Project Report Format

# INTRODUCTION

* 1. Project Overview
  2. Purpose

# LITERATURE SURVEY

* 1. Existing problem
  2. References
  3. Problem Statement Definition

# IDEATION & PROPOSED SOLUTION

* 1. Empathy Map Canvas
  2. Ideation & Brainstorming
  3. Proposed Solution
  4. Problem Solution fit

# REQUIREMENT ANALYSIS

* 1. Functional requirement
  2. Non-Functional requirements

# PROJECT DESIGN

* 1. Data Flow Diagrams
  2. Solution & Technical Architecture
  3. User Stories

# PROJECT PLANNING & SCHEDULING

* 1. Sprint Planning & Estimation
  2. Sprint Delivery Schedule
  3. Reports from JIRA

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

* 1. Feature 1
  2. Feature 2
  3. Database Schema (if Applicable)

# TESTING

* 1. Test Cases
  2. User Acceptance Testing

# RESULTS

* 1. Performance Metrics

# ADVANTAGES & DISADVANTAGES

1. **CONCLUSION**

# FUTURE SCOPE

1. **APPENDIX**

Source Code

GitHub & Project Demo Link

1. **INTRODUCTION**
   1. **PROJECT OVERVIEW**

The Internet and its applications have become an integral part of today’s human  
lifestyle. It has become an essential tool in every aspect. Due to the tremendous demand and necessity, researchers went beyond connecting just computers into the web. These researches led to the birth of an Internet of Things (IoT).Things (Physical Devices) that are connected to the Internet and sometimes these devices can be controlled from the internet is commonly called Internet of Things. Nowadays, there are a number of techniques which are purposefully used and are being built up for well management of garbage or solid waste. Sensors and IOT module i.e. Wi-Fi are the latest trends and are one of the best combinations to be used in the project. Hence a combination of both of these technologies is used in the project. Here we are using raspberry pi. A threshold value is set in the IOT. In these we use ultrasonic sensors .When that value is met then it will be sent to the officials through a module about the overload and also to clear the garbage as soon as possible. The same thing is displayed on the LCD, which is connected to the output port of the controller. IOT through data available on web portal about all area dustbin

* 1. **PURPOSE**

Using technology and innovation to optimize current systems will enable cities to become smarter, more efficient and save resources. Due to the growing population, the amount of waste being produced is vast and rapidly increasing. The management of this waste is therefore a significant area for much-needed improvement.

Currently, waste collection systems are in most cases outdated and result in pick-ups that are unnecessary or on the contrary – long-overdue. Unnecessary pickups result in 70% higher annual collection cost. When routes are planned inefficiently, congestion is created and more fuel is required to complete the collection. Overall, this contributes to a 50% higher carbon footprint.

With the use of IoT solutions for waste management, these issues can be solved by creating a more efficient pathway for garbage trucks. IoT sensor technology can be used to indicate when the emptying is actually needed. This customized and dynamic system for waste management can allow businesses, organizations, and citizens to all benefit

1. **LITERATURE SURVEY**

* 1. **EXISTING PROBLEM**

Proper management of food waste, a major component of municipal solid waste (MSW), is needed, especially in developing Asian countries where most MSW is disposed of in landfill sites without any pre treatment. Source separation can contribute to solving problems derived from the disposal of food waste. An organic waste source separation and collection programme has been operated in model areas in Hanoi, Vietnam, since 2007. This study proposed three key parameters (participation rate, proper separation rate and proper discharge rate) for behaviour related to source separation of household organic waste, and monitored the progress of the programme based on the physical composition of household waste sampled from 558 households in model programme areas of Hanoi. The results showed that 13.8% of 558 households separated organic waste, and 33.0% discharged mixed (unseparated) waste improperly. About 41.5% (by weight) of the waste collected as organic waste was contaminated by inorganic waste, and one-third of the waste disposed of as organic waste by separators was inorganic waste. We proposed six hypothetical future household

behaviour scenarios to help local officials identify a final or midterm goal for the programme. We also suggested that the city government take further actions to increase the number of people participating in separating organic waste, improve the accuracy of separation and prevent non-separators from discharging mixed waste improperly.

**2.2 REFRENCES**

* <https://www.mural.co/templates/empathy-map-canvas>
* https://app.mural.co/t/veboosita1204/m/veboosita1204/1663922358124/34ba1c24af97ed 215427f9e2458e1a0d606b1746?sender=ud2f59c937d58a4b323be0236

# J. A. Nathanson, “Solid-waste management | Britannica.com.”

# L. A. Manaf, M. A. A. Samah, and N. I. M. Zukki, “Municipal solid waste management in Malaysia: Practices and challenges,” Waste Manag., vol. 29, no. 11, pp. 2902–2906, Nov. 2009.

# S. Sharmin and S. T. Al-Amin, “A Cloud-based Dynamic Waste Management System for Smart Cities,” in Proceedings of the 7th Annual Symposium on Computing for Development - ACM DEV ’16, 2016, pp. 1–4.

# B. R. Balakrishnan Ramesh Babu, A. K. Anand Kuber Parande, and C.A. Chiya Ahmed Basha, “Electrical and electronic waste: a global environmental problem,” Waste Manag. Res., vol. 25, no. 4, pp. 307–318, Aug. 2007.

# K. Kawai and L. T. M. Huong, “Key parameters for behaviour related to source separation of household organic waste: A case study in Hanoi, Vietnam,” Waste Manag. Res., vol. 35, no. 3, pp. 246–252, Mar. 2017

**2.3 PROBLEM STATEMENT DEFINITION**

1. **IDEATION & PROPOSED SOLUTION**
   1. **EMPATHY MAP CANVAS**

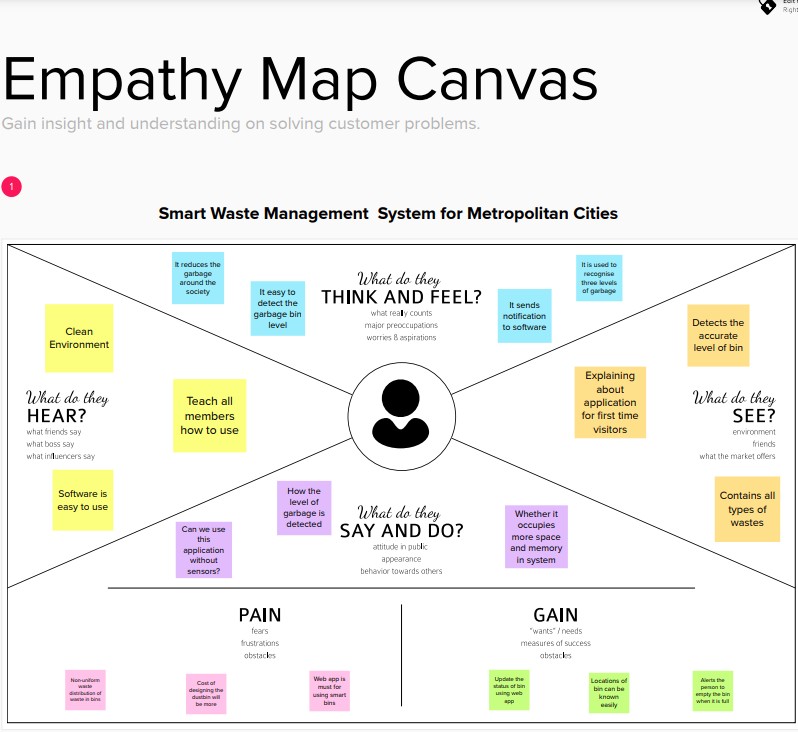
An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s

behaviors and attitudes.

It is a useful tool to helps teams better understand their users.

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.

# Smart Waste Management System for Metropolitan Cities



**Reference:**

https://app.mural.co/t/veboosita1204/m/veboosita1204/1663922358124/34ba1c24af97ed 215427f9e2458e1a0d606b1746?sender=ud2f59c937d58a4b323be0236

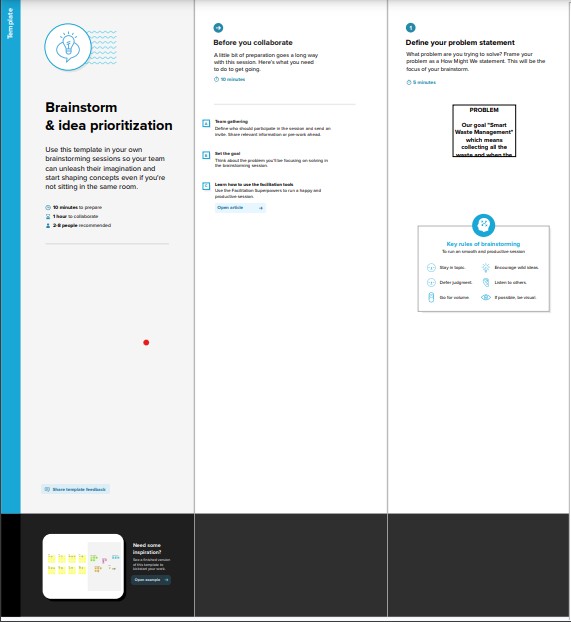
* 1. **IDEATION & BRAINSTORMING**

# Brainstorm & Idea Prioritization Template:

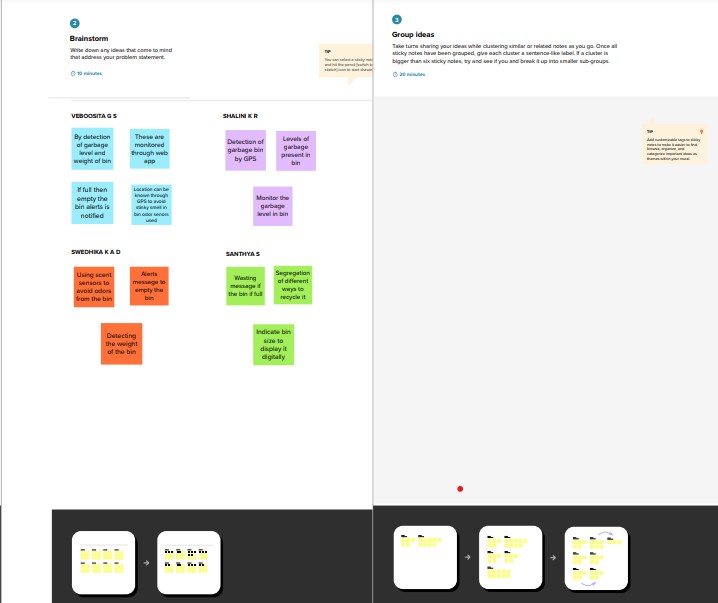
Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

**Reference:** <https://www.mural.co/templates/empathy-map-canvas>

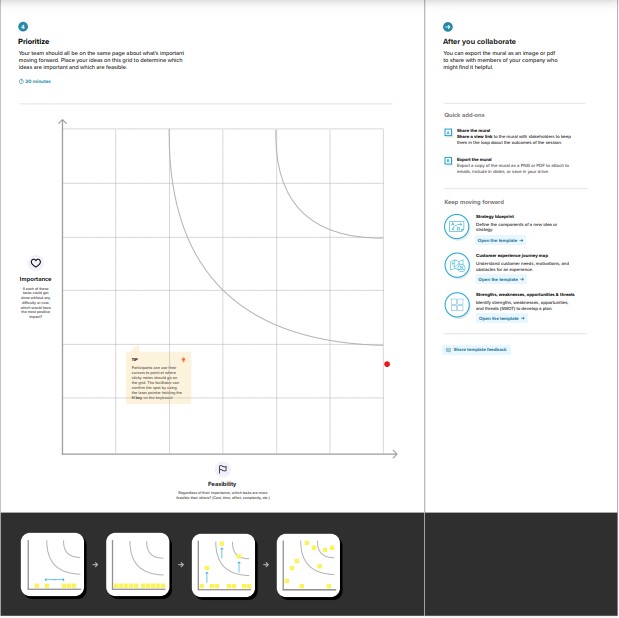
# Step-1: Team Gathering, Collaboration and Select the Problem Statement



**Step-2: Brainstorm, Idea Listing and Grouping**



**Step-3: Idea Prioritization**



* 1. **PROPOSED SOLUTION**

Project team shall fill the following information in proposed solution template.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to be solved) | 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. |
| 2. | Idea / Solution description | The GPS coordinates of the garbage bin will be sent to the IoT platform.  The location of the bins along with bin status can be viewed in the Web Application.  Notifies the admin if the bin value crosses the threshold value |
| 3. | Novelty / Uniqueness | Garbage level of the bins can be monitored through a web App.  We can view the location of every bin in the web application by sending GPS location from the device.  Alerts the authorized person to empty the bin whenever the bins are full. |
| 4. | Social Impact / Customer Satisfaction | At present, we are here to display the live working of the model and give an idea about the actual implications.  For any society to flourish, it is manifestly important that they remain fair and orderly.  Deciding how best to ensure this, in light of the huge growth in both the uptake and complexity of technology that has occurred in the last decade, and which can be expected to continue in the next, this here is one of the products that can be used to contribute to the better management of waste and increase the efficiency of resources. |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| 5. | Business Model (Revenue Model) | C:\Users\USER\Downloads\Urban-smart-waste-management-system.png |
| 6. | Scalability of the Solution | This project-based on IoT gives users the freedom of changing hardware as well as software specifications as per the arising need.  IoT based projects are already designed while keeping future demands in mind and in a rising economy like India where the concept of smart cities is new the demand for our project will keep on increasing.  This project here is a model of the large scale application which spans pan India in different smart cities.  The implementation of this project has been divided into various phases. Starting from the metropolitan cities and moving towards the  concept of smart cities |

* 1. **PROBLEM SOLUTION FIT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Define CS,fit into CC** | **1.CUSTOMER SEGMENT(S**) CS  The Customer for this product Waste holders, such as private individuals, property owners or Companies are our customers. | **6.CUSTOMER CONSTRAINTS** CC  1.Scope 2.Time 3.Risk 4.Quality 5.Benefits 6.Cost | **5.AVAILABLE SOLUTIONS** AS  Uses eco-friendly bags  Digital bins can be used but it can detect the trash level and send notifications to the customers. | **E**  **x**  **p l o r e**  **A** |

|  |  |  |  |
| --- | --- | --- | --- |
| **2.JOBS-TO-BE-DONE/PROBLEMS** J&P  To provide a reliable and efficient service forthe collection, transportation and disposal of waste.  The purpose of the product is to Separate the waste | **9. PROBLEM ROOT CAUSE** RC  Lack of industry expertise. Emission of greenhouse gases.  Poor recycling quality due to lack of education. | 1. **BEHAVIOUR** BE    * If the sensors are not working properly contact the customer care or drop a message. | **Focus on J&P, tap into BE,understand RC** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identify strong TR & EM** | 1. **TRIGGERS** TR    * By using these application users makes clean environment after using it. | **10**.**YOUR SOLUTION** SL  The purpose is of making clean Environment. REDUCE- REUSERECYCLE  Our first job is to explain about the product clearly to the customers and main trick his we have to compare our product to the market available products and, then we need to explain our customers about the advantages and positive thing about the product. | **8.CHANNELS OF BEHAVIOUR** CH  Online:  If it is in online mode, the bin is full it sends the notification to the users  Offline:  If it is offline every day the waste collecting trucks will collect garbage from home. | **Identify strong TR & EM** |
| **4**.**EMOTIONS** EM  **Before:** Before Using this application, the society is suffered for various health issues because of this  waste products  **After:** After sing this application, it is easy and it provides a clean city. |

KDFSFLKSDFKLJDKJJ

1. **REQUIREMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENT**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR**  **No.** | **Functional Requirement**  **(Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR- 1 | Detailed Explanation of bin | You can see bin details in the Dashboard –  capacity, waste type, last measurement, GPS  location and collection schedule. |
| FR- 2 | Monitoring using real time examples | Displays real-time data on fill-levels of bins monitored by smart sensors. With real-time data and predictions, you can eliminate the overflowing bins and stop collecting half-empty  ones |
| FR- 3 | Cost of bins | It helps to identify bins that drive up your  collection costs. The tool calculates a rating for each bin in terms of collection costs. |
| FR- 4 | Adjusting level of garbage | Identify areas with either dense or sparse bin distribution. Make sure all trash types are  represented within a stand. |
| FR- 5 | Eliminate unsufficient garbage | Eliminate the collection of half-empty bins. By using real-time data on fill-levels and pick  recognition, we can show you how full the bins you collect are. |

|  |  |  |
| --- | --- | --- |
| FR- 6 | Planning for waste collection | The tool semi-automates waste collection route planning. Based on current bin fill-levels and predictions of reaching full capacity, you are  ready to respond and schedule waste collection. |

**4.2 NON-FUNCTIONAL REQUIREMENT**

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

|  |  |  |
| --- | --- | --- |
| **FR**  **No.** | **Non-Functional Requirement** | **Description** |
| NFR- 1 | **Usability** | In the design process with user experience as the core, the analysis of users’ product usability can indeed help designers better understand users’ potential needs in waste management, behavior  and experience. |
| NFR- 2 | **Security** | Use a reusable garbage Purchase wisely and recycle  Avoid single use food and drink containers |
| NFR- 3 | **Reliability** | Smart waste management is also about creating better working conditions for waste collectors and  drivers. |
| NFR- 4 | **Performance** | Using a variety of IoT networks ( (NB-IoT,GPRS), the sensors send the data to Sensoneo’s Smart Waste Management Software System, a powerful cloud-based platform, for data driven daily operations, available also as a waste management  app. |
| NFR- 5 | **Availability** | Another purpose of this project is to make the proposed waste management system as cheap as possible. By developing & deploying resilient hardware and beautiful software we empower cities, businesses, and countries to manage waste  smarter. |

|  |  |  |
| --- | --- | --- |
| NFR- 6 | **Scalability** | By using smart waste bins, we able to monitor the garbage frequently and number of bins will be  reduced. |

1. **PROJECT DESIGN**

**5.1 DATA FLOW DIAGRAMS**

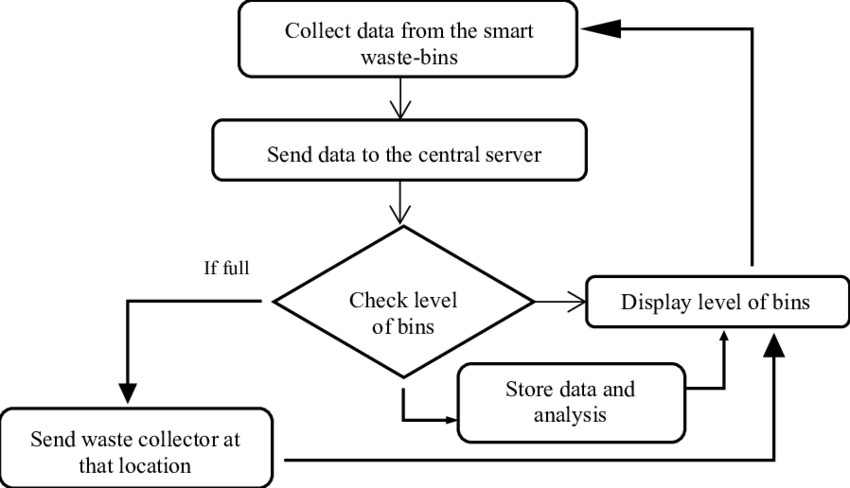
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enter and leaves the system, what changes the information, and where data is stored.

A smart waste management uses analytics to translate the data.

* The test conducted is the situation where the garbage bin is empty or its garbage level is very low.
* Then, the bin is filled.
* The notification is sent by the system.
* The garbage needs to be collected immediately.
* Location prone to overflow.
* The number of bins needed to avoid overflowing waste.
* The number of collection services that could be saved.
* The amount of fuel that could be saved.
* The driving distance that could be saved.
* No missed pickups of trashcans.
* New smart dustbins can be install by just connecting the IoT device to the cloud.

# Example:

**Data Flow Diagram**



**5.2 SOLUTION & TECHNICAL ARCHITECTURE**

The Deliverable shall include the architectural diagram as below and the information as per thetable1 & table 2

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | IBM Watson IOT cloud platform | MQTT Protocol |
| 2. | Application Logic-1 | The waste data are collected using  sensors | Python |
| 3. | Application Logic-2 | The collected data  are monitored using IOT application | IBM Watson STT service |
| 4. | Application Logic-3 | Based on data’s the alerting message will send to the workers for disposing the  wastes. | IBM Watson Assistant |
| 5. | Database | MySQL is a relational database that is based on a tabular design. NoSQL is non-relational and has a document-  based design | MySQL, NoSQL |
| 6. | Cloud Database | In this module will receive real time status updates from all the bins and continuously display it on web application and also push the notifications on client  sides. | IBM DB2, IBM Cloud |

|  |  |  |  |
| --- | --- | --- | --- |
| 7. | File Storage | Data storage makes it easy to back up files for safekeeping and quick recovery in the event of an unexpected  computing crash or cyberattack. | IBM Block Storage or Other Storage Service |
| 8. | External API-1 | External APIs expose a project's internal resources to outside  users or applications | IBM Weather API, etc. |
| 9. | External API-2 | External API allow you to access third party resources that are available through  RESTful web services | Aadhar API, etc. |
| 10. | Machine Learning Model | The proper algorithm makes planning good. It will guide the goodness character and which path should be taken and which garbage bin should be collected  First | Python IDLE or Anaconda navigator or Jupitar |
| 11. | Infrastructure (Server  / Cloud) | Application Deployment on Local System / Cloud Cloud Server Configuration: Cloud deployment is the process of deploying an application through one or more hosting models— software as a service (SaaS), platform as a service (PaaS) and or infrastructure as a service (IaaS) that leverage the cloud Local Server Configuration: A local server gives you exclusive access to  data and objects in a set of Windows | Cloud server- MySQL Local server-HTTP |

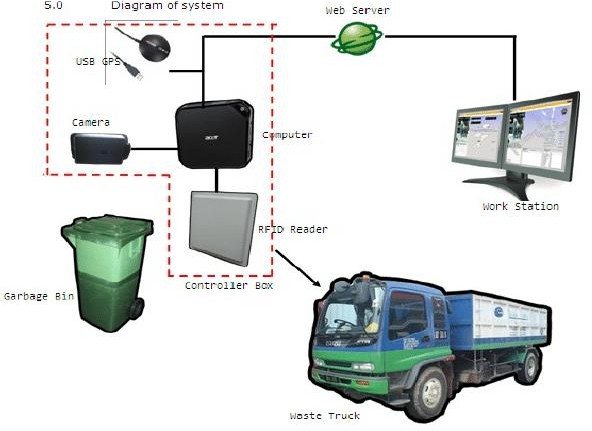
|  |  |  |  |
| --- | --- | --- | --- |
|  |  | folders called data directories |  |

# Table-2: Application Characteristics:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 12. | Open-Source Frameworks | Transport, treatment, and disposal of waste together with monitoring and  regulation. | Technology of Opensource framework is python. |
| 13. | Security Implementations | Fundamental component of data security that dictates who's allowed to access and use company information and resources.  Firewalls use a rule- based access control model with rules expressed in an  access control list. | Firewall |
| 14. | Scalable Architecture | Using smart waste bins, reduce the number of bins inside town and cities because that we can able to monitor the  garbage 24/7. It will be more cost efficient | IOT |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | and scalable when we move to smarter. |  |
| 15. | Availability | By developing & deploying resilient hardware and beautiful software we empower cities, businesses, and  countries to manage waste smarter | IOT, RFID |
| 16. | Performance | The Smart Sensors use ultrasound technology to measure the fill levels (along with other data) in bins several times a day. Using a variety of IoT networks ( (NB-IoT, GPRS), the sensors send the data to Sensor’s Smart Waste Management Software System, a powerful cloud-based platform, for data- driven daily operations, available also as a waste management app | IOT, GPRS |

**TECHNOLOGY ARCHITECTURE:**



**5.3 USER STORIES**

# User Stories

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| Admin | Login | USN-1 | Admin gives a user id and password for each and every workers and  helps to manage | I can access my account / dashboard | Medium | Sprint-2 |
| Assistant Admin | Login | USN-2 | They help us to monitor the garbage level once it is filled alert message  will be thrown with location | I can manage and monitor the garbage level | High | Sprint-1 |
| Driver | Login | USN-3 | They will follow the location | I can drive to reach the | Medium | Sprint-2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
|  |  |  | where the garbage is filled and collect them  in the truck | garbage where it is filled using location and  collect them |  |  |
| Garbage Collector | Login | USN-4 | It will collect the trash and load it into the garbage  truck and send to landfill | I can collect the trash and load them in truck | Medium | Sprint-2 |
| Government Municipality | Login | USN-5 | It will check the process without  involving any issues | I can manage the process smoothly | High | Sprint-1 |

1. **PROJECT PLANNING & SCHEDULING**

**6.1 SPRINT PLANNING & ESTIMATION**

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

Use the below template to create product backlog and sprint schedule

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Login | USN-1 | As Admin, I need to give access for every workers in the municipality. | 20 | High | K.A.D.Swedhika |
| Sprint-1 | Login | USN-2 | As Co-Admin, I’ll control the garbage level by monitoring through website. Once the bin is  filled it will be notified to a trash truck connecting with their location. | 10 | High | K.R.Shalini |
| Sprint-2 | Dashboard | USN-3 | As a Truck Driver, I’ll follow Co-Admin’s  instructions to reach the bins with help of location and save time. | 20 | Low | G.S.Veboosita |
| Sprint-3 | Dashboard | USN-4 | As a Garbage Collector, I’ll collect all the garbage and load them in a trash truck and deliver to the landfills. | 20 | Medium | G.S.Veboosita |
| Sprint-4 | Dashboard |  | As Municipality Officer, I’ll confirm whether everything is processed without any issues. | 20 | High | S.Santhya |

**6.2 SPRINT DELIVERY SCHEDULE**

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

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

**Velocity:**

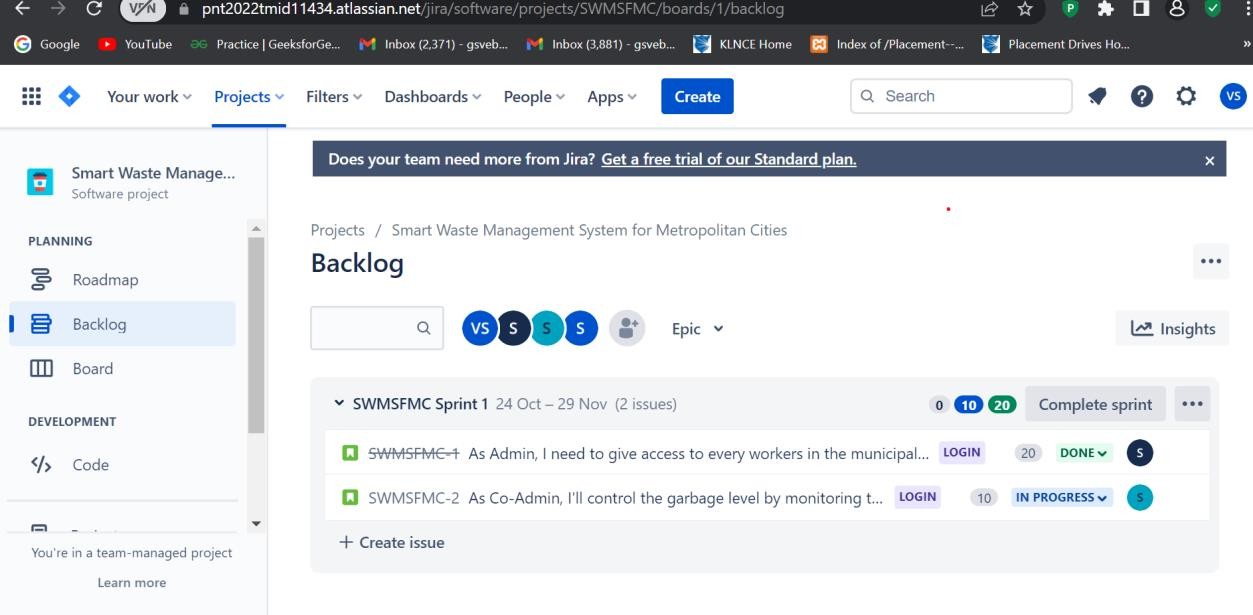
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

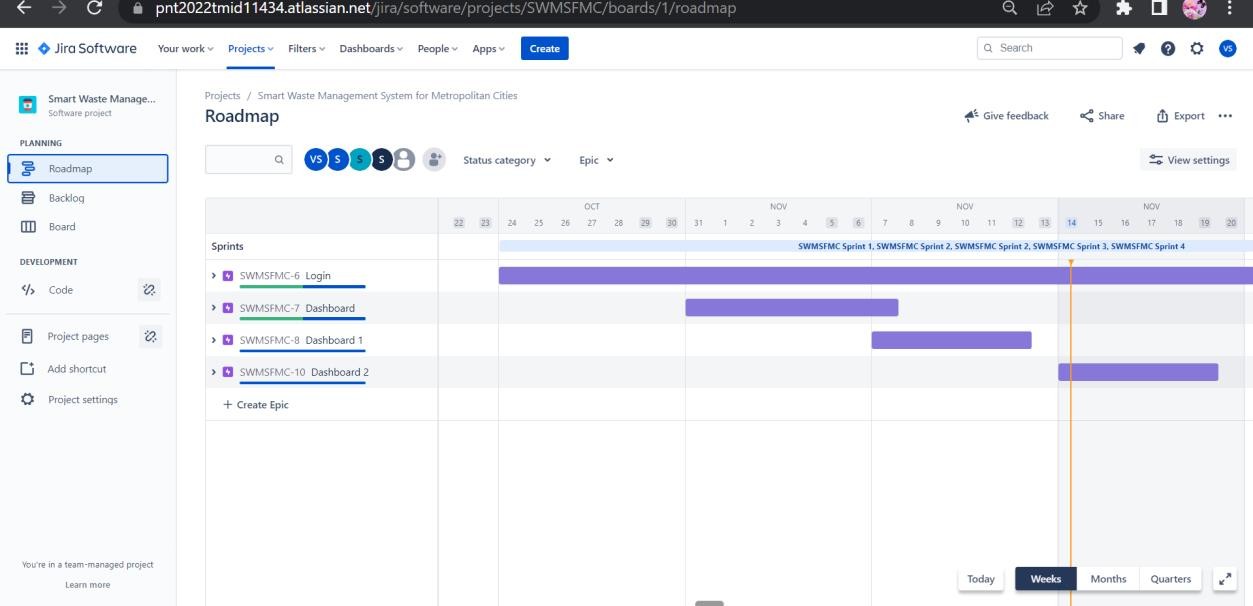


# Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile [software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as [Scrum](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/). However, burn down charts can be applied to any project containing measurable progress over time.

**6.3 REPORTS FROM JIRA**





1. **CODING & SOLUTIONING(EXPLAIN THE FEATURES ADDED IN THE PROJECT ALONG WITH CODE)**

**7.1 FEATURE 1**

**SPRINT 1,2,3,4**

**1.** import wiotp.sdk.device

import time

from geopy.geocoders import Nominatim

import random

geoloc=Nominatim(user\_agent="geoapiExercises")

lat="9.914470"

long="78.143418"

location=geoloc.reverse(lat+","+long)

addr=location.raw['address']

latitude = {'Lat': lat}

longitude = {'Long': long}

print(addr.get('suburb',''))

myConfig = {

"identity": {

"orgId": "zal46w",

"typeId": "Dumpster",

"deviceId":"12345"},

"auth": {

"token": "12345678"

}

}

def myCommandCallback(cmd):

print("Message received from IBM IoT Platform: %s" % cmd.data['command'])

m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)

client.connect()

while True:

level=random.randint(0,100)

weight= random.randint(0, 1000)

if level>50 and weight>500:

client.publishEvent(eventId="status", msgFormat="json", data={'V\_alert':"pick dumpster in the"+addr.get('suburb','')+"street"+addr.get('city','')}, qos=0, onPublish=None)

time.sleep(2)

client.publishEvent(eventId="status", msgFormat="json", data={'V\_alert': ""}, qos=0,

onPublish=None)

level\_d={'Level':level}

weight\_d = {'Weight': weight}

client.publishEvent(eventId="status", msgFormat="json", data=level\_d, qos=0, onPublish=None)

client.publishEvent(eventId="status", msgFormat="json", data=weight\_d, qos=0, onPublish=None)

client.publishEvent(eventId="status", msgFormat="json", data=latitude, qos=0, onPublish=None)

client.publishEvent(eventId="status", msgFormat="json", data=longitude, qos=0, onPublish=None)

print("Published data Successfully: %s", level\_d)

print("Published data Successfully: %s", weight\_d)

print("Published data Successfully: %s", latitude)

print("Published data Successfully: %s", longitude)

client.commandCallback = myCommandCallback

time.sleep(2)

client.disconnect()

**2**. import wiotp.sdk.device

import time

from geopy.geocoders import Nominatim

import random

myConfig = {

"identity": {

"orgId": "zal46w",

"typeId": "Dumpster",

"deviceId":"12345"},

"auth": {

"token": "12345678"

}

}

id= [0]

geoloc=Nominatim(user\_agent="geoapiExercises")

def init():

lat, long = "9.914470", "78.143418"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb1 = addr.get('suburb', '')

city1 = addr.get('city', '')

lat, long = "9.9933491", "78.127579"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb2 = "Tepakulam"

city2 = addr.get('city', '')

lat, long = "9.917916", "78.123496"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb3 = "KK Nagar"

city3 = addr.get('city', '')

mydata = {

'd': {'d\_dump1': 1, 'Suburb1': suburb1, 'City1': city1, 'd\_dump2': 2, 'Suburb2': suburb2, 'City2': city2, 'd\_dump3': 3,

'Suburb3': suburb3, 'City3': city3}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

def dumpster\_1():

lat, long = "9.914470", "78.143418"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb = addr.get('suburb', '')

city = addr.get('city', '')

level = 60

weight = 600

mydata = {'d': {'Level1': level, 'Weight1': weight, 'Lat1': lat, 'Long1': long,'d\_dump1':1,'Suburb1': suburb, 'City1': city}}

if (level > 50 and weight > 500):

mydata = {

'd': {'dump1': dumpid, 'Level1': level, 'Weight1': weight, 'Lat1': lat, 'Long1': long, 'd\_dump1':1,'Suburb1': suburb, 'City1': city}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("pick")

time.sleep(2)

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("dump ", dumpid)

print("Published data Successfully: %s", mydata)

def dumpster\_2():

lat, long = "9.9933491", "78.127579"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb = "Tepakulam"

city = addr.get('city', '')

level = 70

weight = 700

mydata = {'d': {'Level2': level, 'Weight2': weight, 'Lat2': lat, 'Long2': long,'d\_dump2':1,'Suburb2': suburb, 'City2': city}}

if (level > 50 and weight > 500):

mydata = {

'd': {'dump2': dumpid, 'Level2': level, 'Weight2': weight, 'Lat2': lat, 'Long2': long,'d\_dump2':2,'Suburb2': suburb, 'City2': city}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("pick")

time.sleep(2)

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("dump ", dumpid)

print("Published data Successfully: %s", mydata)

def dumpster\_3():

lat, long = "9.917916", "78.123496"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb = "KK Nagar"

city = addr.get('city', '')

level = 88

weight = 700

mydata = {'d': {'Level3': level, 'Weight3': weight, 'Lat3': lat, 'Long3': long,'d\_dump3':3,'Suburb3': suburb, 'City3': city}}

if (level > 50 and weight > 500):

mydata = {

'd': {'dump3': dumpid, 'Level3': level, 'Weight3': weight, 'Lat3': lat, 'Long3': long,'d\_dump3':3,'Suburb3': suburb, 'City3': city}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("pick")

time.sleep(2)

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("dump ", dumpid)

print("Published data Successfully: %s", mydata)

def myCommandCallback(cmd):

print("Message received from IBM IoT Platform: %s" % cmd.data['command'])

m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)

client.connect()

while True:

dumpid = 2

init()

if dumpid == 1:

dumpster\_1()

elif dumpid == 2:

dumpster\_2()

elif dumpid==3:

dumpster\_3()

mydata = {'d': {'d\_dump1': 4}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

client.commandCallback = myCommandCallback

time.sleep(2)

client.disconnect()

3.import wiotp.sdk.device

import time

from geopy.geocoders import Nominatim

import random

myConfig = {

"identity": {

"orgId": "zal46w",

"typeId": "Dumpster",

"deviceId":"12345"},

"auth": {

"token": "12345678"

}

}

id= [0]

geoloc=Nominatim(user\_agent="geoapiExercises")

def init():

lat, long = "9.914470", "78.143418"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb1 = addr.get('suburb', '')

city1 = addr.get('city', '')

lat, long = "9.9933491", "78.127579"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb2 = "Tepakulam"

city2 = addr.get('city', '')

lat, long = "9.917916", "78.123496"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb3 = "KK Nagar"

city3 = addr.get('city', '')

mydata = {

'd': {'d\_dump1': 1, 'Suburb1': suburb1, 'City1': city1, 'd\_dump2': 2, 'Suburb2': suburb2, 'City2': city2, 'd\_dump3': 3,

'Suburb3': suburb3, 'City3': city3}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

def dumpster\_1():

lat, long = "9.914470", "78.143418"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb = addr.get('suburb', '')

city = addr.get('city', '')

level = 60

weight = 600

mydata = {'d': {'Level1': level, 'Weight1': weight, 'Lat1': lat, 'Long1': long,'d\_dump1':1,'Suburb1': suburb, 'City1': city}}

if (level > 50 and weight > 500):

mydata = {

'd': {'dump1': dumpid, 'Level1': level, 'Weight1': weight, 'Lat1': lat, 'Long1': long, 'd\_dump1':1,'Suburb1': suburb, 'City1': city}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("pick")

time.sleep(2)

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("dump ", dumpid)

print("Published data Successfully: %s", mydata)

def dumpster\_2():

lat, long = "9.9933491", "78.127579"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb = "Tepakulam"

city = addr.get('city', '')

level = 70

weight = 700

mydata = {'d': {'Level2': level, 'Weight2': weight, 'Lat2': lat, 'Long2': long,'d\_dump2':1,'Suburb2': suburb, 'City2': city}}

if (level > 50 and weight > 500):

mydata = {

'd': {'dump2': dumpid, 'Level2': level, 'Weight2': weight, 'Lat2': lat, 'Long2': long,'d\_dump2':2,'Suburb2': suburb, 'City2': city}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("pick")

time.sleep(2)

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("dump ", dumpid)

print("Published data Successfully: %s", mydata)

def dumpster\_3():

lat, long = "9.917916", "78.123496"

location = geoloc.reverse(lat + "," + long)

addr = location.raw['address']

suburb = "KK Nagar"

city = addr.get('city', '')

level = 88

weight = 700

mydata = {'d': {'Level3': level, 'Weight3': weight, 'Lat3': lat, 'Long3': long,'d\_dump3':3,'Suburb3': suburb, 'City3': city}}

if (level > 50 and weight > 500):

mydata = {

'd': {'dump3': dumpid, 'Level3': level, 'Weight3': weight, 'Lat3': lat, 'Long3': long,'d\_dump3':3,'Suburb3': suburb, 'City3': city}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("pick")

time.sleep(2)

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

print("dump ", dumpid)

print("Published data Successfully: %s", mydata)

def myCommandCallback(cmd):

print("Message received from IBM IoT Platform: %s" % cmd.data['command'])

m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)

client.connect()

while True:

mydata = {'p': {'suburb1':"Anna Nagar , Madurai",'suburb2':"Tepakulam, Madurai",'suburb3':"KK Nagar, Madurai"}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

dumpid = random.randint(1,3)

init()

if dumpid == 1:

dumpster\_1()

elif dumpid == 2:

dumpster\_2()

elif dumpid==3:

dumpster\_3()

mydata = {'d': {'d\_dump1': 4}}

client.publishEvent(eventId="status", msgFormat="json", data=mydata, qos=0, onPublish=None)

client.commandCallback = myCommandCallback

time.sleep(2)

client.disconnect()

1. **TESTING**

**8.1 TESTCASES**

| Test  case  ID | Feature  Type- Bin Level | Component | Test Case Scenario | | Pre  Requisite | | Availability | | Test  Condition | Expected Result | Actual Result |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test  case  1 | Empty | Ultrasonic  Sensor | When Bin is empty | | Ultrasoncic sensor PIR Motion | | Sensor  Garbage  Bins | | Bin Level == 0 | Displays  Bin level and  space  left | Working as  expected |
|  |  |  |  |  | |  | |  |  |  |  |
| Test  case  2 |  | Accessible Ultrasonic Sensor | When bin level is  below50  % | | Ultrasonic sensor ,  PIR Motion Sensor , ,  Garbage  Bins | | ,bin is  accessible to user | | Bin Level < 50 | Displays  Bin level and  space  left | Working  as expected |
| Test  case  3 | Accessible | Ultrasonic  Sensor | When bin level is  above 50 | | Ultrasonic sensor ,  PIR Motion Sensor , ,  Garbage  Bins | | Bin is  accessible to users  and the  admin gets warning  about the  bin level | | Bin level >50 | Displays  bin level space  left | Working as  expected |

Accesse By

User

User

User

| Test  case4 | Accessible | Ultra sonic  sensor | When bin level is  below75% | Ultrasonic sensor ,  PIR Motion Sensor , ,  Garbage  Bins | Bin is  accessible to users  and the  admin gets warning  about the  bin level | Bin  level<75 | Displays  bin level space  left | Working as  expected |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test  case  5 | Accessible Limit | exceedUltrasonic sensor | When bin level is  above | Ultrasoncic sensor ,  PIR Motion | Bin is not accessible To the | Bin  level>75 | Display  bin level And | Working as  expected |

**8.2 USER ACCEPTANCE TESTING**

# Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Smart Waste Management System project at the time of the release to User Acceptance Testing (UAT).

# Defect Analysis

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Resolution** | **Severity 1** | **Severity 2** | **Severity 3** | **Severity 4** | **Subtotal** |
| By Design | 10 | 4 | 3 | 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 | 78 |

# Test Case Analysis

This report shows the number of test cases that have passed ,failed ,and untested.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Section** | | **Total Cases** | | **Not Tested** | | **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 | |

1. **RESULTS**

**9.1 PERFORMANCE METRICS**