PROJECT REPORT

1.INTRODUCTION

Today big cities around the world are facing a common problem, managing the city waste effectively without making city unclean. Today’s waste management systems involve a large number of employees being appointed to attend a certain number of dumpsters this is done every day periodically. This leads to a very inefficient and unclean system in which some dumpsters will be overflowing some dumpsters might not be even half full. This is caused by variation in population density in the city or some other random factor this makes it impossible to determine which part needs immediate attention. Here a Smart Waste Management System For Metropolitan Cities is introduced in which each dumpster is embedded in a monitoring system that will notify the corresponding personal if the dumpster is full. In this system, it is send to the alert message to the Wastage management admin and also send message to near by dumpster cleaning vehicle when is possible to clean the bin accept the request to collect wastage. This system provides an effective solution to the waste management problem

1.1.Project Overview

- Garbage level detection in bins.
- Getting the weight of the garbage in the bin.
- Alerts the authorized person to empty the bin whenever the bins are full.
- 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.

1.2.Purpose

- The GPS coordinates of the garbage bin will be sent to the IBM 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.

2.LITERATURE SURVEY

SI:NO	TITLE OF THE PAPER	AUTHOR	METHODOLOGY	MERITS	YEAR OF PUBLICATIO N
1	Smart waste bin Management	Parthasarathi Manickaraja	Uses the Ultrasonic sensor to level the dustbin and also Uses the GSM module	Provides an alert message once The level has reached to the authority	2022

2	Smart waste management using IOT	Tejashree Kadus	Technology used is a load cell and a Wi-Fi module	Segregate the waste in the dustbin and provides and Alert message	2020
3	Smart waste management systems using machine learning	David Rutgvist	Uses automated machine learning for a real life smart waste management	It focus on problems of detection of emptying of a recycling Container using sensor measurements	2019
4	Real time solid waste bin monitoring system framework using Wireless sensor network	Thiyaga priya dharshini	Smart bin based on a microcontroller Based platform Arduino which is interfaced with GSM module	Waste management efficiency and it avoids lumping Of wastes	2019
5	Smart waste collection system	Muhamad JavedRamzan	Technology based on sensor based Collection and uses route algorithm	It identifies the status of waste Bin levels along with the location to replace the bin	2018
6	Waste management And tracking	B Keerthana	Technology based on ZigBee.	Less expensive Lock based System with acknowledgment alert system	2017
7	Smart Recycle Bin	Mohd Helmy Abd Wahab, Aeslina Abdul Kadir	A Conceptual Approach of Smart Waste Management with Integrated Web Based System	At the time of trash disposal, the material to be recycled could be identified using RFID technology	2015

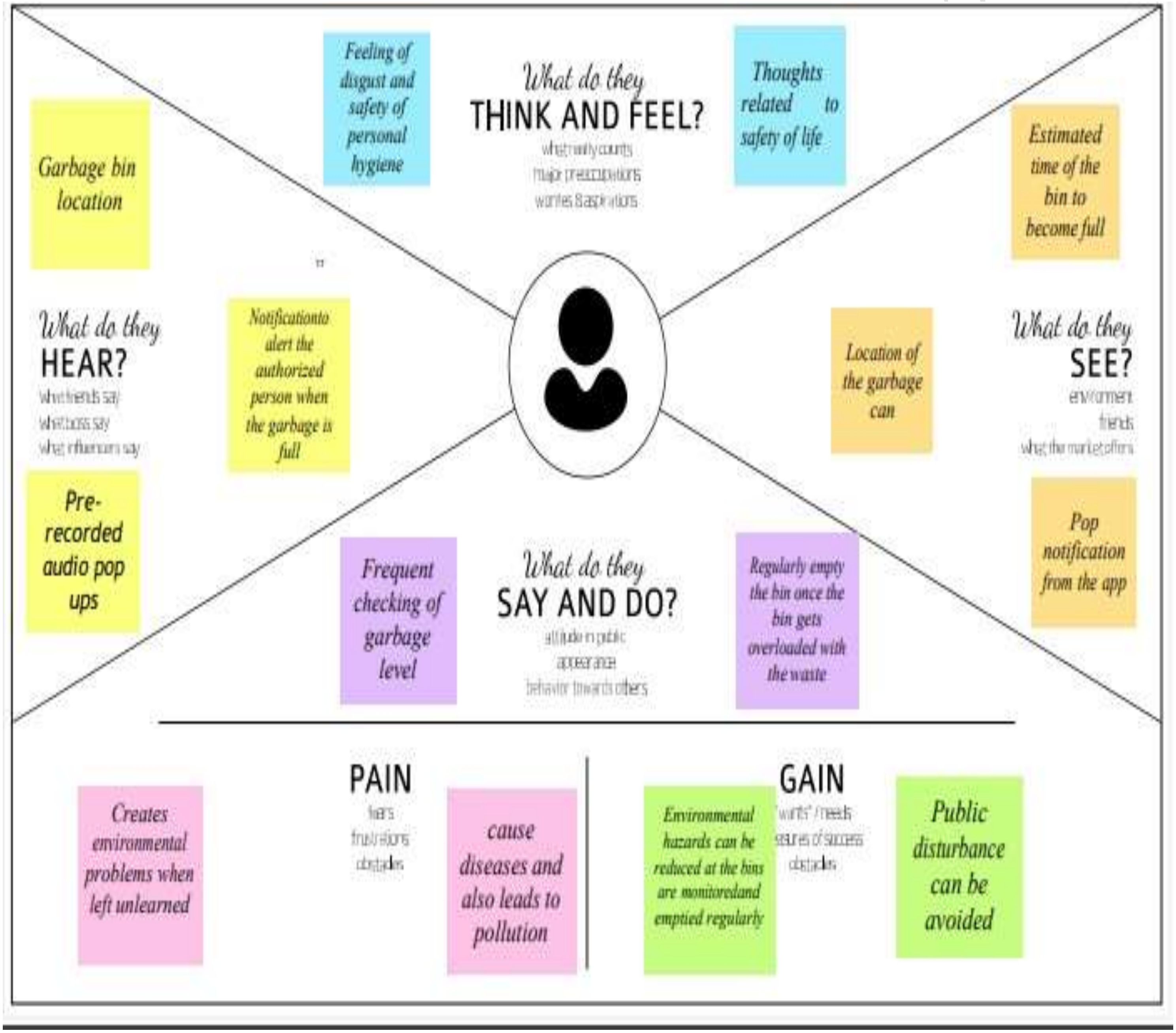
2.1.Existing problem

- Manual systems in which employees clear the dumpsters periodically
- No systematic approach towards clearing the dumpsters

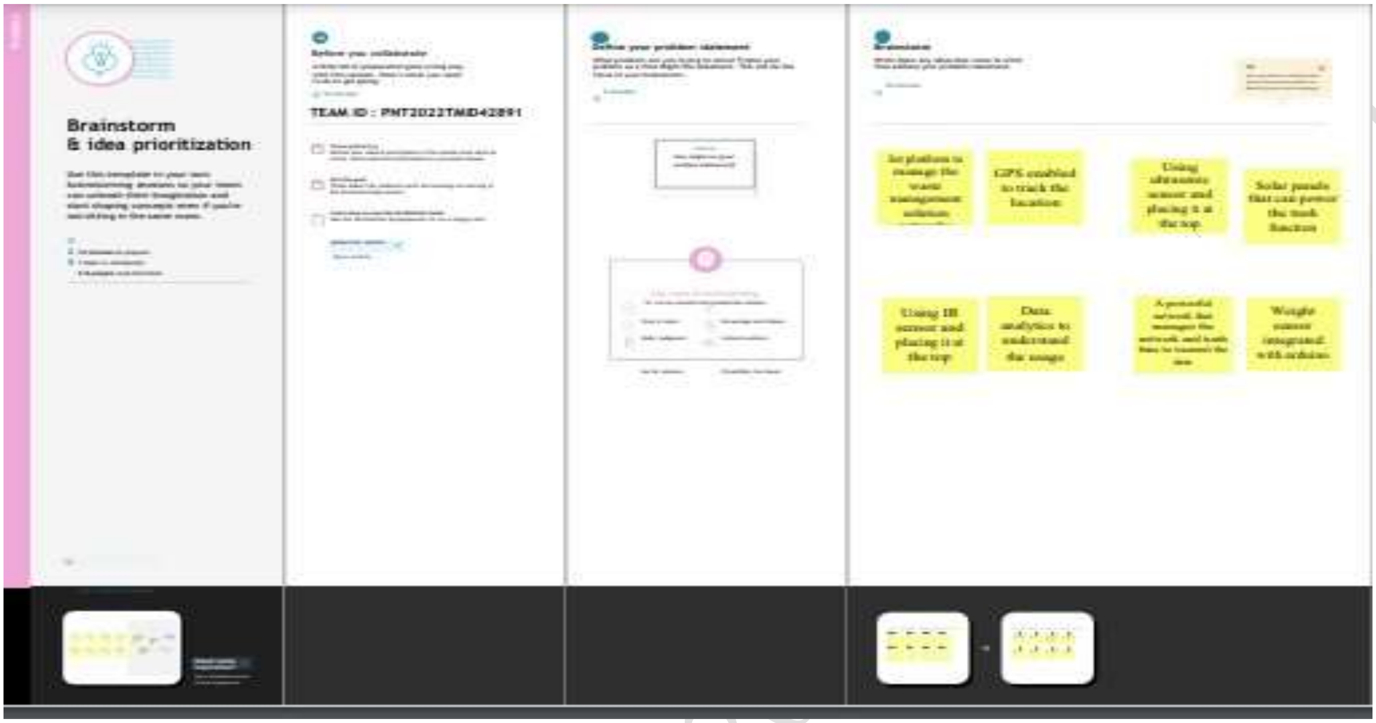
- Unclear about the status of a particular location
- Employees are unaware of the need for a particular location
- Very less effective in cleaning city.

3.IDEATION & PROPOSED SOLUTION

3.1.Empathy Map Canvas


What do they
THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Feeling of disgust and safety of personal hygiene
Thoughts related to safety of life
What do they
HEAR?
what friends say
what boss say
what influencers say
Garbage bin location
Notificationto alert the authorized person when the garbage is full
Pre-recorded audio pop ups
What do they
SEE?
environment
friends
what the market offers
Estimated time of the bin to become full
Location of the garbage can
Pop notification from the app
What do they
SAY AND DO?
attitude in public
appearance
behavior towards others
Frequent checking of garbage level
Regularly empty the bin once the bin gets overloaded with the waste
PAIN
fears
frustrations
obstacles
Creates environmental problems when left unlearned
cause diseases and also leads to pollution
GAIN
"wants" / needs
measures of success
obstacles
Environmental hazards can be reduced at the bins are monitoredand emptied regularly
Public disturbance can be avoided

3.2.Ideation & Brainstorming



Brainstorm & idea prioritization

Before you collaborate

TEAM ID : PNT2022TMID42891

Define your problem statement

Brainstorm

Iot platform to manage the waste management solution

GPS enabled to track the location

Using ultrasonic sensor and placing it at the top

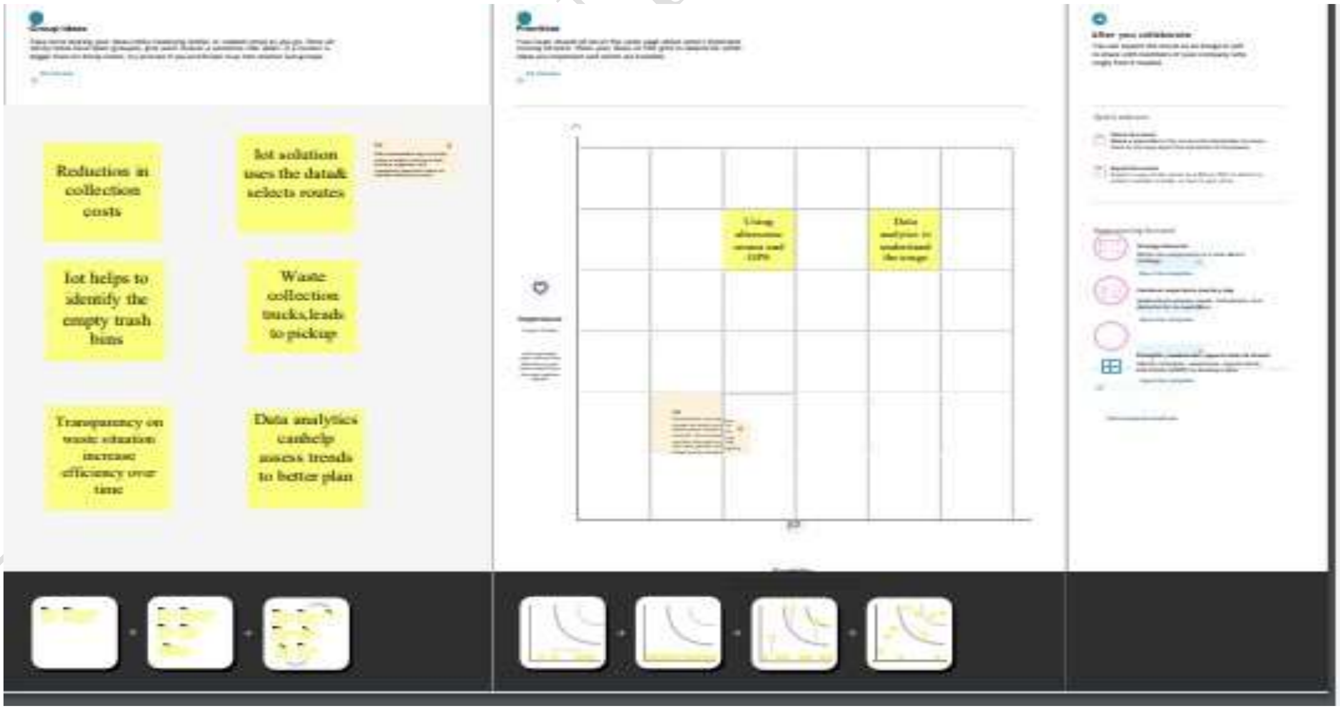
Solar panels that can power the trash function

Using IR sensor and placing it at the top

Data analytics to understand the usage

A powerful network that manages the network and leads bins to transmit the data

Weight sensor integrated with arduino



Group ideas

Reduction in collection costs

Iot solution uses the data& selects routes

Iot helps to identify the empty trash bins

Waste collection trucks,leads to pickup

Transparency on waste situation increase efficiency over time

Data analytics canhelp assess trends to better plan

Prioritize

Using ultrasonic sensor and GPS

Data analytics to understand the usage

After you collaborate

3.3. Proposed Solution

- Using sensors, weighing machine; real time monitoring the level of waste in bins.
- The information gets shared with appropriate authorities and fellow citizens through web application

3.4. Problem Solution fit

- The need-driven waste collection eliminates unnecessary traffic blockage.
- Generate important statistical data for monitoring for waste collection.
- Recycling is promoted between residents, results in clean & sustainable environment.

4.REQUIREMENT ANALYSIS

4.1. Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
NFR -1	Detailed bin inventory.	All monitored bins and stands can be seen on themap, and you can visit them at any time via the Street View feature from Google. Bins or stands are visible on the map as green, orange or red circles.You can see bin details in the Dashboard – capacity,waste type, last measurement, GPS location and collection schedule or pick recognition.
NFR -2	Real time bin monitoring.	The Dashboard displays real-time data on fill-levelsof bins monitored by smart sensors. In addition to the % of fill-level, based on the historical data, the tool predicts when the bin will become full, one of the functionalities that are not included even in the best waste management software. Sensors recognize picks as well; so you can check when the bin was last collected. With real-time data and predictions, you can eliminate the overflowing bins and stop collecting half-emptyones.
NFR -3	Expensive bins.	Smart waste management is also about creating better working conditions for waste collectors anddrivers. Instead of driving the same collection routes and servicing empty bins, waste collectors will spend their time more efficiently, taking careof bins that need servicing.

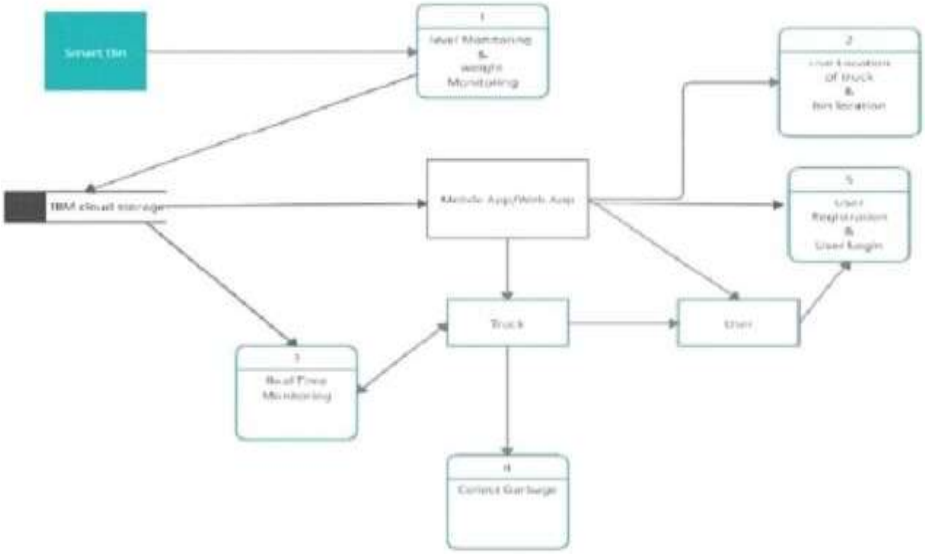
4.2.Non-Functional requirements

FR No	Non- Functional Requirement	Description
FR-N1	Usability	IoT device verifies that usability is a specialand important perspective to analyze user requirements, which can further improve the problems design quality. In the design process with userexperience as the core, the analysis of users' product usability can indeed help designers better understand users' potential needs in waste management, behavior and experience.
FR-N2	Security	Use a reusable bottles Use reusable grocery bags Purchase wisely and recycle Avoid single use food and drink containers.
FR-N3	Reliability	Smart waste management is also about creatingbetter working conditions for waste collectors and drivers. Instead of driving the same collection routes and servicing empty bins, waste collectors will spend their time more efficiently, taking care of bins that need servicing.
FR-N4	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 Sensoneo's Smart Waste Management Software System, a powerful cloud-based platform, for data-driven daily operations, available also as a waste management app.Customers are hence provideddata-driven decision making, and optimization of waste collection routes, frequencies, and vehicle loads resulting in routereduction by at least 30%.
FR-N5	Availability	By developing & deploying resilient hardwareand beautiful software we empower cities, businesses, and countries to manage wastesmarter.
FR-N6	Scalability	Using smart waste bins reduce the number of bins inside town , cities coz we able to monitorthe

5.PROJECT DESIGN

5.1.Data Flow Diagrams:

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



5.2.Technical Architecture:

Table-1 : Components & Technologies:

S. No	Component	Description	Technology
1	User Interface	Web Portal	HTML,CSS,NodeRed ,Javascript.or on

2	Application Logic-1	To calculate the distance of dreck and show the real timelevel in web portal , information getting via ultra sonic sensor and the alert message activatewith the python script to web portal.	Ultrasonic sensor/Python.
3	Application Logic- 2	To calculate the weight of the garbage and show the real time weight in web portal, this info getting via load cell and the alert message activate with python to web portal.	Load cell/Python.
4	Application Logic- 3	Getting location of the Garbage	GSM / GPS
5	Cloud Database.	Database Service on Cloud	IBM DB2, IBM Cloudant etc
6	File Storage	File storage requirements	Github,Local file system.
7	External API-1.	Firebase is a set of hosting services for any type of application. It offers NoSQL and real-time hosting of databases, content, social authentication, and notifications, or services, such as a real-time communication server.	Firebase.
8	Ultrasonic Sensor.	To throw alert message when garbage is getting full.	Distance Recognition Model.
9	Infrastructure (Server/Cloud)	Application Deployment on Local System / Cloud Local Server Configuration:localhostCloud ServerConfiguration: localhost,Firebase.	Localhost,Web portal.

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1	Open-Source Frameworks	NodeRed,Python,IBM Simulator.	IoT

2	Security Implementations	Raspberry Pi is connected to the internet and for example used to broadcast live data, further securitymeasures are recommended and use the UFW(uncomplicated Firewall).	IoT
3	Scalable Architecture	Raspberry pi:Specifications Soc:rspi ZERO W CPU: 32-bit computer with a 1GHz ARMv6RAM: 512MB Networking: Wi-Fi Bluetooth: Bluetooth 5.0, Bluetooth Low Energy(BLE).Storage: MicroSD GPIO: 40-pin GPIO header, populated Ports: micro HDMI 2.0,3.5mm analogue audio-video jack,2x USB 2.0, 2x USB 3.0, Ethernet Dimensions: 88mm x 58mm x19.5mm, 46g	IoT

6 .PROJECT PLANNING & SCHEDULING

6.1.Sprint Planning & Estimation

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers,research publications etc.	09 OCTOBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvasto capture the user Pains & Gains, Prepare list of problem statements	09 OCTOBER 2022
Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	10 OCTOBER 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	11 OCTOBER 2022

Problem Solution Fit	Prepare problem - solution fit document.	11 OCTOBER 2022
Solution Architecture	Prepare solution architecture document.	13 OCTOBER 2022

Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry toexit).	15 OCTOBER 2022
Functional Requirement	Prepare the functional requirement document.	17 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagramsand submit for review.	17 OCTOBER 2022
Technology Architecture	Prepare the technology architecture diagram.	19 OCTOBER 2022
Prepare Milestone & ActivityList	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.	15 NOVEMBER 2022

6.2.Sprint Delivery Schedule

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint EndDate (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	7 Nov 2022

7.CODING & SOLUTIONING

7.1.Feature 1

Python script

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


#Provide your IBM Watson Device Credentials
organization = "ncj2k2"
deviceType = "sample"
deviceId = "Mani1234"
authMethod = "token"
authToken = "fFP5C6f?fF-x+fLdiH"

# Initialize GPIO


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    else :
        print ("led is off")

    #print(cmd)




try:
  deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
  deviceCli = ibmiotf.device.Client(deviceOptions)
  #..............................................

except Exception as e:
  print("Caught exception connecting device: %s" % str(e))
  sys.exit()
```

```
# Connect and send a datapoint "hello" with value "world" into the cloud as an event
of type "greeting" 10 times
deviceCli.connect()

while True:
        #Get Sensor Data from DHT11

        weight=random.randint(0,100)
        level=random.randint(0,100)

        data = { 'weight' : weight, 'level':level  }
        #print data
        def myOnPublishCallback():
            print ("Published Weight = %s Kg" % weight, "level = %s %%" % level, "to
IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish
=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(1)

        deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

Java code :

```
#include <WiFi.h>
#include <PubSubClient.h>
#include "HX711.h"
WiFiClient wifiClient;
String data3;
#define ORG "ncj2k2"
#define DEVICE_TYPE "Ultrasonic_sensor"
#define DEVICE_ID "987654321"
#define TOKEN "bIqHU?Ocx!pYiRSYh+"
#define speed 0.034
#define led 12
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
```

```
PubSubClient client(server, 1883, wifiClient);
void publishData();

// HX711 circuit wiring
const int LOADCELL_DOUT_PIN = 5;
const int LOADCELL_SCK_PIN = 2;
const int trigpin=19;
const int echopin=18;
String command;
String data="";

long duration;
float dist;



void setup()
{
  Serial.begin(57600);
  pinMode(led, OUTPUT);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
  mqttConnect();
}

void loop() {
  bool isNearby = dist < 100;
  digitalWrite(led, isNearby);

  publishData();
  delay(500);

  if (!client.loop()) {
    mqttConnect();
  }
}

void wifiConnect() {
  Serial.print("Connecting to "); Serial.print("Wifi");
  WiFi.begin("Wokwi-GUEST", "", 6);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.print("WiFi connected, IP address: "); Serial.println(WiFi.localIP());
}
```

```
void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }
    initManagedDevice();
    Serial.println();
  }
}

void initManagedDevice() {
  if (client.subscribe(topic)) {
    // Serial.println(client.subscribe(topic));
    Serial.println("IBM subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}
void publishData()
{
  digitalWrite(trigpin,LOW);
  digitalWrite(trigpin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin,LOW);
  duration=pulseIn(echopin,HIGH);
  dist=duration*speed/2;
  if(dist<100){
    String payload = "{\"Alert Distance\":";
    payload += dist;
    payload += "}";

    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);
    if (client.publish(publishTopic, (char*) payload.c_str()))

{
      Serial.println("Publish OK");
    }

  }
    if(dist>101 && dist<400){
    String payload = "{\"normal distance\":";
    payload += dist;
    payload += "}";
    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);
     if(client.publish(publishTopic, (char*) payload.c_str()))

{
      Serial.println("Warning crosses 110cm -- it automaticaly of the loop");
      digitalWrite(led,HIGH);
    }else {
      Serial.println("Publish FAILED");
    }

  }


  }
  void callback(char* subscribeTopic, byte* payload, unsigned

int payloadLength){
  Serial.print("callback invoked for topic:");
  Serial.println(subscribeTopic);
  for(int i=0; i<payloadLength; i++){
    dist += (char)payload[i];
  }
  Serial.println("data:"+ data3);
  if(data3=="lighton"){
    Serial.println(data3);
    digitalWrite(led,HIGH);
  }
  data3="";
}
```

7.2.Feature 2

Node red-flows:

```
[
    {
        "id": "67d9b9581d15c0f4",
        "type": "tab",
        "label": "Flow 4",
        "disabled": false,
        "info": "",
        "env": []
    },
    {
        "id": "b8ac150df4d98308",
        "type": "ibmiot in",
        "z": "67d9b9581d15c0f4",
        "authentication": "apiKey",
        "apiKey": "bb0b8f8343d0f4e1",
        "inputType": "evt",
        "logicalInterface": "",
        "ruleId": "",
        "deviceId": "Mani1234",
        "applicationId": "",
        "deviceType": "sample",
        "eventType": "+",
        "commandType": "",
        "format": "json",
        "name": "IBM IoT",
        "service": "registered",
        "allDevices": "",
        "allApplications": "",
        "allDeviceTypes": false,
        "allLogicalInterfaces": "",
        "allEvents": true,
        "allCommands": "",
        "allFormats": true,
        "qos": 0,
        "x": 70,
        "y": 200,
        "wires": [
            [
                "c0dcb3de6ec26abf",
                "ef60c828df53e1c1",
                "f56ffb9a0a48df90"
            ]
        ]
    },
    {
        "id": "ef60c828df53e1c1",
        "type": "function",
        "z": "67d9b9581d15c0f4",
        "name": "weight",
        "func": "msg.payload=msg.payload.weight\nglobal.set('w',msg.payload)\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "initialize": "",
        "finalize": "",
        "libs": [],
        "x": 470,
        "y": 180,
        "wires": [
            [
                "70f4792a6c2339f1"
            ]
        ]
    },
    {
        "id": "f56ffb9a0a48df90",
        "type": "function",
        "z": "67d9b9581d15c0f4",
        "name": "level",
        "func": "msg.payload=msg.payload.level\nglobal.set('l',msg.payload)\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "initialize": "",
        "finalize": "",
        "libs": [],
        "x": 470,
        "y": 260,
        "wires": [
            [
                "d5cc58a0de0696a5"
            ]
        ]
    },
    {
        "id": "d5cc58a0de0696a5",
        "type": "ui_gauge",
        "z": "67d9b9581d15c0f4",
        "name": "level",
        "group": "453d6b39e8253096",
        "order": 4,
        "width": 0,
        "height": 0,
        "gtype": "wave",
        "title": "level",
        "label": "Cm",
        "format": "{{value}}",
        "min": "10",
        "max": "100",
        "colors": [
            "#00b500",
            "#e6e600",
            "#ca3838"
        ],
        "seg1": "",
        "seg2": "",
        "x": 670,
        "y": 260,
        "wires": []
    },
    {
        "id": "c0dcb3de6ec26abf",
        "type": "debug",
        "z": "67d9b9581d15c0f4",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "statusVal": "",
        "statusType": "auto",
        "x": 510,
        "y": 80,
        "wires": []
    },
    {
        "id": "f72df1465bd0fa53",
        "type": "http in",
        "z": "67d9b9581d15c0f4",
        "name": "",
        "url": "/sensor",
        "method": "get",
        "upload": false,
        "swaggerDoc": "",
        "x": 310,
        "y": 420,
        "wires": [
            [
                "0f10aa9afe1bb01b"
            ]
        ]
    },
    {
        "id": "75bc9c6d57087390",
        "type": "http response",
        "z": "67d9b9581d15c0f4",
        "name": "",
        "statusCode": "",
        "headers": {},
        "x": 750,
        "y": 420,
        "wires": []
    },
    {
        "id": "0f10aa9afe1bb01b",
        "type": "function",
        "z": "67d9b9581d15c0f4",
        "name": "",
        "func": "msg.payload={\"weight\":global.get(\"w\"),\"level\":global.get(\"l\")}\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "initialize": "",
        "finalize": "",
        "libs": [],
        "x": 540,
        "y": 420,
        "wires": [
            [
                "75bc9c6d57087390"
            ]
        ]
    },
    {
        "id": "70f4792a6c2339f1",
        "type": "ui_gauge",
        "z": "67d9b9581d15c0f4",
        "name": "weight",
        "group": "453d6b39e8253096",
        "order": 0,
        "width": 0,
        "height": 0,
        "gtype": "wave",
        "title": "weight",
        "label": "kg",
        "format": "{{value}}",
        "min": 0,
        "max": "100",
        "colors": [
            "#00b500",
            "#e6e600",
            "#ca3838"
        ],
        "seg1": "",
        "seg2": "",
        "x": 710,
        "y": 180,
        "wires": []
    },
    {
        "id": "bb0b8f8343d0f4e1",
        "type": "ibmiot",
        "name": "",
        "keepalive": "60",
        "serverName": "",
        "cleansession": true,
        "appId": "",
        "shared": false
    },
    {
        "id": "453d6b39e8253096",
        "type": "ui_group",
        "name": "Default",
        "tab": "381d0f64fdd864c3",
        "order": 1,
        "disp": true,
        "width": "6",
        "collapse": false
    },
    {
        "id": "381d0f64fdd864c3",
        "type": "ui_tab",
        "name": "Device",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
    }
]
```

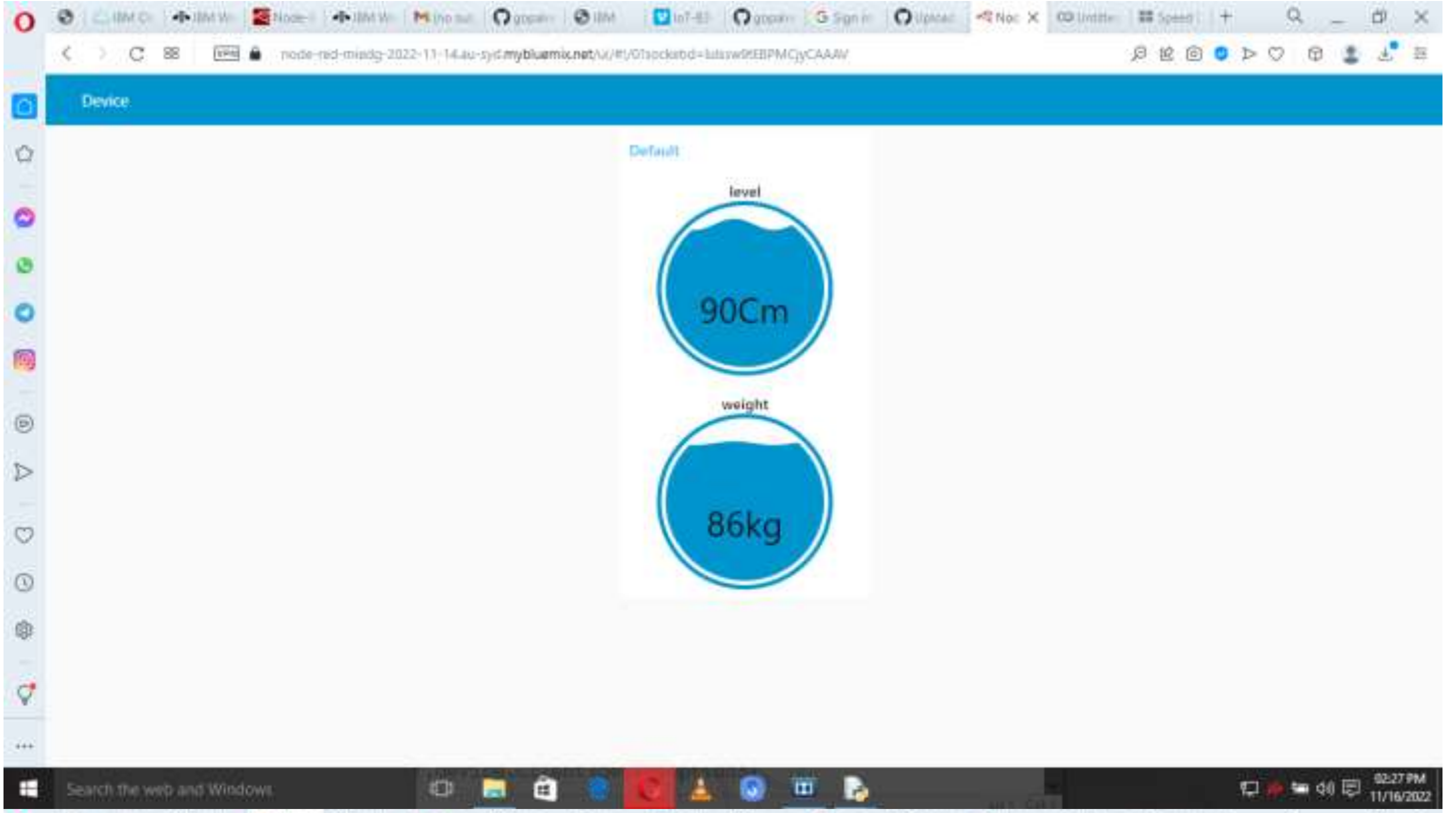
8.TESTING

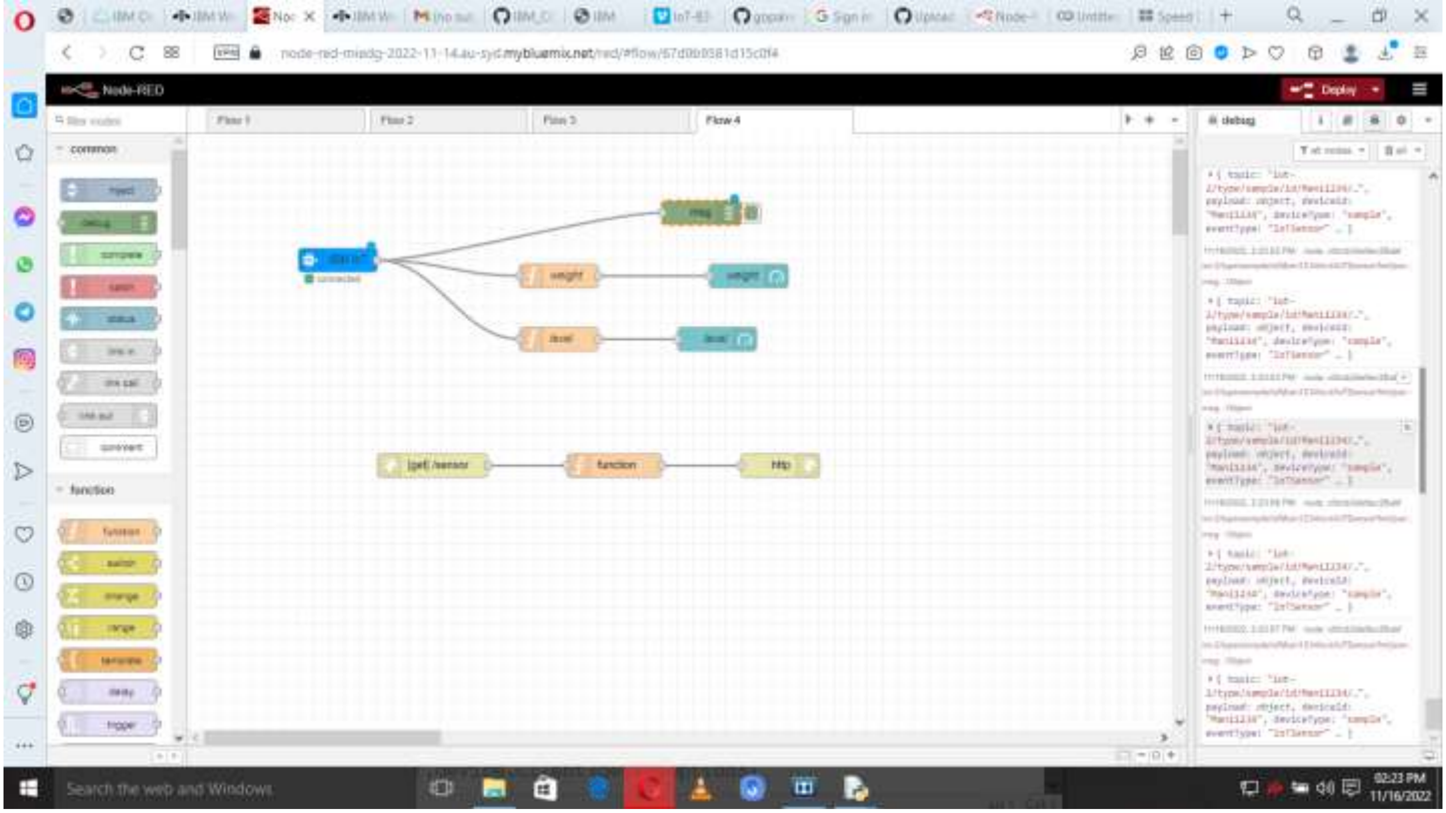
8.1.Test Cases

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

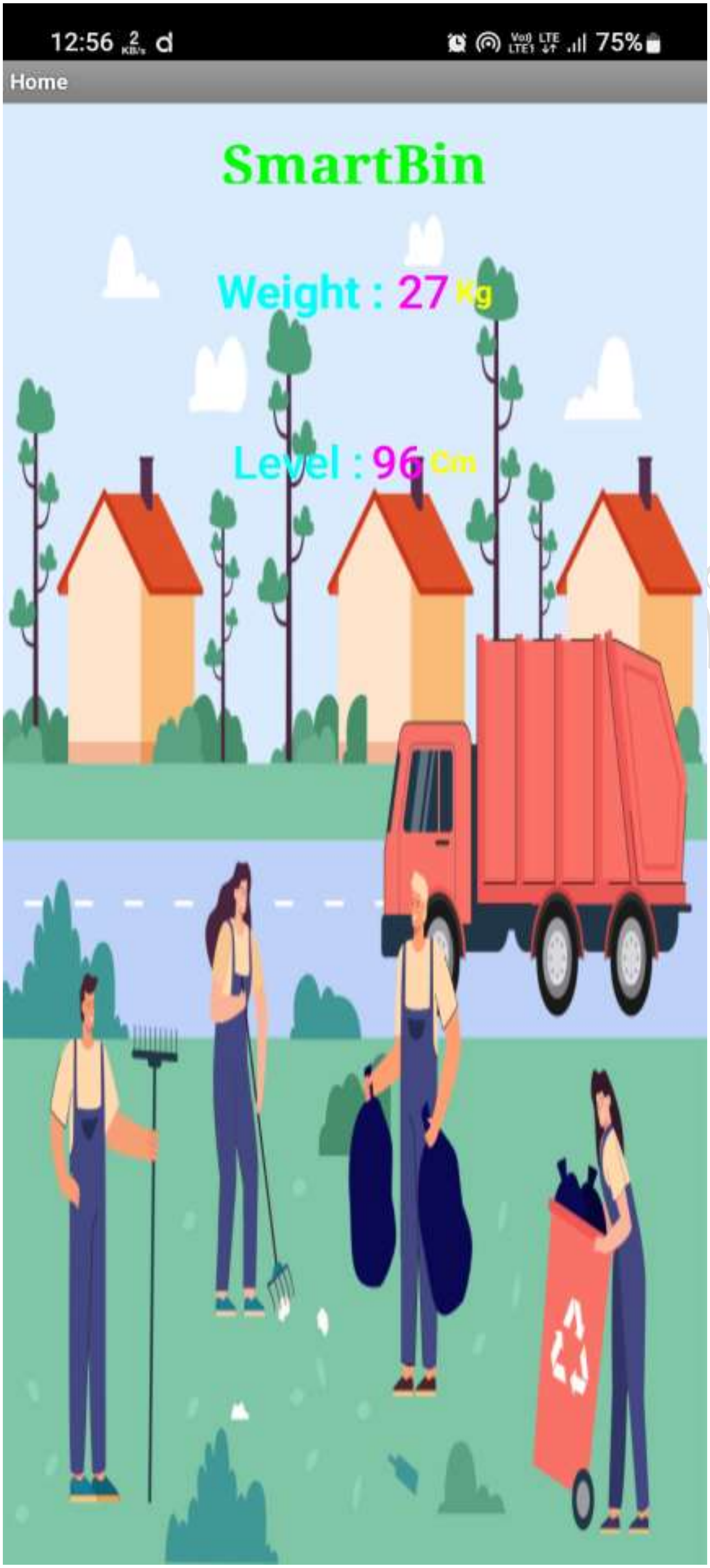
	Severity 1	**Severity 2**	**Severity 3**	**Severity 4**	**Subtotal**
By Design	5	2	2	1	10
Duplicate	1	0	2	0	3
External	2	1	0	1	4
Fixed	9	2	4	12	27
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	0	1
Won't Fix	0	1	0	0	1
Totals	17	6	10	14	47

9.RESULTS





```
▸{ topic: "iot-2/type/sample/id/Mani1234/...", payload: object, deviceId: "Mani1234", deviceType: "sample", eventType: "IoTSensor" … }
▸{ topic: "iot-2/type/sample/id/Mani1234/...", payload: object, deviceId: "Mani1234", deviceType: "sample", eventType: "IoTSensor" … }
▸{ topic: "iot-2/type/sample/id/Mani1234/...", payload: object, deviceId: "Mani1234", deviceType: "sample", eventType: "IoTSensor" … }
▸{ topic: "iot-2/type/sample/id/Mani1234/...", payload: object, deviceId: "Mani1234", deviceType: "sample", eventType: "IoTSensor" … }
▸{ topic: "iot-2/type/sample/id/Mani1234/...", payload: object, deviceId: "Mani1234", deviceType: "sample", eventType: "IoTSensor" … }
```



SmartBin

Weight : 27 Kg

Level : 96 Cm

10.Conclusion

Solid waste management is faced with a number of issues which include lack of throughput, inadequate solid waste data, efficiency problem, delays in collection and resistance to new technologies. Presently, waste management is a major problem for authorities who are responsible for such task because it's a costly service and it hugely impacts the environment as a whole. This study introduced a smart waste monitoring system that uses several sensors and communication technologies to achieve the set task. The proposed system was achieved through the development of theoretical models, layout and decision-making algorithms in the course of the project.There is an enormous amount of room for the development of this project in order for it to meet commercial standards. One of my many recommendations would be that of the addition of other sensors e.g. accelerometer. The accelerometer will make the system save more energy by turning on the system to measure the bin level only when the lid is opened to dispose waste. The system would then update its current state on Thing Speak and turn off, preventing unnecessary measurement when the bin's level has not been altered due to dormancy. Another recommendation is the use of solar panel for power generation making its power supply autonomous

Video reference link - *https://youtu.be/Lh8LkL2rJxE*