KSR COLLGE OF ENGINEERING, TIRUCHENGODE

PROJECT TITLE: SMART WASTE MANAGEMENT SYSTEM FOR METROPOLITIAN CITIES

A NALAIYA THIRAN PROJECT REPORT SUBMITTED BY

TEAM LEADER : KISHORE D (73151915025)

TEAM MEMBER: LINGESH S (73151915027)

TEAM MEMBER: NAVEENA M (73151915038)

TEAM MEMBER: VIGNESH V (73151915072)

TEAM ID:PNT2022TMID12076

Project Report Format

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**

Project Report

Team ID	**PNT2022TMID12076**
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 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.

Paper 1

TITLE: Smart City Waste Management through ICT and IOT driven

AUTHOR NAME: Dipak.G , P.S.Aithal

PUBLICATION YEAR: 2021

DESCRIPTION: The growing population and mass relocation of citizens from urban and semi-urban areas to Smart Cities have resulted in exponential growth in Smart Cities and thereby certain challenges. One of the major challenges Smart Cities are facing is to control, manage and process waste generation on a daily basis. Waste collection and processing at a wider scale is not an easy job. The growing population and resource constraints in waste management activities are the primary reasons, which have made waste management a tough job. To deal with this challenging process, Smart Cities use Smart Waste Management System. (iSmartWMS).

PAPER 2:

TITLE: Household Waste Management System Using IoT

AUTHOR NAME: Pushpa Singh, Krishna Kant Singh

PUBLICATION YEAR: 2020

DESCRIPTION: This model discusses the collection and decomposition of waste in the smart way so that benefit from the waste is maximized and the actual waste is minimized efficiently. This paper focus on the segregation of the waste at two levels: the first level of segregation is on the individual house of the society and the second level of segregation is at the society. Author, discuss the recycling of the biodegradable waste for making compost. The machine learning technique such as KNN is used to generate an alert message for various combinations of three sensor values like level of bio and non-biodegradable waste, concentration of poisonous gas. The overall impact of this research is in the upliftment of the green technologies by reducing pollutants, conserving, resourcing and reusing the energy through the use of technology.

Paper 3:

TITLE: Smart Waste Collection System

AUTHOR NAME: Javed Ramzan; Muhammad Wasif Nisar

PUBLICATION YEAR: 2018

DESCRIPTION: This project named smart waste collection is need of today as there is no efficient waste collection system installed in the earth these days this system is to revolve the waste collection method of the advancing technological 21st century. This system is supported by an android app named "SWC" and firebase real-time data for more efficient user-friendly usage. where's cloud storage also makes it easier for storage of collection records providing the authorized origination to control manage and audit performance data. To compare several collectors, their configurations, and program behavior, we use an accurate simulator that models all heap objects and the pointers among them, but does not model cache or other memory effects. For object-oriented languages, our results demonstrate that an older-first collector, which collects older objects before the youngest ones, copies on average much less data than generational collectors. More importantly, we reopen for consideration the question where in the heap and with which policies copying collectors will achieve their best performance.

Paper 4:

TITLE: Smart Garbage Management System

AUTHOR NAME: Akshat Mishra, Sushmit Mehta, Vivek Solvande

PUBLICATION YEAR: 2018

DESCRIPTION: The proposed system monitors the garbage bin. While monitoring the garbage bin it sends the notification to the authority about the level of garbage filled. If the lower authority ignores the notification, the next notification goes to the higher authority. The proposed system will help them to actually know that where and when to go to collect the garbage. The proposed system manages the effort to check the area by visiting there. The proposed project is quite helpful for both the Brihan Mumbai Municipal Corporation (BMC) and the citizens in that area by time-to time interaction between Brihan Mumbai Municipal Corporation (BMC) and the proposed system. Hence the proposed system makes a better way to manage garbage.

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.

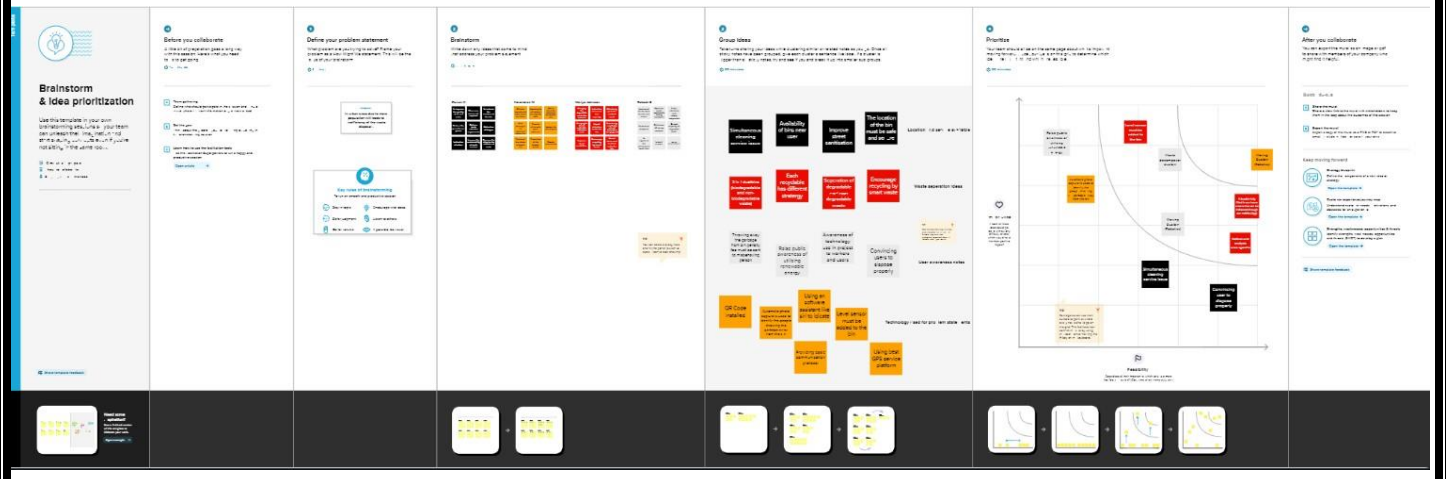
2.2 Problem Statement Definition:

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A small waste garbage collector.	Use the technology anddata to create amore efficient garbage waste. Based on IOT & Cloud.	Some drivers collect empty bins, predefined collection routes of the system causewaste of time, an increase in fuel consumption , and excessiveuse of resources.	The smart waste garbage collector is a specially designed method to dispose the garbage in a smart way.	When it solves the socialissues of hygiene in the country.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

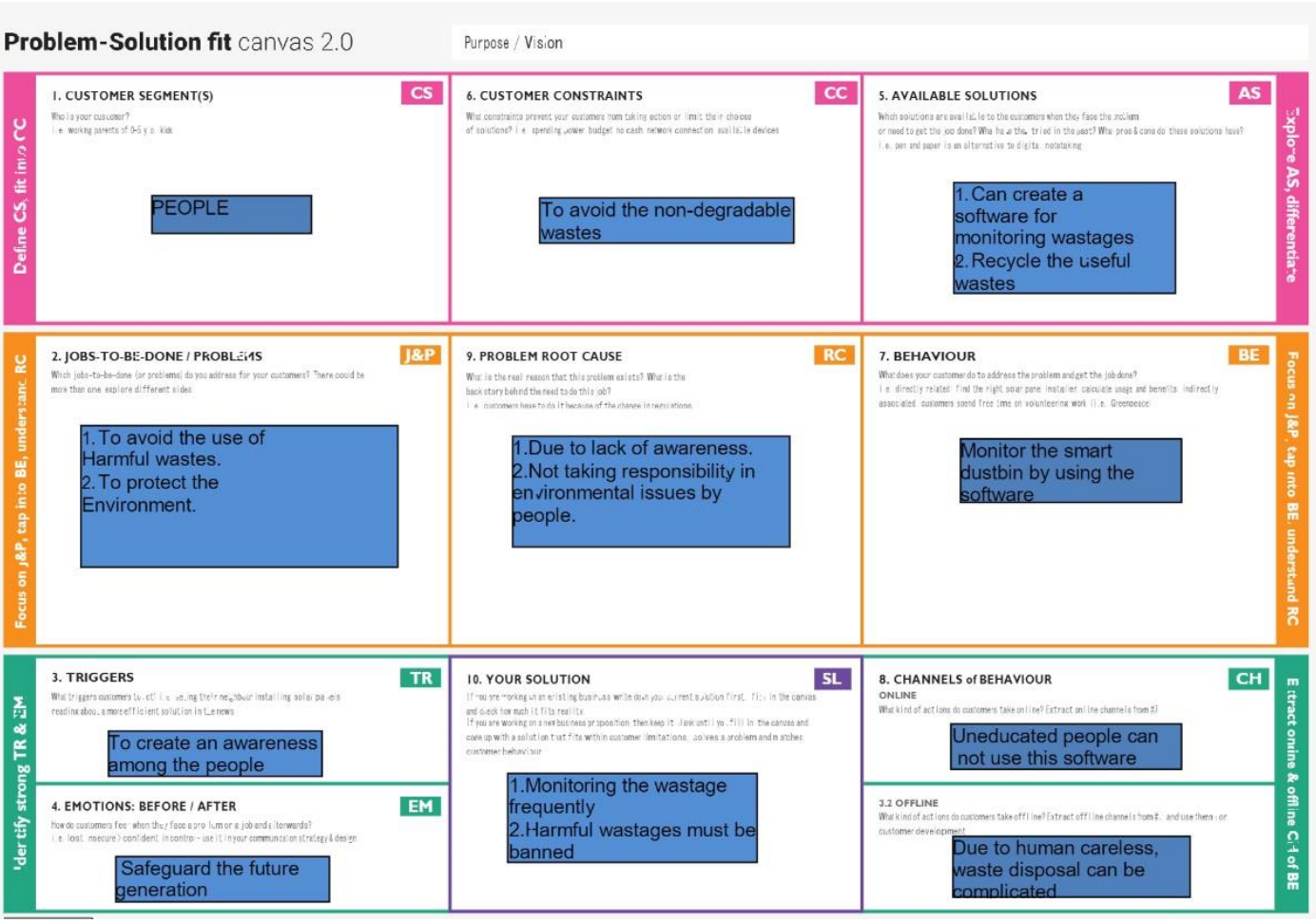


3.3 Proposed Solution

t

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To maintain waste in the metropolitan cities and to maintain the clean environment which is polluted by the wastes and to dispose wastes.
2.	Idea / Solution description	-Software can be implemented -Fine can be implemented -Create awareness
3.	Novelty / Uniqueness	This system provides awareness among the people and safeguard the environment with the help of people.
4.	Social Impact / Customer Satisfaction	Clean cities Healthy environment Peaceful life
5.	Business Model (Revenue Model)	-Offering software as a servicemodel to government. -Making use of useful wastes and making it us money.
6.	Scalability of the Solution	To help government to maintain clean environment.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)
FR-1	Taking sensor reading from theSensor Circuit.
FR-2	Pushing the data to a MySQL database.
FR-3	Retrieving information from database for Calculation garbage bin which fulfils the condition for garbage collection, example: Collect garbage from bins whose levelis over 80% of bin.
FR-4	A client side script to getGarbage collection live Monitoring.

4.2 Non-Functional requirements

Following are the non-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 desi process when user experience is at the Centre.
NFR-2	**Security**	Utilize recyclable bottles. Utilize reusable shoppingbags Spend responsibly and recycle
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** Eat and drink in limited-use containers.	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 network (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.

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:

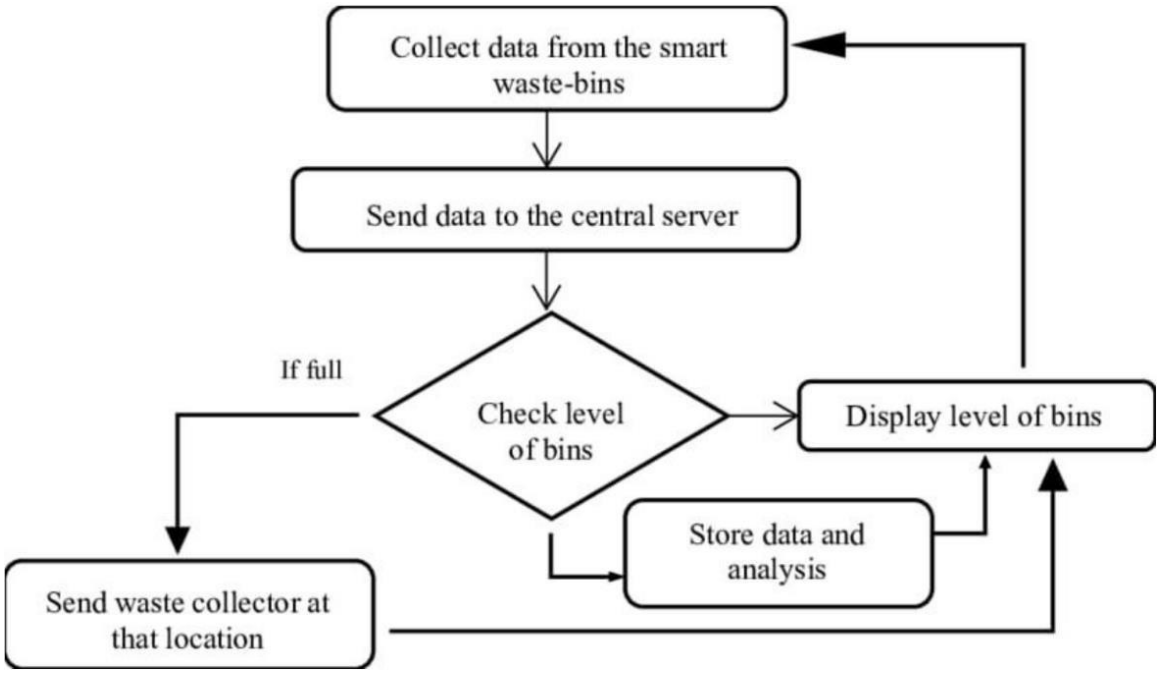
Data Flow Diagrams:

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 requirement graphically.

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 Solution & Technical Architecture:

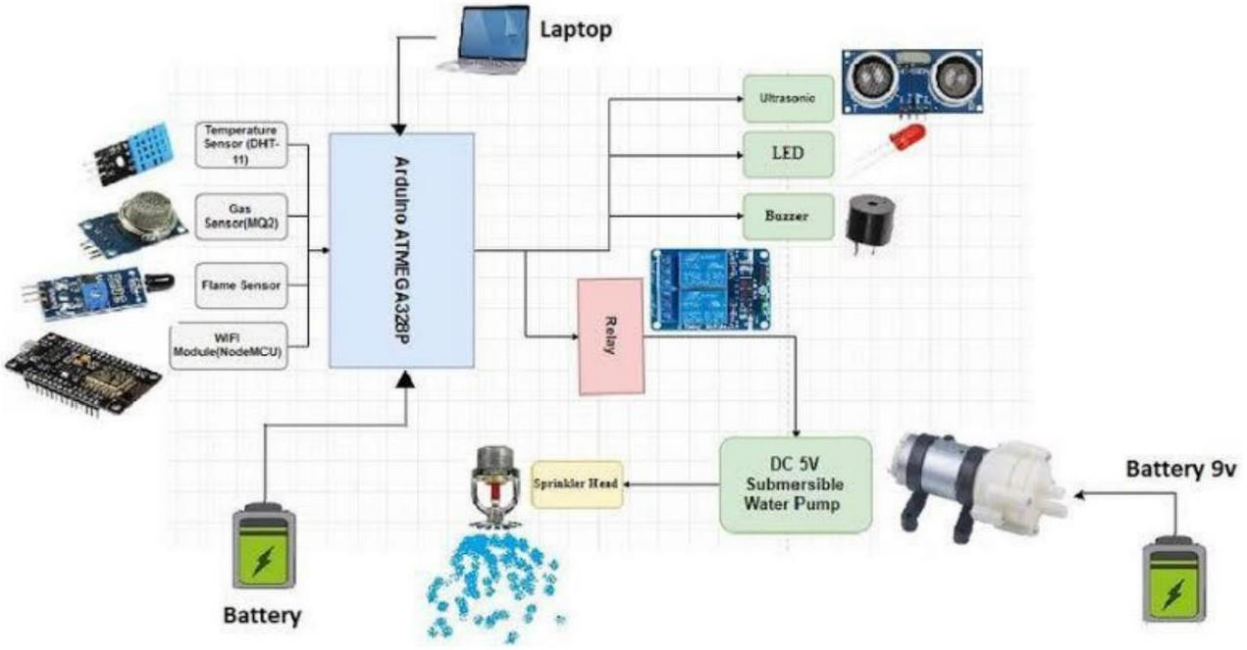


Table-1: Components & Technologies:

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

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	Kishore
Sprint-1	Login	USN-2	As a Co-Admin, I'll control the waste level by monitoring them real time web portal. Oncethe filling happens, I'll notify trash truck with location of bin with bin ID	10	High	Naveena
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	Lingesh
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	Vignesh
Sprint-4	Dashboard	USN-5	As a Municipality officer, I'll make sure everything is proceeding as planned and without any problems	20	High	Kishore

6.2. Sprint Delivery Schedule

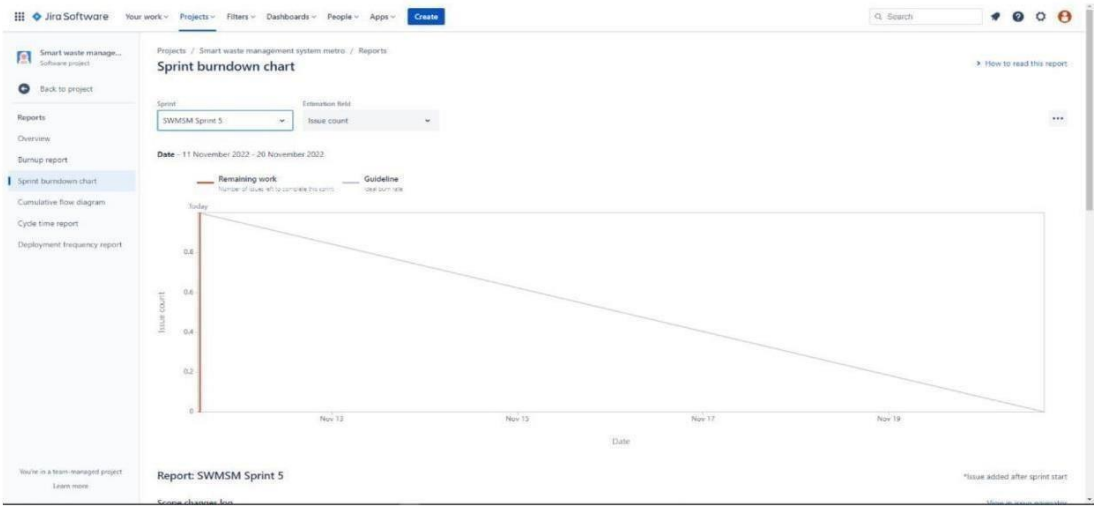
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022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

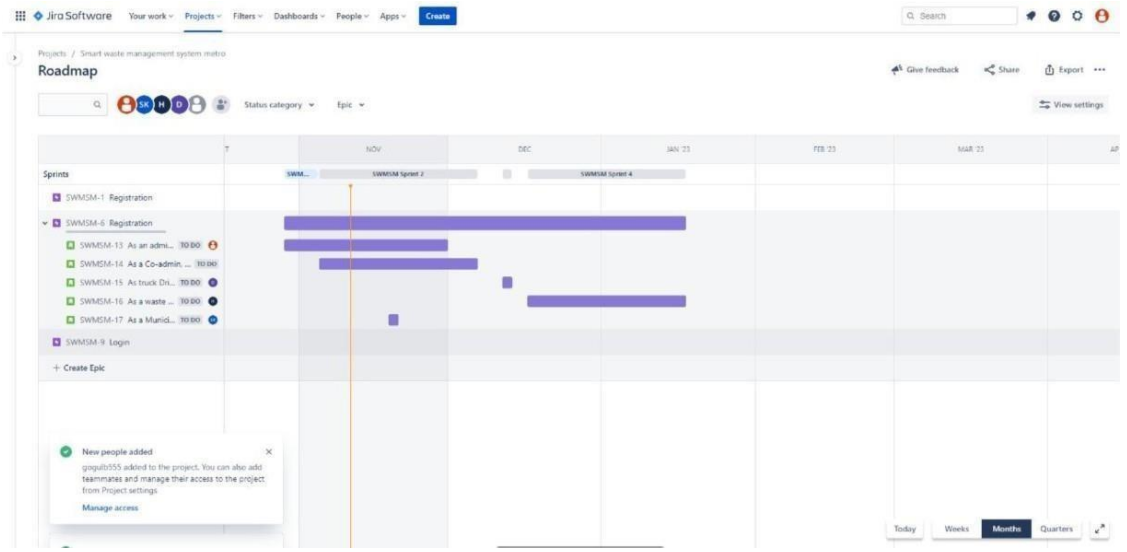
Velocity:
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

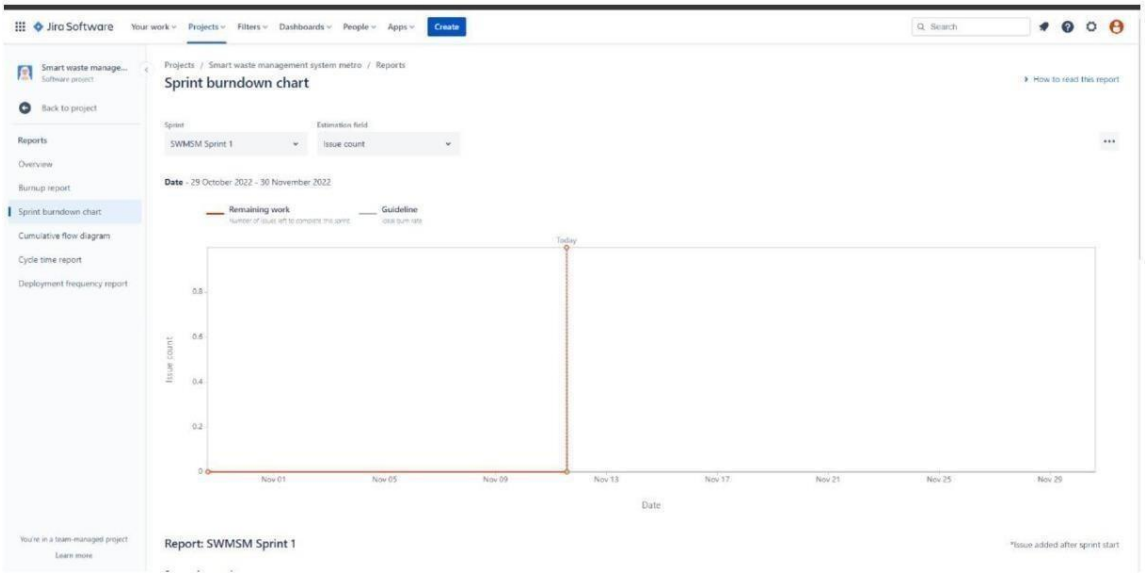
$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

6.3 Reports from JIRA

BURNDOWN CHART

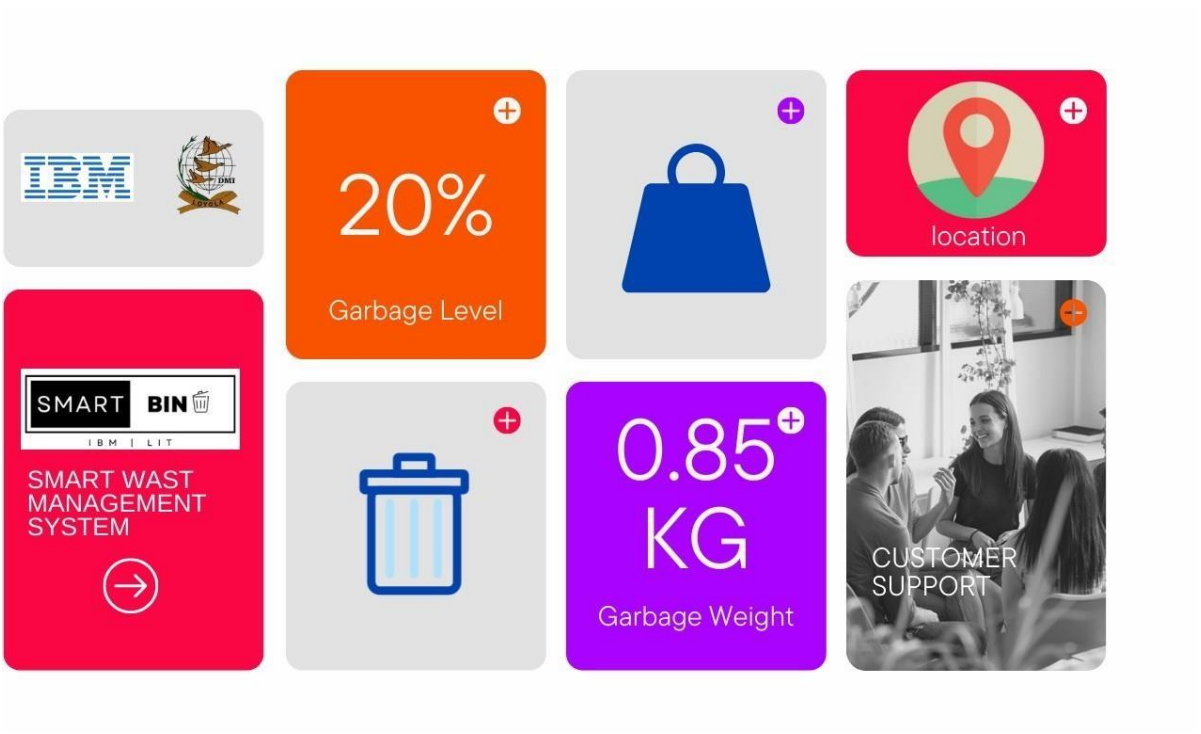




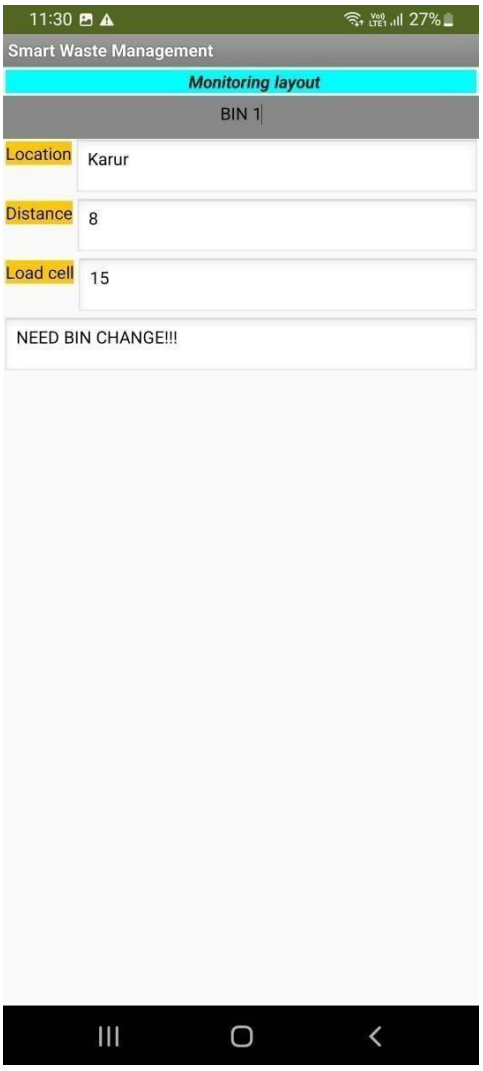

Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Smart waste manage...
Software project
Back to project
Reports
Overview
Burnup report
Sprint burndown chart
Cumulative flow diagram
Cycle time report
Deployment frequency report
You're in a team-managed project
Learn more
Projects / 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
0.8
0.6
0.4
0.2
0
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

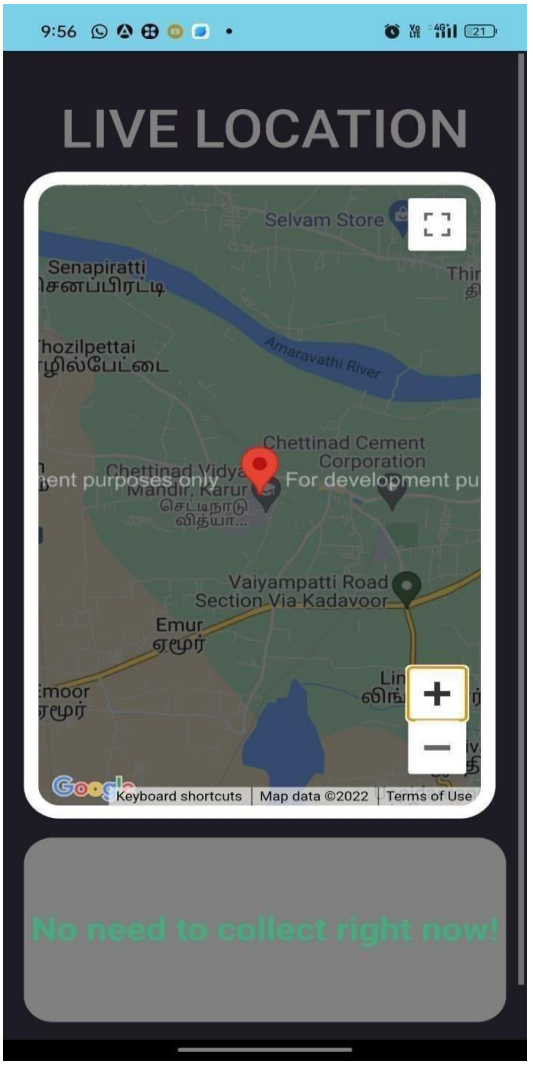
7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1- LOCATION TRACKER



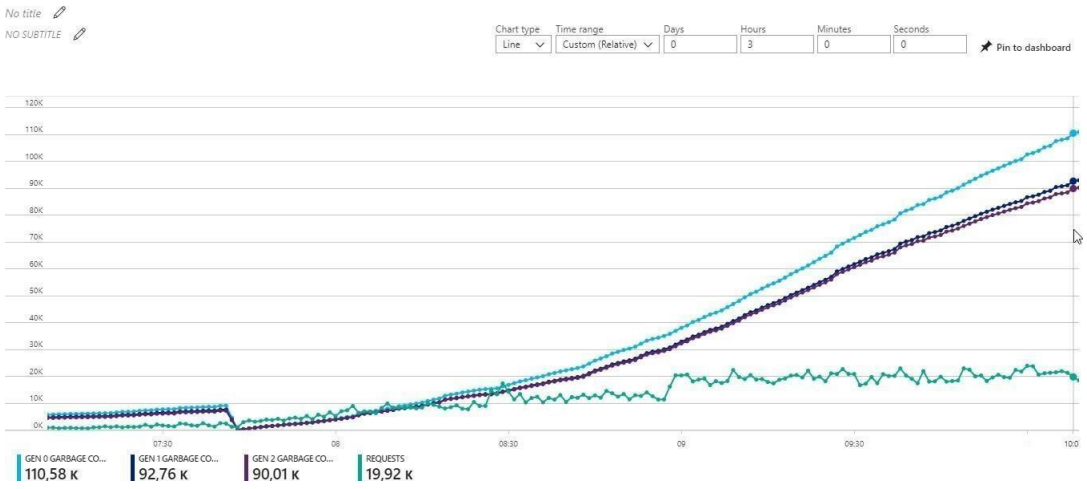
7.2 Feature 2- LIVE UPDATE ON COLLECTED DATA

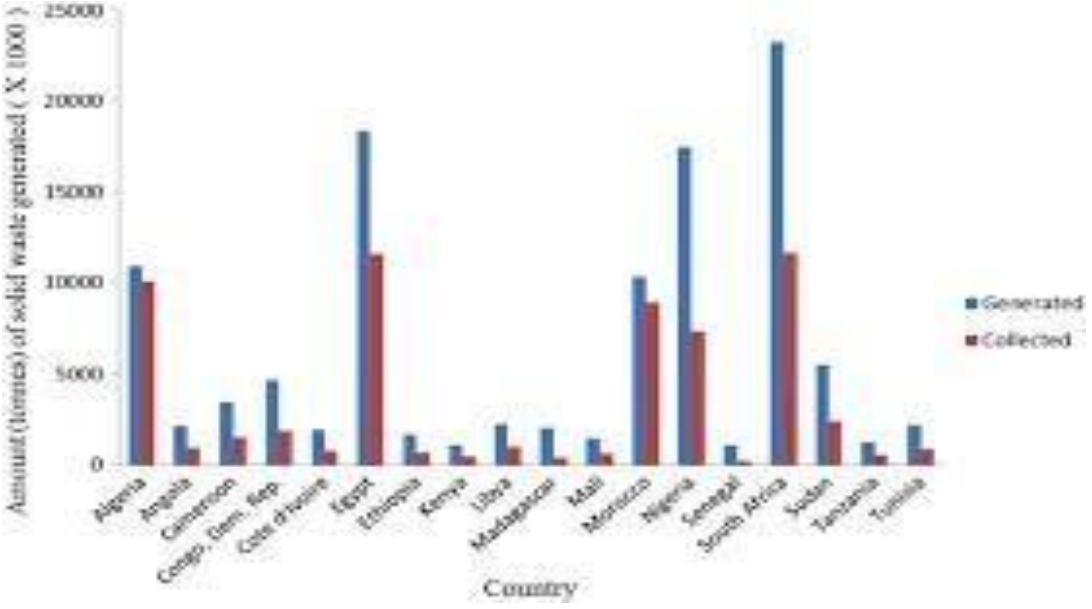




8. RESULTS & TESTING

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 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 : PNT2022TMID12076
```

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)
                                                    while GPIO.input(PIN_ECHO)==0:
                                                    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)
                                                    return distance



                                                    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;
    }

}
```

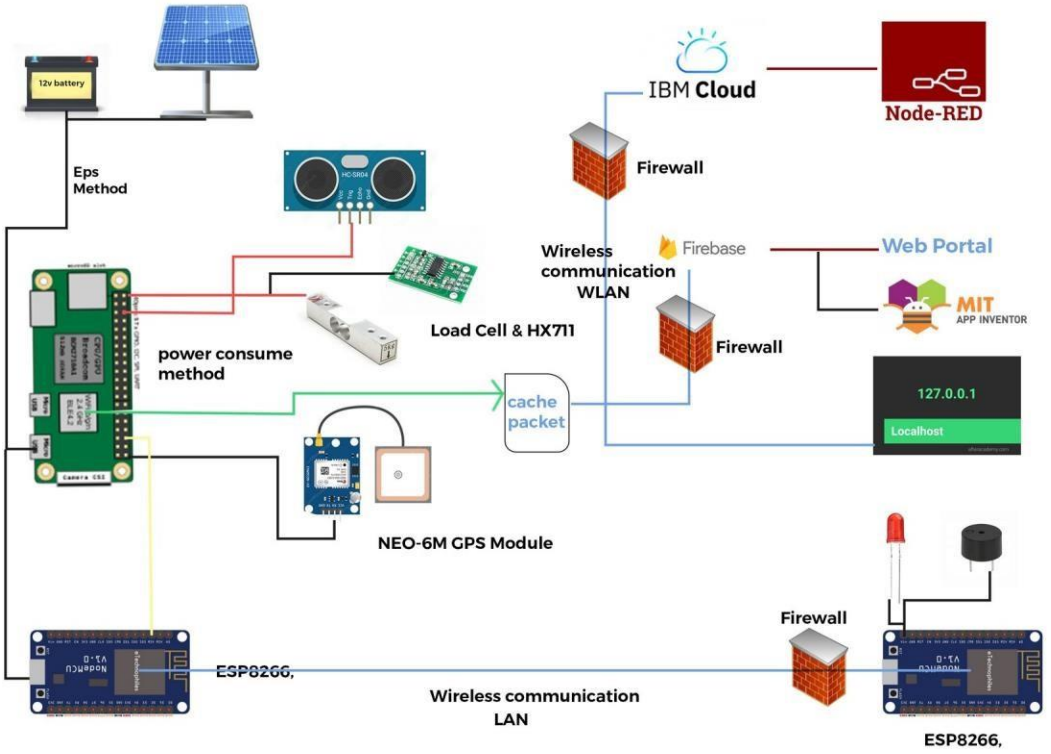
For simulator python code

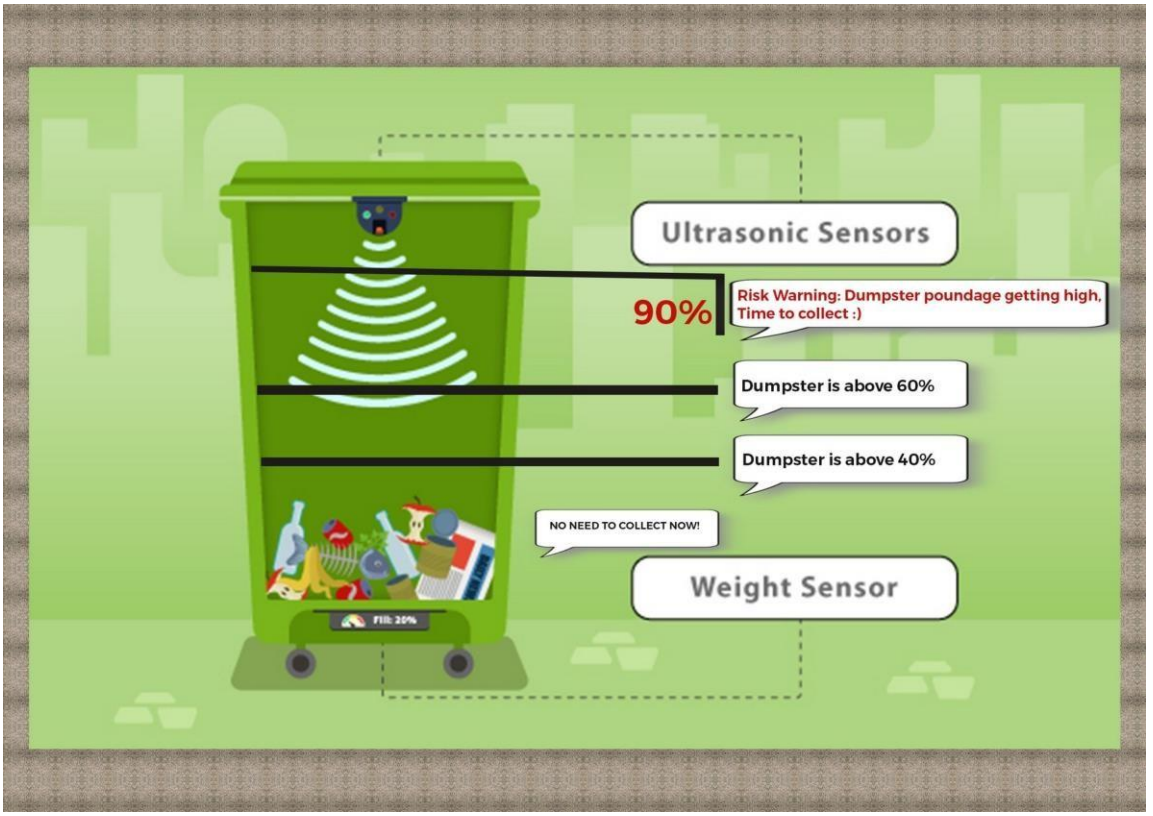
BIN1.PY

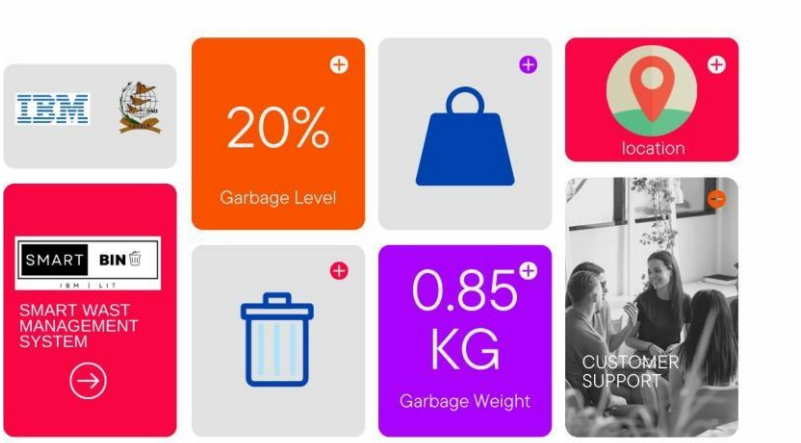
```
import requests
import json
import ibmiotf.application
import ibmiotf.device
import time
import random
import sys

# 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()
```

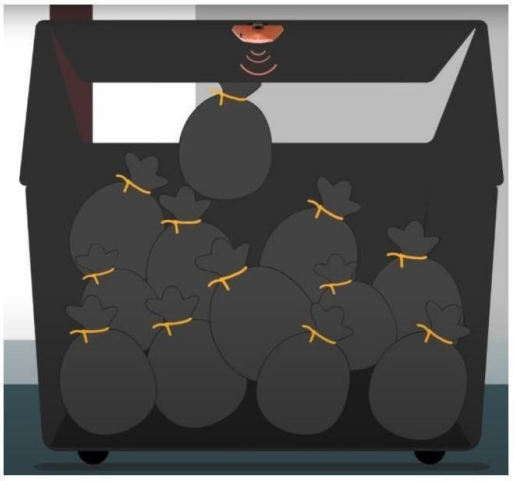
13.2 OUTPUT PICTUR






IBM
20%
Garbage Level
location
SMART BIN
SMART WAST
MANAGEMENT
SYSTEM
0.85
KG
Garbage Weight
CUSTOMER
SUPPORT

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 :)

Testing project smart bin Data Analysis