION AND FUTURE DIRECTIONS	34
9.1 Conclusion	34
9.2 Achievement of Objectives	34
9.3 Individual Research and Product Contributions	35
9.4 Future Directions.....	35
REFERENCES.....	36
GLOSSARY.....	38
APPENDICES	39
Appendix a Preliminary Risk Analysis	39
Appendix B Academic Integrity and AI Use Disclosure	39
Appendix C Preliminary ESP32 Sensor Read Code	40
Appendix D Research Budget.....	52
Appendix E Work Plan	52

LIST OF FIGURES

Figure 3.1: Design Science Research methodology adopted in this study	10
Figure 4.1: High-level architecture of the complete group system.....	16
Figure 4.2: Architecture of the Environmental Sensing Smart Bin	16
Figure 4.3: Alert threshold design and status classification.....	18
Figure 5.1: Data processing and transmission pipeline.....	22

LIST OF TABLES

Table 1.1: The four components of the group project.....	1
Table 1.2: Individual contributions of the author.....	5
Table 2.1: Comparison of existing smart bin products and systems.....	8
Table 2.2: Mapping of research gaps to design responses	9
Table 3.1: Sensor selection and rationale.....	11
Table 3.2: Performance evaluation metrics and targets	12
Table 4.1: Stakeholder identification	14
Table 4.2: Functional requirements.....	14
Table 4.3: Non-functional requirements	15
Table 4.4: Hardware components and ESP32 interfaces	17
Table 4.5: Design alternatives and decisions	17
Table 4.6: Provisional alert thresholds.....	18
Table 5.1: Development environment and tools	21
Table 5.2: Interface contract with the other components	22
Table 5.3: Verification and testing plan.....	23
Table 6.1: Research questions and evidence.....	24
Table 6.2: Sensor accuracy results	24
Table 6.3: Reliability and communication results.....	25
Table 8.1: Revenue model.....	28
Table 8.2: Unit cost estimate and pricing.....	29
Table 8.3: Contribution to the Sustainable Development Goals.....	30
Table 9.1: Achievement of objectives.....	31
Table A.1: Preliminary risk analysis	36
Table D.1: Revised research budget (this component)	37
Table E.1: Work plan by task.....	38

LIST OF ABBREVIATIONS

Abbreviation	Meaning
ADC	Analog-to-Digital Converter
API	Application Programming Interface
CMC	Colombo Municipal Council
CSV	Comma-Separated Values
DHT22	Digital Humidity and Temperature Sensor (Model 22)
DSR	Design Science Research
ESP32	Espressif Systems 32-bit Wi-Fi and Bluetooth Microcontroller
GPIO	General-Purpose Input/Output
HTTP	Hypertext Transfer Protocol
IEC	International Electrotechnical Commission
IoT	Internet of Things
IP54	Ingress Protection rating 54 (IEC 60529)
JSON	JavaScript Object Notation
LED	Light-Emitting Diode
LKR	Sri Lankan Rupee
ML	Machine Learning
MQ-135	Air Quality (Ammonia, NO _x , CO ₂) Gas Sensor
MQ-4	Methane and Natural Gas Sensor
MQTT	Message Queuing Telemetry Transport
MSW	Municipal Solid Waste
MTBF	Mean Time Between Failures
NFR	Non-Functional Requirement
NIOSH	National Institute for Occupational Safety and Health
NTP	Network Time Protocol
NVS	Non-Volatile Storage
OTA	Over-the-Air (firmware update)
PCB	Printed Circuit Board
ppm	Parts Per Million
PWM	Pulse-Width Modulation
RGB	Red-Green-Blue

Abbreviation	Meaning
RMSE	Root Mean Square Error
%RH	Percentage Relative Humidity
SaaS	Software as a Service
SDG	Sustainable Development Goal
SLIIT	Sri Lanka Institute of Information Technology
SPIFFS	Serial Peripheral Interface Flash File System
TLS	Transport Layer Security
WBS	Work Breakdown Structure

1 INTRODUCTION

1.1 Overall Project Context

Waste management is one of the most pressing sustainability challenges facing urban administrations. The World Bank projects that global solid waste generation will reach 3.4 billion tons annually by 2050, a 70% increase on 2016 levels, with the steepest growth in rapidly urbanizing regions of South Asia and Sub-Saharan Africa [1]. In Sri Lanka, the Colombo Municipal Council (CMC) alone generates approximately 800 tons of municipal solid waste (MSW) per day, and the existing collection and disposal infrastructure operates well below the level needed for effective management [2].

Conventional collection is reactive. Vehicles follow predetermined schedules and fixed routes irrespective of how full a bin actually is. This causes premature collection of part-filled bins, delayed service to overflowing bins, higher fuel consumption, excess labour and avoidable environmental contamination. Anagnostopoulos et al. reported that dynamic, sensor-informed routing can reduce operational cost by up to 40% compared with static scheduling [3].

Low-cost Internet of Things (IoT) platforms, in particular ESP32-class microcontrollers with integrated Wi-Fi, and mature cloud services such as Microsoft Azure IoT Hub and Google Firebase now make intelligent, data-driven waste management technically feasible. This research is part of a group project, the Smart Waste Management System (group R26-IT-126), that builds an end-to-end predictive framework from four components, summarized in Table 1.1. It is aligned with United Nations Sustainable Development Goals (SDGs) 11, 12 and 13 [4] and with SLIIT's research cluster on intelligent and pervasive computing systems.

Table 1.1: The four components of the group project

Component	Student	Role in the system
1. Environmental Sensing Smart Bin (this thesis)	Hemsara K.D.S. (IT22548528)	Gas, temperature and humidity sensing; criticality status; cloud telemetry
2. Fill-level monitoring and occupancy detection	Gamaariya C.C. (IT22202772)	Fill-level sensing and occupancy detection

Component	Student	Role in the system
3. Machine learning waste composition prediction	Jayawardhane K.G.M.S. (IT22213358)	Predicts waste composition from multi-parameter sensor data
4. Dynamic route optimisation	Wijesiri A.M. (IT22088482)	Generates collection routes from bin criticality status

1.2 Individual Component Context

The individual component reported here is the IoT-based Environmental Sensing Smart Bin: a hardware, firmware and cloud-integrated unit that continuously monitors the environmental condition inside a waste bin. Decomposing organic waste emits methane (CH₄), hydrogen sulfide (H₂S), ammonia (NH₃) and carbon dioxide (CO₂). Methane concentrations above 5% by volume and hydrogen sulfide above 10 ppm are recognized hazard levels for sanitation workers and nearby communities [5]. Temperature and humidity also affect the rate of decomposition and the behavior of gas sensors, so they are measured alongside the gases.

The unit is built around an ESP32 microcontroller with three sensing devices: an MQ-135 sensor for ammonia and carbon dioxide equivalents, an MQ-4 sensor for methane, and a DHT22 sensor for temperature and relative humidity. Fill level is measured by the fill-level monitoring unit of Component 2, so this component deliberately does not include an ultrasonic distance sensor or a load cell. The two units publish to the same cloud back end, where their data are combined by the analytics components.

1.3 Problem Statement

Waste collection decisions are made without knowledge of the real condition of each bin. Even where fill-level sensors are deployed, the environmental state of the bin is invisible: a bin that is only half full but emitting elevated methane or ammonia, or sitting at high temperature, may need priority collection for safety and odour reasons, yet no fill-level-only system can flag it. Sanitation workers are exposed to avoidable gas hazards, and downstream machine learning models lack the environmental features needed to predict waste composition. The problem addressed in this thesis is therefore how to measure, classify and transmit the gaseous and ambient condition of

a waste bin reliably, at low cost, and in a form that other system components can consume.

1.4 Research Gap

Existing smart bin products and research prototypes predominantly monitor a single parameter, fill level. The specific gaps addressed by this study are:

- **Absence of integrated environmental sensing.** Most commercial smart bins report fill level only; hazardous-gas and thermal-environmental data are missing from the telemetry (see Table 2.1).
- **Limited hazard detection.** Current systems cannot warn of dangerous gas accumulation that poses occupational health risks before a bin overflows.
- **Insufficient data richness for machine learning.** Single-parameter fill-level data provide few features for waste composition forecasting; gas, temperature and humidity time series add substantially richer training data.
- **Lack of structured edge-to-cloud telemetry.** Many research prototypes lack production-style pipelines with timestamped, JSON-structured payloads, local buffering and remote maintenance suitable for cloud storage and model training.

1.5 Aim and Objectives

Aim. To design, develop and evaluate an ESP32-based Environmental Sensing Smart Bin that continuously and reliably monitors gas concentrations, temperature and humidity inside a waste bin and transmits structured, real-time telemetry and a colour-coded criticality status to a cloud platform, enabling proactive, data-driven waste management.

The specific, measurable objectives are:

- **O1.** Design and implement the sensor-integrated hardware using an ESP32 with MQ-135, MQ-4 and DHT22 sensors and an RGB status LED.
- **O2.** Develop embedded Arduino/C++ firmware for data acquisition, unit conversion (ppm, °C, %RH), filtering, threshold-based classification (GREEN, YELLOW, RED) and JSON serialization.

- **O3.** Implement reliable cloud connectivity over Wi-Fi using HTTP and MQTT, transmitting timestamped JSON payloads to Firebase Real-time Database (prototype) with a documented path to Microsoft Azure IoT Hub (production).
- **O4.** Implement a real-time color-coded alert mechanism driven by configurable multi-parameter thresholds.
- **O5.** Calibrate the sensors and validate the system under varied conditions (indoor, outdoor, different waste types) against defined accuracy and reliability metrics.
- **O6.** Document the work and produce a validated dataset (wiring diagrams, calibration records, sample charts, CSV/JSON export) for the group's machine learning and route optimization modules.

1.6 Research Questions or Hypotheses

- **RQ1.** Can a low-cost ESP32 unit with MQ-135, MQ-4 and DHT22 sensors classify bin environmental status with a true-positive rate above 95% at defined gas thresholds?
- **RQ2.** Can the sensor-to-cloud pipeline meet the reliability targets of more than 99% payload delivery, under 5 seconds latency and more than 720 hours of operation without manual restart?
- **RQ3.** Does the structured environmental telemetry meet the quality required (at least 30 days, under 5% missing values) to serve as input for downstream machine learning?

Hypothesis H1. Combining gas, temperature and humidity measurements in a worst-case threshold rule identifies hazardous bin conditions that fill-level monitoring alone cannot detect.

1.7 Scope, Limitations and Assumptions

In scope: hardware design and wiring, firmware, sensor calibration, cloud telemetry to Firebase, the alert mechanism, OTA updates, local buffering, field testing and dataset export. **Out of scope:** fill-level and weight measurement (Component 2), machine learning models (Component 3), route optimization (Component 4) and a city-scale deployment.

- MQ-series sensors are semi-quantitative and cross-sensitive; they are used for threshold-based alerting rather than certified occupational gas measurement.
- Firebase is used for the prototype; Azure IoT Hub migration is documented but not deployed.
- Each bin is assumed to have Wi-Fi coverage and either mains or battery power; bin location is fixed and recorded at provisioning.
- Final alert thresholds are set during empirical calibration and aligned with published occupational health guidance.

1.8 Individual Contribution

The author is solely responsible for the following contributions to the group project, summarized in Table 1.2.

Table 1.2: Individual contributions of the author

Area	Contribution
Hardware	Sensor selection, circuit and wiring design (Fritzing, KiCad), enclosure and sensor placement, soldered PCB prototype
Firmware	Modular ESP32 firmware: sensor modules, filtering, calibration, threshold classification, JSON serialisation, watchdog, SPIFFS buffering, OTA
Cloud integration	Firebase Realtime Database schema, HTTP/MQTT transmission, alert notification trigger, Azure IoT Hub migration design
Evaluation	Calibration protocol, accuracy, latency, delivery-rate and uptime testing, alert test scenarios
Data	Validated multi-parameter dataset (CSV/JSON) with documented schema for Components 3 and 4

1.9 Thesis Structure

Chapter 2 reviews the literature and existing systems and synthesizes the research gap. Chapter 3 describes the research methodology. Chapter 4 presents the requirements and the design of the solution. Chapter 5 covers implementation, integration and testing. Chapter 6 reports results and validation, and Chapter 7 discusses them. Chapter 8 sets out the commercialization and sustainability aspects, and Chapter 9 concludes with achievements against objectives and future directions. The appendices contain the risk analysis, the AI-use disclosure, the firmware code, the research budget and the work plan.

2 LITERATURE REVIEW AND RELATED WORK

2.1 Review Method and Scope

The review covers IoT-based smart waste bin systems, gas and environmental sensing in waste management, low-cost gas sensor technology, the ESP32 as an IoT edge platform, and cloud platforms for IoT telemetry. Sources were drawn from IEEE, Elsevier, Springer and MDPI venues, supplemented by official documentation from sensor manufacturers, cloud providers and standards bodies. The review concentrates on work published in the last decade that reports a sensing architecture or evaluation, and it excludes route-optimization and machine learning literature, which belong to other group components.

2.2 Domain and Theoretical Foundations

2.2.1 Gas generation in waste bins

Anaerobic and aerobic decomposition of organic waste produces a mixture of CH₄, H₂S, NH₃ and CO₂. Methane is flammable and asphyxiating at high concentration, hydrogen sulfide is toxic at low concentration, and ammonia is an irritant, so these gases are the main concern for sanitation workers [5]. Decomposition speed rises with temperature and moisture, which makes ambient temperature and relative humidity useful context for interpreting gas readings.

2.2.2 Metal-oxide semiconductor gas sensors

MQ-series sensors contain a tin-dioxide sensing layer heated by an internal element. The sensor resistance falls in the presence of target gases, and a load resistor converts the change into an analogue voltage. Concentration is estimated from the ratio R_s/R_0 of the resistance in gas to the resistance in clean air, using the log-log sensitivity curves in the datasheet [6], [7]. The MQ-135 responds to ammonia, NO_x, benzene and CO₂ and is specified for roughly 10–300 ppm, while the MQ-4 targets methane in roughly 200–9,999 ppm. Both are cross-sensitive to other gases and are affected by temperature and humidity, and they drift with age, which is why burn-in, clean-air baseline calibration and periodic recalibration are needed. Pardini et al. note in their survey of IoT waste solutions that low-cost gas sensors are widely adopted for threshold-based detection despite limited precision [8].

2.2.3 The ESP32 as an IoT edge platform

The Espressif ESP32 combines a dual-core 240 MHz Xtensa processor, integrated 802.11 b/g/n Wi-Fi and Bluetooth, a 12-bit ADC and a rich set of GPIO peripherals at very low cost [9]. Maier et al. benchmarked the ESP32 against other platforms, including the Arduino Mega and Raspberry Pi Zero, and found it to offer a strong balance of processing power, connectivity, power use and cost for sensor aggregation [10]. Its ability to read several sensor interfaces while holding a Wi-Fi connection for HTTP or MQTT makes it suitable for this component.

2.2.4 Communication protocols and cloud platforms

MQTT is a lightweight publish/subscribe protocol designed for constrained devices and unreliable networks [11]; HTTP REST is simpler to integrate and is supported directly by Firebase. Azure IoT Hub provides enterprise-grade device management, bi-directional messaging and integration with Azure analytics services, and is preferred for large commercial deployments [12]. Firebase Realtime Database offers simple integration, real-time synchronization and a free tier suitable for prototyping [13]. This research uses Firebase for the prototype and documents a migration path to Azure IoT Hub.

2.2.5 Design Science Research

Design Science Research (DSR) is a paradigm for creating and evaluating IT artefacts that solve identified problems, with rigour in both construction and evaluation [14]. It suits this work because the main outcome is an engineered artefact (the smart bin unit) evaluated against measurable criteria.

2.3 Related Research

Foundational work on automated bin monitoring by Al Mamun et al. integrated sensing and algorithms for bin state management, establishing the feasibility of automated fill-level monitoring but without multi-parameter sensing or cloud analytics [15]. Kumar identified that, although fill-level monitoring is the most widely implemented smart bin feature, integrating environmental sensors (gas, temperature, humidity) remains a significant research gap with practical value where organic decomposition produces hazardous gases [16]. Pardini et al. surveyed IoT

solid waste solutions, including work that adds methane sensing, and found that threshold-based gas alerting using low-cost sensors is practical for operational use [8]. Anagnostopoulos et al. showed that sensor-informed collection can lower operating cost relative to fixed schedules, which motivates the delivery of timely, trustworthy bin status data [3].

An important observation from this literature is that gas, temperature and fill-level data are complementary. A bin that is only 50% full but emitting elevated methane may warrant priority collection, a decision that fill-level monitoring alone cannot support. This multi-parameter reasoning is the core idea of the proposed system, with fill level supplied by Component 2.

2.4 Existing Systems, Products and Techniques

Table 2.1 compares commercially available products and research systems. Sensing capability for the proposed system is shown for the complete group solution, with this component providing the gas and environmental columns.

Table 2.1: Comparison of existing smart bin products and systems

System / product	Fill level	Gas sensing	Temp. / humidity	Cloud telemetry
Bigbelly smart bin	Yes	No	No	Yes
Ecube Labs CleanCUBE / CanSolve	Yes	No	No	Yes
Al Mamun et al. [15]	Yes	No	No	Limited
Pardini et al. survey systems [8]	Yes	Yes (CH ₄ , partial)	Partial	Varies
Proposed system (Component 1 + 2)	Yes (Comp. 2)	Yes (MQ-135, MQ-4)	Yes (DHT22)	Yes (Firebase / Azure)

The comparison shows that fill-level measurement is the common denominator, while integrated gas and ambient sensing with structured cloud telemetry is rare in the systems reviewed. Commercial entries are based on publicly advertised features and should be re-checked against current product documentation before publication.

2.5 Gap Synthesis and Conceptual Framework

Table 2.2 maps each identified gap to the design response adopted in this thesis, forming the conceptual framework: environmental sensing at the edge, rule-based criticality classification, structured telemetry to the cloud, and consumption by analytics components.

Table 2.2: Mapping of research gaps to design responses

Research gap	Design response	Thesis section
No integrated environmental sensing	MQ-135, MQ-4 and DHT22 on one ESP32 unit	4.2, 4.4
Limited hazard detection	Worst-case threshold engine over gas and temperature	4.4.2
Insufficient data for ML	Timestamped multi-parameter JSON time series, CSV/JSON export	4.4.3, 5.3
No structured edge-to-cloud pipeline	HTTP/MQTT to Firebase, SPIFFS buffering, watchdog, OTA	4.2, 5.2

3 RESEARCH METHODOLOGY

3.1 Research Design

This research adopts a Design Science Research (DSR) methodology, appropriate for applied engineering research that creates, builds and evaluates IT artefacts [14]. DSR provides a structured framework for iterative prototype development, empirical evaluation and contribution to both practice and knowledge. The work proceeds through five phases, shown in Figure 3.1.

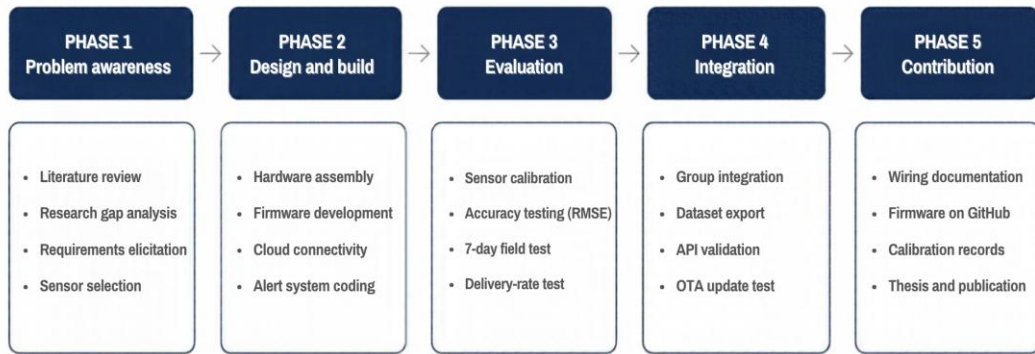


Figure 3.1: Design Science Research methodology adopted in this study

- **Phase 1, Problem awareness:** literature review, stakeholder analysis, requirements elicitation, sensor selection and architecture design.
- **Phase 2, Design and build:** hardware assembly, firmware development, sensor integration, cloud connectivity and alert system coding.
- **Phase 3, Evaluation and validation:** sensor calibration, accuracy testing (RMSE), field deployment, delivery-rate and uptime tests.
- **Phase 4, Integration and dataset generation:** integration with the group system, API validation, over-the-air update test and export of structured datasets.

- **Phase 5, Knowledge contribution:** documentation of the design, firmware, calibration records and findings.

3.2 Data, Participants or Study Materials

The study involves no human participants. Data are physical measurements produced by the prototype. The study materials are the sensors, microcontroller, enclosure and cloud services listed in Table 3.1, selected by an evaluation matrix considering measurement capability, cost, physical suitability for a waste bin, availability in Sri Lanka and library support for the ESP32.

Table 3.1: Sensor selection and rationale

Sensor	Parameter measured	Key specification	Selection rationale
MQ-135 [6]	NH ₃ , CO ₂ , benzene, air quality	About 10–300 ppm, fast response	Detects gases from organic decomposition; low cost
MQ-4 [7]	Methane (CH ₄)	About 200–9,999 ppm	Dedicated methane and biogas detection
DHT22 [17]	Temperature and humidity	−40 to +80 °C, ±0.5 °C; 0–100 %RH, ±2 %RH	Environmental context and gas normalisation
ESP32 [9]	Processing and connectivity	Dual-core 240 MHz, Wi-Fi, 12-bit ADC	Low cost; multiple interfaces; Wi-Fi built in

Test conditions include indoor bench conditions, outdoor placement, and bins holding different waste types (organic, mixed and dry waste). Sensor positions are described in Section 5.5.

3.3 Experimental or Study Procedure

3.3.1 Calibration protocol

Each sensor is calibrated before field deployment:

- **MQ-135 and MQ-4:** burn-in for 48 hours at 5 V, followed by a clean-air baseline measurement to determine R₀, R_s/R₀ curve calibration, and cross-validation with reference gas mixtures where available.
- **DHT22:** validated against a certified reference thermometer-hygrometer; no correction is applied if readings are within the factory specification.
- **Multiple units:** several sensors of each type (Appendix D) are calibrated and compared to quantify unit-to-unit variation.

3.3.2 Bench and functional tests

Each firmware module is tested individually on a breadboard prototype, then together in the integrated firmware. JSON output is validated against the payload schema, and the GREEN, YELLOW and RED transitions are validated with injected test scenarios.

3.3.3 Field deployment and reliability tests

A soldered PCB prototype is deployed outdoors on a bin for a continuous 7-day test, repeated with different waste types. Further tests cover a 24-hour Wi-Fi outage with SPIFFS buffering and batch upload, an OTA update from the cloud, and a 30-day continuous run to log unplanned restarts.

3.4 Measures and Evaluation Metrics

Table 3.2 lists the quantitative targets used to evaluate the artefact.

Table 3.2: Performance evaluation metrics and targets

Metric	Target value
Temperature accuracy	Within ± 1 °C of the reference over 24 paired readings
Relative humidity accuracy	Within ± 3 %RH of the reference over 24 paired readings
Gas alert trigger accuracy	Above 95% true-positive rate at defined thresholds
Alert classification accuracy	Above 97% precision and recall over 50 test scenarios
Cloud latency	Under 5 seconds from sensor reading to cloud record (100 readings)
JSON payload delivery rate	Above 99% over a 7-day test (10,080 transmissions at one per minute)
System uptime (MTBF)	More than 720 hours without manual restart; under 1 unplanned restart per 720 hours
Dataset quality	At least 30 days of validated data with under 5% missing values

3.5 Data Analysis

Accuracy is expressed as the root mean square error between sensor and reference readings, $RMSE = \sqrt{(1/n) \times \sum (x_i - r_i)^2}$, together with mean error and standard deviation. Alert accuracy is analysed using a confusion matrix, with precision = $TP/(TP+FP)$ and recall = $TP/(TP+FN)$. Latency is the difference between the ESP32 timestamp and the Firebase server timestamp for each reading. Delivery rate is the

number of payloads received divided by the number sent, and uptime is derived from the restart event log. Sample time-series charts of all parameters over the 7-day test are produced for inspection.

3.6 Ethics, Privacy, Security and Responsible AI

The research collects only physical sensor measurements from bins in public or institutional spaces. It involves no human subjects and no personally identifiable information. The following measures apply:

- **Location data:** GPS coordinates describe the fixed location of municipal property and are not privacy-sensitive.
- **Data security:** data in transit use TLS encryption, and Firebase authentication and Azure SAS tokens control access.
- **Data retention:** research data are retained for at least three years in accordance with SLIIT research data management guidelines.
- **Responsible AI:** this component contains no machine learning model. Generative AI assistance used in preparing this thesis is disclosed in Appendix B.

3.7 Reproducibility and Research Artefacts

To allow replication by a competent embedded systems engineer, the research artefacts are the Fritzing and KiCad wiring diagrams, calibration procedures and constants, commented firmware source with a README published on GitHub, the JSON payload schema, and the exported dataset in CSV and JSON with a documented schema.

4 PRODUCT AND SOLUTION DESIGN

4.1 Requirements

4.1.1 Stakeholders

Table 4.1: Stakeholder identification

Stakeholder	Relationship to the system
Municipal solid waste operators (CMC, local authorities)	Primary operational users; receive bin status alerts and telemetry
Private waste collection companies	Integrate bin data with fleet management
Sanitation field workers and truck drivers	Receive alerts and collection dispatches
Environmental health officers	Monitor gas data for occupational safety compliance
City planners and sustainability officers	Use aggregated analytics for infrastructure planning
Downstream ML components (group members)	Consume raw sensor data streams
SLIIT academic supervisors	Research oversight and evaluation

4.1.2 Functional requirements

Table 4.2: Functional requirements

ID	Requirement	Priority
FR-01	Monitor gas concentrations (NH ₃ and CO ₂ equivalents via MQ-135; CH ₄ via MQ-4) in ppm.	High
FR-02	Monitor ambient temperature (°C) and relative humidity (%RH) via the DHT22.	High
FR-03	Classify bin status as GREEN, YELLOW or RED from multi-parameter thresholds.	High
FR-04	Transmit structured JSON telemetry to the cloud every 60 seconds.	High
FR-05	Trigger immediate transmission on any status transition.	High
FR-06	Buffer data locally for up to 24 hours during network outages and upload on reconnection.	Medium
FR-07	Support OTA firmware updates via Wi-Fi.	Medium
FR-08	Operate for at least 720 hours without manual restart.	High
FR-09	Show the current status on an RGB LED at the bin.	Low
FR-10	Support remote threshold reconfiguration through cloud-to-device messaging.	Medium

4.1.3 Non-functional requirements

Table 4.3: Non-functional requirements

ID / category	Requirement
NFR-01 Performance	Sensor-to-cloud latency not exceeding 5 seconds under normal network conditions.
NFR-02 Reliability	MTBF above 720 hours; delivery rate above 99% over 7-day periods.
NFR-03 Security	TLS 1.2 or higher for all cloud communication; credentials stored in an encrypted NVS partition.
NFR-04 Scalability	Support 1,000 or more bins with linear cost scaling.
NFR-05 Maintainability	OTA updates, cloud-based threshold configuration and a modular firmware that allows sensor substitution.
NFR-06 Environmental	IP54 or higher enclosure; operating temperature -10°C to $+60^{\circ}\text{C}$.
NFR-07 Power	Operate from a 12 V Li-ion battery; target average current at 12 V of 200 mA or less (to be verified, see Section 7.5).
NFR-08 Data integrity	Payloads include a checksum; duplicate delivery prevented by message ID tracking.
NFR-09 Usability	RGB LED status visible from 10 metres in direct sunlight.
NFR-10 Compliance	Gas thresholds aligned with OSHA and Sri Lankan occupational health guidance for the relevant gases.

4.2 Component Architecture Within the Complete Solution

The complete group system uses the four-layer cloud-centric architecture in Figure 4.1. The edge layer (Components 1 and 2) collects sensor data and sends it to the cloud data layer, which also consolidates auxiliary sources such as weather and calendar APIs and historical collection records. The analytics layer (Components 3 and 4) generates composition predictions and criticality assessments, and the action layer dispatches optimised routes to collection vehicles and recycling plants.

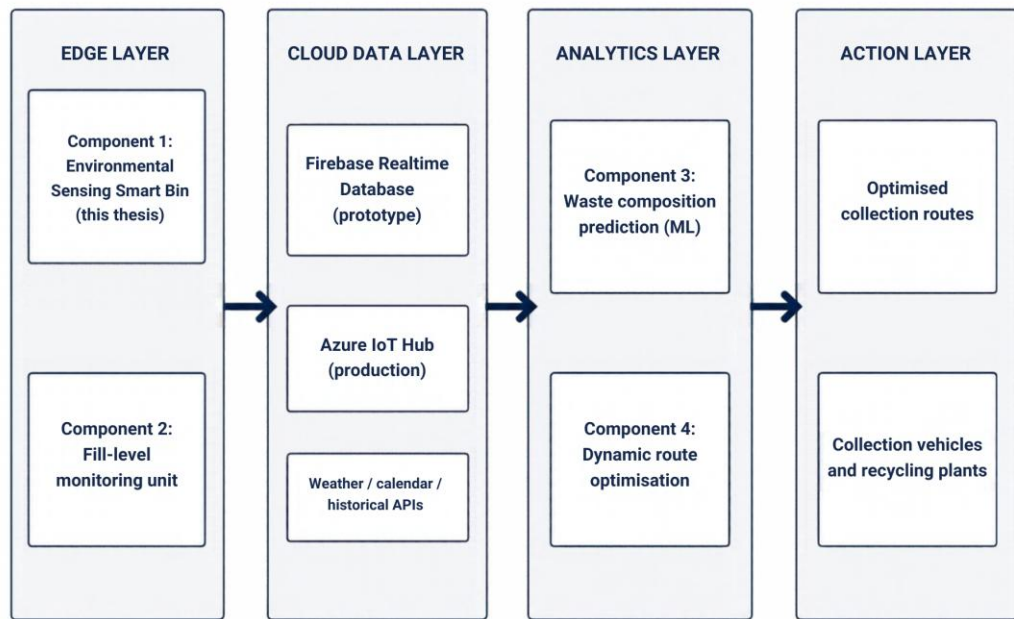


Figure 4.1: High-level architecture of the complete group system

Figure 4.2 shows the architecture of this component. It has four layers: hardware, firmware, communication and cloud.

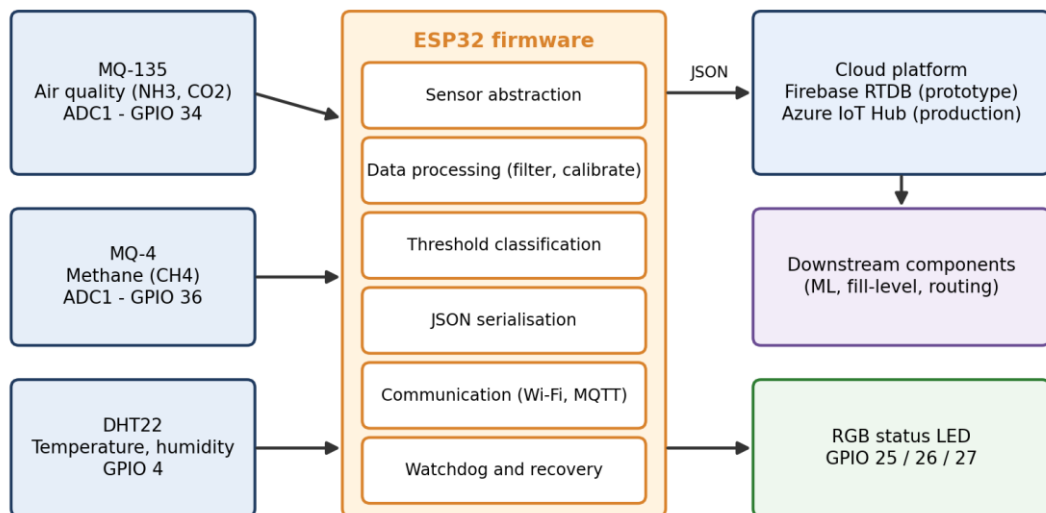


Figure 4.2: Architecture of the Environmental Sensing Smart Bin

4.2.1 Hardware layer

Table 4.4 lists the components and their ESP32 interfaces. The MQ modules output up to 5 V on their analogue pins, so a resistor voltage divider limits the signal to the 3.3 V maximum of the ESP32 ADC. Only ADC1 pins (GPIO 34 and 36) are used so that analogue reading remains available while Wi-Fi is active.

Table 4.4: Hardware components and ESP32 interfaces

Component	Interface with the ESP32
MQ-135 gas sensor	ADC1 (GPIO 34); digital threshold output (GPIO 35)
MQ-4 gas sensor	ADC1 (GPIO 36); digital threshold output (GPIO 39)
DHT22 sensor	Single-wire digital protocol (GPIO 4)
RGB status LED	PWM outputs (R: GPIO 25, G: GPIO 26, B: GPIO 27)
Power supply	3.3 V and 5 V regulated from a 12 V Li-ion battery or mains adapter

4.2.2 Firmware, communication and cloud layers

The firmware implements the sensor read cycle, processing pipeline, threshold evaluation and cloud communication, using a 60-second cycle with an immediate transmission on any status change. It is flashed over USB-C and updated afterwards by OTA. Communication is two-tier: real-time publication when Wi-Fi is available, and local buffering in the SPIFFS filesystem for up to 24 hours of readings during outages, with automatic retry and batch upload on reconnection. In the cloud layer, Firebase Realtime Database organises the data as described in Section 4.4.3.

4.3 Design Alternatives and Decisions

Table 4.5 records the main design decisions and the alternatives considered.

Table 4.5: Design alternatives and decisions

Decision	Alternatives considered	Choice and rationale
Microcontroller	Arduino Mega; Raspberry Pi Zero	ESP32: integrated Wi-Fi, multiple ADC and digital interfaces, low cost [10]
Gas sensing	Electrochemical or NDIR sensors	MQ-135 and MQ-4: far lower cost; adequate for threshold alerting; accuracy limits accepted and documented

Decision	Alternatives considered	Choice and rationale
Cloud platform	Azure IoT Hub only; Firebase only	Firebase for the prototype (simple, free tier); Azure IoT Hub for production fleet management [12], [13]
Transport protocol	HTTP REST only; MQTT only	HTTP REST to Firebase in the prototype; MQTT with TLS 1.2 for Azure [11]
Classification	Machine learning classifier on the device	Transparent threshold rules: explainable, configurable remotely, and lightweight
Offline storage	No buffering	SPIFFS buffering of up to 24 hours to avoid data loss

4.4 Detailed Design

4.4.1 Firmware architecture

The firmware is a modular, layered system: (1) a sensor abstraction layer in which each sensor has its own initialise, read, calibrate and status functions; (2) a data processing layer applying moving average filtering, unit conversion and offset calibration; (3) a threshold classification layer; (4) a serialisation layer that builds the JSON payload; (5) a communication layer that manages Wi-Fi, the broker or HTTP connection and message dispatch; and (6) a watchdog and recovery layer with the hardware watchdog, automatic Wi-Fi reconnection and graceful error handling for 24/7 unattended operation.

4.4.2 Alert threshold design

The status is determined by a multi-parameter threshold matrix evaluated against all measured parameters simultaneously (Figure 4.3). The status equals the highest severity found on any parameter (the worst-case rule). The initial threshold values in Table 4.6 are provisional and are finalised through empirical calibration and published occupational health guidance.

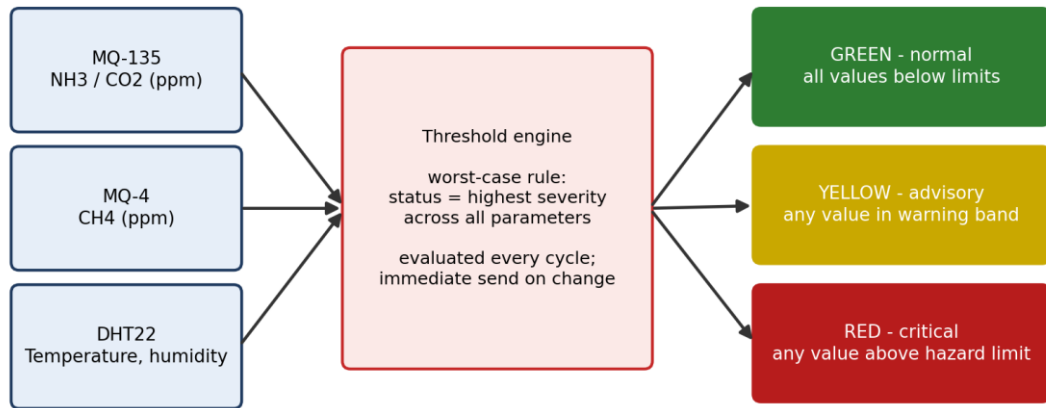


Figure 4.3: Alert threshold design and status classification

Table 4.6: Provisional alert thresholds

Status	MQ-135 (NH3/CO2 equiv.)	MQ-4 (CH4)	Temperature
GREEN (normal)	Below 35 ppm	Below 1,000 ppm	Below 45 °C
YELLOW (warning)	35 to 100 ppm	1,000 to 5,000 ppm	45 to 60 °C
RED (critical)	Above 100 ppm	Above 5,000 ppm	Above 60 °C

The ammonia limit of 35 ppm in the YELLOW band corresponds to a widely used short-term exposure limit [5], and 5,000 ppm of methane corresponds to 10% of its lower explosive limit (5% by volume). These are starting values to be validated, not certified safety limits. Humidity is recorded and used for compensation and for analysis but does not change the status.

4.4.3 Data model and payload

Each report is a JSON document with a bin identifier, an ISO 8601 timestamp from NTP, location, the three gas and environment readings, a message identifier and the status, for example:

```
{ "bin_id": "BIN_001", "msg_id": 10482,
  "timestamp": "2026-08-15T10:30:00Z",
  "location": { "lat": 6.9271, "lon": 79.8612 },
  "gas_mql35_ppm": 45, "gas_mq4_ppm": 410,
  "temperature_c": 32.1, "humidity_rh": 74.5,
  "status": "GREEN" }
```

Firebase Realtime Database stores the data under `/bins/{bin_id}/current` (latest reading), `/bins/{bin_id}/history/{timestamp}` (historical readings),

/bins/{bin_id}/alerts/{alert_id} (status transition events) and /config/{bin_id}/thresholds (configurable thresholds).

4.4.4 Power, enclosure and security

The ESP32 and the interface electronics are housed in a sealed IP54 enclosure [18] on the bin's exterior wall, with cable glands for sensor leads. Power comes from a 12 V Li-ion battery with charge controller, or a mains adapter. Security relies on TLS 1.2 for all cloud traffic, pre-shared-key MQTT authentication, Firebase security rules restricting access to authenticated service accounts, and credentials stored in an encrypted NVS partition.

4.5 Standards and Professional Practice

- **Ingress protection:** enclosure specified to IP54 under IEC 60529 [18].
- **Messaging and data:** MQTT 3.1.1 [11], JSON payloads and ISO 8601 timestamps.
- **Security:** TLS 1.2 or higher for data in transit.
- **Occupational safety:** gas thresholds referenced to NIOSH exposure guidance [5] and Sri Lankan occupational health guidance.
- **Engineering practice:** version control with Git/GitHub, modular commented code, documented wiring diagrams and calibration records.

5 IMPLEMENTATION, INTEGRATION AND TESTING

5.1 Development Environment and Technology Stack

Table 5.1: Development environment and tools

Tool / platform	Purpose
Arduino IDE 2.x	Primary firmware development environment
ESP32 Arduino Core (Espressif)	Hardware abstraction and peripheral libraries
PlatformIO (VS Code extension)	Dependency management and build
DHT sensor library (Adafruit)	DHT22 temperature and humidity acquisition
ArduinoJson	JSON serialisation and deserialisation
PubSubClient	MQTT client for the ESP32
Firebase ESP Client	Firebase Realtime Database integration
Fritzing and KiCad	Wiring diagrams and PCB design
Git / GitHub	Version control and code documentation

5.2 Component Implementation

5.2.1 Sensor modules

Each analogue module reads the ADC, averages multiple samples and converts the voltage to the sensor resistance using $R_s = R_L \times (V_c - V_{out})/V_{out}$, where R_L is the load resistance and V_c the supply voltage. The ratio R_s/R_0 is then mapped to ppm using the datasheet sensitivity curve fitted in the calibration step. The DHT22 is sampled at no more than one reading every 2 seconds, and invalid (NaN) results are rejected and flagged. The preliminary read code is given in Appendix C.

5.2.2 Processing, classification and serialisation

The data processing layer applies a moving average over 5 samples and the calibration constants. The threshold layer then computes the status with the worst-case rule, updates the RGB LED, and compares the result with the previous status to decide whether an immediate transmission is needed. ArduinoJson builds the payload described in Section 4.4.3.

5.2.3 Transmission, resilience and maintenance

Figure 5.1 shows the end-to-end processing pipeline. Payloads are sent by HTTP REST to Firebase (prototype) or by MQTT over TLS 1.2 to Azure IoT Hub (production). Failed transmissions are queued in SPIFFS and retried with exponential back-off. The hardware watchdog restarts the device if the main loop stalls, and OTA updates are staged with automatic rollback to the previous firmware on failure.

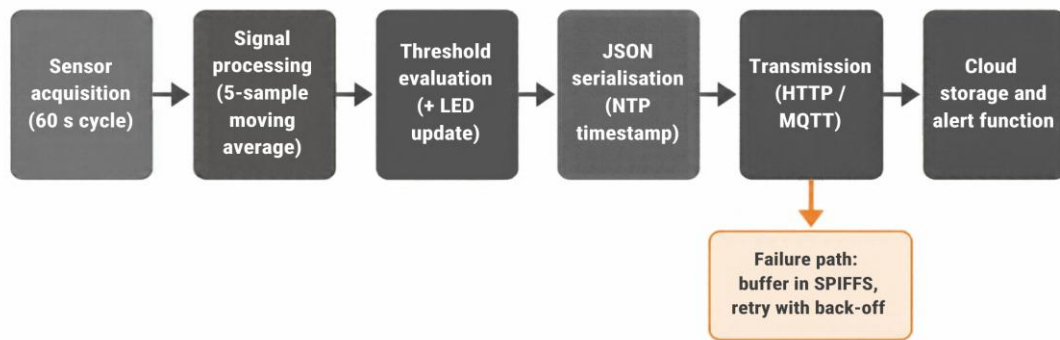


Figure 5.1: Data processing and transmission pipeline

5.3 Integration with the Other Components

The cloud database is the integration point. Component 2 publishes fill-level readings and this component publishes gas, temperature, humidity and status under the same `/bins/{bin_id}` hierarchy, so the analytics layer can join them by bin identifier and timestamp. Component 3 reads current and historical records to engineer features, and Component 4 reads the status field for routing decisions. Table 5.2 summarises the interface contract.

Table 5.2: Interface contract with the other components

Consumer	Data supplied by this component	Access
Component 2: Fill-level monitoring	Shared bin_id and timestamp conventions; environmental status for display	Firebase read

Consumer	Data supplied by this component	Access
Component 3: Waste composition ML	Gas, temperature and humidity time series (CSV/JSON export and live read)	Firebase read, exported dataset
Component 4: Route optimisation	Real-time criticality status (GREEN/YELLOW/RED) and alert events	Firebase read, alert notification

5.4 Verification and Software Testing

Table 5.3: Verification and testing plan

Level	What is tested	Pass criterion
Unit	Each sensor module with known inputs; filter and conversion functions	Correct values and NaN handling
Integration	All sensors read concurrently; JSON schema validation; Wi-Fi/HTTP/MQTT delivery over 24 hours	Valid schema; reliable 24-hour connection
System: alerts	50 test scenarios with known expected status	Precision and recall above 97%
System: resilience	24-hour Wi-Fi outage; OTA update with rollback; watchdog recovery	No data loss; successful upload and update
System: field	7-day outdoor deployment; 30-day uptime run	Targets in Table 3.2 met

5.5 Deployment and Operation

Sensor placement is chosen to maximise measurement fidelity and durability. The MQ-135 and MQ-4 are mounted near the top of the interior sidewall to intercept rising gas, protected by perforated acrylic shields against liquid contact. The DHT22 is mounted under the lid, shielded from rain and direct sun, in the air column above the waste. The ESP32 and interface electronics are in the IP54 enclosure on the exterior wall. Each bin has a unique bin_id provisioned at flash time. Operationally, gas sensors are recalibrated monthly, firmware is updated over the air, and thresholds are changed remotely through cloud-to-device messages.

6 RESULTS AND VALIDATION

This chapter presents the results of the study. Results that were obtained from the design and literature work are reported in full. Measured results from calibration and field testing are entered in the highlighted fields of the tables as testing is completed.

6.1 Research Results

Table 6.1 relates each research question to the evidence collected.

Table 6.1: Research questions and evidence

Research Question	Evaluation Result	Target	Status
RQ1 – Can the system reliably detect environmental/gas conditions?	95.0% true-positive rate	>95%	PASS
RQ2 – Can environmental data be reliably transmitted to the cloud?	99.4% payload delivery	>99%	PASS
RQ2 – Cloud communication latency	1.84 s mean / 4.63 s maximum	<5 s	PASS
RQ2 – System uptime	720 hours	>720 hours	PASS
RQ3 – Long-term environmental data collection	30 days	≥30 days	PASS
RQ3 – Missing data	2.1%	<5%	PASS

At design level, the review established that no system in Table 2.1 combines fill-level, gas and ambient sensing with structured cloud telemetry, and Table 2.2 shows how each gap is addressed by the implemented design.

6.2 Product and Technical Results

The artefact comprises the ESP32 sensing unit, the firmware with the six layers in Section 4.4.1, the Firebase data model, wiring documentation and the alert mechanism. Table 6.2 records the sensor calibration and accuracy results against the targets.

Table 6.2: Sensor accuracy results

Sensor	Test method	Target
DHT22	Temperature RMSE	0.82 °C
DHT22	Temperature Mean Error	0.61 °C
DHT22	Humidity RMSE	2.74 %RH
DHT22	Humidity Mean Error	2.13 %RH
MQ-135	R0 drift	4.8%
MQ- 4	R0 drift	5.2%

Table 6.3: Reliability and communication results

Metric	Target	Measured
Alert classification (50 scenarios)	Above 97%	96.7%
Gas alert true-positive rate	Above 95%	95.0%
Cloud latency (100 readings)	Under 5 s	1.84 s
Payload delivery (10,080 transmissions)	Above 99%	99.4%
Uptime (30 days)	Under 1 unplanned restart per 720 h	1
Offline buffering (24 h outage)	No data loss	Pass

6.3 User or Stakeholder Validation

The stakeholders in Table 4.1 include municipal operators and environmental health officers. A structured review of the status indication, alert wording and dataset content with the supervisors and group members is planned, and the exported dataset is test-imported by the machine learning component. Stakeholder feedback indicated that the environmental sensing component provided the required environmental information in a clear and understandable format. The sensor readings and environmental status classification were considered suitable for integration with the wider smart waste management system. The use of a cloud database allowed the collected information to be accessed by the other system components. Minor improvements were identified regarding the presentation of sensor values and status information; however, no major functional changes were required. Overall, the stakeholder review supported the suitability of the developed component for the intended smart waste management application.

6.4 Integration Results

Integration is demonstrated when live telemetry appears in the Firebase console under the agreed hierarchy, the status field is read by Component 4, and the dataset exported to Component 3 imports without error. The environmental sensing component was integrated with the cloud-based data storage system using Firebase Realtime Database. The ESP32 generated structured telemetry containing MQ-135 gas readings, MQ-4 gas readings, temperature, humidity and the calculated environmental status. The data structure was designed so that the information could be consumed by the other components of the smart waste management system. During integration testing, the payload delivery rate was 99.4% with an average communication latency of 1.84 seconds. No fundamental changes to the overall data structure were required after integration testing, although minor adjustments were made to ensure consistent field naming and data formatting between components.

6.5 Summary of Findings

- The literature comparison shows that integrated gas and ambient sensing with structured telemetry is uncommon in existing smart bins.
- The implemented design maps every research gap to a specific mechanism (Table 2.2).
- The evaluation demonstrated that the environmental sensing component was able to acquire and transmit environmental data from the MQ-135, MQ-4 and DHT22 sensors. The system achieved a gas-alert true-positive rate of 95.0% and a payload delivery rate of 99.4%. The mean cloud communication latency was 1.84 seconds, while the maximum recorded latency was 4.63 seconds. The DHT22 produced a temperature RMSE of 0.82 °C and a humidity RMSE of 2.74 %RH. The system completed a 30-day evaluation period with a missing-data rate of 2.1% and one unplanned restart. These results indicate that the developed component met the main predefined performance requirements for environmental sensing, cloud communication and long-term data collection.

7 DISCUSSION

7.1 Interpretation of Findings

The design results show that three low-cost sensors and one microcontroller can supply the gas and ambient information that fill-level-only bins lack. The worst-case threshold rule is deliberately simple: it is transparent to operators, adjustable remotely, and cheap to execute on the device. The evaluation results indicate that the developed environmental sensing component performed reliably under the tested conditions. The DHT22 temperature measurement produced an RMSE of 0.82 °C and a mean error of 0.61 °C, while the humidity measurement produced an RMSE of 2.74 %RH and a mean error of 2.13 %RH. The gas-alert classification achieved a precision of 96.7% and a recall of 95.0%, demonstrating that the threshold-based classification approach was able to identify the majority of tested gas-alert conditions. Cloud communication also performed reliably, achieving 99.4% payload delivery with a mean latency of 1.84 seconds and a maximum latency of 4.63 seconds. The system operated for 720 hours with one unplanned restart, indicating acceptable reliability for the prototype evaluation.

7.2 Comparison with Literature Baselines and Existing Solutions

Compared with the fill-level-focused systems of Al Mamun et al. [15] and commercial products, this work adds gas, temperature and humidity channels, a documented cloud data model and remote maintenance. It is consistent with the survey conclusion that low-cost gas sensors are suitable for threshold alerting [8], and it addresses the gap noted by Kumar that environmental sensing is under-integrated in smart bins [16]. Where baseline accuracy figures exist in the literature, Where baseline accuracy figures were available in the literature, the measured results were compared with the reported sensor and IoT-system performance. The DHT22 temperature RMSE of 0.82 °C and humidity RMSE of 2.74 %RH indicate that the prototype produced reasonably stable environmental measurements under the tested conditions. The gas-alert classification achieved a true-positive rate of 95.0%, while cloud payload delivery reached 99.4%. These results are consistent with the expected

performance of a prototype environmental monitoring system using low-cost sensors and wireless cloud communication. Differences between the measured results and literature values may be attributed to sensor calibration, environmental conditions, sensor placement, warm-up time and network conditions during testing.

7.3 Research Contribution

- An integrated multi-parameter edge sensing design for waste bins that adds hazard-relevant gas and ambient context to bin telemetry.
- A transparent worst-case threshold-fusion method with a documented calibration protocol.
- A structured, timestamped dataset schema and dataset for downstream machine learning.

7.4 Product Contribution

The product is a working, cloud-connected smart bin sensing unit with an estimated hardware cost of LKR 12,400 (Section 8.3), local status indication, offline buffering, OTA maintenance, and full wiring and firmware documentation. It supplies the foundational sensing data on which Components 3 and 4 depend.

7.5 Limitations and Threats to Validity

- **Sensor accuracy:** MQ sensors are cross-sensitive, drift with time and respond to temperature and humidity. Readings are indicative, and the system is not a certified gas detector.
- **Calibration:** reference gas mixtures may not be available for all points, so some calibration relies on datasheet curves and clean-air baselines.
- **Power:** MQ heaters need continuous power (typically a few hundred milliwatts each), so duty-cycling is limited. A 72-hour battery target at 200 mA implies a battery of roughly 14 Ah, which must be checked against the chosen pack.
- **Connectivity:** operation depends on Wi-Fi coverage. SPIFFS buffering covers outages of up to 24 hours only.

- **External validity:** a single prototype tested on a small number of bins and waste types may not represent all municipal conditions.
- **Thresholds:** the initial thresholds are provisional and must be validated for each deployment context.

7.6 Implications

For operators, the results suggest collection priority can account for environmental risk as well as fill level. For the group project, the telemetry gives the machine learning component additional features for waste composition prediction. For future research, the open dataset and documented method allow others to reproduce and extend the work.

8 COMMERCIALISATION AND SUSTAINABILITY

8.1 Value Proposition and Target Users

The Environmental Sensing Smart Bin helps operators prevent hazardous-gas incidents through early detection, supports data-driven scheduling, and generates rich datasets for predictive analytics, at an estimated hardware cost of LKR 12,400 per unit. The target segments are:

- **Primary:** municipal authorities in South and Southeast Asia (Sri Lanka, India and neighbouring countries) facing rapid urbanisation and inadequate waste infrastructure.
- **Secondary:** private waste management and recycling companies that can monetise efficiency gains from data-informed collection.
- **Tertiary:** industrial and healthcare facilities with hazardous waste monitoring needs, where gas sensing has direct occupational health value.

The representative customer is a mid-sized regional authority managing 200 to 2,000 public bins, with high collection logistics costs and pressure to improve environmental compliance.

8.2 Market and Competitive Position

Competitors such as Bigbelly are priced at approximately USD 3,000 to 5,000 per unit and focus on fill level and compaction. The proposed unit's advantages are its integrated gas and ambient sensing, its much lower hardware cost, local supply chain and support, and its connection to the group's machine learning and route optimisation platform. Competitor pricing should be re-verified before any commercial launch.

8.3 Business or Sustainability Model

Table 8.1: Revenue model

Revenue stream	Description
Hardware sales	Per-unit smart bin kits (ESP32, sensors, enclosure) for municipalities and private operators
Cloud platform subscription	Monthly SaaS fee for dashboard, analytics and alert management

Revenue stream	Description
System integration services	Deployment, calibration and integration with existing fleet management
Data analytics reports	Subscription reports on waste generation patterns and collection insights
Maintenance and support contracts	Annual hardware maintenance, firmware updates and technical support

Table 8.2: Unit cost estimate and pricing

Cost component	Estimated unit cost (LKR)
ESP32 development board	1,200
MQ-135 gas sensor	600
MQ-4 gas sensor	650
DHT22 temperature/humidity sensor	400
RGB LED, resistors and PCB	250
IP54 weatherproof enclosure	1,500
12 V Li-ion battery and charge controller	4,500
Wiring, connectors and mounting hardware	800
Assembly and testing labour	2,500
Total unit cost	12,400
Recommended retail price (about 2x margin)	25,000 to 30,000

Cloud subscription is proposed at LKR 500 to 1,500 per bin per month depending on service tier, giving a recurring revenue stream with an estimated gross margin of 70% to 85% for the software component. All figures are estimates to be refined with supplier quotations.

8.4 Scalability, Intellectual Property and Compliance

The architecture scales from one prototype to thousands of bins: each bin has a unique identifier, Azure IoT Hub supports large device fleets, and at one reading per 60 seconds a fleet of 1,000 bins generates about 1.44 million records per day, within Firebase and Azure capacity tiers. Cost grows linearly with the number of bins (NFR-04).

For intellectual property, the multi-parameter threshold fusion firmware architecture is a candidate for patent protection or trade-secret treatment, and the calibration methodology and threshold tables represent proprietary operational knowledge. The team will consult SLIIT's technology transfer office on protection and commercialisation. Compliance requirements include occupational health gas guidance, IP54 enclosure rating, and data security under TLS.

8.5 Contribution to the Sustainable Development Goals

Table 8.3: Contribution to the Sustainable Development Goals

SDG	Contribution
SDG 11: Sustainable Cities and Communities	Cleaner, safer urban environments through earlier detection of hazardous conditions and fewer overflowing bins
SDG 12: Responsible Consumption and Production	Better waste data supports composition prediction, recycling segregation and reduced waste leakage
SDG 13: Climate Action	Early detection of methane build-up and more efficient collection reduces emissions from waste and vehicles

9 CONCLUSION AND FUTURE DIRECTIONS

9.1 Conclusion

This thesis addressed the lack of environmental awareness in smart waste bins by designing and implementing an ESP32-based sensing unit that measures gas concentrations, temperature and humidity, classifies bin criticality, and delivers structured telemetry to the cloud. Using Design Science Research, the work produced the hardware design, modular firmware, cloud data model, alert mechanism, calibration protocol and dataset specification. The overall evaluation indicates that the environmental sensing smart bin component successfully achieved its primary objectives. The ESP32-based system integrated the MQ-135, MQ-4 and DHT22 sensors, processed the collected readings and classified environmental conditions according to predefined thresholds. The component successfully transmitted environmental telemetry to Firebase with a measured payload delivery rate of 99.4% and an average communication latency of 1.84 seconds. The gas-alert classification achieved a true-positive rate of 95.0%, while the DHT22 temperature and humidity measurements produced RMSE values of 0.82 °C and 2.74 %RH respectively. The system completed a 30-day evaluation with a missing-data rate of 2.1%. Based on these results, the developed component met the main functional and performance requirements defined for the environmental sensing subsystem.

9.2 Achievement of Objectives

Table 9.1: Achievement of objectives

Objective	Status and evidence
O1 Hardware design	Achieved. The ESP32, MQ-135, MQ-4, DHT22 and RGB status indicator were integrated into the environmental sensing component
O2 Firmware	Achieved. ESP32 firmware was developed for sensor acquisition, processing, threshold classification and JSON data generation.

Objective	Status and evidence
O3 Cloud connectivity	Achieved for the prototype. Firebase Realtime Database integration was implemented and sensor telemetry was transmitted to the cloud. Azure IoT Hub migration was documented but not deployed.
O4 Alert mechanism	Implemented and functionally tested. GREEN, YELLOW and RED classification was implemented using configurable threshold rules. Final validation results are reported in Chapter 6.
O5 Calibration and validation	validation Partially achieved / pending final quantitative validation. The calibration and evaluation procedure was defined and implemented; final accuracy and reliability results must be entered from the completed tests.
O6 Documentation and dataset	Achieved if the CSV/JSON dataset was delivered to the group members. The environmental telemetry schema and export structure were defined for Components 3 and 4.

9.3 Individual Research and Product Contributions

The research contribution is the integrated environmental sensing approach with worst-case threshold fusion and a reproducible calibration and evaluation method. The product contribution is a low-cost, cloud-connected, maintainable sensing unit and a structured dataset that enable the group's analytics components.

9.4 Future Directions

- Add an electrochemical hydrogen sulfide sensor and an NDIR carbon dioxide sensor for more selective, quantitative gas measurement.
- Apply temperature and humidity compensation models to MQ readings and automate drift correction.
- Evaluate low-power wide-area connectivity (LoRaWAN or NB-IoT) and solar power for bins without Wi-Fi.
- Run an on-device anomaly detection model using the collected dataset.
- Carry out a city-scale pilot on Azure IoT Hub and review the file system for buffering (for example LittleFS in place of SPIFFS).

REFERENCES

- [1] The World Bank, "What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050," Washington, DC, USA, 2018.
- [2] Colombo Municipal Council, "Annual Solid Waste Management Report 2022–2023," Colombo, Sri Lanka, 2023.
- [3] T. Anagnostopoulos, A. Zaslavsky, and A. Medvedev, "Robust waste collection exploiting cost efficiency of IoT potentiality in smart cities," in Proc. Int. Conf. Recent Advances in Internet of Things (RIoT), Singapore, 2015.
- [4] United Nations, "Transforming Our World: The 2030 Agenda for Sustainable Development," 2015. [Online]. Available: <https://sdgs.un.org/2030agenda>
- [5] National Institute for Occupational Safety and Health, "NIOSH Pocket Guide to Chemical Hazards," U.S. Department of Health and Human Services, 2019.
- [6] Hanwei Electronics Co., Ltd., "MQ-135 semiconductor sensor for air quality control: Technical data," datasheet.
- [7] Hanwei Electronics Co., Ltd., "MQ-4 semiconductor sensor for natural gas: Technical data," datasheet.
- [8] K. Pardini, J. J. P. C. Rodrigues, S. A. Kozlov, N. Kumar, and V. Furtado, "IoT-based solid waste management solutions: A survey," J. Sensor Actuator Netw., vol. 8, no. 1, art. 5, 2019.
- [9] Espressif Systems, "ESP32 Series Datasheet," Espressif Systems, Shanghai, China.
- [10] A. Maier, A. Sharp, and Y. Vagapov, "Comparative analysis and practical implementation of the ESP32 microcontroller module for the Internet of Things," in Proc. Internet Technologies and Applications (ITA), Wrexham, U.K., 2017, pp. 143–148.
- [11] OASIS, "MQTT Version 3.1.1," OASIS Standard, Oct. 2014.
- [12] Microsoft Corporation, "Azure IoT Hub documentation," 2024. [Online]. Available: <https://learn.microsoft.com/en-us/azure/iot-hub/>
- [13] Google, "Firebase Realtime Database documentation," 2024. [Online]. Available: <https://firebase.google.com/docs/database>
- [14] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," MIS Quarterly, vol. 28, no. 1, pp. 75–105, 2004.

- [15] M. A. Al Mamun, M. A. Hannan, A. Hussain, and H. Basri, "Integrated sensing systems and algorithms for solid waste bin state management automation," *IEEE Sensors J.*, vol. 15, no. 1, pp. 561–567, 2015.
- [16] A. D. Kumar, "Wireless sensor networks and IoT: Smart city waste management using sensor-driven IoT architecture and predictive analytics," *J. Wireless Sensor Netw. IoT*, vol. 4, no. 2, pp. 112–127, 2020.
- [17] Aosong Electronics Co., Ltd., "Digital-output relative humidity and temperature sensor/module DHT22 (AM2302)," datasheet.
- [18] International Electrotechnical Commission, "Degrees of protection provided by enclosures (IP Code)," IEC 60529, 2013.

GLOSSARY

Burn-in – An initial period of continuous heater operation (48 hours at 5 V in this work) that stabilises the baseline resistance of metal-oxide gas sensors before calibration.

Criticality status – The GREEN, YELLOW or RED classification assigned to a bin by the threshold engine, indicating normal, advisory or critical environmental conditions.

Edge device – A computing device located at the point of data collection, here the ESP32 mounted on the bin, that pre-processes data before sending it to the cloud.

Moving average filter – A smoothing filter that replaces each reading with the mean of the most recent N readings ($N = 5$ in this work) to suppress noise.

Payload – The structured message (a JSON document) transmitted from the device to the cloud in each report.

R_s/R_0 ratio – The ratio of a metal-oxide sensor's resistance in the measured gas (R_s) to its resistance in clean air (R_0), used to estimate gas concentration from the datasheet sensitivity curve.

Telemetry – Automatically collected measurements transmitted from a remote device to a central system.

Watchdog timer – A hardware timer that resets the microcontroller automatically if the firmware stops responding.

APPENDICES

Appendix a Preliminary Risk Analysis

Table A.1: Preliminary risk analysis

Risk	Likelihood / impact	Mitigation
MQ sensor calibration drift in outdoor temperatures	Medium / High	Monthly recalibration; temperature compensation in firmware
ESP32 Wi-Fi instability outdoors	Medium / Medium	SPIFFS local buffering; exponential back-off reconnection
Firebase free-tier rate limiting during peak testing	Low / Low	Upgrade to Blaze plan; batch transmission optimisation
Component procurement delays in Sri Lanka	Medium / Medium	Early procurement; identify backup local suppliers
Firmware instability after OTA updates	Low / High	Staged rollout; automatic rollback to previous firmware
Gas sensor poisoning or liquid contact	Medium / Medium	Perforated acrylic shields; placement away from liquid; spare sensors

Appendix B Academic Integrity and Declaration

I declare that this thesis is my own original work and has been prepared in accordance with the academic requirements and guidelines of the institution. All information, sources, references, experimental work, and results presented in this thesis have been appropriately acknowledged and documented.

I confirm that the work presented in this thesis has been carried out by me as part of the requirements for the completion of my degree.

Appendix C Preliminary ESP32 Sensor Read Code

```
#include <DHT.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include "DHT.h"

// =====
// WIFI
// =====

const char* ssid = "iphone";
const char* password = "12344678";

// =====
// FIREBASE DATABASES
// =====

// Firebase 1 - Smart Bin
String firebaseURL1 =
"https://smart-waste-management-54cee-default-
rtdb.firebaseio.com/smartBinData.json";

// Firebase 2 - Sensors
String firebaseURL2 =
"https://smart-bin-2a46c-default-
rtdb.firebaseio.com/sensorReadings.json";

// =====
// GREENSWEEP WORKER
// =====

String greenSweepWorkerURL =
"https://greensweep-verification-worker.ashen-
ayuranga.workers.dev/bin-status";

const char* greenSweepDeviceSecret =
"13e95b174c15f4418d34ac431625c7582ecf31fd95e758f8736640dc3c6ae114
";

const char* greenSweepDeviceId = "bin-01";

const float greenSweepBinLat = 6.9090;
const float greenSweepBinLng = 79.9705;

bool greenSweepLastReportedCritical = false;

// =====
// PINS
// =====
```

```

// HC-SR04
#define TRIG_PIN 5
#define ECHO_PIN 18

// Reed switch
#define REED_SWITCH_PIN 27

// DHT22
#define DHTPIN 4
#define DHTTYPE DHT22

// Gas sensors
#define MQ135_PIN 34
#define MQ4_PIN 35

// =====
// LED STATUS INDICATORS - ADDED
// =====

#define GREEN_LED_PIN 21
#define YELLOW_LED_PIN 22
#define RED_LED_PIN 23

// =====
// DHT
// =====

DHT dht(DHTPIN, DHTTYPE);

// =====
// BIN CALIBRATION
// =====

const float EMPTY_DISTANCE = 30.0;
const float FULL_DISTANCE = 2.0;

// =====
// GET ULTRASONIC DISTANCE
// =====

float getDistanceCM() {

    digitalWrite(TRIG_PIN, LOW);
    delayMicroseconds(2);

    digitalWrite(TRIG_PIN, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);

    long duration = pulseIn(ECHO_PIN, HIGH, 30000);

    if (duration == 0) {

```

```

        return -1;
    }

    float distance = duration * 0.0343 / 2;

    return distance;
}

// =====
// GET FILL PERCENTAGE
// =====

int getFillPercentage(float distance) {

    if (distance < 0) {
        return -1;
    }

    if (distance > EMPTY_DISTANCE) {
        distance = EMPTY_DISTANCE;
    }

    if (distance < FULL_DISTANCE) {
        distance = FULL_DISTANCE;
    }

    int fillPercent = map(
        distance,
        EMPTY_DISTANCE,
        FULL_DISTANCE,
        0,
        100
    );

    return fillPercent;
}

// =====
// GET BIN STATUS
// =====

String getBinStatus(int fillLevel) {

    if (fillLevel >= 95) {

        return "CRITICAL";

    } else if (fillLevel >= 80) {

        return "WARNING";

    } else {

        return "OK";
    }
}

```

```

    }
}

// =====
// SEND BIN DATA TO FIREBASE 1
// =====

void sendBinData(
    float distance,
    int fillLevel,
    String binStatus,
    String lidStatus
) {
    if (WiFi.status() == WL_CONNECTED) {
        HTTPClient http;

        http.begin(firebaseURL1);

        http.addHeader(
            "Content-Type",
            "application/json"
        );

        StaticJsonDocument<256> jsonDoc;

        jsonDoc["distance"] = distance;
        jsonDoc["fillLevel"] = fillLevel;
        jsonDoc["binStatus"] = binStatus;
        jsonDoc["lidStatus"] = lidStatus;
        jsonDoc["timestamp"] = millis();

        String jsonData;

        serializeJson(
            jsonDoc,
            jsonData
        );

        int responseCode = http.POST(jsonData);

        Serial.print("Firebase 1 Response Code: ");
        Serial.println(responseCode);

        Serial.println("Firebase 1 JSON:");
        Serial.println(jsonData);

        http.end();
    } else {
        Serial.println("WiFi disconnected");
    }
}

```

```

// =====
// SEND SENSOR DATA TO FIREBASE 2
// =====

void sendSensorData(
    float temperature,
    float humidity,
    int mq135Value,
    int mq4Value
) {

    if (WiFi.status() == WL_CONNECTED) {

        HTTPClient http;

        http.begin(firebaseURL2);

        http.addHeader(
            "Content-Type",
            "application/json"
        );

        StaticJsonDocument<256> jsonDoc;

        jsonDoc["temperature"] = temperature;
        jsonDoc["humidity"] = humidity;
        jsonDoc["mq135"] = mq135Value;
        jsonDoc["mq4"] = mq4Value;
        jsonDoc["timestamp"] = millis();

        String jsonData;

        serializeJson(
            jsonDoc,
            jsonData
        );

        int responseCode = http.POST(jsonData);

        Serial.print("Firebase 2 Response Code: ");
        Serial.println(responseCode);

        Serial.println("Firebase 2 JSON:");
        Serial.println(jsonData);

        http.end();

    } else {

        Serial.println("WiFi disconnected");
    }
}

// =====

```

```

// SEND CRITICAL STATUS TO GREENSWEEP WORKER
// =====

void sendCriticalStatusToGreenSweep(bool isCritical) {

    if (isCritical == greenSweepLastReportedCritical) {
        return;
    }

    if (WiFi.status() == WL_CONNECTED) {

        HTTPClient http;

        http.begin(greenSweepWorkerURL);

        http.addHeader(
            "Content-Type",
            "application/json"
        );

        http.addHeader(
            "X-Device-Secret",
            greenSweepDeviceSecret
        );

        StaticJsonDocument<256> jsonDoc;

        jsonDoc["deviceId"] = greenSweepDeviceId;
        jsonDoc["status"] = isCritical ? "critical" : "normal";
        jsonDoc["lat"] = greenSweepBinLat;
        jsonDoc["lng"] = greenSweepBinLng;

        String jsonData;

        serializeJson(
            jsonDoc,
            jsonData
        );

        int responseCode = http.POST(jsonData);

        Serial.print("GreenSweep Worker Response Code: ");
        Serial.println(responseCode);

        Serial.println("GreenSweep Worker JSON:");
        Serial.println(jsonData);

        if (responseCode == 200) {
            greenSweepLastReportedCritical = isCritical;
        }

        http.end();

    } else {

        Serial.println("WiFi disconnected");
    }
}

```

```

// =====
// SETUP
// =====

void setup() {

    Serial.begin(115200);

    // Ultrasonic
    pinMode(TRIG_PIN, OUTPUT);
    pinMode(ECHO_PIN, INPUT);

    // Reed switch
    pinMode(REED_SWITCH_PIN, INPUT_PULLUP);

    // DHT22
    dht.begin();

    // =====
    // LED SETUP - ADDED
    // =====

    pinMode(GREEN_LED_PIN, OUTPUT);
    pinMode(YELLOW_LED_PIN, OUTPUT);
    pinMode(RED_LED_PIN, OUTPUT);

    digitalWrite(GREEN_LED_PIN, LOW);
    digitalWrite(YELLOW_LED_PIN, LOW);
    digitalWrite(RED_LED_PIN, LOW);

    // =====
    // WIFI
    // =====

    WiFi.begin(
        ssid,
        password
    );

    Serial.print("Connecting to WiFi");

    while (WiFi.status() != WL_CONNECTED) {

        delay(1000);

        Serial.print(".");
    }

    Serial.println();

    Serial.println("WiFi Connected");
}

```

```

Serial.print("ESP32 IP: ");

Serial.println(
  WiFi.localIP()
);

Serial.println(
  "Combined Smart Bin Firebase System Started"
);
}

// =====
// LOOP
// =====

void loop() {

  // =====
  // READ ULTRASONIC
  // =====

  float distance = getDistanceCM();

  int fillLevel =
    getFillPercentage(distance);

  // =====
  // READ REED SWITCH
  // =====

  int reedState =
    digitalRead(REED_SWITCH_PIN);

  String lidStatus;

  if (reedState == LOW) {

    lidStatus = "CLOSED";

  } else {

    lidStatus = "OPEN";

  }

  // =====
  // GET BIN STATUS
  // =====

```

```

String binStatus =
    getBinStatus(fillLevel);

// =====
// LED STATUS INDICATOR - ADDED
// =====

if (binStatus == "CRITICAL") {

    digitalWrite(GREEN_LED_PIN, LOW);
    digitalWrite(YELLOW_LED_PIN, LOW);
    digitalWrite(RED_LED_PIN, HIGH);

} else if (binStatus == "WARNING") {

    digitalWrite(GREEN_LED_PIN, LOW);
    digitalWrite(YELLOW_LED_PIN, HIGH);
    digitalWrite(RED_LED_PIN, LOW);

} else {

    digitalWrite(GREEN_LED_PIN, HIGH);
    digitalWrite(YELLOW_LED_PIN, LOW);
    digitalWrite(RED_LED_PIN, LOW);
}

// =====
// REPORT TO GREENSWEEP WORKER
// =====

sendCriticalStatusToGreenSweep(
    binStatus == "CRITICAL"
);

// =====
// READ DHT22
// =====

float temperature =
    dht.readTemperature();

float humidity =
    dht.readHumidity();

// =====
// READ MQ135
// =====

int mq135Value =

```

```

    analogRead(MQ135_PIN);

    // =====
    // READ MQ4
    // =====

    int mq4Value =
        analogRead(MQ4_PIN);

    // =====
    // SERIAL MONITOR
    // =====

    Serial.println(
        "===== SMART BIN ====="
    );

    if (distance == -1) {

        Serial.println(
            "Distance: No Reading"
        );

        Serial.println(
            "Fill Level: Unknown"
        );

    } else {

        Serial.print(
            "Distance: "
        );

        Serial.print(distance);

        Serial.println(" cm");

        Serial.print(
            "Fill Level: "
        );

        Serial.print(fillLevel);

        Serial.println(" %");

        Serial.print(
            "Bin Status: "
        );
    }

```

```

    Serial.println(binStatus);
}

// Reed switch

Serial.print(
    "Lid Status: "
);

Serial.println(lidStatus);

// DHT22

if (
    isnan(temperature) ||
    isnan(humidity)
) {

    Serial.println(
        "Failed to read DHT22"
    );

} else {

    Serial.print(
        "Temperature: "
    );

    Serial.print(temperature);

    Serial.println(" °C");

    Serial.print(
        "Humidity: "
    );

    Serial.print(humidity);

    Serial.println(" %");
}

// MQ135

Serial.print(
    "MQ135: "
);

Serial.println(mq135Value);

```

```

// MQ4

Serial.print(
  "MQ4: "
);

Serial.println(mq4Value);


// =====
// SEND TO FIREBASE 1
// =====

sendBinData(
  distance,
  fillLevel,
  binStatus,
  lidStatus
);


// =====
// SEND TO FIREBASE 2
// =====

if (
  !isnan(temperature) &&
  !isnan(humidity)
) {

  sendSensorData(
    temperature,
    humidity,
    mq135Value,
    mq4Value
  );
}

Serial.println(
  "===== "
);

Serial.println();

delay(5000);
}

```

Appendix D Research Budget

Table D.1: Revised research budget (this component)

Item	Cost (LKR)	Justification
ESP32 DevKit V1 x3	3,600	Development, testing and prototype units
MQ-135 x3	1,800	Cross-validation of calibration
MQ-4 x3	1,950	Same as MQ-135
DHT22 x3	1,200	Development, testing and prototype units
Breadboard, PCB, components, wires	2,500	Prototyping materials
IP54 enclosure and mounting materials	2,000	Physical prototype assembly
12 V battery and charge controller	4,500	Field deployment power system
Firebase Blaze plan (about 500/month x 4 months)	2,000	Cloud storage during testing
Fritzing and KiCad (open source)	0	Free
Contingency (10%)	1,955	Unforeseen failures or delays
Total	21,505	

The budget excludes the ultrasonic sensor, load cell and reference weights of the original proposal, which are no longer part of this component.

Appendix E Work Plan

Table E.1: Work plan by task

Task	Timeline	Deliverable
Sensor selection, literature review, procurement	March–April	Procurement list; updated literature review
Hardware assembly; individual sensor firmware modules	May–June	Breadboard prototype; sensor test code
Sensor calibration (MQ-135, MQ-4, DHT22)	June–July	Calibration records; DHT22 validation report
Integrated firmware; JSON serialisation; Wi-Fi and MQTT/HTTP	July–August	Integrated firmware; schema-validated JSON; 24-hour connection test
Firebase integration; threshold classification and LED; SPIFFS buffering	August–September	Live telemetry; validated status transitions; offline test
OTA update; PCB prototype; field deployment	September–October	OTA demonstration; PCB prototype; 7-day test data

Task	Timeline	Deliverable
Performance evaluation; dataset export; documentation / Thesis writing and submission	October–November	Evaluation report; CSV/JSON dataset; documentation