GarboGo

Transforming Waste Collection for a Greener Future

EFFICIENT WASTE COLLECTION

KHYATI GAURANA
2021UCS1527

GAURAV KUMAR
2021UCS1539

SANCHI SINGH
2021UCS1565

GarboGo

Transforming Waste Collection for a Greener Future

OVERVIEW

GarboGo

Transforming Waste Collection for a Greener Future

INTRODUCTION

The upsurge in urban population correlates with a heightened demand for municipal services, placing additional stress on waste management facilities. The increase in city residents invariably leads to a proportional rise in the generation of municipal solid waste, prompting the need for the development of supplementary waste disposal sites to accommodate the amplified volume.

In this paper, we focus on optimizing bin collection scheduling within the domain of waste management. In this context, given the multiple criteria to decide against, the Technique of Order Preference by Similarity to Ideal Solution is employed.

GarboGo

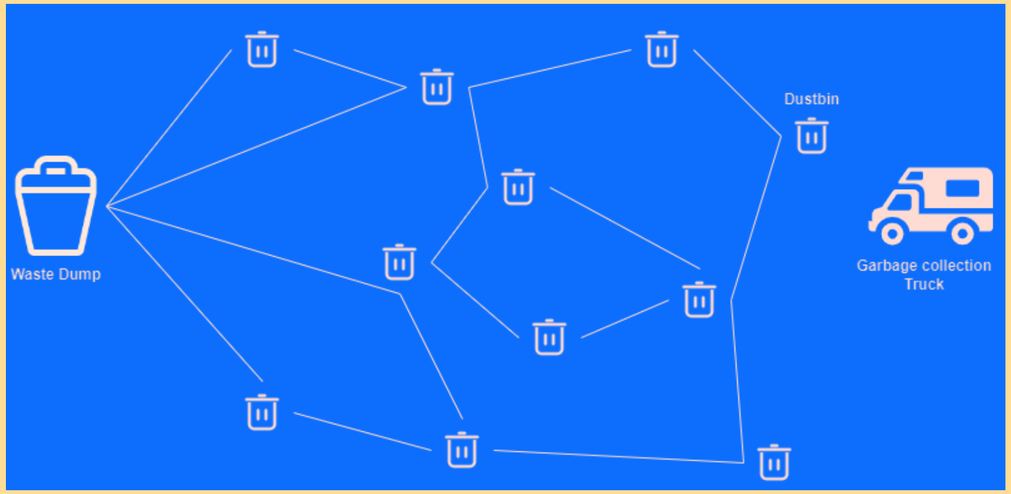
Transforming Waste Collection for a Greener Future

MOTIVATION

Inadequate waste disposal practices lead to the release of significant quantities of greenhouse gasses, particularly methane, as organic waste decomposes. Moreover, the energy-intensive processes involved in waste treatment and disposal contribute to an increased carbon footprint, intensifying the overall environmental impact. Advancements in sensors and wireless communications have paved the way for the integration of Internet of Things (IoT) networks in numerous smart city applications. Waste management can also harness the power of IoT to ascertain the quantity and composition of waste, transmit these data to waste collection organizations, and formulate optimized routes for efficient waste collection.

GarboGo

Transforming Waste Collection for a Greener Future



PROBLEM STATEMENT

The main objectives of this project are:

- Develop an IoT-enabled waste management system that integrates sensors and smart bins to monitor waste levels in real-time.
- Implement an efficient routing algorithm based on the TOPSIS method to optimize waste collection routes and schedules.
- Minimize collection time, fuel consumption, and operational costs while maximizing waste collection efficiency and service quality.

GarboGo

Transforming Waste Collection for a Greener Future

LITERATURE REVIEW

[1] presented an agent-based waste management simulation comparing traditional periodic review with IoT-enabled smart sensor bins. The simulation Included waste generation, management, and validation using economic, environmental, and citizen satisfaction measures.

[2] utilizes real-time data from smart waste bins to manage and maintain cleanliness in smart ities. Fuzzy logic guides the strategic placement of these bins, forming a Smart Garbage Bin Mechanism (SGBM) to enhance solid waste management.

[3] introduces BIN for the CiTy (BINCT), a free intelligent software system designed to optimize waste collection routes using historical and forecast data.

[4] introduces an IoT-based garbage management system for smart cities, where each bin is equipped with a unique ID and a gadget to monitor its condition. These devices promptly send filling level data to aid waste collectors in efficient bin cleaning.

GarboGo

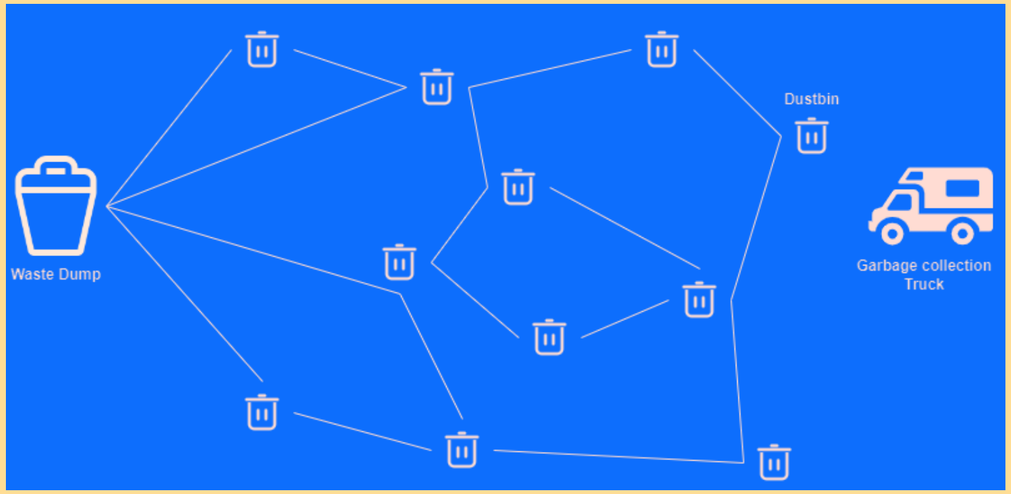
Transforming Waste Collection for a Greener Future

METHODOLOGY

- Our system comprises n bins distributed across a network, each connected to a central database via GPS for real-time monitoring.
- Equipped with sensors, the bins gather crucial data for prioritization. The ultrasonic sensor module measures waste volume (V), while air quality sensors detect toxic gases like ammonia (NH3), methane (CH3), and smoke, determining waste toxicity (v).
- Additionally, the duration of waste accumulation in the bin is measured through time (t) since the last emptying event.
- Sensor data is transmitted to a centralized server where it's stored in a database for further analysis and processing.
- Garbage collection vehicles are dispatched based on selected bins, optimizing routes according to the gathered sensor readings, thus enhancing the efficiency of waste management operations.

GarboGo

Transforming Waste Collection for a Greener Future



NOVELTY

CONTRIBUTIONS

- We use a multi-criteria decision making model called TOPSIS (Technique of Order Preference by Similarity to Ideal Solution) .
- This technique is used in cases where there are multiple influencing factors that affect the decision making process to varying degrees of importance.
- TOPSIS is used here to take the sensor readings of volume of waste, time duration of the garbage in the bin and the toxicity levels of the waste in the bin to calculate a score for each bin.
- These scores are sorted in decreasing order and the bins with the highest scores are selected for garbage collection in routing. The bins are selected such that the waste to be collected from the bins does not exceed the waste carrying capacity of the garbage truck

GarboGo

Transforming Waste Collection for a Greener Future

PROPOSED TECHNIQUE

Our proposed technique utilizes the TOPSIS (Technique of Order Preference by Similarity to Ideal Solution) model, which is ideal for decision-making scenarios with multiple influencing factors of varying importance.

- In waste collection, TOPSIS considers sensor data such as waste volume, duration of waste in the bin, and toxicity levels to calculate a score for each bin.
- These scores are then ranked in descending order, guiding the selection of bins with the highest scores for garbage collection route planning.
- The selection process ensures that the cumulative waste from chosen bins does not exceed the garbage truck's carrying capacity.
- By employing TOPSIS, our approach optimizes waste collection efficiency while considering diverse factors impacting decision making in waste management.

GarboGo

Transforming Waste Collection for a Greener Future

IMPLEMENTATION

HARDWARE

The prototype for this project is presented as a dustbin equipped with 3 sensor modules namely Neo 6M for GPS location, HC-SR04 Ultra-Sonic sensor, MQ-135 gas sensor, Arduino UNO microcontroller board, an ESP32 microcontroller with Wifi and Bluetooth connectivity and a bi-directional level shifter to connect digital I/O pins from Arduino to ESP32 board.

SOFTWARE

The website features an information page where the data about all the bins is displayed along with the fill levels of these bins. These bins are connected to the cloud via the ESP32 module that communicates the sensor readings from these bins to the ThingSpeak IOT Cloud. The web application fetches the data from the ThingSpeak API to display on the website. The website then uses this data queried from the bins to calculate the TOPSIS scores for all the bins. Furthermore, route optimization is performed after this to give the final route which gives the fastest route that visits each bin.

GarboGo

Transforming Waste Collection for a Greener Future

COMPONENTS AND TOOLS:

1. HARDWARE

- Arduino Uno
- Ultrasonic Sensor
- MQ-135
- Neo 6M GPS Module
- Voltage Divided
- ESP-Wroom-32
- Bread board
- Jumper Wires

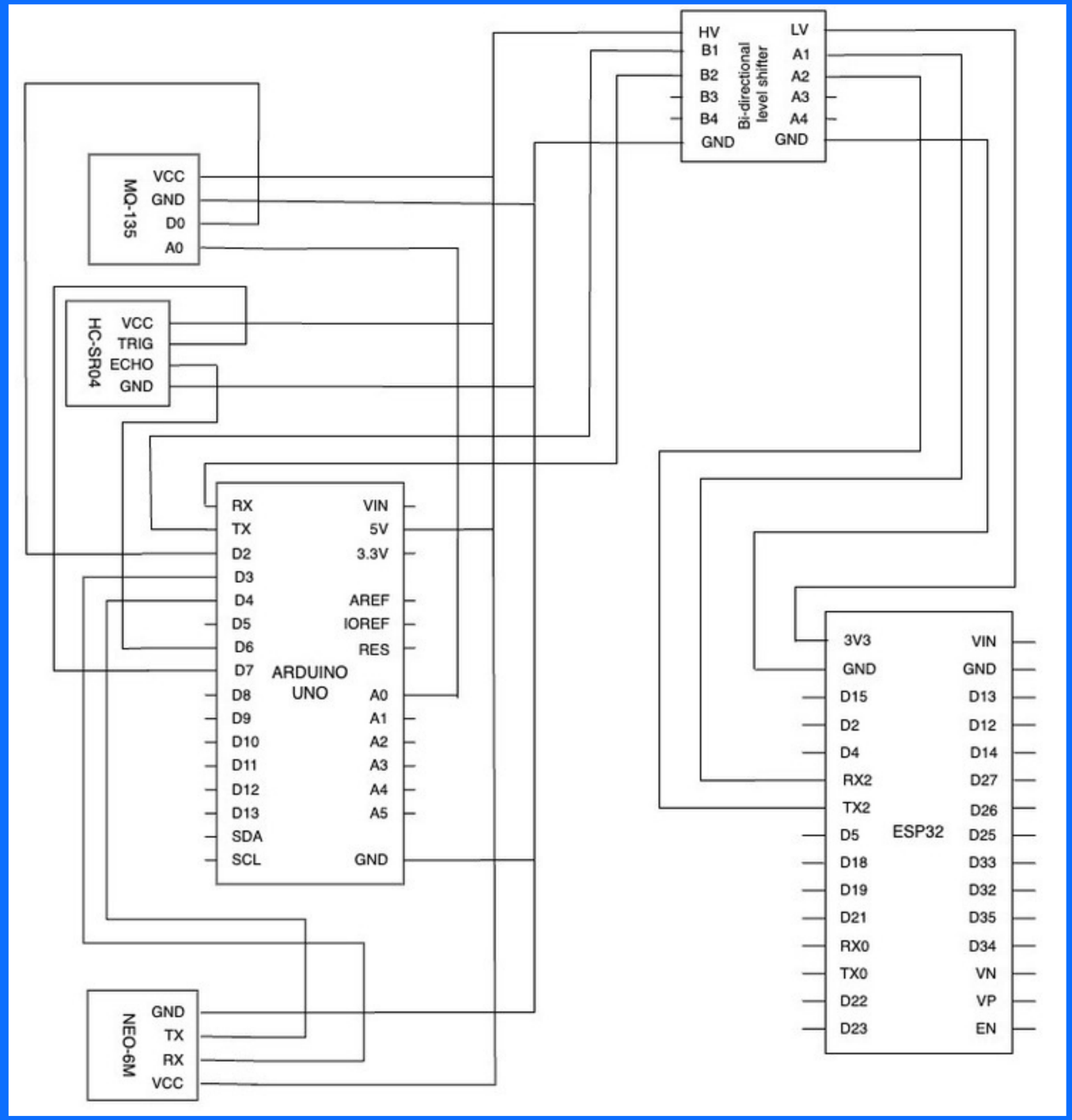
2. SOFTWARE

- React JS
- Python Flask
- Tom Tom Maps API
- Thing-Speak Cloud
- QJIS Network Simulation

GarboGo

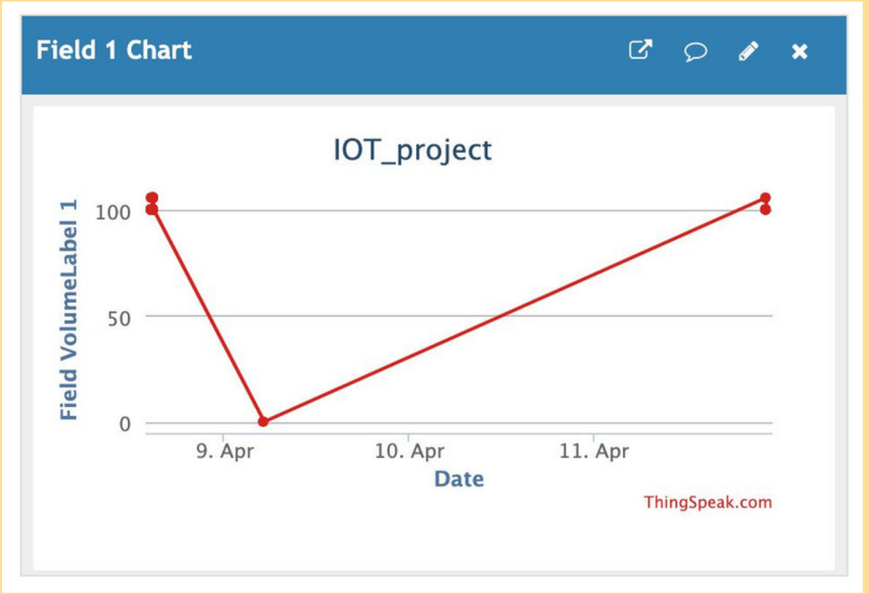
Transforming Waste Collection for a Greener Future

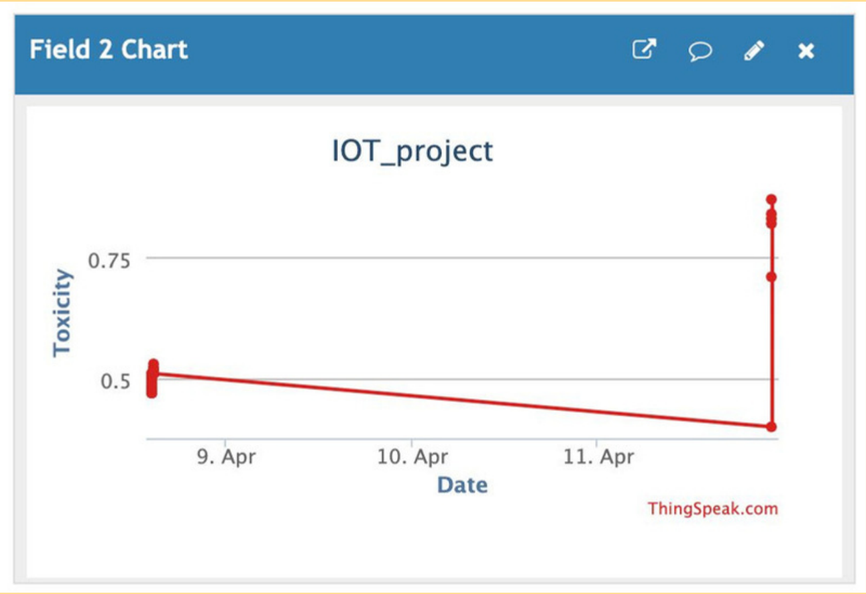
CIRCUIT DIAGRAM

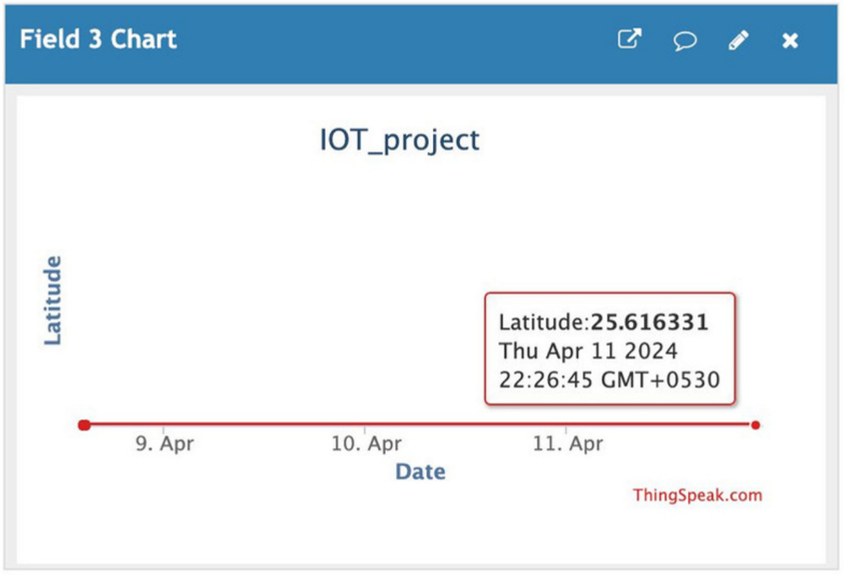


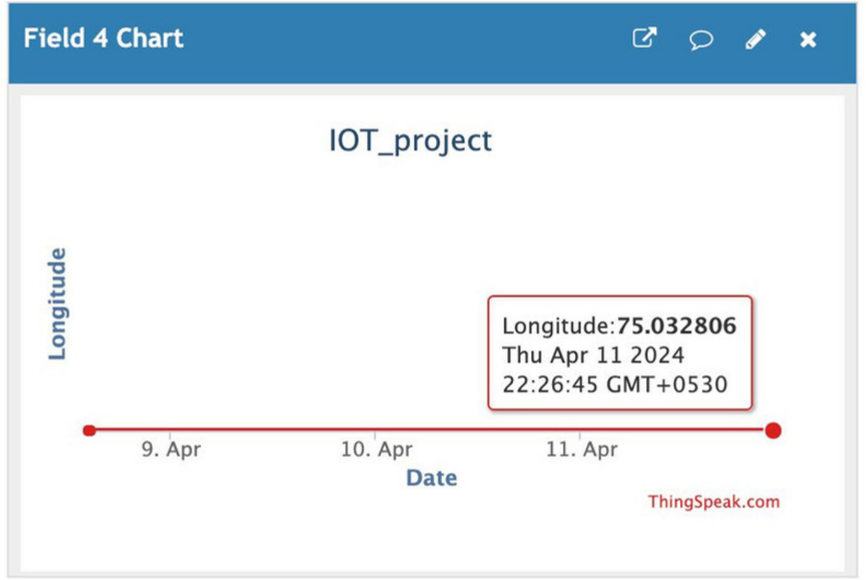
ThingsSpeak Cloud

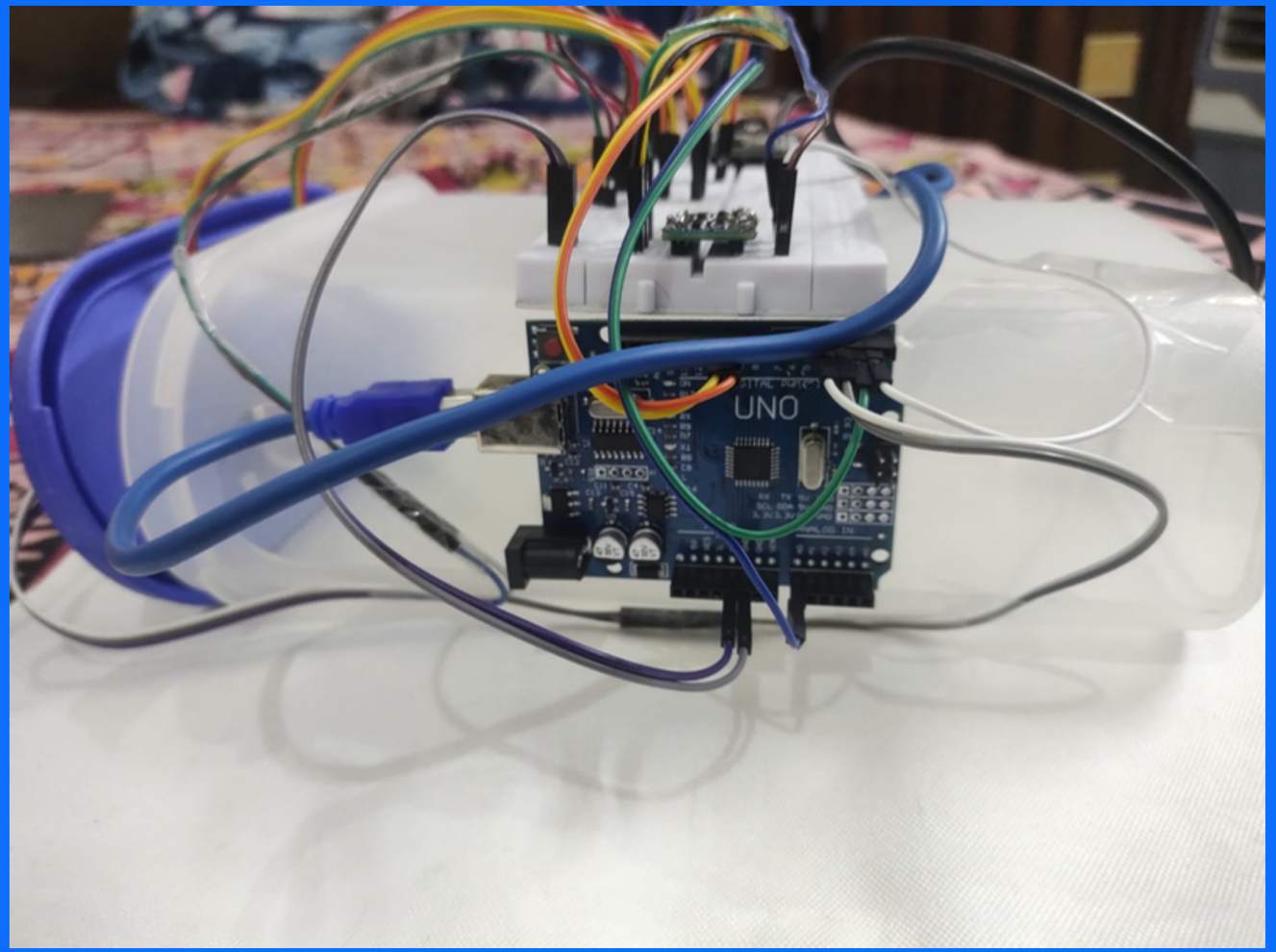
ThingSpeak is an open-source IoT platform that allows users to collect, analyze, and visualize data from sensors or devices in real time. It provides an easy-to-use interface for storing and retrieving data, as well as tools for creating custom visualizations and performing analysis



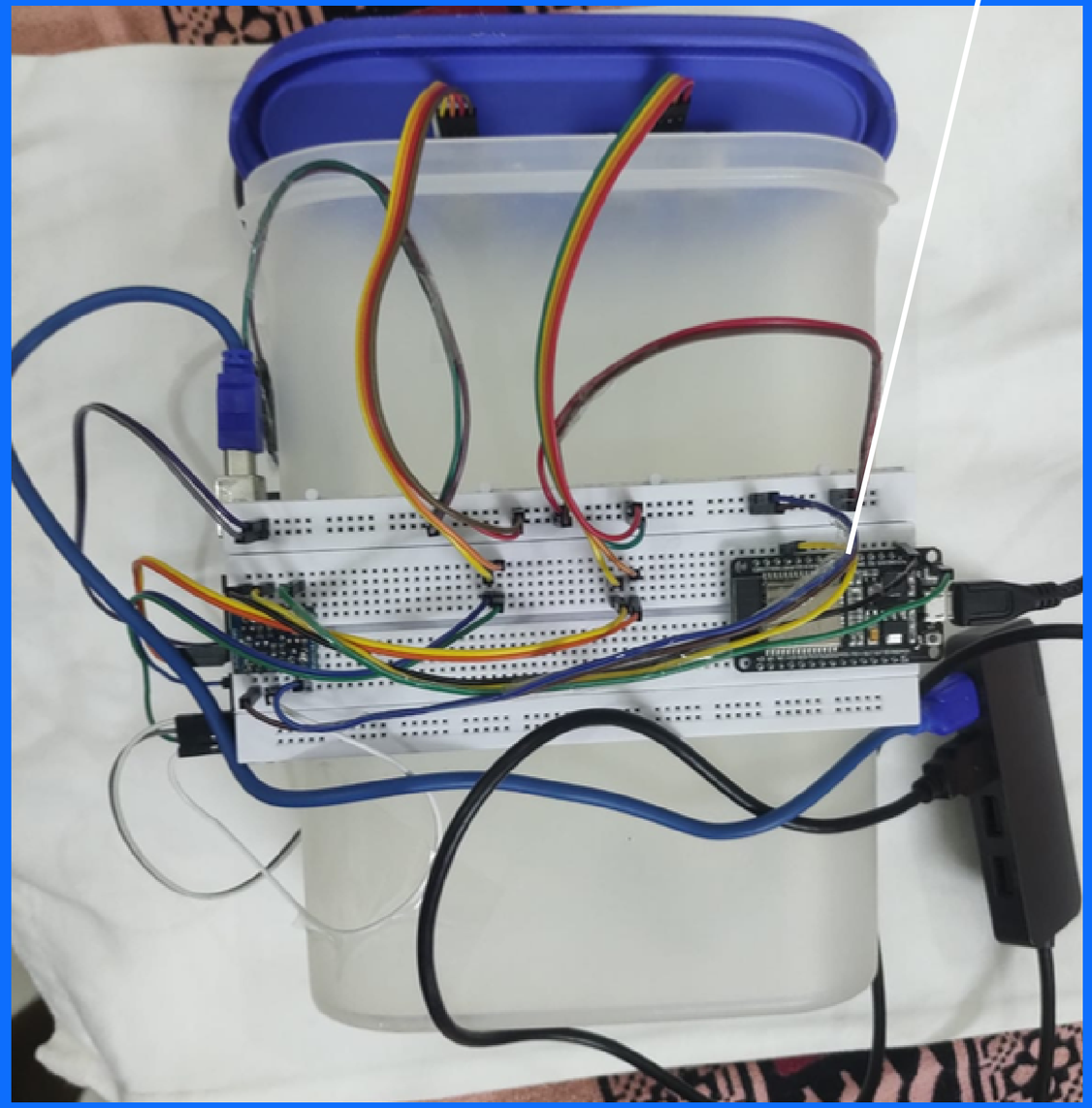






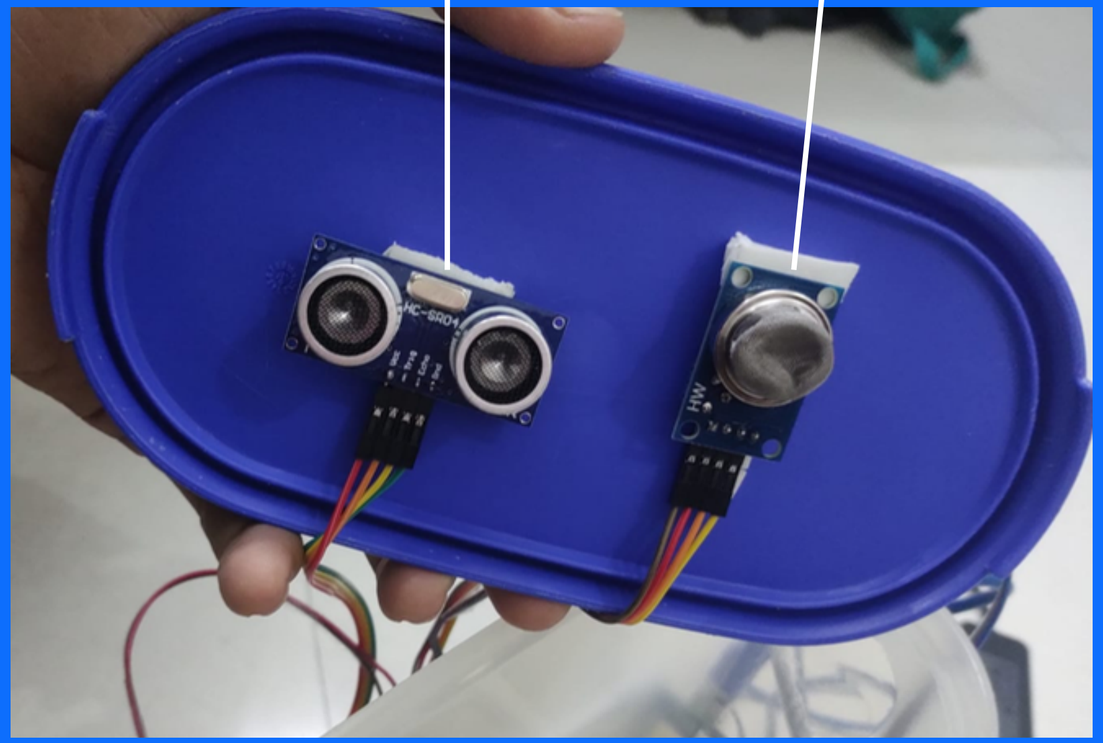

UNO

ESP32



Ultrasonic Sensor

Air quality sensor



GarboGo

Transforming Waste Collection for a Greener Future

TOPSIS ALGORITHM

A graph G with n nodes and edges e. Data regarding waste attributes as weights of G (toxicity ,, volume, time)

- Step 1: Normalize the data for each waste attribute (v, V, t) between 0 and 1. The goal is to maximize toxicity , volume and time
- Step 2: Calculate the weighted normalized values for each bin attribute based on the waste attributes and the assigned weights
- Step 3: Determine the ideal and negative-ideal solutions for each attribute.
- Ideal solution: Maximum normalized value for toxicity, volume, and time; minimum normalized value for distance.
- Negative-ideal solution: Minimum normalized value for toxicity, volume, and time; maximum normalized value for distance.
- Step 4: Calculate the proximity of each bin to the ideal and negative-ideal solutions using a distance measure. Calculate the distance of each bin from the ideal solution and the negative-ideal solution

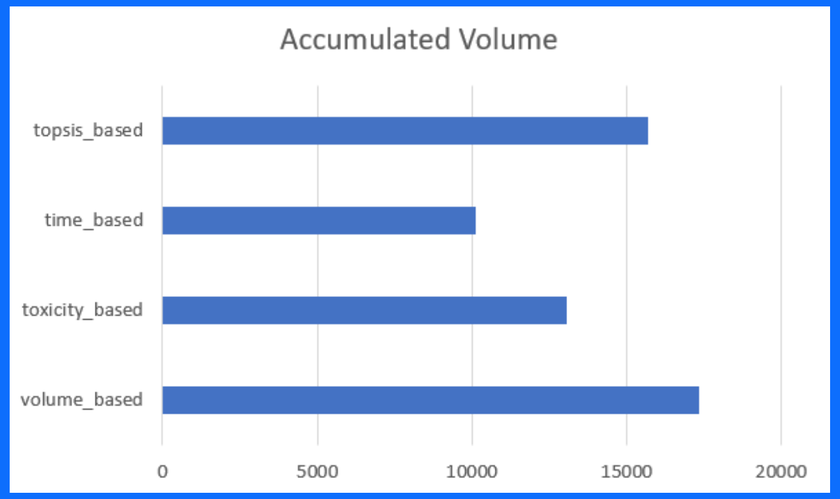
$$D_i^+ = \sqrt{\sum_{j=1}^{m} (\text{Normval}_{ij} - \text{IdealSol}_j)}$$
$$D_i^- = \sqrt{\sum_{j=1}^{m} (\text{Normval}_{ij} - \text{NegIdealSol}_j)}$$

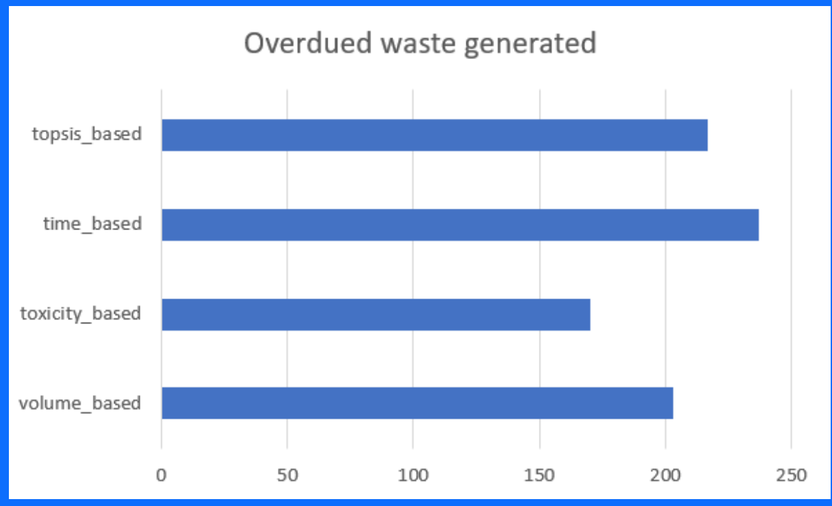
- Compute the TOPSIS score for each bin :

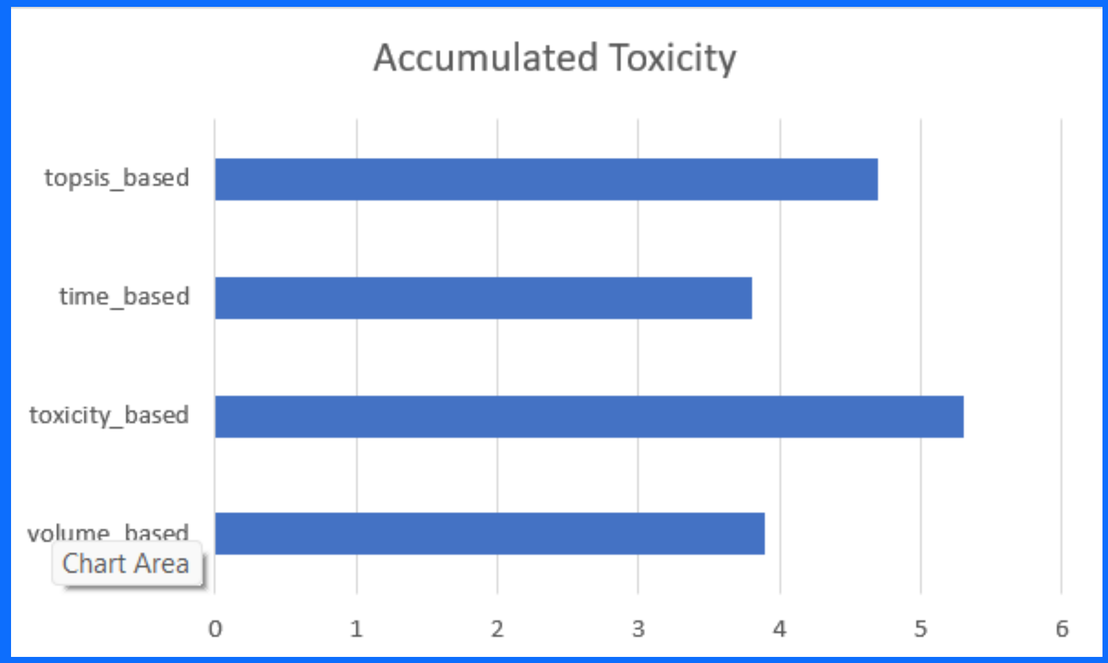
$$TOPSIS_i = \frac{D_i^-}{D_i^+ + D_i^-}$$

GarboGo

RESULT







It can be seen that the results for toxicity obtained by the TOPSIS algorithm approaches the result obtained solely by considering the toxicity parameter. The time and volume based algorithms perform the worst in terms of toxicity

In our proposed scheme performs second to the volume based algorithm which maximizes the volume of the collected waste. The other two schemes perform worst in this case as they are aimed at optimizing the corresponding parameters.

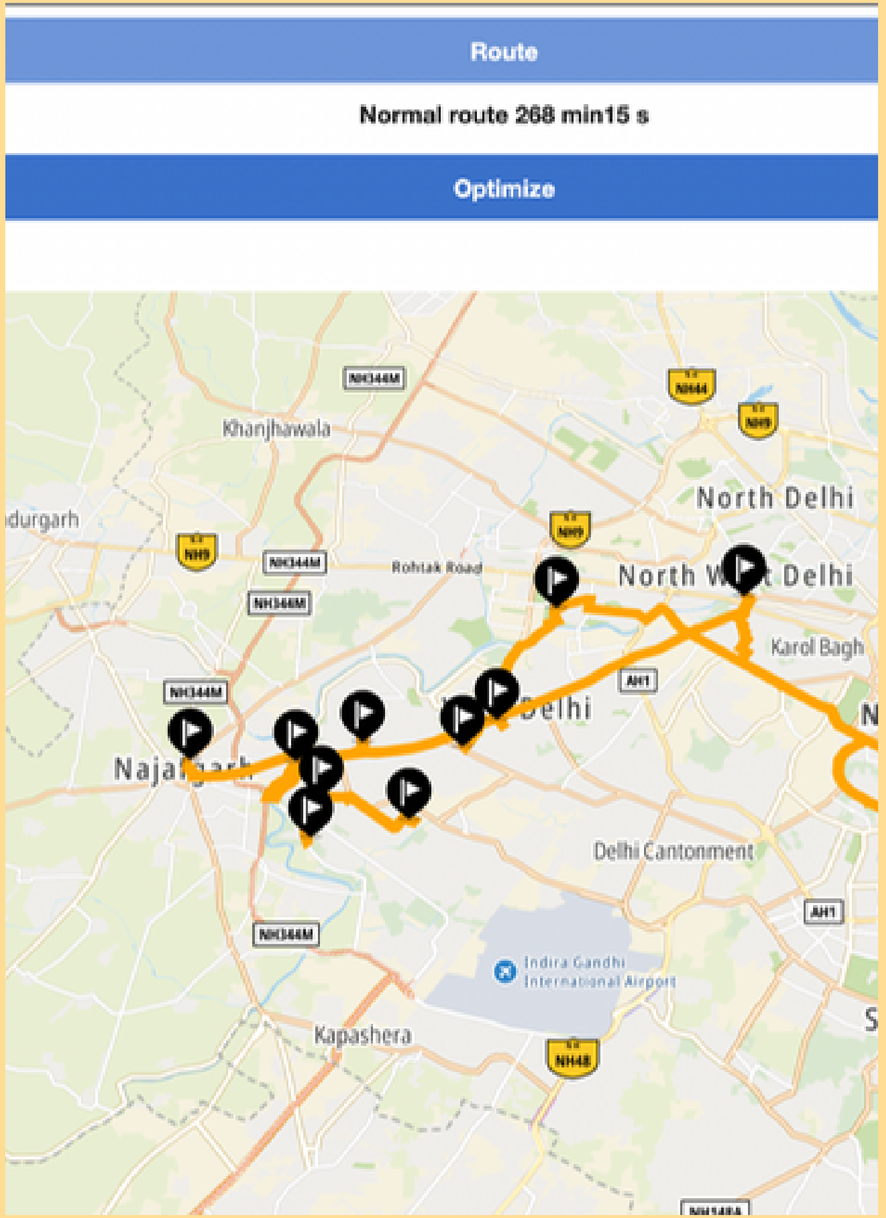
In we see that the duration before the waste is collected is maximum in the case of time-based scheme, however our proposed scheme performs only second to the this scheme.

Tom-Tom Maps API

The Map Display API is a suite of web services designed for developers to create web and mobile applications around mapping. These web services can be used via RESTful APIs.

The given image demonstrates the routing of bins based on the rankings generated by our algorithm.

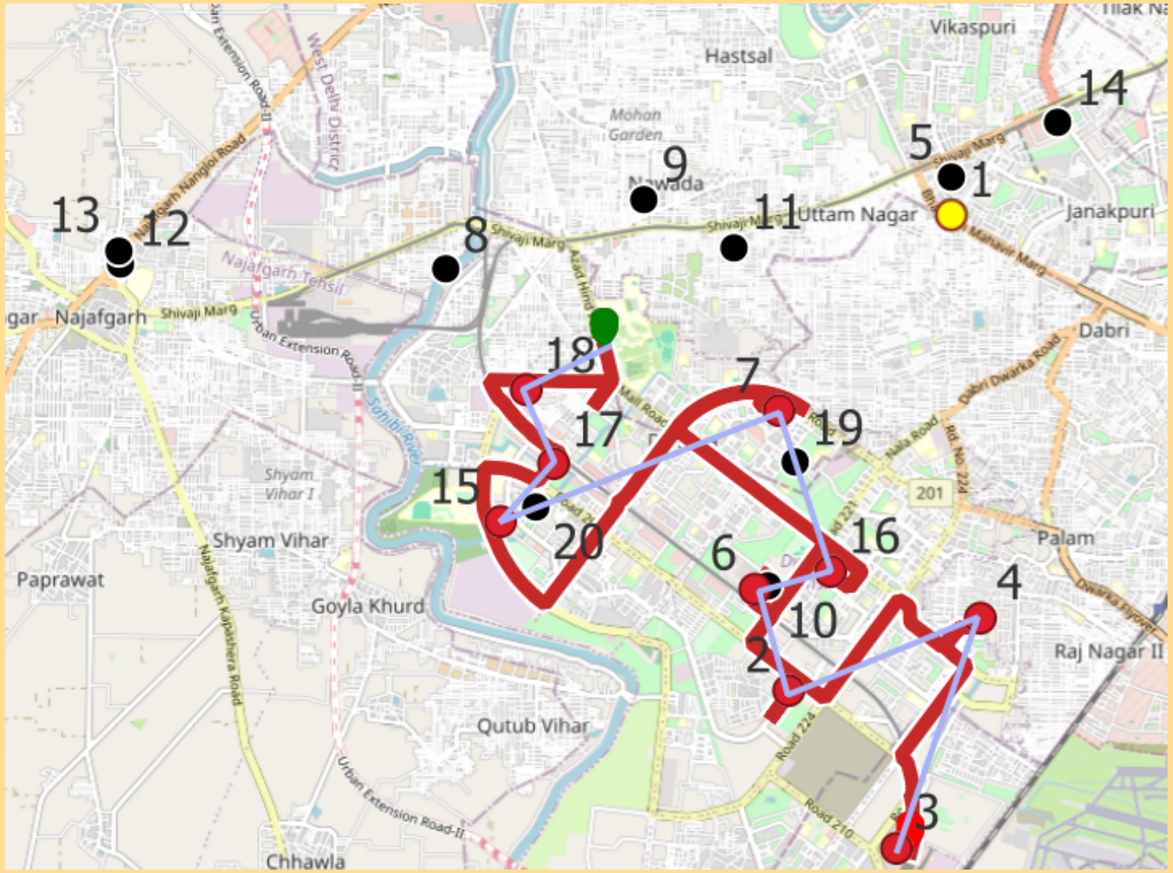
The yellow lines show the route generated by the TomTom Map Service API. The black flags are the bin locations centered around Delhi.



ROUTING SIMULATION

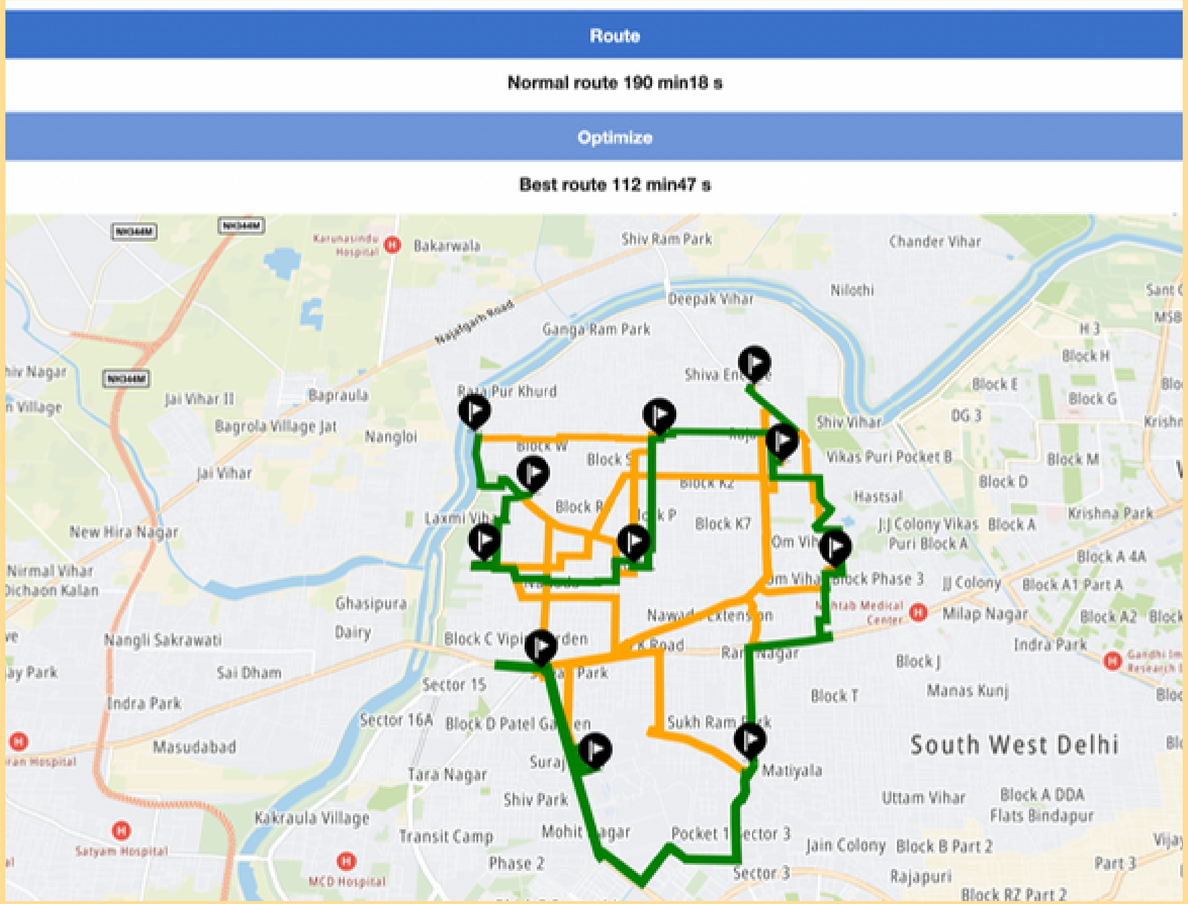
- **TOPSIS SIMULATION:**

The red path includes all the bins which were included according to their TOPSIS scores the other bins are not included.



- **FINAL OPTIMISEDROUTING PATH:**

The green path is the path optimized for minimum time taken and the yellow path is the path travelling to the bins in the order they were added.



GarboGo

Transforming Waste Collection for a Greener Future

FUTURE WORK

RECOMMENDATION 01

Currently our project only shows the optimal route covering all bin locations, this can be further optimized by providing live navigation for waste collection truck drivers.

RECOMMENDATION 02

The bins can be segregated according to different jurisdiction areas for better planning of routes for different trucks.

GarboGo

Transforming Waste Collection for a Greener Future

THANK YOU

CODE

api > myenv > api.py > predict

```python
from flask import Flask
import math
import pandas as pd
import numpy as np
from sklearn.preprocessing import normalize

app = Flask(__name__)

@app.route('/api/ml')
def predict():

  df  = pd.read_csv("dataset\data.csv")

  vol_normalized = normalize(df[['volume']], axis=0)  # axis=0 normalizes along the columns
  bin_ids = df['bin_id']
  # Normalizing the 'time' column
  time_normalized = normalize(df[['time']], axis=0)  # axis=0 normalizes along the columns
  toxicity = df['toxicity']
  df['volume_percentage'] = (df['volume'] / 1200) * 100

  weights = {
      'toxicity': 0.9,
      'volume': 1,
      'time_since_pickup': 0.7
  }

  def calculate_weighted_matrix(toxicity, volume, time_since_pickup, weights, bin_ids):
      weighted_matrix = []

      for t, v, ts, bin_id in zip(toxicity, volume, time_since_pickup, bin_ids):
          weighted_toxicity = t * weights['toxicity']
          weighted_volume = v * weights['volume']
          weighted_time = ts * weights['time_since_pickup']

          weighted_row = [bin_id, weighted_toxicity, weighted_volume, weighted_time]
          weighted_matrix.append(weighted_row)

      return weighted_matrix

  # Calculate scores for each dustbin
  ideal = [1,1,1]
  negIdeal = [0,0,0]
```

```python
# Calculate scores for each dustbin
ideal = [1,1,1]
negIdeal = [0,0,0]

#calculate score
def calcScore(weighted_matrix, ideal,negIdeal):

  score = []

  for row in weighted_matrix:
    bin_id = row[0]
    weighted_value = row[1]
    sum_pos = 0
    sum_neg = 0
    for j,k in zip(weighted_value,ideal):
      sum_pos += (j-k)**2

    for j,k in zip(weighted_value,negIdeal):
      sum_neg += (j-k)**2

    sum_pos = math.sqrt(sum_pos)
    sum_neg = math.sqrt(sum_neg)

    score_i = sum_neg/(sum_pos + sum_neg)
    score.append([bin_id,score_i])

  return score

weighted_mat = calculate_weighted_matrix(toxicity,vol_normalized,time_normalized,weights,bin_ids)

def convertToMatrix(weighted_matrix):

  wt_mat = []
  for row in weighted_mat:
      bin_id = row[0]
      weighted_toxicity = row[1]
      weighted_vol = row[2][0]
      weighted_time = row[3][0]
      wt_mat.append([bin_id,[weighted_toxicity,weighted_vol,weighted_time]])

  return wt_mat

wt_mat = convertToMatrix(weighted_mat)
```

```
scores  = calcScore(wt_mat,ideal,negIdeal)
sorted_Scores = sorted(scores, key=lambda x: x[1], reverse=True)

scores_df = pd.DataFrame(sorted_Scores, columns=['bin_id', 'score'])
top_bin_ids = scores_df.head(10)['bin_id'].tolist()
top_bins_data = df[df['bin_id'].isin(top_bin_ids)][['lat', 'long', 'volume_percentage', 'toxicity']]

return {'bin_list':top_bins_data}
```

src > JS Topsis.js > Topsis

```jsx
import React, { useState, useEffect } from 'react';

function Topsis() {
  const [binsData, setBinsData] = useState([]);

  useEffect(() => {
    async function fetchData() {
      try {
        const response = await fetch('/api/myenv/api'); // Assuming your Flask API is running on the same host
        const data = await response.json();
        console.log(data)
        setBinsData(data.bin_list);
      } catch (error) {
        console.error('Error fetching data:', error);
      }
    }

    fetchData();
  }, []);

  return (
    <div>
      <h1>Top Bins Data</h1>
      <ul>
        {binsData.map(bin => (
          <li key={bin.bin_id}>
            <div>Latitude: {bin.lat}</div>
            <div>Longitude: {bin.long}</div>
            <div>Volume Percentage: {bin.volume_percentage}</div>
            <div>Toxicity: {bin.toxicity}</div>
          </li>
        ))}
      </ul>
    </div>
  );
}
```