SMART WASTE MANAGEMENTSYSTEM

FOR METROPOLITIAN CITIES

A PROJECT REPORT

Submitted by

KALAIYARASI L

SRINIDHI S

SWETHA M

THRISHA E

From

BANNARI AMMAN INSTITUTE OF TECHNOLOGY

SATHYAMANGALAM, ERODE.

1.Introduction

1.1 Project Overview

With rapid increase in population, the issues related to sanitation with respect to garbage management are degrading immensely. It creates unhygienic conditions for the citizens in the nearby surrounding, leading to the spread of infectious diseases and illness. To avoid this problem, IoT based "Smart Waste Management" is the best and trending solution. In the proposed system, public dustbins will be provided with embedded device which helps in real time monitoring of level of garbage in garbage bins. The data regarding the garbage levels will be used to provide optimized route for garbage collecting vans, which will reduce cost associated with fuel. The load sensors will increase efficiency of data relatedto garbage level and moisture sensors will be used to provide data of waste segregation in a dust bin. The analysis of ceaseless data gathered will help municipality and government authorities to improve plans related to smart waste management with the help of various system generated reports.

1.2 Purpose

Smart waste management focuses on solving the previously mentioned solid waste management problems using sensors, intelligent monitoring systems, and mobile applications. The first smart waste management solution to make the waste collection process more efficient is sensors. Sensors can measure the fill level of the containers and provide updated information at any time and notify waste management services to empty them when they are full or almost full. These devices help optimize the best possible route containing fully filled containers and create smart schedules for drivers. The selection of the containers also minimizes the need for trash collection staff because their duties are deduced.

They can also alert the waste management companies or municipalities if an ‘undesirable incident happens such as sudden temperature rise or displacement of the container by their GPS features.

2. Literature survey:

2.1 Existing system

Around 80% of waste collections happen at the wrong time. Late waste collections lead to overflowing bins, unsanitary environments, citizen complaints, illegal dumping, and increased cleaning and collection costs. Early waste collections mean unnecessary carbon emissions, more traffic congestion, and higher running costs. The old way of doing waste management is highly inefficient. And in today’s ever-technological world, an innovative and data-driven approach is the only way forward.

Traditionally, municipalities and waste management companies would operate on a fixed collection route and schedule. This means that waste collection trucks would drive the same collection route and empty every single waste container – even if the waste containerdid not need emptying. This means high labor and fuel costs – which residents ultimately foot the bill for. This is also an unsustainable way of working - the more vehicles on the road carrying out unnecessary collections means more carbon emissions are released into our planet’s atmosphere.

2.2 Reference

[1] Mohammad Aazam, Marc St-Hilaire, Chung-Horng Lung, Ioannis Lambadaris , (2016),”Cloud-based Smart Waste Management for Smart Cities”, IEEE

[2] Dr. N. Sathish Kumar, B. Vijayalakshmi, R. Jenifer Prarthana, A .Shankar, (2016), “IoT Based Smart Garbage alert system using Arduino UNO “, IEEE

[3] Belal Chowdhury, Morshed U. Chowdhury, (2007) “RFID-based Real-timeSmart Waste Management System”, Australasian Telecommunication Networks and Applications Conference, December, Christchurch, New Zealand

[4] Mohd Helmy Abd Wahab, Aeslina Abdul Kadir, Mohd Razali Tomari and Mohamad Hairol Jabbar (2014), “Smart Recycle Bin A Conceptual Approach of Smart Waste Management with Integrated Web based System“, IEEE

[5] F achmin F olianto, Y ong Sheng Low, Wai Leong Yeow , (2015) “Smartbin: Smart Waste Management System”, Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP) Singapore, 7-9 April, IEEE

[6] Gopal Kirshna Shyam, Sunilkumar S. Manvi, Priyanka Bharti, (2017) ” Smart Waste Management using Internet-of- Things (IoT)” Second International Conference On Computing and Communications Technologies(ICCCT’17), IEEE

[7] Keerthana B, Sonali M Raghavendran, Kalyani S, Suja P, V.K.G.Kalaiselvi, (2017), “Internet of Bins Trash Management in India “, IEEE

[8] Bharadwaj B, M Kumudha, Gowri Chandra N, Chaithra G, (2017) “Automation of Smart Waste Management Using IoT to Support “Swachh Bharat Abhiyan” – a practical Approach “ IEEE

[9] Shubham Thakker, R.Narayanamoorthi, (2015), “Smart and Wireless Waste Management An innovative way to manage waste and also produce energy” 2nd International Conference on Innovations in Information Embedded and Communication Systems ICIIECS’15 , IEEE

[10] Artemios G. Voyiatzis, John Gialelis, and Dimitrios Karadimas, (2014) “Dynamic Cargo Routing on-the- Go: The Case of Urban Solid Waste Collection” 2 nd IEEE WiMob 2014 international workshop on smart city and ubiquitous computing application , IEEE

2.3 Problem Statement Solution

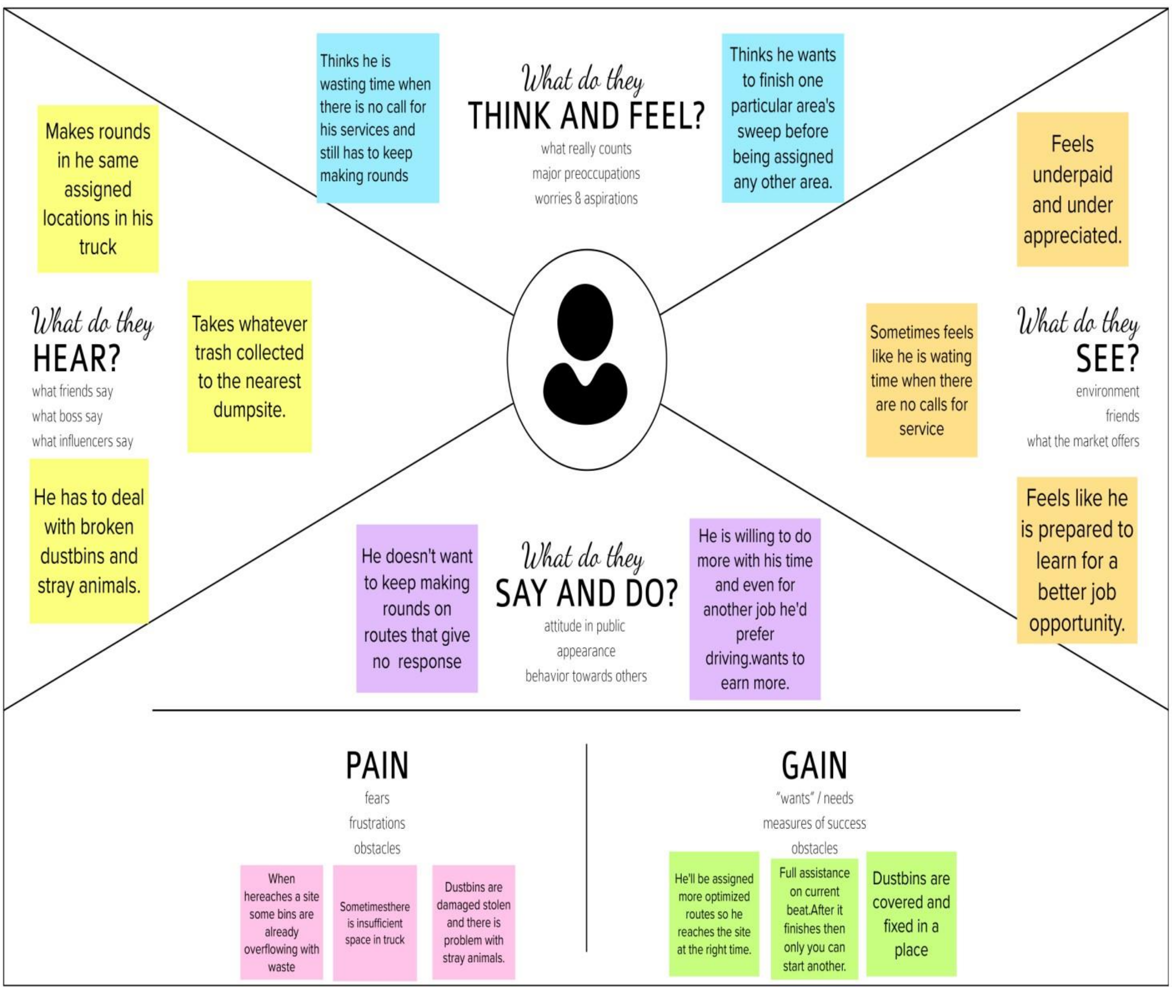
The nation and world are facing a huge problem today of disposal, segregation, and recycling of solid waste and improper management of these wastes are hazardous and dangerous to human health and ecological system. The generation and disposal of waste in large quantities has created a greater concem over time for the world which is adversely affecting the human lives and environmental conditions. Wastes are the one which grows with the growth of the country. A voluminous amount of waste that is generated is disposed of by means which have an adverse effect on the environment. The common method of disposal of the waste is by unplanned and uncontrolled open dumping at the landfill sites. This method is injurious to human health, plant and animal life. This harmfulmethod of waste disposal can generate liquid leachate which can contaminate the surface and ground waters; can harbor disease vectors which spread harmful diseases, can degradethe aesthetic value of the natural environment and is an unavailing use of land resources. Segregation of waste is important for proper disposal of the vast amount of garbagemodern society produces in an environmentally sensible mode. People became adapted to tossing things away and never realize the consequences of their action. The common method of disposal of the industrial waste is by uncontrolled and unplanned and exposed dumping at the river sites and open areas. This method is injurious to plants, human and animal life. There is a rapid increase in capacity and categories of solid as a result of urbanization, constant economic growth and industrialization. Global Waste Market reported that the amount of waste generated worldwide produced is 2.02 billion tonnes.”Wastes are not always waste, it has to be handled, segregated, transported and disposed of as to reduce the risk to the public lives and sustainable environments. The economic value of waste is best comprehended when it is segregated. There is no such system employed of segregation of glass, plastic and metallic wastes at, the industrial level. Dry waste consisting of cans, Aluminium foils, plastics, metal, glass and paper couldbe recycled. If we do not dispose of the waste in a more systematic manner, more than 1400 sq.km of land, which is the size of the city of Delhi, would be required in the countryby the year 2047 to dispose of it.

3. Ideation and proposed solution:

3.1 Empathize & Discover

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviours and attitudes. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenges.



What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations

Thinks he is wasting time when there is no call for his services and still has to keep making rounds

Thinks he wants to finish one particular area's sweep before being assigned any other area.

What do they HEAR?
what friends say
what boss say
what influencers say

Makes rounds in he same assigned locations in his truck

Takes whatever trash collected to the nearest dumpsite.

He has to deal with broken dustbins and stray animals.

What do they SEE?
environment
friends
what the market offers

Feels underpaid and under appreciated.

Sometimes feels like he is wating time when there are no calls for service

Feels like he is prepared to learn for a better job opportunity.

What do they SAY AND DO?
attitude in public
appearance
behavior towards others

He doesn't want to keep making rounds on routes that give no response

He is willing to do more with his time and even for another job he'd prefer driving.wants to earn more.

PAIN
fears
frustrations
obstacles

When hereaches a site some bins are already overflowing with waste

Sometimesthere is insufficient space in truck

Dustbins are damaged stolen and there is problem with stray animals.

GAIN
“wants” / needs
measures of success
obstacles

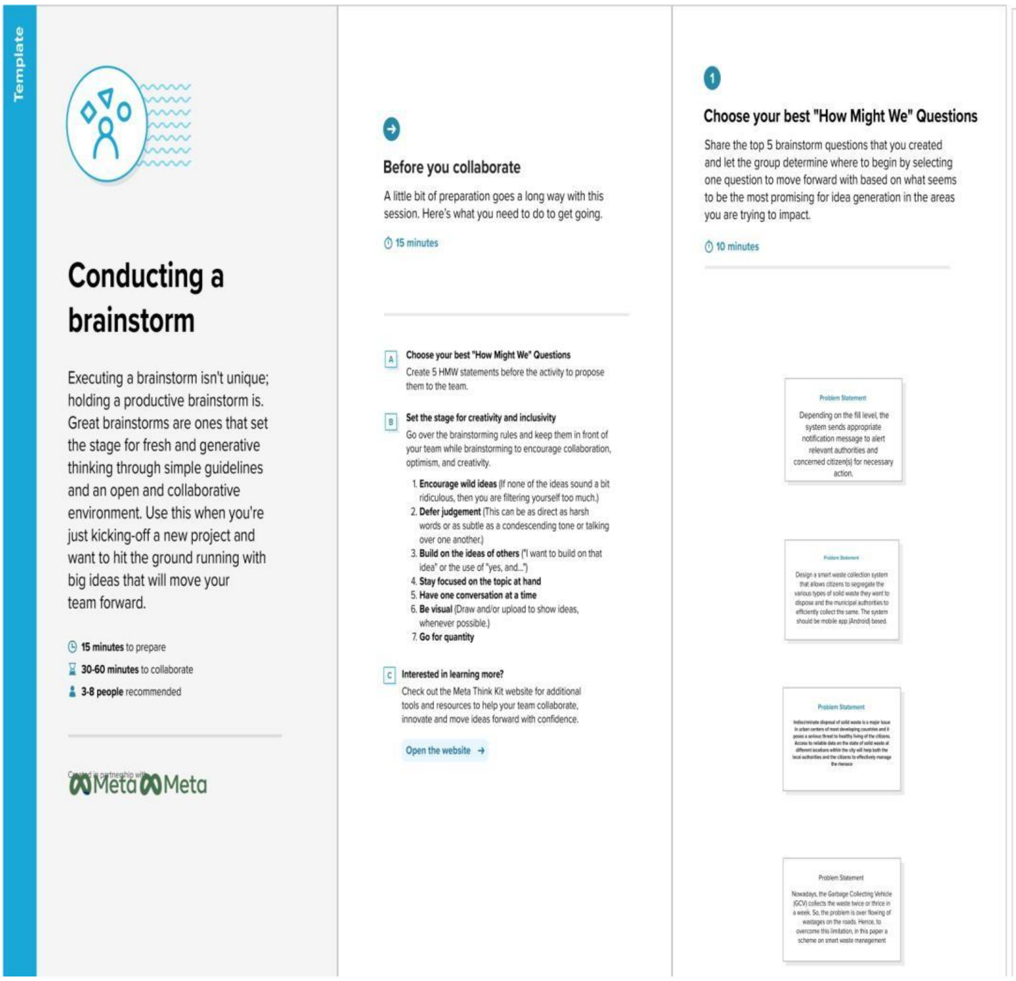
He'll be assigned more optimized routes so he reaches the site at the right time.

Full assistance on current beat.After it finishes then only you can start another.

Dustbins are covered and fixed in a place

3.2 Brainstorm & Idea Prioritization Template

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Conducting a brainstorm

Executing a brainstorm isn't unique; holding a productive brainstorm is. Great brainstorms are ones that set the stage for fresh and generative thinking through simple guidelines and an open and collaborative environment. Use this when you're just kicking-off a new project and want to hit the ground running with big ideas that will move your team forward.

- **15 minutes** to prepare
- **30-60 minutes** to collaborate
- **3-8 people** recommended

Created in partnership with

Meta Meta

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

15 minutes

A **Choose your best "How Might We" Questions**
Create 5 HMW statements before the activity to propose them to the team.

B **Set the stage for creativity and inclusivity**
Go over the brainstorming rules and keep them in front of your team while brainstorming to encourage collaboration, optimism, and creativity.

1. **Encourage wild ideas** (If none of the ideas sound a bit ridiculous, then you are filtering yourself too much.)
2. **Defer judgement** (This can be as direct as harsh words or as subtle as a condescending tone or talking over one another.)
3. **Build on the ideas of others** ("I want to build on that idea" or the use of "yes, and...")
4. **Stay focused on the topic at hand**
5. **Have one conversation at a time**
6. **Be visual** (Draw and/or upload to show ideas, whenever possible.)
7. **Go for quantity**

C **Interested in learning more?**
Check out the Meta Think Kit website for additional tools and resources to help your team collaborate, innovate and move ideas forward with confidence.

Open the website →

1 Choose your best "How Might We" Questions

Share the top 5 brainstorm questions that you created and let the group determine where to begin by selecting one question to move forward with based on what seems to be the most promising for idea generation in the areas you are trying to impact.

10 minutes

Problem Statement

Depending on the fill level, the system sends appropriate notification message to alert relevant authorities and concerned citizen(s) for necessary action.

Problem Statement

Design a smart waste collection system that allows citizens to segregate the various types of solid waste they want to dispose and the municipal authorities to efficiently collect the same. The system should be mobile app (Android) based.

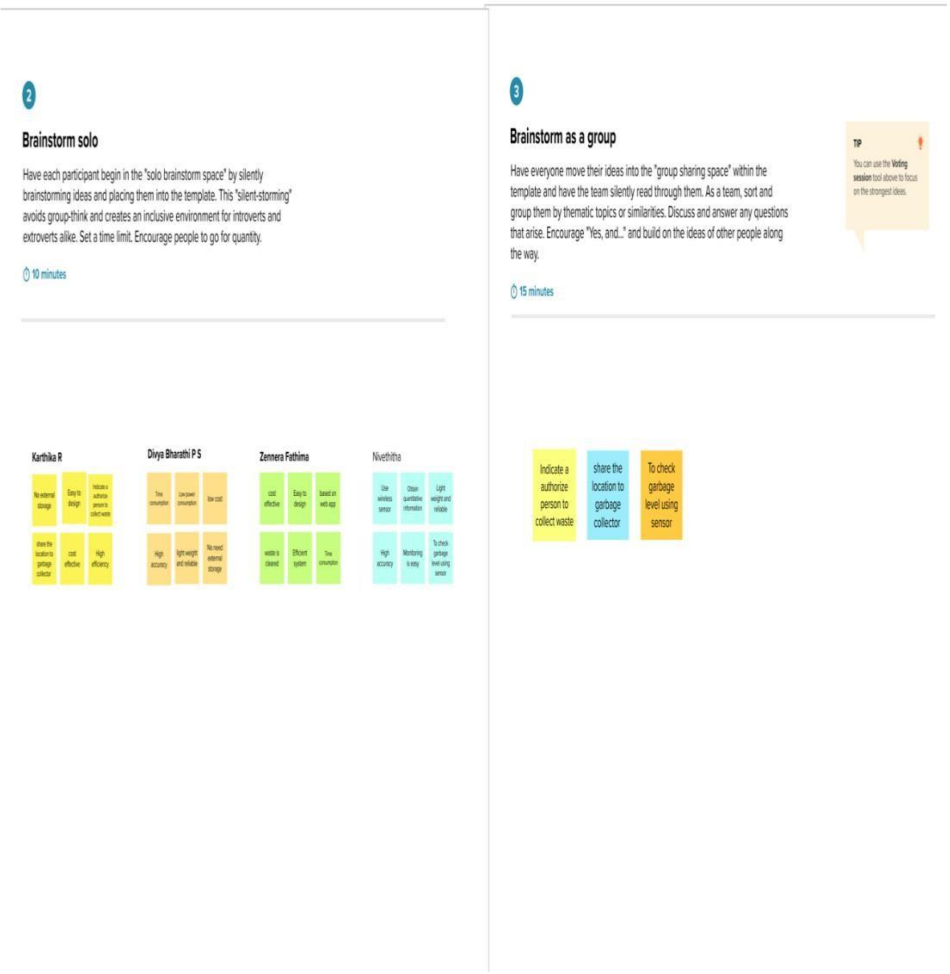
Problem Statement

Indiscriminate disposal of solid waste is a major issue in urban centers of most developing countries and it poses a serious threat to healthy living of the citizens. Access to reliable data on the state of solid waste at different locations within the city will help both the local authorities and the citizens to effectively manage the menace

Problem Statement

Nowadays, the Garbage Collecting Vehicle (GCV) collects the waste twice or thrice in a week. So, the problem is over flowing of wastages on the roads. Hence, to overcome this limitation, in this paper a scheme on smart waste management

Step-2: Brainstorm, Idea Listing and Grouping



2

Brainstorm solo

Have each participant begin in the "solo brainstorm space" by silently brainstorming ideas and placing them into the template. This "silent-storming" avoids group-think and creates an inclusive environment for introverts and extroverts alike. Set a time limit. Encourage people to go for quantity.

10 minutes

Karthika R

No external storage

Easy to design

Indicate a authorize person to collect waste

share the location to garbage collector

cost effective

High efficiency

Divya Bharathi P S

Time consumption

Low power consumption

low cost

High accuracy

light weight and reliable

No need external storage

Zennera Fathima

cost effective

Easy to design

based on web app

waste is cleared

Efficient system

Time consumption

Nivethitha

Use wireless sensor

Obtain quantitative information

Light weight and reliable

High accuracy

Monitoring is easy

To check garbage level using sensor

3

Brainstorm as a group

Have everyone move their ideas into the "group sharing space" within the template and have the team silently read through them. As a team, sort and group them by thematic topics or similarities. Discuss and answer any questions that arise. Encourage "Yes, and..." and build on the ideas of other people along the way.

15 minutes

TIP

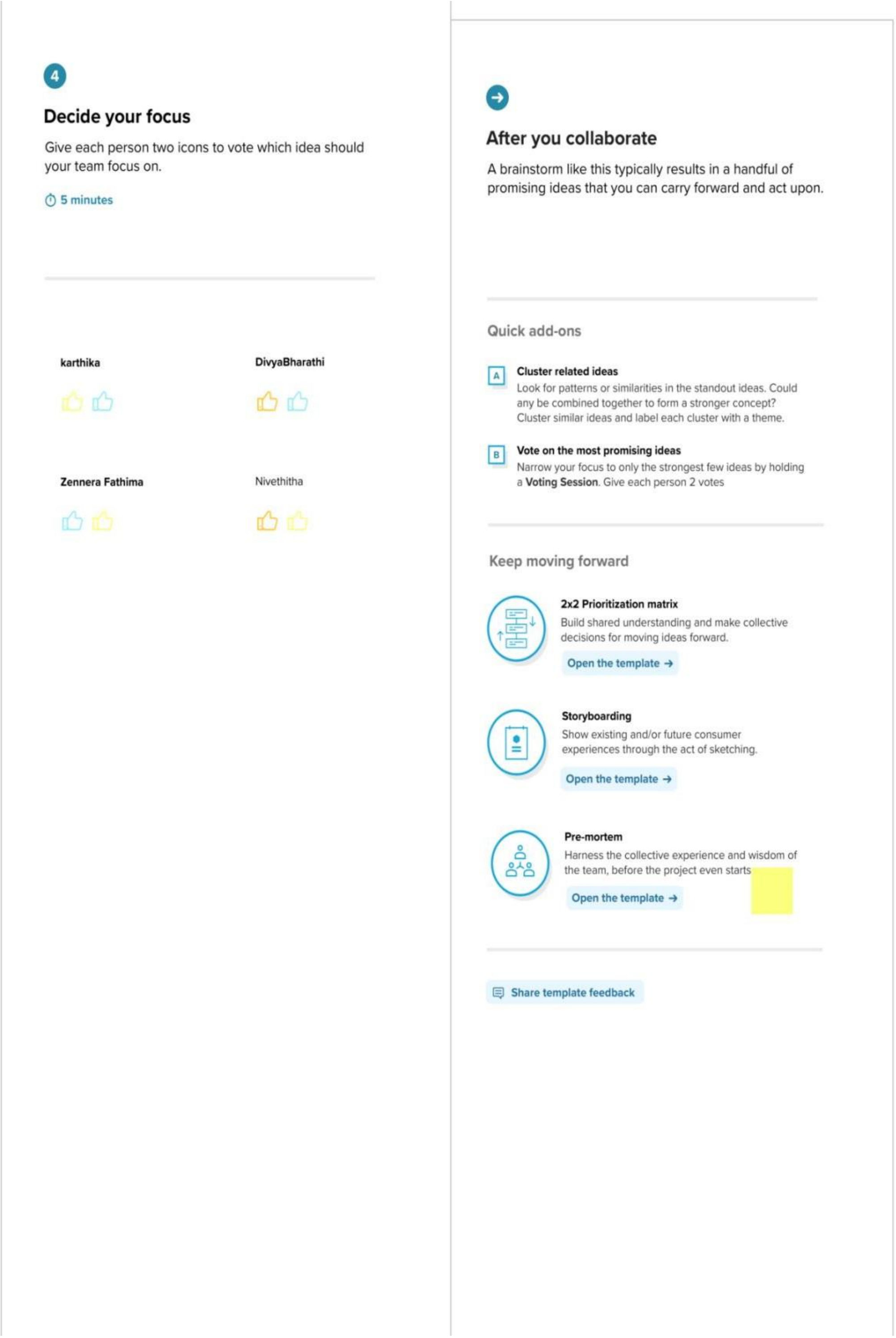
You can use the **Voting session** tool above to focus on the strongest ideas.

Indicate a authorize person to collect waste

share the location to garbage collector

To check garbage level using sensor

Step-3: Idea Prioritization



4

Decide your focus

Give each person two icons to vote which idea should your team focus on.

5 minutes

karthika

DivyaBharathi

Zennera Fathima

Nivethitha

After you collaborate

A brainstorm like this typically results in a handful of promising ideas that you can carry forward and act upon.

Quick add-ons

A **Cluster related ideas**
Look for patterns or similarities in the standout ideas. Could any be combined together to form a stronger concept? Cluster similar ideas and label each cluster with a theme.

B **Vote on the most promising ideas**
Narrow your focus to only the strongest few ideas by holding a **Voting Session**. Give each person 2 votes

Keep moving forward

2x2 Prioritization matrix
Build shared understanding and make collective decisions for moving ideas forward.
Open the template →

Storyboarding
Show existing and/or future consumer experiences through the act of sketching.
Open the template →

Pre-mortem
Harness the collective experience and wisdom of the team, before the project even starts
Open the template →

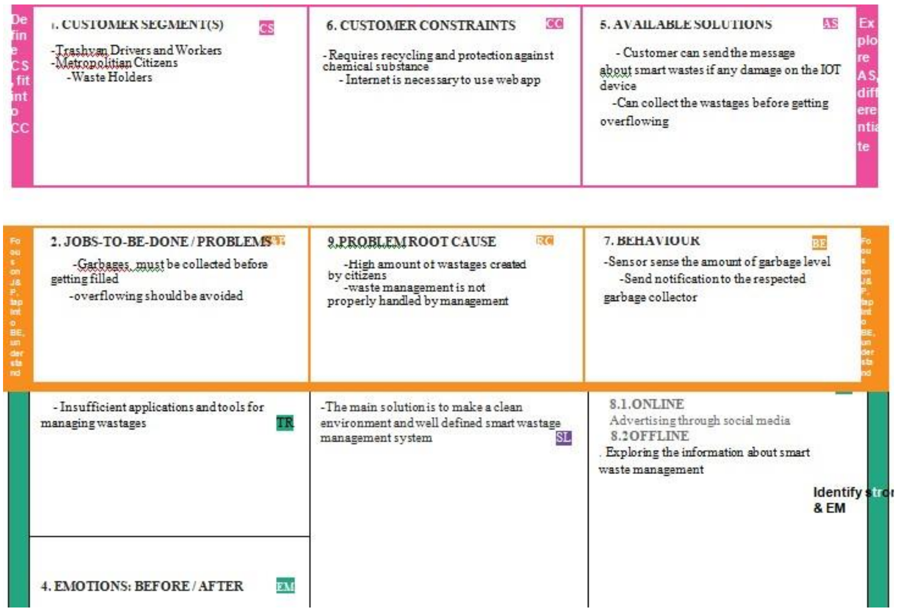
Share template feedback

3.3 Proposed Solution:

S.No.	Parameter	Description
1	Problem Statement (Problem to be solved)	✓ The manual monitoring of wastes in waste bins is a cumbersome process and utilises more human effort, time and cost. ✓ Irregular disposal of wastes causing trouble to people. ✓ Foul smell around the place with uncollected wastes or garbage.
2	Idea / Solution description	✓ This process is achieved by using a ultrasonic sensor to know the levels of garbage bin through cloud connection. ✓ Creating an app, there by the corporation of a particular locality inside a metropolitan city can check the garbage bins whether they are filled or not.
3	Novelty / Uniqueness	✓ Unlike the conventional methods for collecting garbage bins, this method tells us to use the transport only in required places ✓ To reduce the human-effort and difficulty in monitoring the garbage bins.
4	Social Impact / Customer	✓ People can experience a clean

	Satisfaction	environment. ✓ Reduces the human effort involving in the garbage disposal process. ✓ This idea will be very much beneficial for a city corporation for monitoring the cleanliness of various parts of the city.
5	Business Model (Revenue Model)	✓ This reduces a huge fuel cost to the city corporations by reducing the unwantedtransport expenses to unnecessary places. ✓ This project aims to support the municipal corporations. ✓ Provide a clean environment.
6	Scalability of the Solution	✓ A huge time is saved from frequent monitoring of garbage bins through human labours. ✓ It can be updated to automated garbage collection through vehicles. ✓ There is no need of new establishment of things. ✓ Already present garbage bins are modified slightly.

3.4 Problem solution fit:



1. CUSTOMER SEGMENT(S) CS -Trashvan Drivers and Workers -Metropolitian Citizens -Waste Holders	6. CUSTOMER CONSTRAINTS CC - Requires recycling and protection against chemical substance - Internet is necessary to use web app	5. AVAILABLE SOLUTIONS AS - Customer can send the message about smart wastes if any damage on the IOT device -Can collect the wastages before getting overflowing
2. JOBS-TO-BE-DONE / PROBLEMS J&P -Garbages must be collected before getting filled -overflowing should be avoided	9.PROBLEM ROOT CAUSE RC -High amount of wastages created by citizens -waste management is not properly handled by management	7. BEHAVIOUR BE -Sensor sense the amount of garbage level -Send notification to the respected garbage collector
- Insufficient applications and tools for managing wastages TR	-The main solution is to make a clean environment and well defined smart wastage management system SL	8.1.ONLINE Advertising through social media 8.2OFFLINE . Exploring the information about smart waste management
4. EMOTIONS: BEFORE / AFTER EM		

Define CS, fit into CC

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

Identify stro & EM

4. Requirements

4.1 Functional Requirements

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Expensive bins	✓ As we are making up bins with sensors and other costly devices , this is somewhat expensive architecture to built. ✓ And so this requires more security settings as it requires more cost if we need to rebuilt it.

FR-2	Implementing proper monitoring system	✓ All bins can be seen on the map, and you can visit them at any time via the Street View feature from Google. Bins are visible on the map as green, orange or red circles. ✓ You can see bin details in the Dashboard capacity, waste type, last measurement, GPS location and collection schedule or pick recognition.
FR-3	Separation of different kind ofwastes	✓ Separation of different kind of wastes involves people responsibility too and so, proper education need to be provided. ✓ And bins should be implemented accordingly ineach locations. ✓ And especially medical wastes should be disposed in a proper manner.
FR-4	Routing the pickup of trash	✓ Route planning for rubbish pickup is semi- automated using the tool. ✓ You are prepared to act and arrange for garbage collection based on the levels of bin fill that are now present and forecasts of approaching capacity. ✓ To find any discrepancies, compare the planned and actual routes.
FR-5	Get rid of ineffective picks	✓ Get rid of the collection of half-empty trashcans. ✓ Picks are recognised by sensors. ✓ We are able to show you how filled the bins you collect are by utilizing real-time data on fill- levels and pick recognition. ✓ The report details the bin's initial level ofbrimmingness. ✓ Any picks below 80% full that are

		inefficient are seen right away.

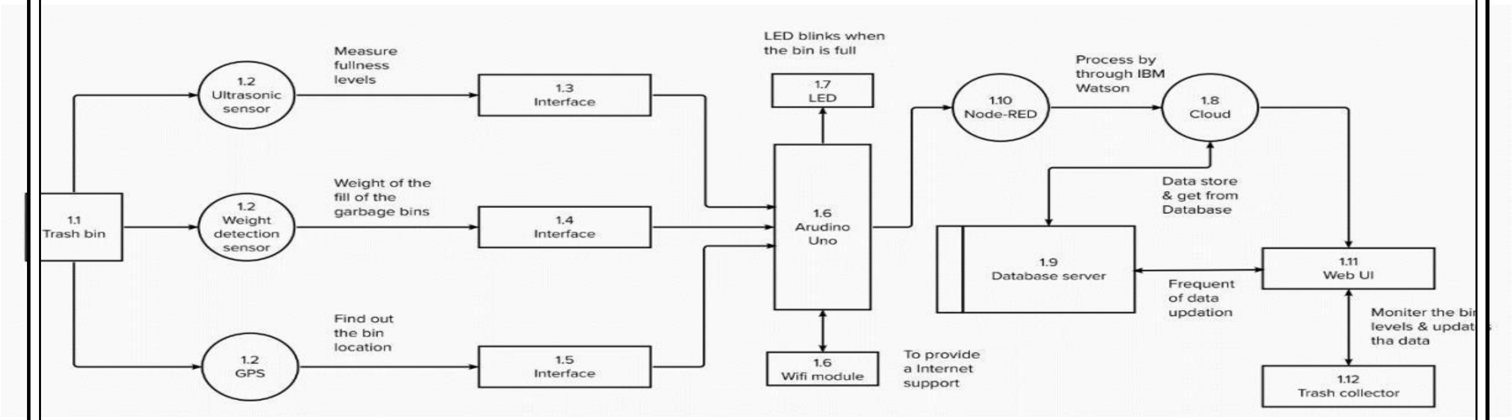
4.2 Non-Functional Requirements

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	✓ The study of customers' product usability can help designers better understand users' possible demands in waste management, behavior, and experience during the design process, which places a focus on the user experience.
NFR-2	**Security**	✓ Security ensures the level of assurance in data collection, processing and conveying. ✓ As this is totally depend upon cloud service we need to make security more particularwithout channel crash.
NFR-3	**Reliability**	✓ Creating better working conditions for waste collectors and drivers is another aspect of smart waste management. Waste collectors will use their time more effectively by attending to empty bins that need service rather than driving the same collection routes.
NFR-4	**Performance**	✓ The system consist of sensors to measure the weight of waste and the level of wasteinside the bin. ✓ Customers are provided with required datadriven and decision making prototypeswhich would help uses to monitor its performance and encounter their quires.

NFR-5	**Availability**	✓ By creating and implementing durable hardware and gorgeous software, we enablecities, companies, and nations to manage garbage more intelligently.
NFR-6	**Scalability**	✓ We have to customize the number of bins inthe town/city which we are going to monitor 24/7 a week and collect data. ✓ Smart waste management aims to optimizeresource allocation, reduce running costs, and increase the sustainability of waste service. ✓ Analytics data to manage collection routes and the placement of bins more effectively.

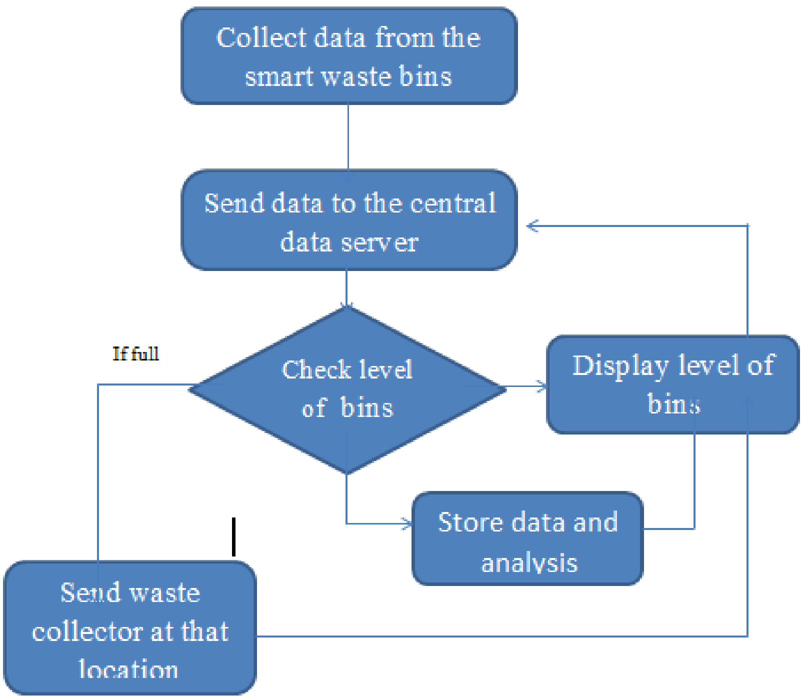
5. Project Design

5.1 Data Flow Diagram



1.1 Trash bin
1.2 Ultrasonic sensor
Measure fullness levels
1.3 Interface
1.2 Weight detection sensor
Weight of the fill of the garbage bins
1.4 Interface
1.2 GPS
Find out the bin location
1.5 Interface
LED blinks when the bin is full
1.7 LED
1.6 Arudino Uno
1.6 Wifi module
To provide a Internet support
1.10 Node-RED
Process by through IBM Watson
1.8 Cloud
Data store & get from Database
1.9 Database server
Frequent of data updation
1.11 Web UI
Moniter the bi levels & updat tha data
1.12 Trash collector

5.2 Solution & Technical Architecture


Collect data from the smart waste bins
Send data to the central data server
If full
Check level of bins
Display level of bins
Store data and analysis
Send waste collector at that location

5.3 User Stories

Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
admin	Web server login	USN-1	As a admin, I have my user name and password foe every worker and co-workers to manage them.	I can manage web account and direct workers.	High	Sprint-1
admin	Login	USN-2	As a co-admin, I'll manage other monitoring activities like garbage level monitoring, location accuracy, garbage separation and removal of waste within a scheduled time.	I can monitor garbage bins activities.	High	Sprint-1

tomer b user)	User	USN-3	Here comes the customer, he/she will have access to mobile apps or login webpages to view progress of bins and to report if any query found.	He/ she has the right to make a query if any	High	Sprint-2
customer	Worker	USN-4	The customer care executive, will try to rectify the queries from customers by contacting coadmin. If case of any critical/ emergency situation query can be conveyed to higher authority.	I can attend calls and respond people by rectifying the problem.	High	Sprint-4
Truck driver	Worker	USN-5	Here, truck driver is a worker who has particular assignments that he has to report when and where the garbage has been picked according to the daily schedule. And should update the happenings in the given website (Webpage login).	I can update my activities on site when the given task has been completed.	Medium	Sprint-5

6.Project Planning and Scheduling

6.1 Sprint Planning & Estimation

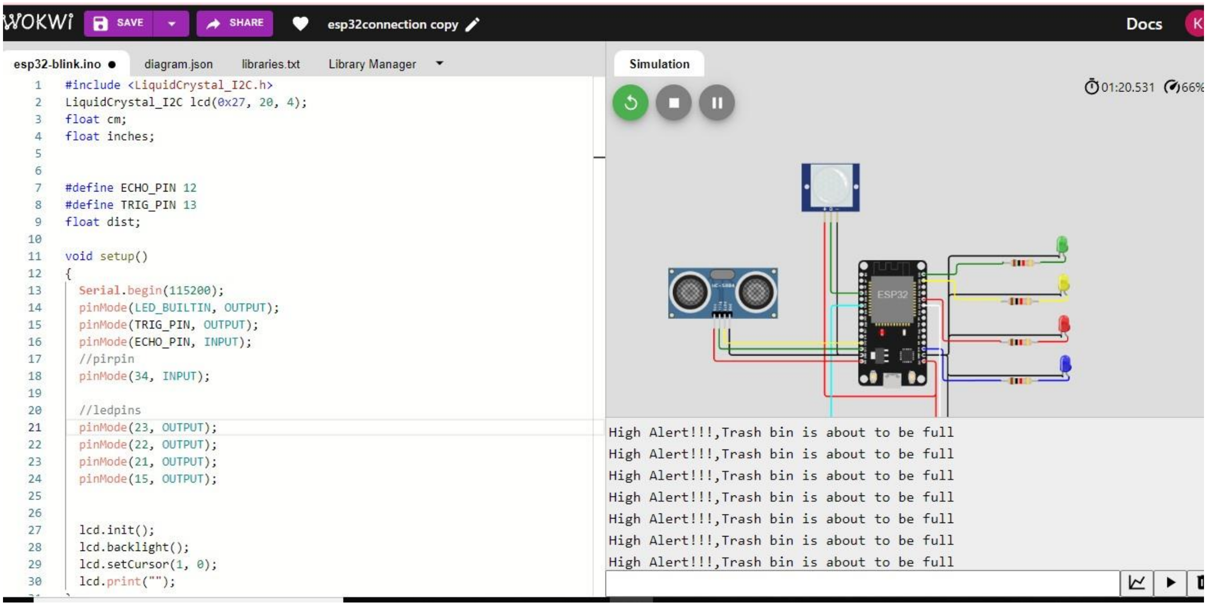
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members

Sprint-1	Registration	USN-1	As a Administrator, I need to give user id and passcode for ever workers over there in municipality	10	High	Nivedhitha. v
Sprint-1	Login	USN-2	As a Co-Admin, I’ll control the waste level by monitoring them vai real time web portal. Oncethe filling happens, I’ll notify trash truck with location of bin with bin ID	10	High	Zennera Fathima. K.A
Sprint-2	Dashboard	USN-3	As a Truck Driver, I’ll follow Co-Admin’s Instruction to reach the filling bin in short rootsand save time	20	Low	Karthika.R
Sprint-1	Dashboard	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	20	Medium	Karthika.R, Divya Bharathi.P.S
Sprint-1	Dashboard	USN-5	As a Municipality officer, I'll make sure everything is proceeding as planned and without any problems	20	High	Divya Bharathi.P.S

6.2 Sprint Delivery Scheduling

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

7.Coding and Solution

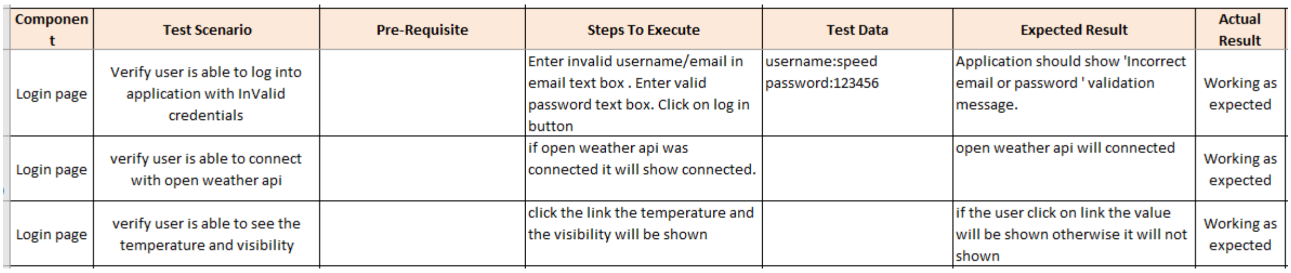


```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 20, 4);
float cm;
float inches;


#define ECHO_PIN 12
#define TRIG_PIN 13
float dist;

void setup()
{
  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  //pirpin
  pinMode(34, INPUT);

  //ledpins
  pinMode(23, OUTPUT);
  pinMode(22, OUTPUT);
  pinMode(21, OUTPUT);
  pinMode(15, OUTPUT);


  lcd.init();
  lcd.backlight();
  lcd.setCursor(1, 0);
  lcd.print("");
```

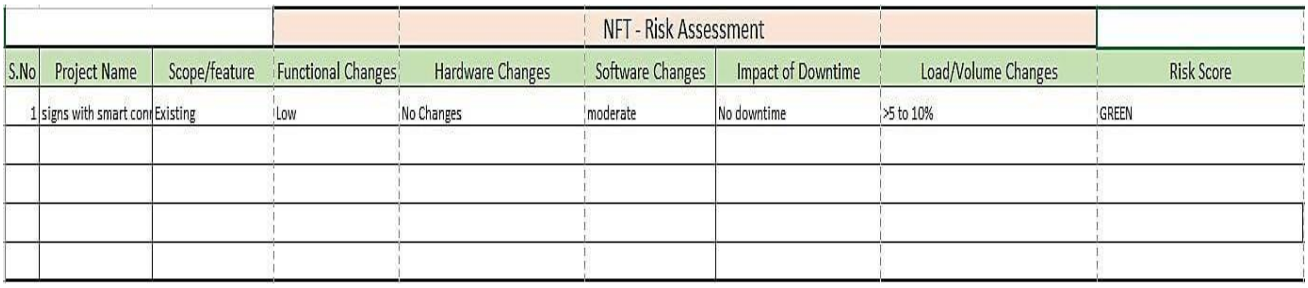
```
High Alert!!!,Trash bin is about to full
High Alert!!!,Trash bin is about to full
High Alert!!!,Trash bin is about to full
High Alert!!!,Trash bin is about to full
High Alert!!!,Trash bin is about to full
High Alert!!!,Trash bin is about to full
High Alert!!!,Trash bin is about to full
```

8.Testing

8.1 Test cases

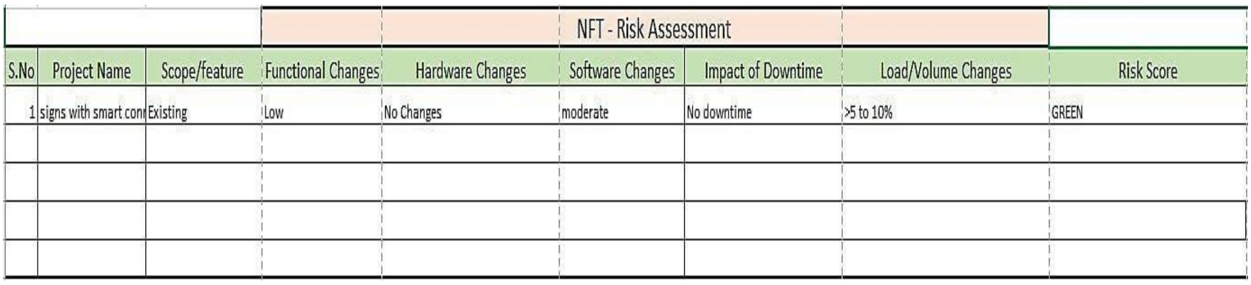


Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result
Login page	Verify user is able to log into application with InValid credentials		Enter invalid username/email in email text box . Enter valid password text box. Click on log in button	username:speed password:123456	Application should show 'Incorrect email or password ' validation message.	Working as expected
Login page	verify user is able to connect with open weather api		if open weather api was connected it will show connected.		open weather api will connected	Working as expected
Login page	verify user is able to see the temperature and visibility		click the link the temperature and the visibility will be shown		if the user click on link the value will be shown otherwise it will not shown	Working as expected



			NFT - Risk Assessment					
S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volume Changes	Risk Score
1	signs with smart con	Existing	Low	No Changes	moderate	No downtime	>5 to 10%	GREEN

Performance Testing



			NFT - Risk Assessment					
S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volume Changes	Risk Score
1	signs with smart con	Existing	Low	No Changes	moderate	No downtime	>5 to 10%	GREEN

8.2 User Acceptance Testing

1.Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [Signs with smart connectivity for better road safety] project at the time of the release to UserAcceptance Testing (UAT).

2.Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and howthey were resolved.

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

Test Case Analysis

Section	Total Cases	Not Teste d	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

Results

9.1 Performance metrics

This project used to measure garbage level and send alert message to trash collector.Reducing waste will not only protect the environment but will also save on costs or reduce expenses for disposal. In the same way, recycling and/or reusing the waste that is produced benefits the environment by lessening the need to extract resources and lowers the potential for contamination.

10. Advantages

It saves time and money by using smart waste collection bins and systems equipped with fill level sensors. As smart transport vehicles go only to the filled containers or bins. It reduces infrastructure, operating and maintenance costs by upto 30%.

1. It decreases traffic flow and consecutively noise due to less air pollution as result of less waste collection vehicles on the roads. This has become possible due to two way communication between smart dustbins and service operators.

2. It keeps our surroundings clean and green and free from bad odour of wastes, emphasizes on healthy environment and keep cities more beautiful.

3. It further reduces manpower requirements to handle the garbage collection process.

4. Applying smart waste management process to the city optimizes management, resources and costs which makes it a "smart city".

5. It helps administration to generate extra revenue by advertisements on smart devices.

10.1. Disadvantages

1. Sensor nodes used in the dustbins have limited memory size.

2. It reduces man power requirements which results into increase in unemployments for unskilled people.

3. The trainining has to be provided to the people involved in the smart waste management system.

11. Conclusion

1) optimization of the garbage collection process, reduction of labor and resource costs, increase in efficiency and comfort of citizens

2) improvement of the ecological situation in the city

3) increasing environmental awareness and motivation of the citizens;

12. Future Scope

There are several future works and improvements for the proposed system, 1. Change the system of user’s authentication and atomic lock of bins which would help in securing the bin from any kind of damage or theft. 2. Concept of green-points that would encourage the involvement of the residents or the end users making the idea successful and helping to achieve joined efforts for the waste management and hence fulfilling the idea of Swachch Bharath. 3. Having a case study or data analytics on the type and times the waste is collected on the type of days or season making the bin filling predictable and removing the dependency on electronic components and fixing the coordinates. 4. Improving graphical interfaces for the Server and complete Android applications has possibility of extending the system adding other use cases and applications for smart cities. 5. Moreover, the proposed solution is flexible and decoupled with respect to the determination of optimal number of bins and vehicles or to the algorithm that define the best route for vehicles. Therefore, future works can be made in the study of models that offer the best results in terms of decision-making.

13. Appendix

Source Code

```
import random
import time
import math
import threading



class HX711:
    def __init__(self, dout, pd_sck, gain=128):
        self.PD_SCK = pd_sck

        self.DOUT = dout

        # Last time we've been read.
        self.lastReadTime = time.time()
        self.sampleRateHz = 80.0
        self.resetTimeStamp = time.time()
        self.sampleCount = 0
        self.simulateTare = False

        # 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()

        self.GAIN = 0
        self.REFERENCE_UNIT = 1  # The value returned by the hx711 that corresponds to your reference unit
AFTER dividing by the SCALE.

        self.OFFSET = 1
        self.lastVal = long(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 convertToTwosComplement24bit(self, inputValue):
       # HX711 has saturating logic.
       if inputValue >= 0x7fffff:
          return 0x7fffff

       # If it's a positive value, just return it, masked with our max value.
       if inputValue >= 0:
          return inputValue & 0x7fffff

       if inputValue < 0:
          # HX711 has saturating logic.
          if inputValue < -0x800000:
             inputValue = -0x800000

          diff = inputValue + 0x800000

          return 0x800000 + diff


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


    def is_ready(self):
        # Calculate how long we should be waiting between samples, given the
        # sample rate.
        sampleDelaySeconds = 1.0 / self.sampleRateHz

        return time.time() >= self.lastReadTime + sampleDelaySeconds


    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

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

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

    self.lastReadTime = time.time()

    # Generate a 24bit 2s complement sample for the virtual HX711.
    rawSample = self.convertToTwosComplement24bit(self.generateFakeSample())

    # Read three bytes of data from the HX711.
    firstByte  = (rawSample >> 16) & 0xFF
    secondByte = (rawSample >> 8)  & 0xFF
    thirdByte  = rawSample & 0xFF

    # Release the Read Lock, now that we've finished driving the virtual 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:
            print("HX711().read_average(): times must >= 1!!  Assuming value of 1.")
            times = 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 an
        # arithmetic mean.
        if times < 5:
            values = int(0)
            for i in range(times):
                values += self.read_long()

            return values / 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)


    def get_value(self, times=3):
        return self.read_average(times) - self.OFFSET


    def get_weight(self, times=3):
        value = self.get_value(times)
        value = value / self.REFERENCE_UNIT
        return value


    def tare(self, times=15):
        # If we aren't simulating Taring because it takes too long, just skip it.
        if not self.simulateTare:
            return 0

        # Backup REFERENCE_UNIT value
        reference_unit = self.REFERENCE_UNIT
        self.set_reference_unit(1)

        value = self.read_average(times)

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

        self.set_offset(value)

        # Restore the reference unit, now that we've got our offset.
        self.set_reference_unit(reference_unit)

        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:
            print("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:
            print("Unrecognised bit_format: \"%s\"" % bit_format)



    def set_offset(self, offset):
        self.OFFSET = offset


    def get_offset(self):
        return self.OFFSET


    def set_reference_unit(self, reference_unit):
        # Make sure we aren't asked to use an invalid reference unit.
        if reference_unit == 0:
            print("HX711().set_reference_unit(): Can't use 0 as a reference unit!!")
            return

        self.REFERENCE_UNIT = reference_unit


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

        # Wait 100us for the virtual HX711 to power down.
        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()

        # Wait 100 us for the virtual 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()

        # Mark time when we were reset.  We'll use this for sample generation.
        self.resetTimeStamp = time.time()


    def generateFakeSample(self):
       sampleTimeStamp = time.time() - self.resetTimeStamp

       noiseScale = 1.0
       noiseValue = random.randrange(-(noiseScale * 1000),(noiseScale * 1000)) / 1000.0
       sample     = math.sin(math.radians(sampleTimeStamp * 20)) * 72.0

       self.sampleCount += 1

       if sample < 0.0:
          sample = -sample

       sample += noiseValue

       BIG_ERROR_SAMPLE_FREQUENCY = 142
       ###BIG_ERROR_SAMPLE_FREQUENCY = 15
       BIG_ERROR_SAMPLES = [0.0, 40.0, 70.0, 150.0, 280.0, 580.0]

       if random.randrange(0, BIG_ERROR_SAMPLE_FREQUENCY) == 0:
          sample = random.sample(BIG_ERROR_SAMPLES, 1)[0]
          print("Sample %d: Injecting %f as a random bad sample." % (self.sampleCount, sample))

       sample *= 1000

       sample *= self.REFERENCE_UNIT

       return int(sample)
```

```
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()
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-9670-1659067550