# 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**

Source Code

GitHub & Project Demo Link

Project Report

|  |  |
| --- | --- |
| **Team ID** | **NM2023TMID14696** |
| **Project Name** | **Smart waste management system for metropolitan cities** |

# INTRODUCTION

## 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**

## 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.

* 1. **LITERATURE SURVEY:**
     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.

* + 1. **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:

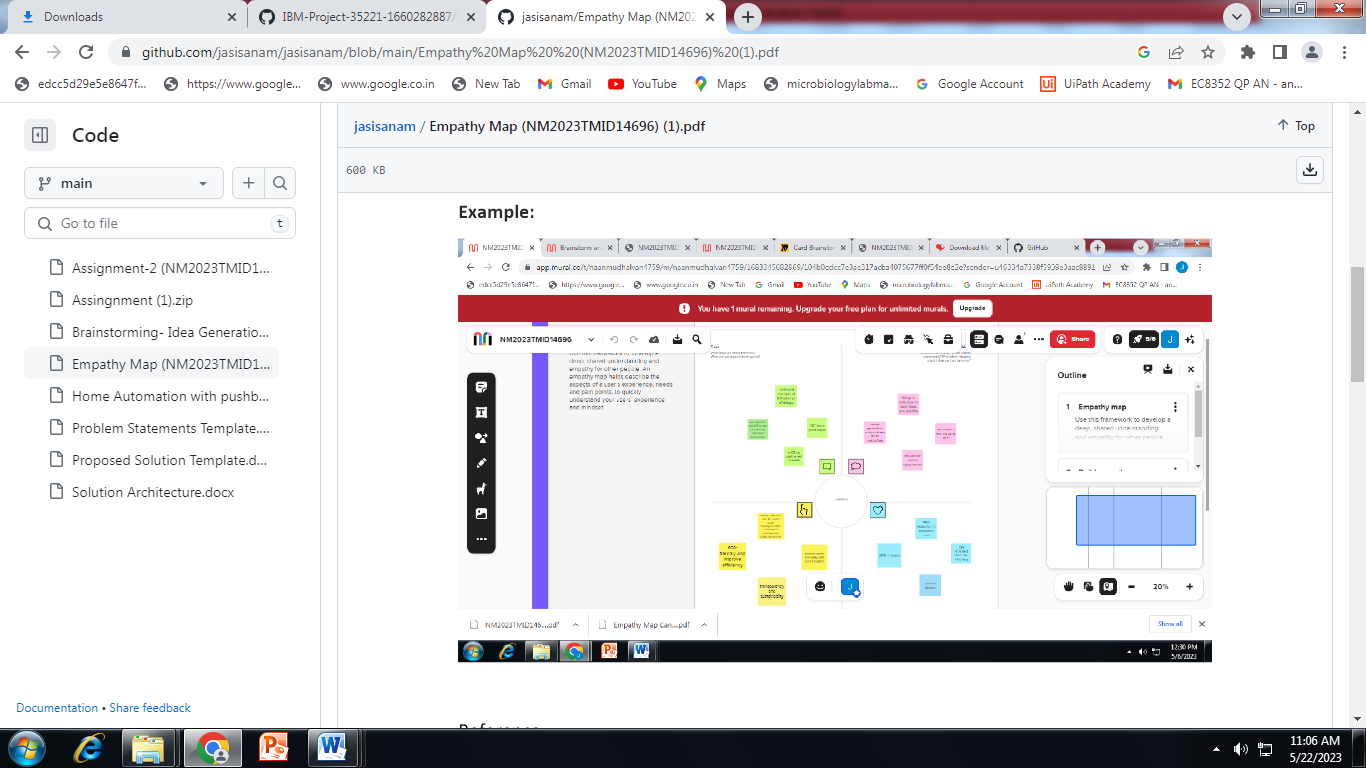
The objective of this paper is to examine the present methods used in India for the welfare of its people in different waste management efforts. The other goal is to offer advice on how to make Indian municipalities' trash disposal procedures better. On secondary research, this essay is founded. The 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.

* + 1. **Problem Statement Definition:**

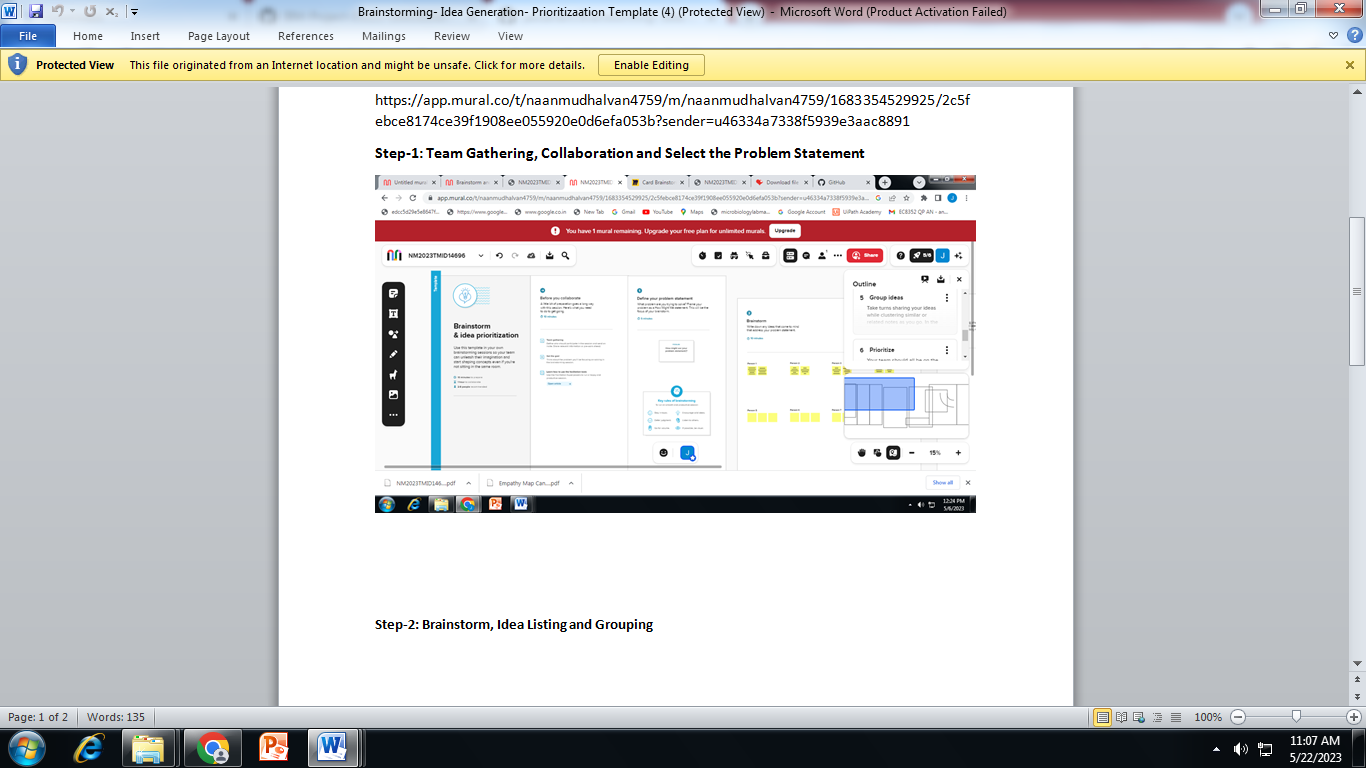
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Problem**  **Statement (PS)** | **I am (Customer)** | **I’m trying to** | **But** | **Because** | **Which makes mefeel** |
| PS-1 | Municipal corporation authority | Get notified when the trash cans are full and be made aware of where the full cans are located. | Don't havethe facilities atthe moment | There is no toolavailable to determine the 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 acity  is invariably  crowd. | Worried |

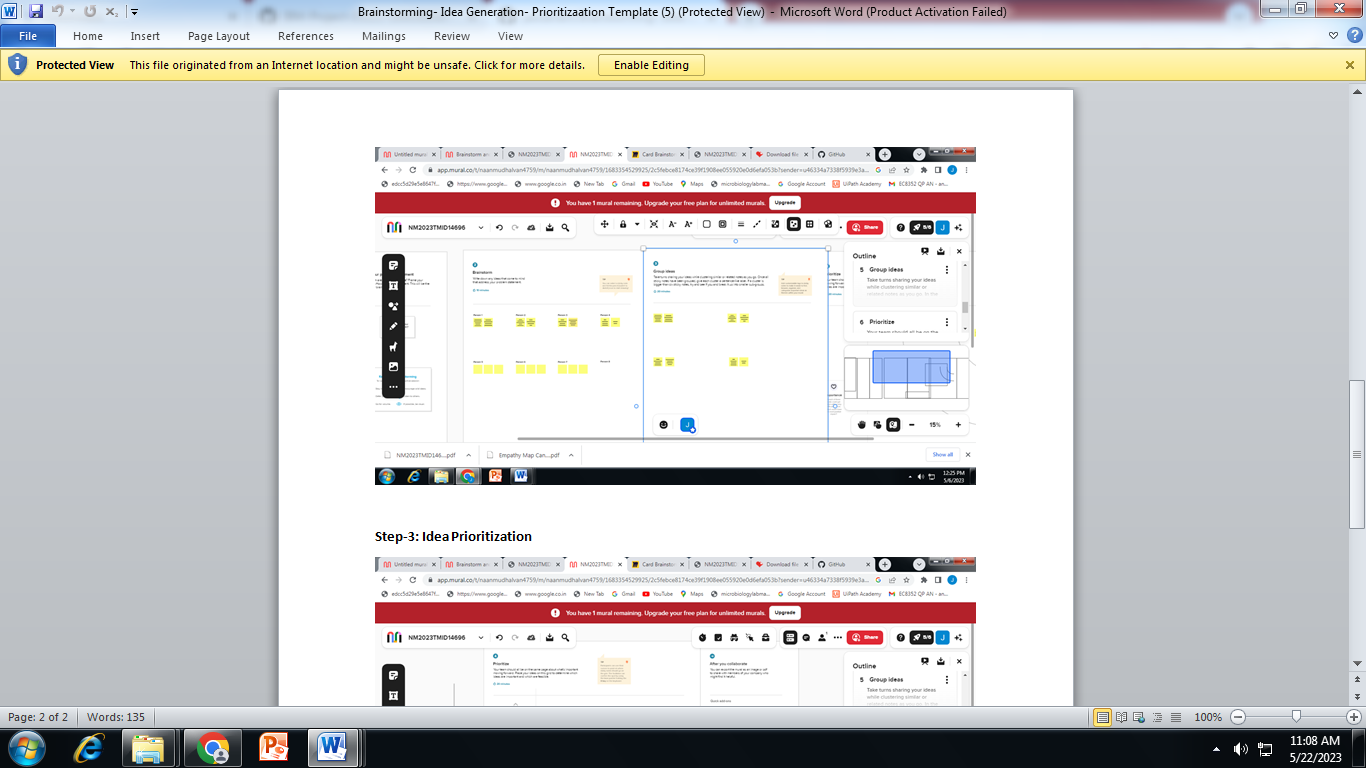
# IDEATION & PROPOSED SOLUTION

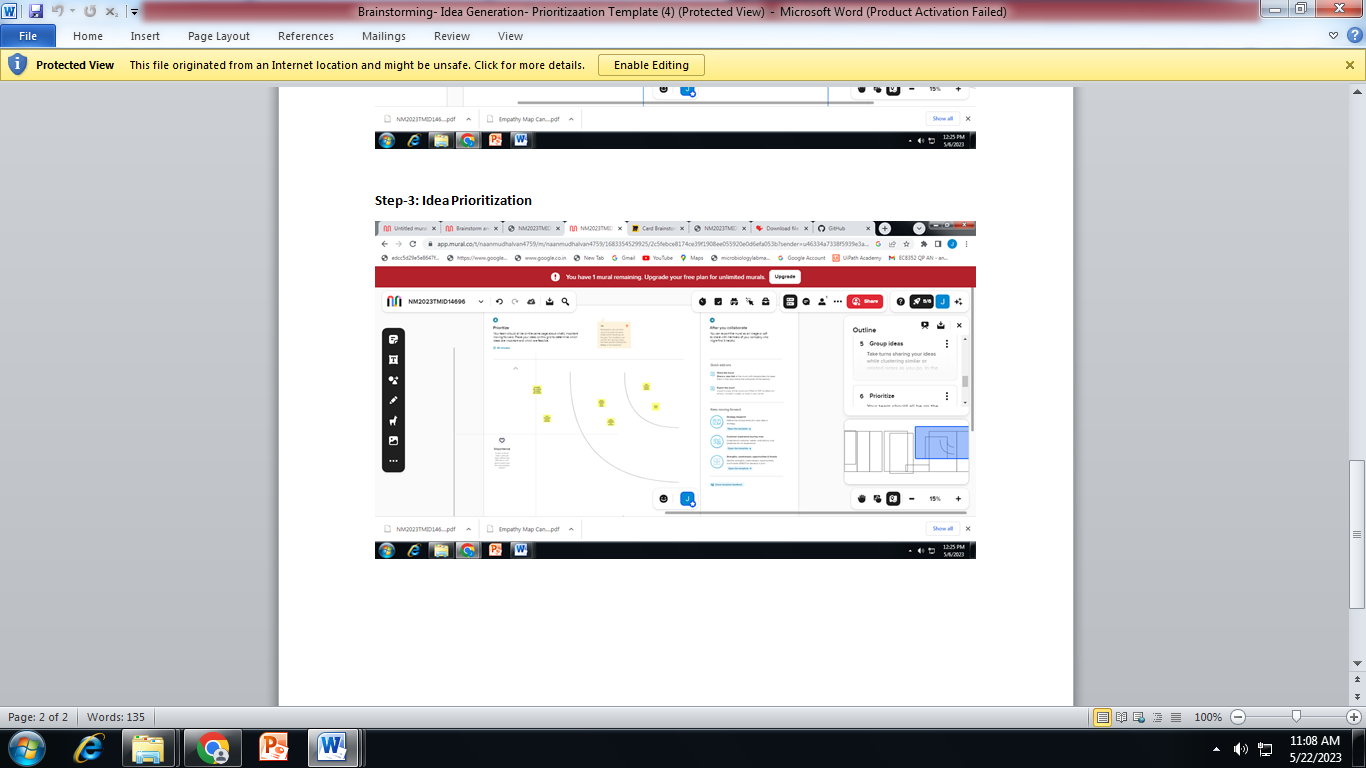
* + 1. **Empathy Map Canvas**



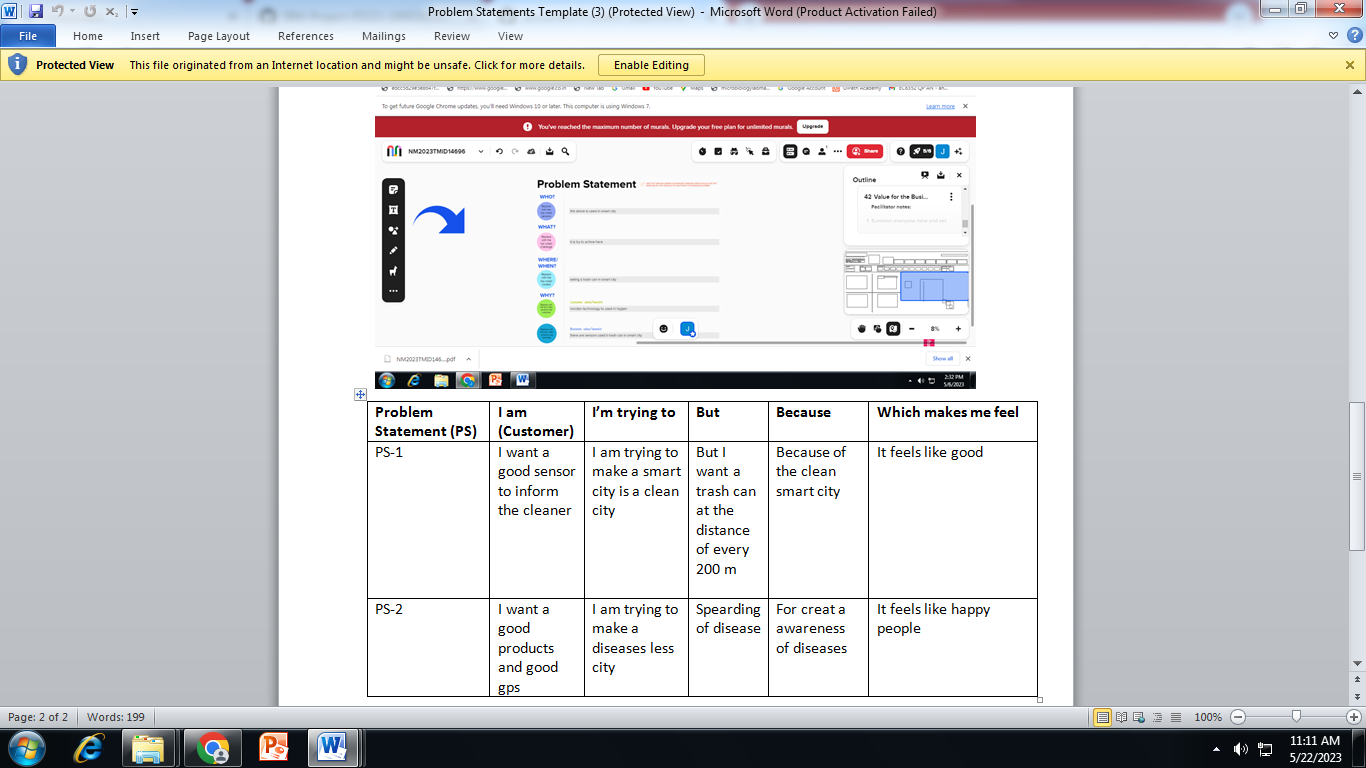
# Ideation & Brainstorming







# Problem Solution fit



* 1. **PROJECT DESIGN**

## 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

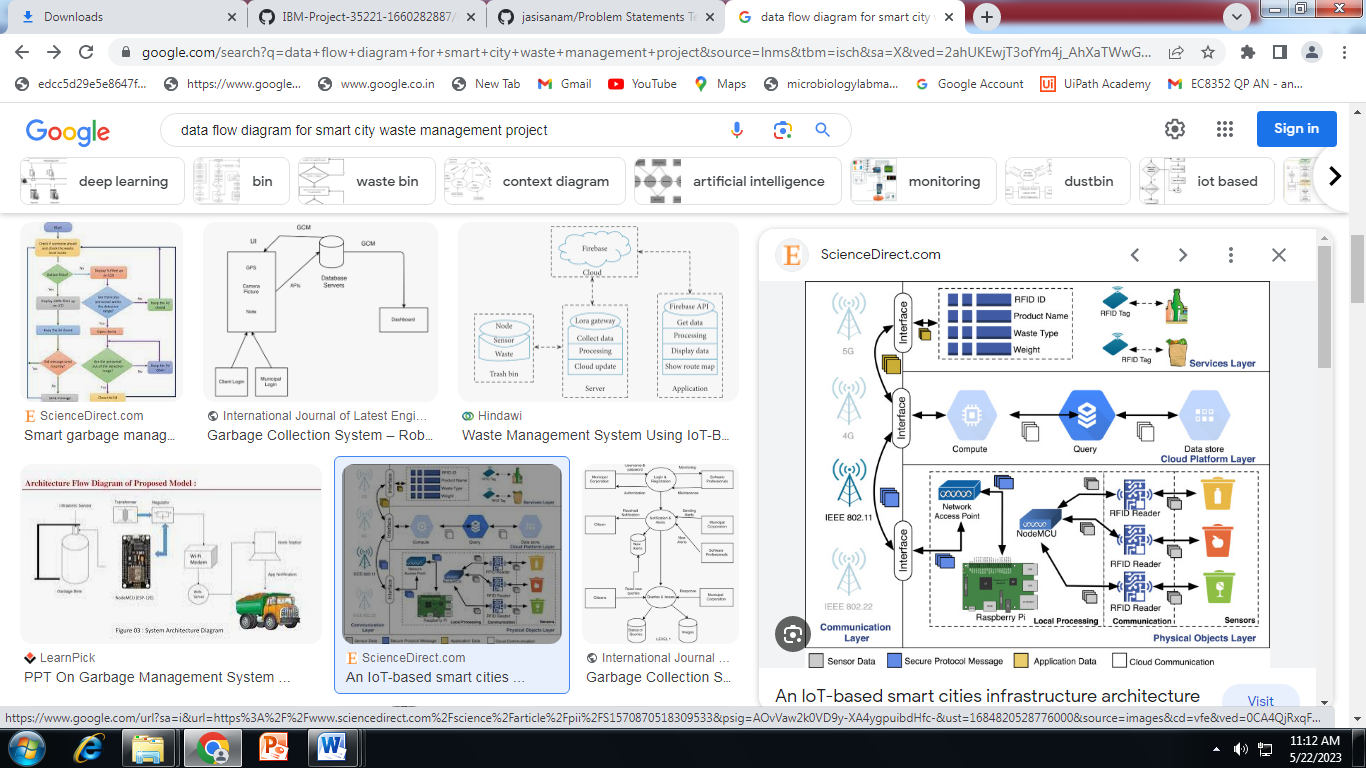
**value, which is set to 80% then the first warning SMS is being sent, as depicted**

* 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

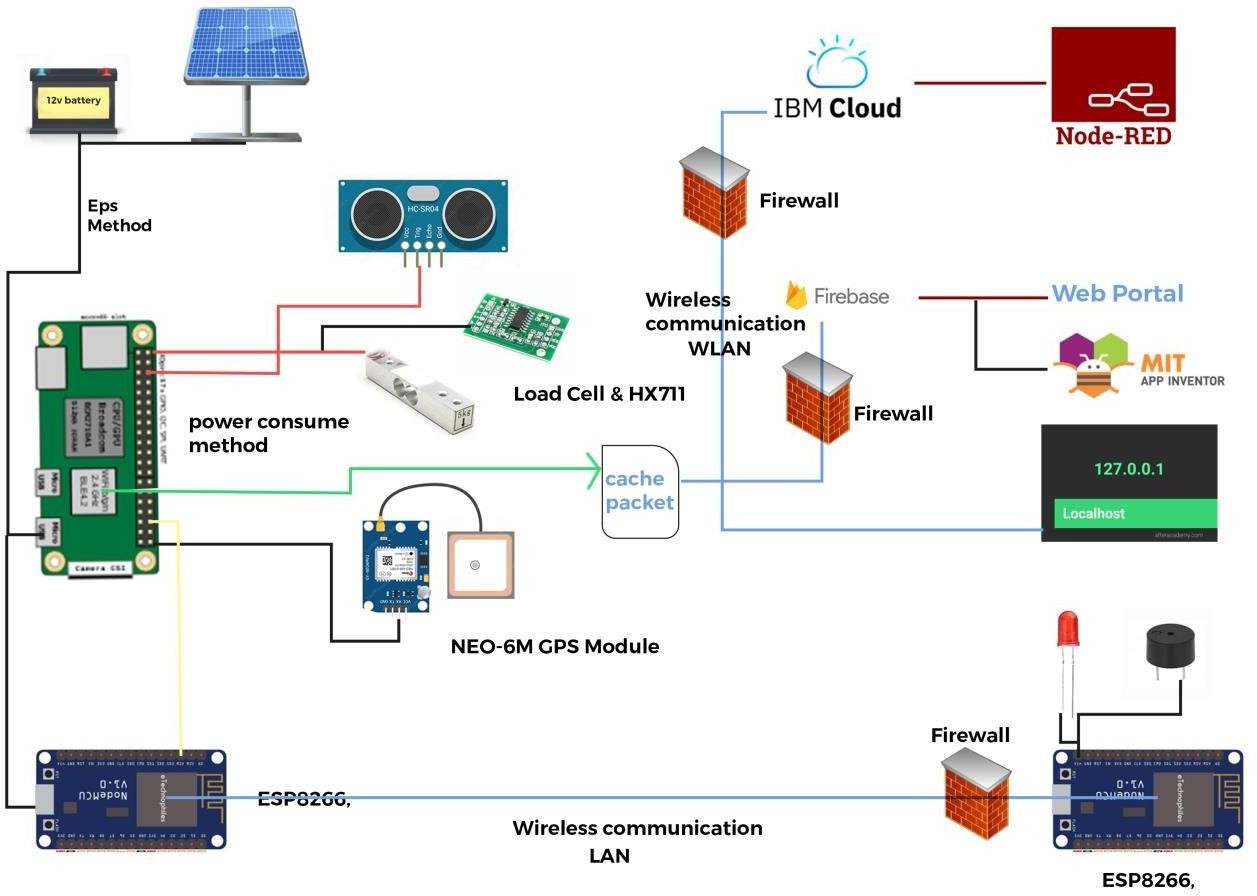
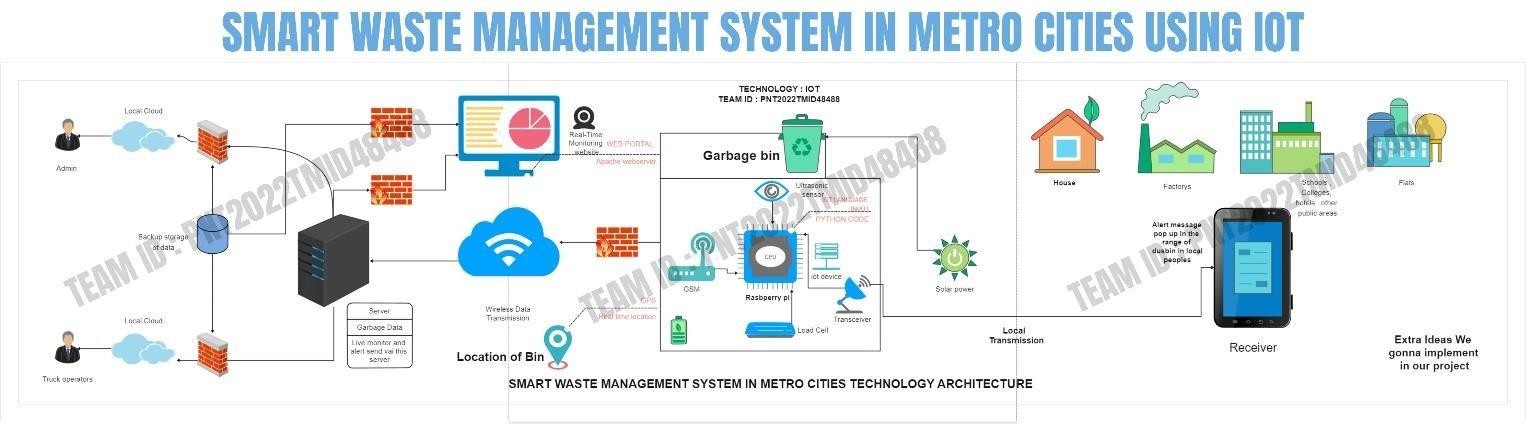
**the garbage needs to be collected immediately**

* 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.

## Data flow diagram:



* 1. **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 |

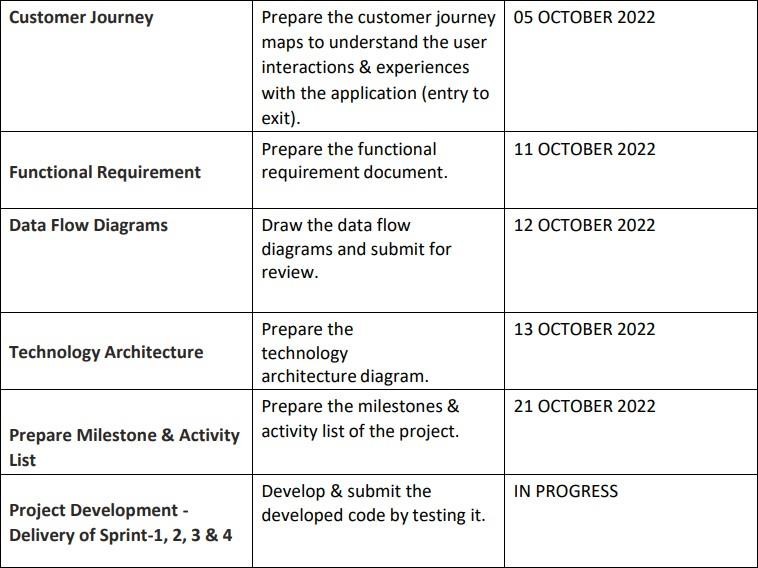
* 1. **User Stories**

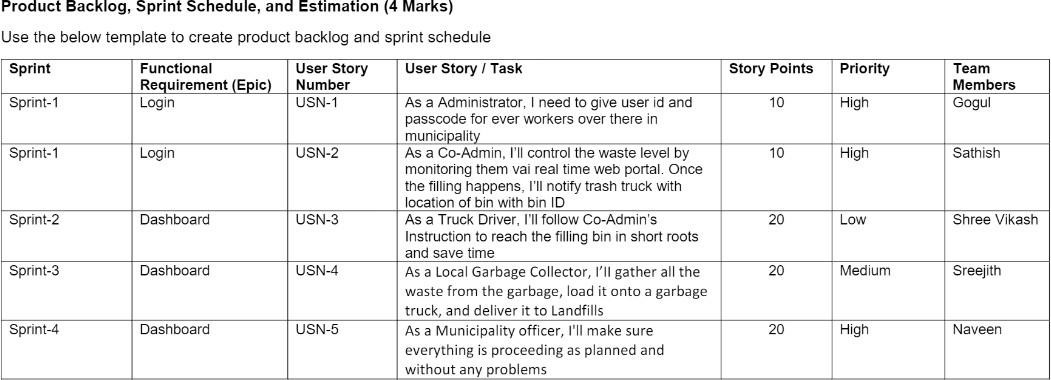
Use the below template to list all the user stories for the product.

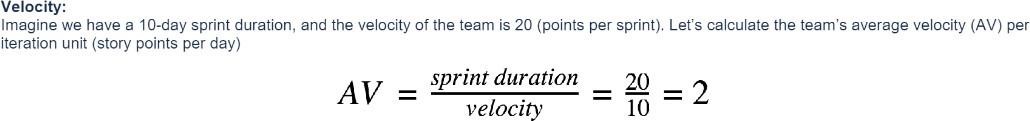
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| Admin | Login | USN-1 | As an administrator, I assigned user namesand passwords 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 |

# PROJECT PLANNING & SCHEDULING

## Sprint Planning & Estimation

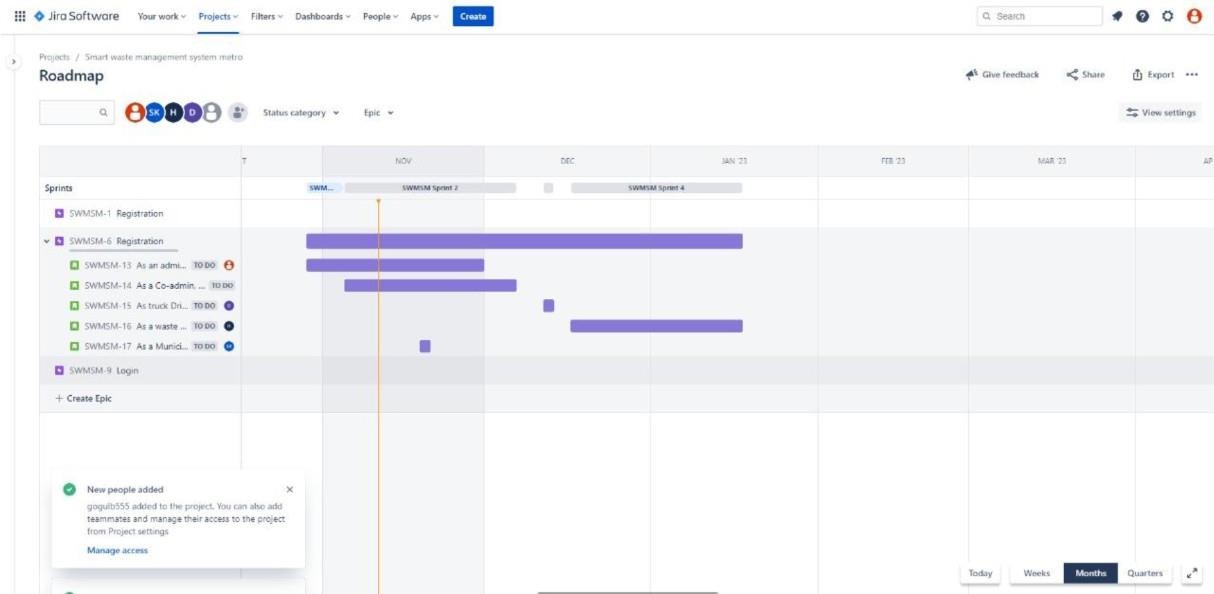
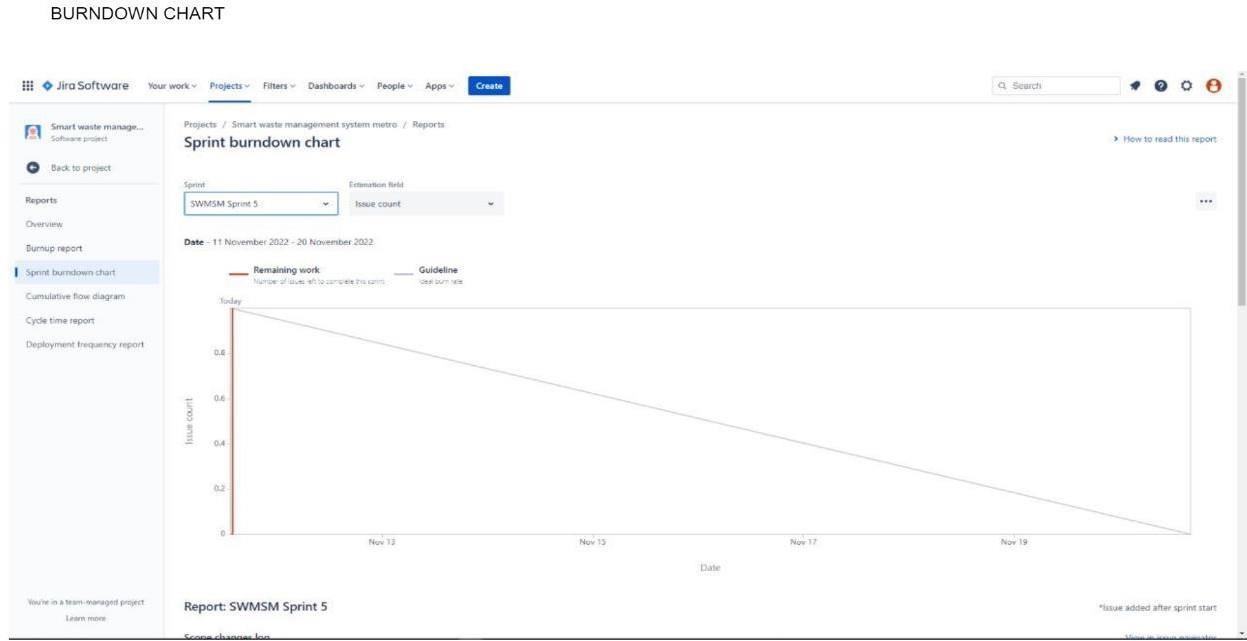






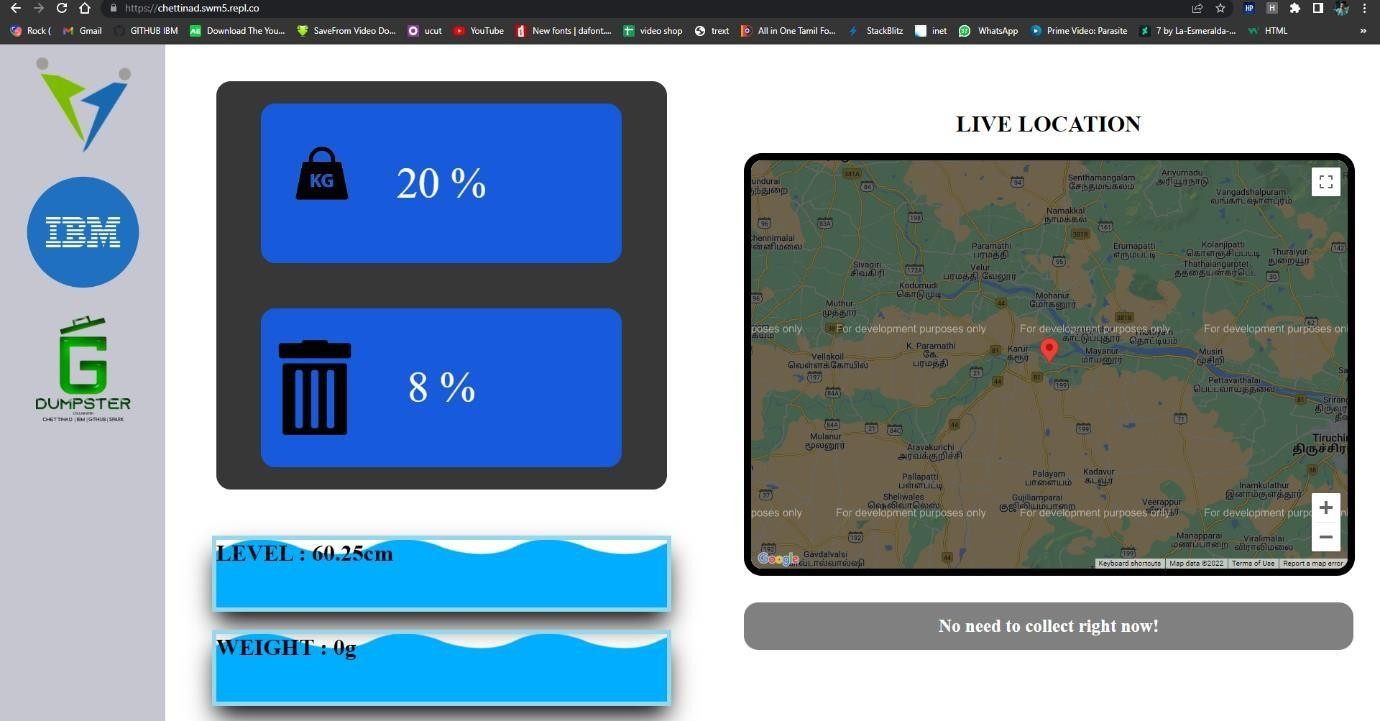
**6.2. Sprint Delivery Schedule**

**6.3 Reports from JIRA**



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

## Feature 1- LOCATION TRACKER



# 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.

# 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.

# 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 : PNT2022TMID48488 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 |
| print("data pushed") |
| c = 1 |
| Break |
| LOAD.py |
| Import  Time |
| 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 |

|  |
| --- |
| 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.byte\_format = 'MSB' |
| self.bit\_format = 'MSB' |
|  |
|  |

|  |
| --- |
| # 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() |

<html>

WEBSITE CODING

Index.html

<!DOCTYPE html>

<head>

<link rel="stylesheet" href="[https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css](https://cdn.jsdelivr.net/npm/bootstrap%404.3.1/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](http://www.gstatic.com/firebasejs/8.10.1/firebase-app.js)"></script>

<script src="https://[www.gstatic.com/firebasejs/8.10.1/firebase-database.js](http://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"

};

</head>

firebase.initializeApp(firebaseConfig)

</script>

<script defer src="database.js"></script>

<body style="background-color:#1F1B24;">

<script src="map.js"></script>

</div>

</div>

<div id="map\_container">

<h1 id="live\_location\_heading" >LIVE LOCATION</h1>

<div id="map"></div>

<div id="alert\_msg">ALERT MESSAGE!</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);

});

}, function (error) {

var marker = new google.maps.Marker({ position: latlong }); marker.setMap(map);

console.log("Error: " + error.code);

});

}

html, body

Style.css

{

height: 100%;

margin: 0px; padding:0px;

#container

{

}

}

display: flex;

flex-direction: row; height: 100%;

width: 100%; position: relative;

#logo\_container

{

}

.logo

{

height: 100%;

width: 12%;

background-color: #C5C6D0; display: flex;

flex-direction: column; vertical-align: text-bottom;

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

{

}

.status

{

background-image: url("/Images/dust.png"); background-repeat: no-repeat;

background-size: 150px; background-position: left center;

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

{

}

#alert\_msg

{

}

.lat

{

}

height: 70%;

width: 90%;

margin-left: 4%; margin-right:4%;

border: 10px solid white; border-radius: 25px;

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;

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)**

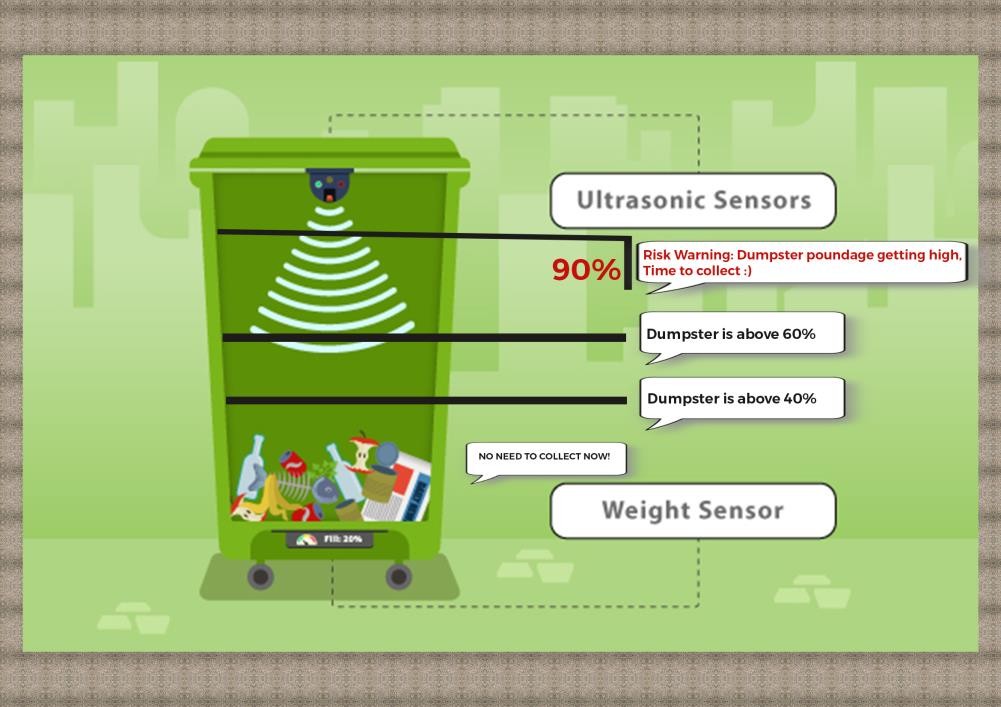
**success=deviceCli.publishEvent ("IoTSensor","json",warn,qos=0,on\_publish= myOnPublishCallback)**

**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()**



* 1. **LINKS GitHub Link:**

