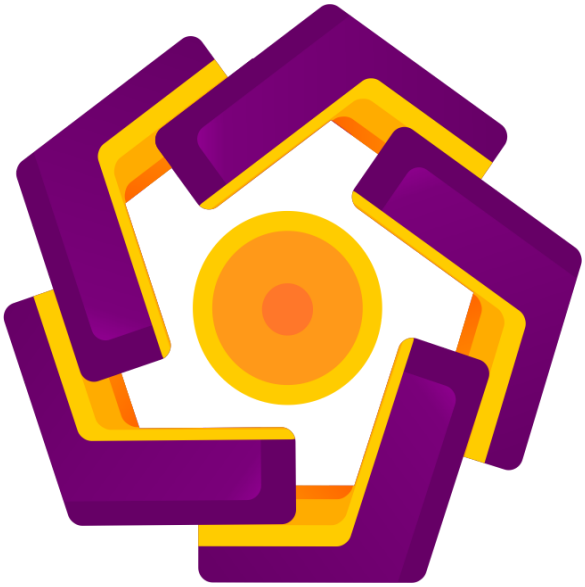
Smart DustBin Berbasis IOT



Di susun oleh

KELOMPOK Surya 16

Anggota:

Rifqi Faalih Khosyi	21.11.3824
Muhammad Rizky K	21.11.3846
Akmal Bayu Wardana	21.11.3852

UNIVERSITAS AMIKOM YOGYAKARTA

FAKULTAS ILMU KOMPUTER

INFORMATIKA

2023/2024

1. **Latar Belakang**

Smart dustbin, atau tempat sampah pintar, muncul sebagai solusi inovatif dalam mengatasi masalah manajemen sampah di perkotaan. Dengan pertumbuhan populasi dan urbanisasi yang terus meningkat, peningkatan volume sampah menjadi isu yang semakin mendesak. Masalah ini dapat berdampak negatif pada lingkungan jika tidak diatasi dengan baik, menciptakan pencemaran dan risiko kesehatan. Seiring perkembangan teknologi informasi dan komunikasi (TIK),

Konsep smart dustbin menggunakan sensor-sensor untuk mendeteksi level sampah di dalamnya, dan mengirim data secara real-time ke sistem manajemen sampah. Ketika tempat sampah hampir penuh, sistem ini memberi peringatan kepada petugas sampah atau mengatur jadwal pengambilan sampah secara otomatis, mengoptimalkan proses pengumpulan sampah dan mengurangi biaya operasional.

Dengan perkembangan kota pintar dan Internet of Things (IoT), smart dustbin menjadi salah satu tren masa depan dalam upaya meningkatkan manajemen sampah, mengurangi dampak lingkungan negatif, dan menciptakan perkotaan yang lebih efisien dan berkelanjutan.

2. **Analisa Proyek IoT**

a. Analisa Kebutuhan Fungsional

Berikut adalah analisis kebutuhan fungsional untuk Smart DustBin :

1. Pemantauan tingkat isi sampah melalui sensor, dengan notifikasi saat mencapai batas tertentu.
2. Integrasi sensor untuk mengukur sejauh mana dustbin terisi
3. Pelacakan data historis tentang penggunaan dan tingkat isi sampah.
4. Pelacakan dimana Smart Dustbin tersebut berada

b. Analisa Kebutuhan Non Fungsional

Berikut adalah analisis kebutuhan non fungsional untuk Smart DustBin :

1. Keamanan:

   - Proteksi data pengguna dan statistik penggunaan dustbin.

2. Kehandalan:

   - Dustbin harus konsisten dalam mendeteksi isi sampah dan memberikan informasi yang akurat.

3. Kinerja:

   - Pengukuran dan pemantauan tingkat isi sampah dengan akurasi tinggi.

4. Antarmuka Pengguna:

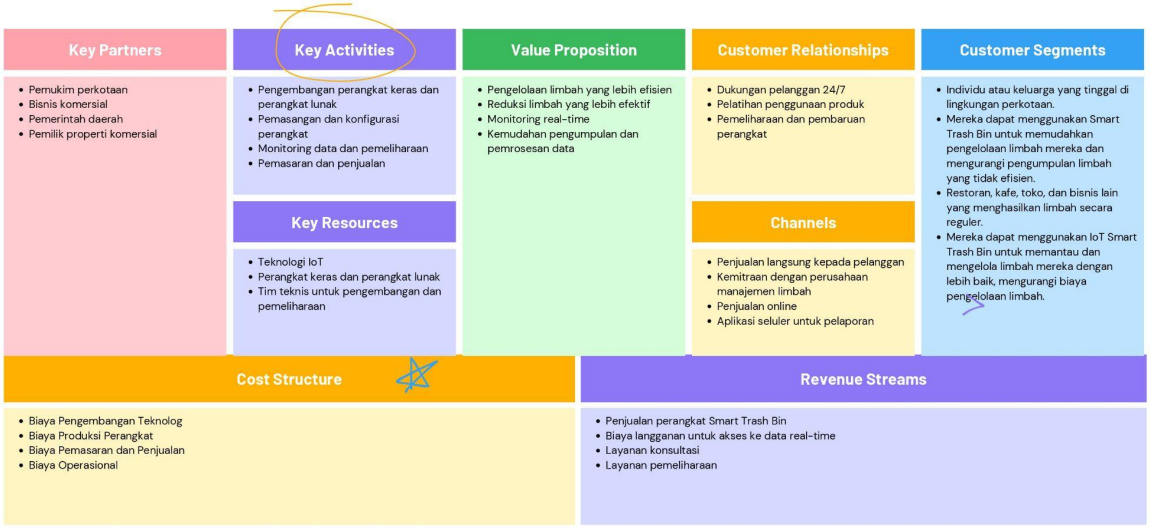
   - Notifikasi yang jelas dan informatif kepada pengguna.

c. Analisa Kebutuhan Pengguna

Analisis Kebutuhan pengguna untuk Smart DustBin :

1. Kemudahan Penggunaan:
    - Desain Ergonomis
2. Efisiensi Pengumpulan Sampah:
    - Sensor Isi Sampah
    - Notifikasi
3. Keamanan dan Perlindungan Data:
    - Perlindungan Data Pribadi
    - Keamanan Fisik
4. Ketersediaan Layanan dan Dukungan:
    - Layanan Pelanggan
    - Pembaruan Perangkat Lunak

d. Analisa Bisnis (Business Model Canvas)



e. Analisa Keamanan IoT (Framework ARMET)

Berikut adalah analisis keamanan sistem Smart DustBin :

1. Identifikasi Ancaman:
    - Mengidentifikasi potensi ancaman keamanan yang dapat mempengaruhi smart dustbin, seperti serangan fisik, serangan jaringan, atau ancaman perangkat lunak.
    - Menganalisis kemungkinan risiko dan dampak dari setiap ancaman yang diidentifikasi.
2. Keamanan Perangkat Keras:
    - Memastikan keamanan fisik smart dustbin untuk mencegah manipulasi atau pencurian data.
    - Menggunakan prosedur otentikasi yang kuat untuk melindungi akses ke perangkat.
3. Keamanan Jaringan:
    - Mengimplementasikan enkripsi data untuk melindungi komunikasi antara smart dustbin dan server atau perangkat lainnya.
    - Menerapkan firewall dan teknologi pengamanan jaringan untuk mencegah akses yang tidak sah.
4. Proteksi Data:
    - Melindungi data pengguna dengan menerapkan enkripsi end-to-end.
    - Mematuhi regulasi privasi data yang berlaku dan memberikan kontrol yang adekuat kepada pengguna terkait pengumpulan dan penggunaan data.
5. Keamanan Perangkat Lunak:
    - Memastikan bahwa perangkat lunak pada smart dustbin diperbarui secara berkala untuk mengatasi kerentanan keamanan yang baru ditemukan.

- Menggunakan mekanisme otentikasi yang kuat dan mengelola kredensial dengan aman.

6. Manajemen Akses:
    - Menerapkan kontrol akses yang tepat untuk memastikan bahwa hanya pengguna yang sah yang memiliki akses ke fungsi dan data tertentu.
    - Memastikan bahwa setiap entitas yang terhubung ke smart dustbin memiliki identitas yang diverifikasi.
7. Monitoring dan Respons Keamanan:
    - Menerapkan sistem pemantauan untuk mendeteksi aktivitas yang mencurigakan atau potensi serangan.
    - Mengembangkan rencana respons keamanan yang efektif untuk menanggapi insiden keamanan dengan cepat.

f. Analisa Kebutuhan Service IoT (Http atau Mqtt)

1. Konektivitas MQTT:

- Pub/Sub (Publisher/Subscriber): Implementasi model pub/sub untuk mentransmisikan informasi antara smart dustbin dan platform layanan.
- QoS (Quality of Service): Tentukan tingkat QoS yang diperlukan untuk memastikan kehandalan pengiriman pesan.

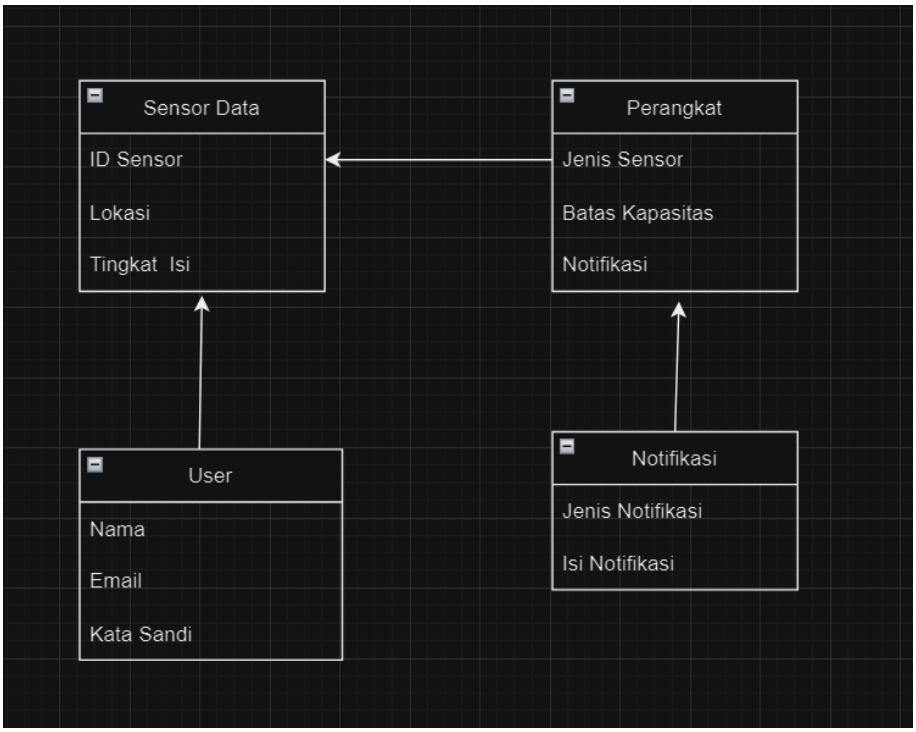
2. Sensor dan Data:

- Integrasi Sensor: Layanan harus dapat mengintegrasikan data dari sensor-sensor smart dustbin, seperti sensor level sampah dan sensor keberadaan sampah.
- Pemrosesan Data: Menyediakan fasilitas untuk pemrosesan data yang efisien dan mampu mengekstrak informasi yang berarti dari sensor.

3. Manajemen Kapasitas:

- Notifikasi Kapasitas: Layanan perlu memberikan notifikasi ketika kapasitas smart dustbin mencapai ambang batas tertentu.
- Monitoring Kapasitas: Memungkinkan pemantauan jarak jauh terhadap kapasitas dan status smart dustbin.

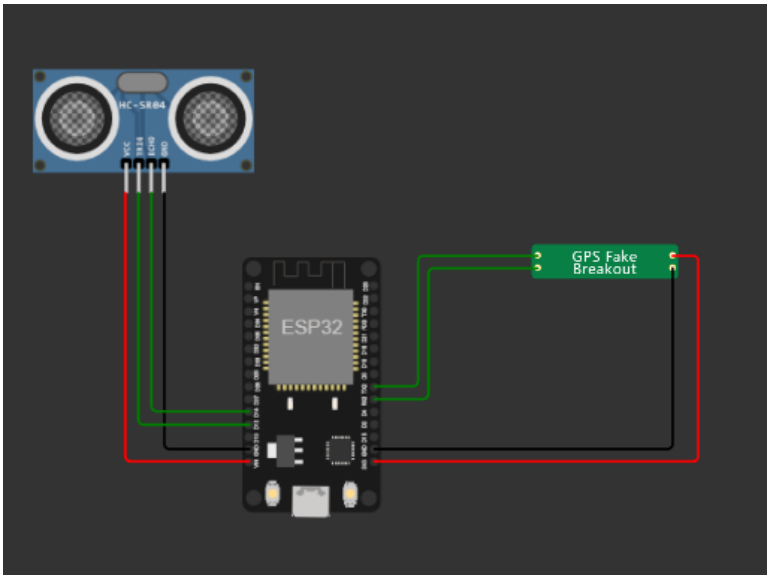
g. Analisa Kebutuhan Server Database IoT (ERD)



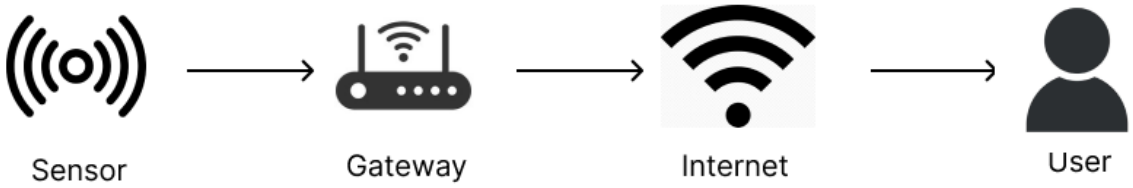
3. Rancangan Server IoT

a. Desain Topologi Server

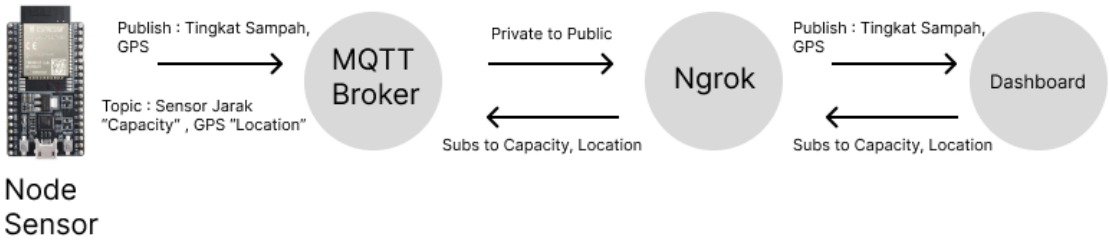
i. Rancangan Node Sensor (mikrokontroller)



ii. Rancangan WSN



iii. Rancangan Server IoT (private ke publik)



b. Kode Program Server

i. Kode Program Server

```
from flask import Flask, render_template

import paho.mqtt.client as mqtt



app = Flask(__name__)



# MQTT Settings

MQTT_BROKER = "test.mosquitto.org"

MQTT_PORT = 1883

MQTT_TOPICS = [("/Thinkitive/longitude", "longitude"),
               ("/Thinkitive/latitude", "latitude"),
               ("/Thinkitive/distance", "distance")]
```

```python
# MQTT Callbacks
def on_connect(client, userdata, flags, rc):
    print("Connected to MQTT broker with result code "+str(rc))
    for topic, _ in MQTT_TOPICS:
        client.subscribe(topic)

def on_message(client, userdata, msg):
    topic = msg.topic
    payload = msg.payload.decode("utf-8")
    print(f"Received message on topic {topic}: {payload}")
    update_data(topic, payload)

# Initialize MQTT Client
mqtt_client = mqtt.Client()
mqtt_client.on_connect = on_connect
mqtt_client.on_message = on_message
mqtt_client.connect(MQTT_BROKER, MQTT_PORT, 60)

# Dashboard Data
dashboard_data = {topic: "-" for _, topic in MQTT_TOPICS}

# Update Dashboard Data
```

```python
def update_data(topic, value):
    dashboard_data[topic] = value


# Flask Routes
@app.route("/")
def index():
    return render_template("aw.py", data=dashboard_data)


if __name__ == "__main__":
    mqtt_client.loop_start()
    app.run(debug=True)
```

ii. Kode Program Node Sensor (mikrokontroller)

```cpp
#include <WiFi.h>
#include <PubSubClient.h>
#include "NMEA.h"
#include <Ultrasonic.h>


const char* ssid = "Wokwi-GUEST";
const char* password = "";
const char* mqqt_server = "test.mosquitto.org";
```

```
WiFiClient espClient;

PubSubClient client(espClient);

unsigned long lastMsg = 0;



#define TRIGGER_PIN 12

#define ECHO_PIN 14

Ultrasonic ultrasonic(TRIGGER_PIN, ECHO_PIN);



float latitude;

float longitude;



NMEA gps(GPRMC);



union {

  char bytes[4];

  float valor;

} velocidadeGPS;



void setup_wifi() {

  delay(10);

  Serial.println();

  Serial.print("Connecting to ");

  Serial.println(ssid);
```

```
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("]");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
```

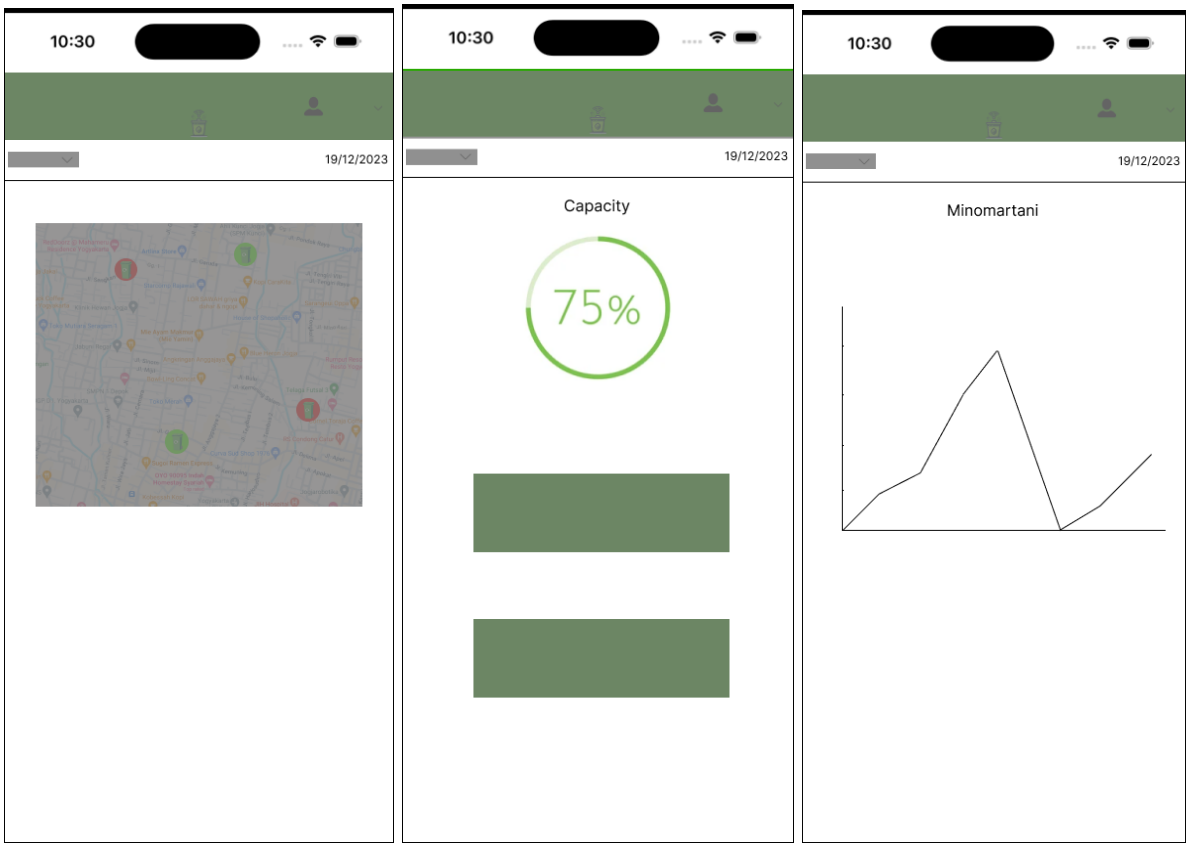
```
  }

}



void reconnect() {

  while (!client.connected()) {

    Serial.print("Attempting MQTT connection...");

    String clientId = "ESP32CLIENT-";

    clientId += String(random(0xffff), HEX);

    if (client.connect(clientId.c_str())) {

      Serial.println("Connected");

      client.publish("/ThinkIOT/Publish.", "Welcome");

      client.subscribe("/ThinkIOT/Subscribe");

    } else {

      Serial.print("failed, rc=");

      Serial.print(client.state());

      Serial.println(" try again in 5 seconds");

      delay(5000);

    }

  }

}



void setup() {

  Serial.begin(115200);
```

```
  setup_wifi();
  client.setServer(mqqt_server, 1883);
  client.setCallback(callback);
  Serial2.begin(9600);
  Serial.println("Data received from GPS Fake:");
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  unsigned long now = millis();
  if (now - lastMsg > 2000) {
    lastMsg = now;

    while (Serial2.available()) {
      char serialData = Serial2.read();
      Serial.print(serialData);

      if (gps.decode(serialData)) {
        Serial.println();
```

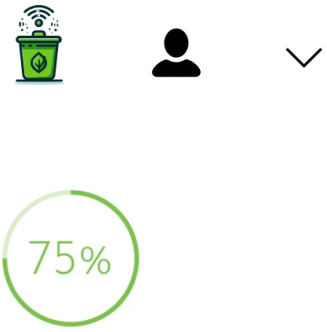
```
        Serial.println();


        Serial.print(" Latitude: ");

        Serial.println(latitude, 8);


        Serial.print("Longitude: ");

        Serial.println(longitude, 8);


        long distance = ultrasonic.read();


        Serial.print("Distance: ");

        Serial.print(distance);

        Serial.println(" cm");


        if (distance < 200) {

          Serial.println("DustBin Full");

        }

      }

    }

  }

}
```

4. **Rancangan Dashboard IoT**

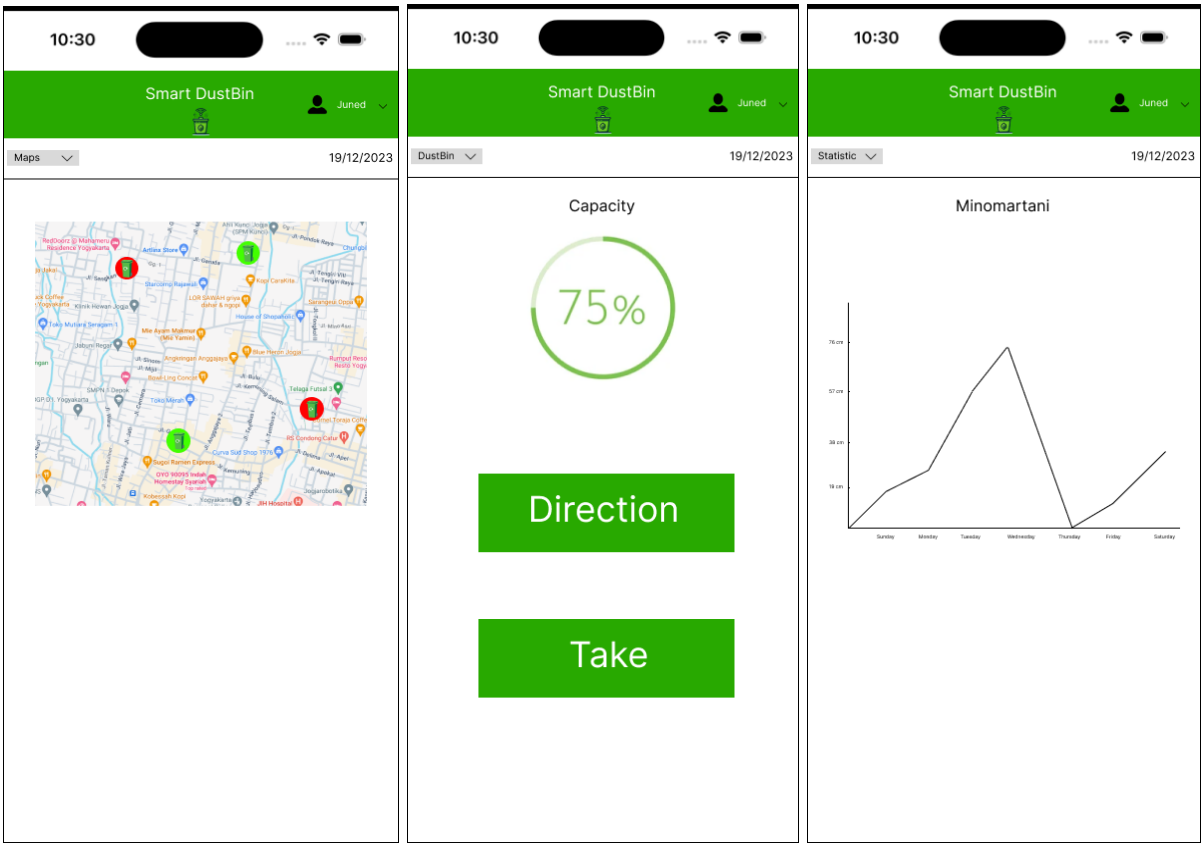
a. Wireframe Dashboard IoT (dijelaskan berdasarkan analisa kebutuhan)



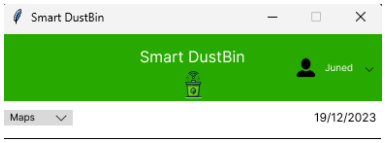
b. Desain Asset

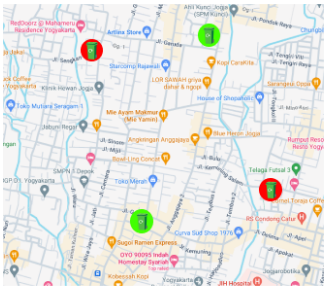


c. High Fidelity Dashboard IoT

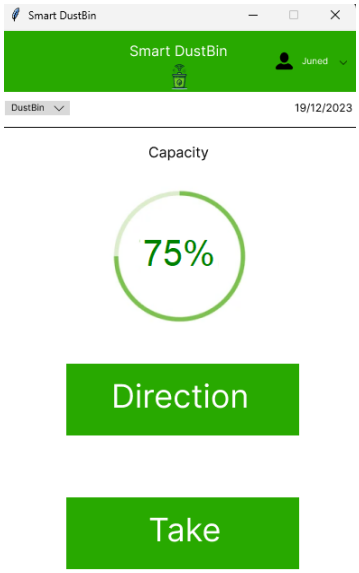


5. **Unit Testing (Blackbox Testing)**

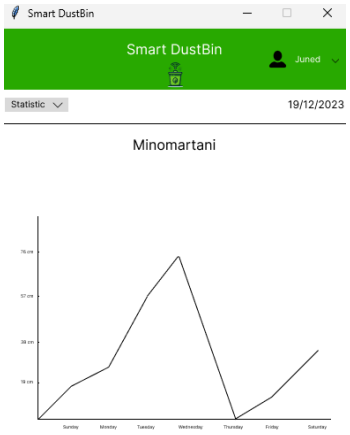




Tampilan yang menunjukkan peta dimana smart dustbin itu berada apabila hijau artinya dustbin masih kosong apabila merah artinya dustbin sudah penuh



Tampilan apabila kita mengklik salah satu tempat sampah maka akan muncul tingkat kepenuhan dari dustbin itu apabila kita pencet direction maka akan menunjukkan rute dimana dustbin itu berada



Statistic suatu dustbin di komplek apabila grafik naik maka tingkat kepenuhan mengalami kenaikan terus, apabila drop ke 0 maka artinya sampahnya sudah diambil oleh petugas

6. Kesimpulan

Dalam proyek pengembangan smart dustbin, ditemukan bahwa pemanfaatan teknologi untuk meningkatkan efisiensi manajemen sampah memiliki dampak positif yang signifikan. Melalui penggunaan sensor dan teknologi Internet of Things (IoT), smart dustbin mampu secara real-time memantau tingkat isi sampah, memungkinkan pengumpulan sampah yang tepat waktu dan efisien. Dengan adanya data yang dihasilkan oleh smart dustbin, proyek ini tidak hanya mengoptimalkan alokasi sumber daya dan mereduksi biaya operasional, tetapi juga memberikan wawasan yang berharga untuk perencanaan perkotaan yang berkelanjutan. Selain itu, penerapan teknologi ini dapat meningkatkan kesadaran masyarakat tentang kebersihan lingkungan dan pentingnya pembuangan sampah yang bertanggung jawab.