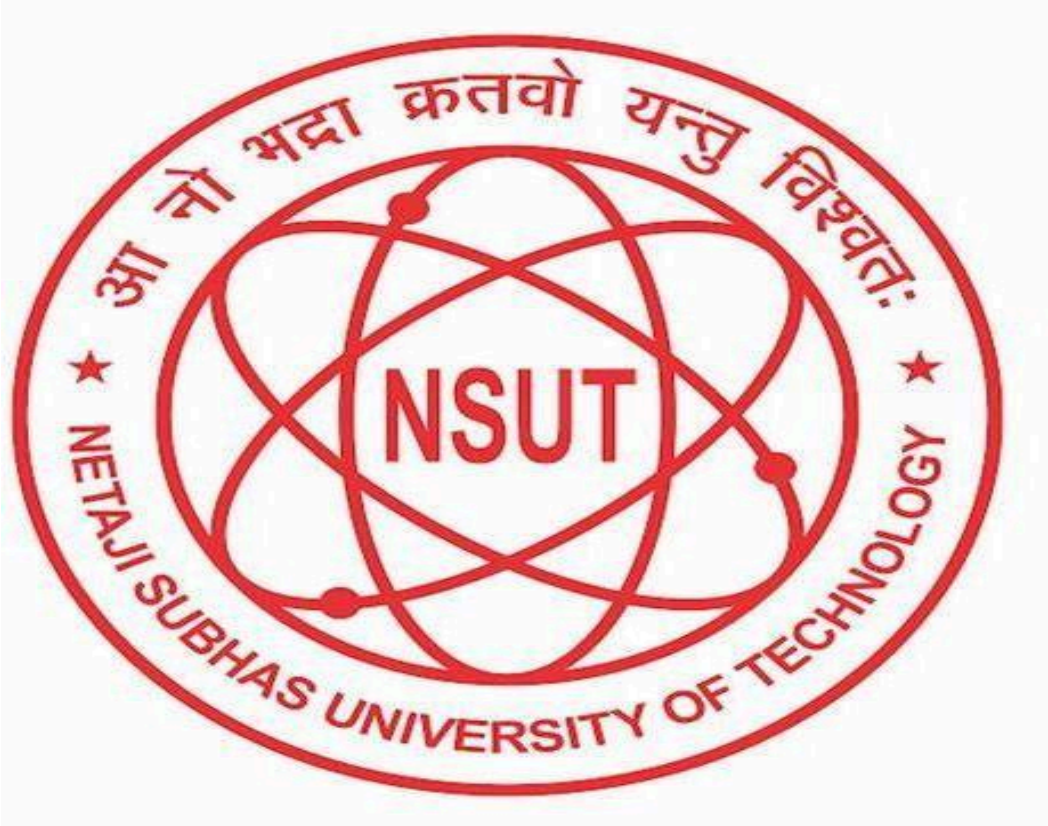
NETAJI SUBHASH UNIVERSITY OF TECHNOLOGY



Project Report

Smart Waste Bin

Submitted to:-
Prof. Dhananjay V. Gadre

Submitted by:-
Atul Kumar Prajapati(2021UEI2807)

INDEX

PROJECT DOCUMENTATION

ABSTRACT

The project focuses on the development and implementation of a smart waste bin system using ESP8266 for waste segregation. The system is designed to autonomously segregate waste into three distinct categories: metal, wet, and dry. The integration of ESP8266 allows for real-time monitoring and control of the waste segregation process, enhancing the efficiency and accuracy of waste management practices.

The smart waste bin system utilizes various sensors, including inductive proximity sensors for detecting metal waste, IR sensors for identifying wet waste, and ultrasonic sensors for measuring the level of waste in each category. These sensors work in tandem to accurately determine the type and quantity of waste being disposed of, enabling the system to direct the waste to its respective container.

The project's implementation involves the integration of hardware components such as sensors, microcontrollers, and communication modules, along with software development for data processing and control. The ESP8266 module facilitates communication between the smart waste bin system and a central server, enabling remote monitoring and management of waste disposal activities.

Through the implementation of this project, we aim to demonstrate the feasibility and effectiveness of using IoT technologies for smart waste management. By automating the waste segregation process, the system aims to reduce manual labor, minimize waste contamination, and promote sustainable waste disposal practices.

Overall, the smart waste bin system represents a significant advancement in waste management technology, offering a scalable and efficient solution for improving waste segregation and disposal practices in various settings.

1 Introduction

1.1 Description

The "Smart Waste Bin" project aims to revolutionize traditional waste management practices by implementing an intelligent system capable of segregating waste into three distinct categories: metal, wet, and dry. The system utilizes a sophisticated arrangement of sensors and a rotating plate mechanism to accurately detect and sort different types of waste, ensuring efficient and eco-friendly disposal. The three containers placed below this plate then rotates depending on which type of waste is coming. The waste then goes to that dustbin container.

The heart of the system is a passage equipped with multiple sensors that analyze the characteristics of the waste passing through. These sensors work in unison to determine the type of waste, distinguishing between metal, wet, and dry materials. Once the waste type is identified, a rotating plate located at the bottom of the passage opens to allow the waste to pass through, directing it to the corresponding container.

The project's design incorporates the principles of automation and IoT (Internet of Things) technology, allowing for seamless operation and real-time monitoring. The use of sensors and intelligent control mechanisms minimizes human intervention, reducing the risk of errors and ensuring efficient waste segregation.

1.2 Motivation

The motivation behind the "Smart Waste Bin" project stems from the growing need for sustainable waste management solutions. Traditional waste disposal methods often lead to environmental pollution and resource wastage. **By implementing an automated waste segregation system, the project aims to promote recycling and reduce the amount of waste sent to landfills**.

Furthermore, the project seeks to address the challenges faced by waste management authorities in sorting and processing large volumes of waste. By streamlining the waste segregation process, the system not only enhances operational efficiency but also contributes to the conservation of natural resources and the preservation of the environment.

1.3 Applications

The "Smart Waste Bin" project has wide-ranging applications in various settings, including residential, commercial, and industrial sectors. Some potential applications of the project include:

1. **Residential Complexes**: Implementing the system in residential buildings can help residents segregate their waste more effectively, promoting recycling and reducing environmental impact.

2. **Commercial Establishments**: Restaurants, hotels, and other commercial establishments can use the system to manage their waste more efficiently, ensuring proper segregation and disposal of waste materials.

3. **Public Areas:** Installing the system in public areas such as parks, malls, and educational institutions can help maintain cleanliness and promote responsible waste disposal practices among the public.

4. **Industrial Facilities**: Industrial plants and manufacturing units can benefit from the system's ability to segregate different types of waste generated during production processes, facilitating recycling and waste management.

Overall, the "Smart Waste Bin" project offers a practical and sustainable solution to the challenges posed by traditional waste management practices, paving the way for a cleaner and greener future.

1.4 Objectives

1. **Efficient Waste Segregation**: Develop a system that can accurately segregate waste into metal, wet, and dry categories to facilitate recycling and proper disposal.

2. **Automated Operation**: Implement sensors and a rotating plate mechanism to automate the waste segregation process, minimizing the need for manual intervention.

3. **Real-time Monitoring**: Enable real-time monitoring of waste disposal activities and system performance to ensure effective operation and
maintenance.

4. **Resource Optimization**: Optimize the use of resources such as energy and space by implementing an efficient waste segregation system.

5. **Environmental Impact**: Reduce the environmental impact of waste disposal by promoting recycling and responsible waste management
practices.

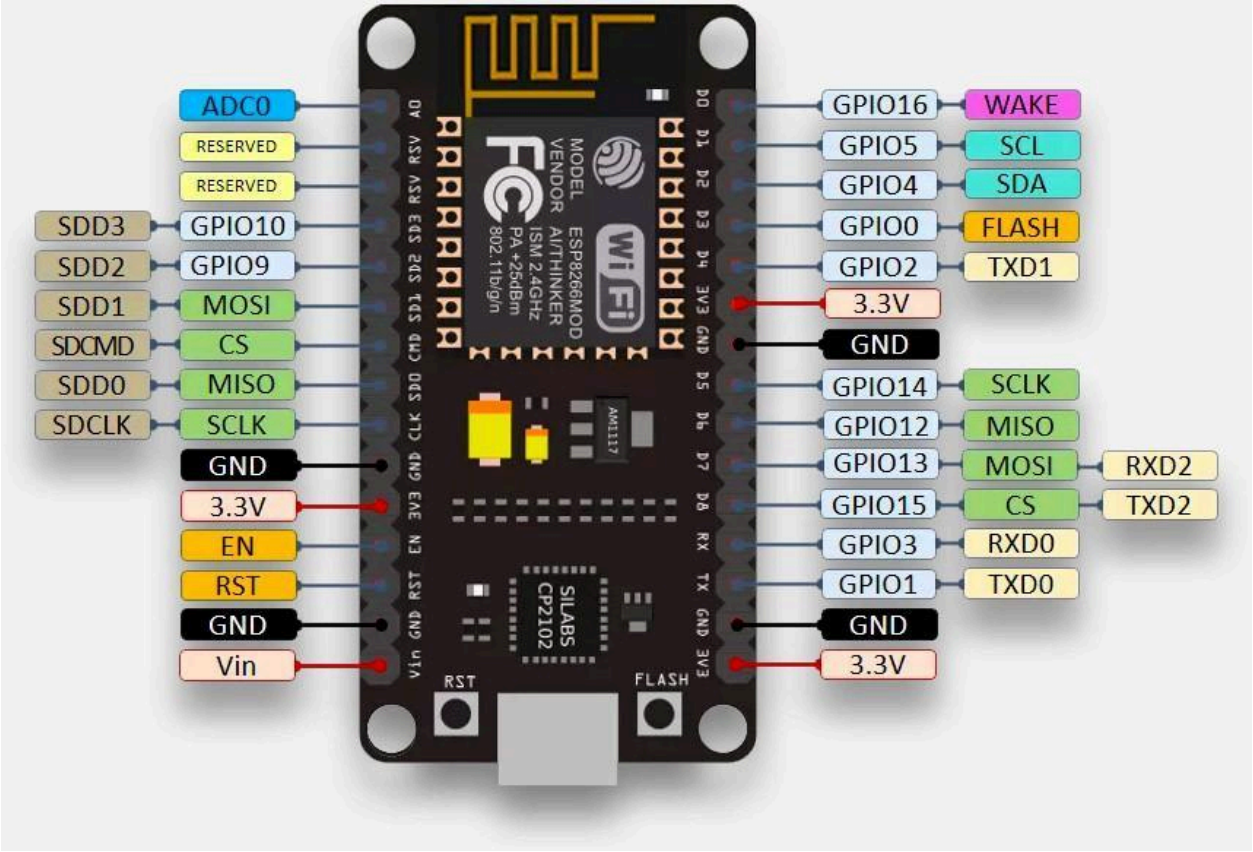
6. **User-friendly Interface**: Provide a user-friendly interface for users to interact with the system and monitor waste segregation activities.

2 Components and Description

S.No.	Components	Price per Piece
1	ESP 8266	250
2	Plastic Containers(3)	120
3	Stepper Motor	75
4	Moisture Sensor	100
5	Servo Motor	70
6	Inductive Proximity Sensor	150

7	IR Sensor	70
8	Ultrasonic Sensor	70
9	Magnetic Sheet	100
10	MDF Board	30

2.1 ESP8266



The ESP8266 module plays a crucial role in the smart waste bin project, serving as the central control unit responsible for coordinating the operation of sensors and actuators involved in waste segregation. Here's a detailed overview of how the ESP8266 module is utilized in this project:

1. **Wireless Communication**: The ESP8266 module enables wireless communication between the smart waste bin system and external devices, such as a central server or a user interface. This allows for remote monitoring and control of the waste segregation process.

2. **Sensor Integration**: The ESP8266 module interfaces with various sensors, including inductive proximity sensors, IR sensors, and ultrasonic sensors, to detect the type of waste passing through the system. The module processes sensor data to determine the appropriate action for segregating the waste.

3. **Actuator Control**: Based on the sensor data, the ESP8266 module controls actuators such as motors to open or close the rotating plate at the bottom of the waste passage. This mechanism directs the waste to the respective containers based on its type (metal, wet, dry).

4. **Data Processing**: The ESP8266 module processes sensor data and sends relevant information, such as waste type and quantity, to a central server for real-time monitoring and analysis. This data can be used to optimize waste management practices and improve overall efficiency.

5. **Energy Efficiency**: The ESP8266 module is designed to operate efficiently, minimizing power consumption while ensuring reliable performance. This is essential for long-term deployment of the smart waste bin system in various environments.

6. **Scalability and Flexibility**: The ESP8266 module's firmware can be easily updated to accommodate changes in sensor configurations or system requirements. This scalability and flexibility make it suitable for use in a wide range of waste management applications.

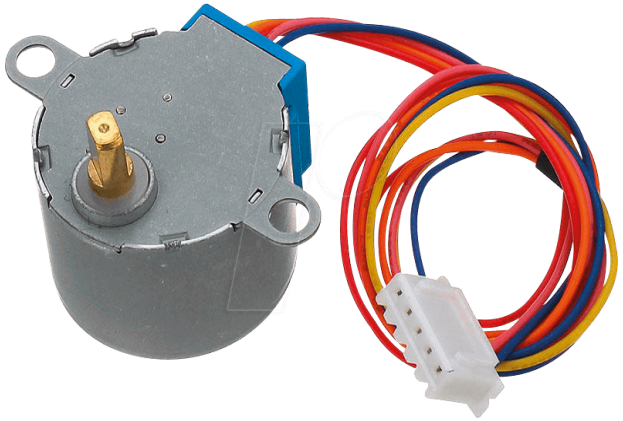
Overall, the ESP8266 module serves as the brain of the smart waste bin system, enabling efficient waste segregation and management

through its wireless communication, sensor integration, and actuator control capabilities. Its reliability, efficiency, and versatility make it an ideal choice for implementing IoT solutions in waste management.

2.2 Plastic Containers

We have used 3 plastic containers to hold 3 types of wastes i.e. metal waste, dry waste and wet waste. The container for dry waste is placed initially below the passage(passage consists of different sensors to detect different types of waste. First of all we detect metal waste by detecting it using Inductive Proximity Sensor. If the given waste detected is metal, then we don't check it further. If waste is not detected by Inductive Proximity Sensor then we check if the waste is wet or not using a moisture sensor. After detecting the waste it is passed to one of the plastic containers).
If the waste is dry. Then we do not rotate the containers as the dry waste container is just below the dry waste. If the waste is metal, we rotate the dry container anticlockwise to get the metal container below the passage and the metal waste is put in the metal container. The wet waste container is placed clockwise with respect to the dry container which will hold wet waste.

2.3 Stepper Motor



The stepper motor used in the smart waste bin project serves a critical function in rotating the containers to hold the desired types of waste (metal, wet, dry). Here's an overview of the stepper motor's role and its operation in the project:

1. **Positioning Control**: The stepper motor is responsible for precisely rotating the containers to position them under the waste passage. This ensures that the waste is deposited into the correct container based on its type.

2. **Rotation Angle**: The stepper motor's rotation angle is carefully calibrated to align with the positions of the containers. This allows for accurate and reliable waste segregation without spillage or misalignment.
3. **Actuation Mechanism**: The stepper motor is connected to a rotating plate mechanism that supports the containers. When activated, the stepper motor rotates the plate, causing the containers to move into position under the waste passage.

4. **Integration with ESP8266**: The stepper motor is controlled by the ESP8266 module, which sends signals to the motor to initiate rotation when waste is detected. This integration ensures seamless operation and synchronization with other components of the system.

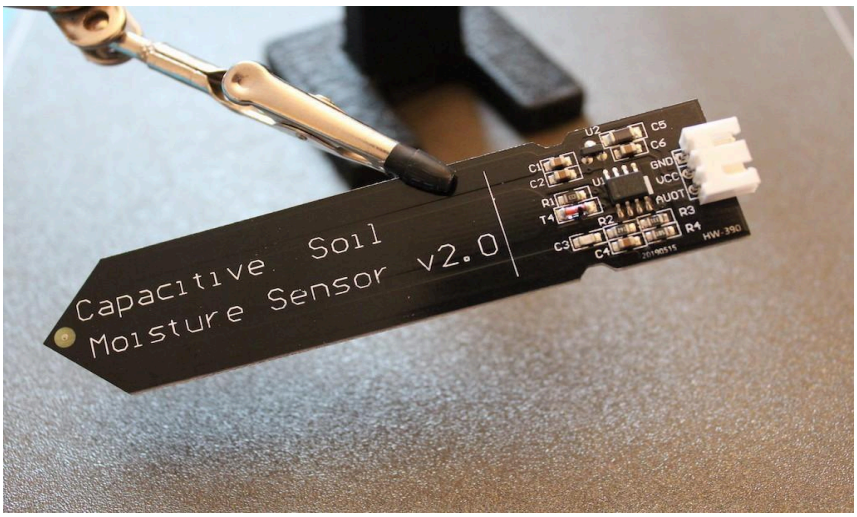
5. **Efficiency and Reliability**: The stepper motor's design ensures efficient operation with minimal power consumption. Its reliability and precision make it suitable for continuous use in waste management applications.

6. **Safety Features**: The stepper motor is equipped with safety features to prevent overloading or overheating. This ensures the motor's longevity and protects it from damage during operation.

Overall, the stepper motor plays a crucial role in the smart waste bin project, enabling precise and reliable rotation of containers for efficient

waste segregation. Its integration with the ESP8266 module ensures smooth operation and enhances the overall functionality of the waste management system.

2.4 Moisture Sensor



The moisture sensor used in the smart waste bin project plays a key role in detecting the presence of wet waste and ensuring that it is segregated correctly. Here's an overview of the moisture sensor's function and operation in the project:

1. **Detection of Wet Waste**: The moisture sensor is designed to detect the moisture content of the waste passing through the system. It can differentiate between wet waste (e.g., food scraps, liquids) and dry waste (e.g., paper, plastic) based on the moisture level.

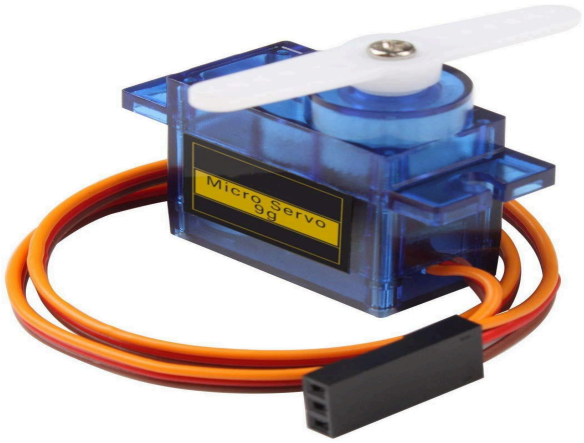
2. **Analog Output**: The moisture sensor provides an analog output signal that varies depending on the moisture content detected. This

signal is processed by the microcontroller (e.g., ESP8266) to determine the type of waste and initiate the segregation process.

3. **Calibration**: The moisture sensor is calibrated to distinguish between different levels of moisture content, ensuring accurate detection of wet waste. This calibration can be adjusted based on the specific requirements of the waste management system.

4. **Integration with ESP8266**: The moisture sensor is integrated into the system along with other sensors, such as the proximity and IR sensors, to provide comprehensive waste detection capabilities. The ESP8266 module processes the sensor data and coordinates the segregation process accordingly.

2.5 Servo Motor



The servo motor used in the smart waste bin project is responsible for controlling the rotation of the plate attached at the bottom of the waste passage. This plate serves to open and close the passage, allowing the waste to be directed to the respective container based on its type (metal, wet, dry). Here's an overview of the servo motor's function and operation in the project:

1. **Plate Rotation**: The servo motor is used to rotate the plate at the bottom of the waste passage. When activated, the servo motor rotates the plate to either open or close the passage, depending on the waste segregation requirements.

2. **Precise Control**: Servo motors are known for their precise control over angular position, making them ideal for applications requiring accurate movement. In the smart waste bin project, the servo motor ensures that the plate is positioned correctly to direct the waste to the desired container.

3. **Integration with Microcontroller**: The servo motor is controlled by the microcontroller (e.g., ESP8266) in the system. The microcontroller sends signals to the servo motor to control its rotation and position, ensuring that the waste is directed accurately.

4. **Durability and Reliability**: Servo motors are designed to be durable and reliable, capable of withstanding continuous operation in a waste management environment. Their robust construction ensures long-term performance with minimal maintenance.

2.6 Inductive Proximity Sensor

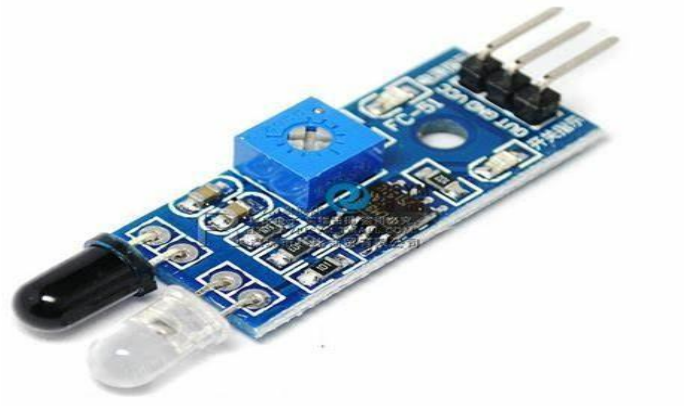
The inductive proximity sensor used in the smart waste bin project is specifically designed to detect metal waste as it passes through the waste passage. Here's an overview of the inductive proximity sensor's function and operation in the project:

1. **Metal Detection**: The inductive proximity sensor is sensitive to metallic objects and can detect the presence of metal waste as it moves through the waste passage. This sensor is crucial for segregating metal waste from other types of waste.

2. **Non-contact Detection**: The inductive proximity sensor operates on the principle of electromagnetic induction, allowing it to detect metal objects without physical contact. This non-contact detection method is ideal for waste segregation applications, as it minimizes wear and tear on the sensor.

3. **Integration with Microcontroller**: The inductive proximity sensor is integrated into the smart waste bin system and is connected to the microcontroller (e.g., ESP8266). The sensor sends signals to the microcontroller when it detects metal waste, triggering the segregation process.

2.7 IR Sensor



The IR sensor used in the smart waste bin project is employed to detect the presence of waste approaching the waste passage. Its primary function is to activate the rest of the sensors (such as the

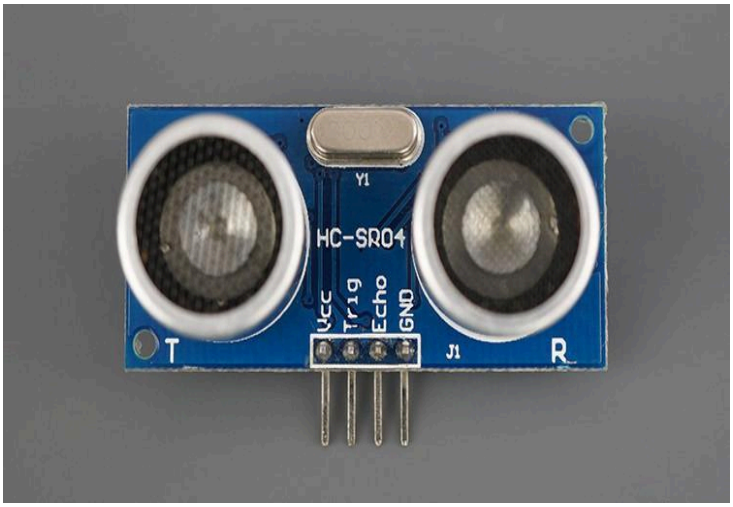
inductive proximity sensor and moisture sensor) only when waste is detected, conserving energy and resources. Here's an overview of the IR sensor's function and operation in the project:

1. **Object Detection**: The IR sensor is sensitive to objects within its detection range and can detect the presence of waste as it approaches the waste passage. This sensor acts as a trigger for activating the other sensors in the system.

2. **Energy Efficiency**: By activating only when waste is detected, the IR sensor helps conserve energy and reduces unnecessary sensor activation. This energy-efficient approach ensures that resources are utilized effectively in the waste management system.

3. **Integration with Microcontroller**: The IR sensor is integrated into the smart waste bin system and is connected to the microcontroller (e.g., ESP8266). When the IR sensor detects waste, it sends a signal to the microcontroller to activate the other sensors for waste segregation.

2.8 ULTRASONIC SENSOR



The ultrasonic sensor used in the smart waste bin project is employed to calculate the distance of waste in the containers placed below it. By measuring the distance, the sensor helps determine the level of waste in each container, enabling efficient waste management. Here's an overview of the ultrasonic sensor's function and operation in the project:

1. **Distance Measurement**: The ultrasonic sensor uses ultrasonic waves to measure the distance between the sensor and the waste in the containers below. By calculating the time taken for the ultrasonic waves to bounce back from the waste, the sensor can determine the distance and, consequently, the level of waste in the container.

2. **Real-time Monitoring**: The ultrasonic sensor provides real-time data on the level of waste in each container, allowing for timely waste collection and disposal. This real-time monitoring helps optimize waste management practices and reduce the risk of overflow or underutilization of containers.

3. **Integration with Microcontroller**: The ultrasonic sensor is integrated into the smart waste bin system and is connected to the microcontroller (e.g., ESP8266). The sensor sends distance measurements to the microcontroller, which processes the data and displays it on the user interface.

3 WORKING

- First of all IR sensors will detect if there is any object or not. If it does not detect anything, nothing will be activated in the system and it will further wait for any waste.
- After detecting it, the object is tested on which type of waste it is by a series of sensors consisting of an inductive proximity sensor and a moisture sensor.

- The waste detected will then be placed in one of the plastic containers placed below this passage.
- The plastic containers will involve a rotation mechanism to hold different types of wastes. There will be no rotation involved when dry waste is detected as we will place dry waste containers everytime below the passage.
- If the metal waste is detected, then we will rotate the containers in such a way that the container initially placed will be rotated anticlockwise to place the metal container below it. It will then hold metal waste. We again rotate the containers to place the dry container again at the initial position.
- For holding wet waste, we will rotate clockwise to make wet waste fall in the desired wet waste container.
- It continuously does this work to accumulate the waste in the desired container.
- The data of the level of waste of each container is recorded at the cloud server.
- An alert will be shown there if any of the dustbin gets filled.

4 MAIN CODE

4.1 Arduino Code

```
#include <Servo.h>
#include <Arduino.h>
#if defined(ESP32)
  #include <WiFi.h>
#elif defined(ESP8266)
  #include <ESP8266WiFi.h>
#endif
#include <Firebase_ESP_Client.h>

// Include the ultrasonic sensor library
#include <Ultrasonic.h>
// Insert your network credentials
#define WIFI_SSID "LAVA"
#define WIFI_PASSWORD "123456789"

// Insert Firebase project API Key
#define API_KEY "AIzaSyA4ez6GPL6kuECR1hrhL_9pEPLZwigD8as"

// Insert Authorized Email and Corresponding Password
#define USER_EMAIL "imtiyazuddin959@gmail.com"
#define USER_PASSWORD "Dsc@1234"

// Insert RTDB URL
#define DATABASE_URL "https://distance-measurement-4a0c6-default-rtdb.asia-southeast1.firebasedatabase.app/"
// Define Firebase objects
FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;

// Variable to save USER UID
String uid;

// Variables to save database paths
```

```
// Variable to save USER UID
String uid;

// Variables to save database paths
String databasePath;
String distancePath;

// Dustbin distances
// float distanceDustbin1 = 0;
// float distanceDustbin2 = 0;
// float distanceDustbin3 = 0;

// Ultrasonic sensor pins
#define trigPin 16   //D0
#define echoPin 13   //D7
Ultrasonic ultrasonic(trigPin, echoPin);

// Timer variables (send new readings every three minutes)
unsigned long sendDataPrevMillis = 0;
unsigned long timerDelay = 1000;

#define sensor A0
int  ir =14;
int inductive = 12;
Servo servo;
int motorPin1 = 5;  // Blue   - 28BYJ48 pin 1
int motorPin2 = 4;  // Pink   - 28BYJ48 pin 2
int motorPin3 = 0; // Yellow - 28BYJ48 pin 3
int motorPin4 = 2;
int motorSpeed = 1;

// Initialize WiFi
```

firebaseproject.ino

```
// Initialize WiFi
void initWiFi() {
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to WiFi ..");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print('.');
    delay(1000);
  }
  Serial.println(WiFi.localIP());
  Serial.println();
}

// Write float values to the database
void sendFloat(String path, float value){
  if (Firebase.RTDB.setFloat(&fbdo, path.c_str(), value)){
    Serial.print("Writing value: ");
    Serial.print (value);
    Serial.print(" on the following path: ");
    Serial.println(path);
    Serial.println("PASSED");
    Serial.println("PATH: " + fbdo.dataPath());
    Serial.println("TYPE: " + fbdo.dataType());
  }
  else {
    Serial.println("FAILED");
    Serial.println("REASON: " + fbdo.errorReason());
  }
}

float calculateDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
```

```
float calculateDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  unsigned long duration = pulseIn(echoPin, HIGH);
  while(digitalRead(echoPin) == HIGH);
  // Speed of sound is approximately 343 meters per second (or 0.0343 cm/microsecond)
  float distance = (duration * 0.0343) / 2.0; // Divide by 2 because the sound travels to the object and back

  return distance;
}

void setup() {
  // put your setup code here, to run once:

  Serial.begin(9600);
  pinMode(ir,INPUT);
  pinMode(inductive,INPUT);
  servo.attach(15);
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);
  pinMode(motorPin3, OUTPUT);
  pinMode(motorPin4, OUTPUT);

  initWiFi();

  // Assign the api key (required)
  config.api_key = API_KEY;

```

```
121
122   // Assign the user sign in credentials
123   auth.user.email = USER_EMAIL;
124   auth.user.password = USER_PASSWORD;
125
126   // Assign the RTDB URL (required)
127   config.database_url = DATABASE_URL;
128
129   Firebase.reconnectWiFi(true);
130   fbdo.setResponseSize(4096);
131
132   // Initialize the library with the Firebase authen and config
133   Firebase.begin(&config, &auth);
134
135   // Getting the user UID might take a few seconds
136   Serial.println("Getting User UID");
137   while ((auth.token.uid) == "") {
138     Serial.print('.');
139     delay(1000);
140   }
141   // Print user UID
142   uid = auth.token.uid.c_str();
143   Serial.print("User UID: ");
144   Serial.println(uid);
145
146   // Update database path
147   databasePath = "/UsersData/" + uid;
148
149   // Update database path for sensor readings
150   distancePath = databasePath + "/distance"; // --> UsersData/<user_uid>/distance
151
152 }
```

```
151
152 }
153
154 void clockwise (){
155   // 1
156   digitalWrite(motorPin1, HIGH);
157   digitalWrite(motorPin2, LOW);
158   digitalWrite(motorPin3, LOW);
159   digitalWrite(motorPin4, LOW);
160   delay(motorSpeed);
161   // 2
162   digitalWrite(motorPin1, HIGH);
163   digitalWrite(motorPin2, HIGH);
164   digitalWrite(motorPin3, LOW);
165   digitalWrite(motorPin4, LOW);
166   delay (motorSpeed);
167   // 3
168   digitalWrite(motorPin1, LOW);
169   digitalWrite(motorPin2, HIGH);
170   digitalWrite(motorPin3, LOW);
171   digitalWrite(motorPin4, LOW);
172   delay(motorSpeed);
173   // 4
174   digitalWrite(motorPin1, LOW);
175   digitalWrite(motorPin2, HIGH);
176   digitalWrite(motorPin3, HIGH);
177   digitalWrite(motorPin4, LOW);
178   delay(motorSpeed);
179   // 5
180   digitalWrite(motorPin1, LOW);
181   digitalWrite(motorPin2, LOW);
182   digitalWrite(motorPin3, HIGH);
183   digitalWrite(motorPin4, LOW);
```

```
  delay(motorSpeed);
  // 6
  digitalWrite(motorPin1, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin4, HIGH);
  delay (motorSpeed);
  // 7
  digitalWrite(motorPin1, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin4, HIGH);
  delay(motorSpeed);
  // 8
  digitalWrite(motorPin1, HIGH);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin4, HIGH);
  delay(motorSpeed);
}

////////////////////////////////////////////////////////////////////////////
//set pins to ULN2003 high in sequence from 4 to 1
//delay "motorSpeed" between each pin setting (to determine speed)

void counterclockwise(){
  // 1
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
```

```
  delay(motorSpeed);
  // 2
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay (motorSpeed);
  // 3
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 4
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 5
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 6
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, HIGH);
  delay (motorSpeed);
  // 7
  digitalWrite(motorPin4, LOW);
```

```
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, HIGH);
  delay(motorSpeed);
  // 8
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, HIGH);
  delay(motorSpeed);
}


int irfun(){
  int ir_value=digitalRead(ir);
  return ir_value;
}

int inductivefun(){
  int inductive_value =digitalRead(inductive);
  return inductive_value;
}

int moisture_sensor(){
  int val = analogRead(sensor);
  Serial.println("Analog Output: ");
  Serial.println(val);
  delay(500);
  return val;
}

void finalvalue(){
  // Send new readings to database
```

```
void finalvalue(){
  // Send new readings to database
  if (Firebase.ready() && (millis() - sendDataPrevMillis > timerDelay || sendDataPrevMillis == 0)){
    sendDataPrevMillis = millis();

    // Get latest sensor readings
    float distance = calculateDistance();
    Serial.println(distance);

    // Send reading to database:
    sendFloat(distancePath, distance);
}
void loop() {
  // put your main code here, to run repeatedly:
  servo.write(180);

  int ir = irfun();
  if(ir==0){
      Serial.println("waste aaya");
      int inductive = inductivefun();
      if(inductive==0){
        Serial.println("waste is metal");
        for(int i = 0; i < 171; i++)
          counterclockwise();
        delay(2000); // rotate slowly from 0 degrees to 180 degrees, one by one degree
        servo.write(-180);
        delay(5000); // control servo to go to position in variable 'pos'
        servo.write(180);
        delay(1000);
        for(int n = 0; n < 171; n++) // good
          clockwise();
        delay(2000);
```

```
312         finalvalue();
313         }
314
315
316
317       else
318       {
319        int val=moisture_sensor();
320        Serial.println(val);
321         if(val<900){
322
323           Serial.println("waste is wet");
324
325           for(int n = 0; n < 171; n++) // good
326             clockwise();
327           delay(5000);
328           servo.write(-180);
329           delay(5000);
330           servo.write(180);
331           delay(5000);
332           for(int i = 0; i < 171; i++)
333             counterclockwise();
334           delay(2000);
335
336           finalvalue();
337       }
338         else{
339           Serial.println("waste is dry");
340           delay(5000);
341           servo.write(-180);
342           delay(5000);
343           servo.write(180);
```

```
338         else{
339           Serial.println("waste is dry");
340           delay(5000);
341           servo.write(-180);
342           delay(5000);
343           servo.write(180);
344           delay(1000);
345
346           finalvalue();
347         }
348        }
349
350       }
351     else{
352        Serial.println("khuda daal");
353        delay(1000);
354       }
355
356   }
```

4.2 Web Page Code

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Smart Waste Monitoring System</title>
<style>
  body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background: linear-gradient(135deg, #FFD700 0%, #8B4513 100%);
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: flex-start;
    min-height: 100vh;
  }
  h1 {
    margin-top: 20px;
    margin-bottom: 20px;
    text-align: center;
    color: #fff;
    text-shadow: 2px 2px 2px rgba(0,0,0,0.5);
  }
  .container {
```

```
.container {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  width: 80%;
  margin-top: 20px;
}
.row {
  display: flex;
  justify-content: center;
  margin-bottom: 4cm;
}
.dustbin {
  width: 150px;
  height: 250px;
  background-color: #F0E68C; /* Light goldenrod yellow */
  border-radius: 15px;
  box-shadow: 0px 5px 10px rgba(0, 0, 0, 0.3); /* Adding shadow */
  padding: 20px;
  box-sizing: border-box;
  text-align: center;
  position: relative;
  overflow: hidden; /* Hide overflow for handle */
```

```
.dustbin {
  margin: 10px;
}
.dustbin:before {
  content: '';
  position: absolute;
  top: 0;
  left: 50%;
  transform: translateX(-50%);
  width: 30px;
  height: 10px;
  background-color: #000;
  border-radius: 5px;
}
.lid {
  position: absolute;
  top: -30px;
  left: 0;
  width: 100%;
  height: 30px;
  background-color: #333;
  border-top-left-radius: 15px;
  border-top-right-radius: 15px;
```

```
62    .lid {
71    }
72    .brown-indicator {
73      position: absolute;
74      bottom: 0;
75      left: 0;
76      width: 100%;
77      height: 0;
78      background-color: #8B4513; /* Brown color for indicator */
79      transition: height 0.5s; /* Add smooth transition */
80    }
81    .dustbin h2 {
82      margin-top: 0;
83      color: #fff;
84    }
85  </style>
86  </head>
87  <body>
88  <h1>Smart Waste Monitoring System</h1>
89  <div class="container">
90    <div class="row">
91      <div class="dustbin" id="dustbin1">
92        <div class="lid"></div>
```

```
91      <div class="dustbin" id="dustbin1">
92        <div class="lid"></div>
93        <div class="brown-indicator"></div>
94        <h2>Dustbin 1</h2>
95      </div>
96      <div class="dustbin" id="dustbin2">
97        <div class="lid"></div>
98        <div class="brown-indicator"></div>
99        <h2>Dustbin 2</h2>
100     </div>
101     <div class="dustbin" id="dustbin3">
102       <div class="lid"></div>
103       <div class="brown-indicator"></div>
104       <h2>Dustbin 3</h2>
105     </div>
106   </div>
107 </div>
108
109 <script src="https://www.gstatic.com/firebasejs/9.6.10/firebase-app.js"></script>
110 <script src="https://www.gstatic.com/firebasejs/9.6.10/firebase-database.js"></script>
111 <script>
112   // Initialize Firebase
```

```
<script>
  // Initialize Firebase
  const firebaseConfig = {
    apiKey: "YOUR_API_KEY",
    authDomain: "YOUR_AUTH_DOMAIN",
    databaseURL: "YOUR_DATABASE_URL",
    projectId: "YOUR_PROJECT_ID",
    storageBucket: "YOUR_STORAGE_BUCKET",
    messagingSenderId: "YOUR_MESSAGING_SENDER_ID",
    appId: "YOUR_APP_ID",
    measurementId: "YOUR_MEASUREMENT_ID"
  };

  firebase.initializeApp(firebaseConfig);
  const database = firebase.database();

  // Listen for changes in the database
  database.ref('dustbin1').on('value', (snapshot) => {
    const wasteLevel = snapshot.val();
    updateBrownIndicator('dustbin1', wasteLevel);
  });

  database.ref('dustbin2').on('value', (snapshot) => {
    const wasteLevel = snapshot.val();
```

```
  database.ref('dustbin2').on('value', (snapshot) => {
    const wasteLevel = snapshot.val();
    updateBrownIndicator('dustbin2', wasteLevel);
  });

  database.ref('dustbin3').on('value', (snapshot) => {
    const wasteLevel = snapshot.val();
    updateBrownIndicator('dustbin3', wasteLevel);
  });

  function updateBrownIndicator(dustbinId, wasteLevel) {
    const dustbin = document.getElementById(dustbinId);
    const indicator = dustbin.querySelector('.brown-indicator');
    indicator.style.height = wasteLevel + 'px';
  }
</script>
</body>
</html>
```

5 ZERO PCB FABRICATION

Zero PCB fabrication, also known as Zero Defect Manufacturing (ZDM), aims to produce PCBs with zero defects. While achieving absolute perfection might be challenging, the following steps can help minimize defects in PCB fabrication:

1. **Design Review**: Ensure the PCB design is thoroughly reviewed for errors and manufacturability before fabrication begins.

2. **Material Selection**: Use high-quality materials suitable for the application and ensure they meet the required specifications.

3. **Manufacturing Process Optimization**: Optimize the manufacturing process to minimize errors and defects. This includes optimizing machine settings, ensuring proper handling of materials, and implementing quality control measures.

4. **Quality Control**: Implement rigorous quality control measures throughout the fabrication process. This includes inspecting materials, monitoring process parameters, and conducting tests to ensure the PCBs meet the required standards.

5. **Training and Skill Development**: Ensure that personnel involved in PCB fabrication are properly trained and skilled to perform their tasks effectively.

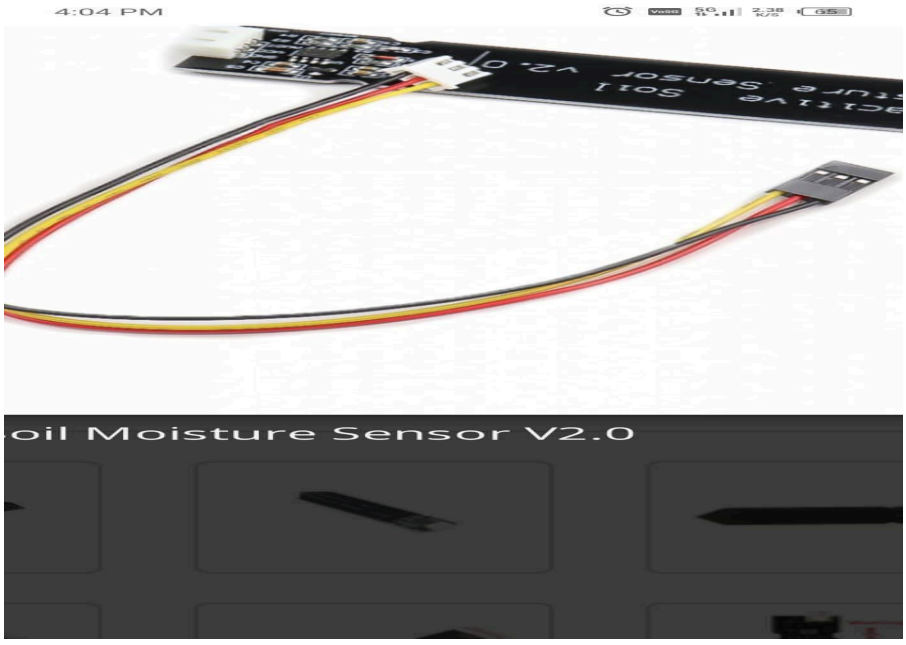
6. **Traceability**: Maintain traceability throughout the fabrication process to identify and address any issues that may arise.

7. **Continuous Improvement**: Continuously monitor and improve the PCB fabrication process to reduce defects and improve overall quality. By following these steps, PCB manufacturers can significantly reduce defects in PCB fabrication and move closer to achieving zero defects.

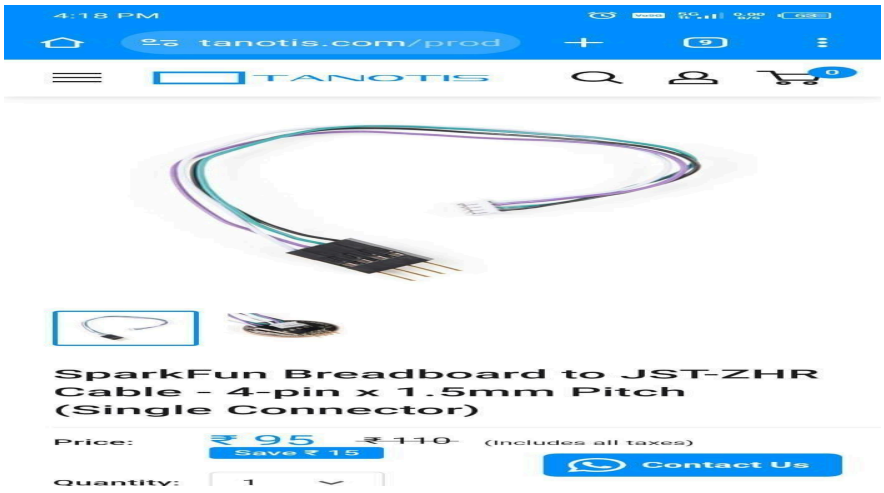
5.1 Connectors

Some of the connectors that we have used in Zero PCB fabrication are as follows:

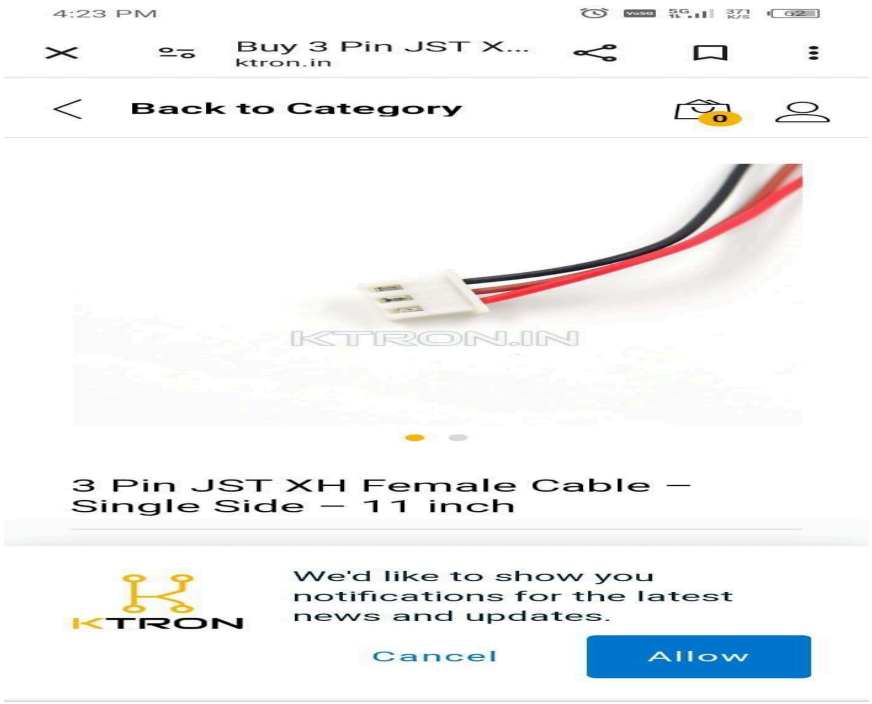
1. **Molex to JST Connector**



2. **Breadboard to JST-ZHR Cable**



3. 3 Pin JST-XH Female Cable



4. 3 Pin JST Connector Male

5. 4 Pin JST Cable Connector Male

6 Conclusion and Future Development

6.1 Conclusion

The smart waste bin project has successfully demonstrated the feasibility and effectiveness of using IoT technology for efficient waste management. By integrating sensors such as the inductive proximity sensor, IR sensor, moisture sensor, and ultrasonic sensor, along with

actuators like the servo motor, the project has achieved precise waste segregation into metal, wet, and dry categories.

The use of the ESP8266 module for wireless communication and control has enabled real-time monitoring and management of waste disposal activities. The project has showcased how automation can streamline waste segregation processes, reduce manual labor, and promote sustainable waste disposal practices.

6.2 Future Developments

1. **Enhanced Sensor Integration**: Further integration of advanced sensors, such as image recognition cameras or chemical sensors, could improve the accuracy and efficiency of waste segregation.

2. **Data Analytics**: Implementing data analytics techniques could provide insights into waste generation patterns, enabling better waste management strategies and resource allocation.

3. **Optimization Algorithms**: Developing optimization algorithms could help optimize waste collection routes and schedules based on real-time data from the sensors.

4. **IoT Platform Integration**: Integrating the smart waste bin system with existing IoT platforms could enable seamless data sharing and integration with other smart city initiatives.

5. **User Interface Improvements**: Enhancing the user interface with more intuitive controls and real-time data visualization could improve user experience and engagement.

6. **Scalability and Adaptability**: Designing the system to be scalable and adaptable to different waste management scenarios and environments could increase its utility and impact.

In conclusion, the smart waste bin project represents a significant step towards sustainable waste management practices. With further development and integration of advanced technologies, it has the potential to revolutionize waste management processes and contribute to a cleaner, greener future.