TRASH KUN - AI POWERED TRASH CAN

A PROJECT REPORT

Submitted by

A. SATYANARAYANA (2116210701235)
K. S. J. SNEHA (2116210701253)

in partial fulfillment for the award of

the degreeof

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING

COLLEGEANNA UNIVERSITY,

CHENNAI

MAY 2024

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Thesis titled **“TRASH KUN – AI POWERED TRASH CAN**” is the bonafide work of “**A. SATYANARAYANA (2116210701235) and K.S.J.SNEHA (2116210701253)”** who carried out the work under my supervision. Certified furtherthat to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred onan earlier occasion on this or any other candidate.

SIGNATURE

Mrs. Anita Ashishdeep B.E, M.Tech

Project Coordinator

Assistant Professor (SG)

Department of Computer Science and

Engineering

Rajalakshmi Engineering College

Chennai - 602 105

Submitted to Project Viva-Voce Examination held on________________

Internal Examiner **External Examine**r

ACKNOWLEDGMENT

First, we thank the almighty god for the successful completion of the project. Our sincere thanks to our chairman **Mr. S. Meganathan B.E., F.I.E.,** for his sincere endeavor in educating us in his premier institution. We would like to express our deepgratitude to our beloved Chairperson **Dr. Thangam Meganathan Ph.D.,** for her enthusiastic motivation which inspired us a lot in completing this project and Vice Chairman **Mr. Abhay Shankar Meganathan B.E., M.S.,** for providing us with the requisite infrastructure.

We also express our sincere gratitude to our college Principal, **Dr. S. N. Murugesan M.E., PhD.,** and **Dr. P. KUMAR M.E., PhD, Director computing and information science , and Head Of Department of Computer Science and Engineering** and our project coordinator **Mrs. Anita Ashishdeep** for her encouragement and guiding us throughout the project towards successful completion of this project and to our parents, friends, all faculty members and supporting staffs for their direct and indirect involvement in successful completion of the project for their encouragement and support.

A. SATYANARAYNA (210701235)

K.S.J. SNEHA (210701253)

ABSTRACT

This project presents an AI-powered smart trash can designed to automate waste segregation, enhancing the efficiency of waste management. The system leverages a combination of hardware and software components to classify and sort waste into organic and inorganic categories. The primary hardware components include an ESP32 microcontroller, a servo motor, an ultrasonic sensor, and a custom-built partitioned trash can. When the ultrasonic sensor detects the presence of an object, the ESP32 captures an image of the waste and sends it to the server for classification. The server processes the image using the trained model to determine whether the waste is organic or inorganic. This classification result is then sent back to the ESP32, which activates the servo motor to direct the waste into the appropriate compartment of the trash can. The project also integrates a MongoDB database to log each classification result, providing valuable data for smart waste management and analysis. The system architecture is designed to ensure real-time processing, accurate classification, and seamless communication between the hardware and software components. This automated waste segregation system aims to reduce the need for manual sorting, improve recycling rates, and contribute to more efficient waste management practices. By combining machine learning with IoT (Internet of Things) technology, the project demonstrates a practical application of AI in addressing environmental challenges and promoting sustainable practices. The modular design allows for scalability and future enhancements, making it a versatile solution for various waste management scenarios.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

Waste management is a critical issue that impacts the environment, public health, and resource sustainability. Traditional waste disposal methods often lead to improper sorting of waste, resulting in contamination of recyclables, increased landfill waste, and inefficient resource recovery. Addressing these challenges requires innovative solutions that leverage modern technologies to automate and improve waste segregation processes.

The AI-powered smart trash can is designed to revolutionize waste management by automating the sorting process using advanced technologies such as ultrasonic sensors, cameras, machine learning models, and IoT (Internet of Things) frameworks. This smart trash can aims to enhance the efficiency and accuracy of waste sorting into organic and inorganic categories, thereby reducing the environmental footprint and promoting sustainable waste management practices.

The system architecture of the smart trash can integrates several key components. An ultrasonic sensor detects the presence of waste and triggers a camera to capture an image. This image is then sent to a web server running a Flask application, where a pre-trained machine learning model processes the image to classify the waste. Based on the classification result, a signal is sent to an ESP32 microcontroller to control a servo motor, which sorts the waste by moving a partition inside the trash can. Additionally, the classification data is logged into a MongoDB database, providing valuable insights for further analysis and smart waste management.

The machine learning model at the core of the system is built using TensorFlow and TensorFlow Hub, leveraging a pre-trained EfficientNet model for image classification. The model is trained on a dataset of various waste categories, enabling it to accurately distinguish between different types of waste. The Flask web server handles image processing requests and interacts with the MongoDB database to log waste data, facilitating real-time data analytics and reporting.

This smart trash can is not only a technological innovation but also a step towards achieving sustainable development goals. By automating the waste sorting process, it reduces human error, improves recycling rates, and minimizes the contamination of recyclables. The integration of IoT and machine learning technologies ensures that the system is both efficient and scalable, making it suitable for deployment in various settings such as households, offices, and public spaces.

In conclusion, the AI-powered smart trash can represents a significant advancement in waste management technology. It combines the power of machine learning with the connectivity of IoT to provide an automated, efficient, and sustainable solution for waste sorting. This innovation holds the potential to transform waste management practices, contributing to a cleaner and more sustainable environment.

CHAPTER 2
LITERATURE SURVEY

The development of an AI-powered smart trash can integrates advancements in various domains, including waste management, machine learning, and IoT technologies. This literature survey explores previous research and innovations that contribute to the foundational knowledge and technical feasibility of this project.

1. *Waste Management and Sorting*

Effective waste management is essential for environmental sustainability and resource conservation. Various studies have focused on improving waste sorting at the source to enhance recycling efficiency.

- Ongondo et al. explored the importance of source separation in waste management systems, emphasizing the need for innovative technologies to improve sorting accuracy and efficiency [1].
- Guerrero et al. reviewed global waste management practices, highlighting the challenges and potential solutions for urban waste management, including the integration of smart technologies [2].

2. *Automated Waste Sorting Technologies*

Automated waste sorting technologies have been the subject of extensive research, focusing on enhancing accuracy and efficiency in waste classification.

- Kauppinen et al. developed an automated waste sorting system using computer vision and machine learning techniques to classify waste into different categories. Their study demonstrated significant improvements in sorting accuracy compared to manual sorting methods [3].
- Zhao et al. investigated the use of convolutional neural networks (CNNs) for

waste classification, showing that deep learning models could accurately identify and sort waste materials [4].

3. *IoT and Smart Waste Management*

The Internet of Things (IoT) has enabled the development of smart waste management systems that provide real-time monitoring and data analytics.

- Al Mamun et al. designed a smart bin system using IoT sensors to monitor waste levels and optimize collection routes. Their study highlighted the potential of IoT to improve waste collection efficiency and reduce operational costs [8].
- Aazam et al. discussed the integration of cloud computing and IoT for smart waste management, emphasizing the benefits of data analytics and predictive maintenance in enhancing system performance [9].

4. *Case Studies and Implementations*

Several case studies and real-world implementations provide insights into the practical applications and challenges of deploying smart waste management systems.

- Longhi et al. implemented a smart bin system in Italy that used RFID technology and sensors to monitor waste disposal. Their findings underscored the importance of user engagement and system reliability for successful implementation [10].
- Folianto et al. developed a smart waste bin with an ultrasonic sensor and GSM module for real-time waste level monitoring. The study demonstrated significant improvements in waste collection efficiency and user satisfaction [11].

5. *Integration of Machine Learning and IoT*

The integration of machine learning with IoT technologies enhances the capabilities of smart waste management systems by enabling real-time data processing and decision-making.

- Belić et al. proposed a smart waste management system that combines machine learning algorithms with IoT sensors to predict waste generation and optimize collection schedules. Their study showed that such systems could significantly reduce waste management costs and environmental impact [12].

2.1 EXISTING SYSTEM

The existing waste management systems largely rely on manual sorting and segregation of waste, which is both labor-intensive and prone to errors. These systems typically involve the collection of mixed waste from households and businesses, followed by transportation to central sorting facilities. At these facilities, workers manually separate recyclable materials from non-recyclable waste. This process is not only time-consuming but also exposes workers to health hazards associated with handling waste.

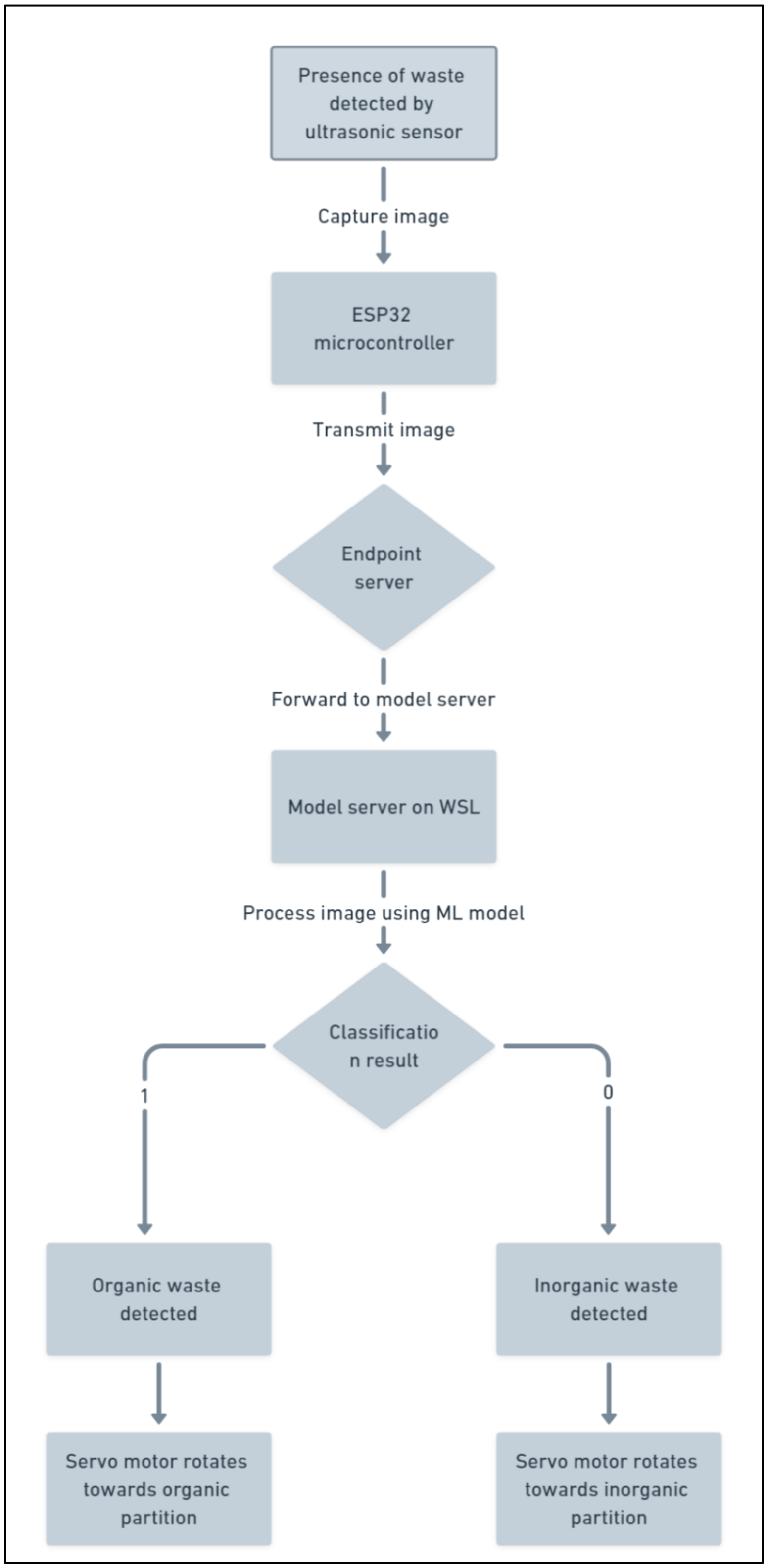
In some urban areas, waste segregation at the source is encouraged, where households are required to separate organic waste from recyclables and non-recyclables. However, compliance rates are often low due to a lack of awareness and inconvenience, leading to inefficiencies in the waste management process. Additionally, the existing systems are limited in their ability to monitor and analyze the type and amount of waste generated, hindering efforts to optimize waste management practices.

There have been some advancements with the introduction of smart bins equipped with sensors and basic automation. These bins can monitor fill levels and alert waste management services when they need to be emptied. While this helps in optimizing the collection schedules, it does not address the core issue of waste segregation.

Overall, the existing systems are characterized by their reliance on manual labor, inefficiencies in sorting processes, and limited use of advanced technologies. These limitations highlight the need for innovative solutions that can automate waste segregation, provide real-time data insights, and enhance the overall efficiency of waste management practices. The proposed AI-powered smart trash can system aims to address these gaps by integrating machine learning, IoT, and automated control mechanisms to revolutionize waste management.

CHAPTER 3

PROJECT DESCRIPTION



This project involves the development of an AI-powered smart trash can designed to automatically segregate waste into organic and inorganic categories. The system integrates both hardware and software components to achieve efficient waste management. The hardware setup includes an ESP32 microcontroller, an ultrasonic sensor, a servo motor, and a partitioned trash can. When the ultrasonic sensor detects waste, it triggers the ESP32 to capture an image of the waste and send it to a server. The server, configured in Windows Subsystem for Linux (WSL), runs a machine learning model that classifies the waste as organic or inorganic. The classification result is then sent back to the ESP32, which activates the servo motor to direct the waste into the appropriate compartment of the trash can. Additionally, a MongoDB database logs each classification result for further analysis and smart waste management insights.

This automated system aims to reduce manual sorting efforts, improve recycling rates, and enhance the overall efficiency of waste management through real-time processing and accurate classification.

3.1. PROPOSED SYSTEM

The proposed AI-powered smart trash can system aims to revolutionize waste segregation through automation and intelligent processing. The system comprises an ESP32 microcontroller, ultrasonic sensor, camera, and servo motor integrated into a partitioned trash can. When the ultrasonic sensor detects waste, the ESP32 captures an image and transmits it to an endpoint server. This server forwards the image to a model server running on the Windows Subsystem for Linux (WSL), where a pre-trained machine learning model classifies the waste as organic or inorganic.

Classification results, represented as "1" for organic and "0" for inorganic, are logged into a MongoDB database for future analytics. The model server sends the classification result back to the endpoint server, which relays it to the ESP32. The ESP32 then activates the servo motor to direct the waste into the appropriate compartment of the trash can based on the classification.

This system not only automates the segregation process but also provides valuable data for smart waste management strategies. By reducing the need for manual sorting and increasing accuracy in waste classification, the system aims to enhance recycling efforts, promote sustainable waste management practices, and contribute to environmental conservation.

3.2 REQUIREMENTS

3.2.1 HARDWARE REQUIREMENTS

The hardware requirements for the system are as follows

S.No	Name	Description
1	ESP32 Microcontroller	Low-cost, low-power SoC with Wi-Fi and Bluetooth, used for connectivity and control.
2	Servo Motor	Rotary actuator for precise angular position control, used to move the partition.
3	Ultrasonic Sensor	Measures distance using ultrasonic waves, detects object presence.
4	Breadboard	Solderless device for constructing electronic circuits, connects components.
5	Jump Wires	Wires for making connections between components on the breadboard.
6	Cardboard Dustbin	Prototype container for the smart trash segregation system.

3.2.2 SOFTWARE REQUIREMENTS

The software requirements and dependencies for the application is mentioned below

S.No	Name	Description
1	Model Training Environment	AMD Ryzen 4600H, GTX 1650, WSL, TensorFlow, for training waste classification model.
2	Model Server Environment	WSL, Flask, TensorFlow, MongoDB, hosts and serves model predictions.
3	Endpoint Server Environment	Windows, Flask, handles requests, triggers image capture, forwards to model server.
4	Arduino IDE	Development environment for ESP32 programming, compiling, and uploading code.

3.3 SYSTEM ARCHITECTURE DIAGRAM

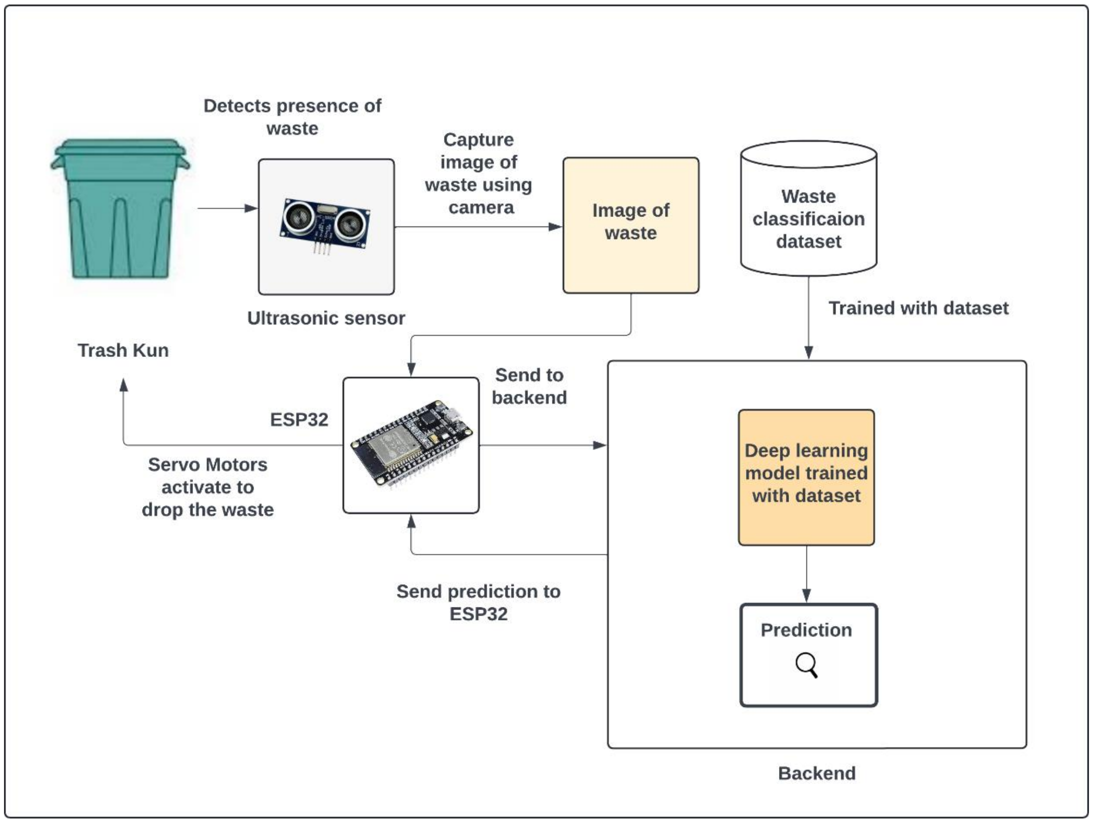
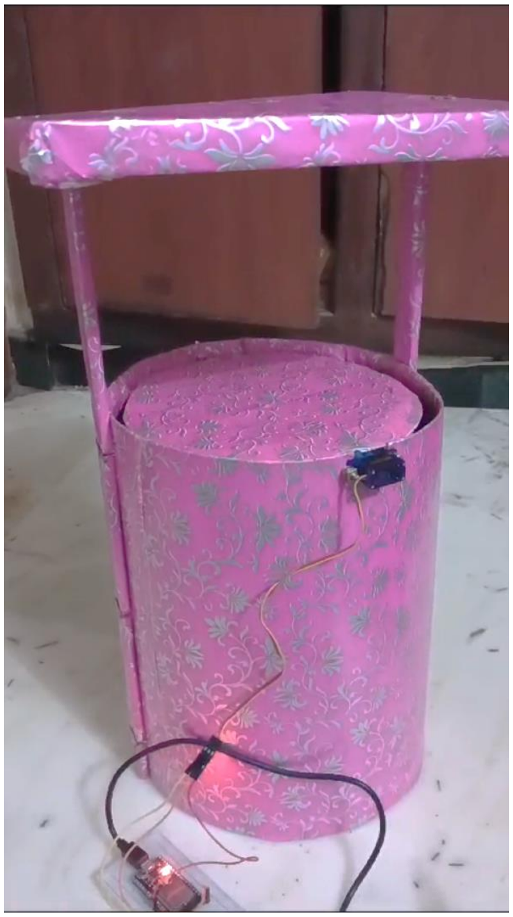
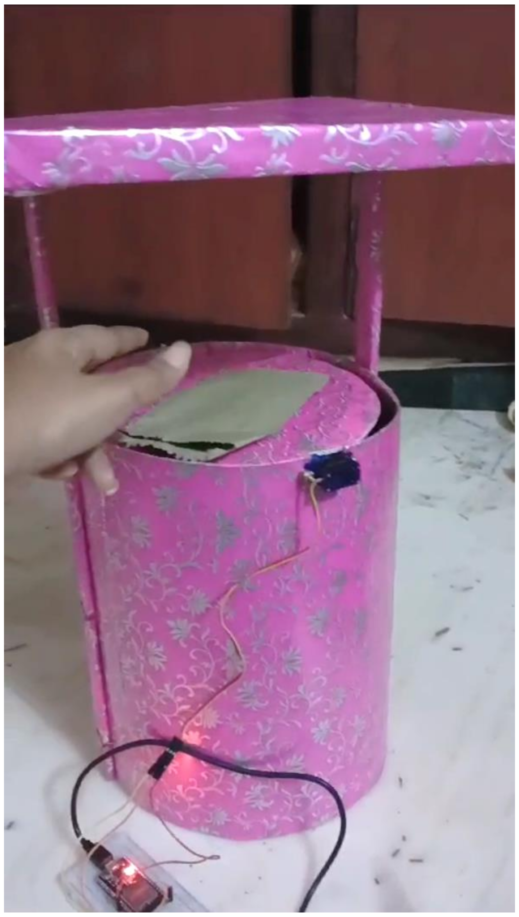

Detects presence of waste
Capture image of waste using camera
Image of waste
Waste classificaion dataset
Ultrasonic sensor
Trained with dataset
Trash Kun
Send to backend
ESP32
Deep learning model trained with dataset
Servo Motors activate to drop the waste
Send prediction to ESP32
Prediction
Backend

Fig. System Architecture Diagram

The system architecture comprises hardware components including an ultrasonic sensor, camera module, and servo motor, interfaced with an ESP32 microcontroller. The ESP32 communicates with software components, including an endpoint server and a model server running on the Windows Subsystem for Linux (WSL). The endpoint server receives and forwards data to the model server for processing using a pre-trained machine learning model. Classification results are logged into MongoDB Atlas. Communication between components is facilitated by a Wi-Fi module. This architecture enables automated waste segregation, with the ESP32 controlling the servo motor to direct waste based on classification results, enhancing efficiency in waste management.

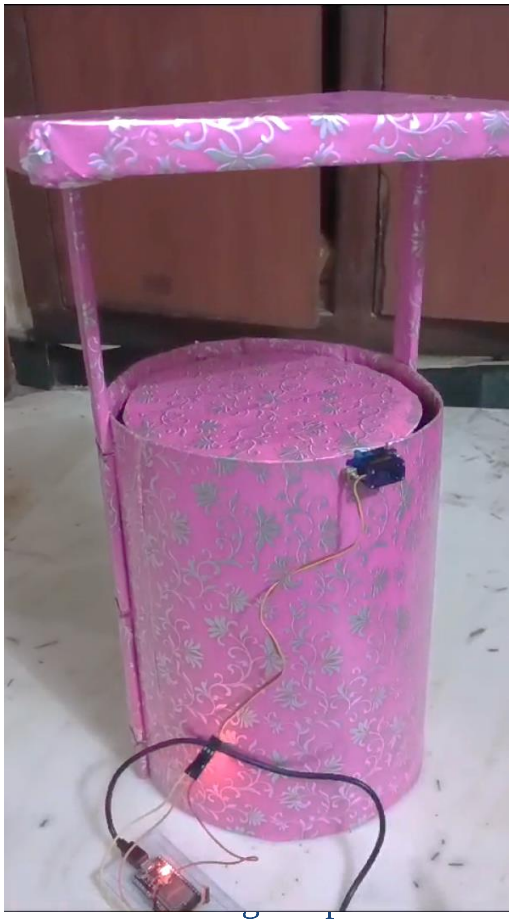
3.4 OUTPUT

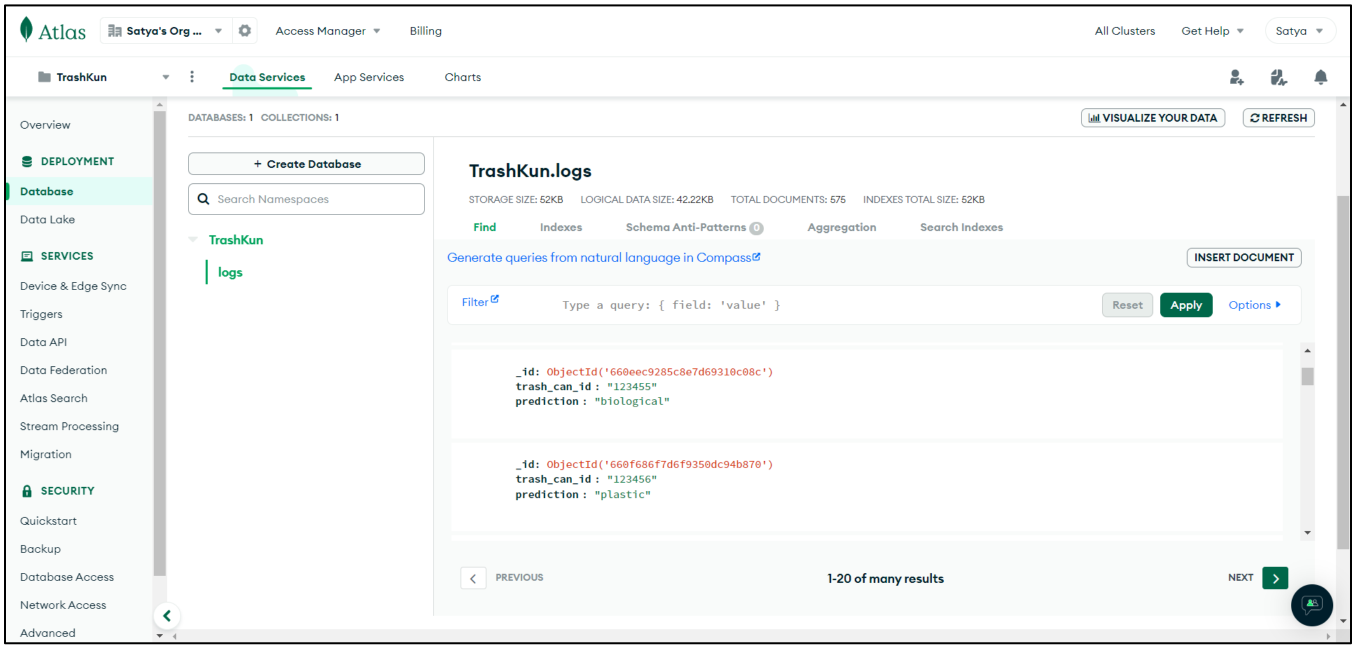

Intial Setup


Trash is placed

Trash is left to settle

Trash falls into the organic compartment





DATABASES: 1 COLLECTIONS: 1

TrashKun.logs

STORAGE SIZE: 52KB LOGICAL DATA SIZE: 42.22KB TOTAL DOCUMENTS: 575 INDEXES TOTAL SIZE: 52KB

```
_id: ObjectId('660eec9285c8e7d69310c08c')
trash_can_id : "123455"
prediction : "biological"

_id: ObjectId('660f686f7d6f9350dc94b870')
trash_can_id : "123456"
prediction : "plastic"
```

1-20 of many results

Cloud Database (MongoDB) for Logs

CHAPTER 4
CONCLUSION AND FUTURE ENHANCEMENT

4.1 CONCLUSION

The implementation of the AI-powered trash can system demonstrates its effectiveness in automating waste sorting processes, thereby contributing to efficient waste management practices. By leveraging machine learning algorithms and IoT technologies, the system accurately classifies trash into organic and inorganic categories, facilitating proper disposal and recycling. The integration of real-time data processing and servo motor control ensures timely and precise actions, enhancing user experience and system reliability. Overall, the project signifies the potential of innovative solutions in addressing environmental challenges and promoting sustainability.

4.2 FUTURE ENHANCEMENT

To further enhance the system, future developments could focus on improving classification accuracy through advanced machine learning techniques. Additionally, scalability and integration with cloud platforms can extend the system's reach and capabilities, enabling widespread deployment and remote management. Incorporating sensor fusion technologies and prioritizing energy efficiency are also essential for optimizing system performance and reducing operational costs. These enhancements will contribute to the continued evolution of smart waste management solutions, addressing the growing challenges of urban waste management while promoting sustainable practices.

APPENDIX

SOURCE CODE:

WSL Server

```
import time
from flask import Flask, request, jsonify
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import tensorflow_hub as hub
import numpy as np
import os
from flask_cors import CORS
from pymongo import MongoClient
import requests

app = Flask(__name__)
CORS(app)

# Connect to MongoDB
client =
MongoClient('mongodb+srv://satya:12345@trashkun.dxsnir6.mongodb.net/?retryWrites
=true&w=majority&appName=TrashKun')
db = client['TrashKun']
logs_collection = db['logs']

# Load the TensorFlow model
model = load_model("model.h5", custom_objects={'KerasLayer': hub.KerasLayer})
```

```
classes = os.listdir("/mnt/d/Academics/Projects/DL/TrashAI/dataset/")
n = len(classes)
label_ind = {i: classes[i] for i in range(n)}

def preprocess_image(img_path):
    img = image.load_img(img_path, target_size=(224, 224))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array = img_array / 255.0
    return img_array

@app.route("/predict", methods=["POST"])
def predict():
    if "image" not in request.files:
        return jsonify({"error": "No images uploaded"})
    img_file = request.files["image"]
    img_path = "trash.jpeg"
    img_file.save(img_path)
    img_array = preprocess_image(img_path)
    try:
        predictions = model.predict(img_array)
        # os.remove(img_path)
        predList = predictions.tolist()
        l = [(predList[0][i], label_ind[i]) for i in range(n)]
        ans = max(l)
        print(ans)
        # Log prediction to MongoDB
        trash_can_id = request.form.get("trash_can_id")
        log = {"trash_can_id": trash_can_id, "prediction": ans[1]}
        logs_collection.insert_one(log)
        if(ans[1] in ["biological","cardboard","clothes","paper"]):
            return "1"
        else:
            return "0"

    except Exception as e:
        # os.remove(img_path)
        return jsonify({"error": str(e)})

if __name__ == '__main__':
    app.run(debug=False,host="0.0.0.0")
```

Windows Server

```
import time
from flask import Flask, jsonify, request, send_from_directory
from flask_cors import CORS
import requests
import base64

app = Flask(__name__)
CORS(app)

# api="http://192.168.0.105:5000/"
api= "http://192.168.182.184:5000/"
```

```
@app.route("/capture_and_predict", methods=["GET"])
def capture_and_predict():
    try:
        response=requests.post(api+"capture",data="capture")
        time.sleep(1)
        if(response.status_code==200):
            image_file = {'image': open('trash.jpeg', 'rb')}
            if image_file:
                     response  =  requests.post("http://localhost:5000/predict",  files=image_file,
data={"trash_can_id": "12345612"})
                if response.status_code == 200:
                    print("Response: ",response.text)
                    return response.text
                else:
                    return "Error processing image"
            else:
                return "Failed to capture image"
        else:
            return "Post req to caputre Failed"

    except Exception as e:
        return "Error capturing image: " + str(e)

@app.route("/capture_image", methods=["POST"])
def capture_image():
    try:
        image_data = request.data.decode('utf-8').split(",")[1]  # Extract the base64 string from the data
        image_bytes = base64.b64decode(image_data)

        # Code to save the image to the file system
        with open("trash.jpeg", "wb") as f:
            f.write(image_bytes)

        return jsonify({"message": "Image captured and saved successfully"}), 200
    except Exception as e:
        return jsonify({"error": str(e)}), 500

data=None
@app.route("/capture",methods=["POST","GET"])
def capture():
    global data
    if(request.method=="POST"):
        print("Hellooooo")
        try:
            data="1"
            print("Innn")
            return jsonify({"stat":"ok"}),200
        except Exception as e:
            return jsonify({"error":str(e)}),500
    elif(request.method=="GET"):
        try:
            print("In get")
            if(data!=None):
                temp=data
                data=None
                return "1",200
            else:
                return jsonify({"res":"no data"}),200
        except Exception as e:
            return jsonify({"error":str(e)}),500

HTML_DIRECTORY = "./"

@app.route("/camera")
def camera():
    return send_from_directory(HTML_DIRECTORY, "index.html")

if __name__ == '__main__':
    app.run(debug=True,host="0.0.0.0")
```

Camera Website

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Camera Capture</title>
</head>
```

```
<body>
    <h1>Camera Capture</h1>
    <video id="video" width="640" height="480" autoplay></video>
    <br>
    <button id="captureBtn">Capture Image</button>

    <script>
        const video = document.getElementById('video');
        const captureBtn = document.getElementById('captureBtn');

        async function setupCamera() {
            try {
                const stream = await navigator.mediaDevices.getUserMedia({ video: true });
                video.srcObject = stream;
                video.onloadedmetadata = () => {
                    video.play();
                };
            } catch (err) {
                console.error('Error accessing the camera:', err);
            }
        }

        function captureImage() {
            const canvas = document.createElement('canvas');
            canvas.width = video.videoWidth;
            canvas.height = video.videoHeight;
    const ctx = canvas.getContext('2d');
    ctx.drawImage(video, 0, 0, canvas.width, canvas.height);
    const imageData = canvas.toDataURL('image/jpeg');

    sendImageToServer(imageData);
}

function sendImageToServer(imageData) {
    fetch("http://192.168.182.184:5000/capture_image", {
        method: 'POST',
        headers: {
            'Content-Type': 'multipart/form-data'
        },
        body: imageData
    })
        .then(response => {
            if (!response.ok) {
                throw new Error('Failed to send image to server');
            }
            console.log('Image sent successfully');
        })
        .catch(error => {
            console.error('Error sending image to server:', error);
        });
}

window.onload = () => {
```

```
            setupCamera();
        };

        captureBtn.addEventListener('click', captureImage);

        setInterval(async () => {
            try {
                const response = await fetch("http://192.168.182.184:5000/capture")
                console.log(response);
                const val = await response.text();
                console.log(val);
                if (val == "1") {
                    console.log("Capture req received");
                    captureImage();
                }
                else {
                    console.log("No Cap req");
                }
            }
            catch (e) {
                console.log("No get");
            }
        }, 1000)
    </script>
</body>

</html>
```

ESP32 Code

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <Servo.h>

const char* ssid = "SatyaM31";
const char* password = "Satya123";
const char* serverUrl = "http://192.168.182.184:5000/capture_and_predict";

Servo servoMotor;
const int servoPin = 2;
const int ultrasonicTriggerPin = 5;
const int ultrasonicEchoPin = 6;

void setup() {
  Serial.begin(115200);
  servoMotor.attach(servoPin);
  pinMode(ultrasonicTriggerPin, OUTPUT);
  pinMode(ultrasonicEchoPin, INPUT);
  connectToWiFi();
  servoMotor.write(0);
  delay(5000);
}

void loop() {
  if (detectObject()) {
    sendSignalToServer();
    delay(6000);
  }
}

void connectToWiFi() {
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
  }
}

bool detectObject() {
  digitalWrite(ultrasonicTriggerPin, LOW);
  delayMicroseconds(2);
  digitalWrite(ultrasonicTriggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(ultrasonicTriggerPin, LOW);

  unsigned long duration = pulseIn(ultrasonicEchoPin, HIGH);
  unsigned long distance = duration * 0.034 / 2;
  return distance < 20;
}

void sendSignalToServer() {
  HTTPClient http;
  http.begin(serverUrl);
  int httpResponseCode = http.GET();
  delay(2000);
  if (httpResponseCode > 0) {
    delay(1000);
    String response = http.getString();
    controlServo(response);
  }
  http.end();
}

void controlServo(String response) {
  if (response == "1") {
    servoMotor.write(0);
    delay(1000);
    servoMotor.write(90);
    delay(5000);
  } else if (response == "0") {
    servoMotor.write(180);
    delay(1000);
    servoMotor.write(90);
    delay(5000);
  }
}
```

REFERENCES

[1] Ongondo, F. O., Williams, I. D., & Cherrett, T. J. (2011). How are WEEE doing? A global review of the management of electrical and electronic wastes. Waste Management, 31(4), 714-730.

[2] Guerrero, L. A., Maas, G., & Hogland, W. (2013). Solid waste management challenges for cities in developing countries. Waste Management, 33(1), 220-232.

[3] Kauppinen, T., Kaarna, A., & Seppänen, T. (2016). Automated waste sorting—A review of sensor-based waste sorting methods. Measurement Science Review, 16(4), 157-170.

[4] Zhao, J., Zhang, X., & Wang, S. (2018). An efficient object detection system for waste sorting using deep learning. IEEE Access, 6, 55872-55884.

[5] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, 25, 1097-1105.

[6] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 770-778).

[7] Tan, M., & Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946.

[8] Al Mamun, M. A., Hannan, M. A., & Hussain, A. (2016). Real-time solid waste bin monitoring system framework using wireless sensor network. IEEE Sensors Journal, 15(8), 4561-4571.

[9] Aazam, M., & Huh, E. N. (2014). Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT. In Proceedings of the 2014 IEEE 6th International Conference on Cloud Computing Technology and Science (pp. 712-717).

[10] Longhi, S., Marzioni, D., Alidori, E., Buo, D. D., Prist, M., Grisostomi, M., & Pirro,

M. (2012). Solid waste management architecture using wireless sensor network technology. IEEE Sensors Journal, 13(10), 3547-3555.

[11] Folianto, F., Low, K. S., & Yeoh, W. O. (2015). Smartbin: Smart waste management system. 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP) (pp. 1-2).

[12] Belić, M., Latinović, T., Milinković, D., & Šijan, G. (2020). Machine learning and IoT for intelligent waste management. In 2020 18th International Conference on Emerging eLearning Technologies and Applications (ICETA) (pp. 272-277).

[13] Silva, S., Khan, M., Han, K., & Han, K. (2019). Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart waste management. Sensors, 19(8), 1973.

[14] Zhang, C., Ren, J., Wang, S., & Han, Z. (2021). Development of a smart waste bin using IoT technologies. Sustainable Cities and Society, 64, 102519.

[15] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. IEEE Communications Surveys & Tutorials, 17(4), 2347-2376.

[16] Gaur, A., Scotney, B., Parr, G., & McClean, S. (2015). Smart city architecture and its applications based on IoT. Procedia Computer Science, 52, 1089-1094.