TESIS

OPTIMASI RUTE MENGGUNAKAN ALGORITMA *GREEDY* PADA PENGANGKUTAN SAMPAH DI KOTA MAKASSAR

ROUTE OPTIMIZATION USING GREEDY ALGORITHM ON WASTE TRANSPORTATION IN MAKASSAR CITY

OLEH :

ALVIAN TRI PUTRA DARTI AKHSA

2016130017

Dosen Pembimbing :
Dr. Ir. Zahir Zainuddin, M.Sc
Prof. Dr. Ir. Andani Achmad, MT

PROGRAM PASCASARJANA

PROGRAM STUDI SISTEM KOMPUTER

STMIK HANDAYANI MAKASSAR

2018

TESIS

OPTIMASI RUTE MENGGUNAKAN ALGORITMA *GREEDY* PADA PENGANGKUTAN SAMPAH DI KOTA MAKASSAR

Disusun dan diajukan oleh

ALVIAN TRI PUTRA DARTI AKHSA
NOMOR POKOK : 2016130017

Telah dipertahankan di depan Dewan Penguji Ujian Tesis

Pada hari **Kamis, 04 Oktober 2018**

Dan dinyatakan telah memenuhi syarat

Menyetujui,

Komisi Penasihat,

Dr. Ir. Zahir Zainuddin, M.Sc.	**Prof. Dr. Ir. Andani Achmad, M.T**
Ketua	Anggota

Mengetahui,

Ketua Program Studi S2 Sistem Komputer,	Direktur Program Pascasarjana STMIK Handayani,
Dr. Ir. Zahir Zainuddin, M.Sc.	**Dr. Rabiatul Adawiyah, M.Pd**

KATA PENGANTAR

Bismillahirahmanirrahim,

Alhamdulillah, Puji syukur penulis panjatkan kehadirat Allah SWT yang telah melimpahkan Rahmat-Nya kepada penulis, sehingga penulis dapat menyusun dan menyelesaikan laporan penelitian ini sebagai tugas akhir sebelum menyelesaikan program studi sistem komputer di pascasarjana Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Handayani Makassar.

Penulis menyadari, tanpa bantuan dan motivasi serta bimbingan baik moril maupun materi dari berbagai pihak, maka proposal ini tidak dapat terselesaikan dengan baik. Oleh sebab itu, penulis mengucapkan banyak terima kasih kepada :

1. Kedua orang tua penulis, Ir. Akhmad Sanusi dan Hj. Darmiati yang tak mengenal lelah untuk membimbing penulis agar menjadi manusia yang berdaya guna di hari kelak dan tak henti-hentinya memberikan bantuan doa, moril, serta materialnya.
2. Kepada ibu Dr. Rabiatul Adawiyah, M.Pd selaku direktur pascasarjana STMIK Handayani Makassar
3. Bapak Dr. Ir. Zahir Zainuddin, M.Sc., selaku ketua program studi S2 sistem komputer.
4. Bapak Dr. Ir. Zahir Zainuddin, M.Sc., selaku dosen pembimbing pertama yang bersedia meluangkan waktu untuk membimbing penulis.

5. Bapak Prof. Dr. Ir. Andani Achmad, MT, selaku dosen pembimbing kedua yang selalu memberikan arahan agar penulis dapat menyelesaikan tugas akhir ini dengan baik.
6. Bapak Dr. Ir. Zulfajri B. Hasanuddin, M.Eng, Dr. IT. Supriadi Sahibu, S.Kom, MT, dan Dr. Eng. Agussalim, MT selaku dosen penguji yang telah memberikan kritik dan saran mulai pembuatan proposal hingga penyusunan laporan ini.
7. Teman-teman seperjuangan pascasarjana angkatan 2016 yang telah banyak membantu penulis mulai penyusunan proposal hingga penyusunan laporan ini.

Tak lupa pula penulis mengucapkan banyak terima kasih kepada semua pihak yang telah membantu mulai dari penyusunan proposal, pembuatan aplikasi hingga penyusunan laporan tesis yang tidak dapat penulis sebutkan satu-persatu. Semoga Allah Swt membalas semua bantuan yang telah diberikan kepada penulis. Aamiin.

Makassar, 15 Oktober 2018

Penulis

ABSTRAK

Alvian Tri Putra D.A (2016130017). Optimasi Dan Implementasi Algoritma *Greedy* Pada Pengangkutan Sampah Di Kota Makassar, (dibimbing oleh **Zahir Zainuddin** dan **Andani Achmad**)

Optimasi adalah suatu bentuk mengoptimalkan sesuatu hal yang sudah ada, ataupun merancang dan membuat sesusatu secara optimal saat ini pola pengangkutan sampah di Kota Makassar khususnya di kecamatan tamalanrea kelurahan tamalanrea, dimana dump truck melakukan pengangkutan sampah 2 kali dalam sepekan hal ini menyebabkan seringnya terjadi penumpukan sampah pada setiap TPS serta rute pengangkutan tidak efisien.

Sampah yang tidak dikelola dengan baik tentunya akan berdampak terhadap nilai dan fungsi lingkungan, oleh karena itu diperlukan suatu sistem optimasi pengangkutan sampah yang efektif dimana Informasi volume TPS diambil dari sensor yang terpasang disetiap TPS yang kemudian disimpan ke database web server dan dilakukan optimasi menggunakan algoritma *greedy* dengan mencari nilai maksimum pada setiap langkahnya. Nilai fitness maximum didapatkan dari hasil optimasi total volume TPS dengan jarak terpendek.

Berdasarkan hasil perhitungan Algoritma *Greedy* optimasi rute terpendek dan volume TPS menghasilkan rute kunjungan yaitu dimulai dari (RT.1 -> RT.6 -> RT.5 -> RT.4 -> RT.3 -> RT.18 -> RT.17 -> RT.16 -> RT.15 -> RT.5 -> RT.8 -> RT.7 -> RT.10 -> RT.11 -> RT.12 -> RT.13 -> RT.2 -> RT.14 -> RT.9) dengan total jarak tempuh 18.332 Km dan volume sampah sebanyak 10.000 liter.

Kata kunci: Web *server*, sensor, algoritma *greedy*, jarak, rute

ABSTRACT

Alvian Tri Putra D.A (2016130017). Optimization and Implementation of Greedy Algorithm in Garbage Transport in Makassar City is (guided by **Zahir Zainuddin and Andani Achmad**)

Optimization is a form of things that already exist, or make an optimal arrangement at this time in trash patterns in Makassar City, especially in Tamalanrea subdistrict, Tamalanrea, where dump trucks transport garbage 2 times a week, causing frequent accumulation of garbage. at each polling station and inefficient transportation routes.

Waste that is not managed properly will have an impact on the value and function of the environment, therefore it is an information system that allows to access TPS which is then saved to the web server database and perform optimization using greedy algorithms. by looking for maximum values at each step. The maximum fitness value obtained from the optimization of the total TPS volume with the shortest distance.

Based on the Greedy Algorithm and TPS volume optimization algorithm calculation results, the visit starts from (RT.1 -> RT.6 -> RT.5 -> RT.4 -> RT.3 -> RT.18 -> RT. 17 -> RT.16 -> RT.15 -> RT.5 -> RT.8 -> RT.7 -> RT.10 -> RT.11 -> RT.12 -> RT.13 -> RT. 2 -> RT.14 -> RT.9) with a total mileage of 18,332 Km and a volume of waste of 10,000 liters.

Keywords: web server, sensor, *greedy* algorithm, distance, route

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Nomor **Halaman**

DAFTAR TABEL

Nomor **Halaman**

BAB I

PENDAHULUAN

A. Latar Belakang

Pengumpulan sampah adalah salah satu masalah logistik yang paling kompleks yang dihadapi kota manapun [1]. Dalam beberapa tahun terakhir kenaikan harga BBM, biaya operasional dan beban regulasi yang berkembang telah menyebabkan perusahaan pengumpulan sampah baik negara maupun swasta untuk mengoptimalkan rute pengumpulan sampah mereka [1]. Menurut penelitian VN. Bhat (1996) biaya transportasi merupakan antara 70% dan 80% dari seluruh biaya operasional dalam pengumpulan sampah [2]. Oleh karena itu bahkan perbaikan kecil di rute pengumpulan sampah dapat menyebabkan penghematan besar [1].

Pada sektor publik, transportasi pengiriman surat dan pengumpulan sampah menghasilkan biaya yang cukup besar. Tingginya biaya yang diserap transportasi pengumpulan sampah bila dibandingkan dengan kegiatan logistik lainnya menjadi potensi besar untuk optimalisasi dan penghematan [3]. Di masa lalu, pengumpulan sampah dilakukan tanpa menganalisis permintaan dan pembangunan rute ke truk pengangkut. Meningkatnya pembangunan di Kota Makassar serta pertambahan jumlah penduduk, tingkat aktivitas dan tingkat sosial ekonomi masyarakat menyebabkan meningkatnya jumlah timbulan sampah di Kota Makassar.

Karena urbanisasi berkelanjutan ini, pentingnya sistem pengumpulan yang meningkatkan efisiensi dan optimalisasi pengangkutan sampah karena keterbatasan truk pengangkut [4]. Saat ini pola pengangkutan sampah di Kota Makassar sudah terjadwal yang ditetapkan oleh pemerintah Kota Makassar khususnya di kecamatan tamalanrea kelurahan tamalanrea, dimana dump truck mengangkut sampah di setiap rumah 2 kali dalam sepekan hal ini menyebabkan seringnya terjadi penumpukan sampah pada setiap rumah karena waktu pengangkutan yang tidak efisien serta rute yang qharus dilaluli oleh truk pengangkut sampah tidak terstruktur sehingga memerlukan waktu yang lama dalam proses pengangkutan sampah.

Khusus di Kota Makassar dengan jumlah penduduk mencapai 1,4 juta jiwa, menghasilkan sekitar 4500 m3 sampah setiap harinya, volume sampah di Kota Makassar bertambah 200 ton per hari, dimana setiap bulannya sampah berkisar antara 600 ton – 800 ton, sehingga bisa di prediksi kalau Volume sampah di Kota Makassar cukup tinggi. Kota dengan luasan 177.557 ha, ini mampu memproduksi sampah hingga 550 ton, atau sekira 4.000 meter kubik per hari. , sedangkan dinas pertamanan dan kebersihan Kota Makassar hanya mampu menangani sekitar 3500 m3 setiap hari. Berarti, ada sekitar 1000 m3 sampah di Kota Makassar yang tidak tertangani di tengah.

Sampah yang tidak dikelola dengan baik tentunya akan berdampak terhadap nilai dan fungsi lingkungan, oleh karena itu diperlukan suatu sistem **Optimasi Rute Menggunakan Algoritma *Greedy* Pada**

Pengangkutan Sampah Di Kota Makassar yang dapat membantu program pemerintah untuk mewujudkan lingkungan yang bersih serta memberikan kepuasan pelayanan kepada masyarakat di Kota Makassar.

B. Rumusan Masalah

Berdasarkan latar belakang yang telah dibahas dan di identifikasi, maka perumusan masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana mendapatkan informasi volume TPS dari mikrokontroler yang terpasang disetiap TPS ?
2. Bagaimana menambah efisiensi terhadap cara pengangkutan sampah di TPS masyarakat ?
3. Bagaimana menyajikan informasi titik lokasi angkutan kepada operator truk yang bertugas di lapangan ?

C. Tujuan Penelitian

1. Membuat *script code* untuk pengambilan data dari server *thingspeak* dan menyimpan informasi data volume TPS di database *web server*.
2. Mengimplementasikan algoritma *greedy* untuk mencari rute terbaik pengangkutan sampah dengan nilai fitness maximum di dapatkan dari hasil optimasi total volume TPS setiap RT dengan jarak terpendek.
3. Membuat sistem optimasi untuk menampilkan rute yang dapat diakses oleh operator truk sebagai panduan rute kunjungan pengangkutan TPS.

D. Batasan Masalah

1. Tidak membahas faktor hujan, kemacetan dan kondisi jalan
2. Hanya menampilkan rute angkutan dari titik awal truk ke setiap RT yang dikalkulasi dari setiap TPS.

E. Manfaat Penelitian

Adapun manfaat dari penelitian ini diharapkan dapat menghasilkan rancangan sistem untuk optimalisasi pengangkutan sampah di Kota Makassar yang dapat membantu pemerintah dalam menghemat bahan bakar truk pengangkut sampah serta waktu kerja yang efisien untuk mewujudkan lingkungan yang bersih dan meningkatkan kepuasan pelayanan kepada masyarakat.

F. Metedologi Penelitian

1. Jenis dan Sumber Data
    a. Primer

        Data primer adalah data yang berasal dari pihak yang bersangkutan dimana data rumah tangga per wilayah serta data wilayah pengangkutan dari Dinas Pertamanan & Kebersihan Kota Makassar dan Kecamatan Tamalanrea.

    b. Sekunder

        Data sekunder adalah data yang berasal dari luar yang berkaitan dengan penelitian yang dilakukan, berupa referensi buku.

G. Metode Analisis

Metode analisis merupakan suatu bentuk penganalisaan di dalam menguraikan informasi ke dalam bagian-bagian atau komponen-komponen dengan maksud mengidentifikasi, mengevaluasi setiap permasalahan-permasalahan yang timbul. Metode analisis meliputi :

1. Analisis Data

   Metode yang digunakan pada analisis data yaitu metode kualitatif berupa penganalisaan suatu data dan informasi yang bersifat teoritis.

2. Analisis System

   Metode analisis sistem yang digunakan adalah UML (Unified Modeling Language) yaitu sebuah standar untuk merancang dan mendokumentasikan sistem piranti lunak terkhusus pada sistem yang secara procedural.

3. Analisis Pengujian

   Metode pengujian yang digunakan yaitu *white box* testing didasarkan pada pengecekan terhadap detail perancangan, menggunakan struktur kontrol dari desain program secara procedural untuk membagi pengujian ke dalam beberapa kasus pengujian.

BAB II

TINJAUAN PUSTAKA

A. Optimasi

Optimasi berasal dari kata dasar optimal yang berarti terbaik, tertinggi, paling menguntungkan, menjadikan paling baik. Optimasi adalah suatu proses untuk mencapai hasil yang ideal atau optimasi (nilai efektif yang dapat dicapai). Optimasi dapat diartikan sebagai suatu bentuk mengoptimalkan sesuatu hal yang sudah ada, ataupun merancang dan membuat sesusatu secara optimal. (https://id.wikipedia.org/wiki/Optimasi).

B. Sistem

Sistem adalah setiap sesuatu yang terdiri dari obyek-obyek, atau komponenkomponen yang berkaitan, tertata dan saling berhubungan satu sama lain sedemikian rupa sehingga unsur-unsur tersebut menjadi satu kesatuan dari pemrosesan atau pengolahan data tertentu. Menurut Lukas dalam buku Sistem Informasi Manajemen menyatakan bahwa: “Sistem adalah kumpulan atau himpunan dari unsur, komponen, atau variabel-variabel yang terorganisir, saling berinteraksi, saling tergantung satu sama lain dan terpadu” (https://arifashkaf.wordpress.com/2015/10/14/pengertian-sistem-dan-contohnya-softskill/).

C. *Vehicle Routing Problem* (VRP)

Vehicle Routing Problem (*VRP*) pertama kali diperkenalkan oleh Dantzig dan Ramser, pada tahun 1959 dan semenjak itu telah dipelajari secara luas. Oleh Fisher, *VRP* didefinisikan sebagai sebuah cara pencarian

atas penggunaan yang efisien dari sejumlah kendaraan yang harus melakukan perjalanan untuk mengunjungi sejumlah tempat untuk mengantar dan menjemput orang atau barang. Istilah konsumen menunjukkan pemberhentian untuk mengantar dan menjemput orang/barang. Setiap konsumen harus dilayani oleh satu kendaraan saja. Penentuan pasangan kendaraan-konsumen ini dilakukan dengan mempertimbangkan kapasitas kendaraan dalam satu kali angkut, untuk meminimalkan biaya yang diperlukan. Biasanya penentuan biaya minimal erat kaitannya dengan jarak yang minimal.

VRP atau *Vehicle Routing Problem* adalah sebuah cakupan masalah yang didalamnya ada sebuah problem dimana ada sejumlah rute untuk sejumlah kendaraan yang berada pada satu atau lebih depot yang harus ditentukan jumlahnya agar tersebar secara geografis supaya bisa melayani konsumen-konsumen yang tersebar. Setiap kendaraan memiliki kapasitas angkut, dan setiap pelanggan memiliki *demand*. Tujuan dari *VRP* adalah mengantarkan barang pada konsumen dengan jarak minimum melalui rute-rute kendaraan yang keluar-masuk depot.

D. Algoritma *Greedy*

Algoritma *Greedy* merupakan algoritma yang lazim untuk memecahkan persoalan optimasi meskipun hasilnya tidak selalu merupakan solusi yang optimum. Sesuai arti harafiah, *Greedy* berarti tamak. Prinsip utama dari algoritma ini adalah mengambil sebanyak mungkin apa yang dapat diperoleh sekarang. Untuk memecahkan

persoalan dengan algoritma *greedy*, kita memerlukan elemen-elemen sebagai berikut.

a. Himpunan Kandidat (C)

   Himpunan ini berisi elemen-elemen pembentuk solusi.

b. Himpunan Solusi, (S)

   Himpunan ini berisi kandidat yang terpilih sebagai solusi persoalan. Dengan kata lain, himpunan solusi adalah himpunan bagian dari himpunan kandidat.

c. Fungsi Seleksi

   Fungsi seleksi merupakan fungsi yang ada pada setiap langkah memilih kandidat yang paling memungkinkan guna mencapai solusi optimal.

d. Fungsi Kelayakan (Feasible)

   Fungsi kelayakan adalah fungsi yang memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak dan tidak melanggar batasan atau constraints yang ada.

e. Fungsi Objektif

   Fungsi objektif adalah fungsi yang memaksimumkan atau meminimumkan nilai solusi.

Skema umum algoritma *greedy* adalah sebagai berikut.

a. Inisialisasi S dengan kosong.

b. Pilih sebuah kandidat C dengan fungsi seleksi.

c. Kurangi C dengan kandidat yang sudah dipilih dari langkah (b) di atas.
d. Periksa apakah kandidat yang dipilih tersebut bersama-sama dengan himpunan solusi membentuk solusi yang layak atau feasible (dengan fungsi kelayakan).
e. Periksa apakah himpunan solusi sudah memberikan solusi yang lengkap serta optimal (dengan fungsi objektif).

Contoh permasalahan yang dapat diselesaikan dengan algoritma *greedy* adalah masalah penukaran uang koin. Strategi *greedy* adalah pada setiap langkah, pilihlah koin dengan nilai terbesar dari himpunan koin yang tersisa.

- Misal: A = 32, koin yang tersedia: 1, 5, 10, dan 25.
- Himpunan Kandidat: himpunan koin yang merepresentasikan nilai 1, 5, 10, 25, paling sedikit mengandung satu koin untuk setiap nilai.
- Himpunan Solusi: total nilai koin yang dipilih tepat sama jumlahnya dengan nilai uang yang ditukarkan
- Fungsi Seleksi: pilihlah koin yang bernilai tertinggi dari himpunan kandidat yang tersisa.
- Fungsi Layak: memeriksa apakah nilai total dari himpunan koin yang dipilih tidak melebihi jumlah uang yang harus dibayar.
- Fungsi Objektif: jumlah koin yang digunakan minimum.
- Langkah 1: pilih 1 buah koin 25 (Total = 25)

Langkah 2: pilih 1 buah koin 5 (Total = 25 + 5 = 30)

Langkah 3: pilih 2 buah koin 1 (Total = 25+5+1+1= 32)

Solusi: Jumlah koin minimum = 4 (solusi optimal!)

Optimum global dengan menggunakan algoritma *greedy* belum tentu merupakan solusi optimum (terbaik), tetapi sub-optimum atau pseudo-optimum. Alasannya adalah sebagai berikut.

1. Algoritma *greed*y tidak beroperasi secara menyeluruh terhadap semua alternatif solusi yang ada (sebagaimana pada metode Exhaustive Search).
2. Terdapat beberapa fungsi seleksi yang berbeda, sehingga harus dipilih fungsi yang tepat agar algoritma yang digunakan menghasilkan solusi optimal.

   Secara umum rumus dari algoritma *greedy* adalah:

$$Berat\ barang\ total = \sum_{j \in N} W_j X_j \leq K \text{ di mana } X_j \in \{0,1\}$$

$$Volume\ barang\ total = \sum_{j \in N} V_j X_j \leq K \text{ di mana } X_j \in \{0,1\}$$

$$Keuntungan\ total = \sum_{j \in N} P_j X_j \text{ di mana } X_j \in \{0,1\}$$

Wj adalah berat barang, Vj adalah volume barang, Pj adalah keuntungan barang, Xj adalah bernilai 0 jika barang tersebut tidak dipilih untuk dimasukkan ke dalam kapasitas Knapsack dan 1 jika barang tersebut terpilih untuk dimasukkan ke dalam kapasitas Knapsack, dan K adalah kapasitas media pengiriman.

Pada penyelesaian masalah Knapsack dengan menggunakan algoritma *greedy* dapat dipilih 3 cara sebagai berikut.

1. *Greedy* by Profit, pilih benda-benda dengan keuntungan maksimum dan benda-benda tersebut memiliki berat yang masih dapat ditampung oleh sisa kapasitas Knapsack.
2. *Greedy* by Weight, pilih benda-benda dengan berat minimum dan benda-benda tersebut memiliki volume yang masih dapat ditampung oleh sisa kapasitas Knapsack.
3. *Greedy* by Volume, pilih benda-benda dengan volume minimum dan benda-benda tersebut memiliki berat yang masih dapat ditampung oleh sisa kapasitas Knapsack.
4. *Greedy* by Weight Density, pilih benda-benda dengan keuntungan per berat yang nilainya maksimum dan benda-benda tersebut memiliki berat dan volume yang masih dapat ditampung oleh sisa kapasitas Knapsack. Rumus untuk mendapatkan density adalah:

$$D_j = \frac{P_j}{W_j}$$

5. *Greedy* by Volume *Density*, pilih benda-benda dengan keuntungan per volume yang nilainya maksimum dan benda-benda tersebut memiliki berat dan volume yang masih dapat ditampung oleh sisa kapasitas Knapsack. Rumus untuk mendapatkan density adalah:

$$D_j = \frac{P_j}{V_j}$$

Pada sebagian masalah, algoritma g*reedy* tidak selalu berhasil memberikan solusi yang optimal. Jika jawaban terbaik mutlak tidak diperlukan, maka algoritma *greedy* sering berguna untuk menghasilkan solusi hampiran (approximation), dibandingkan dengan menggunakan algoritma yang lebih rumit untuk menghasilkan solusi yang eksak. Bila algoritma *greedy* optimum, maka keoptimalannya itu dapat dibuktikan secara matematis.

E. REST

Representational State Transfer atau disingkat REST adalah suatu arsitektur dalam web service untuk melakukan pertukaran data yang berorientasi pada sumber daya Mobile atau resource melalui protokol HTTP. Berikut penjelasan Roy Fielding mengenai Representational State Transfer: “Representational State Transfer is intended to evoke an image of how a well-designed Web Application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing the next state of the application) being transferred to the user and rendered for their use”. REST secara spesifik merujuk pada suatu prinsip-prinsip arsitektur jaringan yang menggariskan pendefinisian dan pengalamatan sumber daya. Istilah ini sering digunakan untuk mendeskripsikan semua antarmuka sederhana yang menyampaikan data dalam domain spesifik melalui HTTP tanpa tambahan lapisan pesan seperti SOAP atau pelacakan sesi menggunakan cookie HTTP .

F. Google Direction Web Service

Google Directions Service adalah Web service yang menghitung arah antara lokasi menggunakan permintaan REST. Arah dapat menentukan asal-usul, tujuan dan waypoints baik sebagai string teks (misalnya "Chicago, IL") atau sebagai koordinat lintang / bujur. Layanan Google Directions Service dapat memberikan respon beberapa titik arah.

G. Roadmap Penelitian

Guna mendukung langkah-langkah penelitian yang akan dilakukan, berikut ini dipaparkan secara ringkas tentang beberapa penelitian yang telah dilakukan oleh peneliti lain khususnya yang membahas tentang masalah optimasi pengangkutan sampah. Kami menemukan beberapa hasil penelitian yang membantu kami dalam mempertajam pembahasan dan rencana penelitian yang akan kami lakukan, diantaranya :

1. Danang Triwibowo (2015), Aplikasi Model Optimasi untuk Meningkatkan Efisiensi Pengangkutan Sampah di Kota Cilegon, mengestimasi inefisiensi pengangkutan sampah di Kota Cilegon. Inefisiensi dihitung dengan membandingkan biaya saat ini dengan biaya hasil optimasi rute dengan model vehicle routing problem.
2. Dea Widya Hutami (2017) Implementasi Algoritma *Nearest Insertion Heuristic* dan *Modified Nearest Insertion Heuristic* Pada Optimasi Rute Kendaraan Pengangkut Sampah (Studi Kasus: Dinas Kebersihan dan

Pertamanan Kota Malang), *Metode Nearest Insertion Heuristic* lebih rumit daripada metode *Modified Nearest Insertion Heuristic*, akan tetapi metode *Nearest Insertion Heuristic* menghasilkan rute yang lebih pendek karena jarak total yang dihasilkan jauh lebih singkat begitupula dengan waktu yang dibutuhkan

3. Arna Fariza (2014), Optimasi Penjadwalan Pengangkutan Sampah Di Surabaya Secara Adaptif Menggunakan Metode Algoritma Genetika, algoritma genetika menghasilkan solusi optimal untuk menyelesaikan permasalahan dalam pencarian rute pengangkutan sampah
4. Himmawati Puji Lestari (2013), Penerapan Algoritma Koloni Semut untuk Optimisasi Rute Distribusi Pengangkutan Sampah Di Kota Yogyakarta, Menggunakan algoritma koloni semut, pengambilan sampah oleh Badan Lingkungan Hidup Kota Yogyakarta menjadi lebih efektif, maka keluhan masyarakat akan menumpuknya sampah dapat diminimalisir.
5. M. Rasyid Ridha (2016), Studi Optimasi Rute Pengangkutan Sampah Kota Marabahan Dengan Sistem Informasi Geografis, Hasil analisis menunjukkan bahwa pengangkutan sampah eksisting menggunakan pola Stationary Container System (SCS) dapat menghemat biaya operasional pengangkutan sampah.

Dari beberapa penelitian diatas, menitik beratkan pada bagaimana proses optimasi pengangkutan sampah dilakukan dengan metode gabungan. Dari road map diatas, penulis ingin membuat suatu model optimalisasi pengangkutan sampah yang dapat menghitung efisiensi waktu pengangkutan sampah serta jalur rute pengangkutan sampah dengan memanfaatkan algoritma *greedy*.

H. Pengujian Perangkat Lunak

Untuk menguji program aplikasi yang dirancang, penulis menggunakan metode *white box*. *White box* testing adalah pengujian yang didasarkan pada pengecekan terhadap detail perancangan, menggunakan struktur kontrol dari desain program secara procedural untuk membagi pengujian ke dalam beberapa kasus pengujian. Secara sekilas dapat diambil kesimpulan *white box* testing merupakan petunjuk untuk mendapatkan program yang benar secara 100%.

a. Pengujian *white box* :

   1. Untuk mengetahui cara kerja suatu perangkat lunak secara internal.
   2. Untuk menjamin operasi-operasi internal sesuai dengan spesifikasi yang telah ditetapkan dengan menggunakan struktur kendali dari prosedur yang dirancang.

b. Pelaksanaan pengujian *white box* :

1. Menjamim seluruh independent path dieksekusi paling sedikit satu kali. Independent path adalah jalur dalam program yang menunjukkan paling sedikit satu kumpulan proses ataupun kondisi baru.
2. Menjalani logical decision pada sisi dan false.
3. Mengeksekusi pengulangan (*looping*) dalam batas-batas yang ditentukan.
4. Menguji struktur data internal.

Berdasarkan konsep pengujian *white box* (*structural*) testing atau *glass box* testing yaitu memeriksa kalkulasi internal path untuk mengidentifikasi kesalahan.

1. Langkah-langkah white box:
    a. Mendefinisikan semua alur logika
    b. Membangun kasus untuk digunakan dalam pengujian
    c. Melakukan pengujian.
2. Kelebihan White Box Testing
    a. Kesalahan logika. Digunakan pada sintaks ‘if’ dan pengulangan. Dimana White Box Testing akan mendeteksi kondisi-kondisi yang tidak sesuai dan mendeteksi kapan proses pengulangan akan berhenti.
    b. Ketidaksesuaian asumsi. Menampilkan asumsi yang tidak sesuai dengan kenyataan, untuk di analisa dan diperbaiki.

c. Kesalahan ketik. Mendeteksi bahasa pemrograman yang bersifat case sensitive.

3. Kelemahan White Box Testing
    a. Untuk perangkat lunak yang tergolong besar, White Box Testing dianggap sebagai strategi yang tergolong boros, karena akan melibatkan sumber daya yang besar untuk melakukannya.
4. Jenis white box:
    a. Basis path: Metode identifikasi yang berdasarkan pada jalur,, struktur atau koneksi yang ada dari suatu sistem ini biasa disebut juga sebagai branch testing,, karena cabang-cabang dari kode atau fungsi logika diidentifikasi dan dites, atau disebut juga sebagai control-flow testing.
    b. Cyclomatic Complexity: Adalah pengukuran software yang memberikan pengukuran kuantitatif dari kompleksitas logika program. Pada konteks metode basis path testing , nilai yang dihitung bagi cyclomatic complexity menentukan jumlah jalur-jalur yang independen dalam kumpulan basis suatu program dan memberikan jumlah tes minimal yang harus dilakukan untuk memastikan bahwa semua pernyataan telah dieksekusi sekurangnya satu kali. Jalur independen adalah tiap jalur pada program yang memperlihatkan 1 kelompok baru dari pernyataan proses atau kondisi baru.

c. Graph Matrix: Adalah matrik berbentuk segi empat sama sisi, dimana jumlah baris dan kolom sama dengan jumlah node, dan identifikasi baris dan kolom sama dengan identifikasi node, serta isi data adalah keberadaan penghubung antar node (edges). Beberapa properti yang dapat ditambahkan sebagai pembobotan pada koneksi antar node di dalam graph matrix, sebagai berikut:
    a. Kemungkinan jalur (Edge) akan dilalui / dieksekusi.
    b. Waktu proses yang diharapkan pada jalur selama proses transfer dilakukan.
    c. Sumber daya (resources) yang dibutuhkan selama proses transfer dilakukan pada jalur.

5. Control Structur Testing, meliputi :
    a. Condition testing : Suatu metode disain test case yang memeriksa kondisi logika yang terdapat pada modul program.
    b. Data flow testing : Metode data flow testing memilih jalur program berdasarkan pada lokasi dari definisi dan penggunaan variabel-variabel pada program.
    c. Loop testing : suatu teknik *white box* testing yang berfokus pada validitas konstruksi loop secara eksklusif. Ada 4 kelas dari loop, yaitu :
        1. Simple Loops
        2. Nested Loops

3. Concatenated Loops
4. Unstructured Loops

5. Contoh kasus : Imperial Taxi Services (ITS) :

V(G) = R = 6

V(G) = E – N + 2 = 21 – 17 + 2 = 6

V(G) = P + 1 = 5 + 1 = 6

Rumus : V(G) = R = E – N + 2 = P + 1 ……(1).

Keterangan :

V(G) = cyclometic complexity graph

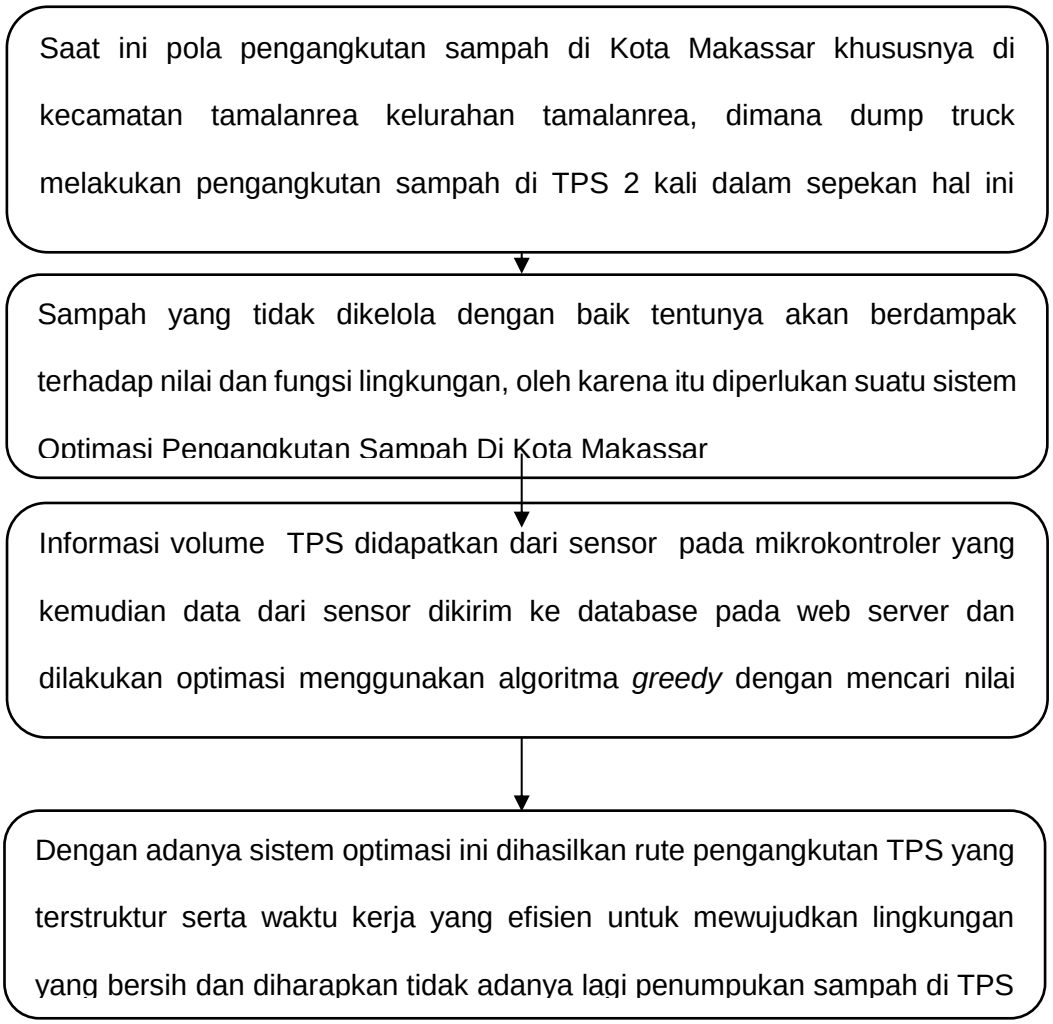
R = jumlah region dalam program flow graph

E = jumlah edge

N = jumlah node

P = jumlah decision (percabangan)

I. Kerangka Konseptual

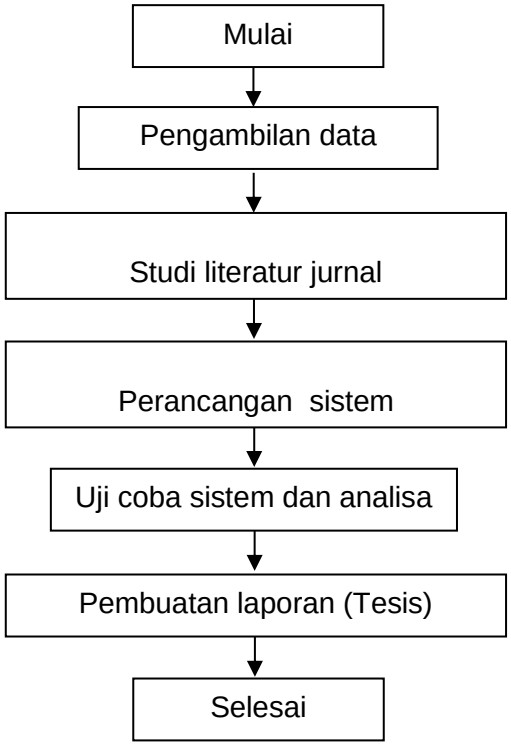


BAB III

METODOLOGI PENELITIAN

A. Tahapan Penelitian

Tahapan penelitian yang akan dilakukan dimulai dari awal pengerjaan hingga akhir penelitian digambarkan secara umum dalam alur berikut ini:



Penelitian ini dimulai dengan melakukan pengambilan data di kelurahan yaitu data masyarakat yang berlangganan untuk dilakukan pengangkutan sampah. Serta membuat *script code* untuk pengambilan data informasi volume TPS dari server *thingspeak* dan menyimpan informasi volume TPS ke dalam database *web server*.

Setelah itu dilakukan studi literatur mengenai metode optimasi volume TPS dengan melihat beberapa jurnal optimasi dan forum-forum diskusi mengenai optimasi. Studi literatur ini akan menjadi referensi pada tahapan perancangan sistem. Perancangan sistem bertujuan untuk membangun suatu aplikasi yang dapat membantu program pemerintah untuk mewujudkan lingkungan yang bersih serta memberikan kepuasan pelayanan kepada masyarakat di Kota Makassar.

Segala teori, metodologi penelitian, uji coba dan analisa dari hasil penelitian yang diperoleh dari tahapan sebelumnya akan dituangkan dalam bentuk laporan, dalam hal ini naskah tesis.

B. Jenis Dan Lokasi Penelitian

Dalam melakukan penelitian ini, jenis penelitian kualitatif yang digunakan adalah Survei, Design and Creation. Dipilihnya jenis penelitian ini oleh penulis dikarenakan konsep dari Survei, Design and Creation sangat tepat untuk mengelola penelitian ini. Adapun lokasi penelitian ini dilakukan di kelurahan tamalanrea Kota Makassar.

C. Analisis Sistem Lama

Sistem yang berjalan saat ini dimana pola pengangkutan sampah di Kota Makassar sudah terjadwal yang ditetapkan oleh pemerintah Kota Makassar khususnya di kecamatan tamalanrea kelurahan tamalanrea, dimana dump truck mengangkut sampah di setiap rumah 2 kali dalam seminggu hal ini menyebabkan seringnya terjadi penumpukan sampah

pada setiap rumah karena waktu pengangkutan yang tidak efisien serta rute yang harus dilaluli oleh truk pengangkut sampah tidak terstruktur sehingga memerlukan waktu yang lama dalam proses pengangkutan sampah.

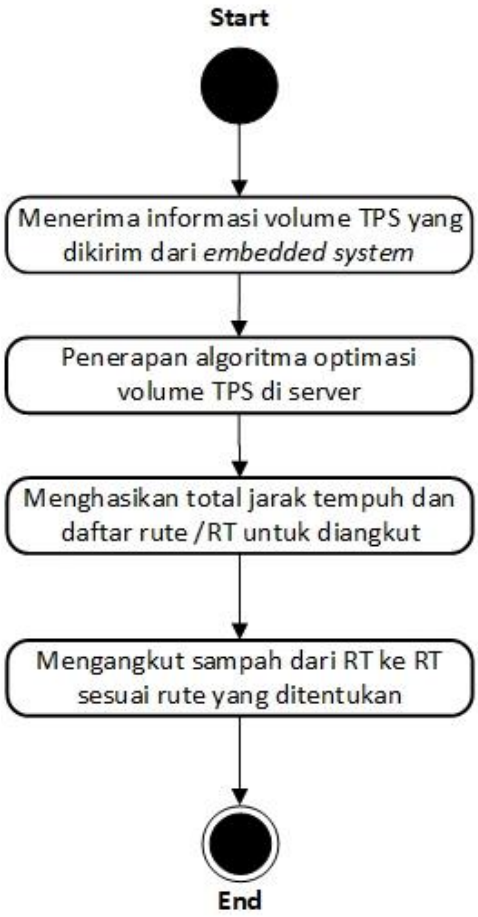
Gambar 3.1 Analisis Sistem Lama

D. Sistem Yang Diusulkan

Adapun sistem yang diusulkan yaitu menerima informasi volume TPS yang dikirim dari mikrokontroler ke server *thingkspeak* dan menyimpan informasi data volume TPS di database *web server* kemudian diterapkan algoritma *greedy* untuk mencari rute terbaik pengangkutan sampah dengan nilai *fitness maximum* di dapatkan dari hasil optimasi total volume TPS

dengan jarak terpendek yang menghasilkan rute angkutan yang dapat diakses oleh operator truk sebagai panduan rute kunjungan pengangkutan TPS dari titik awal ke setiap RT.

Dengan aplikasi ini diharapkan tidak adanya lagi penumpukan sampah dalam waktu lama di setiap TPS, serta membantu pemerintah untuk mewujudkan lingkungan yang bersih dan memberikan kepuasan pelayanan kepada masyarakat di Kota Makassar.



Gambar 3.2 Analisis Sistem Baru

E. Metode Pengumpulan Data

Adapun metode pengumpulan data yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Observasi

   Observasi yang dilakukan yaitu mengamati secara langsung kondisi tempat sampah dan penanganannya di Kota Makassar.

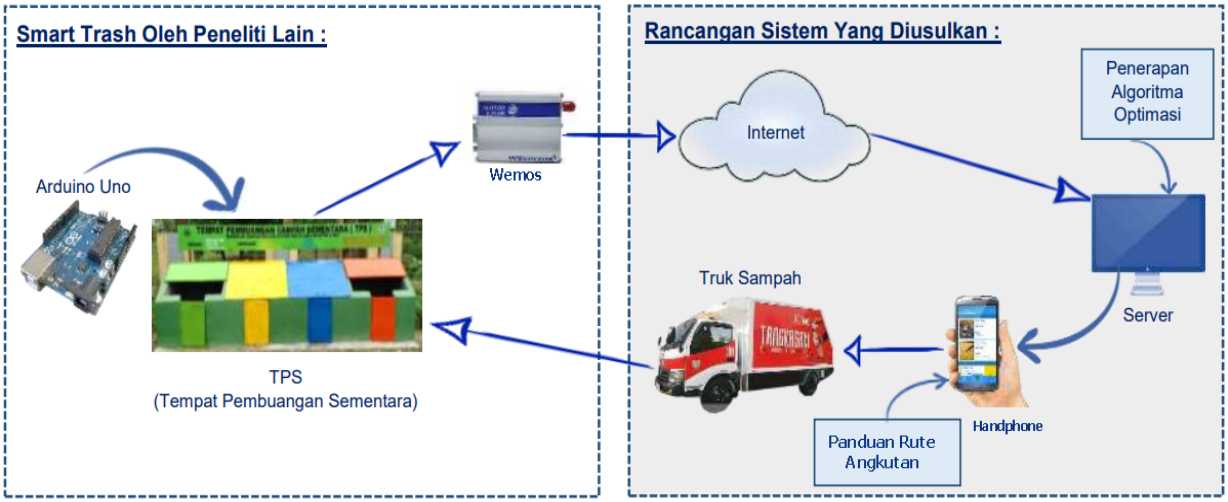
2. Studi Literatur

   Pengumpulan data dengan cara mengumpulkan literatur, jurnal, paper dan bacaan-bacaan yang ada kaitannya dengan judul penelitian.

3. Wawancara

   Teknik wawancara adalah teknik pengumpulan data dengan cara mengajukan beberapa pertanyaan kepada subyek yang berhubungan dengan objek penelitian, data, hal ini melakukan wawancara dengan masyarakat, sopir dump truck dan dinas kebersihan yang bersangkutan.

F. Gambaran Umum Sistem



Gambar 3.3 Gambaran Umum Sistem

Berdasarkan gambar ilustrasi diatas, berikut penjelasan rancangan sistem yang diusulkan :

1. *Internet*, berfungsi menerima informasi volume TPS yang dikirim dari wemos pada mikrokontroler yang terpasang di setiap TPS.
2. *Server*, berfungsi untuk menyimpan informasi data volume TPS ke dalam database pada web server kemudian menerapkan algoritma *greedy* untuk mencari rute terbaik pengangkutan sampah dengan nilai fitness maximum di dapatkan dari hasil optimasi total volume TPS dengan jarak terpendek.
3. *Handphone*, berfungsi melihat panduan rute kunjungan pengangkutan ke setiap RT, yang dapat diakses melalui *handphone* sopir truk pengangkutan sampah.
4. Truk sampah, bertugas mengangkut sampah di setiap RT sesuai urutan rute kunjungan yang dihasilkan oleh sistem.

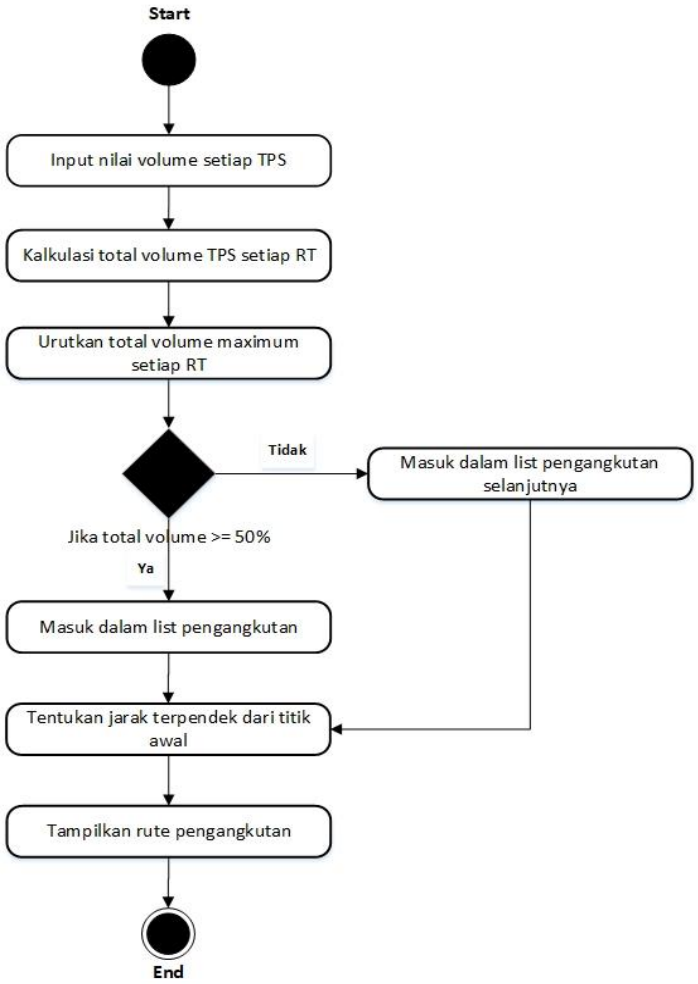
G. Deskripsi Sistem

Berdasarkan gambar ilustrasi diatas, pada bagian server akan menerima informasi volume TPS yang dikirim dari wemos ke server *thingspeak* kemudian menyimpan informasi volume TPS ke dalam database *web server*. Lalu diterapkan algoritma *greedy* untuk mencari rute terbaik pengangkutan sampah dengan nilai *fitness maximum* di dapatkan dari hasil optimasi total volume TPS dengan jarak terpendek, yang menghasilkan rute kunjungan untuk sopir truk angkutan sampah.

Setiap hari pukul 07.00 pagi akan dilakukan optimasi rute dengan total jarak tempuh terpendek yang di akumulasikan dengan volume TPS pada setiap RT, dengan rute kunjungan truk sampah dari titik awal ke setiap titik RT dihasilkan dari nilai *fitness maximum* hasil optimasi dari total volume TPS setiap RT dengan jarak tependek.

H. Rancangan Algoritma *Greedy*

Algoritma greedy membentuk solusi langkah per langkah (step by step). Terdapat banyak pilihan yang perlu dieksplorasi pada setiap langkah solusi. Oleh karena itu, pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan. Keputusan yang telah diambil pada suatu langkah tidak dapat diubah lagi pada langkah selanjutnya.

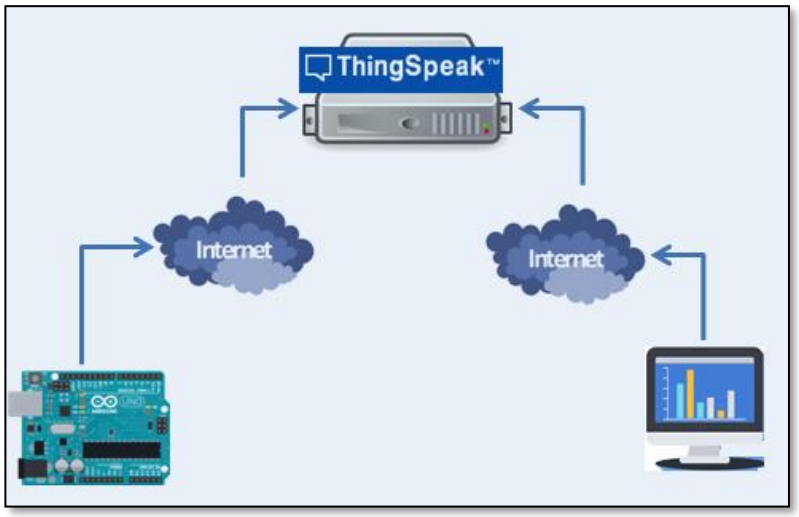


Gambar 3.4 Flowchart Algoritma *Greedy*

Dari flowchart algoritma greedy diatas dimana total volume setiap TPS di kalkulasikan dalam setiap RT kemudian diurutkan total volume *maximum* setiap RT, jika total volume lebih besar atau sama dengan 50% maka masuk dalam list pengangkutan sebaliknya jika belum memenuhi maka masuk dalam list pengangkutan selanjutnya.

Hasil dari algoritma greedy yaitu menampilkan rute terbaik pengangkutan sampah dengan nilai *fitness maximum* di dapatkan dari hasil optimasi total volume TPS dengan jarak terpendek, yang menghasilkan rute kunjungan untuk sopir truk angkutan sampah.

I. Rancangan *Get* Data Dari *Server Thingspeak*



Gambar 3.5 Rancangan *Get* Data Dari *Server Thingspeak*

Pada gambar diatas *wemos* yang terpasang di TPS mengirim data melalui jaringan internet yang kemudian diteruskan ke server *thingspeak,* pada web server dibuat script untuk *get* data dari server *thingspeak* ke server web untuk disimpan ke database server kemudian ditampilkan di halaman monitoring pada web server

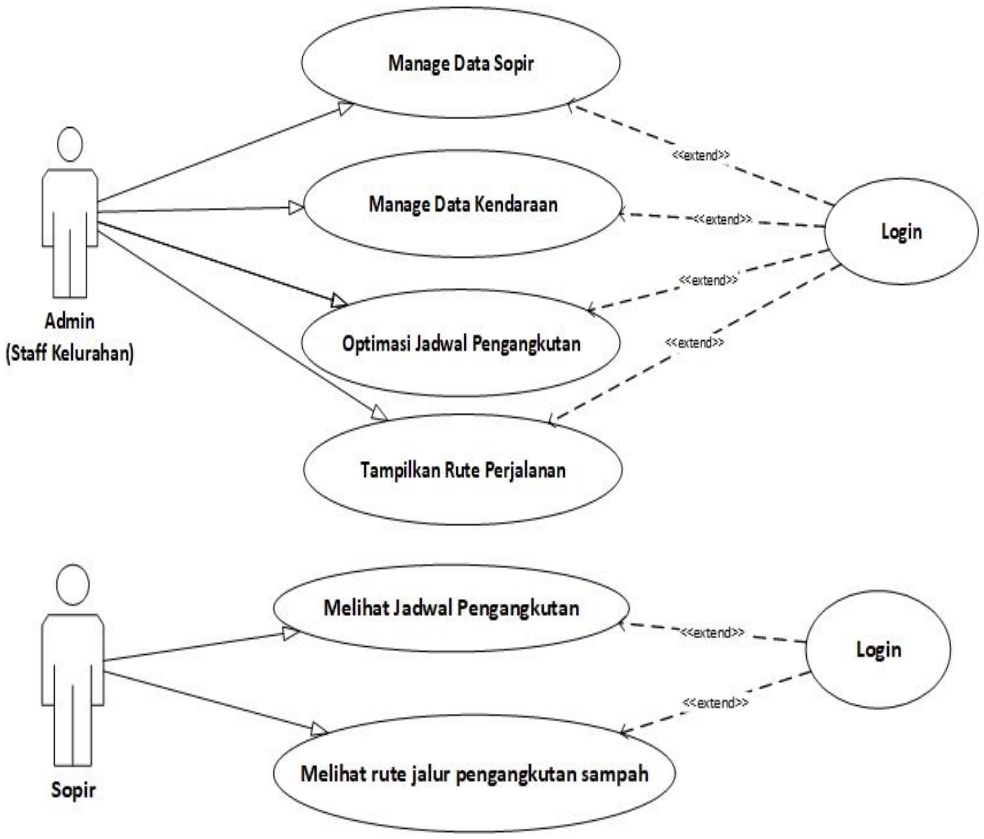
J. Perancangan Sistem

Dengan menggunakan UML (Use Case, Class Diagram, Activity Diagram dan Sequence Diagram) untuk menggambarkan kinerja dari proses yang terjadi pada aplikasi ini. Use Case digunakan untuk menggambarkan menu yang disediakan oleh aplikasi dan menunjukkan actor / pengguna yang diperbolehkan menggunakan aplikasi tersebut. Class Diagram digunakan untuk menggambarkan secara umum script yang berupa fungsi yang disediakan oleh program.

Activity diagram digunakan untuk menggambarkan input yang akan diberikan oleh actor ke aplikasi dan umpan balik yang akan diberikan oleh sistem kepada actor. Sedangkan Sequence diagram digunakan untuk menggambarkan bagaimana kinerja aplikasi yang terjadi didalam sistem sehingga menghasilkan output. Berikut ini merupakan diagram yang akan menggambarkan alur program sesuai dengan fungsinya masing-masing.

1. Use Case Diagram

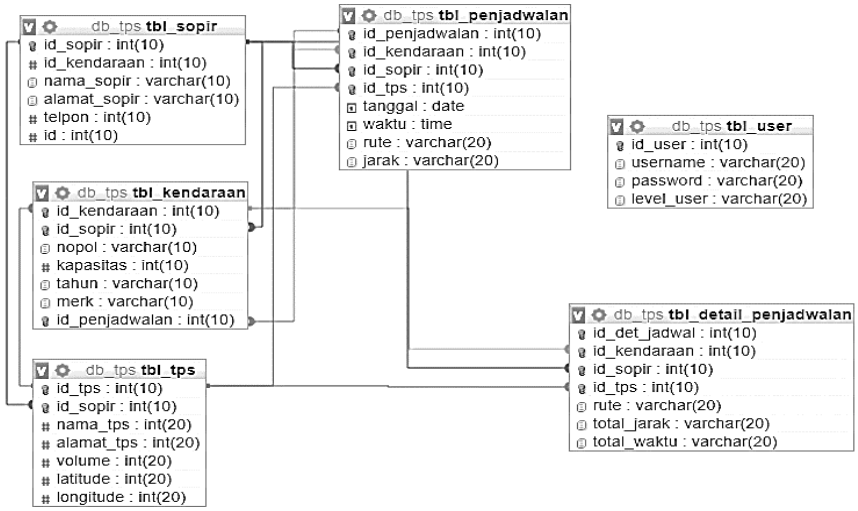
Berdasarkan penentuan pengguna sebagai actor peneliti menempatkan 2 aktor yaitu admin dalam hal ini staff kelurahan yang mempunyai hak akses penuh untuk manage data di website serta sopir yang akan menggunakan aplikasi ini. Admin akan menggunakan aplikasi berbasis web sedangkan sopir akan menggunakan aplikasi untuk melihat rute perjalanan pengangkutan TPS di setiap RT.



Gambar 3.6 Use Case Diagram

2. Class Diagram

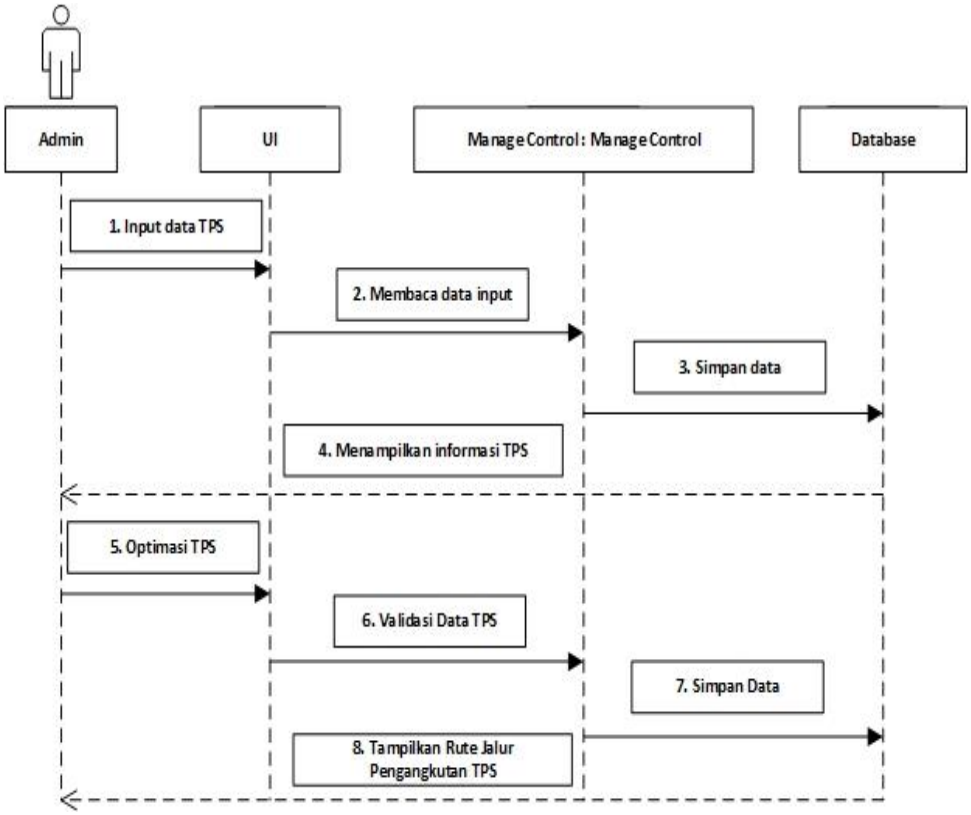
Pada tahap ini akan di identifikasikan kelas-kelas yang akan dijadikan media komunikasi antara aktor dengan sistem. Kelas Interface yang di identifikasikan sesuai dengan kebutuhan Aktor terhadap sistem.



Gambar 3.7 Class Diagram

3. Sequence Diagram

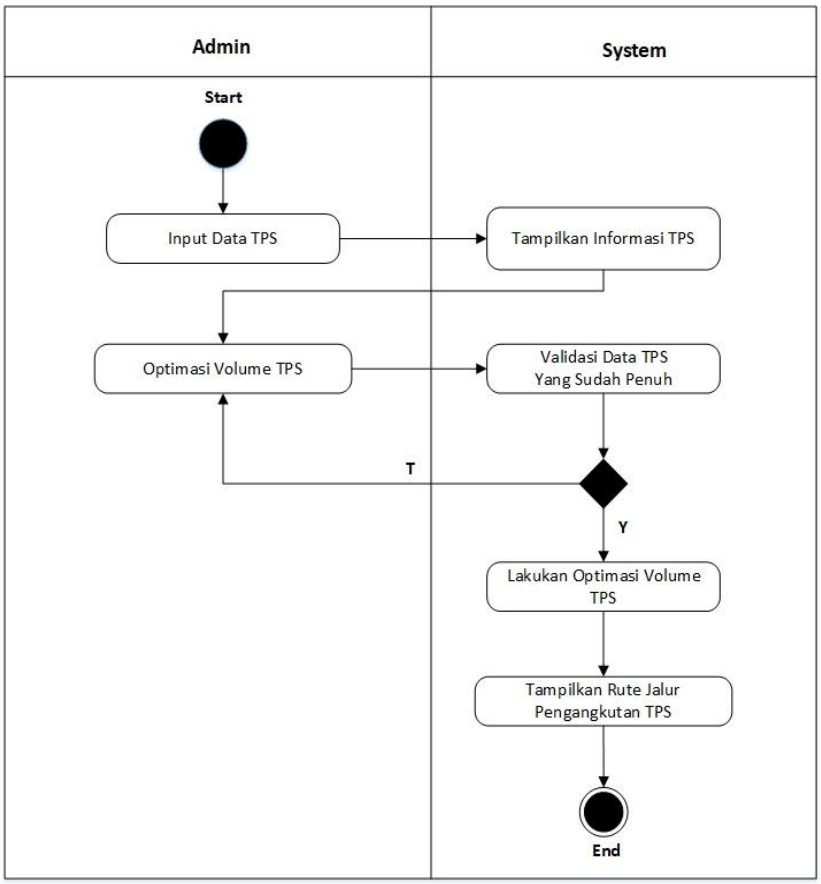
Pada sequence diagram dijelaskan proses penyimpanan data TPS kedalam database yang kemudian dilakukan validasi optimasi TPS dan menampilkan rute jalur pengangkutan TPS di setiap RT.



Gambar 3.8 Sequence Diagram

4. Activity Diagram

Pada activity diagram dijelaskan proses penyimpanan data TPS kedalam database yang kemudian dilakukan validasi optimasi TPS dan menampilkan rute jalur pengangkutan TPS.



Gambar 3.9 Activity Diagram

K. Instrumen Penelitian

Instrument penelitian yang digunakan dalam penelitan yaitu :

1. Perangkat Keras

Perangkat keras yang digunakan untuk mengembangkan dan mengumpulkan data pada aplikasi ini adalah sebagai berikut:

a. Laptop ASUS K46CB dengan spesifikasi :

1) Prosesor Intel® Core™ i3 (1.80 Ghz)

2) Display 14” WXGA LED, Max. Resolution 1366 x 768

3) RAM 8 GB DDR3 Memory 4) Harddisk 500GB

2. Perangkat Lunak

Adapun perangkat lunak yang digunakan dalam aplikasi ini adalah sebagai berikut :

a. Sistem Operasi Windows 10 Enterprise 64-bit
b. Xampp
c. Visual Code
d. Laravel 5.6
e. MySQL
f. Navicat

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

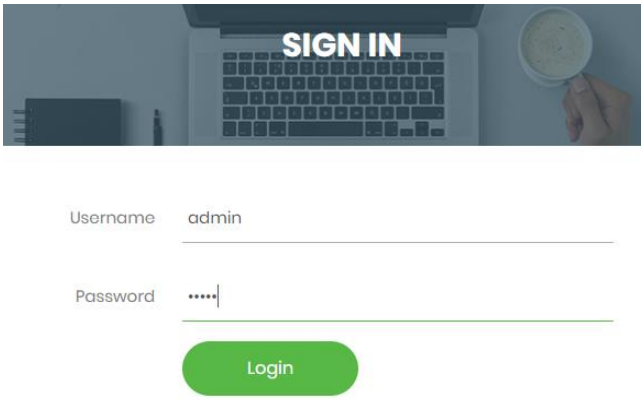
A. Implementasi Sistem

Implementasi merupakan tahap uji coba sistem yang telah dibuat. Implementasi bertujuan untuk mengkaji mengenai rangkaian sistem baik software maupun hardware dan untuk melakukan uji coba mengenai perangkat lunak sistem (*software*) maupun perangkat keras (*hardware*).

Pada implementasi antar muka ini, menjelaskan tentang halaman utama pada aplikasi yang merupakan penghubung dengan sub-sub menu lainnya yang ada pada aplikasi. Berikut adalah penjelasan dari implementasi sistem aplikasi penentuan jenis part of speech berbasis web yang dijelaskan di bawah ini.

1. Halaman Login Admin

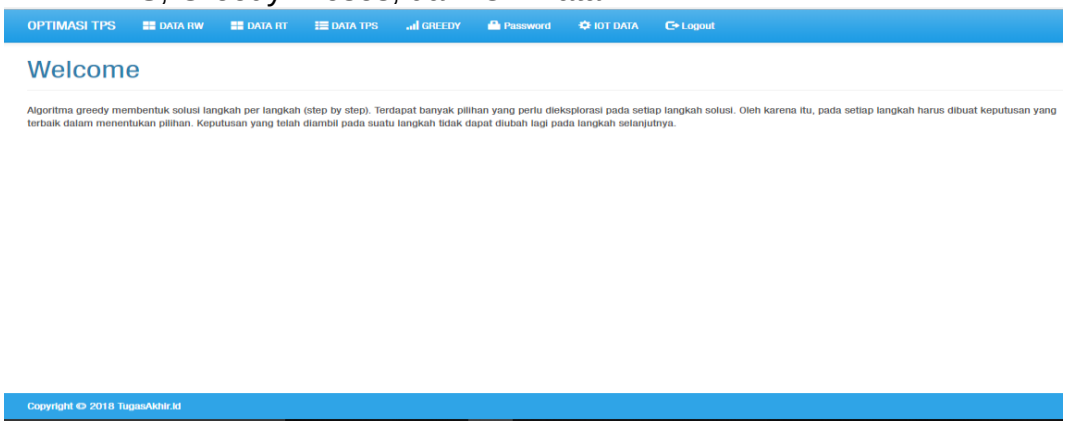
Halaman login berikut ini, pengguna atau admin harus memasukkan username dan password yang sesuai agar dapat mengakses aplikasi.



Gambar 4.1 Halaman login admin

2. Halaman Utama

Halaman menu utama berikut ini berisi informasi mengenai menu-menu dan sub menu apa saja yang tersedia di dalam aplikasi. Menu tersebut antara lain, Dashboard, Data RW, Data RT, Data TPS, Greedy Proses, dan IOT Data.



Gambar 4.2 Halaman Utama

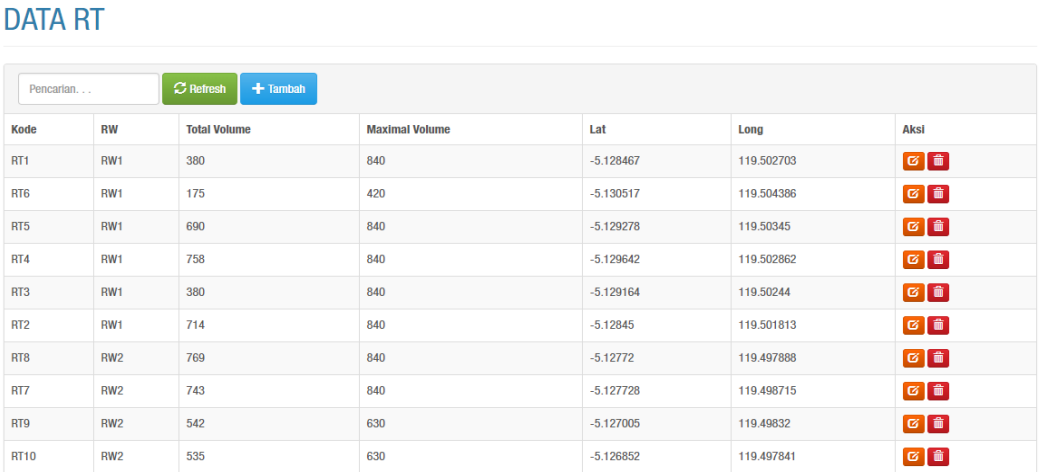
3. Halaman Menage Data RW

Halaman manage data RW sub menu dari menu Master Data yang berfungsi untuk menampilkan data RW yang telah ada sebelumnya. Admin juga dapat melakukan manipulasi data (update dan delete) terhadap data RW tersebut.

Gambar 4.3 Halaman Menage Data RW

4. Halaman Menage Data RT

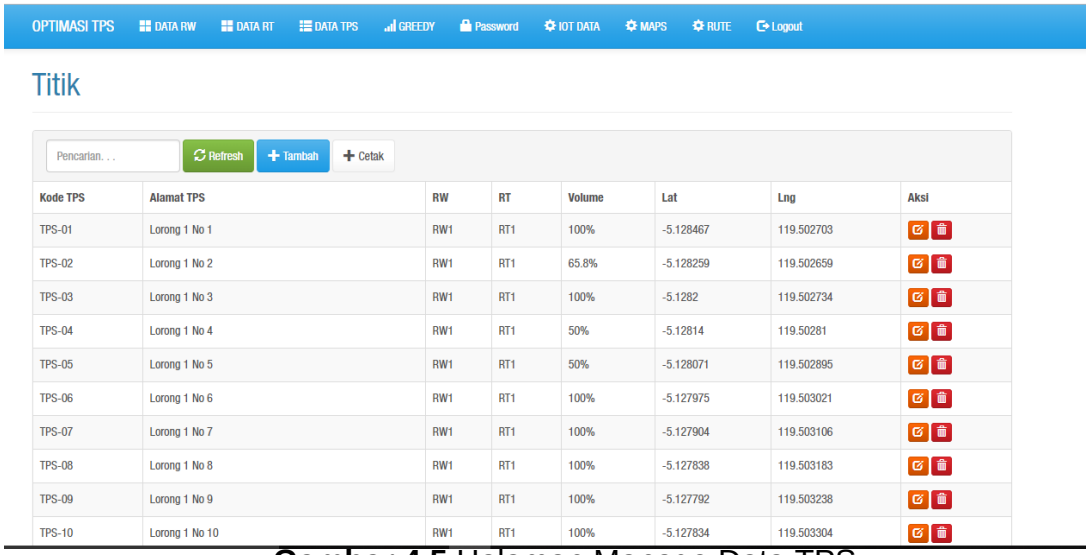
Halaman manage data RT sub menu dari menu Master Data yang berfungsi untuk menampilkan data RT yang telah ada sebelumnya.



Gambar 4.4 Halaman Menage Data RT

5. Halaman Manage Data TPS

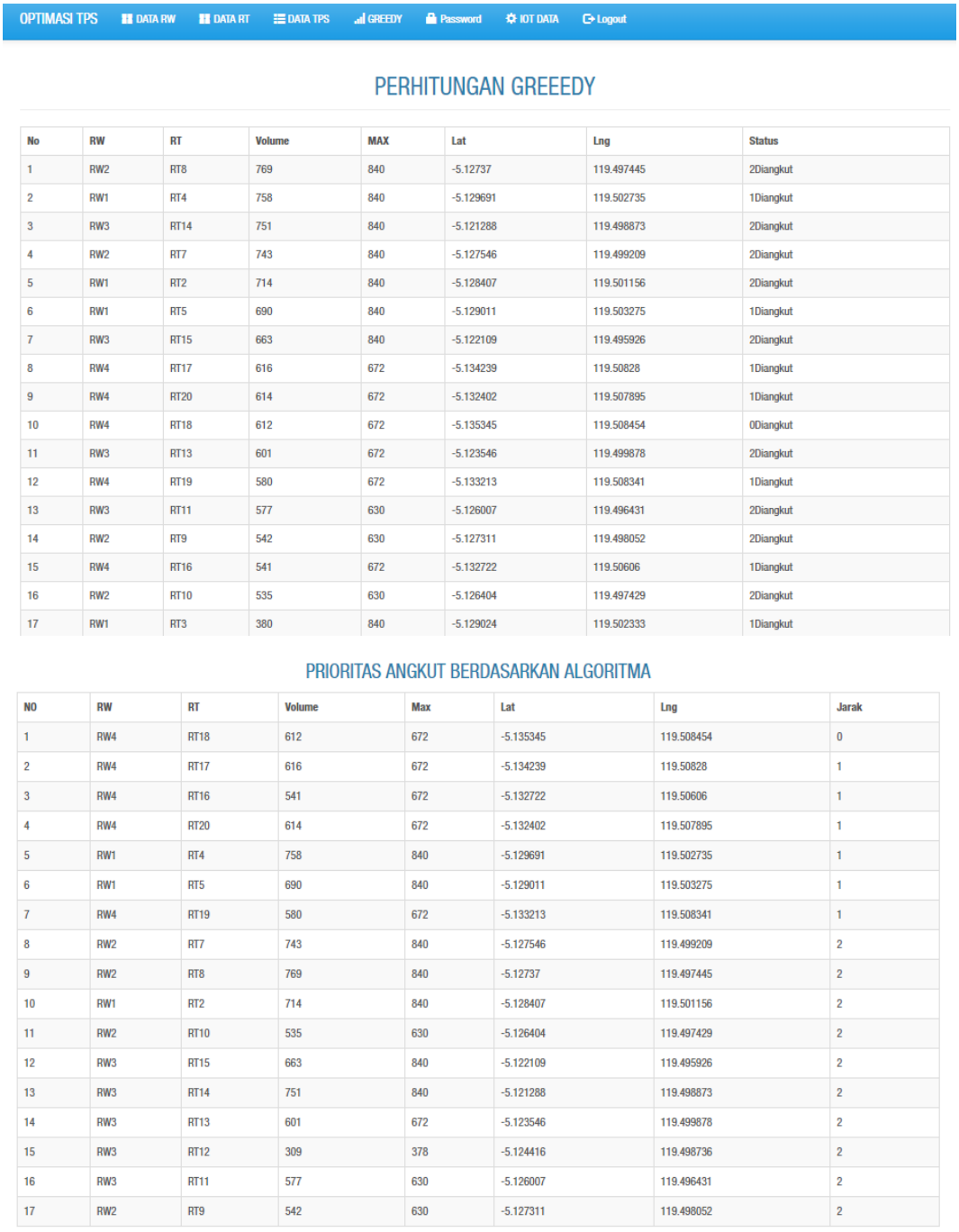
Halaman manage data TPS sub menu dari menu Master Data.



Gambar 4.5 Halaman Manage Data TPS

6. Halaman Perhitungan Greedy

Halaman perhitungan greedy sub menu dari menu Master Data yang berfungsi untuk menampilkan hasil dari perhitungan *greedy.*



No	RW	RT	Volume	MAX	Lat	Lng	Status
1	RW2	RT8	769	840	-5.12737	119.497445	2Diangkut
2	RW1	RT4	758	840	-5.129691	119.502735	1Diangkut
3	RW3	RT14	751	840	-5.121288	119.498873	2Diangkut
4	RW2	RT7	743	840	-5.127546	119.499209	2Diangkut
5	RW1	RT2	714	840	-5.128407	119.501156	2Diangkut
6	RW1	RT5	690	840	-5.129011	119.503275	1Diangkut
7	RW3	RT15	663	840	-5.122109	119.495926	2Diangkut
8	RW4	RT17	616	672	-5.134239	119.50828	1Diangkut
9	RW4	RT20	614	672	-5.132402	119.507895	1Diangkut
10	RW4	RT18	612	672	-5.135345	119.508454	0Diangkut
11	RW3	RT13	601	672	-5.123546	119.499878	2Diangkut
12	RW4	RT19	580	672	-5.133213	119.508341	1Diangkut
13	RW3	RT11	577	630	-5.126007	119.496431	2Diangkut
14	RW2	RT9	542	630	-5.127311	119.498052	2Diangkut
15	RW4	RT16	541	672	-5.132722	119.50606	1Diangkut
16	RW2	RT10	535	630	-5.126404	119.497429	2Diangkut
17	RW1	RT3	380	840	-5.129024	119.502333	1Diangkut

NO	RW	RT	Volume	Max	Lat	Lng	Jarak
1	RW4	RT18	612	672	-5.135345	119.508454	0
2	RW4	RT17	616	672	-5.134239	119.50828	1
3	RW4	RT16	541	672	-5.132722	119.50606	1
4	RW4	RT20	614	672	-5.132402	119.507895	1
5	RW1	RT4	758	840	-5.129691	119.502735	1
6	RW1	RT5	690	840	-5.129011	119.503275	1
7	RW4	RT19	580	672	-5.133213	119.508341	1
8	RW2	RT7	743	840	-5.127546	119.499209	2
9	RW2	RT8	769	840	-5.12737	119.497445	2
10	RW1	RT2	714	840	-5.128407	119.501156	2
11	RW2	RT10	535	630	-5.126404	119.497429	2
12	RW3	RT15	663	840	-5.122109	119.495926	2
13	RW3	RT14	751	840	-5.121288	119.498873	2
14	RW3	RT13	601	672	-5.123546	119.499878	2
15	RW3	RT12	309	378	-5.124416	119.498736	2
16	RW3	RT11	577	630	-5.126007	119.496431	2
17	RW2	RT9	542	630	-5.127311	119.498052	2

Gambar 4.6 Halaman Perhitungan *Greedy*

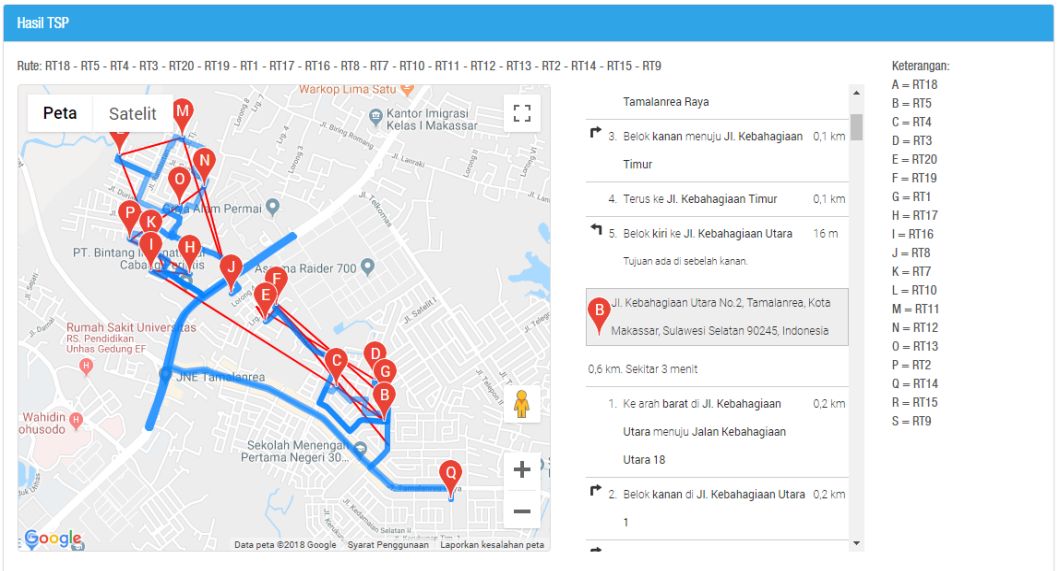
SIMULASI SETELAH DIANGKUT

No	RW	RT	Volume	Max	Lat	Lng	Jarak
1	RW4	RT18	0	672	-5.135345	119.508454	0
2	RW1	RT1	380	840	-5.128311	119.502596	1
3	RW1	RT6	175	420	-5.130494	119.504142	1
4	RW1	RT5	0	840	-5.129011	119.503275	1
5	RW1	RT4	0	840	-5.129691	119.502735	1
6	RW1	RT3	380	840	-5.129024	119.502333	1
7	RW4	RT20	0	672	-5.132402	119.507895	1
8	RW4	RT19	0	672	-5.133213	119.508341	1
9	RW4	RT17	0	672	-5.134239	119.50828	1
10	RW4	RT16	0	672	-5.132722	119.50606	1
11	RW2	RT8	73	840	-5.12737	119.497445	2
12	RW2	RT7	0	840	-5.127546	119.499209	2
13	RW2	RT10	0	630	-5.126404	119.497429	2
14	RW3	RT11	0	630	-5.126007	119.496431	2
15	RW3	RT12	0	378	-5.124416	119.498736	2
16	RW3	RT13	0	672	-5.123546	119.499878	2
17	RW3	RT14	0	840	-5.121288	119.498873	2
18	RW1	RT2	0	840	-5.128407	119.501156	2
19	RW3	RT15	0	840	-5.122109	119.495926	2
20	RW2	RT9	0	630	-5.127311	119.498052	2

Gambar 4.7 Simulasi Setelah Diangkut

7. Halaman Rute Penangkutan

Halaman ini menampilkan rute pengangkutan sampah yang nantinya dapat dilihat oleh sopir sebagai panduan navigasi dalam pengangkutan.



Gambar 4.8 Halaman Rute Pengangkutan

8. **Halaman Monitoring Data *Thingspeak***

Halaman ini menampilkan rute pengangkutan sampah yang nantinya dapat dilihat oleh sopir sebagai panduan navigasi dalam pengangkutan.

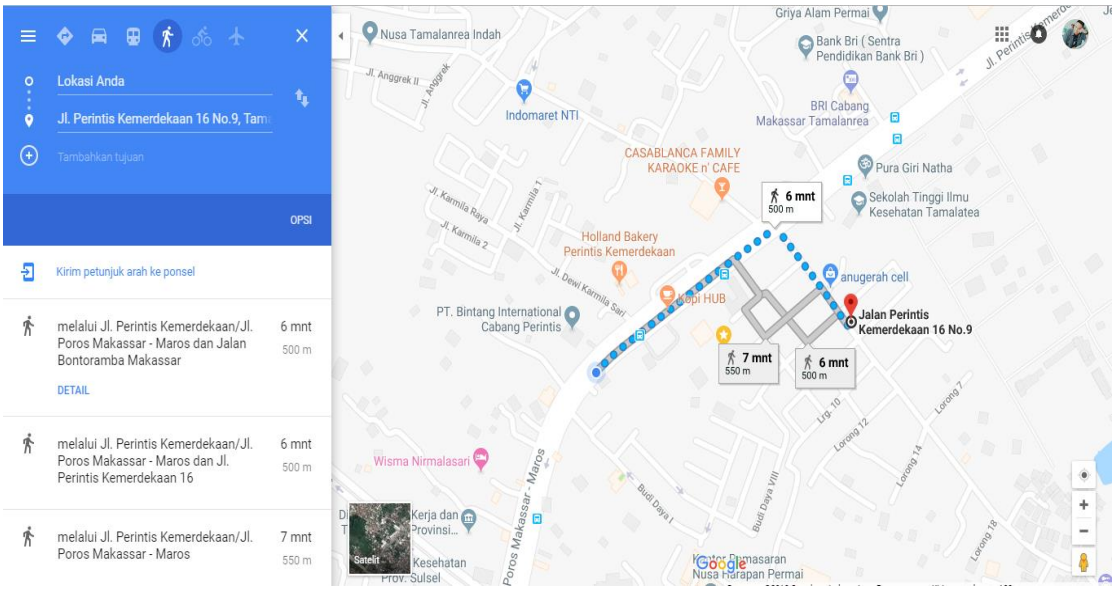
Data Thingspeak

Smart Trash Monitoring

No	Nama TPS	RW	RT	Volume	Lat	Lng
1	TPS 01 / Field 1	RW1	RT1	0	-5.128311	119.502596
2	TPS 02 / Field 2	RW1	RT1	0	-5.128259	119.502659
3	TPS 03 / Server	RW1	RT1	0	-5.1282	119.502734
4	TPS 01 / Field 1	RW1	RT1	0	-5.128311	119.502596
5	TPS 02 / Field 2	RW1	RT1	0	-5.128259	119.502659
6	TPS 03 / Server	RW1	RT1	0	-5.1282	119.502734
7	TPS 01 / Field 1	RW1	RT1	0	-5.128311	119.502596
8	TPS 02 / Field 2	RW1	RT1	0	-5.128259	119.502659
9	TPS 03 / Server	RW1	RT1	0	-5.1282	119.502734
10	TPS 01 / Field 1	RW1	RT1	0	-5.128311	119.502596
11	TPS 02 / Field 2	RW1	RT1	38	-5.128259	119.502659

Gambar 4.9 Halaman Monitoring Data *Thingspeak*

9. **Halaman Rute Pengangkutan Setiap TPS**



Gambar 4.10 Halaman Pengangkutan Setiap TPS

10. Jadwal Rute Truk Sampah

Sopir	Kapasitas Truk	Rute Rencana Truk Pengangkut Sampah	Total Jarak	Total Durasi Waktu	Tanggal
Andi	10 M3	RT.1 -> RT.6 -> RT.5 -> RT.4 -> RT.3 -> RT.18 -> RT.17 -> RT.16 -> RT.15 -> RT.5 -> RT.8 -> RT.7 -> RT.10 -> RT.11 -> RT.12 -> RT.13 -> RT.2 -> RT.14 -> RT.9	18.431 Km	2 Jam 20 Menit	18/09/2018

Sopir	Kapasitas Truk	Rute Rencana Truk Pengangkut Sampah	Total Jarak	Total Durasi Waktu	Tanggal
Andi	10 M3	RT.1 -> RT.6 -> RT.5 -> RT.4 -> RT.3 -> RT.18 -> RT.17 -> RT.16 -> RT.15 -> RT.5 -> RT.8 -> RT.7	10.431 Km	1 Jam 20 Menit	20/09/2018

B. Implementasi Algoritma Greedy

Algoritma greedy membentuk solusi langkah per langkah (step by step) pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan. Keputusan yang telah diambil pada suatu langkah tidak dapat diubah lagi pada langkah selanjutnya.

Hasil dari algoritma greedy yaitu menampilkan rute terbaik pengangkutan sampah dengan nilai *fitness maximum* di dapatkan dari hasil

optimasi total volume TPS dengan jarak terpendek, yang menghasilkan rute kunjungan untuk sopir truk angkutan sampah.

RW I (Bontoramba)					
RT	**Latitude**	**Longitude**	**Total Jarak Dari Titik Awal**	**Jumlah TPS**	**(Jumlah TPS * 38 Liter)**
RT.1	-5.128467	119.502703	1,7 Km	20	760 Liter
RT.2	-5.128450	119.501813	1,8 Km	20	760 Liter
RT.3	-5.129164	119.502440	1,8 Km	20	760 Liter
RT.4	-5.129642	119.502862	1,7 Km	20	760 Liter
RT.5	-5.129278	119.503450	1,6 Km	20	760 Liter
RT.6	-5.130517	119.504386	1,4 Km	10	380 Liter

Tabel 4.1 RW I Bontoramba

RW II (BTN Karmila Sari)					
RT	**Latitude**	**Longitude**	**Total Jarak Dari Titik Awal**	**Jumlah TPS**	**(Jumlah TPS * 38 Liter)**
RT.7	-5.127728	119.498715	2.9 Km	20	760 Liter
RT.8	-5.127720	119.497888	3,1 Km	20	760 Liter
RT.9	-5.127005	119.498320	3,1 Km	15	570 Liter
RT.10	-5.126852	119.497841	3,1 Km	15	570 Liter

Tabel 4.2 RW II Karmila Sari

RW III (Perumahan NTI)					
RT	**Latitude**	**Longitude**	**Total Jarak Dari Titik Awal**	**Jumlah TPS**	**(Jumlah TPS * 38 Liter)**
RT.11	-5.125431	119.496980	3,6 Km	15	570 Liter

RT.12	-5.124335	119.499289	3,6 Km	10	380 Liter
RT.13	-5.123131	119.498035	3,8 Km	15	570 Liter
RT.14	-5.121253	119.498820	4,2 Km	20	760 Liter
RT.15	-5.122744	119.497250	4,0 Km	20	760 Liter

Tabel 4.3 RW III Perumahan NTI

RW IV (BTP Blok A)					
RT	**Latitude**	**Longitude**	**Total Jarak Dari Titik Awal**	**Jumlah TPS**	**(Jumlah TPS * 38 Liter)**
RT.16	-5.133148	119.506587	1,1 Km	16	608 Liter
RT.17	-5.134279	119.507283	850 M	16	608 Liter
RT.18	-5.134610	119.508488	750 M	16	608 Liter
RT.19	-5.133313	119.507831	1.0 Km	16	608 Liter
RT.20	-5.132610	119.508060	1.0 Km	16	608 Liter

Tabel 4.4 RW IV BTP Blok A

Berikut adalah langkah-langkah algoritma *greedy* :

1. **Mencari nilai *maximum* volume setiap TPS yang di kalkulasikan dalam setiap RT.**

   Percobaan Kasus :

RT	Volume	Max
RT.1	657	760
RT.2	728	760
RT.3	525	760
RT.4	725	760
RT.5	675	760

RT.6	296	380
RT.7	708	760
RT.8	731	760
RT.9	526	570
RT.10	516	570
RT.11	549	570
RT.12	334	380
RT.13	541	570
RT.14	724	760
RT.15	654	760
RT.16	528	608
RT.17	582	608
RT.18	580	608
RT.19	300	608
RT.20	294	608
Total Volume = 11.173 Liter		

Tabel 4.5 Tabel Mencari Nilai *Maximum* Volume

2. Urutkan Total Volume Terbesar Setiap RT

Keterangan = * Volume >= 50% = Diangkut

* Volume < 50% = Belum Diangkut

RT	Volume	Max	Status
RT.8	731	760	Diangkut
RT.2	728	760	Diangkut
RT.4	725	760	Diangkut
RT.14	724	760	Diangkut
RT.7	708	760	Diangkut
RT.5	675	760	Diangkut
RT.1	657	760	Diangkut
RT.15	654	760	Diangkut

RT.17	582	608	Diangkut
RT.18	580	608	Diangkut
RT.11	549	570	Diangkut
RT.13	541	570	Diangkut
RT.16	528	608	Diangkut
RT.9	526	570	Diangkut
RT.3	525	760	Diangkut
RT.10	516	570	Diangkut
RT.12	334	380	Diangkut
RT.19	300	608	Belum Diangkut
RT.6	296	380	Diangkut
RT.20	294	608	Belum Diangkut

Tabel 4.6 Tabel Urutan Total Volume

3. Tabel Pengangkutan Sesuai Kapasitas Dump Truk

RT	Total Volume	Sisa Volume	Total Volume Diangkut
RT.8	731	579	152
RT.2	728	0	728
RT.4	725	0	725
RT.14	724	0	724
RT.7	708	0	708
RT.5	675	0	675
RT.1	657	0	657
RT.15	654	0	654
RT.17	582	0	582
RT.18	580	0	580
RT.11	549	0	549
RT.13	541	0	541
RT.16	528	0	528
RT.9	526	0	526
RT.3	525	0	525
RT.10	516	0	516

RT.12	334	0	334
RT.19	300	300	0
RT.6	296	0	296
RT.20	294	294	0
Total	**11.173**	**1.173**	**10.000 Liter**

Tabel 4.7 Tabel Pengangkutan Sesuai Kapasitas Dump Truk

4. Prioritas Angkutan Sesuai Jarak Terpendek Dari Titik Awal

- Rumus Mencari Jarak Terpendek

```
function distHaversine($coord_a, $coord_b){
   $R = 6371;
   $coord_a = explode(",",$coord_a);
   $coord_b = explode(",",$coord_b);
   $dLat = rad(($coord_b[0]) - ($coord_a[0]));
   $dLong = rad($coord_b[1] - $coord_a[1]);
   $a = sin($dLat/2) * sin($dLat/2) + cos(rad($coord_a[0])) *
cos(rad($coord_b[0])) * sin($dLong/2) * sin($dLong/2);
   $c = 2 * atan2(sqrt($a), sqrt(1-$a));
   $d = $R * $c;
   return number_format($d, 0, '.', ',');  }
```

RT	Volume	Jarak Dari Titik Awal
RT.1	657	1,7 Km
RT.6	296	1.4 Km
RT.5	675	1.6 Km
RT.4	725	1.7 Km
RT.3	525	1.8 Km
RT.18	580	750 M
RT.17	582	850 M
RT.16	528	1.1 Km
RT.15	654	4.0 Km
RT.8	731	3.1 Km
RT.7	708	2.9 Km
RT.10	516	3.1 Km
RT.11	549	3.6 Km
RT.12	334	3.6 Km
RT.13	541	3.8 Km

RT.2	728	1.8 Km
RT.14	724	4.2 Km
RT.9	526	3.1 Km

Tabel 4.8 Tabel Prioritas Angkutan Sesuai Jarak Terpendek

Berdasarkan hasil perhitungan algoritma *greedy* optimasi rute terpendek dan volume TPS menghasilkan rute kunjungan untuk sopir truk angkutan sampah pada table dibawah ini :

Rute	RT	Total Jarak Tempuh
Rute -> 1	RT.1	18.332 Km
Rute -> 2	RT.6	
Rute -> 3	RT.5	
Rute -> 4	RT.4	
Rute -> 5	RT.3	
Rute -> 6	RT.18	
Rute -> 7	RT.17	
Rute -> 8	RT.16	
Rute -> 9	RT.15	
Rute -> 10	RT.8	
Rute -> 11	RT.7	
Rute -> 12	RT.10	
Rute -> 13	RT.11	
Rute -> 14	RT.12	
Rute -> 15	RT.13	
Rute -> 16	RT.2	
Rute -> 17	RT.14	
Rute -> 18	RT.9	

Tabel 4.9 Tabel Rute Kunjungan

C. Pengujian Perangkat Lunak

Pengujian dimaksud untuk mengetahui apakah perangkat lunak yang dibuat telah memenuhi tujuan dari perancangan dari perangkat lunak itu

sendiri. Sebelum penerapan sistem, terlebih dahulu harus dipastikan bahwa sistem harus bebas dari kesalahan logika yang mungkin dapat terjadi sehingga dapat sesuai dengan harapan. Metode pengujian program yang dilakukan untuk menguji kesalahan logika dengan menggunakan metode pengujian *white box*. Dimana materi yang diujikan antara lain:

1. Pengujian Optimasi Algoritma *Greedy*
2. Pengujian Rute Kunjungan
3. Pengujian Get Data Dari Server *Thingspeak*

Jika materi yang diuji memperoleh hasil dimana nilai *Cyclomatic complexity (CC), Region* dan *Independent Path* bernilai sama maka di dapatkan kesimpulan bahwa, modul dari materi uji terbebas dari kesalahan logika. Sesuai referensi Rivayi Arifanto (2016).

D. Teknik Pengujian Perangkat Lunak

Teknik atau metode pengujian yang digunakan terhadap perangkat lunak yang telah dibangun adalah metode pengujian *basis path.* Metode ini bertujuan untuk mengukur kekompleksan logika dari perancangan prosedur utama. Untuk menghitung tingkat kompleksitas logika program maka digunakan metode *Cyclomatic complexity (CC).*

Cyclomatic complexity (CC) dapat dihitung dengan menggunakan rumus*:*

CC = E – N + 2

Dimana :

E = Jumlah Edge pada Flowgraph

N = Jumlah Node pada Flowgraph

1. Pengujian Optimasi Algoritma *Greedy*

Node	Source Code
1	 PERHITUNGAN GREEEDY
2	function rad($x){ return $x * M_PI / 180; } function distHaversine($coord_a, $coord_b){ # jarak kilometer dimensi (mean radius) bumi $R = 6371; $coord_a = explode(",",$coord_a); $coord_b = explode(",",$coord_b); $dLat = rad(($coord_b[0]) - ($coord_a[0])); $dLong = rad($coord_b[1] - $coord_a[1]); $a = sin($dLat/2) * sin($dLat/2) + cos(rad($coord_a[0])) * cos(rad($coord_b[0])) * sin($dLong/2) * sin($dLong/2); $c = 2 * atan2(sqrt($a), sqrt(1-$a)); $d = $R * $c; # hasil akhir dalam satuan kilometer return number_format($d, 0, '.', ',');}
3	$db->query("TRUNCATE TABLE tb_titik_greedy"); $db->query("TRUNCATE TABLE tb_titik_greedy_dump"); $rows1 = $db->get_results("SELECT nama_titik, kode_kelompok,id_rt,kode_titik, SUM(volume) as volume,lat,lng

	,(COUNT(kode_titik)*38) as max FROM tb_titik GROUP BY id_rt; ");
4	foreach($rows1 as $row): $kode_kelompok= $row->kode_kelompok; #$kode_titik=$row->kode_titik ; $kode_titik=$row->id_rt ; $volume=$row->volume; $lat=$row->lat; $lng=$row->lng; $max=$row->max; $asal = "-5.13792,119.511767"; $b=$lat.",".$lng; $jarak= distHaversine($asal, $b);
5	$db->query("INSERT INTO tb_titik_greedy (kode_kelompok, kode_titik, nama_titik,volume,max, lat, lng,jarak) VALUES ('$kode_kelompok', '$kode_titik', '$nama_titik', '$volume','$max','$lat', '$lng','$jarak')");

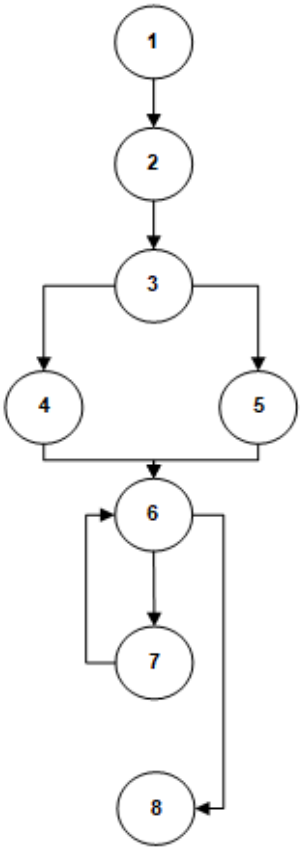
6	$max_batas=$max/2;
	if ($volume>=$max_batas){
	if ($volume<=$batas) {
	$batas-=$volume;
	$volume2=0;
	}else if ($volume>$batas){
	$volume2=$volume-$batas;
	$batas-=$volume2;}}

7	$max_batas=$max/2;
	if ($volume>=$max_batas){
	if ($volume<=$batas) {
	$batas-=$volume;
	$volume2=0;
	}else if ($volume>$batas){
	$volume2=$volume-$batas;
	$batas-=$volume2;}}

8	<tr >
	<td><?=$no++?></td>
	<td><?=$row->kode_kelompok?></td>
	<td><?=$row->kode_titik ?></td>
	<td><?=$row->volume?></td>
	<td><?=$row->max?></td>
	<td><?=$row->lat?></td>
	<td><?=$row->lng?></td>
	<td><?=$row->jarak;
	if ($row->volume>=$max_batas){
	echo 'Diangkut';
	}
	else {
	echo 'Belum Diangkut';
	}?>
	</td>

	</tr > <?php endforeach;?> </table>

Flowgraph :



Gambar 4.10 *Flowgraph* Pengujian Optimasi Algoritma *Greedy*

Dari gambar diatas maka dapat ditentukan Cyclomatic Complexity sebagai berikut :

V(G) = E-N+2 = 9-8+2 = 3	E = Jumlah busur pada flow graph yaitu 9 N = Jumlah simpul pada flow graph yaitu 8
Basis Flow	**Jalur bebas (independent path)**
Jalur 1	1-2-3-4-6-8
Jalur 2	1-2-3-5-6-8
Jalur 3	1-2-3-4-6-7-8

2. Pengujian Rute Kunjungan

Node	Source Code

1	<script>
	var titik = <?=json_encode(array_values($arr_rute))?>;
	//var titik = ["A","B","C","D"];
	var cost_per_kilo = <?=get_option('cost_per_kilo')?>;
	function initMap() {
	var directionsService = new google.maps.DirectionsService;
	var directionsDisplay = new google.maps.DirectionsRenderer;
	var map = new google.maps.Map(document.getElementById('map'), {
	zoom: 7,center: {lat : -5.13792,lng : 119.511767}});
	directionsDisplay.setMap(map);
	directionsDisplay.setPanel(document.getElementById('right-panel'));
	calculateAndDisplayRoute(directionsService, directionsDisplay, map);
	}

2	function calculateAndDisplayRoute(directionsService,
	directionsDisplay, map) {directionsService.route({
	origin: <?=json_encode($origin)?>,
	destination: <?=json_encode($detination)?>,
	waypoints: <?=json_encode($waypoint)?>,
	optimizeWaypoints: false,travelMode: 'DRIVING'
	}, function(response, status) { if (status === 'OK') {
	directionsDisplay.setDirections(response);
	$('.rute').html('Rute: <?=implode(' - ', $arr_nama_titik)?>');
	$('.keterangan').html('<?=$ket_str?>');

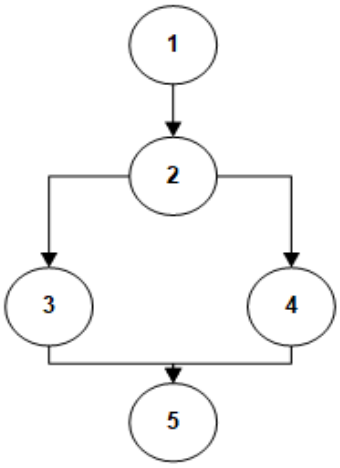
3	var total = 0;
	var result = directionsDisplay.getDirections();
	var myroute = result.routes[0];
	for (var i = 0; i < myroute.legs.length; i++) {
	total += myroute.legs[i].distance.value;}
	total = total / 1000;
	$('#total').html(total);
	var cost = Math.round(total * cost_per_kilo) ;
	//$('#biaya').html(cost.toLocaleString());
	$('#biaya').html(cost.toLocaleString());

4	var flightPlanCoordinates = <?=json_encode($arr_poly)?>; var flightPath = new google.maps.Polyline({ path: flightPlanCoordinates, geodesic: true,

	strokeColor: '#FF0000',
	strokeOpacity: 1.0,
	strokeWeight: 2 });
	flightPath.setMap(map);
	} else {
	window.alert('Directions request failed due to ' + status); }});}

5	$(function(){
	initMap();
	})
	</script>

Flowgraph :



Gambar 4.11 *Flowgraph* Pengujian Rute Kunjungan

Dari gambar diatas maka dapat ditentukan Cyclomatic Complexity sebagai berikut :

V(G) = E-N+2 = 5-5+2 = 2	E = Jumlah busur pada flow graph yaitu 5 N = Jumlah simpul pada flow graph yaitu 5

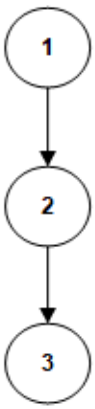
Basis Flow	Jalur bebas (independent path)
Jalur 1	1-2-3-5
Jalur 2	1-2-4-5

3. Pengujian Get Data Dari Server *Thingspeak*

Node	Source Code
1	<?php $url = "https://thingspeak.com/channels/496213/feeds.xml?results=500"; $xml = simplexml_load_file($url); $description = (string) $xml->name; print_r($description); $field1 = $xml->xpath('//feed/field1'); $field2 = $xml->xpath('//feed/field2'); $field3 = $xml->xpath('//feed/field3');
2	$i=0; foreach ($field1 as $onefield1) { $i++; $field1[$i]; $field2[$i]; $field3[$i]; $volume1= $field1[$i]; $volume2= $field2[$i]; $volume3= $field3[$i];

3	$db->query("INSERT INTO tb_titik_web (kode_kelompok, id_rt,kode_titik, nama_titik,volume, lat, lng) VALUES ('RW1', 'RT1','1', 'TPS 01 / Field 1', '$volume1','-5.128311', '119.502596')"); $db->query("INSERT INTO tb_titik_web (kode_kelompok, id_rt,kode_titik, nama_titik,volume, lat, lng) VALUES ('RW1', 'RT1','1', 'TPS 02 / Field 2', '$volume2','-5.128259', '119.502659')"); $db->query("INSERT INTO tb_titik_web (kode_kelompok,id_rt, kode_titik, nama_titik,volume, lat, lng) VALUES ('RW1', 'RT1','1', 'TPS 03 / Server', '$volume3','-5.1282', '119.502734')"); } ?>

Flowgraph :



Gambar 4.12 *Flowgraph* Pengujian Get Data Dari Server *Thingspeak*

Dari gambar diatas maka dapat ditentukan Cyclomatic Complexity sebagai berikut :

V(G) = E-N+2 = 2-3+2 = 1	E = Jumlah busur pada flow graph yaitu 2 N = Jumlah simpul pada flow graph yaitu 3
Basis Flow	**Jalur bebas (independent path)**
Jalur 1	1-2-3

E. Hasil Pengujian Perangkat Lunak

Berdasarkan hasil pengujian sistem yang dilakukan menggunakan metode pengujian *white box* maka dapat dianalisa bahwa perancang dapat mengetahui cara kerja dari aplikasi yang dirancang secara terperinci sesuai spesifikasi dan menilai apakah setiap fungsi atau prosedur yang dirancang sudah berjalan dengan baik dan benar dan menyimpulkan *independent path* dan keseluruhan flowgraph yang telah terurai. Dari hasil perhitungan diatas diperoleh nilai V(G), CC (*Cyclomatic Complexity*), dan *independent path*, maka dapat disimpulkan bahwa sistem bebas dari kesalahan logika.

BAB V

KESIMPULAN DAN SARAN

A. Kesimpulan

1. Sistem yang dibuat dapat mengatur proses logistik pengangkutan sampah dengan menerima data volume TPS dari server *thingspeak* ke *server* web untuk disimpan ke database *server* kemudian ditampilkan di halaman web untuk monitoring.
2. Algoritma *Greedy* dapat diimplementasikan untuk pencarian rute terpendek terhadap sistem yang real time dengan waktu komputasi yang lebih cepat.
3. Berdasarkan hasil perhitungan Algoritma *Greedy* optimasi rute terpendek dan volume TPS menghasilkan rute kunjungan yaitu dimulai dari (RT.1 -> RT.6 -> RT.5 -> RT.4 -> RT.3 -> RT.18 -> RT.17 -> RT.16 -> RT.15 -> RT.5 -> RT.8 -> RT.7 -> RT.10 -> RT.11 -> RT.12 -> RT.13 -> RT.2 -> RT.14 -> RT.9) dengan total jarak tempuh 18.332 Km dan volume sampah sebanyak 10.000 liter.

B. Saran

1. Pengembangan algoritma untuk implementasi secara pararel
2. Adanya perluasan lingkup studi kasus, yaitu tidak hanya pada satu kelurahan tapi dalam lingkup perkotaan.
3. Menggunakan algoritma berbeda untuk mencari rute terpendek nantinya untuk menjadi perbandingan dengan algoritma greedy.

DAFTAR PUSTAKA

[1] I. Markov, S. Varone and M. Bierlaire, "Vehicle Routing for a Complex Waste Collection Problem," 2014.

[2] V. N. Bhat, "A Model for the Optimal Allocation of Trucks for Solid Waste Management," 1996.

[3] M. T. P. B. B. d. Aguiar, "Optimization Techniques for the Mixed Urban Rural Solid Waste Collection Problem," 2010.

[4] D. B. Liesje, J. Beliën and J. V. Ackere, Municipal Solid Waste Collection and Management Problems: A Literature Review, pp. 1-4, 2013.

[5] S. D. d. M. Chaerul, "EVALUASI SISTEM PENGANGKUTAN SAMPAH DI WILAYAH BANDUNG UTARA," 2010.

[6] Raden Rogers Dwiputra Setiady, " Perancangan Dan Implementasi Metode Heuristic Untuk Vehicle Routing Problem Pada Pengangkutan Sampah," 2016.

[7] Danang Triwibowo, " Aplikasi Model Optimasi untuk Meningkatkan Efisiensi Pengangkutan Sampah di Kota Cilegon," 2015.

[8] Dea Widya Hutami, " Implementasi Algoritma Nearest Insertion Heuristic dan Modified Nearest Insertion Heuristic Pada Optimasi Rute Kendaraan Pengangkut Sampah (Studi Kasus: Dinas Kebersihan dan Pertamanan Kota Malang)," 2017.

[9] Arna Fariza, "Optimasi Penjadwalan Pengangkutan Sampah Di Surabaya Secara Adaptif Menggunakan Metode Algoritma Genetika," 2014.

[10] M. Rasyid Ridha, "Studi Optimasi Rute Pengangkutan Sampah Kota Marabahan Dengan Sistem Informasi Geografis," 2016.

[11] Amrin Amin, "Optimalisasi Pengangkutan Sampah Di Pusat Kota Ternate," 2011.

LAMPIRAN

1. Script Home

```
<?php
include'functions.php';
if(empty($_SESSION['login']))

header("location:login.php");
?>
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8"/>
    <meta http-equiv="X-UA-Compatible" content="IE=edge"/>
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <link rel="icon" href="favicon.ico"/>

    <title>OPTIMASI TPS</title>
    <link href="assets/css/cerulean-bootstrap.min.css" rel="stylesheet"/>
    <link href="assets/css/select2.min.css" rel="stylesheet"/>
    <link href="assets/css/general.css" rel="stylesheet"/>
    <script src="assets/js/jquery.min.js"></script>
    <script src="assets/js/bootstrap.min.js"></script>
    <script src="assets/js/select2.min.js"></script>
    <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCiDdGyp6n2hKHPECuB6JZIT-8dVHCpwI0&language=id&region=ID&libraries=places"></script>
    <script>

(document).ready(function() {
          $('.s2').select2();
        });
    </script>
  </head>
  <body>
    <nav class="navbar navbar-default navbar-static-top">
      <div class="container">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar" aria-expanded="false" aria-controls="navbar">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <a class="navbar-brand" href="?">OPTIMASI TPS</a>
        </div>
        <div id="navbar" class="navbar-collapse collapse">
```

```
        <ul class="nav navbar-nav">
          <li><a href="?m=kelompok"><span class="glyphicon glyphicon-th-large"></span> DATA RW</a></li>
      <li><a href="?m=RT"><span class="glyphicon glyphicon-th-large"></span> DATA RT</a></li>
          <li><a href="?m=titik"><span class="glyphicon glyphicon-th-list"></span> DATA TPS</a></li>

          <!-- <li><a href="?m=bobot"><span class="glyphicon glyphicon-star"></span> Bobot</a></li> -->
                    <!-- <li><a href="?m=dfs"><span class="glyphicon glyphicon-signal"></span> DFS</a></li> -->
                    <li><a href="?m=greedy"><span class="glyphicon glyphicon-signal"></span> GREEDY</a></li>
          <!-- <li><a href="?m=hitung"><span class="glyphicon glyphicon-signal"></span> AG</a></li> -->
          <li><a href="?m=password"><span class="glyphicon glyphicon-lock"></span> Password</a></li>
          <!-- <li><a href="?m=pengaturan"><span class="glyphicon glyphicon-cog"></span> Pengaturan</a></li> -->
                    <li><a href="?m=titik_web"><span class="glyphicon glyphicon-cog"></span> IOT DATA</a></li>
          <li><a href="?m=maps"><span class="glyphicon glyphicon-cog"></span> MAPS</a></li>
          <li><a href="?m=rute_angkut"><span class="glyphicon glyphicon-cog"></span> RUTE</a></li>
          <li><a href="aksi.php?act=logout"><span class="glyphicon glyphicon-log-out"></span> Logout</a></li>
        </ul>
      </div>
    </nav>

    <div class="container">
    <?php

if(file_exists($mod.'.php'))
        include $mod.'.php';
      else
        include 'home.php';
    ?>
    </div>
    <footer class="footer bg-primary">
      <div class="container">
        <p>Copyright &copy; <?=date('Y')?> TugasAkhir.Id</p>
```

```
    </div>
  </footer>
</html>
```

2. Script Algoritma *Greedy*

```
<div class="page-header">

<center><h2>PRIORIATAS ANGKUT BERDASARKAN VOLUME</h2></center>
</div>
<style>
    #right-panel {
      font-family: 'Roboto','sans-serif';
      line-height: 30px;
      padding-left: 10px;
    }

    #right-panel select, #right-panel input {
      font-size: 15px;
    }

    #right-panel select {
      width: 100%;
    }

    #right-panel i {
      font-size: 12px;
    }

    #map{
      height: 500px;
      float: left;
      width: 63%;
    }
    #right-panel {
      float: right;
      width: 34%;
      height: 500px;
      overflow: auto;
    }
</style>
 <table class="table table-bordered table-hover table-striped">
   <thead>
     <tr>
       <th>No</th>
       <th>RW</th>
       <th>RT</th>
       <th>Volume</th>
       <th>MAX</th>
       <th>Lat</th>
       <th>Lng</th>
       <th>Status</th>

     </tr>
   </thead>
   <?php

   #
   function rad($x){ return $x * M_PI / 180; }
          function distHaversine($coord_a, $coord_b){
              # jarak kilometer dimensi (mean radius) bumi
              $R = 6371;
              $coord_a = explode(",",$coord_a);
              $coord_b = explode(",",$coord_b);
              $dLat = rad(($coord_b[0]) - ($coord_a[0]));
              $dLong = rad($coord_b[1] - $coord_a[1]);
              $a = sin($dLat/2) * sin($dLat/2) + cos(rad($coord_a[0])) * cos(rad($coord_b[0])) * sin($dLong/2) * sin($dLong/2);
              $c = 2 * atan2(sqrt($a), sqrt(1-$a));
```

```
                $d = $R * $c;
                # hasil akhir dalam satuan kilometer
                return number_format($d, 0, '.', ',');
            }

    $batas=10000;
    $truck=0;
    $sisa;

    $db->query("TRUNCATE TABLE tb_titik_greedy");
    $db->query("TRUNCATE TABLE tb_titik_greedy_dump");


    $rows1 = $db->get_results("SELECT nama_titik, kode_kelompok,id_rt,kode_titik,
                    SUM(volume) as volume,lat,lng,(COUNT(kode_titik)*38) as max
                    FROM tb_titik GROUP BY id_rt; ");
    foreach($rows1 as $row):

            $kode_kelompok = $row->kode_kelompok;
            #$kode_titik=$row->kode_titik ;
            $kode_titik=$row->id_rt ;
            $volume=$row->volume;
            $lat=$row->lat;
            $lng=$row->lng;
            $max=$row->max;
            ## cara penggunaannya
            ## contoh ada 2 koordinat (latitude dan longitude)
            $asal = "-5.13792,119.511767";
            $b=$lat.",".$lng;

            $jarak= distHaversine($asal, $b);



    $db->query("INSERT INTO tb_titik_greedy (kode_kelompok, kode_titik, nama_titik,volume,max, lat, lng,jarak)
            VALUES ('$kode_kelompok', '$kode_titik', '$nama_titik', '$volume','$max','$lat', '$lng','$jarak')");



    $max_batas=$max/2;

    if ($volume>=$max_batas){

        if ($volume<=$batas) {
            $batas-=$volume;
            $volume2=0;
        }
        else if ($volume>$batas){

            $volume2=$volume-$batas;
            $batas-=$volume2;
```

```
        }
    }
    else
$volume2=$volume;
    $db->query("INSERT
INTO
tb_titik_greedy_dump
(kode_kelompok,
kode_titik,
nama_titik,volume,max, lat,
lng,jarak)
          VALUES
('$kode_kelompok',
'$kode_titik',   '$nama_titik',
'$volume2','$max','$lat',
'$lng','$jarak')")    ;

    endforeach;



    //$rows = $db-
>get_results("SELECT *
FROM tb_titik_greedy
ORDER by volume DESC
");
    $rows = $db-
>get_results("SELECT *
FROM tb_titik_greedy
WHERE volume>=max/2
ORDER by volume DESC
");
    $no=1;
    foreach($rows as $row):


    ?>

    <tr >
        <td><?=$no++?></td
>
        <td><?=$row-
>kode_kelompok?></td>
        <td><?=$row-
>kode_titik ?></td>
        <td><?=$row-
>volume?></td>
        <td><?=$row-
>max?></td>
        <td><?=$row-
>lat?></td>
        <td><?=$row-
>lng?></td>
        <td>
        <?=$row-
>$max_batas;
        echo 'Diangkut';
        ?>


        </td>

    </tr >
    </font>
    <?php endforeach;?>
    </table>
    <table>

    <h2><center>BELUM
DIANGKUT</center></h2>


    <table class="table
table-bordered table-hover
table-striped">
    <thead>
        <tr>
            <th>NO</th>
            <th>RW</th>
            <th>RT</th>
            <th>Volume</th>
            <th>Max</th>
            <th>Lat</th>
            <th>Lng</th>
            <th>Status</th>

        </tr>
    </thead>
    <?php
```

```
   $rows3 = $db->get_results("SELECT * FROM tb_titik_greedy WHERE volume<=max/2 ORDER by volume DESC ");
   $no=1;
   foreach($rows3 as $row):?>
   <tr >
      <td><?=$no++?></td>
      <td><?=$row->kode_kelompok?></td>
      <td><?=$row->kode_titik ?></td>
      <td><?=$row->volume?></td>
      <td><?=$row->max?></td>
      <td><?=$row->lat?></td>
      <td><?=$row->lng?></td>
      <td>
      <?=$row->$max_batas;
      echo 'Belum Diangkut';
      ?>

      </td>

   </tr >
   </font>
   <?php endforeach;?>
   </table>
   <table>


   <h2><center>PRIORITAS ANGKUT BERDASARKAN JARAK</center></h2>
```

```
   <table class="table table-bordered table-hover table-striped">
   <thead>
      <tr>
         <th>NO</th>
         <th>RW</th>
         <th>RT</th>
         <th>Volume</th>
         <th>Max</th>

         <th>Lat</th>
         <th>Lng</th>
         <th>Status</th>

      </tr>
   </thead>

   <?php


      $rows2 = $db->get_results("SELECT * FROM tb_titik_greedy WHERE volume>=max/2 ORDER by jarak ASC ");
   $no=1;
   foreach($rows2 as $row):?>
   <tr>
      <td><?=$no++?></td>
      <td><?=$row->kode_kelompok?></td>
      <td><?=$row->kode_titik ?></td>
      <td><?=$row->volume?></td>
      <td><?=$row->max?></td>
      <td><?=$row->lat?></td>
      <td><?=$row->lng?></td>
      <td>
```

```
        <?=$row->max_batas;
        echo 'Diangkut';
        ?>

        </td>

    </tr>
    <?php endforeach;?>
    </table>



    <h2><center>SIMULASI SETELAH DIANGKUT</center></h2>

    <table class="table table-bordered table-hover table-striped">
    <thead>
        <tr>
            <th>No</th>
            <th>RW</th>
            <th>RT</th>
            <th>Sisa Volume</th>
            <th>Max</th>
            <th>Lat</th>
            <th>Lng</th>
            <th>Status</th>

        </tr>
    </thead>

    <?php


        $rows2 = $db->get_results("SELECT * FROM tb_titik_greedy_dump1 ORDER by jarak ASC ");
    $no=1;
    foreach($rows2 as $row):?>

    <tr>

<td><?=$no++?></td>
        <td><?=$row->kode_kelompok?></td>
        <td><?=$row->kode_titik ?></td>
        <td><?=$row->volume?></td>
        <td><?=$row->max?></td>
        <td><?=$row->lat?></td>
        <td><?=$row->lng?></td>
        <td><?=$row->max_jarak;
        if ($row->volume>0){
        echo 'Belum Diangkut';
        }
        else {
        echo 'Diangkut';
        }
        ?>

        </td>

    </tr>
    <?php
    $sisa+=$row->volume;
    endforeach;?>
    </table>

    SISA TOTAL = <?php echo $sisa;?>
    <br>
```

```
<div class="panel panel-primary ">
  <div class="panel-heading">
    <h3 class="panel-title">Hasil TSP</h3>
  </div>
  <div class="panel-body">
    <div class="row">
      <div class="col-md-10">
        <p class="rute"></p>
        <div>
          <div id="map" class="thumbnail"></div>
          <div id="right-panel" class="small">
            <p style="font-weight: bold;" class="text-danger">
              Total Jarak: <span id="total"></span> Km<br />
              Total Durasi: <span id="durasi"></span>
            </p>
          </div>
        </div>
      </div>
      <div class="col-md-2">
        <p class="keterangan"></p>
      </div>
    </div>
  </div>
</div>
<?php
$success = true;
$a = 1;
$b = 1;
$c = 75;
$d = 25;
$titik_tujuan = $_GET['titik_tujuan'];


  $arr= $db->get_results("SELECT * FROM tb_titik_greedy WHERE volume>=max/2 ORDER by jarak ASC ");
  ############ TITIK AWAL DUMP TRUCK
  $origin = array(
    'lat' => -5.13792,
    'lng' => 119.511767,
  );
  $detination=$origin;

  $waypoint = array();
  for($a = 1; $a < count($arr) - 1; $a++){
    $waypoint[] = array(
      'location'=> array(
        'lat' => $arr[$a]->lat * 1,
        'lng' => $arr[$a]->lng * 1,
      ),
      'stopover' => TRUE,
    );
  }


  $ket_str = "Keterangan: <br />";
  $ascii = 65;

    $arr= $db->get_results("SELECT * FROM tb_titik_greedy WHERE volume>=max/2 ORDER by jarak ASC ");
```

```
    $arr_nama=array();

    foreach($arr as $key){
        $chr = chr($ascii++);
        $arr_rute[] = $chr;
        #$ket_str.=  "$chr  =
$key->nama_titik <br />";
        $ket_str.=  "$chr  =
$key->kode_titik <br />";

          $kode_titik.=  "$key-
>kode_titik <br />";

    }


    $arr_poly = array();
    $arr_nama_titik          =
array();
    foreach($arr as $key){

        $arr_poly[] = array(
            'lat' => $key->lat * 1,
            'lng' => $key->lng *
1,
        );

        $arr_nama_titik[]=
$key->kode_titik;
    }



?>
<script>
var         titik                =
<?=json_encode(array_val
ues($arr_rute))?>;
//var         titik                =
["A","B","C","D"];
var       cost_per_kilo      =
<?=get_option('cost_per_ki
lo')?>;
function initMap() {
    var directionsService =
new
google.maps.DirectionsSer
vice;
    var directionsDisplay =
new
google.maps.DirectionsRe
nderer;
    var      map      =      new
google.maps.Map(docume
nt.getElementById('map'), {
      zoom: 7,
      center: {
        lat : -5.13792,
        lng : 119.511767
      }
    });

directionsDisplay.setMap(
map);

directionsDisplay.setPanel(
document.getElementById(
'right-panel'));

calculateAndDisplayRoute(
directionsService,
directionsDisplay, map);
}

function
calculateAndDisplayRoute(
directionsService,
directionsDisplay, map) {
    directionsService.route({
      origin:
<?=json_encode($origin)?
>,
      destination:
<?=json_encode($detinatio
n)?>,
      waypoints:
<?=json_encode($waypoin
t)?>,
      optimizeWaypoints:
false,
```

```
    travelMode: 'DRIVING'
  }, function(response, status) {
    if (status === 'OK') {


directionsDisplay.setDirections(response);

      $('.rute').html('Rute: <?=implode(' - ', $arr_nama_titik)?>');


$('.keterangan').html('<?=$ket_str?>');

      //mencari total jarak
    var total = 0;
      var durasi = 0;
      var result = directionsDisplay.getDirections();
      var myroute = result.routes[0];

      console.log(myroute);

      for (var i = 0; i < myroute.legs.length; i++) {
total+=myroute.legs[i].distance.value;
durasi+=myroute.legs[i].duration.value;
      }
      total = total / 1000;
      $('#total').html(total);
      var cost = Math.round(total * cost_per_kilo) ;

$('#biaya').html(cost.toLocaleString());

      var str_durasi = '';

      var hari = durasi / 24 / 60 / 60;
      if(hari >= 1)
        str_durasi+= Math.ceil(hari) + ' hari ';

      var jam = (durasi % (60 * 60 * 24)) / 60 / 60;
      if(jam >= 1)
        str_durasi+= Math.ceil(jam) + ' jam ';

      var menit = (durasi % (60 * 60)) / 60 ;
      if(menit >= 1)
        str_durasi+= Math.round(menit) + ' menit ';

('#durasi').html(str_durasi);
var flightPlanCoordinates = <?=json_encode($arr_poly)?>;
      var flightPath = new google.maps.Polyline({
        path: flightPlanCoordinates,
        geodesic: true,
        strokeColor: '#FF0000',
        strokeOpacity: 1.0,
        strokeWeight: 2
      });

flightPath.setMap(map);
          } else {

window.alert('Directions request failed due to ' + status);
    }  });}
$(function(){
  initMap();
})
</script>
```