



SMART WASTE MANAGEMENT SYSTEM FOR METROPOLITIAN CITIES

NALAIYA THIRAN PROJECT BASED LEARNING

On

**PROFESSIONAL READINESS FOR INNOVATION,
EMPLOYABILITY AND ENTREPRENEURSHIP**

A PROJECT REPORT

KARTHIKRAJAN L S 19106056

LOGESWARAN R 19106064

MADHU VARSHINI S K 19106066

PON MALAR S 19106082

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

HINDUSTHAN COLLEGE OF ENGINEERING AND TECHNOLOGY

Approved by AICTE, New Delhi, Accredited with 'A' Grade by NAAC

(An Autonomous Institution, Affiliated to Anna University, Chennai)

COIMBATORE – 641 032

November 2022



HINDUSTHAN COLLEGE OF ENGINEERING AND TECHNOLOGY

Approved by AICTE, New Delhi, Accredited with 'A' Grade by NAAC
(An Autonomous Institution, Affiliated to Anna University, Chennai)
COIMBATORE – 641 032



INTERNAL MENTOR

Dr.J.Ramya

Associate Professor,

Department of Electronics and Communication Engineering

Hindusthan College of Engineering and Technology,

Coimbatore-641 032

INDUSTRY MENTOR

DINESH(IBM)

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that project report “**SMART WASTE MANAGEMENT SYSTEM FOR METROPOLITAN CITIES**” is the bonafide work of “**KARTHIKRAJAN, LOGESHWARAN, MADHU VARSHINI, PON MALAR** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr.P.Vijayalakshmi

HEAD OF THE DEPARTMENT

Professor and Head,
Department of ECE,
Hindusthan College of Engineering
and Technology,
Coimbatore - 641 032

SIGNATURE

Dr.J.Ramya

SUPERVISOR

Associate Professor,
Department of ECE,
Hindusthan College of Engineering
and Technology,
Coimbatore - 641 032.

Submitted for Project Viva-Voice conducted on

INTERNAL EXAMINER

EXTERNAL EXAMINER

PROJECT CONTENT

1. INTRODUCTION

- a. Project Overview
- b. Purpose

2. LITERATURE SURVEY

- a. Existing problem
- b. References
- c. Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

- a. Empathy Map Canvas
- b. Ideation & Brainstorming
- c. Proposed Solution
- d. Problem Solution fit

4. REQUIREMENT ANALYSIS

- a. Functional requirement
- b. Non-Functional requirements

5. PROJECT DESIGN

- a. Data Flow Diagrams
- b. Solution & Technical Architecture
- c. User Stories

6. PROJECT PLANNING & SCHEDULING

- a. Sprint Planning & Estimation
- b. Sprint Delivery Schedule

- c. Reports from JIRA

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

- a. Feature 1
- b. Feature 2
- c. Database Schema (if Applicable)

8. TESTING

- a. Test Cases
- b. User Acceptance Testing

9. RESULTS

- a. Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

INTRODUCTION

1.1 PROJECT OVERVIEW

Solid trash is becoming major threat to our emerging nations. Recent population growth and rural-urban migration may be to blame for this rise. Non-renewable materials that we use every day to suit our requirements and subsequently discard make up garbage. Garbage production increases dramatically along with growth in the usage of paper, textiles, bottles, and product packaging. The form and kind of solid waste relies on a variety of variables, including the level of living and way of life of the local populace as well as the local natural resources. Urban garbage is divided into two categories: organic waste and inorganic waste. Three further categories of organic waste exist: nontransferable, ferment-able, and putrescence. A common illustration is the overflowing garbage cans that are seen throughout, which cause pollution of the environment. As a result, there are more infections since there is space for insects to breed. Regardless of substance, origin, or potential hazards, solid waste needs to be managed systematically in order to maintain a high standard of living and environmental best practices. It is vital to include solid waste management in environmental planning because it is a crucial component of our environmental hygiene. Recent developments in computing have spawned fresh ideas and opportunities, such as the Internet of Things, which allows things (embedded systems) connected to the internet to be controlled and interacted with online. In 1999, Kevin Ashton, a former head of MIT's Auto-ID Center, coined the phrase "Internet of Things" (IoT). IoTs aim to connect the surrounding things via wired and wireless networks with the help of people. The object engages in communication and information sharing in order to offer users an advanced, intelligent service. When it comes to the suggested solid waste management system, the bins are linked to the internet to transmit real-time information about the bin status. In order to prevent living conditions that are unclean, the recent rapid expansion in population has resulted in a greater amount of trash disposals. When the system is put into use, the bin is connected to a micro controller-based system that has ultrasonic sensors and a Wi-Fi module. The data would be received, analyzed, and processed in the Thing Speak cloud, which shows the amount of trash in the bin on a graph on its website. The primary motivation for solid waste management is the mitigation and eradication of waste materials' detrimental effects on the environment and human health, which ultimately improves quality of life. As cloud computing has been used in other fields like, this is a newer breakthrough. To determine how full each container is with solid trash, ultrasonic sensors are used. The sensor's data is subsequently sent through a Wi-Fi communication link to an IoT cloud platform called Thing Speak. The system sends the proper notification message (in the form of a tweet) for each set fill level to inform the essential authorities and the concerned citizen(s) to take the needed action. Additionally, the fill level is continuously observed on Thing Speak.

1.2 PURPOSE

Solid waste disposal without consideration is a significant problem in the urban areas of the majority of developing nations, and it seriously jeopardizes the residents' ability to live a healthy lifestyle. Both the local government and the residents will benefit from having access to trustworthy data on the state of solid trash at various points throughout the city for managing the threat. In this study, the Internet of Things (IoT) and cloud computing technologies are used to create an intelligent solid waste monitoring system. Ultrasonic sensors are used to measure the solid waste fill levels in each of the containers, which are placed in strategic locations around the community. The sensor data is sent over a Wireless Fidelity (Wi-Fi) communication channel to the Thing-speak IoT cloud platform. The system sends the required notification message (in the form of a tweet) to inform the proper authorities and the concerned citizen(s) of the need for action based on the fill level. Additionally, the fill level is continuously observed on Thing-speak. The system's performance demonstrates that the suggested method can be helpful for effective trash management in connected and smart communities.

LIERATURE SURVEY

2.1 EXISTING PROBLEM

The ever-growing network is referred to as the "Internet of Things" (IoT). The number of internet-connected gadgets in use globally today. In spite of the increasing Internet of Things sector, the current Covid-19 pandemic, and it is estimated the number of IoT connections will reach 30 billion by the year 2025[1].

The enabling technologies that hasten the creation and implementation of domain-specific IoT systems include advanced smart sensors, cloud computing, big data, lightweight communication protocols, open-source server software, and web development tools[2].

The physical and digital worlds can be more seamlessly connected by these interconnected gadgets, advancing productivity, culture, and quality of life. Smart Homes, Smart Cities, Agriculture, Wearable, Smart Grids, Industrial Internet Telehealth, and Smart Supply chain Management are just a few of the domain-specific applications for which IoT has already demonstrated viable techniques[3].

Beginning with waste that is created by city dwellers and dumped in trash cans as soon as it is created, the traditional waste management process begins. Municipal department trucks collect the trash and deliver it to the recycling facilities according to a set schedule. Municipalities and businesses find it difficult to keep track of the outdoor bins, such as when to clean them or whether they are fully or partially filled. The management of these wastes through prevention, tracking, and treatment is one of the most urgent concerns of our time[4].

2.01 billion tonnes of urban solid garbage are produced globally each year, 33% of which are not treated in an environmentally responsible manner. Global trash is predicted to exceed 3.40 billion tonnes by 2050, more than double the rate of population growth during that time[5].

With today's technology, it is no longer necessary to manually inspect waste in bins, which is a time-consuming process that costs more in terms of labour, money, and time[6].

To solve the aforementioned shortcomings of traditional waste management systems, a number of WSN and IoT-based remote monitoring systems have been created and implemented[11].

Some of the monitoring systems used short-range wireless networking technologies like Bluetooth, Infrared, ZigBee, and Wi-Fi to track the bins[8].

Due to the benefits of IoT services, researchers have conducted a number of waste management studies centred on IoT technologies to address the aforementioned problems with solid waste management. The necessity of using waste management techniques and sustainability principles is increased by the fact that certain businesses, such as those in the construction and food processing sectors, constantly create a part of garbage with noticeable residue[9].

A few studies have also discussed smart bin monitoring systems that use NB-IoT, Sigfox, and Lo-Ra wide area networks[10].

2.2 REFERENCE

1. Song, Y.; Yu, F.R.; Zhou, L.; Yang, X.; He, Z. Applications of the Internet of Things (IoT) in Smart Logistics: A Comprehensive Survey. *IEEE Internet Things J.* 2020, 8, 4250–4274.
2. Jino Ramson, S.R.; Vishnu, S.; Shanmugam, M. Applications of internet of things (IoT)—An Overview. In Proceedings of the 2020 5th International Conference on Song, Y.; Yu, F.R.; Zhou, L.; Yang, X.; He, Z. Applications of the Internet of n Devices, Circuits and Systems (ICDCS), Coimbatore, India, 5–6 March 2020.
3. Yang, Q.; Wang, H. Privacy-Preserving Transactive Energy Management for IoT-aided Smart Homes via Blockchain. *arXiv* 2021, arXiv:2101.03840.
4. Maria, C.; Góis, J.; Leitão, A. Challenges, perspectives, of, greenhouse, gases, emissions, from, municipal, solid, waste, management in Angola. *Energy Rep.* **2020**, 6, 364–369. .
5. Kaza, S.; Yao, L.; Bhada-Tata, P.; Van Woerden, F. *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050*; World Bank Publications: Washington, DC, USA, 2018.
6. Venkateela, L.K. Status and challenges of solid waste management in Tirupati city. *Mater. Today Proc.* **2020**, 33, 470–474.
7. Prabakar, D.; Lakshmy, N.; Vishnu, S. Dynamic Channel State Information based Relay Selection in Device-to-Device Communication. In Proceedings of the 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2–4 July 2020.
8. Suresh, N.; Limbo, A.; Hashiyana, V.; Ujakpa, M.M.; Nyirenda, C. An internet of things (IoT) based solid waste monitoring system. In Proceedings of the 2nd International Conference on Intelligent and Innovative Computing Applications, Online, 24–25 September 2020.
9. Amaral, R.E.C.; Brito, J.; Buckman, M.; Drake, E.; Ilatova, E.; Rice, P.; Sabbagh, C.; Voronkin, S.; Abraham, Y.S. Waste Management and Operational Energy for Sustainable Buildings: A Review. *Sustainability* **2020**, 12, 5337.
10. Harith, M.Z.M.Z.; Hossain, M.A.; Ahmedy, I.; Idris, M.Y.I.; Soon, T.K.; Noor, R.M. Prototype Development of IoT Based Smart Waste Management System for Smart City. *IOP Conf. Ser. Mater. Sci. Eng.* **2020**, 884, 012051.

2.3 PROBLEM STATEMENT DEFINITION

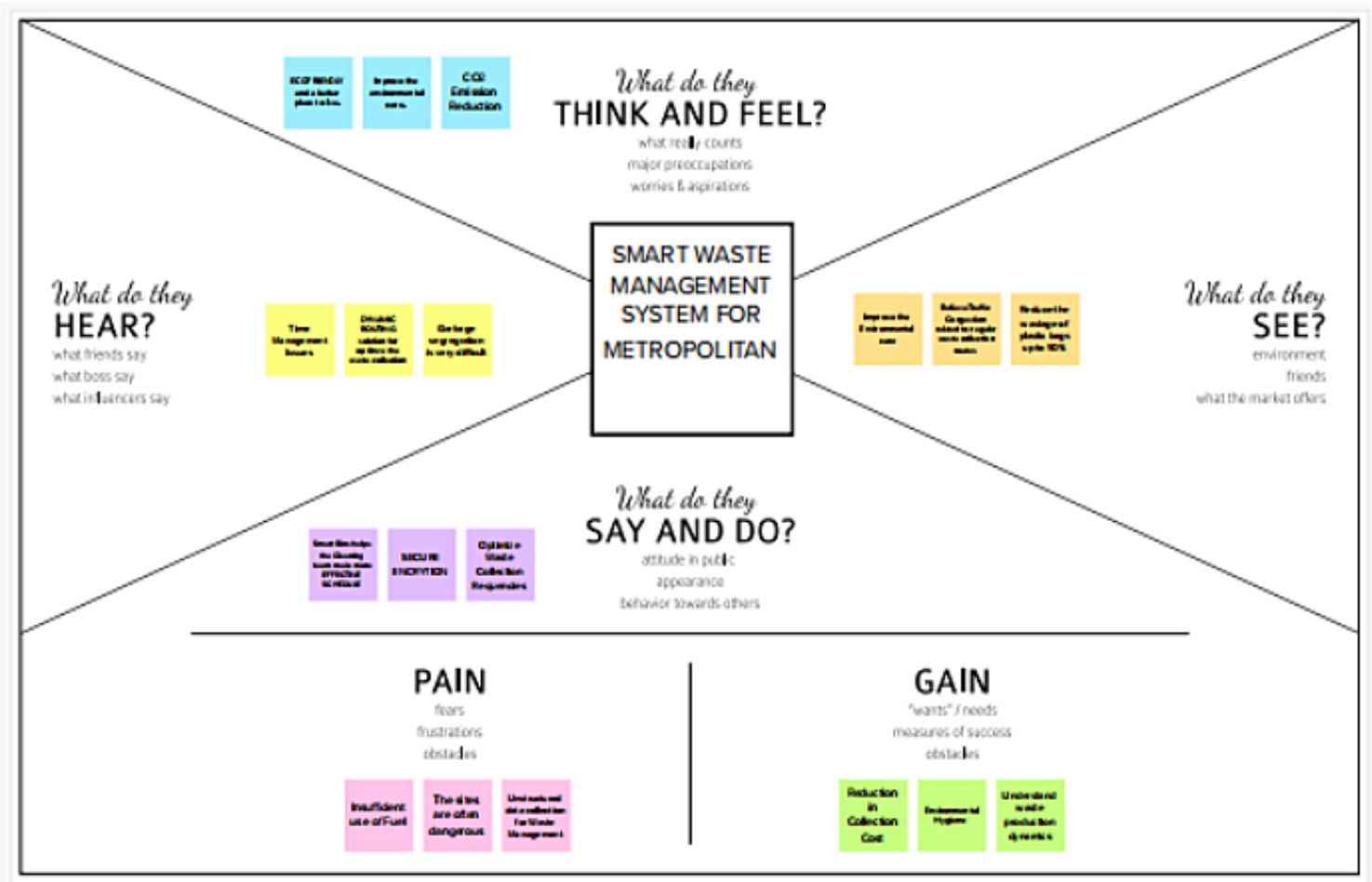
Team Id	PNT2022TMID10141
Project Name	Smart waste management system for Metropolitan Cities



Problem Statement (PS):	Waste management involves the regular collection, transportation as well as processing and disposal or recycling and monitoring of different types of waste materials. These services can save your business a considerable amount of money, and can also prevent the environment from being harmed.
I am (CIVILIAN)	A civilian, who uses trash bins to dump the wastes on regular basis.
I'm trying to	Dump wastes regularly
But	Trash bins placed locally is overfilled with garbage and it may be contaminated with bacteria and viruses
Because	Proper maintenance is not taken care of regularly, and the garbage collector is not aware of the bins that are full and unhygienic.
Which makes me feel	People infected with disease and virus, foul smell, bad environment and affects ecosystem.


IDEATION & PROPOSED SOLUTION

3.1 Empathy Map



3.2 Ideation & Brainstorming

Team Id	PNT2022TMID10141
Project Name	Smart waste management system for Metropolitan Cities



Brainstorm ideas for your presentation

Use this handy table to generate ideas for your presentation. You can use it to generate ideas for your presentation, or you can use it to generate ideas for your presentation.

1. Brainstorm ideas for your presentation

2. Prioritize your ideas

3. Develop your ideas

4. Present your ideas

Brainstorm ideas for your presentation

Use this handy table to generate ideas for your presentation. You can use it to generate ideas for your presentation, or you can use it to generate ideas for your presentation.

1. Brainstorm ideas for your presentation

2. Prioritize your ideas

3. Develop your ideas

4. Present your ideas

Develop your presentation structure

Once you have generated ideas for your presentation, you can use this table to develop your presentation structure. You can use it to generate ideas for your presentation, or you can use it to generate ideas for your presentation.

1. Develop your presentation structure

2. Prioritize your ideas

3. Develop your ideas

4. Present your ideas

Develop your presentation structure

Once you have generated ideas for your presentation, you can use this table to develop your presentation structure. You can use it to generate ideas for your presentation, or you can use it to generate ideas for your presentation.

1. Develop your presentation structure

2. Prioritize your ideas

3. Develop your ideas

4. Present your ideas

Develop your presentation structure

Once you have generated ideas for your presentation, you can use this table to develop your presentation structure. You can use it to generate ideas for your presentation, or you can use it to generate ideas for your presentation.

1. Develop your presentation structure

2. Prioritize your ideas

3. Develop your ideas

4. Present your ideas

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Proposed Statement (Problem to be solved)	To detect waste in the bins using Smart bins and Sensors.
2.	Idea / Solution Description	Utilizing sensors, sophisticated monitoring systems, and mobile applications, smart waste management aims to address the aforementioned solid waste management issues.
3.	Novelty / Uniqueness	Reduce, Recycle and Reuse
4.	Social Impact / Customer Satisfaction	A reduction in the waste collections needed by up to 80%, resulting in less manpower, emissions, fuel use and traffic congestion.
5.	Business Model (Revenue Model)	By offering different waste management and disposal services as well as recycling options to clients in the municipal, commercial, industrial, and residential sectors, Waste Management makes money.
6.	Scalability of the solution	The Smart Garbage Monitoring System is very useful in real time, it keeps the environment hygiene and easy work so it will reach worldwide.

3.4 Problem Solution Fit

Define CS, fit into	1. CUSTOMER SEGMENT (CS) <p>Waste can be recycled if collected and managed efficiently.</p> <p>A reduction in the number of waste collections needed by up to 80%, resulting in less manpower, emissions, fuel use and traffic congestion.</p>	6. CUSTOMER (CC) <p>In this modern world, creating a healthy and sanitized environment for the upcoming generations.</p>	5. AVAILABLE SOLUTIONS (AS) <p>The mode of communication is through digital platform. Collecting the data from garbage can periodically.</p> <p>Through the mobile application users can also view the capacity left in the trash bin.</p>	Explore AS
	Focus on J&P, tap into BE, understand	2. JOBS-TO-BE-DONE / PROBLEMS (J&P) <p>The people in a community throw their waste in nearby garbage can, by that the local garbage collector get notified how much left capacity of garbage can so it can be emptied time to time.</p> <p>Assisting with planning and decision-making; Setting waste reduction, recycling or diversion, objectives and targets; Identifying waste generation and recycling trends; Determining the viability and capacity of existing solid</p>	4. PROBLEM ROOT CAUSE (RC) <p>Collecting waste material manually is not efficient and hygienic to solve this problem we use the smart waste collection method.</p>	
Identify strong TR & EM		3. TRIGGERS (TR) <p>This would help to maintain a polluted free environment for children and adult. Helps to live a healthy disease-free life</p> <p>8. EMOTIONS BEFORE / AFTER (EM)</p> <p>Homes physical habitats, transports chemical pollutants, threatens aquatic life</p> <p>clean air, fresh water, medicines and food security</p>	10. YOUR SOLUTION (SL) <p>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. The fill level of solid waste in each of the containers, which are strategically situated across the communities, is detected using ultrasonic sensors. A Wireless Fidelity (Wi-Fi) communication link is used to transmit the sensor data to an IoT cloud platform known as ThingSpeak. Depending on the fill level, the system sends appropriate notification message (in form of tweet) to alert relevant authorities and concerned citizen(s) for necessary action. Also, the fill level is monitored on ThingSpeak in real-time. The system performance shows that the proposed solution may be found useful for efficient waste management in smart and connected communities.</p>	9. CHANNELS of BEHAVIOUR (CH) <p>Offline: The IoT-based smart waste management not only helps in modernizing the conventional garbage collecting method.</p> <p>Online: IoT-based smart waste management is also beneficial in terms of environmental issues. It can help the garbage collector to efficiently use fuel and other resources available.</p>

REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Expensive Garbage bins	This architecture is somewhat expensive to build because we are creating bins with expensive sensors and other devices. Therefore, this calls for more security settings and would be more expensive to rebuild.
FR-2	Monitoring System Implementation and Activities	The Google Street View function allows you to visit any bin at any time. All bins are shown on the map. Bins appear as green, orange, or red circles on the map. The Dashboard displays information about each bin, including its capacity, waste type, most recent measurement, GPS location, and pick-up schedule.
FR-3	Guide waste collection routes	Planning is crucial, since we must establish specific routes and sites where bins are collected after they are full. So, clearly sketch out the routes that the truck that collects trash must take. If everyone has a clear plan, there is no need to waste time and fuel looking for places.
FR-4	Separation of all kinds of Waste	Separating various types of garbage requires human responsibility, hence appropriate education must be offered. And dumpsters ought to be placed where they are needed in each site. And in particular, medical wastes need to be disposed of properly.

4.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The use of sensors to gauge the trash can's fill level is the current state of technology in the field of smart waste management. Sent to the cloud for additional processing and analysis, measured data. Truck routes may be made more efficient and trash collection can be scheduled by using this data. IoT

		<p>gadget confirms usability is a unique and important factor.</p> <p>Analysis of user requirements is crucial because it can help the design's quality.</p>
NFR-2	Security	<p>The degree of assurance in data gathering, processing, and transmission is ensured by security. Given that this is entirely dependent on cloud services, we must increase security with greater care to prevent channel crashes.</p>
NFR-3	Reliability	<p>Bettering the working conditions for waste collectors is a key component of smart waste management. Waste collectors can work more productively by attending to bins that require service rather than driving the same collection routes and filling empty bins. By looking after trash cans and keeping an eye on bin activities, this method is more dependable at all costs.</p>
NFR-4	Performance	<p>The Smart Sensors assess the fill levels in bins (along with other data) numerous times per day using ultrasound technology. The sensors transmit data to Sensor's Smart Waste Management Software System, a potent cloud-based platform with data-driven daily operations and a waste management app, using a range of IoT networks (NB IoT, GPRS). Customers are given the necessary data-driven and decision-making prototypes, which let users track performance and address customer needs.</p>
NFR-5	Availability	<p>The term "availability" refers to both already existing solutions and new renovation technologies that we include into the system that we are currently creating from scratch.</p> <p>Users were able to operate this system easily because it had a wide range of user-accessible solutions, including sensors, GPS detectors, and other devices. The term "availability" refers to both already existing solutions and new renovation technologies that we include into the system that we are currently creating from scratch.</p>
NFR-6	Scalability	<p>The number of bins in the town or city that we will monitor around-the-clock seven days a week and gather data from must be customized. Therefore, we must count all of the bins and provide services to each bin according to a proper rotational schedule.</p>

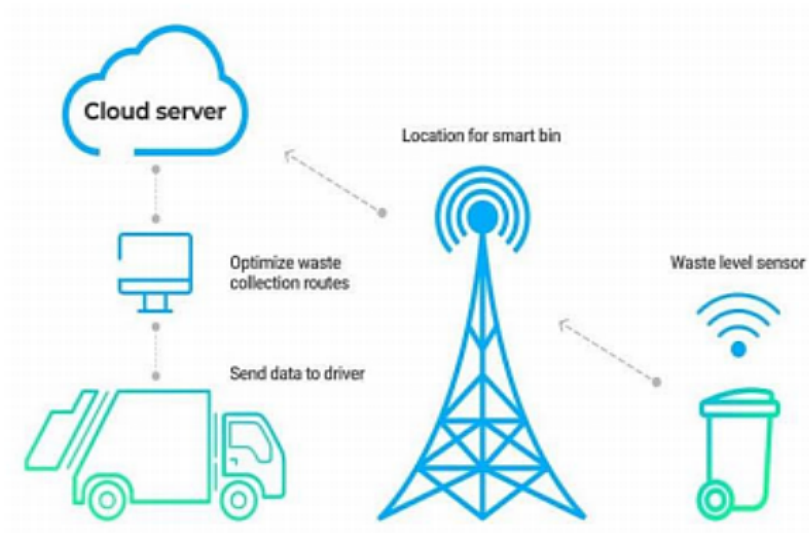
PROJECT DESIGN

5.1 Data Flow Diagrams

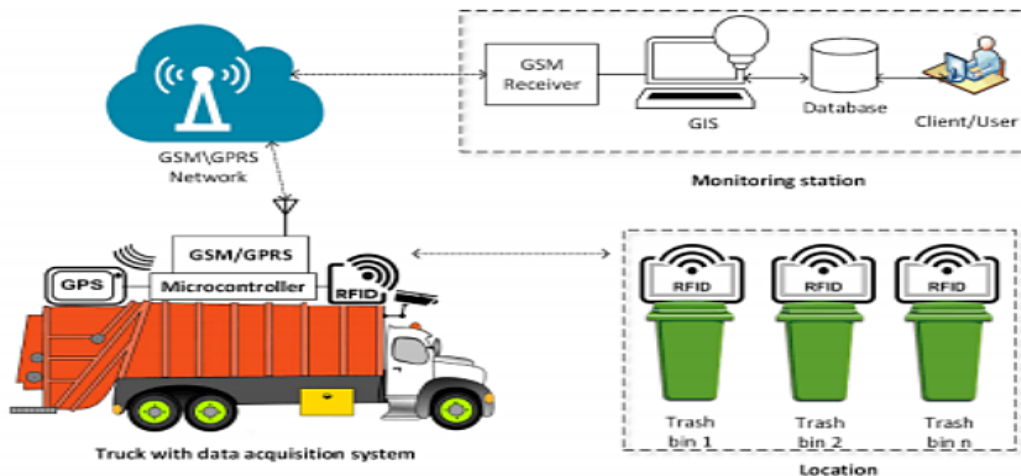
Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

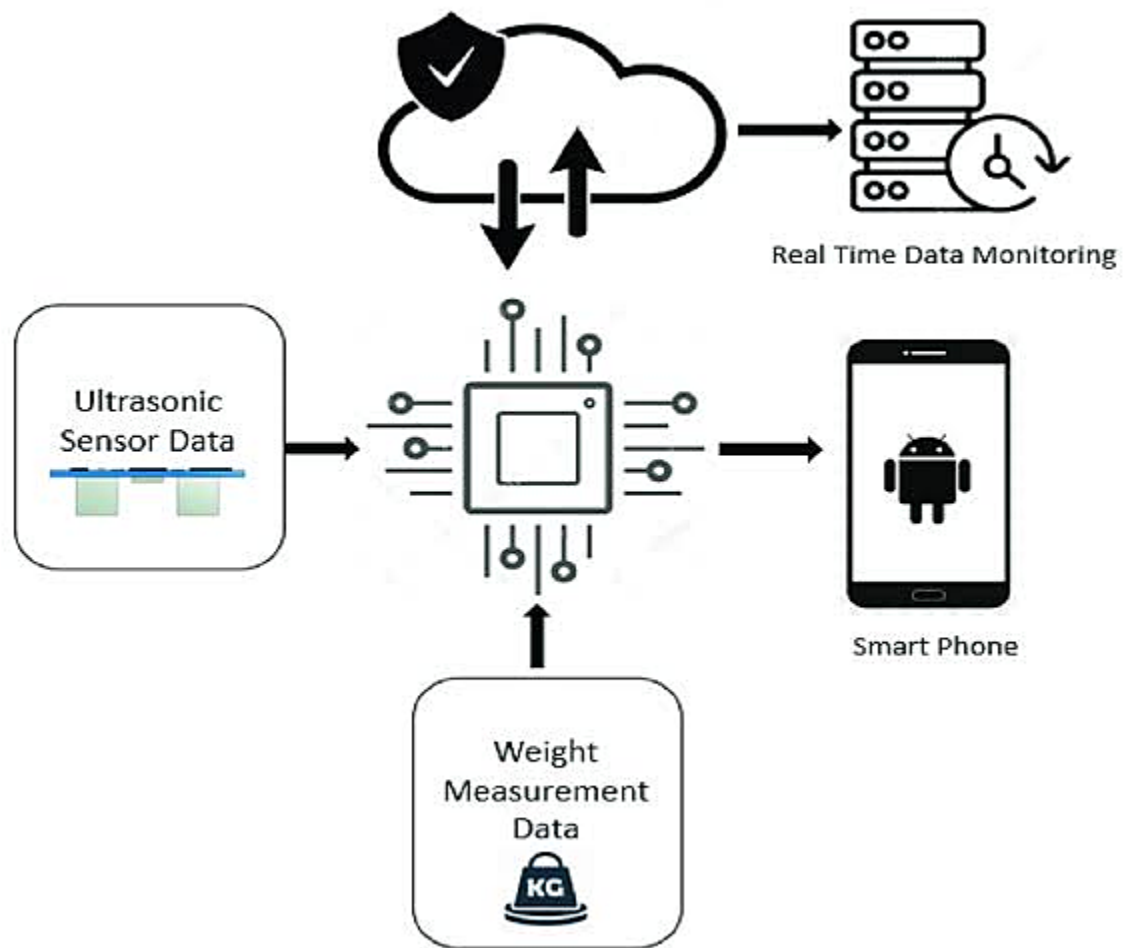
Example: Simple diagram



Example: Smart garbage bins



5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Admin (Manager)	Web server login	USN-1	Manager, Username and Password can be accessible for worker and co-workers to manage them.	I can deal with web account and assist workers.	High	Sprint-1
Co-admin (Assistant Manager)	Login	USN-2	As a Assistant Manager, I'll manage other monitoring system in garbage bins, accuracy of location, separation and removal of waste in the scheduled time.	I can monitor garbage bins activities	High	Sprint-2
Customer (Web user)	User	USN-3	As a Assistant Manager, I'll manage other monitoring system in garbage bins, accuracy of location, separation and removal of waste in the scheduled time.	The customer is allowed to submit any questions..	High	Sprint-3
Customer (Care Executive)	Worker 1	USN-4	The customer care executive, will attempt to resolve customer inquiries by getting in touch with the co-admin. Inquiries about serious or urgent situations should be directed to higher authorities.	Customer service can reply to calls and help consumers by fixing the issue.	High	Sprint-4

Truck driver (Employee)	Worker 2	USN-5	Here, truck driver is a worker who, in accordance with the daily timetable, has Specific responsibilities that he must report when and where the trash has been picked up. and ought to update the events on	When the assigned task is over, I can provide an update on my activities both on-site and on time.	Moderate	Spring-5
----------------------------	----------	-------	--	--	----------	----------

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Title	Description	Details
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers, research publication etc .	28 SEPTEMBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem Statements.	24 SEPTEMBER 2022
Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	25 SEPTEMBER 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	23 SEPTEMBER 2022
Problem Solution Fit	Prepare problem - solution fit Document.	30 SEPTEMBER 2022
Solution Architecture	Prepare solution architecture Document.	28 SEPTEMBER 2022
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry to exit).	20 OCTOBER 2022
Functional Requirements	Prepare the functional requirement document.	08 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	09 OCTOBER 2022
Technology Architecture	Prepare the technology architecture diagram.	10 OCTOBER 2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	22 OCTOBER 2022
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS

6.2 Sprint Delivery Schedule

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

Use the below template to create product backlog and sprint schedule

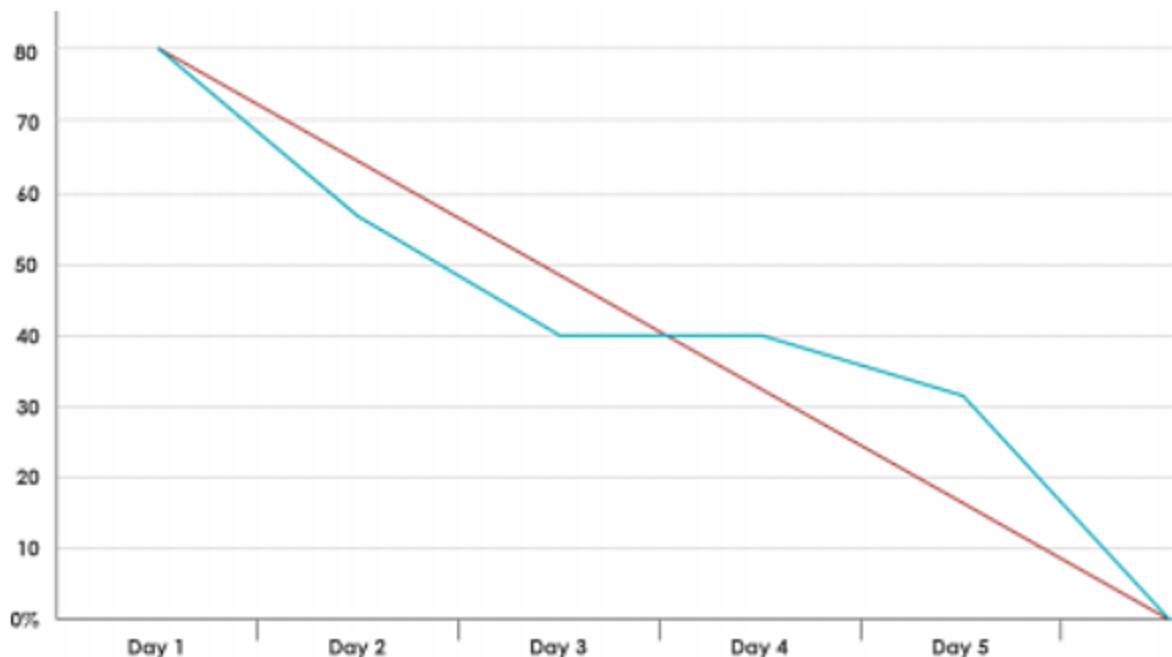
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	LS.KARTHIK RAJAN
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	LOSESHWARA N.R
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	S.KMADHU VARHINI
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	S.PON MALAR
Sprint-1	Login	USN-5	As a user, I can log into the application by Entering email & password	1	High	LS.KARTHIK RAJAN

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	30	30 OCT 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	49	6 NOV 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	50	7 NOV 2022

Velocity:

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

Burndown Chart:



6.3 Reports from JIRA

		T	NOV	DEC	JAN
<div> <div>SWMSFMC-1 Sprint 1</div> <div> <div>✓ SWMSFMC-5 DELIVERY OF SPRINT-1.</div> <div>✓ SWMSFMC-6 ESP32-blink.ino</div> <div>✓ SWMSFMC-7 ESP32-json</div> <div>✓ SWMSFMC-8 libraries</div> <div>✓ SWMSFMC-9 Sensor workup Connection</div> </div> </div>	DONE				
<div> <div>SWMSFMC-2 Sprint 2</div> <div> <div>✓ SWMSFMC-10 Delivery of Sprint-2</div> <div>✓ SWMSFMC-11 Diagram.json</div> <div>✓ SWMSFMC-12 Libraries.txt</div> <div>✓ SWMSFMC-13 Sensor code</div> <div>✓ SWMSFMC-14 wokwi-project</div> </div> </div>	DONE				
<div> <div>SWMSFMC-3 Sprint 3</div> <div> <div>✓ SWMSFMC-15 Delivery of Sprint-3</div> <div>✓ SWMSFMC-16 diagram.json</div> <div>✓ SWMSFMC-17 esp32-blink.ino</div> <div>✓ SWMSFMC-18 libraries</div> <div>✓ SWMSFMC-19 wokwi-project</div> </div> </div>	DONE				
<div> <div>SWMSFMC-4 Sprint 4</div> <div> <div>✓ SWMSFMC-20 Delivery of Sprint-4.</div> <div>✓ SWMSFMC-21 diagram.json</div> <div>✓ SWMSFMC-22 esp32-blink.ino</div> <div>✓ SWMSFMC-23 libraries</div> <div>✓ SWMSFMC-24 wokwi-project</div> </div> </div>	DONE				

CODING & SOLUTIONING

(Explain the features added in the project along with code)

7.1 Feature

SENSOR SYSTEM

Electronic devices that detect and react to a certain type of input from the physical environment are called sensors and modules (which also have additional electronic circuitry in addition to the sensor). A crucial part of the internet of things is played by sensors (IoT). They make it feasible to develop an ecosystem for gathering information about a particular environment and processing it so that it may be monitored, managed, and controlled more effectively.

SENSOR PROGRAM

```
from future import with_statement
import logging
from logging.config import fileConfig

from sqlalchemy import
engine_from_config
from sqlalchemy import pool
from flask import current_app
from alembic import context
# this is the Alembic Config object, which provides
# access to the values within the .ini file in use.config =
context.config
# Interpret the config file for Python
logging.# This line sets up loggers basically.
fileConfig(config.config_file_name)
logger = logging.getLogger('alembic.env')# add your
```

```

model's MetaData object here # for
'autogenerate' support
# from myapp import mymodel#
target_metadata =
mymodel.Base.metadata
config.set_main_option(
'sqlalchemy.url',
str(current_app.extensions['migrate'].db.engine.url).replace('%', '%%'))
target_metadata =
current_app.extensions['migrate'].db.metadata# other values from the
config, defined by the needs of env.py,
# can be acquired:
# my_important_option =
config.get_main_option("my_important_option")# ... etc.def
run_migrations_offline():
"""Run migrations in 'offline' mode.

This configures the context with just a URL and not an
Engine, though an Engine is acceptable here as well. By
skipping the Engine creation
we don't even need a DBAPI to be available.

Calls to context.execute() here emit the given string to the script output.
"""
url = config.get_main_option("sqlalchemy.url")
context.configure(
url=url, target_metadata=target_metadata, literal_binds=True
)
with context.begin_transaction():context.run_migrations()def
run_migrations_online():
"""Run migrations in 'online' mode. In this

```

```

scenario we need to create an
Engine and associate a connection with the
context. """
# this callback is used to prevent an auto-migration from being
  generated # when there are no changes to the schema
# reference: http://alembic.zzzcomputing.com/en/latest/cookbook.html
def process_revision_directives(context, revision, directives):
    if getattr(config.cmd_opts, 'autogenerate', False):
        script = directives[0]
        if script.upgrade_ops.is_empty():
            directives[:] = []
            logger.info('No changes in schema detected.')
        connectable = engine_from_config(
            config.get_section(config.config_ini_section),
            prefix='sqlalchemy.', poolclass=pool.NullPool,
        )
        with connectable.connect() as connection:
            context.configure(
                connection=connection, target_metadata=target_metadata,
                process_revision_directives=process_revision_directives,
                **current_app.extensions['migrate'].configure_args
            )
        with context.begin_transaction():
            context.run_migrations()
    if context.is_offline_mode():
        run_migrations_offline()
    else:
        run_migrations_online()

```

EXPLANATION

Smart waste bin sensor sends real-time data to BioEnable smart waste management platform via wireless networks. GSM and WCDMA networks offer 2G and 3G telecommunication modules. Sensor-based waste collection bins are used to determine whether a bin is full or empty, allowing the waste collection schedule to be adjusted and costs to be reduced.

7.2 FEATURE 2

USER MODULE

Users communicate their needs and wants as part of an IoT ecosystem and offer feedback to a networked intelligence to advance each other's capacity to control the actuators of the system at hand. A mobile or other handheld device is a "user device."

PROGRAM

FOR TRACKING SITE ROUTES

```
from flask import Blueprint

site = Blueprint('site',name)
@site.route('/')def
index():

return "<h1>Welcome to Our Waste Management
System</h1>"@site.route('/health')

def health_check():return
"ok"
```

For Validate

```
def validate(roles,data):if
roles is None:

return Truefor role in
roles:
print(role) print(data)
print(role in data)if role not
in data:
```

return False
return True

EXPLANATION

The paths of the region can be defined in the User Module by utilising a Blueprint to Access the Waste Management. By specifying this Module, the user can additionally check the health. Users can establish roles and data for route validation, which verifies whether the provided routes are clean of junk.

7.3 DATABASE SCHEME

Aim: To create a database in Cloudbant DB to store location data.

Steps followed:

- Logged in to IBM Cloud account
- Navigated to `./resources`
- Clicked on the “Create Resource +” button
- Searched for “Cloudbant”
- Chose the “Lite Version” and clicked on “Create”

The screenshot displays the IBM Cloud console interface for creating a new resource. The top navigation bar includes the IBM Cloud logo, a search bar, and links to Catalog, Docs, Support, and Manage. The user's account name, 'Meow Man's Account', is visible on the right. The main content area is titled 'Authentication method' and shows the 'IAM' method selected. Below this, the 'Plan' section offers two options: 'Lite' (selected with a checkbox) and 'Standard'. The 'Lite' plan is described as 'Full functionality for development and evaluation with a set capacity. Only one Lite plan instance per account.' and is 'Free - Only in multi-tenant'. The 'Standard' plan is described as 'Granular control over provisioned throughput capacity allocated. Billing prorated hourly.' and starts at '\$75.00/month'. The 'Capacity' section shows the selected plan's limits: 20 Reads per second, 10 Writes per second, 5 Queries per second, and 1 GB Storage included. A 'Cost calculator' button is located at the bottom left. On the right side, a 'Summary' panel shows the selected '1 Cloudbant Lite' plan with its specifications and a 'Free' price tag. The 'Create' button is highlighted in blue, and an 'Add to estimate' button is also visible.

Plan	Description	Price
Lite	Full functionality for development and evaluation with a set capacity. Only one Lite plan instance per account.	Free - Only in multi-tenant
Standard	Granular control over provisioned throughput capacity allocated. Billing prorated hourly.	Starting at \$75.00/month

Capacity	Reads per second	Writes per second	Queries per second	Storage
20	20	10	5	1 GB

Summary	Price
1 Cloudbant Lite	Free

- The Cloudant database resource was created successfully

IBM Cloud

Search resources and products...

Catalog Docs Support Manage Meow Man's Account

Resource list / Cloudant-jp Active child-safety ibm

Details Actions...

Manage

Service credentials

Plan

Connections

Overview Capacity Docs

Launch Dashboard

Deployment details

CRN crn:v1:bluemix:public:cloudantnosqldb:in-che:a/b77a43c0d7e04aa8abc85cac562dba69:f1fc02fc-6402-486d-b705-aa3ece0c66bf::

Location Chennai

External endpoint https://2a6f6e22-619b-4c94-95bb-5957f5396206-bluemix.cloudant.com

External endpoint (preferred) https://2a6f6e22-619b-4c94-95bb-5957f5396206-bluemix.cloudantnosqldb.appdomain.cloud

Authentication methods IBM Cloud IAM

Activity Tracker event types Management Save

Disk encryption Yes. Automatically generated disk encryption key.

Waiting for cloud.ibm.com... iils

- Clicked on Launch Dashboard

← → ↺ 2a6f6e22-619b-4c94-95bb-5957f5396206-bluemix.cloudant.com/dashboard.html#/_all_dbs

Databases Database name Create Database {} JSON

Your Databases

Name	Size	# of Docs	Partitioned	Actions
------	------	-----------	-------------	---------

Showing 1–0 of 0 databases. Databases per page 20 « 1 »

Log Out

- Clicked on “Create Database”. Entered “meowman” as the database name and the “Non-partitioned” option

←

Databases

Database name

Create Database

{ } JSON

Your Databases

Name	Size	# of Docs	Partitioned
------	------	-----------	-------------

Create Database

Database name

meowman

Partitioning

☒ Non-partitioned - recommended for most workloads

☐ Partitioned

▼ Which should I choose?

If your data can be modelled within the constraints that it imposes, partitioning can improve performance for large databases. See [guide](#) and the extra [service limits](#) for more details.

If in doubt, choose a non-partitioned database.

Cancel

Create

Showing 1–0 of 0 databases

- The database “meowman” was created successfully

←

meowman

:

Document ID

Options

{ } JSON

All Documents


Query

Permissions

Changes

Design Documents

Create Document



No Documents Found

Showing 0 documents. Documents per page: 20

Result:

A database to store the location data was created successfully on Cloudant DB.

CHAPTER 8

TESTING

a. TEST CASES

Finding errors is the goal of testing. The process of testing include looking for any potential flaws or weaknesses in a piece of work. It offers a technique to examine the operation of parts, subassemblies, assemblies, and/or a final good. In order to make sure that the software system complies with its specifications, handles user exceptions, and doesn't fail in an unacceptable way, it must be put through a procedure called software testing. There are many different test types. A particular testing requirement is addressed by each test type.

b. USER ACCEPTANCE TESTING

Before introducing the software application to the invention environment, the user or client performs a type of testing known as user acceptance testing (UAT) to confirm that they are familiar with and comprehend the software system. After functional, integration, and system testing are complete, UAT is carried out as the last stage of testing. User acceptance testing, a crucial stage of any project, necessitates active user involvement. Additionally, it makes sure the system satisfies the functional requirements.

Test Results:

All of the aforementioned test scenarios were roughly successful with a few errors.

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICES

Performance metrics are defined as numbers and information that are indicative of the activities, capacities, and general calibre of a company. Performance measurements can take many different forms, such as sales, profit, ROI, customer satisfaction, customer reviews, personal reviews, general quality, and reputation in the market. When evaluated through various industries, performance metrics might differ greatly.

Metrics of performance are essential to the success of an organisation. Because these metrics aid in directing and assessing an organization's success, it is crucial that firms choose their primary performance metrics and concentrate on these areas. Important success elements are only helpful if they are recognised and monitored. In order for business metrics to provide accurate responses and for the proper questions to be posed, attentive management is also required.

s.no	parameter	values
1	Model summary	These may be established by organisational policy, adherence to a standard that has been made public, or study of the needs based on use, measurement capability, or other factors.
2	Accuracy	Training Some Defects in Accuracy Occurred Validation Exactness Suc-cessfully Passed

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- It saves time and money.
- It decreases traffic flow and consecutively noise due less air pollution.
- It keeps our surroundings clean and green and free from bad odour of wastes.
- It further reduces manpower requirements.
- It reduces infrastructure,operating and maintenance costs by upto 30%.
- The city becomes a "smart city" by optimising management resources and expenses through the application of smart waste management processes

DISADVANTAGES

- System requires more 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.
- In RFID based system, RFID tags are affected by surrounding metal objects.
- It reduces man power requirements which result into increase in unemployments for unskilled people.
- The training has to be provided to the people involved in the smart waste management system.

CHAPTER 11

CONCLUSION

Implementing a smart waste management system employing an ultrasonic sensor, an Arduino, and Wi-Fi modules such sensing modules and user modules is the project effort. When the rubbish level reaches its peak, this method ensures that dustbins will be cleaned as quickly as possible. This lowers the total number of trips made by the waste collection vehicle and, as a result, lowers the overall cost of rubbish collection. In the end, it contributes to maintaining cleanliness in society. As a result, the efficient garbage management system increases the effectiveness of garbage collection. Smartdustbin assists in cleaning up the environment. This project guarantees timely waste collection, which in turn guarantees reduced environmental contamination, no disease spread, and a cleaner environment.

CHAPTER 12

FUTURE SCOPE

Technological advancements are where the future of waste management begins and continues. The waste management sector must develop its work field by being digitised and data-driven, just like every other industry. Future is competitive and clever! Businesses, in particular, must always remain one step ahead of the competition. The data is gathered over time as smart waste management technologies are used. It is possible to identify fill patterns, improve driver routes and schedules, and cut operational expenses by using the data collected by hand sensors. The price of these sensors is continually falling, making the use of smart bins more viable and appealing to business or city leaders. When we refer to the future as clever, we also refer to it as pragmatic. The choice of containers reduces the requirement for waste pickup workers. Collection techniques require the least amount of labour and time possible, and they are eventually lucrative. Using user-friendly, comprehensive, compact platforms and mobile apps for both ends of the waste management processes reduces management time and costs in addition to hardware.

CHAPTER 13

APPENDIX

Source code

```
from flask import Blueprint, jsonify, request, abort, Response, send_file
from app.models import Basket, User, Waste, Vehicle, Employee, commit, Area, SoftwareVersion,
BasketType from
app.validate import validateimport
json
from datetime import datetimefrom io
importBytesIO
api = Blueprint('api',name, url_prefix='/api')

@api.route('/')def
index():
return jsonify({"message": "the api is working"})
@api.route('baskets'
) def get_baskets():
baskets = Basket.query.all()

baskets_list = [basket.format() for basket in baskets]return
jsonify({
    "baskets": baskets_list, "total_baskets":
    len(baskets_list)
})

@api.route('baskets/<int:basket_id>'
) def get_basket(basket_id): basket =
Basket.query.get(basket_id)return jsonify({

@api.route('baskets/<int:basket_id>/wastes ')
def get_wastes_of_basket(basket_id): basket =
```

```

Basket.query.get(basket_id)

wastes = [waste.format() for waste in basket.wastes]total_size
= 0.0
for waste in wastes:

    total_size = total_size + waste['size']
    print(type(waste['size']), waste['size'])
    print(total_size) print(total_size)return
    jsonify({ "basket_id": basket.id,
        "total_size": total_size,
        "wastes": wastes
    })

@api.route('baskets', methods=['POST'])def add_new_basket():data =
request.json

roles = ['longitude', 'latitude', 'area_code'] abort(400) if not
    validate(roles, data) elseNonelongitude = data['longitude']
latitude = data['latitude']
area_code = data['area_code']type_id = data['type']area =
Area.query.get(area_code)if not area:
    abort(422)

basket_type = BasketType.query.get(type_id)if not
    basket_type:
    abort(422)

    basket = Basket(longitude=longitude, latitude=latitude, area=area,basketType=basket_type).save()

return jsonify({ "success":
    True,

@api.route('baskets/<int:basket_id>',
methods=['DELETE'])def delete_baskets(basket_id):

basket = Basket.query.get(basket_id)

basket = basket.delete() if basket else basketreturn
jsonify({

```

```

        'success': bool(basket)

    })

    @api.route('baskets',
    methods=['PATCH'])def
    update_all_baskets():
    data = request.json

    software_version = data['software_version']

    baskets = Basket.query.update({Basket.software_version: software_version}).commit()
    return jsonify({ "baskets_update":
        baskets
    })

    @api.route('baskets/<int:basket_id>', methods=['PATCH'])def
    update_the_basket(basket_id):
    data = request.json basket_level =
    data['level']if basket_level is None:
        abort(400)try:
            basket = Basket.query.get(basket_id)
            basket.wastes_height = basket_level
        basket.save()except:
            abort(422)
    return jsonify({ "success":
        True,
    })

    @api.route('areas')def
    get_areas():
    areas = Area.query.all()
    areas_list = [area.format() for area in areas]return
    jsonify({"total_areas": len(areas_list),"areas": areas_list
    })

    @api.route('areas/<int:area_code>')def
    get_area(area_code):
    area = Area.query.get(area_code)return
    jsonify({
        "area": area.format()
    })

    @api.route('areas/<int:area_code>/baskets') def

```

```

get_basket_belong_to_area(area_code): baskets =
Area.query.get(area_code).baskets
baskets_list = [basket.format() for basket in baskets]return
jsonify({
    "total_baskets": len(baskets_list),
    "baskets": baskets_list
})
@api.route('areas/<int:area_code>/users')def
get_user_belong_to_area(area_code):
users = Area.query.get(area_code).users users_list =
[user.format() for user in users]return jsonify({
    "total_users": len(users_list),"users":
    users_list
})
@api.route('areas',
methods=['POST'])def
insert_new_area():
data = request.json

roles = ['area_code', 'area_name', 'area_size', 'longitude', 'latitude', 'city']if not
validate(roles, data):
    abort(422)

code = data['area_code'] name =
data['area_name'] size =
data['area_size'] longitude =
data['longitude']latitude =
data['latitude']

city = data['city']
area = Area(code=code, name=name, size=size, longitude=longitude,
latitude=latitude, city=city)

area.save(True)return
jsonify({"success":
True, "area":

```



```

area.format()
})
@api.route('vehicles')def
get_vehicles():
vehicles = Vehicle.query.all()

vehicles_list = [vehicle.format() for vehicle in vehicles]return
jsonify({
    "vehicles": vehicles_list
})
@api.route('vehicles/<int:vehicle_plate_no>'
) def get_vehicle(vehicle_plate_no):
vehicle = Vehicle.query.get(vehicle_plate_no)return
jsonify({
    "vehicle": vehicle.format()
})
@api.route('vehicles',
methods=['POST'])defcreate_vehicle():
data = request.json

roles = ['plate_number', 'container_size', 'tank_size', 'employee_ssn']if not
validate(roles, data):
    abort(400)

plate_number = data['plate_number']
container_size = data['container_size']
tank_size = data['tank_size'] employee_ssn =
data['employee_ssn']
driver = Employee.query.get(employee_ssn)

if not driver:

abort(404)# try:
vehicle = Vehicle(plate_number=plate_number, container_size=container_size,tank_size=tank_size,
driver=driver)

```

```

vehicle.save(True)return
jsonify({ "success": True,
"vehicle": [vehicle.format()]
})

# except:
#abort(422)

@api.route('employees') def
get_employees():
employees = Employee.query.all()

employees_list = [employee.format() for employee in employees]return
jsonify({
"total_employees": len(employees_list),"employees":
employees_list
})
@api.route('employees/<int:employee_ssn>'
) def get_employee(employee_ssn):
employee = Employee.query.get(employee_ssn)return
jsonify({
"employee": employee.format()
})
@api.route("employees",
methods=['POST'])def
create_new_employee(): data =
request.json
ssn = data['ssn']
full_name = data['full_name']

user_name = data['user_name'] password =
data['password'] date_of_birth =
data['data_of_birth']
phone = data['phone']
print(ssn)

```

```
if not ssn or not full_name or not user_name or not password or not date_of_birth or not phone:
    abort(400)
```

```
employee = Employee(SSN=ssn, full_name=full_name, user_name=user_name, password=password,
DOB=date_of_birth,
phone=phone).save(True) return jsonify({
    "success": True,

    # "employee": [employee.format()]

})

@api.route("employees",
methods=['PATCH'])def
update_supervisor():
    # TODO update the supervisor for all employee return ''

@api.route('employees/<int:employee_ssn>', methods=['DELETE'])def
delete_employee(employee_ssn):
    employee = Employee.query.get(employee_ssn) if
    employee is None:
        abort(404)

    employee.update() return jsonify({
        "success": True

    })

@api.route('users')def get_all_users(): users =
User.query.all()

users_list = [user.format() for user in users] return
jsonify({"user": users_list})

@api.route('users', methods=['POST'])def create_new_user():
    data = request.json

    user_name = data['user_name'] first_name =
    data['first_name'] last_name = data['last_name'] email =
    data['email']
    password = data['password'] gender = data['gender']
```

```

area = data['area_code'] area = Area.query.get(area)if not area:
abort(404)

roles = ['user_name', 'first_name', 'last_name', 'email', 'password', 'gender']

abort(400)

if not validate(roles, data) else Nonetry:
user = User(user_name=user_name, first_name=first_name,
last_name=last_name, email=email, password=password,
gender=gender, area=area).save(True)return
jsonify({
"success": True, 'user':
user.format()
})

except:
abort(422)

@api.route('wastes') def get_waste():
data = request.args

basket_id = data.get('basket_id', 0, int)

wastes = Waste.query.all() if not basket_id else Waste.query.filter_by(basket_id=basket_id).all()
wastes_list = [waste.format() for waste in wastes]total_size = 0for waste
in wastes_list: total_size += +waste['size']

return jsonify({ "total_wastes_size": total_size,"wastes": wastes_list,
})

@api.route('wastes', methods=['POST'])def insert_new_waste():data =
request.json

basket = Basket.query.get(data['basket_id'])

is_full = basket.set_wastes_height(data['waste_height'])if is_full:abort(422)

waste_size = basket.get_waste_volume(data['waste_height'])

```

```

waste = Waste(size=waste_size, type='bio', DOC=datetime.utcnow(),
basket=basket).save()
return jsonify({

"basket_level": basket.get_basket_level(),"waste": waste.format()
})

@api.route('wastes', methods=['DELETE'])def delete_waste():# waste =
Waste.query.filter_by(type='bio').delete()# db.session.commit()
waste = Waste.query.all()print(waste)

return "

@api.route('test', methods=['POST', "GET"])def test():data =
request.jsonreturn jsonify({
"value": data['value']

})

@api.route("/baskets_types") def get_basket_type():
types_of_baskets = BasketType.query.all()

type_list = [type_of_basket.format() for type_of_basket in types_of_baskets]return
jsonify({
"types": type_list

})

@api.route("/baskets_types", methods=["POST"])def create_basket_type():data =
request.json length = data["length"]height = data["height"]width = data["width"]
micro_controller = data["micro_controller"]roles = ['length', 'height', 'width']

abort(400) if not validate(roles, data) else Nonetry:
    basket_type = BasketType(length=length, height=height, width=width,

    micro_controller=micro_controller).save()return
    jsonify({
        "success": True,
        'Type': basket_type.format()

    })

except:

```

```

    abort(422)
@api.route('/baskets/<int:basket_id>/versions')
def get_basket_software_version(basket_id):
    basket = Basket.query.get(basket_id)
    software_versions =
    SoftwareVersion.query.filter_by(basket_id=basket_id).order_by(SoftwareVersion.date.desc()).all()
    list_software_version = []
    status =
    'update'
    for software_version in software_versions:
        if software_version.version == basket.software_version:
            status = 'rollback'
            list_software_version.append(software_version.format('current'))
        else:
            continue
    list_software_version.append(software_version.format(status))
    return jsonify({
        "software_versions": list_software_version,
        "current_version":
        basket.software_version
    })

@api.route("/software_versions/<string:version>")
def get_file(version):
    software = SoftwareVersion.query.get(version)
    file_name =
    "{}.bin".format(software.version)
    return send_file(BytesIO(software.file), attachment_filename=file_name,
        as_attachment=True)

@api.route("/software_versions", methods=["POST"])
def post_file():
    file = request.files['file']

    update_type = request.form.get("update_type", None)

    basket_id = request.form.get("basket_id", None)
    basket =
    Basket.query.get(basket_id)
    last_version =
    SoftwareVersion.query.filter_by(basket_id=basket_id).order_by(SoftwareVersion.date.desc()).first()
    if last_version:
        major, minor, patch = last_version.version.split(".")
        if update_type == "patch":
            patch =
            int(patch) + 1

```

```

elif update_type == "minor": minor =
    int(minor) +
1 patch = 0
elif update_type == "major": major =
int(major) + 1 patch = 0
    minor = 0 else:
    abort(422) print(last_version.version.split())
    version = "{}.{}.{}".format(major, minor, patch) else:
    version = "0.1.0"
print(version)
print(last_version)
    software_version = SoftwareVersion(version=version, file=file.read(), basket=basket)
software_version.save(True) return
jsonify({
    "success": True,

    "version": software_version.version

}), 201

```

Github Link

<https://github.com/IBM-EPBL/IBM-Project-37667-1660316539>