** The Islamia University of Bahawalpur**

**Smart Automated Dustbin**

**A project presented to**

**Department of Computer Science, IUB Bahawalpur**

**In partial fulfillment**

**of the requirement for the degree of**

***Bachelor of Science in Computer Science (2021-2025)***

**By**

**Aqsa Fareed  
F21NDOCS1M01079  
Minal Shafiq  
F21NDOCS1M01081  
Sana Ayub  
F21DOCS1M01063**

**2021-2025**

**DECLARATION**

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Aqsa   
 F21NDOCS1M01079  
  
 Minal   
 F21NDOCS1M01081  
  
 Sana   
 F21DOCS1M01063

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (CS) “Smart Automated Dustbin” was developed by   
  
 Aqsa Fareed F21NDOCS1M01079, Minal Shafiq F21NDOCS1M01081, Sana Ayub F21DOCS1M01063, under the supervision of Aamir Jamsheed and that in his opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Sciences.

Supervisor

-----------------------------------

External Examiner

------------------------------------

Chairman Department of Computer Science

-----------------------------------

**Acknowledgement**

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task.

We are greatly indebted to our project supervisor “Aamir Jamshed”. Without their personal supervision, advice and valuable guidance, completion of this project would have been doubtful. We are deeply indebted to them for their encouragement and continual help during this work.

And we are also thankful to our parents and family who have been a constant source of encouragement for us and brought us the values of honesty & hard work.

Aqsa

F21NDOCS1M01079

-----------------------------

Minal   
F21NDOCS1M01081

-----------------------------

Sana   
 F21DOCS1M01063

-----------------------------

**Abbreviations**

|  |  |
| --- | --- |
| **SRS** | Software Require Specification |
| **PC** | Personal Computer |
| **SDD** | Software Design Document. |
| **API** | Application Programming Interface. |

**Table of Content**

[1 Introduction 1](#_Toc198176263)

[1.1 Brief 1](#_Toc198176264)

[1.2 Relevance to Course Modules 1](#_Toc198176265)

[1.3 Project Background 1](#_Toc198176266)

[1.4 Related Material and Literature 1](#_Toc198176267)

[1.5 Analysis from Literature Review (in the context of your project) 2](#_Toc198176268)

[1.6 Methodology and Software Lifecycle for This Project 2](#_Toc198176269)

[1.6.1 Rationale behind Selected Methodology 2](#_Toc198176270)

[Rationale behind Selected Methodology 2](#_Toc198176271)

[2. Problem Definition 3](#_Toc198176272)

[2.1 Problem Statement 3](#_Toc198176273)

[2.2 Deliverables and Development Requirements 3](#_Toc198176274)

[Project Deliverables: 3](#_Toc198176275)

[Development Requirements: 3](#_Toc198176276)

[2.3 Current System 4](#_Toc198176277)

[3. Requirement Analysis 5](#_Toc198176278)

[3.1 Use Case Diagram 5](#_Toc198176279)

[3.2 Functional Requirements 5](#_Toc198176280)

[3.3 Non-Functional Requirements 6](#_Toc198176281)

[4. Design and Architecture 7](#_Toc198176282)

[4.1 System Architecture 7](#_Toc198176283)

[4.2 Data Representation [Diagram + Description] 8](#_Toc198176284)

[4.3 Process Flow/Representation 9](#_Toc198176285)

[5. Implementation 10](#_Toc198176286)

[5.1 Algorithm 10](#_Toc198176287)

[Pseudocode: 10](#_Toc198176288)

[5.2 External APIs 11](#_Toc198176289)

[5.3 User Interface 11](#_Toc198176290)

[6. Testing and Evaluation 12](#_Toc198176291)

[6.1 Manual Testing 12](#_Toc198176292)

[6.1.1 System Testing 12](#_Toc198176293)

[6.1.2 Unit Testing 12](#_Toc198176294)

[6.1.3 Functional Testing 13](#_Toc198176295)

[6.1.4 Integration Testing 13](#_Toc198176296)

[6.2 Automated Testing 14](#_Toc198176297)

[7. Conclusion and Future Work 15](#_Toc198176298)

[7.1 Conclusion 15](#_Toc198176299)

[7.2 Future Work 15](#_Toc198176300)

[REFRENCES 16](#_Toc198176301)

# 1 Introduction

This chapter provides an overview of the Smart Automated Dustbin project, its purpose, how it works, and how it connects with the subjects studied during the BS Computer Science program. It also discusses the background of the idea, relevant research, and the development approach followed for completing this project.

## Brief

The Smart Automated Dustbin project aims to bring innovation to waste management by automatically monitoring the fill level of a dustbin and notifying cleaning staff via email when it needs to be emptied. The project uses an ESP32 microcontroller, ultrasonic sensors for distance measurement, LEDs to indicate the bin status, a buzzer for alerts, and servo motors for automatic lid control. Email alerts are sent using the ESP-Mail-Client library through Gmail's SMTP protocol. The system minimizes false detections by verifying the garbage level using five consecutive readings. This report covers all technical, functional, and testing aspects of the system.

## Relevance to Course Modules

This project relates directly to the modules studied during the BS in Computer Science, such as:

* **Internet of Things (IoT)** – ESP32, sensors, actuators, and smart systems
* **Embedded Systems** – Working with microcontrollers and hardware components
* **Software Engineering** – Requirement analysis, SDLC, and documentation
* **Computer Networks** – Use of SMTP protocol for communication
* **Programming Fundamentals** – Writing clean and optimized code in C++

## Project Background

The idea behind the Smart Automated Dustbin originated from the increasing need for efficient waste management in both public and private spaces. Overflowing dustbins are not only unpleasant but also unhygienic. Traditional bins do not offer a way to monitor their fill level. This project seeks to address that by using real-time monitoring and automated alerts, reducing manual checking and ensuring timely cleaning.

## Related Material and Literature

Similar solutions have been implemented using IoT platforms where garbage bins are fitted with sensors and connected to centralized monitoring systems. Studies suggest that automated waste tracking improves efficiency by more than 30%. Research papers also highlight the need for intelligent systems to support municipal services and reduce operational costs. (Nakanishi et al., 1996; Balasubramanian & Prasanna, 2020)

## Analysis from Literature Review (in the context of your project)

Unlike many existing systems that use a single reading for detection, our dustbin project incorporates a false detection filter by using five consistent readings. This enhances reliability and reduces error rates. Additionally, the system does not rely on costly cloud infrastructure; instead, it uses direct email alerts, making it cost-effective and accessible.

## Methodology and Software Lifecycle for This Project

The Iterative Waterfall Model was used for this project. First, the requirements were gathered and analyzed. Then, hardware selection and coding were carried out in phases, with testing after each phase. Feedback and errors from testing were used to refine and improve the next iteration. (Behforooz & Hudson, 1996)

### Rationale behind Selected Methodology

The Waterfall model was chosen because it offers a clear, phase-wise structure suitable for hardware-software integrated projects. It allowed testing and evaluation after every step before moving forward, ensuring stable and reliable integration of hardware components with the firmware.

## Rationale behind Selected Methodology

*The Iterative Waterfall Model* was selected because it allowed development in clear, manageable phases. It also offered flexibility to go back and adjust things when issues were found. Since the project includes both software and physical hardware, this model made it easier to manage integration issues, test components individually, and verify output at every stage before final deployment.

# 2. Problem Definition

This chapter defines the exact problem our project is addressing, outlines the project deliverables, and compares it briefly with the current systems in use. It also explains what makes our solution different and necessary.

## 2.1 Problem Statement

In many public and private areas, dustbins often overflow because cleaning staff do not have real-time information about their status. This leads to unhygienic environments, bad smell, and visual pollution. The lack of a smart notification system causes delays in cleaning. Manual checking of each bin is inefficient and wasteful. Also, false triggering due to temporary obstructions like hands or random objects can lead to unnecessary alerts.

The problem is to design a smart and reliable system that can detect garbage levels accurately, avoid false detections, and inform the cleaning team automatically via email when the bin is half or completely full.

## 2.2 Deliverables and Development Requirements

### Project Deliverables:

* A working prototype of the Smart Automated Dustbin
* Source code for the ESP32 microcontroller
* Email notification system using SMTP
* Real-time level detection with visual (LEDs) and audio (buzzer) alerts
* A final documentation/report of the project

### Development Requirements:

#### *Hardware:*

* ESP32 microcontroller
* Ultrasonic sensors (2 units: 1 inside the lid, 1 outside for user detection)
* Servo motor (for lid movement)
* LEDs (Red, Yellow, Green)
* Buzzer
* Jumper wires, breadboard
* 5V power supply or battery

#### *Software:*

* Arduino IDE
* ESP-Mail-Client library
* Gmail SMTP settings
* Serial Monitor for testing and debugging

#### *Other:*

* Internet connection for Wi-Fi enabled board
* Valid sender Gmail account for SMTP email

## 2.3 Current System

Currently, most dustbins are not automated. Cleaning staff must manually check each bin at regular intervals, which wastes time and labor. Some advanced systems use GSM modules to send SMS alerts, but they often send false notifications because they do not verify readings. Also, many cloud-based IoT solutions are costly and difficult to implement for small-scale or budget-constrained projects.

Our system is designed to be simple, cost-effective, and reliable. It uses basic components but offers high efficiency by filtering out false detections and only sending alerts when truly necessary. It also visually indicates the dustbin status using LEDs and provides a buzzer alert during the email sending process.

*Figure 2.1: Smart Automated Dustbin Prototype*

| **Component Name** | **Quantity** | **Purpose** |
| --- | --- | --- |
| ESP32 Wi-Fi Module | 1 | Main microcontroller |
| Ultrasonic Sensor | 2 | Level detection & user presence |
| Servo Motor | 1 | Lid movement |
| LED (Red, Yellow, Green) | 3 | Status indication |
| Buzzer | 1 | Alert feedback |
| Breadboard + wires | 1 set | Circuit building |

*Table 2.1: List of Hardware Components Used*

# 3. Requirement Analysis

This chapter outlines the functional and non-functional requirements of the Smart Automated Dustbin project. It also includes a Use Case Diagram to explain the system interactions and user roles.

## 3.1 Use Case Diagram

Figure 3.1: Use Case Diagram for Smart Automated Dustbin

## 3.2 Functional Requirements

Functional requirements define the specific behavior and functions of the system. The Smart Automated Dustbin is expected to perform the following functions:

• Detect garbage level using an ultrasonic sensor

• Confirm readings using **5 consecutive measurements**

• Detect **“Half Full”** status (between 10cm and 20cm)

• Detect **“Full”** status (less than 10cm)

• Send an email notification to the cleaner for both half and full status

• Alert the environment using a buzzer and **LEDs** based on fill level

• Detect a person approaching the dustbin and automatically open the lid using a servo motor

• Reset the system once the garbage is emptied

## 3.3 Non-Functional Requirements

Non-functional requirements describe the system's quality attributes. For this project, they include:

• **Reliability** – The system should send alerts only after verifying with 5 readings to prevent false detection.

• **Performance** – Real-time distance measurement and response within seconds.  
• **Usability** – Easy to use and interpret, with LEDs and buzzer for feedback.  
• **Maintainability** – Code should be clean and well-documented for easy updates.  
• **Portability** – The system should be compact and deployable in different locations.  
• **Security** – Email credentials and communication must remain secure (used App Passwords for Gmail).

# 4. Design and Architecture

This chapter covers the design details and architectural structure of the Smart Automated Dustbin. It includes the system architecture, data representation, and process flow diagrams. The design was focused on keeping the system cost-effective, reliable, and easy to maintain.

## 4.1 System Architecture

The system consists of both hardware and software components working together. The core controller is the **ESP32**, which collects data from two ultrasonic sensors. One sensor is placed on the inner lid of the dustbin to monitor the garbage level, and the other is placed on the outer side to detect the presence of a person.

Once the garbage level is measured and verified (through 5 readings), the ESP32 sends an alert via email using the SMTP protocol. Based on the detection, corresponding LEDs are turned on (Green for Empty, Yellow for Half Full, Red for Full), and a buzzer is triggered for additional alerting.

The second sensor detects when someone approaches the bin. If a person is detected, the servo motor is activated to open the lid automatically.

*Figure 4.1: System Architecture of Smart Automated Dustbin*

## 4.2 Data Representation [Diagram + Description]

The main data being processed is the **distance** measured by the ultrasonic sensor, which determines the status of the dustbin (Arduino, 2023). Here's how the distances are interpreted:

* + 0–10 cm: **Full**
  + 11–20 cm: **Half Full**
  + 21 cm and above: **Empty**

***Figure 4.2: Garbage Level Threshold Data Flow***

Each distance value is collected and stored temporarily. After five readings, the program checks how many were within the critical thresholds before making a final decision.

## 4.3 Process Flow/Representation

The following describes the basic flow of operations in the Smart Automated Dustbin:

1. Power is supplied and the ESP32 connects to Wi-Fi
2. Ultrasonic sensor inside the lid checks garbage level
3. Five readings are taken to avoid false detection
4. Based on the result:
   * + If Full → Red LED ON, Buzzer ON, Email Sent
     + If Half Full → Yellow LED ON, Buzzer Short Beeps, Email Sent
     + If Empty → Green LED ON, No Email
5. Simultaneously, the second ultrasonic sensor detects approaching person
6. If a person is detected → Servo motor opens lid
7. System resets after delay and loops back

*Figure 4.3: Process Flow Diagram*

# 5. Implementation

This chapter explains how the Smart Automated Dustbin was implemented, including the logic (algorithm), external APIs/libraries used, and user interface design. It outlines how the components interact with each other in real time to achieve the final system functionality.

## 5.1 Algorithm

The Smart Dustbin system follows a simple and efficient algorithm to ensure accuracy in detection and response. Below is the high-level pseudocode that represents the behavior of the system:

### Pseudocode:

START

Connect to Wi-Fi

Initialize all pins (LEDs, Buzzer, Servo, Ultrasonic Sensors)

LOOP:

fullCount = 0

halfCount = 0

Repeat 5 times:

Read distance from ultrasonic sensor

IF distance ≤ 10cm THEN

fullCount++

ELSE IF distance ≤ 20cm THEN

halfCount++

WAIT 3 seconds

IF fullCount ≥ 5 THEN

Show Red LED

Turn ON Buzzer (Long)

IF Email not sent THEN

Send Email ("FULL")

Mark email as sent

ELSE IF halfCount ≥ 5 THEN

Show Yellow LED

Turn ON Buzzer (Short Beeps)

IF Email not sent THEN

Send Email ("HALF FULL")

Mark email as sent

ELSE

Show Green LED

Turn OFF Buzzer

Reset email flags

Check outer sensor for person

IF person detected THEN

Open lid using Servo

Wait few seconds

Close lid

WAIT and repeat loop

This logic helps in minimizing false triggers and improves the overall reliability of the system.

## 5.2 External APIs

The system uses the **ESP-Mail-Client** library to send emails through Gmail’s SMTP server. This helps avoid using any cloud platform or dashboard, keeping the setup lightweight and cost-effective. (TutorialsPoint, 2023).

|  |  |  |
| --- | --- | --- |
| **Name of API** | **Description** | **Purpose of Usage** |
| ESP-Mail-Client | A library for sending emails over SMTP in ESP32 | Used to send email alerts when bin is full/half full |
| WiFi.h | Inbuilt ESP32 Wi-Fi library | Used for connecting ESP32 to Wi-Fi |
| Servo.h | Arduino servo motor library | |  | | --- | | Controls the opening/closing of dustbin lid | |

Table 5.1: Details of APIs used in the project

These libraries were integrated into the code using the Arduino IDE. The email-sending function is executed only when a status (FULL or HALF FULL) is verified by the readings.

## 5.3 User Interface

Since this is a hardware-based IoT system, the user interface is physical and very simple:

* + **Red LED** – Indicates the dustbin is FULL
  + **Yellow LED** – Indicates the dustbin is HALF FULL
  + **Green LED** – Indicates the dustbin is EMPTY
  + **Buzzer** – Gives short or long beeps when the bin is half or fully full
  + **Servo Motor** – Automatically opens the lid when a person is nearby
  + **Email Notification** – Sent directly to the cleaner’s email as a styled HTML message

The user doesn't need to interact with any app or dashboard. The combination of light, sound, and email ensures both local and remote feedback is clear and timely.

# 6. Testing and Evaluation

This chapter outlines the various testing methods used to evaluate the Smart Automated Dustbin. Since this is a hardware and firmware-based project, manual testing played a critical role. We performed system testing, unit testing, functional testing, and integration testing to ensure all components work properly together.

## 6.1 Manual Testing

Manual testing was done at each stage of development to catch bugs early and ensure correct behavior. Tests were repeated multiple times to confirm consistency.

### 6.1.1 System Testing

System testing ensured the entire system—hardware and software—functioned as expected. Each sensor, LED, buzzer, and email function was tested with different fill levels and external conditions.

### 6.1.2 Unit Testing

#### Unit Testing 1: Garbage Level Detection Logic

**Testing Objective:** To ensure that garbage level detection and LED response work properly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Detect distance = 8cm | Ultrasonic reading | Red LED ON, buzzer long beep, email sent | Pass |
| 2 | Detect distance = 15cm | Ultrasonic reading | Yellow LED ON, buzzer short beeps, email sent | Pass |
| 3 | Detect distance = 25cm | Ultrasonic reading | Green LED ON, no buzzer, no email | Pass |

Garbage Level Detection Logic

#### Unit Testing 2: Person Detection and Lid Opening

**Testing Objective:** To ensure that the servo motor opens the lid when a person is detected.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Person detected within 20cm | Outer ultrasonic | Servo motor rotates (lid opens) | Pass |
| 2 | No person detected | Outer ultrasonic | Lid remains closed | Pass |

Person Detection and Lid Opening

### 6.1.3 Functional Testing

Functional testing was conducted to check that each feature behaves according to the requirements.

#### Functional Testing 1: Email Alert System

**Objective:** To confirm the correct email is sent for each garbage level.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Distance = 8cm | Confirmed by 5 readings | Email sent with status “FULL” | Pass |
| 2 | Distance = 16cm | Confirmed by 5 readings | Email sent with status “HALF FULL” | Pass |
| 3 | Distance = 25cm | Confirmed by 5 readings | No email sent | Pass |

*Email Alert System*

### 6.1.4 Integration Testing

Integration testing ensured all modules worked smoothly when combined.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | System startup | All components | Wi-Fi connects, components initialize | Pass |
| 2 | Full condition + person detected | Sensor + Servo | Email sent, buzzer ON, lid opens | Pass |
| 3 | Half full condition + no person | Sensor + LED | Email sent, yellow LED, buzzer ON | Pass |
| 4 | Empty + person detected | Sensor + Servo | No email, green LED ON, lid opens | Pass |

*Integration Testing*

## 6.2 Automated Testing

As this project primarily involves hardware components and sensor readings, automated testing was limited. However, for debugging and simulation, the **Serial Monitor in Arduino IDE** was used to observe sensor values and system responses.

**Tools used:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tool Name** | **Tool Description** | **Applied on** | **Results** |
| Arduino Serial Monitor | Monitors live output and logs | Sensor distance, email status, LEDs | Useful for debugging |
| ESP-Mail-Client Library | Simulates email process | SendMail function | Verified email delivery |

Automated Testing

# 7. Conclusion and Future Work

This chapter summarizes what was achieved through the project and suggests ideas for how the system can be improved in the future.

## 7.1 Conclusion

The Smart Automated Dustbin project successfully demonstrates how IoT technology can improve everyday systems like waste management. The system was able to detect garbage levels accurately using an ultrasonic sensor, avoid false triggers through five consistent readings, and send real-time email alerts using SMTP protocol. It also provided clear physical feedback using LEDs and a buzzer and automatically opened the lid when a person approached.

One of the main achievements was the accuracy and reliability of the system, especially its ability to distinguish between actual garbage and random or false detections. The use of cost-effective components and libraries made the system affordable and practical for public places, homes, offices, and institutions.

Overall, the project met its main objectives and proved that automation in waste handling can be achieved with simple but smart solutions.

## 7.2 Future Work

While the current system performs well, there are several areas where it can be enhanced in the future:

• Add a mobile app or web dashboard to monitor multiple dustbins remotely

• Integrate GSM-based fallback alerts in case Wi-Fi is unavailable

• Use solar panels for power in outdoor environments

• Store sensor and alert data in a cloud database for analysis and reporting

• Add a camera to capture images when the bin is full for verification

• Improve the enclosure design to make the system more weatherproof and robust

• Extend the system to handle multiple types of waste (e.g., recyclable, organic, hazardous)

These upgrades would make the system more advanced, flexible, and suitable for large-scale implementation in smart cities or public institutions.

# REFRENCES

1. Behforooz, A., & Hudson, F. J. (1996). *Software Engineering Fundamentals*. Oxford University Press. Chapter 8, pp. 255–235.  
   🔹 *Use in*: **1.6 Methodology and Software Lifecycle** — when discussing SDLC models like Iterative Waterfall.
2. Page Author. “SMTP protocol and its use in IoT systems.” https://www.tutorialspoint.com/simple-mail-transfer-protocol-smtp  
   🔹 *Use in*: **5.2 External APIs** — when explaining SMTP use for email sending.
3. Arduino Reference – Ultrasonic Sensor (HC-SR04). https://www.arduino.cc/en/Tutorial/Ping  
   🔹 *Use in*: **4.2 Data Representation** — when describing how distance is calculated with sensors.
4. ESP-Mail-Client Library Documentation. <https://github.com/mobizt/ESP-Mail-Client>  
   🔹 *Use in*: **5.2 External APIs** and **6.2 Automated Testing**
5. Nakanishi, H., Yoshida, C., Nishimura, T., & Ishida, T. (1996). *FreeWalk: Supporting Casual Meetings in a Network*. CSCW 96, ACM, pp. 308–314.  
   🔹 *Use in*: **1.4 Related Material and Literature** — as a reference to past IoT research implementations.
6. K. Balasubramanian & M. Prasanna, “Smart Waste Management Using IoT,” *International Journal of Engineering Research & Technology*, Vol. 9, Issue 7, July 2020.  
   🔹 *Use in*: **1.4 Related Material and Literature** and **1.5 Analysis from Literature Review**