



Professional Engineering Practice and Industrial Management

2nd Year, 2nd Semester

Final Report

Eco BinBot

Submitted to

Sri Lanka Institute of Information Technology

In partial fulfillment of the requirements for the
Bachelor of Science Special Honors Degree in Information Technology

18th of June 2024





Declaration

We declare that this project report or part of it was not a copy of a document done by any organization, university any other institute or a previous student project group at SLIIT and was not copied from the Internet or other sources.

Project Details

Project Title	Eco BinBot
Project ID	PEP_18

Group Members

Reg. No	Name	Signature
IT22069122	BANDARANAYAKE K B H M C C	
IT22319455	ATHUKORAL V T	
IT22349392	DULAKSHI P M D	
IT22342126	WIJETUNGA S S	

Abstract

In today's fast-paced world, people often forget to check their garbage bins, leading to overflowing bins, unpleasant odors, and pest infestations. One of the primary issues is the lack of real-time monitoring, which leads to overflowing bins and delaying waste collection. These inefficiencies result in unsanitary conditions, health hazards, and environmental degradation. Our project addresses these issues with an IoT-based smart bin system equipped with sensors, ensuring timely waste management. Functionalities of the sensors and units are checking the waste level constantly, checking for toxic gases, charging the system using solar panel, provide human friendly mobile application and automatic lid opening. The Eco BinBot system includes a user interface called Blynk mobile application that allows users to access real-time data on waste levels, battery voltage and gas detection with an alert message. Results from our project indicate significant improvements in waste management efficiency, reduction in manual intervention, and promotion of environmental sustainability. The Eco BinBot offers a scalable and eco-friendly solution to modern waste management challenges.

Keywords – ESP8266 module , Blynk app, HC-SR04 ultra sonic sensor, automatic flip opening, monitoring waste level, gas detection, servo motor, MQ-135 Air quality gas detection module, solar panel, power unit.

Acknowledgement

We would like to express our sincere gratitude to our PEPIM lecturer Madam Narmada Gamage, supervisor Ms. Nethmi, Sir Ashvinda and all the instructors, for their invaluable guidance and support throughout this project. We also extend our thanks to the Sri Lanka Institute of Information Technology for providing us with the necessary resources and platform to develop this project and providing us a knowledge bank of project management skills in the Y2S2 PEPIM module.

Table of Contents

1. Introduction.....	7
1.1 Problem Statement.....	7
1.2 Product Scope.....	7
1.3 Project Report Structure	8
2. Methodology	9
2.1 Requirements and Analysis	9
2.2 Design.....	11
2.2.1 High level architecture diagram.....	11
2.2.2 Circuit diagram.....	12
2.2.3 User interface	13
2.3 Implementation.....	14
2.3.1 Sensor Modules	14
2.4 Testing.....	22
3. Conclusion	29
3.1 Assessment of the Project Results	29
3.2 Lessons Learned	29
3.3 Future Work	29
4. References.....	30
Appendix A: Design Diagrams	31
Appendix B: Selected Code Listings	32

List of Figures

Figure 2.1 : <i>Use case diagram for the web browser user interface of the Eco BinBot smart bin.</i>	10
Figure 2.2: <i>High-level architecture diagram of the smart Eco BinBot bin...</i>	Error! Bookmark not defined.
Figure 2.3 : <i>Circuit diagram of the smart Eco BinBot bin</i>	Error! Bookmark not defined.
Figure 2.4 : <i>Blynk app – user interface.....</i>	13
Figure 2.5 : <i>flowchart used to program the ESP module to process the data gathered via the Sonar sensor.....</i>	14
Figure 2.6 : <i>flowchart used to program the ESP module to process the data gathered via the Gas sensor</i>	16
Figure 2.7 : <i>flowchart used to program the ESP module to process the data gathered via the ultrasonic sensor</i>	18
Figure 2.8 : <i>flowchart used to program the ESP module to process the data gathered via the Charging unit.....</i>	19
Figure 0.1 : <i>Software requirements for the smart Eco BinBot.....</i>	31
Figure 0.2 : <i>Hardware requirements for the smart Eco BinBot</i>	31

List of Tables

Table 2.1 : Automatic Lid Opening	22
Table 2.2 : Fill level Monitoring.....	24
Table 2.3 : Gas level detection.....	25
Table 2.4 : User Interface.....	27
Table 2.5 : ESP module	28
Table 2.6 : Charging Unit	28

List of Acronyms and Abbreviations

ESP8266	ESP8266 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability, developed by Espressif Systems.
Use case diagram	Describing the functionality of a system in terms that the client can understand.
High level architecture diagram	A visual representation that outlines the overall structure and interconnectedness of a system at a high level of abstraction.
Real time monitoring	Real-time monitoring refers to the continuous, immediate observation and analysis of various parameters or processes using technology to

1. Introduction

1.1 Problem Statement

- In today's fast-paced world, individuals often inadvertently neglecting routine tasks such as checking their garbage bins.
- This oversight can lead to overflowing bins, unpleasant odors, and even pest infestations.
- Our proposed project aims to address these concerns through the implementation of an IoT-based smart bin system. We aim to alleviate the burden on individuals by providing a convenient and automated solution to waste management through the integration of Internet of Things (IoT) technology with dustbins.

1.2 Product Scope

- In scope –
 - The Eco BinBot's main objective is to keep the environment clean by notifying the user when the garbage bin has reached a level which needs to be collected. The bin is equipped with sensors for monitoring waste levels and gas emissions with an automatic flip opening with the aid of a servo motor and it uses solar panel for power. This bin can be used by individual homeowners, companies or the municipal council. Our initial plan is to create a product that can be used in an industrial site, where they can automate the garbage collection system by integrating the bin with the municipal council. The main outcome of the project is to increase the quality of life of the users by giving them a chance to not worry whether their garbage bins will overflow or emit unpleasant smells by the ease of remotely monitoring the bin using the user's mobile phone by notifying by the Blynk mobile app when the bin fills up or it starts emitting unpleasant gases and also when the power is down.

- Out scope –
 - This device is not programmed to detect all the gases that are emitted from waste material in the Eco BinBot.
 - A garbage compactor was not added to the design due to time constraints.
 - The device will not function properly if kept in a low light area for a long time.
 - The reaction time of the door is inadequate due to the limited interrupt handling capability of the ESP8266 microcontroller. The door opens and closes at a slow rate, which can be attributed to the low performance of the ESP8266 chip. While suitable for hobbyist projects, it lacks the industrial-grade capabilities required for this application.
 - The sensors and other components used in the system are of relatively low quality. As commercially available developer-level components, they may not be designed to handle voltage, current, and noise levels present in an industrial environment reliably.
 - The input signals from the sensors are prone to disturbances from external noise and grounding issues. This can lead to erratic behavior or incorrect readings, further impacting the system's responsiveness.
 - The main platform is a dot-board where components were manually soldered. This introduces potential issues such as poor heat dissipation, cold solder joints, and bit flips in the input signals, all of which can degrade the system's performance and reliability.
 - Cloud Syncing errors between the Blynk cloud and the 2 main ESP8266 boards.

1.3 Project Report Structure

- The report is organized as the introduction covers the problem statement and product scope, the methodology section describes the requirements, design, implementation, and testing processes. The conclusion summarizes the project including the lesson learned and future work, and finally the references and appendices which supported to develop this report.

2. Methodology

2.1 Requirements and Analysis

- Functional requirements –
 - Automatic flip opening with the aid of the servo motor using sonar sensor.
 - Waste level monitoring in the bin using sonar sensor displaying the waste level status on the Blynk mobile app.
 - Gas level detection and giving an alert msg in the presence of toxic gases.
 - Automated notifications given to the users through the user interface, which is the Blynk app, leveraging the ESP8266 Wi-Fi capability.
 - System powered by solar panels and monitor battery voltage in the user interface.
 - ESP8266 module processing sensor data and transmit it to the Blynk app.
- Non- functional requirements –
 - Communication latency between the bin and the server is minimal to ensure timely alerts.
 - The sensors and components are equipped in the bin with safety standards to prevent hazards.
 - The system maintains operational availability 24/7
 - The user interface is user-friendly, allowing new users to navigate easily with the basic features and understand the alerts and notifications clearly.
 - The system provides environmental sustainability by reducing manual waste management efforts.

- Use case diagram –

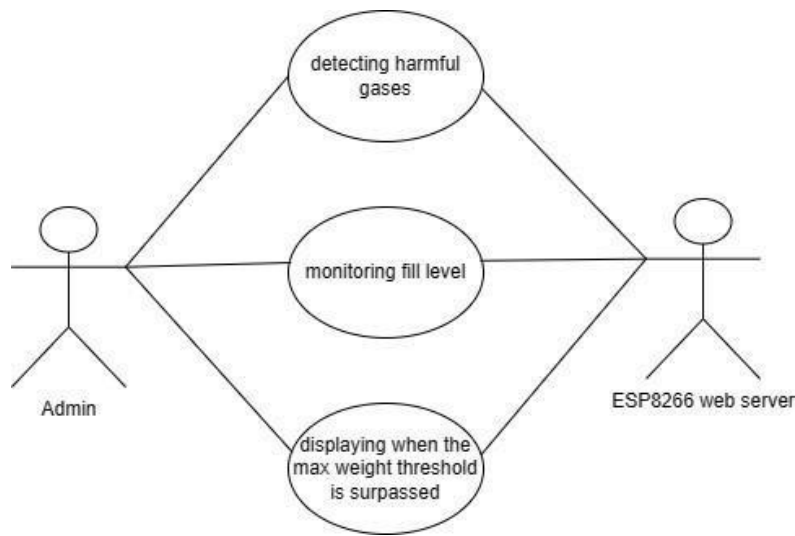


Figure 2.1 : *Use case diagram for the web browser user interface of the Eco BinBot smart bin.*

- The above diagram depicts the use case diagram for the user interface (Blynk app) of our Eco BinBot smart bin.
- The two actors that directly interact with our system are administrator (human) and the ESP8266 web server (software).
- Detecting harmful gases, monitoring fill level and battery voltage are the use cases that describe the functionalities of the user interface provided to the administrator.
- The administrator monitors the real-time sensor data displayed through the Blynk app.
- The ESP8266 web server retrieves processed sensor data and display it remotely through the user interface (Blynk app).

2.2 Design

2.2.1 High level architecture diagram.

- The high-level architecture diagram for the Smart Eco BinBot provides a comprehensive overview of the system's components and their interactions. Initially, the design included an HC-SR505 mini-PIR sensor module to detect motion and trigger the automatic flip opening of the bin. However, during the progress presentation, we encountered issues with the PIR sensor, and it did not function as expected. As a result, we decided to use an ultra-sonic sensor (HC-SR04) in conjunction with a servo motor to achieve automatic flip opening. The ultra sonic sensor also measures the distance to the waste level inside the bin, and when the waste reaches a certain threshold, it triggers the servo motor to open the bin automatically. Additionally, we incorporated the MQ-135 gas sensor to detect various hazardous gases ensuring air quality. The ESP8266 module serves as the central processing unit, handling data from the sensors and communicating with the user interface via Wi-Fi. Due to time constraints, we could not implement the HX711 weighing sensor module, which was intended to measure the weight of the waste. The entire system is powered by a solar panel, depicting sustainability. The high-level architecture ensures efficient waste level monitoring and gas detection, enhancing the functionality and safety of the waste management system.

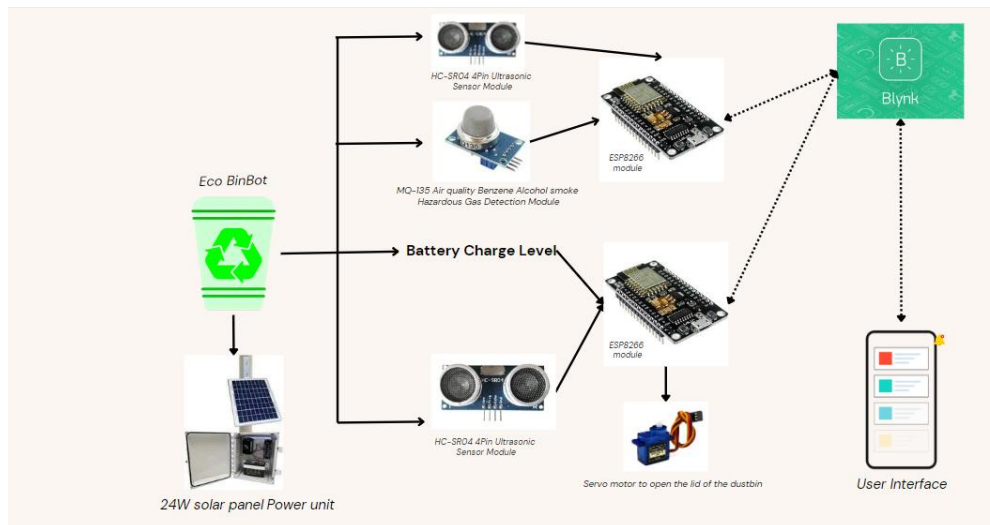


Figure 2.2 : High Level Architecture Diagram of the Smart Eco Bin

2.2.2 Circuit diagram

- The circuit diagram illustrates the integration of various components for an automated waste management system. At the core is the ESP8266 module, which processes data and communicates with the Blynk app. The HC-SR04 ultrasonic sensor measures the waste level, triggering the servo motor for automatic flip opening when a certain threshold is reached. The MQ-135 gas sensor monitors air quality by detecting hazardous gases. A solar panel and batteries provide sustainable power to the system, managed by a charging module.

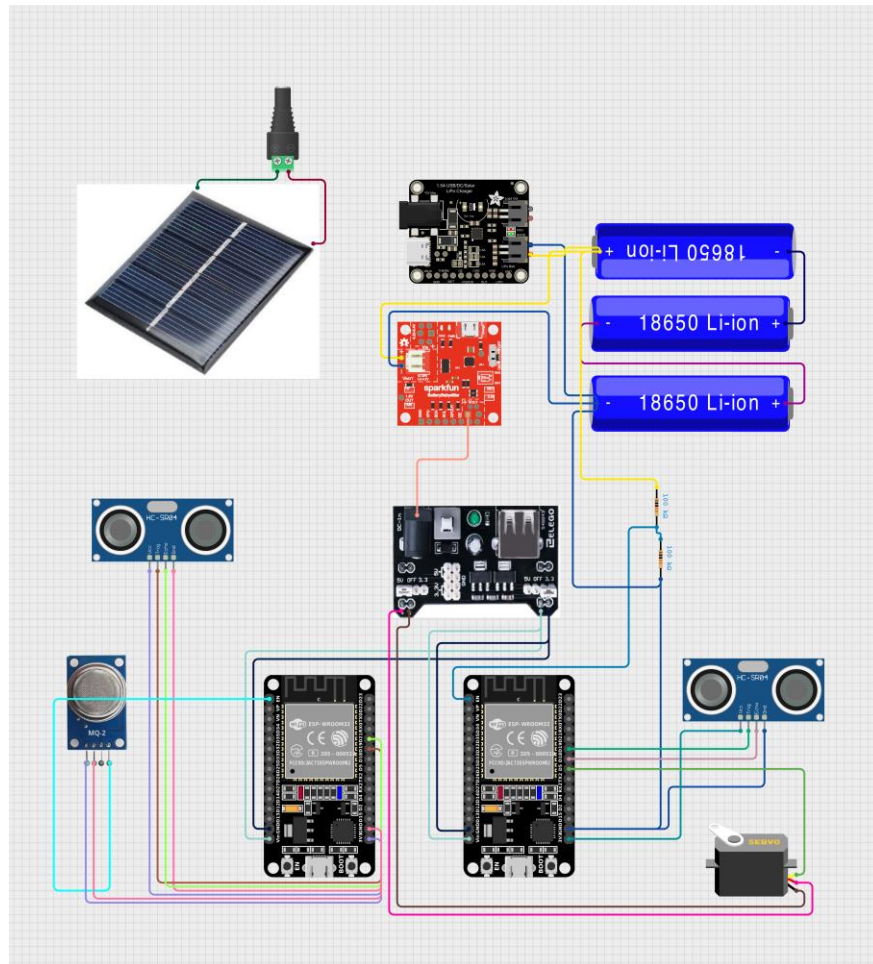


Figure 2 3 : Circuit Diagram of the Smart Eco BinBot

2.2.3 User interface

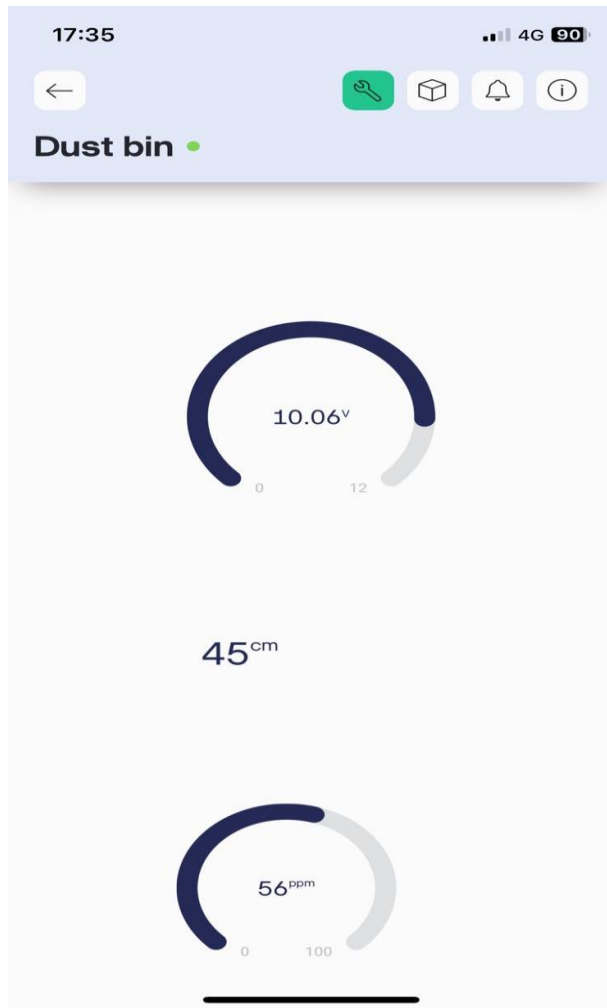


Figure 2.3 : Blynk app – user interface

- The Blynk app serves as the user interface for our automated Eco BinBot , providing real-time monitoring and control. Users can access live data on waste levels, air quality, and battery voltage directly from their smartphones. Alert notification comes when hazardous gases are detected and if the waste bin reaches full. This user friendly interface enhances convenience and ensures efficient management and monitoring of the waste disposal process.

2.3 Implementation

2.3.1 Sensor Modules

Ultrasonic Sensor

- Two ultrasonic sensors are utilized to continuously measure the distance to detect the fill level of the bin and to detect the presence of a person within a predefined range to trigger the automatic lid opening function.
- Process: Sensor emits a pulse of ultrasonic wave, once it encounters an object it reflects the wave, and a receiver picks it up. Using the time interval between the sending and reception of the echo the distance is calculated. If it matches the predefined distances the ESP sends the data to the mobile application through the server.
- Output:
 - Fill level Detection- Sensor attached to the underside of the bin lid is used. The calculated distance to the surface level of garbage is displayed
 - Person Detection- Sensor attached to the outside of the bin lid is used. Sensor detects if an object is present within a threshold level to automate lid opening.

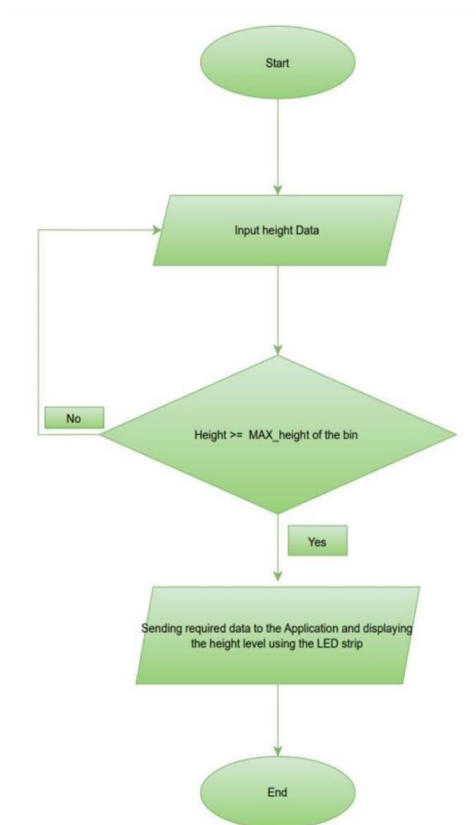


Figure 2.4 : *flowchart used to program the ESP module to process the data gathered via the Sonar sensor.*

- Code snippets used:

- For Fill level Monitoring- Display the fill level in Blynk cloud interface

```
double distance = getSonarDistance();  
Serial.print("Distance: ");  
Serial.print(distance);  
Serial.println(" cm");  
  
Blynk.virtualWrite(V1, distance);
```

- For automated bin lid opening- activate the servo motor depending on the distance measurements

```
if (distance <= distanceThreshold) {  
    // If an object is within the threshold, open the dustbin door (90 degrees)  
    moveServo(servoOpenPosition);  
} else {  
    // If no object is within the threshold, close the dustbin door (0 degrees)  
    moveServo(servoClosedPosition);  
}
```

(see appendix C for more)

MQ-2 gas sensor

- To continuously monitor and determine harmful gas levels in the bin.
- Process: Get sensor readings from the gas sensor. Compare the values to a predefined threshold. Trigger an alert on the web application if the threshold level is exceeded.

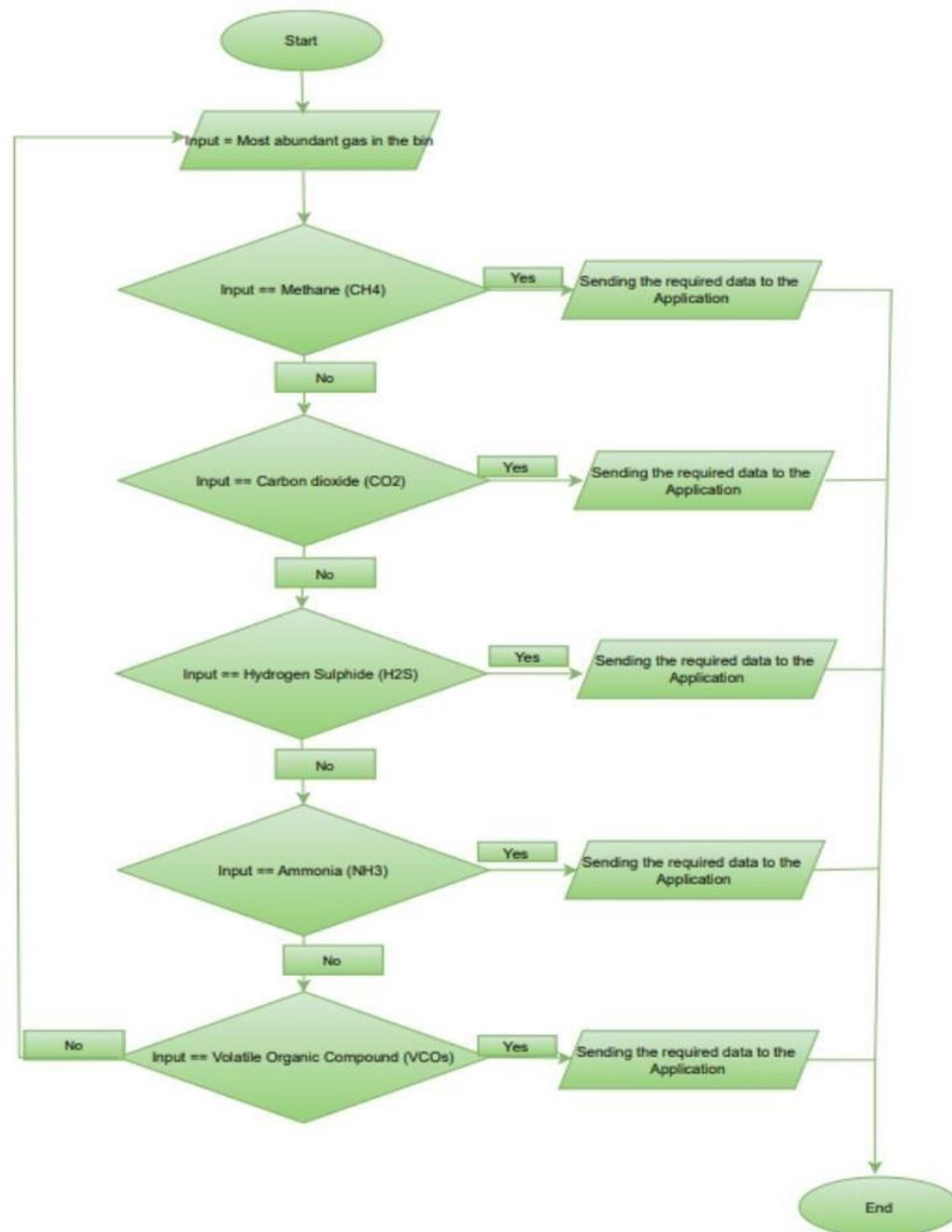


Figure 2.5 : flowchart used to program the ESP module to process the data gathered via the Gas sensor

- Code Snippet:

```
double getAlcoholLevel() {  
    double sensorValue;  
    sensorValue = analogRead(MQ2pin); // read analog input pin A0  
  
    Serial.print("Sensor Value: ");  
    Serial.print(sensorValue);  
  
    // Determine the status  
    if (sensorValue > Threshold) {  
        Serial.println(" | Smoke detected!");  
    } else {  
        Serial.println(" | Normal!");  
    }  
    return sensorValue;  
}
```

(see appendix C for more)

Servo motor control:

- Servo motor is integrated to open and close the bin lid based on the presence of a person detected by the ultrasonic sensor. It is controlled to move to a specific position defined by an angle based on sensor inputs.

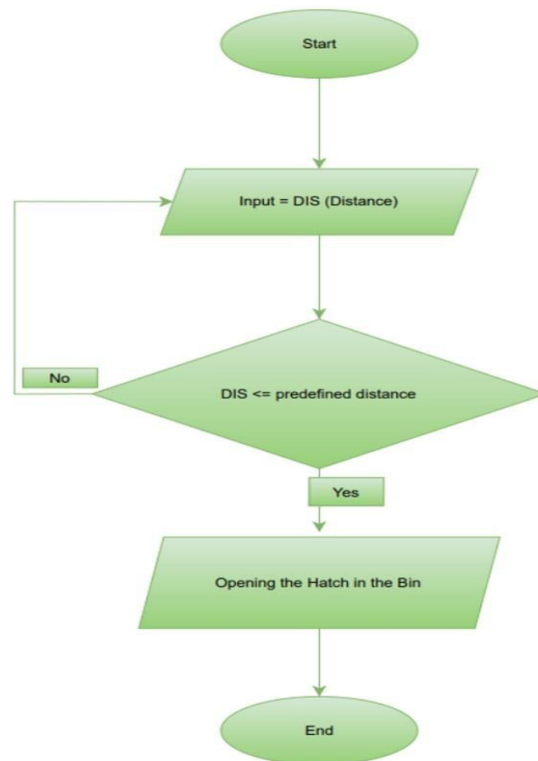


Figure 2.6 : *flowchart used to program the ESP module to process the data gathered via the ultrasonic sensor*

- Code Snippet to move the server motor depending on the current position of the actuator and the sensor value:

```
if (currentPosition < targetPosition) {  
    for (int pos = currentPosition; pos <= targetPosition; pos++) {  
        myServo.write(pos); // Move the servo to the current position  
        delay(servoSpeed); // Adjust the speed of the servo  
    }  
} else {  
    for (int pos = currentPosition; pos >= targetPosition; pos--) {  
        myServo.write(pos); // Move the servo to the current position  
        delay(servoSpeed); // Adjust the speed of the servo  
    }  
}  
(see appendix C for more)
```

Charging Unit

- Process: Charging the charging unit using solar power if the battery charge level is lower than the predefined level, if not supply power to the ESP module.

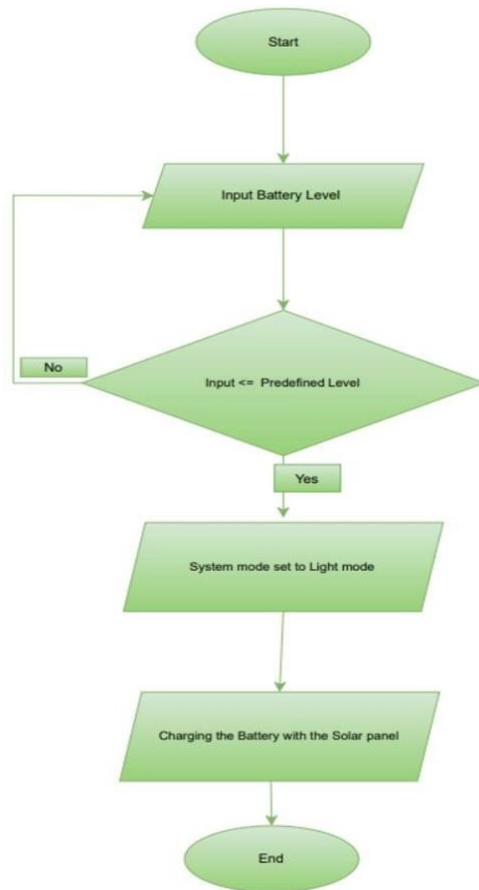


Figure 2.7 : *flowchart used to program the ESP module to process the data gathered via the Charging unit*

- Code snippet to display the battery voltage in the user interface:

```
Serial.print("Battery Voltage: ");  
Serial.println(batteryVoltage);
```

```
// Send the battery voltage to Blynk Virtual Pin V3  
Blynk.virtualWrite(V3, batteryVoltage);
```

(see appendix C for more)

ESP8266 Module

- This module is used to provide Wireless connectivity and to handle communication with the Blynk platform. It connects to a Wi-Fi network and sends sensor data to the Blynk server for remote monitoring.
- ESP8266 allows the Eco BinBot to interact with its environment through sensors and actuaries, process this information, and communicate wirelessly with the network, all of which are essential for the smart functionalities of the bin.
- Code snippet to establish the connection

```
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// WiFi credentials
char ssid[] = "pass";
char pass[] = "pass1124";

// Initialize Blynk
Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);

(see appendix C for more)
```

User Interface

- Blynk application provides the user an interface to monitor the real time data detected by the sensors via a remote device connected to a Wi-Fi network. It facilitates methods to display sensor data in forms of charts, graphs and meters, so that the user can easily grasp the bin state and make decisions whether to empty the garbage or make predictions.
- Code Snippets:

```
#include <BlynkSimpleEsp8266.h>
```

```
Blynk.virtualWrite(V1, distance); // Send the fill level to Blynk Virtual Pin V1  
Blynk.virtualWrite(V2, alchlLevel); // Send the gas level to Blynk Virtual Pin V2  
Blynk.virtualWrite(V3, batteryVoltage); // Send the battery voltage to Blynk  
Virtual Pin V3  
(see appendix C for more)
```

Software Development tools

- Arduino IDE: To write, compile and upload codes to ESP8266 microcontroller.
- Blynk Platform: Facilitate remote monitoring of the bin using a cloud service.
- **Libraries Used:**
 - ESP8266WiFi library – Connects ESP to Wi-Fi
 - BlynkSimpleEsp8266 library – Connects ESP8266 to Blynk Cloud
 - Servo library – Controls servo motor movements
- Programming Languages
 - C ++ : To program ESP8266 on Arduino IDE.

2.4 Testing

Table 2.1 : Automatic Lid Opening

Test Scenario ID	Lid_opening	Test Case ID	Lid_opening-1
Test Case Description	Use of PIR sensor to detect motion only within a specified range (40cm)	Test Priority	High
Pre-Requisite	Program the ESP module to display true when something is detected	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Move a hand in front of the sensor, 30cm afar.	Motion at 30cm distance	Display true on the serial monitor	Display true on the serial monitor	Pass	Detection successful
2	Move a hand in front of the sensor, 70cm afar	Motion at 70cm distance	Display false on the serial monitor	Display true on the serial monitor	Fail	Doesn't meet the requirements. Try using a Ultrasonic sensor

Test Scenario ID	Lid_opening	Test Case ID	Lid_opening-2
Test Case Description	Use of Ultrasonic sensor to detect a surface only within a specified range (40cm)	Test Priority	High
Pre-Requisite	Program the ESP module to display the distance to a detected object	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Keep a hand in front of the sensor, 30cm afar	Time Duration taken for the echo pulse to return	30cm	30cm	Pass	Successfully detected.
2	Keep a hand in front of the sensor, 70cm afar	Time Duration taken for the echo pulse to return	Out of range	Out of range	Pass	Successful Execution.

Test Scenario ID	Lid_opening	Test Case ID	Lid_opening3
Test Case Description	Servo motor movement according to sensor inputs	Test Priority	High
Pre-Requisite	Ultrasonic sensor, ESP8266 and Servo motor connected to each other	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Keep a hand in front of the sensor, 30cm away from the bin	Ultrasonic pulse	Bin lid position 140	Bin lid position 140	Pass	Successfully opened
2	Move further away from the sensor	No return wave	Bin lid position 0	Bin lid position 0	Pass	Lid is successfully closed

Table 2.2 : Fill level Monitoring

Test Scenario ID	Fill_level	Test Case ID	Fill_level-1
Test Case Description	Get the fill level inside the garbage bin using Ultrasonic sensor	Test Priority	High
Pre-Requisite	Program the ESP module to calculate the fill level using bin height and distance to the garbage surface	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Put 10 cm tall box inside the bin	Time Duration taken for the echo pulse to return	70 – 60 = 10cm	10cm	Pass	Accurate distance measures obtained

Test Scenario ID	Fill_level	Test Case ID	Fill_level-2
Test Case Description	Check for Blynk warning messages when the specified limit (70cm) is reached or exceeded	Test Priority	High
Pre-Requisite	Program the ESP module to make the Blynk App generate an alert when the distance between sensor and garbage surface is less than or equal to 10cm Sensor, ESP and the blynk cloud should be connected via Wi-Fi	Post-Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Fill the bin up to the bin opening	Time Duration taken for the echo pulse to return	Warning: Empty the bin ASAP!!	Warning: Empty the bin ASAP!!	Pass	Successfully alerts the user.

Table 2.3 : Gas level detection

Test Scenario ID	Gas_Sensor	Test Case ID	Gas_sensor1
Test Case Description	Display whether smoke is detected or not	Test Priority	High
Pre-Requisite	Program the ESP to compare sensor value to a predefined threshold value	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Turn on a lighter near the sensor	CO2 gas	“Smoke Detected” on the serial monitor	“Smoke Detected” on the serial monitor	Pass	Successfully alerts the user.
2	Keep the sensor in an environment without any harmful gases	No harmful gases	Serial monitor: “Normal”	Serial monitor: “Normal”	Pass	

Test Scenario ID	Gas_Sensor	Test Case ID	Gas_Sensor2
Test Case Description	Check for Blynk warning messages when the specified smoke level is reached or exceeded	Test Priority	High
Pre-Requisite	Program the ESP module to make the Blynk App generate a alert when the sensor value exceeds predefined limit. Sensor, ESP and the blynk cloud should be connected via Wi-Fi	Post-Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Turn on a lighter near the sensor	CO2 gas	Alert: Smoke Detected!!	Alert: Smoke Detected!!	Pass	Successfully alerts the user.
2	Changing Batteries with differently charged batteries to check the accuracy of the data communication	Voltage level	Pre Calculated Voltage Level	Correct value displayed in the blynk app	Pass	Successfully displays the voltage level
3	Sitting in front of the Sonar sensor to check whether the servo motor is triggered	Distance between subject and the dustbin	Opening the door when the body is in, or distance is less than the predefined distance	Door opens	pass	Sensor and the servo motor is working and communicating correctly

Table 2.4 : User Interface

Test Scenario ID	User_interface	Test Case ID	UI_1
Test Case Description	Check if correct data is displayed on the Blynk app	Test Priority	High
Pre-Requisite	Sensors, ESP and the blynk cloud should be connected via Wi-Fi	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Fill the bin up to 45 cm	Send the fill level to Blynk Virtual Pin V1	45cm in the blynk V1 meter	45cm in the blynk V1 meter	Pass	Accurate sensor data depiction
2	Turn up a lighter	Send the gas level to Blynk Virtual Pin V2	Display 100 and a warning message	Display 100 and a warning message	Pass	Accurate sensor data depiction
3	Charge upto 12v	Send the battery voltage to Blynk Virtual Pin V3	Display 12v on V3	Display 12v on V3	Pass	Accurate voltage level depiction

Table 2.5 : ESP module

Test Scenario ID	ESP	Test Case ID	ESP_1
Test Case Description	Verify Wi-Fi connectivity and Blynk platform connection	Test Priority	High
Pre-Requisite	ESP8266 configured with Wi-Fi credentials	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Power on ESP8266	NA	Successfully connected to Wi-Fi	Successfully connected to Wi-Fi	Pass	
2	Power on ESP8266		Successfully connected to Blynk server	Successfully connected to Blynk server	Pass	

Table 2.6 : Charging Unit

Test Scenario ID	Battery_level	Test Case ID	Battery1
Test Case Description	Check the battery level on Blynk app when the charging unit is charged	Test Priority	Medium
Pre-Requisite	Power unit connected and Blynk is configured to monitor the battery level	Post- Requisite	NA

Test Execution Steps:						
S.No	Action	Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Charge the battery using solar power	Solar panel	12v	0v	Fail	Battery is not charged. Circuit is damaged.
2	Charge the battery using a normal battery	Battery	12v	12v	Pass	The charging unit is successfully charged.

3. Conclusion

3.1 Assessment of the Project Results

- The Eco BinBot project successfully achieved its primary objectives, demonstrating significant potential in enhancing waste management practices. The system effectively monitored the fill level and toxicity of waste, automatically controlled the bin lid, and utilized solar power, aligning with the eco-friendly goal. Despite some initial technical challenges, such as integrating the weight sensor and PIR sensor, the project culminated in a functional prototype that meets user needs. The Blynk platform proved to be a reliable choice for real-time monitoring and notifications, simplifying user interaction and ensuring timely alerts.

3.2 Lessons Learned

- Several key lessons were learned throughout the development of the Eco BinBot:
 - **Importance of Robust Sensor Integration:** Initial attempts with the PIR sensor and weight sensor highlighted the necessity of selecting and calibrating sensors that align precisely with project requirements.
 - **Power Source Reliability:** The transition from solar power to a normal battery underscored the importance of having backup power solutions and thoroughly testing power components under various conditions.
 - **User-Centric Design:** Feedback from potential users emphasized the need for clear, real-time alerts and a user-friendly interface, which was successfully implemented using the Blynk platform.

3.3 Future Work

- To build on the current achievements, future work will focus on:
 - **Enhanced Gas Detection:** Expanding the range of detectable gases to provide a more comprehensive assessment of bin contents.
 - **Incorporating a Garbage Compactor:** Integrating a compactor to increase the bin's capacity and reduce the frequency of emptying.
 - **Improving Solar Power Efficiency:** Optimizing the solar power setup to ensure reliable operation even in low light conditions.
 - **GPS Tracking and Collaboration:** Adding GPS tracking to the smart bin to facilitate coordination between homeowners, garbage collectors, and authorities, potentially improving waste collection efficiency.

4. References

- [1] Gov.in. [Online]. Available:
https://www.nielit.gov.in/gorakhpur/sites/default/files/Gorakhpur/alevel_iot_13april20_S M.pdf[Accessed: 20-Apr-2024].
- [2] e19-3yp-Smart-Waste-Management-System: Smart Waste Management System that aims to make the process of waste collection more efficient. The system utilizes waste bins and manages them through mobile and web applications. With the help of the system, users can easily check the status of the bins and follow a schedule for waste collection based on the bin's status.
- [3] Arduino, "Arduino trash Bin with waste level detection," 12-Feb-2023. [Online]. Available:
<https://www.youtube.com/watch?v=ZP0wLe3jIXk>. [Accessed: 20-Apr-2024].
- [4] MYBOTIC, "DIY Smart Contactless IOT Dustbin With Notification using ESP32 and Blynk 2.0," 21-Oct-2021. [Online]. Available:
<https://www.youtube.com/watch?v=aIUDZkVQ0YE>[Accessed: 20-Apr-2024].
- [5] Wheelie Bin Solutions, "What is a smart bin and how does it work?," Wheelie Bin Solutions. [Online]. Available: <https://wheeliebinsolutions.co.uk/blogs/advice/what-is-a-smart-bin-and-how-does-it-work> [Accessed: 20-Apr-2024].
- [6] mybotic, "How to build a ESP8266 Web Server," Instructables, 29-Jul-2016. [Online]. Available: <https://www.instructables.com/How-to-Build-a-ESP8266-Web-Server/>.
[Accessed: 20Apr-2024]

Appendix A: Design Diagrams

Software Requirements
Arduino IDE
Libraries for interfacing with sensors (e.g., Ultrasonic sensor library, PIR sensor library)
Development environment for designing user interfaces (e.g., Android Studio for mobile app development, HTML/CSS/JavaScript for web dashboard).
Documentation tools for project management and reporting (e.g., Microsoft Word, Google Docs).

Figure 0.1 : *Software requirements for the smart Eco BinBot*

Hardware Requirements
ESP8266 microcontroller
Ultrasonic sensor for garbage level detection
PIR sensor for human presence detection, Gas sensor for detecting toxicity levels.
Battery or solar cell for powering the smart bin system

Figure 0.2 : *Hardware requirements for the smart Eco BinBot*

Appendix B: Selected Code Listings

Automatic Lid Opening

Set up the ultrasonic sensor pins

```
pinMode(trigPin, OUTPUT); // Set the trigPin as an OUTPUT
pinMode(echoPin, INPUT); // Set the echoPin as an INPUT
```

Measure distance using the ultrasonic sensor

```
long duration;
int distance;
```

Read the echoPin, return the sound wave travel time in microseconds

```
duration = pulseIn(echoPin, HIGH);
```

Calculate the distance

```
distance = duration * 0.034 / 2; // Speed of sound wave divided by 2 (round trip)
```

For Fill level Monitoring- Display the fill level in Blynk cloud interface

```
double distance = getSonarDistance();
Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");
```

```
Blynk.virtualWrite(V1, distance);
```

For automated bin lid opening- activate the servo motor depending on the distance measurements

```
if (distance <= distanceThreshold) {
    // If an object is within the threshold, open the dustbin door (90 degrees)
    moveServo(servoOpenPosition);
} else {
    // If no object is within the threshold, close the dustbin door (0 degrees)
    moveServo(servoClosedPosition);
}
```


Fill level monitoring

```
// Timer for Blynk
BlynkTimer timer;

void loop() {
  Blynk.run();
  timer.run();
}

double getSonarDistance() {
  // Clears the TRIG_PIN
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);

  // Sets the TRIG_PIN on HIGH state for 10 microseconds
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  // Reads the ECHO_PIN, returns the sound wave travel time in microseconds
  duration = pulseIn(ECHO_PIN, HIGH);

  // Calculating the distance
  distanceCm = duration * SOUND_VELOCITY / 2;

  return distanceCm;
}

void sendSensorData() {
  double distance = getSonarDistance();
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");

  Blynk.virtualWrite(V1, distance);
}
```

Gas detection sensor

```
#define Threshold 400
#define MQ2pin A0

// Set a timer to send sensor data every second
timer.setInterval(1000L, sendSensorData);
}

void loop() {
  Blynk.run();
  timer.run();
}

void sendSensorData() {
  double alchlLevel = getAlcoholLevel();
  Serial.println(alchlLevel);
  Blynk.virtualWrite(V2, alchlLevel);
}

double getAlcoholLevel() {
  double sensorValue;
  sensorValue = analogRead(MQ2pin); // read analog input pin A0

  Serial.print("Sensor Value: ");
  Serial.print(sensorValue);

  // Determine the status
  if (sensorValue > Threshold) {
    Serial.println(" | Smoke detected!");
  } else {
    Serial.println(" | Normal!");
  }
  return sensorValue;
}
```

Battery level monitoring

```
// Define the analog pin
const int batteryPin = A0;

// Define resistor values (in ohms)
const float R1 = 20000.0; // 20k ohm resistor
const float R2 = 6600.0; // 6k ohm resistor

void loop() {
  // Read the battery voltage
  int sensorValue = analogRead(batteryPin);

  // Convert the analog reading to voltage
  float voltage = sensorValue * (3.3 / 1023.0); // ESP8266 ADC range is 0-3.3V

  // Calculate the actual battery voltage
  float batteryVoltage = voltage * ((R1 + R2) / R2);

  // Print the battery voltage to the serial monitor
  Serial.print("Battery Voltage: ");
  Serial.println(batteryVoltage);

  // Send the battery voltage to Blynk Virtual Pin V3
  Blynk.virtualWrite(V3, batteryVoltage);
}
```

Blynk initialization

```
#define BLYNK_TEMPLATE_ID "TMPL6ajCvEf5Y"  
#define BLYNK_TEMPLATE_NAME "Dust bin"  
#define BLYNK_AUTH_TOKEN "DfYbYhXEV3WddioYkrQYftavtXdKec50"
```

```
// Initialize Blynk  
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);  
// Run Blynk  
  Blynk.run();
```

Include libraries

```
#include <ESP8266WiFi.h>  
#include <BlynkSimpleEsp8266.h>  
#include <Servo.h>
```