Project Report

1. **INTRODUCTION**
    1. Project Overview
    2. Purpose
2. **LITERATURE SURVEY**
    1. Existing problem
    2. References
    3. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
    1. Empathy Map Canvas
    2. Ideation & Brainstorming
    3. Proposed Solution
    4. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
    1. Functional requirement
    2. Non-Functional requirements
5. **PROJECT DESIGN**
    1. Data Flow Diagrams
    2. Solution & Technical Architecture
    3. User Stories
6. **PROJECT PLANNING & SCHEDULING**
    1. Sprint Planning & Estimation
    2. Sprint Delivery Schedule
    3. Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
    1. Feature 1
    2. Feature 2
    3. Database Schema (if Applicable)
8. **TESTING**
    1. Test Cases
    2. User Acceptance Testing
9. **RESULTS**
    1. Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**

Source Code

GitHub & Project Demo Link

Project Report

Team ID	**PNT2022TMID42867**
Project Name	**Smart waste management system for metropolitan cities**

1. INTRODUCTION

1.1 Project Overview:

With the increasing population and industrialization of nations throughout the globe, waste has become a great concern for all of us. Over years, researchers figured that only waste management is not enough for its proper treatment and disposal techniques to preserve our environment and keeping it clean in this era of globalization. With the help of technology researchers have, introduced IoT based Smart Waste Management solutions and initiatives that ensures reduced amount of time and energy required to provide waste management services and reduce the amount of waste generated. Unfortunately, developing countries are not being able to implement those existing solutions due to many factors like socio-economic environment. Therefore, in this research we have concentrated our thought on developing a smart IoT based waste management system for developing countries like INDIA that will ensure proper disposal, collection, transportation and recycling of household waste with the minimum amount of resources being available

1.2 Purpose:

We amalgamate technology along with waste management in order to effectively create a safe and a hygienic environment. Smart waste management is about using technology and data to create a more efficient waste industry. Based on IoT (Internet of Things) technology, smart waste management aims to optimize resource allocation, reduce running costs, and increase the sustainability of waste services. This makes it possible to plan more efficient routes for the trash collectors who empty the bins, but also lowers the chance of any bin being full for over a week. A good level of coordination exists between the garbage collectors and the information supplied via technology. This makes them well aware of the existing garbage level and instigate them whenever the bins reach the threshold level. They are sent with alert messages so that they can collect the garbage on time without littering the surrounding area. The fill patterns of specific containers can be identified by historical data and managed accordingly in the long term. In addition to hardware solutions, mobile applications are used to overcome the challenges in the regular waste management system, such as keeping track of the drivers while they are operating on the field. Thus, smart waste management provides us with the most optimal way of managing the waste in an efficient manner using technology.

2. LITERATURE SURVEY:

2.1 Existing problem:

Waste management has become an alarming challenge in local towns and cities across the world. Often the local area bins are overflowing and the municipalities are not aware of it. This affects the residents of that particular area in numerous ways starting from bad odour to unhygienic and unsafe surroundings. Poor waste management - ranging from non-existing collection systems to ineffective disposal -causes air pollution, water and soil contamination. Open and unsanitary areas contribute to contamination of drinking water and can cause infection and transmit diseases. Toxic components such as Persistent Organic Pollutants (POPs) pose particularly significant risks to human health and the environment as they accumulate through the food chain. Animals eating contaminated plants have higher doses of contaminants than if they were directly exposed. Precipitation or surface water seeping through waste will absorb hazardous components from landfills, agricultural areas, feedlots, etc. and carry them into surface and groundwater. Contaminated groundwater also poses a great health risk, as it is often used for drinking, bathing and recreation, as well as in agricultural and industrial activities. Landfills and waste transfer stations can attract various pests (insects, rodents, gulls, etc.) that look for food from waste. These pests can spread diseases through viruses and bacteria (i.e., salmonella and e-coli), which are a risk to human health.

2.2 References:

PAPER 1:

TITLE: IoT Based Waste Management for Smart City

AUTHOR NAME: Parkash Tambare, Prabu Venkatachalam

PUBLICATION YEAR: 2016

DESCRIPTION:

In the current situation, we frequently observe that the trash cans or dust cans that are located in public spaces in cities are overflowing due to an increase in the amount of waste produced each day. We are planning to construct "IoT Based Waste Management for Smart Cities" to prevent this from happening because it makes living conditions for people unsanitary and causes unpleasant odours in the surrounding area. There are numerous trash cans scattered throughout the city or on the campus that are part of the proposed system. Each trash can is equipped with a low-cost embedded device that tracks the level of the trash cans and an individual ID that will enable it to be tracked and identified.

PAPER 2:

AUTHOR NAME: Mohammad Aazam, Marc St-Hilaire, Chung-Horng Lung, Ioannis Lambadaris

PUBLICATION YEAR: 2016

DESCRIPTION:

Each bin in the Cloud SWAM system that Mohammad Aazam et al suggested has sensors that can detect the amount of waste inside. There are separate bins for organic, plastic/paper/bottle/glass, and metal waste. This way, each form of waste is already divided, and it is known how much and what kind of waste is collected thanks to the status. Different entities and stakeholders may benefit from the accessibility of cloud-stored data in different ways. Analysis and planning can begin as soon as garbage is collected and continue through recycling and import/export-related activities. Timely garbage collection is provided via the Cloud SWAM system. A timely and effective method of waste collection improves health, hygiene, and disposal.

PAPER 3:

TITLE: Arduino Microcontroller Based Smart Dustbins for Smart Cities

AUTHOR NAME: K. Suresh, S. Bhuvanesh and B. Krishna Devan

PUBLICATION YEAR: 2019

DESCRIPTION:

In this paper, a technique for cleaning up our surroundings and environment is described. The Indian government just began work on a smart city initiative, and in order for these towns to be smarter than they already are, the garbage collection and disposal system must be improved upon. Self-Monitoring Automated Route Trash (SMART) dustbins are intended for use in smart buildings such as colleges, hospitals, and bus stops, among other places. In this study, we have employed the PIR and Ultrasonic sensors to detect human presence, the Servomotor to open the dustbin lid, and the Ultrasonic sensor to detect the level of rubbish. Signals between two trash cans are transmitted using a communication module, and the GSM module sends the message to the operator.

PAPER 4:

AUTHOR NAME: Mohd Helmy Abd Wahab, Aeslina Abdul Kadir, Mohd Razali Tomari and Mohamad Hairol Jabbar

PUBLICATION YEAR: 2014

DESCRIPTION:

Proposed a smart recycle bin that can handle the recycling of plastic, glass, paper, and aluminium cans. It generates a 3R card after automatically determining the value of the trash thrown away. The recycle system makes it possible to accumulate points for placing waste into designated recycle bins. By allowing the points to be redeemed for goods or services, such a system promotes recycling activities. The system keeps track of information on disposal procedures, materials disposed of, user identification, and points accrued by the user.

PAPER 5:

TITLE: Waste Management Initiatives in India For Human Wellbeing

AUTHOR NAME: Dr. Raveesh Agarwal, Mona Chaudhary and Jayveer Singh

PUBLICATION YEAR: 2015

DESCRIPTION:

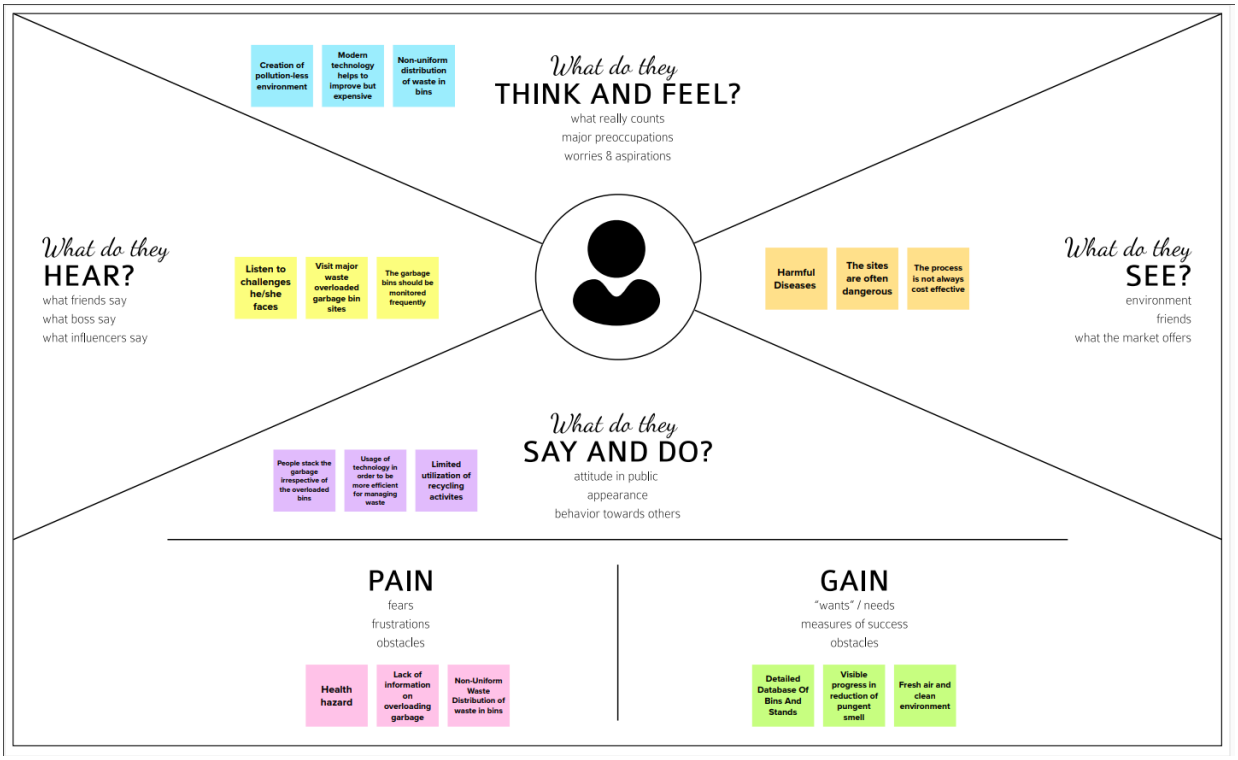
The objective of this paper is to examine the present methods used in India for the welfare of its people in different waste management efforts. The other goal is to offer advice on how to make Indian municipalities' trash disposal procedures better. On secondary research, this essay is founded. The system is improved by looking at the reports that have already been written about waste management and the suggestions made for improvement by planners, NGOs, consultants, government accountability organisations, and important business leaders. It provides in-depth understanding of the various waste management programmes in India and identifies areas where waste management might be improved for societal benefit. The essay makes an effort to comprehend the crucial part that our nation's official waste management sector plays in the waste management process.

2.3 Problem Statement Definition:

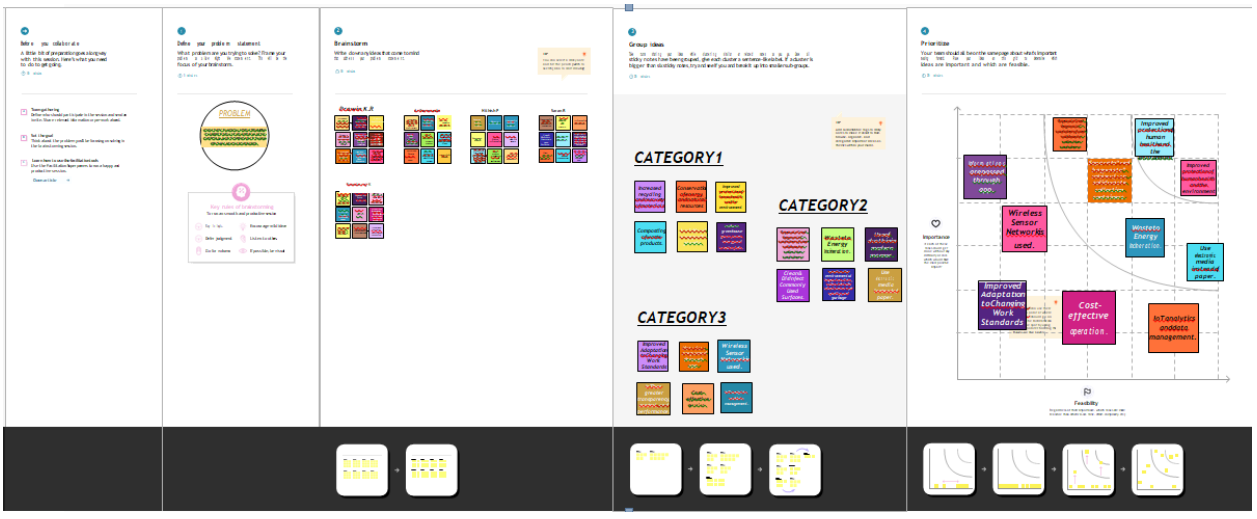
Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes mefeel**
PS-1	Municipal corporation authority	Get notified when the trash cans are full and be made aware of where the full cans are located.	Don't have the facilities at the moment	There is no tool available to determinethe level of bins.	Frustrated
PS-2	Individual working for a private limited corporation	Get rid of the example of a surplus of waste	The trash cans are always filled	I occupy a metropolitan where there is a city is invariably crowd.	Worried

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



S. No	Parameter	Description
1.	Problem Statement (Problem to be solved)	✓ The manual monitoring of wastes in trash cans is a laborious operation that requires additional time, money, and human labor ✓ Unsafe trash disposal is generating problems for people. ✓ Bad odor all around the place from uncollected trash or rubbish.
2.	Idea / Solution description	✓ This procedure uses a cloud connection and non-bio degradable wastes and an ultrasonic sensor to determine the level of a rubbish container ✓ By developing an app, the company of a certain neighborhood inside a large metropolis will be able to check the trash cans to see if they are full or not.
3.	Novelty / Uniqueness	✓ In contrast to the traditional ways for collecting trash cans, this strategy instructs us to utilize the transportation only when necessary. ✓ Keeping an eye on the trash cans easier and less labor-intensive for humans.
4.	Social Impact / Customer Satisfaction	✓ People can experience a clean atmosphere. ✓ Reduces the amount of labor required from humans for waste disposal. ✓ For a municipal corporation to monitor the cleanliness of different areas of the city, this proposal will be quite helpful.
5.	Business Model (Revenue Model)	✓ By cutting back on unneeded transportation costs to pointless locations, this lowers a significantamount of fuel costs for city businesses. ✓ This initiative intends to assistmunicipal corporation. ✓ Provide a sanitary atmosphere.

3.3 Proposed Solution

3.4 Problem Solution fit

Smart waste management system

Team ID PNT2022TMID42867

STEP 1

Problem Solving Cards

-Basic question

#Problem Statement

1.Whats most valuable to thecustomer?

2.Whatare we the bestat?

3.Whereare we looking to improve?

STEP 2

Framing Statements

STEP 3

Ideas

Problem Solution

Example ideas:

Previously there were numerousstrives on waste management andeducating people to dispose waste properly and as they fai ed to achievesignficant rests. we havefigured out thescopes that could be devdp. To sdvethis problem wehave designeda processthatensu1es proper disposal and efficient waste collection. The procedures we designed involves creative initiat ve that willinspe peopleto dump in designated area or bins and innovative method by using Decreasing Time algorithmorOTA for monitoringgarbage generation and collection of the garbage's.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Real time bin monitoring.	The Dashboard shows statistics on the amount of fill in bins as it is being tracked by smart sensors. The application also forecasts when the bin will fill up based on past data in addition to the percentage of fill level, which is one of the features that even the finest waste management software lacks. As picks are also recognized by the sensors, you can determine when thebin was last emptied. You can get rid of the overflowing bins and cease collecting half-empty ones using real-time data and forecasts.
FR-2	Eliminate inefficient picks.	Get rid of the collection of half-empty trash cans. Picks are recognized by sensors. We can demonstrate to you how full the bins you collect are using real-time data on fill-levels and pick recognition.
FR-3	Plan waste collection routes.	Route planning for rubbish pickup is semi-automated using the tool. You are prepared to act and arrange for garbage collection based on the levels of bin fill that are now present and forecasts of approaching capacity. To find any discrepancies, compare the planned and actual paths.
FR-4	Adjust bin distribution.	Ensure the best possible bin distribution. Determine which regions have a dense or sparse distribution of bins. Ensure that each form of waste has a representative stand. You can make any required adjustments to bin position or capacity based on past data.
FR-5	Expensive bins.	We assist you in locating containers that increase collection prices. The tool determines a collection cost rating for each bin. The tool takes local average depo-bin discharge into account. The tool determines the distance from depo-bin discharge and rates bins (1–10).
FR-6	Detailed bin inventory.	On the map, you can see every monitored bin and stand, and you can use Google Street View at any timeto visit them. On the map, bins or stands appear as green, orange, or red circles. The Dashboard displays information about each bin, including its capacity, trash kind, most recent measurement, GPS position, andpick-up schedule.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Usability is a unique and significant perspective to examine user needs, which may further enhance the design quality, according to IoT devices. Analyzing how well people interact with a product may help designers better understand customers' prospective demands for waste management, behavior, and experience in the design process when userexperience is at the Centre.
NFR-2	**Security**	Utilize recyclable bottles. Utilize reusable shopping bags. Spend responsibly and recycle Eat and drink in limited-use containers.
NFR-3	**Reliability**	Creating improved working conditions for garbage collectors and drivers is another aspect of smart waste management. Waste collectors will use their time more effectively by attending to bins that require service rather than travelling the same collection routes and servicing empty bins.
NFR-4	**Performance**	The Smart Sensors assess the fill levels in bins (alongwith other data) numerous times each day using ultrasonic technology. The sensors feed data to Senone's Smart Waste Management Software System, a robust cloud-based platform with data-driven daily operations and a waste management app, using a variety of IoT networks (NB-IoT, GPRS). As a consequence, customers receive data-driven decision-making services, and garbage collection routes, frequency, and truck loads are optimized, resulting in at least a 30% decrease in route length.
NFR-5	**Availability**	By creating and implementing robust hardware and gorgeous software, we enable cities, companies, and nations to manage garbage more intelligently.
NFR-6	**Scalability**	Using smart trash bins allows us to scale up and monitor the rubbish more efficiently while also reducing the number of bins needed in towns and cities.

4.2 Non-Functional requirements

Following are the non-functional requirements of the proposed solution.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirementgraphically.

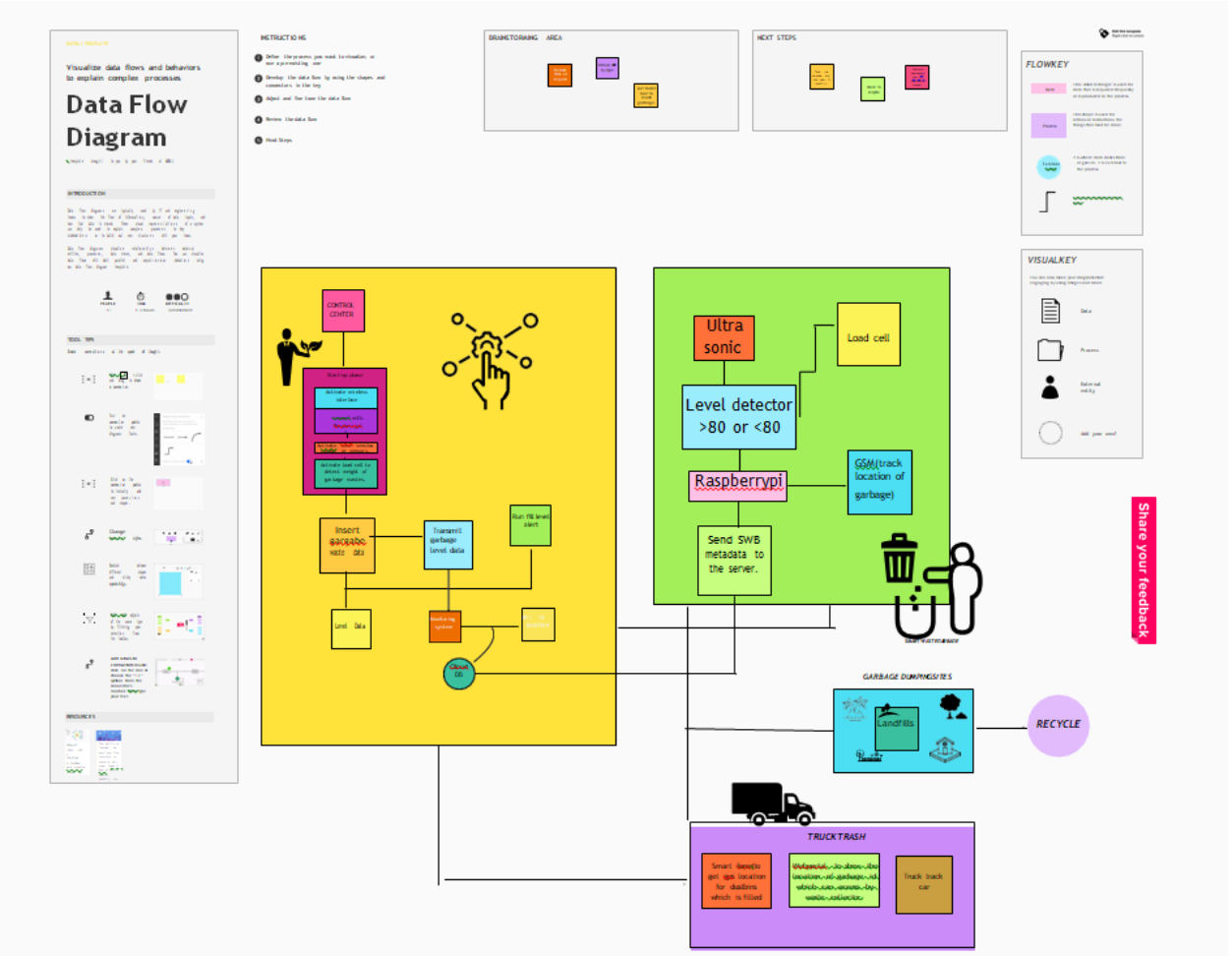
It shows how data enters and leaves the system, what changes the information, and where data is stored.

A smart waste management platform uses analytics to translate the data gather in your

bins into actionable insights to help you improve your waste services.
You can receive data on metric such as:

- The first test conducted is the situation where the garbage bin is empty or its garbage level is very low
- Then, the bin is filled with more garbage until its level has surpassed the first threshold **value, which is set to 80% then the first warning SMS is being sent, as depicted**
- The first notification SMS sent by the system, once the waste reaches the level of 85% full
- The second notification SMS sent by the system, indicating that bin is at least 95% full and **the garbage needs to be collected immediately**
- Locations prone to overflow
- The number of bins needed to avoid overflowing waste
- The number of collection services that could be saved
- The amount of fuel that could be saved
- The driving distance that could be saved.

5.2 Data flow diagram:



5.2 Solution & Technical Architecture:

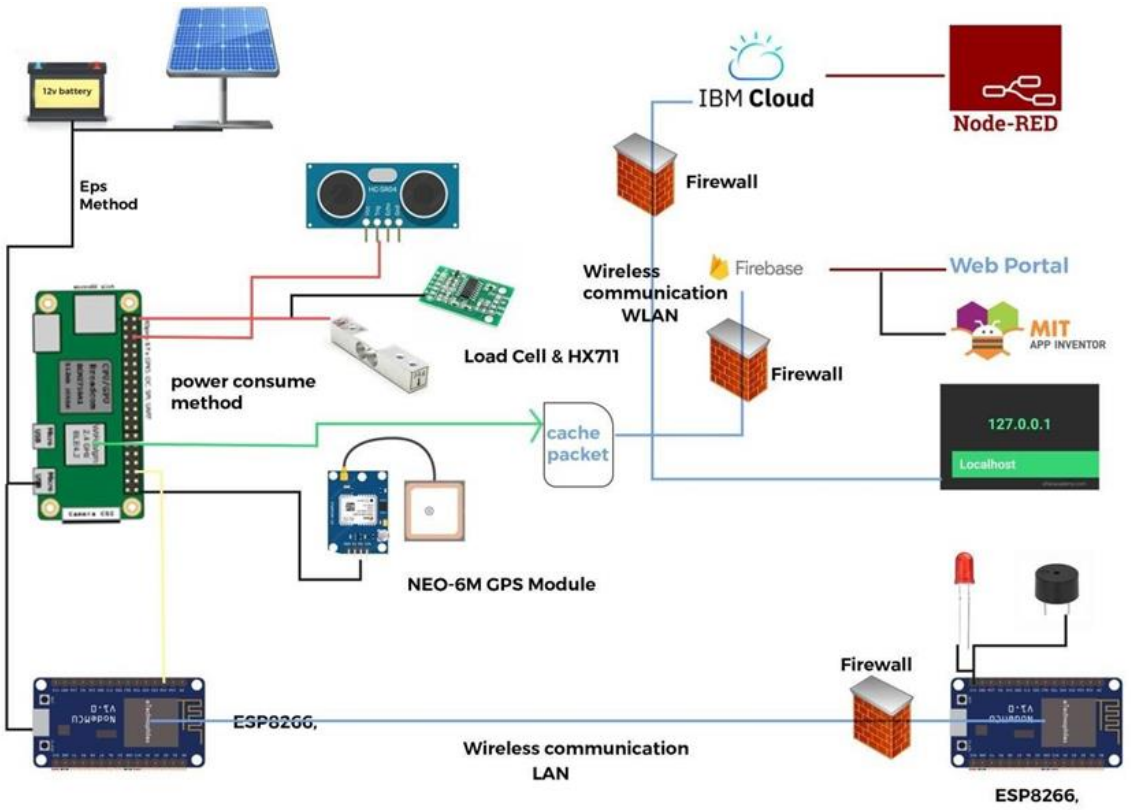


Table-1: Components & Technologies:

S.no	Component	Description	Technology
1.	User Interface	Mobile Application	HTML, CSS, JavaScript.
2.	Application Logic	Logic for a process in the application	Javascript
3.	Database	Data Type, Configurations etc.	Firebase, ibm cloud
4.	Cloud Database	Database Service on Cloud	IBM Cloud
5.	File Storage	File storage requirements	Local Filesystem and IBM cloud
6.	Infrastructure (Server / Cloud)	Application Deployment on CloudLocal Server Configuration	Local and Cloud Foundry

Table-2:Application Characteristics:

S.no	Characteristics	Description	Technology
1.	Open-Source Frameworks	GitHub	Internet hosting service
2.	Security Implementations	Application security: Veracode.	Network automation
3.	Scalable Architecture	It provides the room for expansion more databaseof smart bins added additionally can be updated.	Cloud storage
4.	Availability	As the system control is connected to web server itis available 24*7 and can be accessed whenever needed.	Server, Appleixe, reple
5.	Performance	Performance is high it uses 5mb caches	Wireless Sensor Network

5.3 User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Admin	Login	USN-1	As an administrator, I assigned user namesand passwords to each employee and managed them.	I can control my online account and dashboard.	Medium	Sprint-1
Co-Admin	Login	USN-2	As a Co-Admin, I'll control the waste level monitor. If a garbage filling alert occurs, I will notify the trash truck of the location and rubbish ID.	I can handle the waste collection.	High	Sprint-1
Truck Driver	Login	USN-3	As a Truck Driver, I'll follow Co Admin'sinstruction to reach the filled garbage.	I can take the shortest path to reach the waste filled route specified.	Medium	Sprint-2
Local Garbage Collector	Login	USN-4	As a Local Garbage Collector, I'II gather all the waste from the garbage, load it onto a garbage truck, and deliver it to Landfills	I can collect the trach, pullit to the truck, and send it out.	Medium	Sprint-3
Municipalit yofficer	Login	USN-5	As a Municipality officer, I'll make sure everything is proceeding as planned andwithout any problems.	All of these processes areunder my control.	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers,research publications etc.	28 SEPTEMBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements	24 SEPTEMBER 2022
Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	25 SEPTEMBER 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	23 SEPTEMBER 2022
Problem Solution Fit	Prepare problem - solution fit document.	30 SEPTEMBER 2022
Solution Architecture	Prepare solution architecture document.	28 SEPTEMBER 2022
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry to exit).	05 OCTOBER 2022
Functional Requirement	Prepare the functional requirement document.	11 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	12 OCTOBER 2022
Technology Architecture	Prepare the technology architecture diagram.	13 OCTOBER 2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	21 OCTOBER 2022
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Login	USN-1	As a Administrator, I need to give user id and passcode for ever workers over there in municipality	10	High	Gogul
Sprint-1	Login	USN-2	As a Co-Admin, I'll control the waste level by monitoring them vai real time web portal. Once the filling happens, I'll notify trash truck with location of bin with bin ID	10	High	Sathish
Sprint-2	Dashboard	USN-3	As a Truck Driver, I'll follow Co-Admin's Instruction to reach the filling bin in short roots and save time	20	Low	Shree Vikash
Sprint-3	Dashboard	USN-4	As a Local Garbage Collector, I'll gather all the waste from the garbage, load it onto a garbage truck, and deliver it to Landfills	20	Medium	Sreejith
Sprint-4	Dashboard	USN-5	As a Municipality officer, I'll make sure everything is proceeding as planned and without any problems	20	High	Naveen

6.2. Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

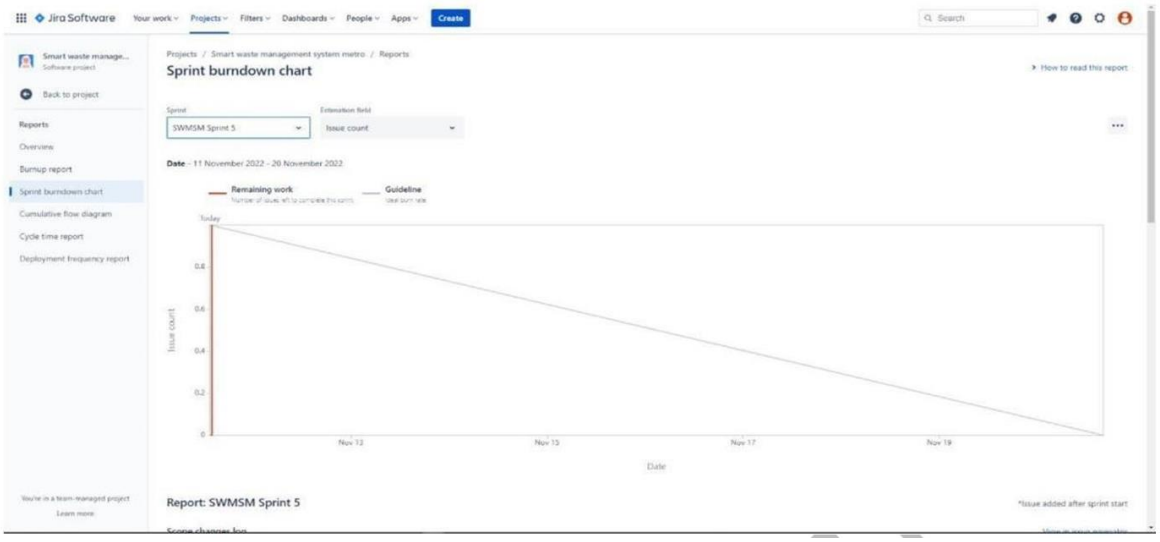
Velocity:

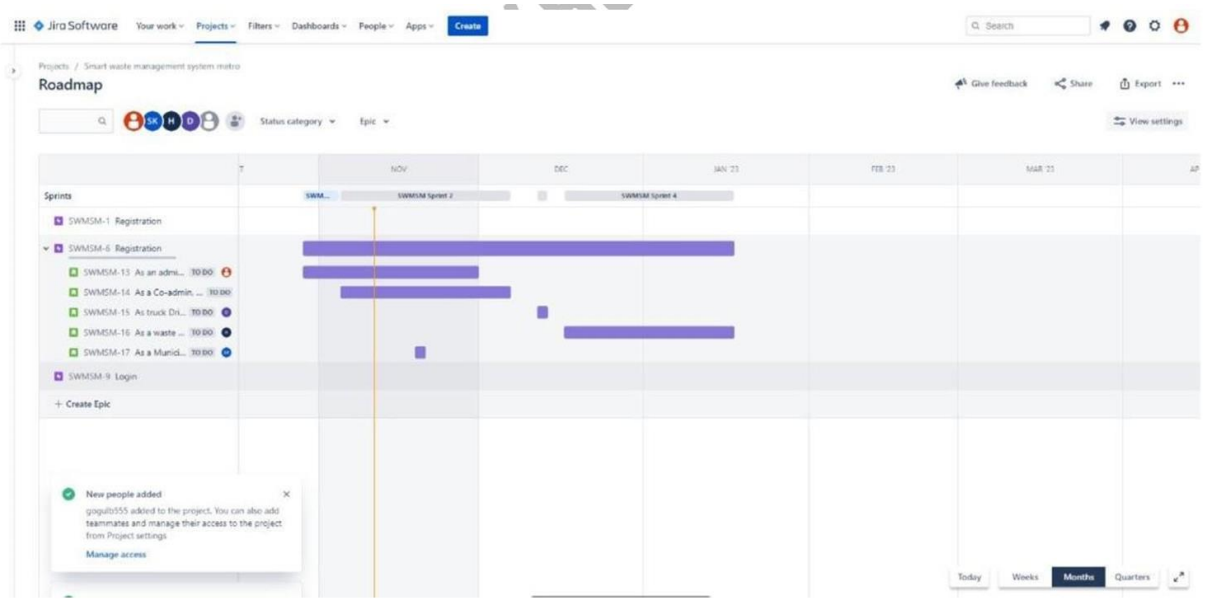
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

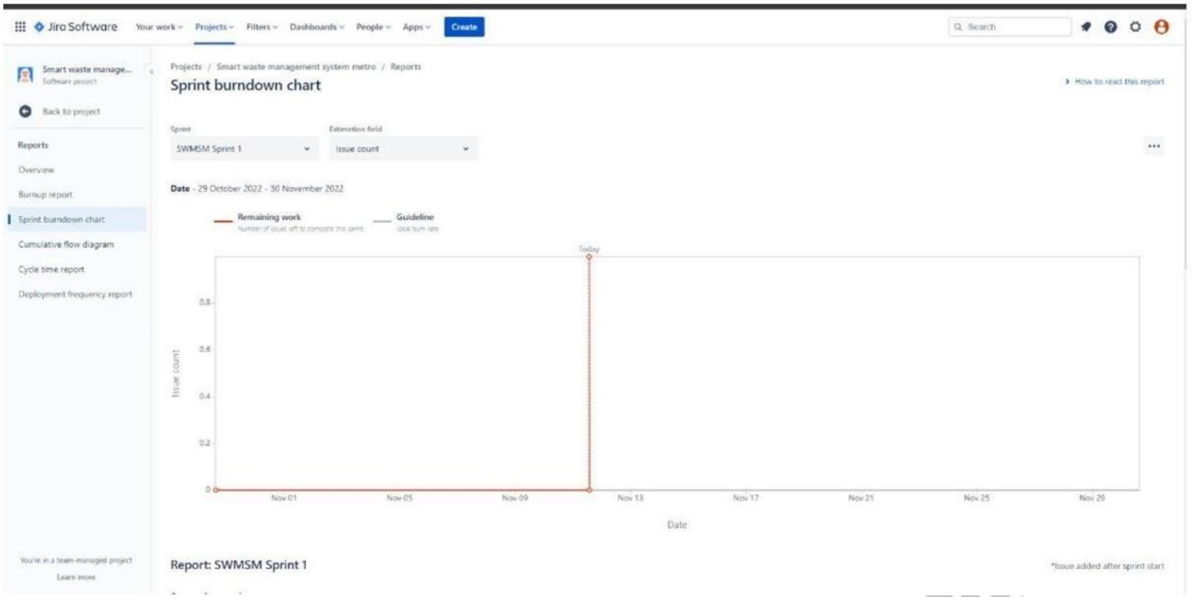
$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

6.3 Reports from JIRA

BURNDOWN CHART

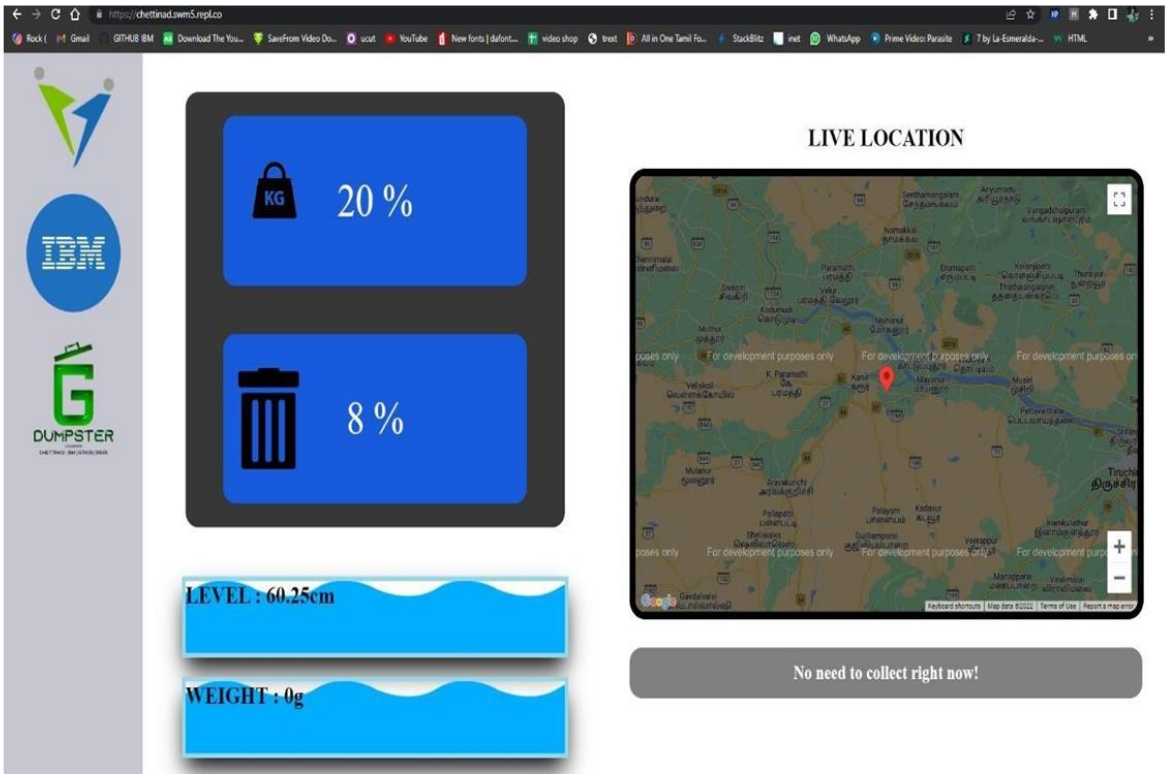




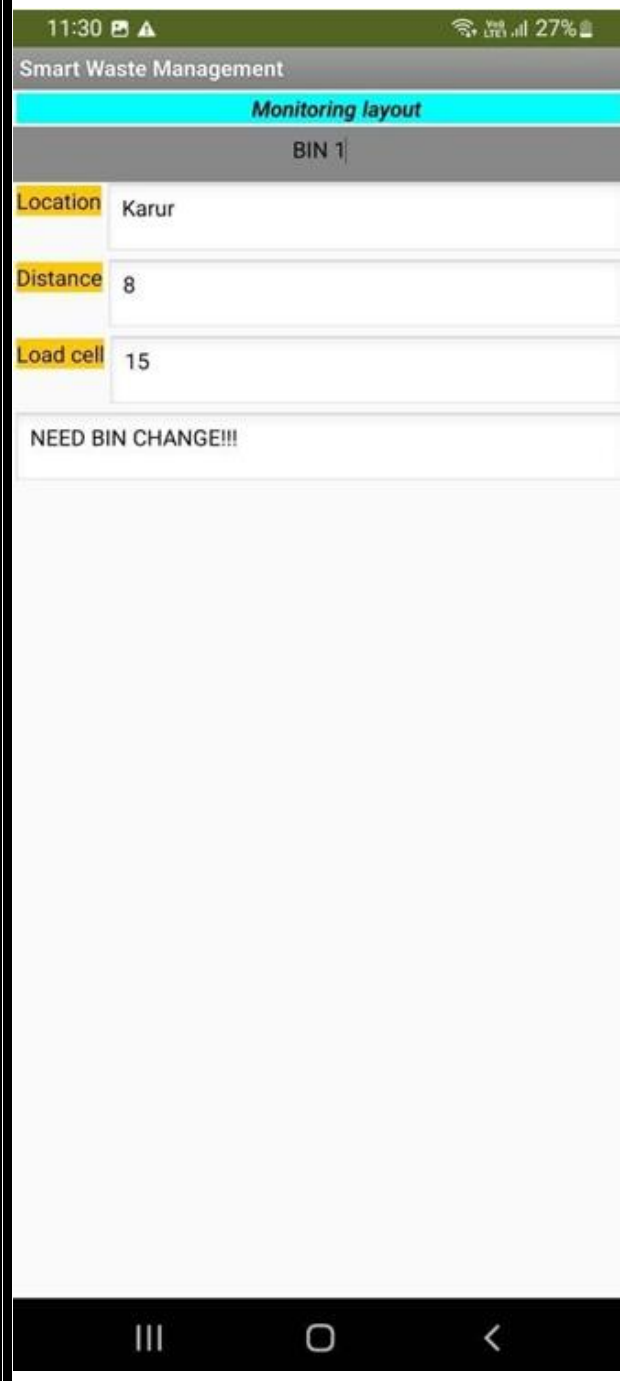

Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Smart waste manage...
Software project
Back to project
Reports
Overview
Burnup report
Sprint burndown chart
Cumulative flow diagram
Cycle time report
Deployment frequency report
You're in a team-managed project
Learn more
Projects / Smart waste management system metro / Reports
Sprint burndown chart
How to read this report
Sprint
SWMSM Sprint 1
Estimation field
Issue count
Date - 29 October 2022 - 30 November 2022
Remaining work
Guideline
Today
Issue count
0
0.2
0.4
0.6
0.8
Nov 01
Nov 05
Nov 09
Nov 13
Nov 17
Nov 21
Nov 25
Nov 29
Date
Report: SWMSM Sprint 1
*Issue added after sprint start

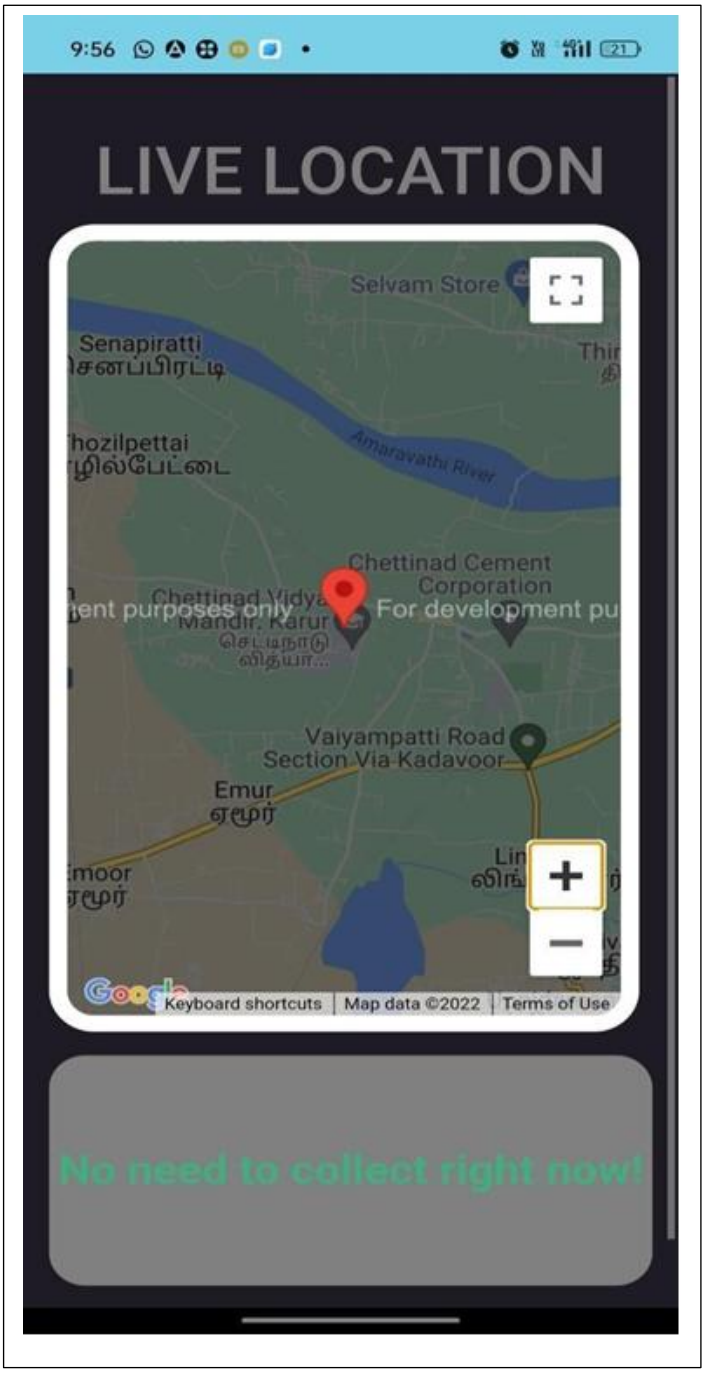
7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1- LOCATION TRACKER



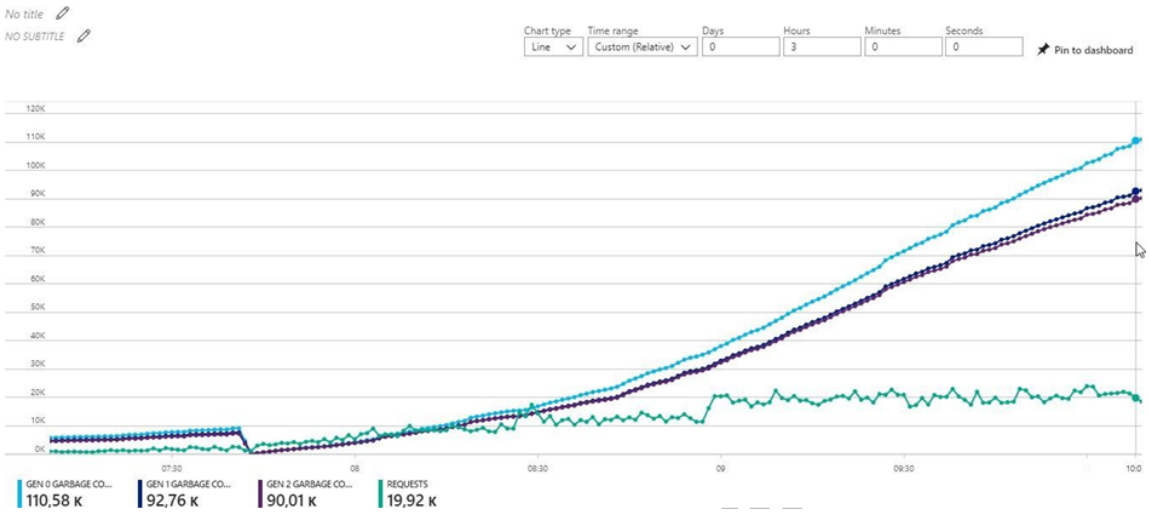
7.2 Feature 2- LIVE UPDATE ON COLLECTED DATA

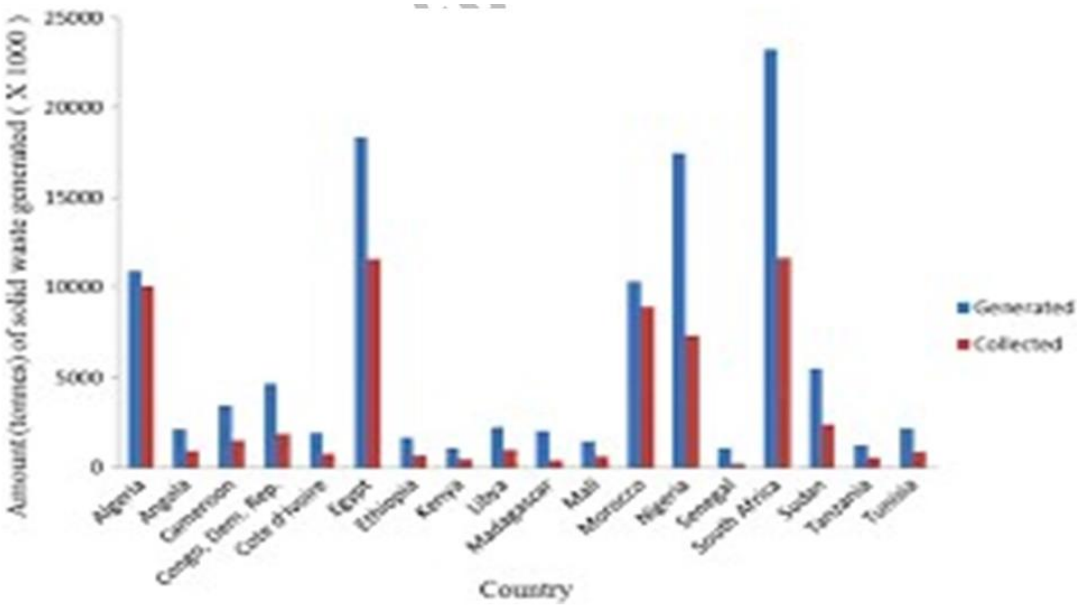




8. RESULTS & TESTING

8.1 Performance Metrics





9. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- Reduction in Collection Cost
- No Missed Pickups
- Reduced Overflows
- Waste Generation Analysis
- CO2 Emission Reduction

DISADVANTAGES:

- System requires a greater number of waste bins for separate waste collection as per population in the city.
- This results into high initial cost due to expensive smart dustbins compare to other methods.
- Sensor nodes used in the dustbins have limited memory size.

10. CONCLUSION

A Smart Waste Management system that is more effective than the one in use now is achievable by using sensors to monitor the filling of bins. Our conception of a "smart waste management system" focuses on monitoring waste management, offering intelligent technology for waste systems, eliminating human intervention, minimizing human time and effort, and producing a healthy and trash-free environment. The suggested approach can be implemented in smart cities where residents have busy schedules that provide little time for garbage management. If desired, the bins might be put into place in a metropolis where a sizable container would be able to hold enough solid trash for a single unit. The price might be high.

11. FUTURE SCOPE

There are several future works and improvements for the proposed system, including the following:

1. Change the system of user authentication and atomic lock of bins, which would aid in protecting the bin from damage or theft.

2. The concept of green points would encourage the involvement of residents or end users, making the idea successful and aiding in the achievement of collaborative waste management efforts, thus fulfilling the idea of Swachh Bharath.

3. Having case study or data analytics on the type and times waste is collected on different days or seasons, making bin filling predictable and removing the reliance on electronic components, and fixing the coordinates.

4. Improving the Server's and Android's graphical interfaces

12) APPENDIX

Source Code

 # Project : Smart Waste Management # Team ID : PNT2022TMID42867

MAIN.py

```
c = 1
import time
for i in range(1,2):
while True:
if c == 1:
import distance
d=distance.distancesensor()
c = 2
elif c == 2:
import load
w = int(load.loop())
c = 3
else:
import database as db
if w < 5000 and w > 4000:
load = "90 %"
elif w < 4000 and w > 3000:
load = "60 %"
elif w < 3000 and w > 100:
load = "40 %"
else:
load = "0 %"
if d > 30:
distance = "90 %"
elif d < 30 and d >20:
distance = "60 %"
elif d < 20 and d > 5:
distance = "40 %"
else:
distance = "7 %"
if load == "90 %" or distance == "90 %":
m = "Risk Warning: Dumpster poundage getting high, Time to collect :)"
elif load == "60 %" or distance == "60 %":
m ="dumpster is above 60%"
else :
m = "          "
db.database(d,w,m,load,distance)
print("data pushed")
c = 1
break
```

LOAD.py

```
import
time
import sys


EMULATE_HX711=False
```

```
referenceUnit = 1


if not EMULATE_HX711:
import RPi.GPIO as GPIO
from hx711 import HX711
else:
from emulated_hx711 import HX711


def cleanAndExit():
print("Cleaning...")


if not EMULATE_HX711:
GPIO.cleanup()
print("Bye!")
sys.exit()


hx = HX711(5, 6)


# I've found out that, for some reason, the order of the bytes is not always the same between versions of python, numpy and the hx711 itself.
# Still need to figure out why does it change.
# If you're experiencing super random values, change these values to MSB or LSB until to get more stable values.
# There is some code below to debug and log the order of the bits and the bytes.
# The first parameter is the order in which the bytes are used to build the "long" value.
# The second paramter is the order of the bits inside each byte.
 # According to the HX711 Datasheet, the second parameter is MSB so you shouldn't need to modify it.
hx.set_reading_format("MSB", "MSB")
```

```
# HOW TO CALCULATE THE REFFERENCE UNIT
# To set the reference unit to 1. Put 1kg on your sensor or anything you have and know exactly how much it weights.
# In this case, 92 is 1 gram because, with 1 as a reference unit I got numbers near 0 without any weight
# and I got numbers around 184000 when I added 2kg. So, according to the rule of thirds:
# If 2000 grams is 184000 then 1000 grams is 184000 / 2000 = 92.
hx.set_reference_unit(113)
#hx.set_reference_unit(referenceUnit)


hx.reset()


hx.tare()


print("Tare done! Add weight now...")


# to use both channels, you'll need to tare them both
#hx.tare_A()
#hx.tare_B()


def loop():
try:
# These three lines are usefull to debug wether to use MSB or LSB in the reading formats
# for the first parameter of "hx.set_reading_format("LSB", "MSB")".
# Comment the two lines "val = hx.get_weight(5)" and "print val" and uncomment these three lines to see what it prints.
# np_arr8_string = hx.get_np_arr8_string()
# binary_string = hx.get_binary_string()
# print binary_string + " " + np_arr8_string
# Prints the weight. Comment if you're debbuging the MSB and LSB issue.
val = hx.get_weight(5)
print(val)
return val


# To get weight from both channels (if you have load cells hooked up
# to both channel A and B), do something like this
#val_A = hx.get_weight_A(5)
#val_B = hx.get_weight_B(5)
#print "A: %s  B: %s" % ( val_A, val_B )
```

```
        hx.power_down()
        hx.power_up()
        time.sleep(0.1)


        except (KeyboardInterrupt, SystemExit):
        cleanAndExit()
```

DISTANCE.py

```
import RPi.GPIO as GPIO
                                                    import time


                                                    def distancesensor():
                                                    try:


                                                    GPIO.setmode(GPIO.BOARD)
                                                    GPIO.setwarnings(False)
                                                    PIN_TRIGGER = 23
                                                    PIN_ECHO = 33
                                                    GPIO.setup(PIN_TRIGGER, GPIO.OUT)
                                                    GPIO.setup(PIN_ECHO, GPIO.IN)
                                                    GPIO.output(PIN_TRIGGER, GPIO.LOW)


                                                    time.sleep(2)
                                                    GPIO.output(PIN_TRIGGER, GPIO.HIGH)
                                                    time.sleep(0.00001)
                                                    GPIO.output(PIN_TRIGGER, GPIO.LOW)

                                                    pulse_start_time = time.time()
                                                    while GPIO.input(PIN_ECHO)==1:
                                                    pulse_end_time = time.time()
                                                    pulse_duration = pulse_end_time - pulse_start_time
                                                    global distance
                                                    distance = round(pulse_duration * 17150, 2)
                                                    print(distance)
                                                          while
                                                    GPIO.input(PI
                                                    N_ECHO)==0:


                                                    finally:
                                                    GPIO.cleanup()
```

HX711.py

```
import RPi.GPIO as
GPIO
```

```
import time
import threading
class HX711:


def __init_(self, dout, pd_sck, gain=128):
self.PD_SCK = pd_sck


self.DOUT = dout


# Mutex for reading from the HX711, in case multiple threads in
client
# software try to access get values from the class at the same time.
self.readLock = threading.Lock()
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(self.PD_SCK, GPIO.OUT)
GPIO.setup(self.DOUT, GPIO.IN)


self.GAIN = 0


# The value returned by  the hx711 that corresponds to your
reference
# unit AFTER dividing by the SCALE.
self.REFERENCE_UNIT = 1
self.REFERENCE_UNIT_B = 1


self.OFFSET = 1
self.OFFSET_B = 1
self.lastVal = int(0)


self.DEBUG_PRINTING = False


self.byte_format = 'MSB'
self.bit_format = 'MSB'


self.set_gain(gain)
```

```
# Think about whether this is necessary.
time.sleep(1)


def convertFromTwosComplement24bit(self, inputValue):
return -(inputValue & 0x800000) + (inputValue & 0x7fffff)


def is_ready(self):
return GPIO.input(self.DOUT) == 0


def set_gain(self, gain):
if gain is 128:
self.GAIN = 1
elif gain is 64:
self.GAIN = 3
elif gain is 32:
self.GAIN = 2


GPIO.output(self.PD_SCK, False)


# Read out a set of raw bytes and throw it away.
self.readRawBytes()


def get_gain(self):
if self.GAIN == 1:
return 128
if self.GAIN == 3:
return 64
if self.GAIN == 2:
return 32


# Shouldn't get here.
return 0


def readNextBit(self):
# Clock HX711 Digital Serial Clock (PD_SCK).  DOUT will be
# ready 1us after PD_SCK rising edge, so we sample after
# lowering PD_SCL, when we know DOUT will be stable.
```

```
GPIO.output(self.PD_SCK, True)
GPIO.output(self.PD_SCK, False)
value = GPIO.input(self.DOUT)

# Convert Boolean to int and return it.
return int(value)



def readNextByte(self):
byteValue = 0

# Read bits and build the byte from top, or bottom, depending
# on whether we are in MSB or LSB bit mode.
for x in range(8):
if self.bit_format == 'MSB':
byteValue <<= 1
byteValue |= self.readNextBit()
else:
byteValue >>= 1
byteValue |= self.readNextBit() * 0x80

# Return the packed byte.
return byteValue

def readRawBytes(self):
# Wait for and get the Read Lock, incase another thread is already
# driving the HX711 serial interface.
self.readLock.acquire()

# Wait until HX711 is ready for us to read a sample.
while not self.is_ready():
pass

# Read three bytes of data from the HX711.
firstByte  = self.readNextByte()
secondByte = self.readNextByte()
thirdByte  = self.readNextByte()
```

```
# HX711 Channel and gain factor are set by number of bits read
# after 24 data bits.
for i in range(self.GAIN):
# Clock a bit out of the HX711 and throw it away.
self.readNextBit()


# Release the Read Lock, now that we've finished driving the
HX711
# serial interface.
self.readLock.release()


# Depending on how we're configured, return an orderd list of raw
byte
# values.
if self.byte_format == 'LSB':
return [thirdByte, secondByte, firstByte]
else:
return [firstByte, secondByte, thirdByte]
def read_long(self):
# Get a sample from the HX711 in the form of raw bytes.
dataBytes = self.readRawBytes()
if self.DEBUG_PRINTING:
print(dataBytes,)
# Join the raw bytes into a single 24bit 2s complement value.
twosComplementValue = ((dataBytes[0] << 16) |
(dataBytes[1] << 8)  |
dataBytes[2])


if self.DEBUG_PRINTING:
print("Twos: 0x%06x" % twosComplementValue)
# Convert from 24bit twos-complement to a signed value.
signedIntValue                                                =
self.convertFromTwosComplement24bit(twosComplementValue)


# Record the latest sample value we've read.
self.lastVal = signedIntValue


# Return the sample value we've read from the HX711.
return int(signedIntValue)
def read_average(self, times=3):
```

```
# Make sure we've been asked to take a rational amount of samples.
if times <= 0:
raise ValueError("HX711()::read_average(): times must >= 1!!")


# If we're only average across one value, just read it and return it.
if times == 1:
return self.read_long()


# If we're averaging across a low amount of values, just take the
# median.
if times < 5:
return self.read_median(times)


# If we're taking a lot of samples, we'll collect them in a list, remove
# the outliers, then take the mean of the remaining set.
valueList = []


for x in range(times):
valueList += [self.read_long()]


valueList.sort()


# We'll be trimming 20% of outlier samples from top and bottom of collected set.
trimAmount = int(len(valueList) * 0.2)


# Trim the edge case values.
valueList = valueList[trimAmount:-trimAmount]


# Return the mean of remaining samples.
return sum(valueList) / len(valueList)
# A median-based read method, might help when getting random value spikes
# for unknown or CPU-related reasons
def read_median(self, times=3):
if times <= 0:
```

```
raise ValueError("HX711::read_median(): times must be greater than zero!")
# If times == 1, just return a single reading.
if times == 1:
return self.read_long()
valueList = []
for x in range(times):
valueList += [self.read_long()]
valueList.sort()

# If times is odd we can just take the centre value.
if (times & 0x1) == 0x1:
return valueList[len(valueList) // 2]
else:
# If times is even we have to take the arithmetic mean of
# the two middle values.
midpoint = len(valueList) / 2
return sum(valueList[midpoint:midpoint+2]) / 2.0
# Compatibility function, uses channel A version
def get_value(self, times=3):
return self.get_value_A(times)
def get_value_A(self, times=3):
return self.read_median(times) - self.get_offset_A()
def get_value_B(self, times=3):
# for channel B, we need to set_gain(32)
g = self.get_gain()
self.set_gain(32)
value = self.read_median(times) - self.get_offset_B()
self.set_gain(g)
return value
# Compatibility function, uses channel A version
def get_weight(self, times=3):
return self.get_weight_A(times)
def get_weight_A(self, times=3):
value = self.get_value_A(times)
value = value / self.REFERENCE_UNIT
return value
def get_weight_B(self, times=3):
value = self.get_value_B(times)
value = value / self.REFERENCE_UNIT_B
return value
# Sets tare for channel A for compatibility purposes
def tare(self, times=15):
return self.tare_A(times)
def tare_A(self, times=15):
# Backup REFERENCE_UNIT value
```

```
backupReferenceUnit = self.get_reference_unit_A()
self.set_reference_unit_A(1)
value = self.read_average(times)


if self.DEBUG_PRINTING:
print("Tare A value:", value)
self.set_offset_A(value)


# Restore the reference unit, now that we've got our offset.
self.set_reference_unit_A(backupReferenceUnit)
return value
def tare_B(self, times=15):
# Backup REFERENCE_UNIT value
backupReferenceUnit = self.get_reference_unit_B()
self.set_reference_unit_B(1)


# for channel B, we need to set_gain(32)
backupGain = self.get_gain()
self.set_gain(32)


value = self.read_average(times)
if self.DEBUG_PRINTING:
print("Tare B value:", value)
self.set_offset_B(value)
# Restore gain/channel/reference unit settings.
self.set_gain(backupGain)
self.set_reference_unit_B(backupReferenceUnit)
return value
def set_reading_format(self, byte_format="LSB",
bit_format="MSB"):
if byte_format == "LSB":
self.byte_format = byte_format
elif byte_format == "MSB":
self.byte_format = byte_format
else:
raise ValueError("Unrecognised byte_format: \"%s\"" %
byte_format)
if bit_format == "LSB":
self.bit_format = bit_format
elif bit_format == "MSB":
self.bit_format = bit_format
else:
```

```
raise ValueError("Unrecognised bitformat: \"%s\"" % bit_format)
# sets offset for channel A for compatibility reasons
def set_offset(self, offset):
self.set_offset_A(offset)


def set_offset_A(self, offset):
self.OFFSET = offset
def set_offset_B(self, offset):
self.OFFSET_B = offset


def get_offset(self):
return self.get_offset_A()


def get_offset_A(self):
return self.OFFSET


def get_offset_B(self):
return self.OFFSET_B
def set_reference_unit(self, reference_unit):
self.set_reference_unit_A(reference_unit)
def set_reference_unit_A(self, reference_unit):
# Make sure we aren't asked to use an invalid reference unit.
if reference_unit == 0:
raise ValueError("HX711::set_reference_unit_A() can't accept 0 as a reference unit!")
return


self.REFERENCE_UNIT = reference_unit
def set_reference_unit_B(self, reference_unit):
# Make sure we aren't asked to use an invalid reference unit.
if reference_unit == 0:
raise ValueError("HX711::set_reference_unit_A() can't accept 0 as a reference unit!")
return
self.REFERENCE_UNIT_B = reference_unit
def get_reference_unit(self):
return get_reference_unit_A()
def get_reference_unit_A(self):
return self.REFERENCE_UNIT
def get_reference_unit_B(self):
return self.REFERENCE_UNIT_B
```

```
def power_down(self):
# Wait for and get the Read Lock, incase another thread is already
# driving the HX711 serial interface.
self.readLock.acquire()


# Cause a rising edge on HX711 Digital Serial Clock (PD_SCK). We then
# leave it held up and wait 100 us. After 60us the HX711 should be
# powered down.
GPIO.output(self.PD_SCK, False)
GPIO.output(self.PD_SCK, True)


time.sleep(0.0001)


# Release the Read Lock, now that we've finished driving the HX711
# serial interface.
self.readLock.release()
def power_up(self):
# Wait for and get the Read Lock, incase another thread is already
# driving the HX711 serial interface.
self.readLock.acquire()
# Lower the HX711 Digital Serial Clock (PD_SCK) line.
GPIO.output(self.PD_SCK, False)
# Wait 100 us for the HX711 to power back up.
time.sleep(0.0001)


# Release the Read Lock, now that we've finished driving the HX711
# serial interface.
self.readLock.release()
# HX711 will now be defaulted to Channel A with gain of 128.  If this
# isn't what client software has requested from us, take a sample and
# throw it away, so that next sample from the HX711 will be from the
# correct channel/gain.
if self.get_gain() != 128:
self.readRawBytes()
def reset(self):
```

```
                                                    self.power_down()
                                                    self.power_up()
```

WEBSITE CODING

Index.html

```
<!DOCTYPE html>
<html>

<head>
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width">
        <title>Garbage Management System</title>
        <link rel="icon" type="image/x-icon" href="/Images/DUMPSTER.png">
        <link href="style.css" rel="stylesheet" type="text/css" />
        <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>
        <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-database.js"></script>

        <script>
                var firebaseConfig =
                {
                        apiKey: "AIzaSyB9ysbnaWc3IyeCioh-aJQT_UCMd5CBFeU",
                        authDomain: "fir-test-923b4.firebaseapp.com",
                        databaseURL: "https://fir-test-923b4-default-rtdb.firebaseio.com",
                        projectId: "fir-test-923b4",
                        storageBucket: "fir-test-923b4.appspot.com",
                        messagingSenderId: "943542145393",
                        appId: "1:943542145393:web:9b5ec7593e6a3cbd7966d0",
                        measurementId: "G-BN7JNX1Q7B"
                };
                firebase.initializeApp(firebaseConfig)
        </script>
        <script defer src="database.js"></script>
</head>

<body style="background-color:#1F1B24;">
        <script src="map.js"></script>


                <div id="map_container">
                        <h1 id="live_location_heading" >LIVE LOCATION</h1>
                        <div id="map"></div>
                        <div id="alert_msg">ALERT MESSAGE!</div>
   </div>
        </div>
<center><a href="https://goo.gl/maps/G9XET5mzSw1ynHQ18"
  type="button" class="btn btn-dark">DUMPSTER</a></center>

        <script
                src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBBLyWj-3FWtCbCXGW3ysEiI2fDfrv2v0Q&callback=myMap"></script></div>
</body>

</html>
```

Database.js

```
const cap_status = document.getElementById('cap_status');
const alert_msg = document.getElementById('alert_msg');

var ref = firebase.database().ref();
```

```
ref.on("value", function(snapshot)
 {
   snapshot.forEach(function (childSnapshot) {
      var value = childSnapshot.val();

                        const alert_msg_val = value.alert;
                        const cap_status_val = value.distance_status;


                        alert_msg.innerHTML= `${alert_msg_val}`;
   });
 }, function (error) {
   console.log("Error: " + error.code);
 });
```

Map.js

```
const database = firebase.database();

function myMap()
{
        var ref1 = firebase.database().ref();

                ref1.on("value", function(snapshot)
                 {
                   snapshot.forEach(function (childSnapshot) {
                      var value = childSnapshot.val();
                                const latitude = value.latitude;
                                const longitude = value.longitude;

                                var latlong = { lat: latitude, lng: longitude}
                                var mapProp =
                                {
                                        center: new google.maps.LatLng(latlong),
                                        zoom: 10,
                                };
                                var map = new google.maps.Map(document.getElementById("map"), mapProp);

                                var marker = new google.maps.Marker({ position: latlong });
                                marker.setMap(map);
                   });
                 }, function (error) {
                   console.log("Error: " + error.code);
                 });

}
```

Style.css

```
html, body
        {
        height: 100%;
                margin: 0px;
                padding:0px;
        }
#container
{
        display: flex;
        flex-direction: row;
        height: 100%;
        width: 100%;
        position: relative;
}
#logo_container
{
        height: 100%;
        width: 12%;
        background-color: #C5C6D0;
        display: flex;
        flex-direction: column;
        vertical-align: text-bottom;
}
.logo
{
        width:70%;
```

```
		margin: 5% 15%;

/*		border-radius: 50%; */


}
#logo_3
{
		vertical-align: text-bottom;

}
#data_container
{
		height: 100%;
		width: 20%;
		margin-left: 1%;
		margin-right: 1%;
		display: flex;
		flex-direction: column;
}
#data_status
{
		height:60%;
		width:8%;
		margin:7%;
		background-color: #691F6E;
		display: flex;
		flex-direction: column;
		border-radius:20px;
}
#load_status
{
		background-image: url("/Images/KG.png");
		background-repeat: no-repeat;
		 background-size:     170px;
		background-position: left center;
}
#cap_status
{
		background-image: url("/Images/dust.png");
		background-repeat: no-repeat;
		 background-size: 150px;
		background-position: left center;
}
.status
{
		width: 80%;
		height: 40%;
		margin:5% 10%;
		background-color:#185adc;
		border-radius:20px;
		display: flex;
		justify-content: center;
		align-items: center;
		color: white;
		font-size: 60px;


}
.datas
{
		width:86%;
		margin:2.5% 7%;
		height:10%;
  background: url(water.png);
   background-repeat: repeat-x;
   animation: datas 10s linear infinite;

   box-shadow: 0 0 0 6px #98d7eb, 0  20px 35px rgba(0,0,0,1);

}
#map_container
{
		height: 100%;
		width: 100%;
		display: flex;
		flex-direction: column;
}
```

```
#live_location_heading
{
		margin-top:10%;
		text-align: center;
  color:  GREY;
}



#map
{
		height: 70%;
		width: 90%;
		margin-left: 4%;
		margin-right:4%;
		border: 10px solid white;
		border-radius: 25px;
}
#alert_msg
{
		width:92%;
		height:20%;
		margin:4%;
		background-color:grey;
		border-radius: 20px;
		display: flex;
		justify-content: center;
		align-items: center;
		color: #41af7f;
		font-size: 25px;
		font-weight: bold;
}
.lat
{
		margin: 0px;
		font-size:0px;
}




@keyframes datas{
   0%
   {
      background-position: -500px 100px;
   }
   40%
   {
      background-position: 1000px -10px;
   }

   80% {
      background-position: 2000px 40px;
   }
   100% {
      background-position: 2700px 95px;
   }

}
```

For simulator python code

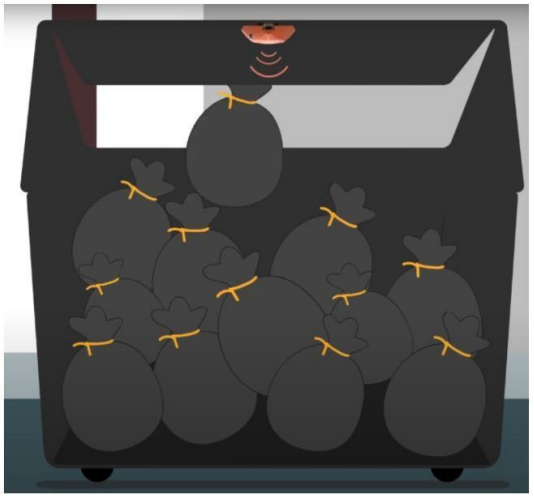
BIN1.PY

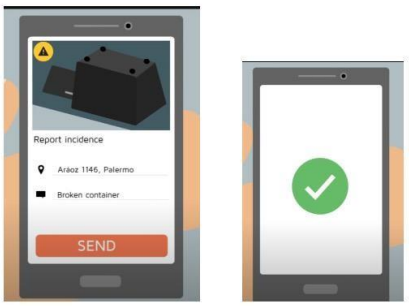
```
import requests
import json
import ibmiotf.application
import ibmiotf.device
import time
import random
import sys

# watson device details

organization = "4yi0vc"
devicType = "BIN1"
deviceId = "BIN1ID"
authMethod= "token"
authToken= "123456789"

#generate random values for randomo variables (temperature&humidity)



def myCommandCallback(cmd):
   global a
   print("command recieved:%s" %cmd.data['command'])
   control=cmd.data['command']
   print(control)

try:
      deviceOptions={"org": organization, "type": devicType,"id": deviceId,"auth-method":authMethod,"auth-
token":authToken}
      deviceCli = ibmiotf.device.Client(deviceOptions)
except Exception as e:
      print("caught exception connecting device %s" %str(e))
      sys.exit()

#connect and send a datapoint "temp" with value integer value into the cloud as a type of event for every 10 seconds
deviceCli.connect()

while True:

   distance= random.randint(10,70)
   loadcell= random.randint(5,15)
   data= {'dist':distance,'load':loadcell}

   if loadcell < 13 and loadcell > 15:
      load = "90 %"
   elif loadcell < 8 and loadcell > 12:
        load = "60 %"
```

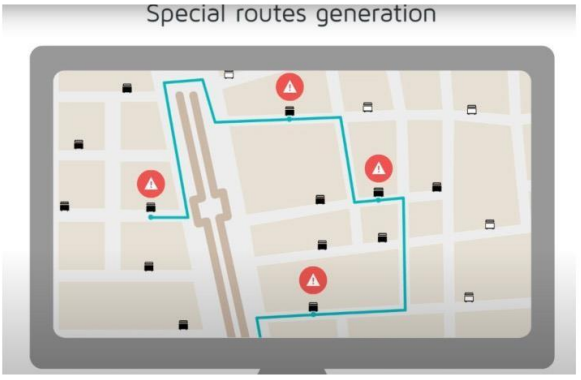
```
    elif loadcell < 4 and loadcell > 7:
        load = "40 %"
    else:
        load = "0 %"

    if distance < 15:
        dist = 'alert :' ' Dumpster poundage getting high, Time to collect :) ' '90 %'


    elif distance < 40 and distance >16:
        dist = 'alert :' 'dumpster is above '' 60%'

    elif distance < 60 and distance > 41:
        dist = 'alert :' 'dumpster is above ''40 %'
    else:
        dist ='alert :' 'No need to collect right now ''17 %'



    if load == "90 %" or distance == "90 %":
        warn  = 'alert pushed to ibm sucessfully :'

    elif load == "60 %" or distance == "60 %":

        warn = 'alert pushed to ibm sucessfully :'
    else :
        warn = 'alert pushed to ibm sucessfully :'
    def myOnPublishCallback(lat=10.678991,long=78.177731):
       print("Gandigramam, Karur")
       print("published distance = %s " %distance,"loadcell:%s " %loadcell,"lon = %s " %long,"lat = %s" %lat)
       print(load)
       print(dist)
       print(warn)

    time.sleep(4)
    success=deviceCli.publishEvent ("IoTSensor","json",warn,qos=0,on_publish= myOnPublishCallback)

    success=deviceCli.publishEvent ("IoTSensor","json",data,qos=0,on_publish= myOnPublishCallback)
    if not success:
       print("not connected to ibmiot")
    time.sleep(4)
    deviceCli.commandCallback=myCommandCallback
#disconnect the device
deviceCli.disconnect()
```

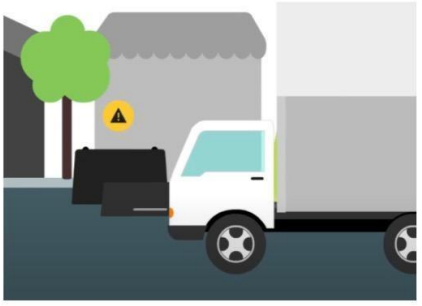
13.2 OUTPUT PICTURE

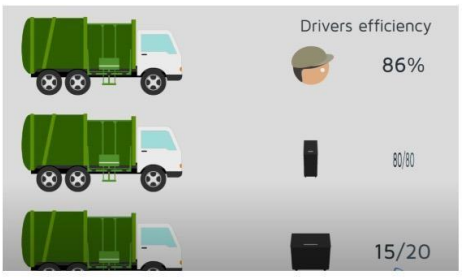
BENIFITS OF DUMPSTER











- WE CONNECT WITH YOUR ASSETS
- YOUR BUSSINESS BECOME EFFICIENT
- WE GIVE INFORMATION IN REAL TIME
- TO MAKE BETTER ANALYSIS FOR BIG DATA
- ALL CIVILIZED PERSONS RESPOSIBLITY TO KEEP WORLD CLEAN IS BASIC NEED

RECYCLE QUOTES :)

Website

Scan to install app

REAL TIME EXPERIENCE