PERANCANGAN GUI *FLEET MONITORING AND CONTROL SISTEM (FMCS)* UNTUK *GUIDED BUS*

TUGAS AKHIR

Oleh

AULIA HENING DARMASTI

NIM: 13213136

Program Studi Teknik Elektro

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2017

PERANCANGAN GUI *FLEET MONITORING AND CONTROL SISTEM (FMCS)* UNTUK *GUIDED BUS*

Oleh:

AULIA HENING DARMASTI

Tugas Akhir ini telah diterima dan disahkan sebagai persyaratan untuk memperoleh gelar

SARJANA TEKNIK

di

PROGRAM STUDI TEKNIK ELEKTRO
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

Bandung, 7 Mei 2017

Disetujui oleh :

Pembimbing I,

Ir. Arief Syaichu
Rohman, MEngSc, Ph.D

Pembimbing II,

Arif Sasongko, S.T., M.Sc.,Ph.D

Pembimbing III,

Ir. Agung Darmawan

ABSTRAK

PERANCANGAN GUI *FLEET MONITORING AND CONTROL SISTEM (FMCS)* UNTUK *GUIDED BUS*

Oleh

Aulia Hening Darmasti

NIM: 13213136

PROGRAM STUDI TEKNIK ELEKTRO

Pada saat ini, belum tersedia fasilitas transportasi massal yang cukup mumpuni di kota Bandung. Kenyamanan dan waktu menjadi faktor utama bagi masyarakat untuk memilih moda transportasi. Sementara pada saat ini diantara semua pilihan transportasi tidak ada satupun yang menjanjikan kenyamanan dan bebas dari macet. Untuk itu dirancang Fleet Monitoring and Control System (FMCS) untuk guided bus sebagai jawaban dari ketiadaan pilihan transportasi yang nyaman, aman dan bebas macet. FMCS mampu untuk melakukan pemantauan posisi armada diseluruh trayek, menampilkan data fisik dari masing-masing armada, penjadwalan, hingga memberi komando dari control station agar pergerakan armada tetap sesuai dengan penjadwalan. Salah satu komponen penting dari FMCS ialah GUI (Graphical User Interface). GUI ini berfungsi sebagai perantara antar sistem dan operator untuk saling berinteraksi. Lewat GUI, operator dapat melakukan pemantauan posisi armada baik dengan peta geografis, peta diagramatis, memantau data fisik armada, mengatur penjadwalan, menangani situasi darurat, hingga memberi instruksi pada armada. GUI yang dibangun dengan menggunakan Windows Forms pada Microsoft Visual Studio ini memiliki kapasitas untuk menampilkan hingga 40 posisi armada dan 10 data lengkap armada secara bersamaan tanpa membebani PC yang digunakan. Selain itu GUI ini juga dirancang dengan memperhitungkan kenyaman pengguna dalam pemakaiannya. Kedepannya, GUI dapat dikembangkan dengan mengurangi ketelitian array segmentasi. Hal ini disebabkan karena urgensi untuk melakukan segmentasi trayek sesuai dengan panjang bus tidak terlalu dibutuhkan.

Kata kunci: Pemantauan, Pengendalian, Antarmuka, Desain Interaksi.

ABSTRACT

FLEET MONITORING AND CONTROL SISTEM (FMCS) FOR GUIDED BUS GRAPHICAL USER INTERFACE DESIGN

By

Aulia Hening Darmasti

NIM: 13213136

DEPARTMENT OF ELECTRICAL ENGINEERING

Comfortability and time are the main factors for citizen to choose the transportation option. Meanwhile there is none of transportation option which promise comfort and traffic-free. For those reasons, Fleet Monitoring and Control System (FMCS) for guided bus are designed to fulfilled the absence of convenient, safe, and traffic-free transportation options. FMCS is able to monitor fleet positions throughout the route, displaying physical data of each fleet, maintaining scheduling algorithm, and giving command from control station so that fleet movement remains in accordance with the scheduling. One of important part of FMCS is GUI (Graphical User Interface). GUI serves as an intermediary between systems and operators to interact each other. Through GUI, operators can monitor fleet positions with geographic maps, diagrammatic maps, monitor fleet physical data, manage scheduling, handle emergency situations, and giving command to fleet. This GUI built using Windows Forms in Microsoft Visual Studio and has aa capacity to display up to 40 fleet positions and 10 complete fleet data simultaneously without burdening the PC that used. In addition, GUI is also designed to take into account the convenience of users in its use. In the future, GUI can be developed by reduce the accuracy of segmentation array. Segmenting route display according to the bus length not that urgent for this GUI.

Keyword: Monitoring, Controlling, Interface, UX Design.

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah subhanahu wa ta'ala, karena atas rahmat dan karuna-Nya penulis dapat menyelesaikan penulisan tugas akhir berjudul "Perancangan GUI untuk *Fleet Monitoring and Control Sistem* pada *Guided Bus*" ini tepat pada waktunya. Shalawat dan salam tercurah kepada Rasulullah juga keluarga, sahabat, dan umatnya hingga akhir zaman.

Buku ini merupakan dokumentasi dari pengerjaan tugas akhir selama tingkat akhir penulis di elektro, yang dibuat dalam rangka memenuhi syarat kelulusan dari program studi Teknik Elektro di Institut Teknologi Bandung.

Dalam pelaksanaan seluruh pengerjaan tugas akhir ini, penulis mengalami berbagai masalah. Namun, atas bantuan dari berbagai pihak, penulis dapat melaksanakannya dengan baik. Maka dari itu, penulis mengucapkan terima kasih kepada :

1. Allah SWT, atas karunia dan kasih sayang-Nya,
2. Ayah dan Ibu, atas seluruh dukungannya baik dari segi materi, moril, dan doa-doanya di setiap shalat,
3. De Alka dan De Aura, yang menjadi penghibur selama pengerjaan TA ini dan seluruh keluarga besar yang telah memberi dukungan kepada penulis,
4. Bapak Arief Syaichu dan Bapak Arif Sasongko selaku dosen pembimbing atas bimbingan, ilmu, dan antusiasme yang selalu beliau berikan kepada kami,
5. Bapak Agung Darmawan selaku pimpro sekaligus pembimbing dari PT. LEN Industri.
6. Pak Firdaus, Pak Ahmad Husnan, Mas Ilmi dan Mas Wahid dari PT. LEN Industri,
7. Shah Dehan Lazuardi dan Ali Zaenal selaku teman satu tim mulai dari tubes, lomba, hingga pengerjaan tugas akhir ini,
8. Vannisa Rindita dan Adinda Rana Trisanti selaku sahabat penulis yang selalu memberi dukungan kepada penulis selama mengerjakan tugas akhir ini,
9. Seluruh teman di Ruang Riset Mandiri (Goa Depan) yaitu Fadhil, Irham, Audi, Tandut, Dinan, Iki, Vito, Eki, Juan, Rama, Billie, Andin, Ikar, Ducun, dan

Santo yang telah sama-sama berjuang melalui kegiatan tugas akhir dalam suka dan duka.

10. Serta pihak-pihak yang tidak dapat dituliskan satu persatu namanya, yang telah membantu kegiatan tugas akhir ini hingga selesai.

Tak ada gading yang tak retak, penulis menyadari bahwa tugas akhir ini jauh dari kata sempurna, maka dari itu penulis dengan terbuka menerima kritik dan masukkan yang membangun sebagai bahan perbaikan untuk masa yang akan datang. Akhir kata, penulis berharap semoga laporan ini dapat bermanfaat bagi siapapun yang membacanya.

Bandung, 7 Mei 2017

Penulis

DAFTAR ISI

DAFTAR GAMBAR

BAB I

PENDAHULUAN

Pada bab ini akan dijelaskan mengenai dasar dari pembuatan GUI untuk FMCS pada Guided Bus. Diantaranya latar belakang yang mendasari pembuatan, rumusan masalah yang digarisbawahi dari latar belakang dan berkembang menjadi tujuan utama tugas akhir ini. Setelah itu akan dijelaskan juga lingkup permasalahan, metodologi, hingga sistematika penulisan dokumen ini.

1.1 Latar Belakang

Beberapa dasawarsa terakhir, dunia telah mengalami perkembangan teknologi yang cukup pesat. Hal ini menyebabkan meningkatnya mobilitas penduduk dan tentu saja, meningkatnya juga kebutuhan akan transportasi. Pemerintah kota Bandung, sesuai dengan itu, turut menyediakan berbagai jenis transportasi massal bagi warganya.

Tetapi ternyata solusi yang disediakan oleh pemerintah belum cukup menjadi pilihan yang baik bagi masyarakat. Kemudahan dan kenyamanan menjadi sorotan utama bagi masyarakat dalam memilih transportasi. Dan pada fakta yang kita ketahui, penggunaan transportasi massal di Bandung masih belum dapat menghindarkan kita dari kemacetan dan ketidaknyamanan.

Masalah utama yang tidak dapat dihindari dari transportasi massal yang telah ada ialah tidak adanya sebuah sistem yang dapat memantau dan mengendalikan moda transportasi. Sehingga tentu ngetem, berhenti disembarang tempat, bahkan penarikan tarif diatas normal menjadi alasan-alasan yang membuat masyarakat kota Bandung tidak nyaman menggunakan transportasi massal.

Bekerja sama dengan PT. LEN Industri yang sedang merintis prototype guided bus berbahan bakar listrik, Fleet Monitoring and Control Sistem untuk guided bus dirancang sebagai upaya dalam mengatasi permasalahan tersebut. Salah satu subsistem yang dibutuhkan dalam FMCS adalah GUI yang digunakan oleh operator untuk melakukan pemantauan.

Dalam makalah ini akan dijabarkan mulai dari perancangan, implementasi, hingga pengujian dan kesimpulan dalam pembuatan GUI untuk FMCS.

1.2 Rumusan Masalah

Berdasarkan pada latar belakang yang telah dijabarkan sebelumnya, maka rumusan masalah dalam tugas akhir ini dapat dijabarkan sebagai berikut:

- Bagaimana cara menciptakan sistem yang dapat memantau seluruh armada dalam satu kesatuan?
- Bagaimana cara menciptakan sistem yang dapat mengendalikan pergerakan seluruh armada?

1.3 Tujuan Tugas Akhir

Tujuan umum dilakukannya tugas akhir ini adalah sebagai berikut:

1. Pemenuhan salah satu syarat kelulusan pada program sarjana program studi Teknik Elektro.
2. Riset pengembangan sistem manajemen untuk *guided bus.*
3. Riset untuk implementasi dan aplikasi Internet of Things dalam dunia transportasi.

Tujuan khusus dilakukannya tugas akhir ini adalah sebagai berikut:

1. Melakukan perancangan dan implementasi sistem yang dapat mengirimkan data-data armada.
2. Membuat sistem komunikasi antara armada dan kendali pusat
3. Melakukan perancangan dan implementasi algoritma penjadwalan serta display kendali pusat.

1.4 Lingkup Permasalahan

Tugas akhir ini dibuat dengan asumsi dan batasan sebagai berikut:

- GUI yang dibuat masih berupa prototipe pertama.

- Pengujian pada integrasi keseluruhan dilakukan dengan menggunakan kendaraan biasa (bukan guided bus listrik) sehingga data yang dikirim hanya berupa data posisi.
- GUI dirancang untuk menampilkan maksimal 40 marker posisi secara bersamaan jika diinginkan tanpa *lag/error*.

1.5 Metodologi

Dalam pengerjaan tugas akhir ini, penulis beserta tim melakukan serangkaian langkah yang terangkum dalam susunan metodologi berikut:

1. Perumusan Masalah

   Pada tahap ini dilakukan *brainstorming* untuk mencari latar belakang serta merumuskan masalah-masalah utama dalam topik yang telah ditentukan.
2. Menentukan Spesifikasi

   Setelah pokok-pokok permasalahan dijabarkan, ditentukan spesifikasi yang harus dipenuhi agar desain yang dibuat dapat menjadi solusi bagi permasalahan.
3. Studi Literatur

   Pada tahap ini dilakukan pengajian teori serta teknik yang berhubungan dengan solusi yang akan dirancang pada tugas akhir ini.
4. Perancangan dan Implementasi

   Perancangan produk berangkat dari spesifikasi-spesifikasi sistem yang telah ditentukan sebelumnya, dan kemudian menjadi dasar bagi proses implementasi sistem.
5. Pengujian dan Evaluasi Sistem

   Tahap pengujian sistem dilakukan untuk memvalidasi performansi dan kinerja sistem, serta tahap untuk melakukan perbaikan agar mencapai spesifikasi yang diinginkan.
6. Kesimpulan

   Berisi kesimpulan dari seluruh aktivitas pengujian, analisis, serta ketercapaian dari tujuan tugas akhir.

1.6 Sistematika Penulisan

Buku tugas akhir in disusun dalam lima bab yang sistematika penulisannya adalah sebagai berikut:

a. BAB I PENDAHULUAN

   Pada bab ini dijabarkan latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, serta sistematika penulisan buku tugas akhir.

b. BAB II TINJAUAN PUSTAKA

   Pada bab ini dituliskan konsep-konsep dan studi pustaka yang berhubungan dengan topik yang dibahas.

c. BAB III DESAIN DAN IMPLEMENTASI

   Pada bab ini dijelaskan tentang seluruh tahap implementasi. Mulai dari spesifikasi sistem, perancangan, hingga proses implementasi.

d. BAB IV HASIL PENGUJIAN

   Pada bab ini dijabarkan proses pengujian dan hasil dari pengujian terhadap sistem yang telah dibuat.

e. BAB V PENUTUP

   Pada bab ini dituliskan kesimpulan yang diperoleh dari hasil tugas akhir yang telah dimuat pada bab-bab sebelumnya beserta saran pengembangan.

BAB II

GUI UNTUK FMCS PADA *GUIDED BUS*

Bab ini menjabarkan seluruh ilmu dan referensi yang digunakan selama pengerjaan GUI untuk FMCS pada Guided Bus ini. Berangkat dari Microsoft Visual Studio sebagai IDE utama, hingga penjelasan tentang bahasa pemrograman C# dan Object Oriented Programming yang digunakan untuk membangun GUI.

2.1 Microsoft Visual Studio

Gambar 2. 1 Logo Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (suite) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web. Visual Studio mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro, dan Visual SourceSafe.

Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam native code (dalam bentuk bahasa mesin yang berjalan di atas Windows) ataupun managed code (dalam bentuk Microsoft Intermediate Language di atas .NET Framework). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi Silverlight, aplikasi Windows Mobile (yang berjalan di atas .NET Compact Framework).

Visual Studio kini telah menginjak versi Visual Studio 9.0.21022.08, atau dikenal dengan sebutan Microsoft Visual Studio 2008 yang diluncurkan pada 19 November 2007, yang ditujukan untuk platform Microsoft .NET Framework 3.5. Versi

sebelumnya, Visual Studio 2005 ditujukan untuk platform .NET Framework 2.0 dan 3.0. Visual Studio 2003 ditujukan untuk .NET Framework 1.1, dan Visual Studio 2002 ditujukan untuk .NET Framework 1.0. Versi-versi tersebut di atas kini dikenal dengan sebutan Visual Studio .NET, karena memang membutuhkan Microsoft .NET Framework. Sementara itu, sebelum muncul Visual Studio .NET, terdapat Microsoft Visual Studio 6.0 (VS1998).

2.2 Bahasa Pemrograman C#

C# atau yang dibaca C sharp adalah bahasa pemrograman sederhana yang digunakan untuk tujuan umum, dalam artian bahasa pemrograman ini dapat digunakan untuk berbagai fungsi misalnya untuk pemrograman server-side pada website, membangun aplikasi desktop ataupun mobile, pemrograman game dan sebagainya. Selain itu C# juga bahasa pemrograman yang berorientasi objek, jadi C# juga mengusung konsep objek seperti inheritance, class, polymorphism dan encapsulation.

Dalam prakteknya C# sangat bergantung dengan framework yang disebut .NET Framework, framework inilah yang nanti digunakan untuk mengcompile dan menjalankan kode C#. C# dikembangkan oleh Microsoft dengan merekrut Anders Helsberg. Tujuan dibangunnya C# adalah sebagai bahasa pemrograman utama dalam lingkungan .NET Framework (lihat C#). Banyak pihak juga yang menganggap bahwa Java dengan C# saling bersaing, bahkan ada juga yang menyatakan jika pernah belajar Java maka belajar C# akan sangat mudah dan begitu juga sebaliknya. Anggapan tersebut sebenarnya tidak salah karena perlu diketahui sebelum adanya C# Microsoft mengembangkan J++ dengan maksud mencoba membuat Java agar berjalan pada platform Windows, karena adanya masalah dari pihak luar maka Microsoft menghentikan proyek J++ dan beralih untuk mengembangkan bahasa baru yaitu C#.

Dalam pemrograman C# (mungkin juga berlaku untuk beberapa pemrograman lainnya) memiliki 5 struktur dasar yang harus diingat yaitu,

1. Resource atau library : pendefinisian library apa yang harus ada pada program kita atau library apa yang kita impor.
2. Namespace : nama dari project.

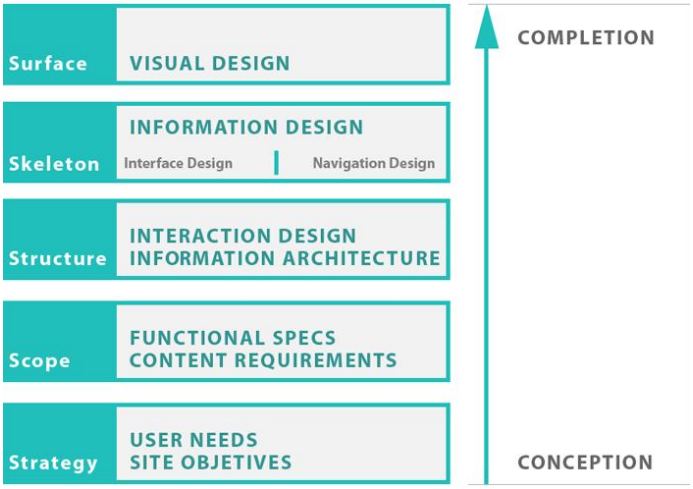
3. Nama Class : nama dari Class yang kita buat dan bisa juga langsung diberi penanda seperti Main Class yang menandakan bahwa Class tersebut Class utama.
4. Deklarasi Method : pendeklarasian method sebagai awalan untuk menjalankan method atau perintah yang ada di dalamnya, jika didefinisikan dengan "Main" maka method tersebut yang dijalankan pertama kali oleh compiler.
5. Method atau Command : method atau perintah yang kita berikan untuk di eksekusi oleh compiler.

2.3 Object Oriented Programming

OOP (Object Oriented Programming) adalah suatu metode pemrograman yang berorientasi kepada objek. Tujuan dari OOP diciptakan adalah untuk mempermudah pengembangan program dengan cara mengikuti model yang telah ada di kehidupan sehari-hari. Jadi setiap bagian dari suatu permasalahan adalah objek, objek itu sendiri merupakan gabungan dari beberapa objek yang lebih kecil lagi. Sebuah objek yang besar dibentuk dari beberapa objek yang lebih kecil, kemudian objek-objek itu saling berkomunikasi, dan saling berkirim pesan kepada objek yang lain. OOP memiliki konsep yang terbagi-bagi dalam beberapa bagian berikut:

1. Kelas Abstrak (Class Abstraksi): merupakan deskripsi abstrak informasi dan tingkah laku dari sekumpulan data. Dapat diilustrasikan sebagai suatu cetak biru(blueprint) atau prototipe yang digunakan untuk menciptakan objek.
2. Enkapsulasi (encapsulation): adalah kombinasi data dan fungsionalitas dalam sebuah unit tunggal sebagai bentuk untuk menyembunyikan detail informasi.
3. Pewarisan (Inheritance): programmer dapat mendefinisikan suatu kelas baru dengan mewarisi sifat dari kelas lain yang sudah ada. Dengan konsep pewarisan, seorang programmer dapat menggunakan kode yang telah ditulisnya pada kelas super berulang kali pada kelas-kelas turunannya tanpa harus menulis ulang semua kodekode itu.
4. Polimorfisme (polymorphism): kemampuan objek objek yang berbeda kelas namun terkait dalam pewarisan untuk merespon secara berbeda terhadap suatu pesan yang sama.

2.3 UX Design



Gambar 2. 2 Hirarki Desain Interaksi

User Experience berkonsentrasi pada bagaimana sebuah produk terasa dan apakah itu memecahkan masalah bagi pengguna. UX Design adalah sebuah proses membuat sebuah website atau aplikasi yang dibuat mudah untuk dgunakan serta tidak membingungkan ketika digunakan oleh pengguna. Lima elemen utama dari User Experience Design menurut Jasse James Garret yaitu Strategi, Scope, Structure, Skeleton, dan Surface.

1. Strategi

Hal penting dalam strategi di UX Design adalah **kebutuhan pengguna**. Dari kebutuhan pengguna ini akan diamati hal apa yang pengguna inginkan, yang membuat pengguna bahagia, dan lain sebagainya. Hal ini menjadi pondasi penting karena desain yang dibuat akan percuma apabila tidak sesuai dengan kebutuhan pengguna.

2. Cakupan

Strategi akan menjadi batasan ketika kebutuhan pengguna telah diterjemahkan dalam bentuk requirement. Dua hal yang dilakukan pada elemen cakupan yaitu **menentukan kebutuhan fungsional dan kebutuhan informasi/konten**. Kebutuhan fungsional adalah daftar dari fungsi dari tampilan yang kita kembangkan, sementara kebutuhan konten adalah daftar dari informasi apa saja yang akan ditampilkan dan dalam bentuk apa.

3. Struktur

Pada elemen ini akan dikaji bagaimana pengguna berinteraksi dengan kumpulan fitur yang telah kita definisikan pada elemen cakupan. Kumpulan bentuk konten dalam content requirement kemudian diterjemahkan menjadi **arsiktektur informasi.**

4. Skeleton

Pada tahap ini bentuk dari tampilan yang kita kembangkan sudah mulai terlihat. Skeleton akan memperlihatkan tombol apa akan berada dimana, dilayar mana konten akan ditampilkan, seberapa besar ukuran konten yang ditampilkan, dan beberapa hal lain. UX Designer bertugas untuk mengatur layar agar **semua elemen berada dalam posisi yang tepat dan informasinya dalam proporsi yang baik**. Ditahap ini dilakukan riset untuk memastikan semua komponen sudah memenuhi user needs yang optimal.

5. Permukaan

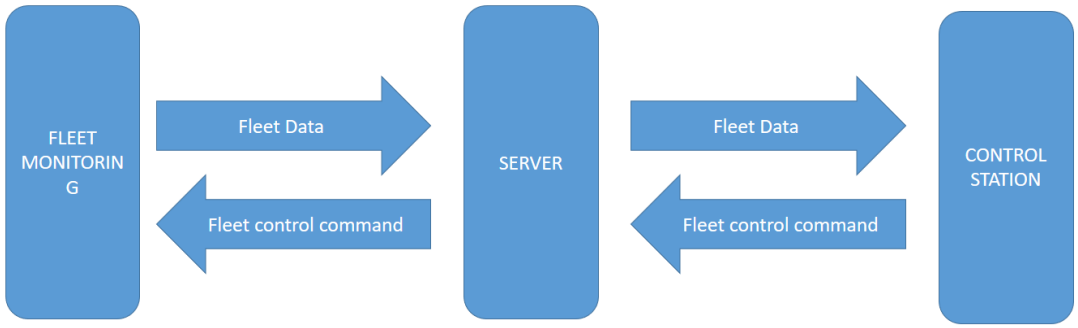
Pada elemen ini desain virtual baru dilaksanakan. Kita bisa melihat apa yang ditampilkan, apa warna dari tiap elemen, bagaimana tiap warna mempengaruhi informasi maupun interaksi dan lain-lain. Di bagian ini **warna, gambar, typography, animasi, efek dan berbagai komponen visual** sudah dikembangkan**.**

BAB III

DESAIN DAN IMPLEMENTASI

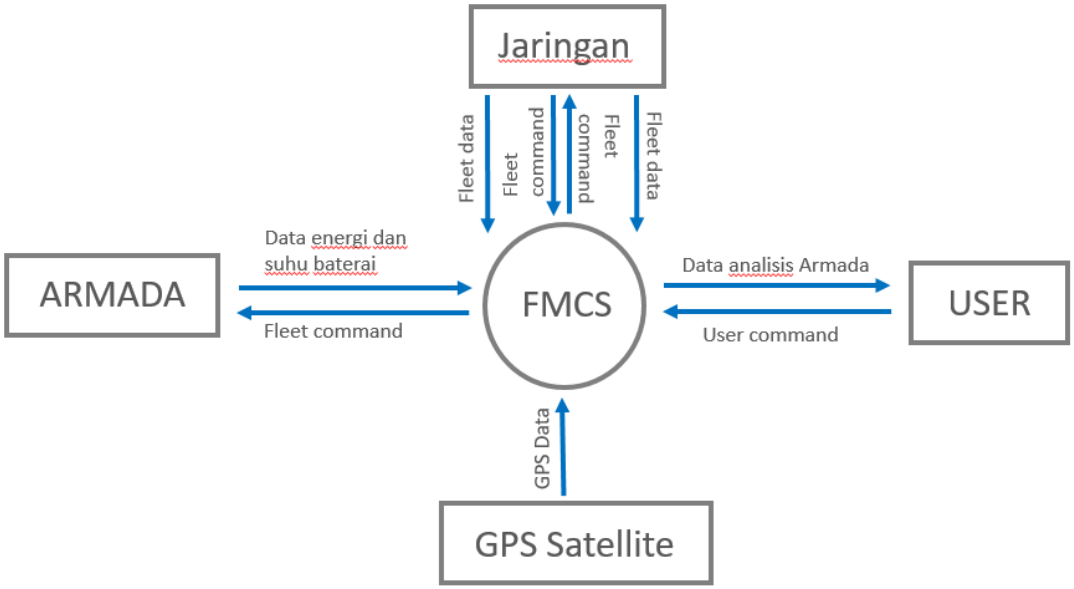
Fleet Monitoring and Control Sistem (FMCS) merupakan sistem pemantauan dan kontrol real-time pada guided bus. Sistem ini akan mengirimkan data dari setiap armada ke control station dengan menggunakan *fleet hardware*. Data yang dikirimkan berupa posisi dan kondisi fisik armada yang kemudian akan diterima dan diolah di GUI. Dengan adanya sistem ini, operator dari *Control Station* dapat mengawasi seluruh pergerakan armada secara realtime.

Sistem FMCS untuk *guided bus* ini terdiri dari tiga subsistem utama, yaitu fleet hardware, server, dan GUI. Ketiga subsistem ini terintegrasi menjadi satu kesatuan. *Fleet hardware* akan diletakkan pada tiap *guided bus*, sementara server dan GUI berada pada ruang *control station. Fleet hardware* dengan server dan server dengan GUI terhubung dengan komunikasi dua arah, karena server bertugas sebagai penghubung antar *fleet hardware* dan GUI.



Gambar 3. 1 Diagram Blok Keseluruhan Sistem

Selama sistem FMCS ini aktif, akan terjadi aliran informasi pada setiap pengiriman data dari *fleet hardware* atau GUI. Skema Data Flow Diagram (DFD) sistem keseluruhan dapat dilihat pada gambar berikut:



Gambar 3. 2 DFD Level 0 Sistem

Skema diatas menggambarkan DFD level 0, yaitu sistem secara keseluruhan, batasan, serta komponen yang terlibat pada FMCS. Pada sistem ini terdapat 4 bagian yang terlibat, yaitu armada, user, cloud dan satelit GPS. User yang dimaksud dalam hal ini adalah operator yang mengoperasikan GUI.

Sistem ini menerima input dari tiga komponen, yaitu armada, satelit GPS dan user. Dari armada, sistem menerima input berupa energi dan suhu baterai. Dari satelit GPS, sistem menerima input berupa data GPS lokasi armada. Dari user, sistem menerima input berupa fleet command.

Secara keseluruhan, skenario penggunaan sistem adalah sebagai berikut:

1. Fleet hardware akan mengirimkan data-data berupa posisi dan kondisi fisik armada selama armada aktif dan berjalan. *Fleet hardware* akan mengirimkan seluruh data dengan menggunakan jaringan GSM dan protokol MQTT.
2. Data diterima oleh server dan diteruskan ke GUI.
3. GUI menerima seluruh data yang dikirim kemudian mengolah data posisi dengan menggunakan algoritma penjadwalan dan memberikan *feedback* berupa kecepatan yang harus dituruti oleh *driver* agar armada sesuai dengan sistem penjadwalan yang telah dittentukan.

4. GUI menampilkan seluruh posisi kendaraan pada peta geografis dan diagramatis, juga menampilkan seluruh data yang diterima dalam satu tabel.
5. *Feedback* kecepatan dikirimkan dengan melalui server untuk disampailkan ke fleet hardware dan kemudian ditampilkan pada *fleet hardware.*

Pada bab ini akan dijabarkan spesifikasi yang harus dipernuhi oleh GUI, proses perancangan GUI, serta implementasi GUI.

3.1 Spesifikasi

Berdasarkan kebutuhan aliran informasi pada sistem yang telah dijabarkan sebelumnya, GUI FMCS harus dapat memenuhi spesifikasi sebagai berikut:

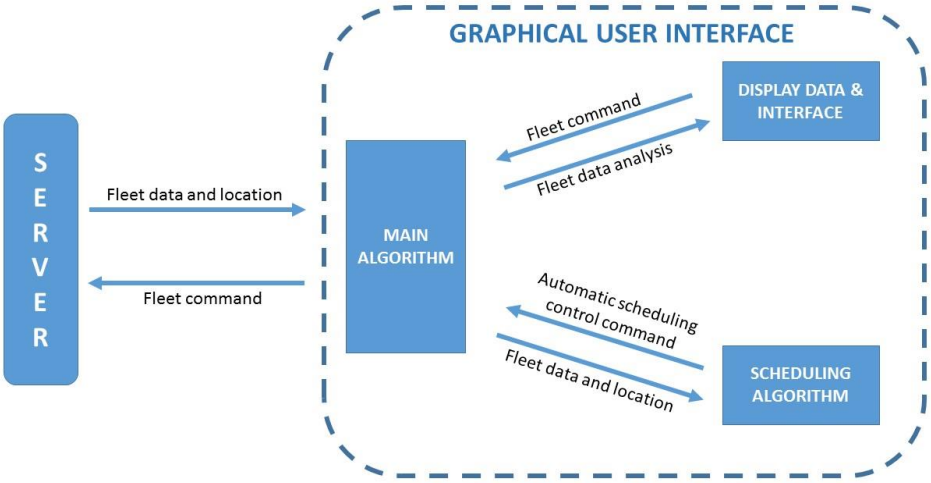
1. Mampu menampilkan data-data berupa posisi, rpm kendaraan, kondisi fault, dan level energi baterai tiap armada,
2. Mampu mengolah data yang diterima,
3. Mampu mengirimkan perintah berupa instruksi kecepatan ke armada

Serta tambahan fitur sebagai berikut:

1. Mudah untuk digunakan (*user friendly*).

3.2 Perancangan

GUI bertugas untuk menerima data berupa data fleet dan posisi dari masing-masing fleet melalui server dan menyampaikannya ke algoritma penjadwalan. Oleh algoritma penjadwalan, data fleet dan lokasi diolah menjadi control command otomatis yang dikembalikan ke algoritma utama. Secara garis besar, diagram blok keseluruhan GUI adalah sebagai berikut :



Gambar 3. 3 Diagram Blok GUI

Pada GUI, terdapat tiga fungsi utama yang digunakan untuk menjalankan FMCS ini, yaitu algoritma utama, display data & interface, serta algoritma penjadwalan.

Gambar 3. 4 Diagram Blok Algoritma Utama

Algoritma utama menerima 3 jenis data. Yang pertama *fleet data and location* dari server, kemudian *scheduling command* dari algoritma penjadwalan, serta *fleet command* dari operator. Semua data ini diolah dalam algoritma utama dan dikeluarkan dalam bentuk *fleet data analysis* dan *fleet data and location* yang akan ditampilkan pada display data, serta *fleet command* yang akan diberikan kepada masing-masing hardware lewat server.

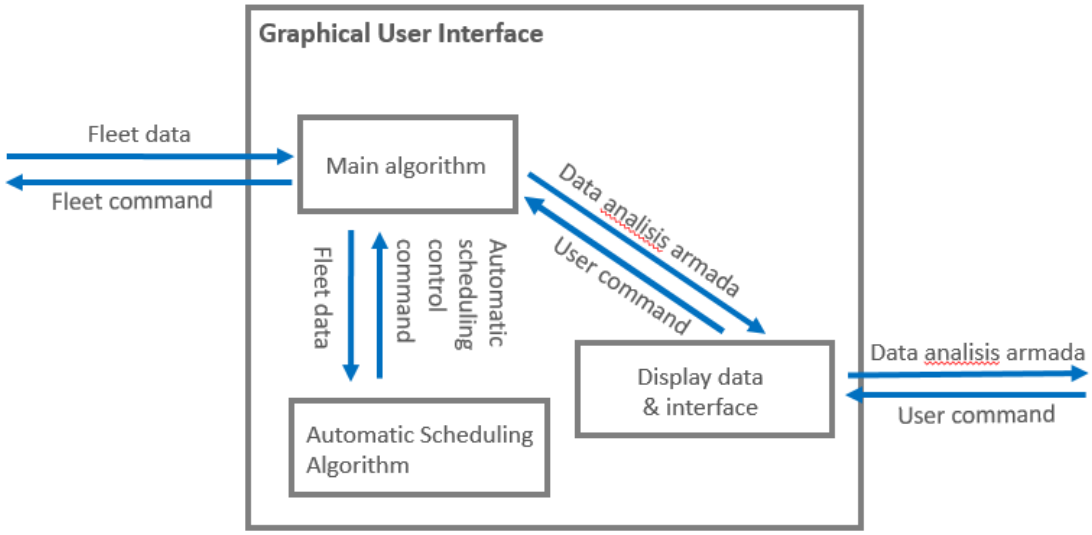
Gambar 3. 5 Diagram Blok Display dan Interface

Display Data & Interface bertugas menyampaikan *fleet command* dari operator ke algoritma utama dan menampilkan *fleet data analysis* yang telah diolah oleh algoritma utama.

Gambar 3. 6 Diagram Blok Algoritma Penjadwalan

Scheduling algorithm akan menerima *fleet data and location* dari server dan bertugas mengirimkan *automatic scheduling control command* ke algoritma utama.

Secara keseluruhan, aliran komunikasi yang terjadi dalam GUI dapat digambarkan pada DFD level 2 berikut:



Gambar 3. 7 DFD Level 2 GUI

Pada subbab selanjutnya, akan dijelaskan secara detail implementasi terhadap interface dan algoritma utama.

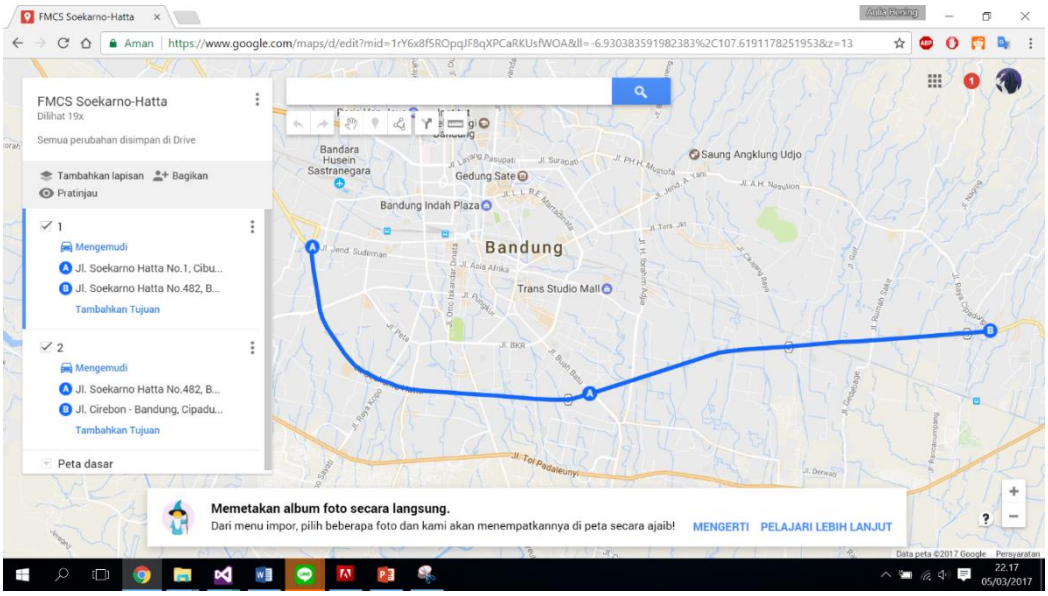
3.3 Implementasi

3.3.1 Tampilan peta geografis

Pada spesifikasi diinginkan data posisi memiliki error maksimal 12 meter. Hal ini menjadi dasar dibuatnya array segmentasi koordinat geografis. Agar error dari GPS dapat diminimalisir, maka dibuat array koordinat disepanjang jalur trayek Elang-Cibiru. Hal ini dibuat untuk menghandle situasi apabila data koordinat yang dikirim GPS meleset, ia akan tetap ditampilkan pada posisi yang benar.

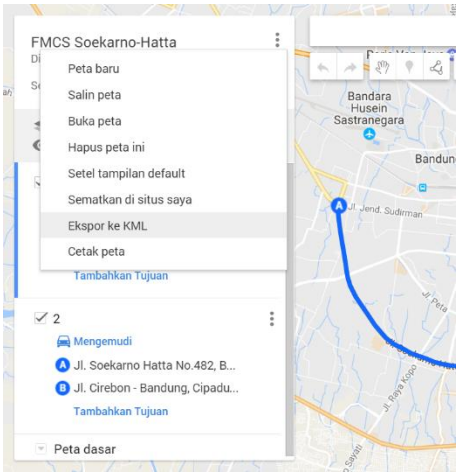
Agar error yang diperoleh maksimal 12 meter, dilakukan pencacahan koordinat setiap 12 meter sepanjang rute Soekarno-Hatta). Adapun untuk melakukan pencacahan, langkah pertama yang dilakukan adalah mencari koordinat garis lurus pada Google Map:

1. Masuk ke https://www.google.com/maps/d dan klik "Buat Peta Baru". Tandai Elang sebagai titik mulai dan Cibiru sebagai titik akhir, rute Soekarno-Hatta telah jadi.



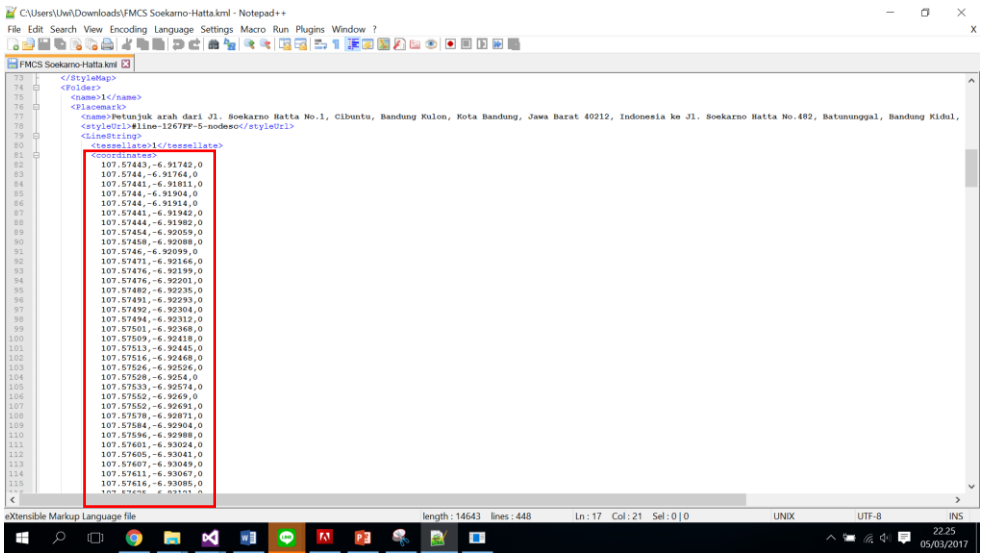
Gambar 3. 8 Tampilan MyMaps Google Map

2. Klik titik tiga di pojok kanan atas sebelah nama peta, dan pilih "Ekspor ke KML". KML merupakan format file yang digunakan untuk menampilkan data geografis pada browser.



Gambar 3. 9 Langkah Ekespor ke KML

3. Setelah mendownload file, buka file di notepad dan akan didapatkan array koordinat yang menunjukkan seluruh titik perubahan garis lurus yang terdapat pada sepanjang rute.



Gambar 3. 10 Array Koordinat Garis Lurus

4. Data koordinat garis lurus ini kemudian disatukan dalam sebuah file .txt yang akan diproses oleh sebuah program pembagi interval dalam bahasa python.

Metode pencarian jarak terdekat antara dua titik koordinat salah satunya adalah menggunakan **Formula Haversine**, yang mempunyai rumus sebagai berikut :

1. $a = \left(\sin\frac{\Delta lat}{2}\right)^2 + \cos(lat1rads) \times \cos(lat2rads) \times \left(\sin\frac{\Delta long}{2}\right)^2$
2. $c = 2 \times atan2\left(\sqrt{a}, \sqrt{1-a}\right)$
3. $d = R \times c$

Dengan $\Delta\, lat$ merupakan selisih antara latitude 1 dan latitude 2, $\Delta\, long$ selisih dari longitude 1 dan longitude 2, dan R jari-jari bumi. Sehingga implementasinya dalam python menjadi sebagai berikut :

```
def getPathLength(lat1,lng1,lat2,lng2):
    '''calculates the distance between two lat, long coordinate pairs'''
    R = 6371000 # radius of earth in m
    lat1rads = math.radians(lat1)
    lat2rads = math.radians(lat2)
    deltaLat = math.radians((lat2-lat1))
    deltaLng = math.radians((lng2-lng1))
    a = math.sin(deltaLat/2) * math.sin(deltaLat/2) + math.cos(lat1rads)
    * math.cos(lat2rads) * math.sin(deltaLng/2) * math.sin(deltaLng/2)
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1-a))
    d = R * c
    return d
```

Yang selanjutnya harus dilakukan adalah mencari azimuth dengan menggunakan bearing. Azimuth merupakan sudut yang terbentuk antara titik utara peta dengan titik sasaran. Ilustrasi :

- 45 derajat azimuth = 45 derajat bearing ke timur dari utara (N 45 E)
- 135 derajat azimuth = 45 derajat bearing ke timur dari selatan (S 45 E)

Karena itu azimuth dapat juga dicari dengan menggunakan bearing dengan rumus berikut :

$$bearing = \left(atan2\left(\Delta\, long, \log\left(\frac{\tan\left(\frac{lat2}{2} + \frac{\pi}{4}\right)}{\tan\left(\frac{lat1}{2} + \frac{\pi}{4}\right)} \right)^{0} + 360 \right) mod\ 360 \right.$$

Yang jika diimplementasikan dalam python adalah sebagai berikut :

```
def calculateBearing(lat1,lng1,lat2,lng2):
    '''calculates the azimuth in degrees from start point to end point'''
    startLat = math.radians(lat1)
    startLong = math.radians(lng1)
    endLat = math.radians(lat2)
    endLong = math.radians(lng2)
    dLong = endLong - startLong
    dPhi = math.log(math.tan(endLat/2.0+math.pi/4.0)/math.tan(startLat/2.0+math.pi/4.0))
    if abs(dLong) > math.pi:
        if dLong > 0.0:
            dLong = -(2.0 * math.pi - dLong)
        else:
            dLong = (2.0 * math.pi + dLong)
    bearing = (math.degrees(math.atan2(dLong, dPhi)) + 360.0) % 360.0;
    return bearing
```

Setelah didapatkan azimuth dan jarak, dicari latitude dan longitude konversi hasil perhitungan dengan menggunakan fungsi getDestinationLatLong() yang terdapat pada line 35 – line 48 di lampiran coordinate_interval.py. Pada algoritma utama, program

akan menerima banyak titik koordinat pada file input jlurus.txt dan mencari koordinat-koordinat sepanjang interval diantara dua titik.

```
if __name__ == "__main__":
    #INI YANG DIGANTI
    #point interval in meters
    interval = 12.0
    #direction of line in degrees
    #start point
    for a in range(0,621):
        if (a%2 == 0):
            lng1 = farray[a]
            lat1 = farray[a+1]
            lng2 = farray[a+2]
            lat2 = farray[a+3]

            azimuth = calculateBearing(lat1,lng1,lat2,lng2)
            coords = main(interval,azimuth,lat1,lng1,lat2,lng2)
            print(coords, file=dest)
```

Karena yang diinginkan adalah koordinat disetiap 12 meter, maka input interval diisi dengan 12.0. Sementara 621 adalah jumlah koordinat yang akan diinput file input.

Hasil akhirnya didapatkan koordinat-koordinat yang diinginkan setiap 12 meter sepanjang trayek Soekarno-Hatta pada file output.

3.3.2 Peta diagramatis

Peta diagramatis (atau banyak disebut *schematic maps*) merupakan peta yang umum digunakan untuk menunjukkan pergerakan kendaraan pada pengguna (baik penumpang maupun operator control station). Pembuatan peta diagramatis bertujuan untuk memudahkan pengguna memantau posisi kendaraan dengan tampilan yang lebih mudah dimengerti, dibandingkan dengan menampilkan pada peta yang sesungguhnya (peta geografis).

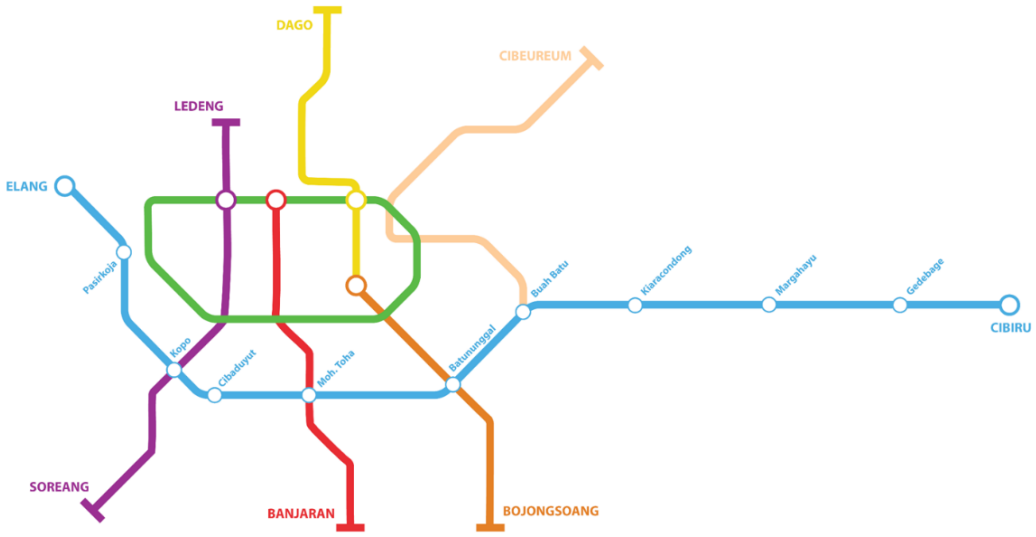
Peta diagramatis ini dibuat dengan cara menghilangkan informasi yang tidak dibutuhkan dan cenderung menggunakan simbol-simbol yang dapat memberi informasi tambahan seperti halte kecil, halte besar, interchange, dan nama halte.

Karena bertujuan untuk memudahkan pengguna memahami trayek, peta diagramatis digambarkan dengan garis lurus. Metode pembuatannya sendiri terbagi dua:

1. Isometric grid (menggunakan kemiringan 30 derajat)
2. Orthogonal method (menggunakan kemiringan 45 derajat)

Pada project ini digunakan tampilan orthogonal karena lebih user-friendly.

Pemodelan peta diagramatis dengan garis lurus ini menyebabkan peta menjadi tidak berskala (karena itu peta diagramatis banyak disebut sebagai peta tanpa skala), karena penarikan garis dilakukan hanya berdasarkan pendekatan dan hanya mempertimbangkan tampilan. Peta diagramatis didesain secara manual dengan menggunakan Adobe Illustrator dan kemudian hasilnya di *import* ke visual studio.



Gambar 3. 11 Peta Diagramatis

Menampilkan marker pada peta diagramatis dilakukan secara manual dengan menggunakan picturebox dan segmentasi point x,y. Proses segmentasi hingga didapatkan koefisien pengali point x dan y dapat dilihat pada tabel berikut:

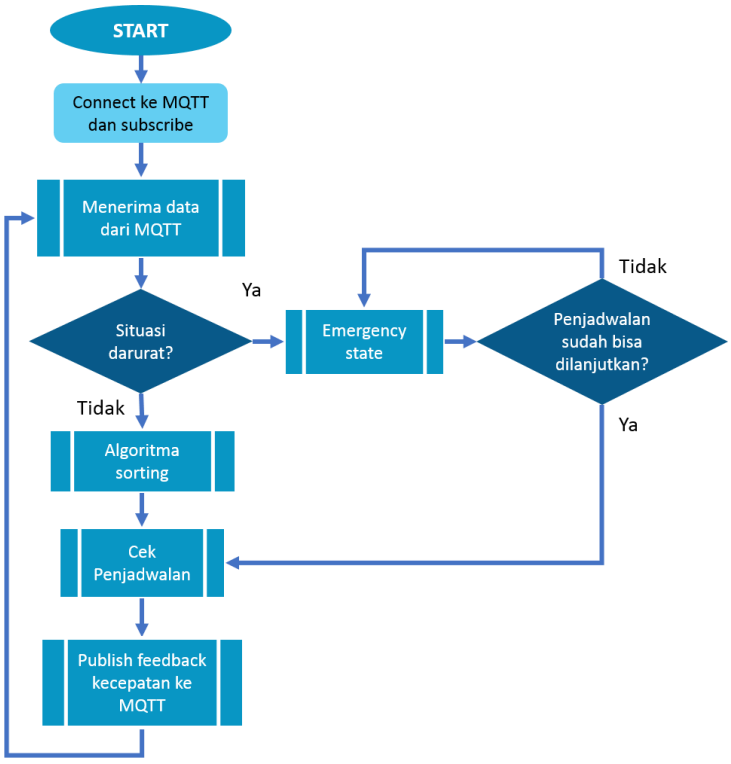
dx	dy	Halte dan Belokan	x	y	dxy	fdxy	xi	fxi	i-ke	kx	ky
		Elang	102	174							
51	52	Pasirkoja	153	226	72,83543094	73	511	511	511	0,099804305	0,10176125
1	62	**Belokan1**	**154**	**288**	62,00806399	62	411,821	**412**	**923**	0,002427184	0,15048544
42	44	Kopo	196	332	60,82697428	61	405,179	405	1328	0,103753086	0,10859259
23	25	**Belokan2**	**219**	**357**	33,97176475	34	163,348	**163**	**1491**	0,140981595	0,15349693
12	0	Cibaduyut	231	357	12	12	57,6522	58	1549	0,206896552	0
82	0	Moh Toha	313	357	82	82	545	545	2094	0,150458716	0
114	0	**Belokan3**	**427**	**357**	114	114	558,684	**559**	**2653**	0,203935599	0
11	-8	Batununggal	438	349	13,60147051	14	54,3165	54	2707	0,203703704	-0,1481481
62	-64	Bubat	500	285	89,10667764	89	289	289	2996	0,214532872	-0,2214533
6	-7	**Belokan4**	**506**	**278**	9,219544457	9	31,3163	**31**	**3027**	0,193548387	-0,2258065
89	0	Kircon	595	278	89	89	309,684	310	3337	0,287096774	0
116	0	Margahayu	711	278	116	116	988	988	4325	0,117408907	0
113	0	Gedebage	824	278	113	113	954	954	5279	0,118448637	0
95	0	Cibiru	919	278	95	95	920	920	6199	0,10326087	0

dx = jarak antar dua point x dy = jarak antar dua point y

x = point patokan
y = point patokan
kx = koefisien pengali x
ky = koefisien pengali y
xi = range array yang ditampung antar dua titik
fxi = range array antar dua titik yang sudah dibulatkan
dxy = jarak antara dua titik
fdxy = jarak antar dua titik yang dibulatkan

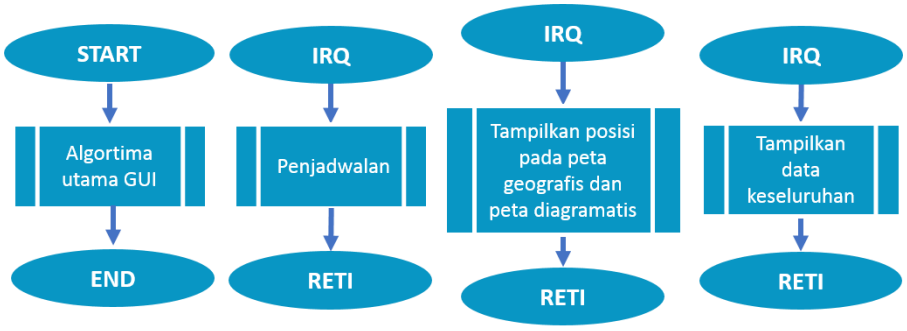
3.3.3 Algoritma utama GUI

GUI terdiri dari satu algoritma utama yang tersusun dari beberapa fungsi dan metode didalamnya. Algoritma utama merupakan seluruh kesatuan integrasi dari penerimaan data, ekstraksi data, pengolahan data dengan jadwal penjadwalan, menampilkan data yang telah diolah, hingga pengiriman *feedback* kepada *fleet hardware*. Satu sistem besar GUI dapat digambarkan dalam flowchart berikut:



Gambar 3. 12 Flowchart Algoritma Utama GUI

Dari satu proses utama tersebut, terdapat beberapa IRQ (Interrupt ReQuest) dalam bentuk *timer* yang akan menginterrupt proses utama jika dipanggil. Ketiga IRQ itu diantaranya dalah algoritma penjadwalan, algoritma display posisi dan algoritma display data.



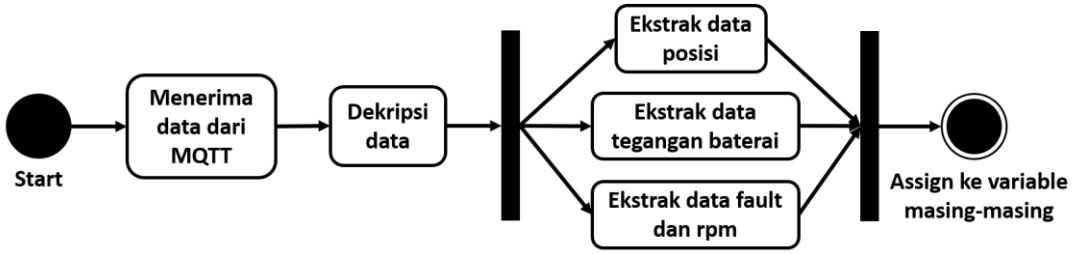
Gambar 3. 13 Keseluruhan Sistem GUI

Berikut ini akan dijabarkan lebih lanjut mengenai subproses ekstrak data dari MQTT, algoritma sorting, penanganan situasi darurat, display posisi dan display data keseluruhan, juga pengiriman komando ke *fleet hardware*.

3.3.3.1 Mengekstrak Data dari MQTT

GUI terhubung dengan MQTT menggunakan metode *subscribe*. *Subscribe* merupakan metode untuk berlangganan pada suatu topik tertentu, sehingga setiap kalinya ada client yang melakukan *publish* pada suatu topik yang sama, client yang melakukan *subscribe* akan menerima pesan tersebut.

Penerimaan hasil *publish* dihandle oleh fungsi `clientSub.MqttMsgPublishReceived` yang memiliki diagram aktivitas sebagai berikut:



Gambar 3. 14 Diagram Aktivitas Penerimaan data dari MQTT

Data yang telah didekrip kemudian diklasifikasikan sesuai jenis informasinya masing-masing, diantaranya:

1. Data posisi berupa koordinat (longitude, latitude)
2. Data tegangan baterai yang akan diolah menjadi persentase level baterai,
3. Data fault yang akan diterjemahkan menjadi informasi fault pada kelistrikan,
4. Data rpm yang akan dikonversi menjadi kecepatan armada.

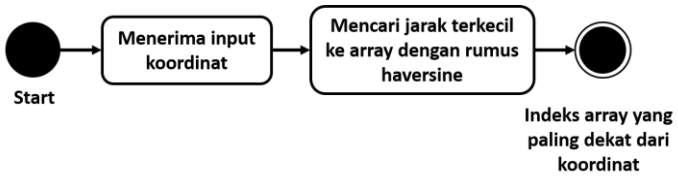
Data-data tersebut kemudian di assign ke variable dalam array yang telah disiapkan.

3.3.3.2 Algoritma Sorting

Sistem segmentasi pada geografis diimplementasikan dengan cara mencari posisi terdekat dalam array dari koordinat yang dikirimkan. Algoritma sorting akan menghitung jarak dari masing-masing koordinat dalam array ke koordinat yang diterima dari MQTT, kemudian mencari yang jaraknya paling kecil, yang artinya posisinya paling dekat. Data yang diterima kemudian dibulatkan ke elemen array dengan jarak terkecil.

Pengolahan algoritma sorting dibantu dengan fungsi haversine yang akan menghasilkan perhitungan jarak dari masing-masing array koordinat ke koordinat MQTT yang diterima.

Detail diagram aktivitas algoritma sorting adalah sebagai berikut:

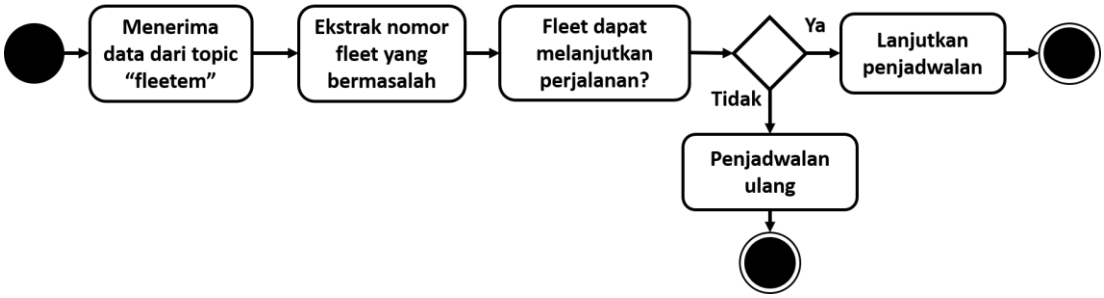


Gambar 3. 15 Diagram Aktivitas Algoritma Sorting

Pada algoritma sorting ini, digunakan fungsi haversine yang berbeda dengan yang digunakan untuk melakukan segmentasi array. Fungsi haversine disini tidak memperhitungkan *azimuth* dan *bearing.* Hal ini disebabkan karena algoritma sorting hanya berkisar satuan meter sehingga pengaruh kelengkungan bumi dapat diabaikan.

3.3.3.3 Penanganan Situasi Darurat

Pada GUI disediakan algoritma untuk mengatasi situasi darurat. Pada kasus ini, *fleet hardware* yang mengalami masalah akan mengirim nomor fleet mereka ke topic “fleetem” ke MQTT. Ketika GUI menerima topik ini, otomatis setiap armada akan diinstruksikan untuk berhenti ke halte terdekat didepannya dan sistem FMCS dihentikan sementara. Berikut diagram aktivitas dari algoritma emergency:

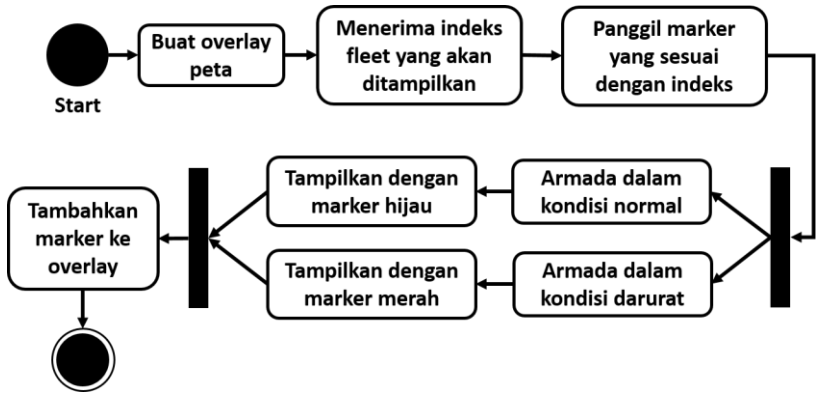


Gambar 3. 16 Diagram Aktivitas Situasi Darurat

Tugas operator kemudian memastikan apakah fleet yang bermasalah dapat melanjutkan perjalanannya atau tidak. Jika iya, penjadwalan tetap dilakukan untuk jumlah fleet yang sama, sementara jika tidak, penjadwalan akan diatur ulang untuk jumlah fleet n-1.

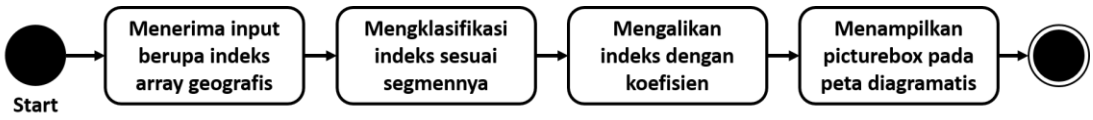
3.3.3.4 Menampilkan posisi dan data armada

Layar utama dari GUI FMCS adalah halaman dengan peta geografis. Halaman ini akan menampilkan seluruh posisi armada secara bersamaan dalam satu screen. Pada halaman ini disediakan fitur berupa marker penanda, apabila ada fleet yang mengalami situasi darurat, maka warna markernya akan berubah menjadi merah, sementara dalam kondisi normal akan berwarna hijau. Alur untuk menampilkan marker pada peta geografis adalah sebagai berikut:



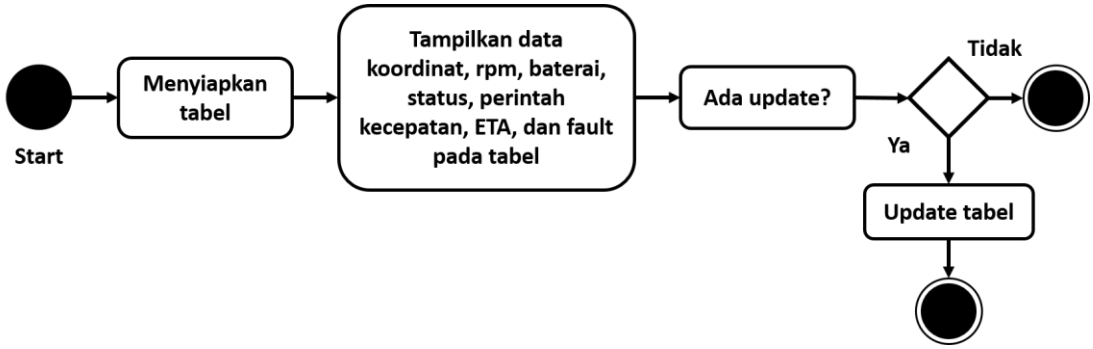
Gambar 3. 17 Diagram Aktivitas Display Marker Geografis

Sementara halaman selanjutnya menunjukkan tampilan dalam peta diagramatis. Proses pada peta diagramatis ini berkesinambungan dengan peta geografis, yaitu menerima input berupa indeks array geografis. Diagram aktivitasnya adalah sebagai berikut:



Gambar 3. 18 Diagram Aktivitas Display Marker Diagramatis

Selain dapat menampilkan seluruh posisi secara bersamaan, GUI juga dapat menampilkan data-data yang dibutuhkan. Data yang ditampilkan diantaranya adalah data koordinat, rpm, baterai, status, kecepatan penjadwalan, ETA, serta fault. Data yang ditampilkan pada tabel ini akan diupdate setiap ada update. Diagram aktivitasnya adalah sebagai berikut:

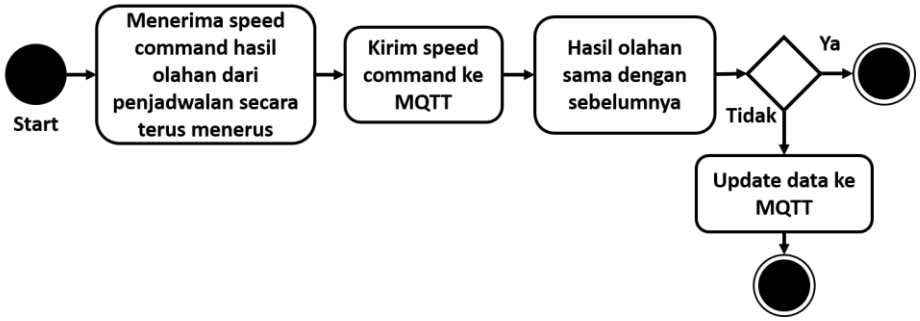


Gambar 3. 19 Diagram Aktivitas Pengupdatean Tabel

Tabel data keseluruhan ini dibuat agar selain operator dapat memantau data fleet secara personal (dengan mengklik marker di peta geografis), operator juga dapat memantau seluruh data fleet secara bersamaan.

3.3.3.5 Memberi komando kecepatan

Seperti yang telah dijelaskan, salah satu spesifikasi GUI ialah dapat mengirimkan feedback berupa instruksi kecepatan pada *fleet hardware.* Setelah menerima data yang dikirim dari MQTT, GUI akan mengolah data tersebut pada algoritma penjadwalan dengan output berupa komando kecepatan. GUI kemudian bertugas untuk mengirim balik komando kecepatan ke fleet hardware setiap ada update instruksi kecepatan yang harus dituruti oleh driver. Diagram aktivitasnya adalah sebagai berikut:

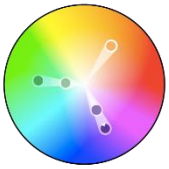


Gambar 3. 20 Diagram Aktivitas Pengiriman Komando Kecepatan

Jika instruksi kecepatan tetap sama seperti sebelumnya (misal, konstan di 86,4 km/jam, maka GUI hanya akan mengirim sekali kecepatan tersebut hingga kecepatan tersebut tidak lagi 86,4 km/jam).

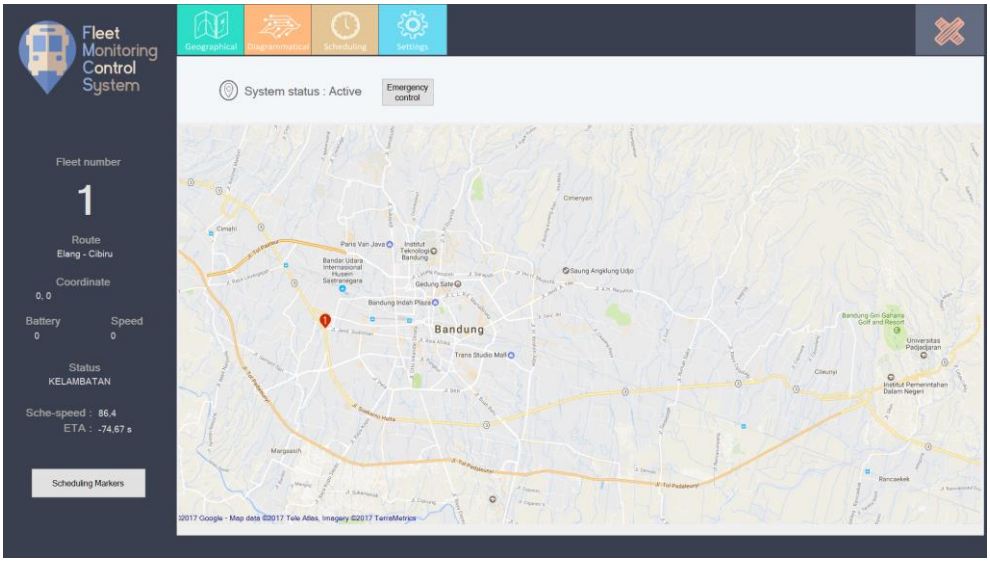
3.4.4 Tampilan GUI

Tampilan GUI dibuat dengan **skema warna triadic**. Skema warna triadic membawa kontras skema warna yang tinggi dengan mempertahankan nada yang sama. Pembuatan skema triadic dibuat dengan memilih tiga warna dengan garis dengan jari-jari yang sama dalam roda warna. Pada skema warna GUI ini dipilih dengan jari-jari 65% dari roda warna agar warna yang ditampilkan dapat membawa kesan **segar, bersemangat dan tidak menjemukan.**



Gambar 3. 21 Roda Warna dengan Jari-jari 65%

Default screen pada GUI diletakkan pada peta geografis karena menampilkan posisi armada pada peta merupakan inti dari pembuatan GUI ini. Pada bar disebelah kiri, diberikan informasi mengenai data personal GUI. *Mousehover* pada masing-masing marker akan memunculkan *tooltip* mengenai nomor marker, dan *mouseclick* pada masing-masing marker akan mengubah informasi pada sidebar tergantung dari marker yang di klik. Misalkan operator melakukan klik pada marker fleet 4, maka sidebar akan menampilkan informasi mengenai fleet 4.



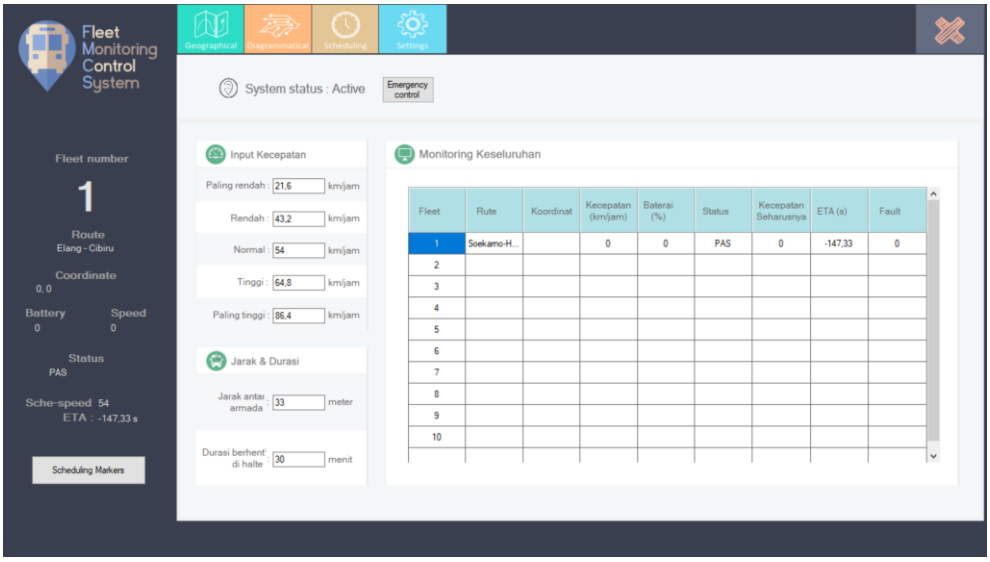
Gambar 3. 22 Tampilan Halaman Peta Geografis

Tombol *scheduling marker* pada bagian bawah sidebar dapat menampilkan seluruh marker dari range penjadwalan yang harus dituruti oleh armada. Karena itu scheduling marker ini ditempatkan sebagai *toggle button* agar marker penjadwalan tidak memenuhi tampilan peta geografis.

Pada halaman kedua terdapat peta diagramtis. Sesuai yang telah diterangkan pada subbab “Peta Diagramatis”, peta ini dibuat pada Adobe Illustrator. Pemilihan warna kontras pada penyusunan peta diagramtis ini bertujuan agar operator dapat melihat dengan jelas perbedaan satu rute dengan rute lainnya. Pemilihan skema warna untuk *metro map* ini telah banyak digunakan pada *metro map* umumnya.

Gambar 3. 23 Halaman Tampilan Peta Diagramatis

Pada halaman scheduling, terdapat 1 kolom pengaturan penjadwalan dan 1 monitor keseluruhan. Pada kolom sebelah kiri, operator dapat mengubah input kecepatan penjadwalan, jarak antar bus, dan durasi berhenti disetiap halte. Pada dasarnya angka-angka ini sudah ditentukan oleh sistem, namun operator dapat merubahnya jika dibutuhkan. Sementara pada kolom sebelah kanan, terdapat monitoring keseluruhan, disini operator dapat mengamati seluruh data yang diterima dari server pada tabel.



Gambar 3. 24 Tampilan Halaman Penjadwalan

Pada panel utama (disebelah status sistem), terdapat tombol emergency control. Jika terjadi situasi darurat, maka tombol ini akan berwarna merah. Operator harus melakukan konfirmasi keberlanjutan penjadwalan fleet dengan tombol ini.

BAB IV

HASIL PENGUJIAN

4.1 Pengujian Kemampuan Fungsional GUI

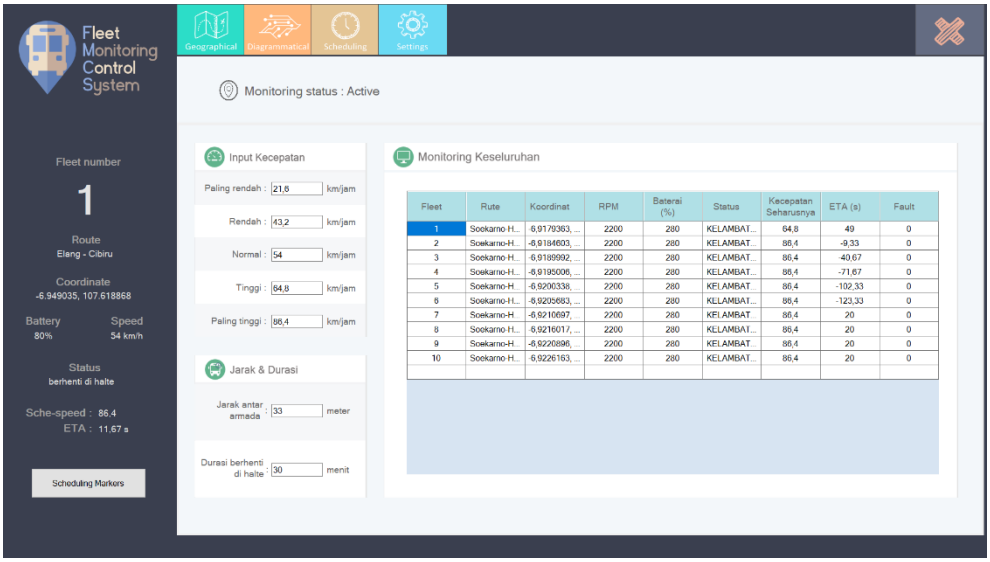
Pada pengujian ini, lingkup spesifikasi yang akan diuji adalah spesifikasi nomor 4, yaitu menampilkan 10 armada dan data-data tiap armada yang akan ditampilkan di GUI pada *control station*.

Akan ditampilkan posisi dari 40 armada pada peta geografis dan 10 data lengkap armada pada tabel monitoring. Data-data tersebut adalah koordinat tiap armada, rpm tiap armada, kondisi fault tiap armada, dan level energi baterai tiap armada. GUI ini akan dipantau dan dikendalikan oleh operator di *control station*.

Syarat yang harus dipenuhi untuk pengujian ini yaitu koneksi internet yang cukup baik pada *fleet hardware*, server, dan PC yang digunakan untuk menampilkan GUI. Selain itu pada PC yang digunakan untuk menampilkan GUI tidak dijalankan aplikasi lain selain FMCS ini. Prosedur yang akan dilakukan pada proses pengujian ini adalah:

- Mengaktifkan server dan memulai GUI.
- Mengaktifkan aplikasi dummy data
- Dikirimkan 10 data koordinat, fault, dan rpm dari 10 armada secara bersamaan
- Mengamati kemampuan GUI menampilkan data lengkap dari 10 armada secara bersamaan pada tabel monitoring.
- Dikirimkan 40 data posisi dari aplikasi *dummy data* secara bersamaan yang terus bergerak setiap waktunya di sepanjang trayek Soekarno-Hatta.
- Mengamati kemampuan GUI menampilkan posisi dari 40 armada secara bersamaan pada peta geografis.

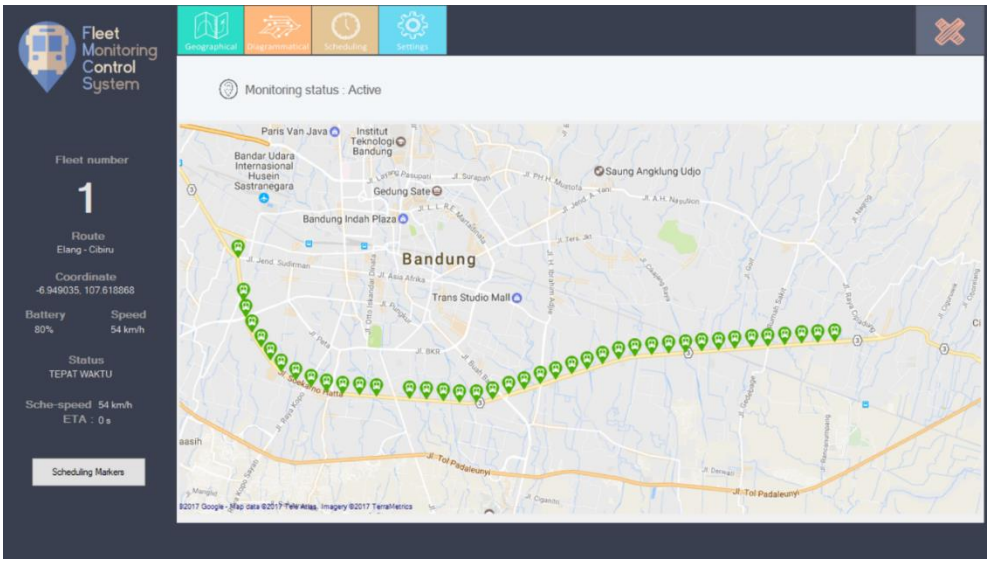
Pengujian tahap pertama dilakukan dengan mengirimkan 10 data lengkap armada berupa koordinat, rpm, fault, dan persentase baterai dari dummy data. Status armada, kecepatan seharusnya, dan ETA merupakan hasil olahan dari algoritma penjadwalan yang telah diintegrasikan dengan GUI. Hasil pengujian adalah sebagai berikut :



Gambar 4. 1 Hasil Pengujian Menampilkan 10 Data Lengkap

Dari pengamatan ketika dilakukan pengiriman data, GUI dapat menampilkan seluruh data dengan baik tanpa menyebabkan aplikasi *lag/error*.

Pengujian tahap kedua dilakukan dengan mengirimkan data posisi 40 armada dari dummy data, dan kemudian ditampilkan pada peta geografis. Hasil pengujian adalah sebagai berikut:



Gambar 4. 2 Hasil Pengujian Menampilkan 40 Posisi Armada

Dari pengamatan yang dilakukan ketika pengujian, dapat diamati bahwa GUI sudah mampu menampilkan posisi 40 kendaraan yang sedang bergerak secara bersamaan tanpa menyebabkan aplikasi *lag/error*.

4.2 Pengujian UX dan UI Design

Pada pengujian ini, lingkup spesifikasi yang akan diuji adalah fitur FMCS, yaitu aplikasi memiliki UX Design yang user-friendly bagi pengguna. Subjek penelitian akan diberikan penjelasan singkat mengenai GUI kemudian mengoperasikannya. Hasil pengujian akan didapat dari kuesioner yang diberikan kepada subjek setelah ujicoba operasi GUI dilakukan.

Pengujian ini dilakukan pada spesifikasi PC sebagai berikut:

- GUI pada laptop Intel i7-6500U 2.5 GHz, RAM 4 GB, OS Windows 10, layar 14 inch dengan resolusi 1920x1080 pixel, dan GPU NVIDIA GeForce 940M.

Syarat yang digunakan pada pengujian ini adalah subjek penelitian sebelumnya tidak pernah mengetahui cara kerja dan sistem GUI FMCS. Sementara asumsi yang digunakan adalah subjek mengerti bahasa inggris sederhana, mengenal bahasa operasi sederhana dan mampu mengoperasikan PC.

Pengujian pertama merupakan pengujian UX Design, dilakukan secara langsung kepada 12 orang subjek.

- Subjek akan diterangkan tentang sistem dan GUI secara umum.
- Subjek diminta untuk meng-*explore* GUI selama 2 menit.
- Subjek diberikan beberapa *task* acak untuk dilakukan, yaitu:
  - Masuk ke halaman peta diagramatis,
  - Masuk ke halaman penjadwalan,
  - Menampilkan data personal armada,
  - Menampilkan data seluruh armada,
  - Keluar dari aplikasi.
- Subjek diberikan kuesioner mengenai tampilan GUI.

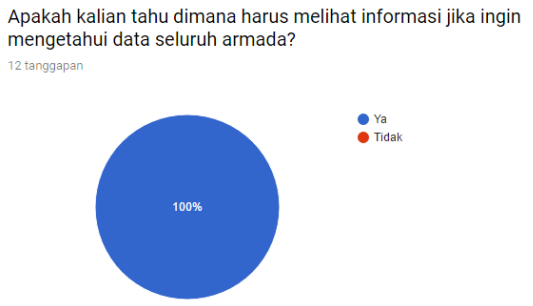
Pengujian kedua merupakan pengujian UI Design. Pengujian ini dilakukan secara virtual kepada 11 orang subjek hanya dengan memperlihatkan screenshot. Penilaian untuk UX Design akan ditanyakan hanya kepada subjek pengujian langsung, sementara penilaian untuk UI Design akan ditanyakan kepada seluruh subjek percobaan.

Pada pengujian ini akan diperoleh data dari kuesioner yang diisi subjek setelah mengoperasikan GUI. Data yang akan diamati adalah:

- Ketersampaian konten/informasi dari UX Design
- Keefektifan User Interface
- Kenyamanan user dalam menggunakan GUI
- Estetika tampilan User Interface

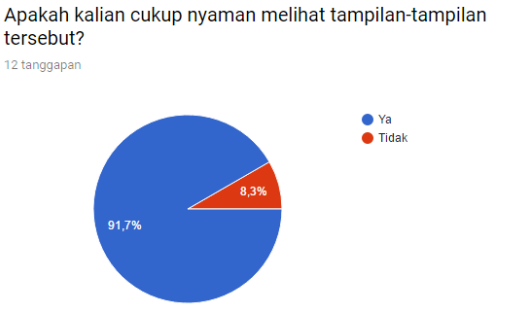
Karena data diambil dengan menggunakan kuesioner, maka hasil akhir akan berupa nilai kuantitatif dalam bentuk persentase.

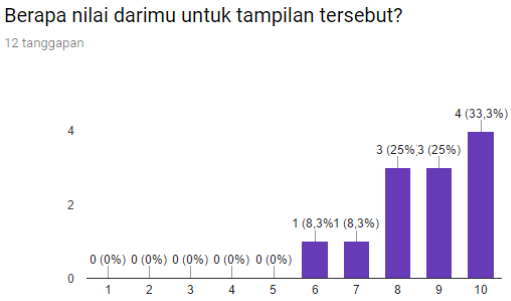
Pengujian dilakukan terhadap 12 subjek secara langsung. Penulis memberi penjelasan singkat tentang GUI dan memberi waktu bagi subjek untuk menjelajahi GUI kemudian meminta subjek untuk mengisi kuesioner. Hasil kuesionernya adalah sebagai berikut:



Dengan kalkulasi 5x12 pertanyaan dan 2 diantaranya tidak dapat dijawab oleh user, dapat disimpulkan bahwa 96,66% informasi/konten tersampaikan pada user.

Selain itu subjek diminta untuk memberikan pendapat dan nilai terhadap UI Design yang telah dibuat dan hasilnya adalah sebagai berikut:





Dari hasil kuesioner dapat disimpulkan 91.7% subjek cukup nyaman dengan tampilan GUI, 91.7% mudah memahami GUI dan 100% setuju bahwa GUI cukup efektif.

Pengujian kedua dilakukan terhadap 11 subjek secara virtual. Penulis memberi penjelasan singkat tentang GUI pada kuesioner dan meminta subjek hanya menilai tampilannya saja. Hasilnya adalah sebagai berikut:

Dari hasil kuesioner dapat disimpulkan 90.9% subjek cukup nyaman dengan tampilan GUI, 90.9% mudah memahami GUI dan 100% setuju bahwa GUI cukup efektif.

Dari seluruh hasil pengujian terhadap UX dan UI Design, dengan nilai rata-rata diatas 90% dapat disimpulkan GUI sudah cukup user-friendly bagi pengguna.

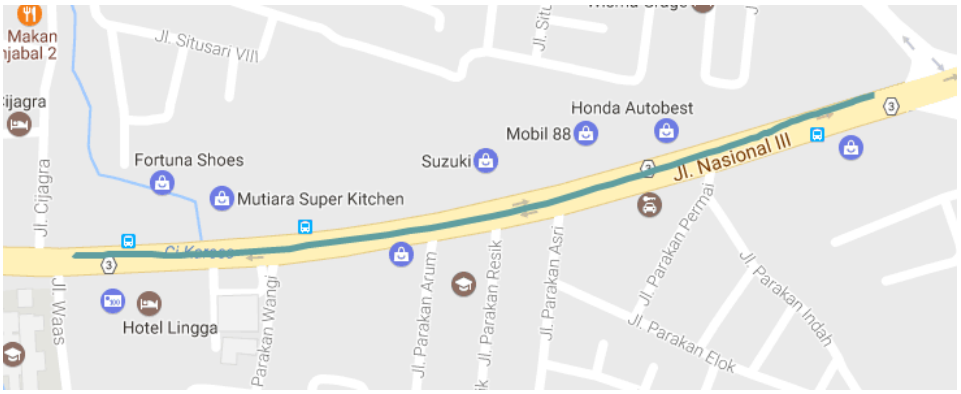
4.3 Pengujian di Lapangan

Pengujian dilakukan pada rute Soekarno-Hatta dengan segmen dimulai dari perempatan Jalan Cijagra hingga Perempatan Buah Batu. Test dilakukan dengan cara membawa *fleet hardware* dengan menggunakan motor. Data yang dikirimkan berupa data koordinat GPS.

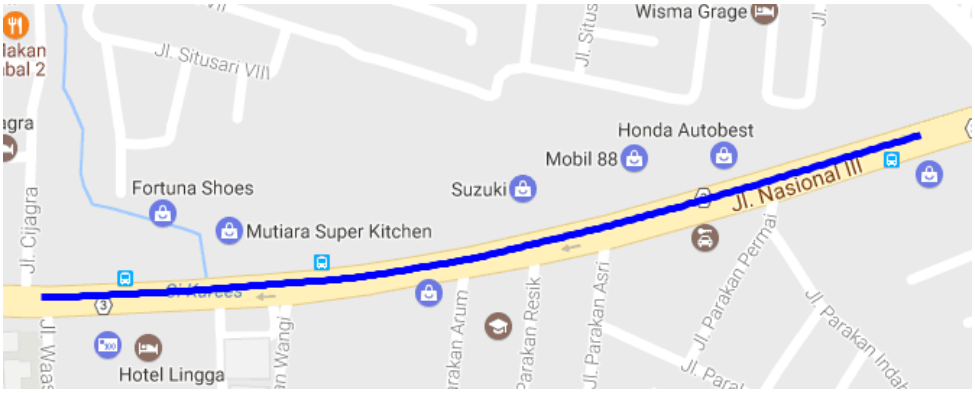
Gambar 4. 3 Hasil Pengujian Lapangan

Dari gambar diatas, dapat disimpulkan GUI sudah dapat menampilkan marker fleet 1 beserta range penjadwalannya. Selain itu data-data personal fleet juga sudah dapat ditampilkan pada sidebar.

Pada pengujian ini juga dilakukan dua perekaman rute. Yang pertama adalah rute yang ditampilkan pada GUI dan yang kedua rute yang diplot dari hasil pengiriman koordinat GPS salah satu anggota tim. Hasil pengujian adalah sebagai berikut:



Gambar 4. 4 Plot Rute Koordinat GPS HP



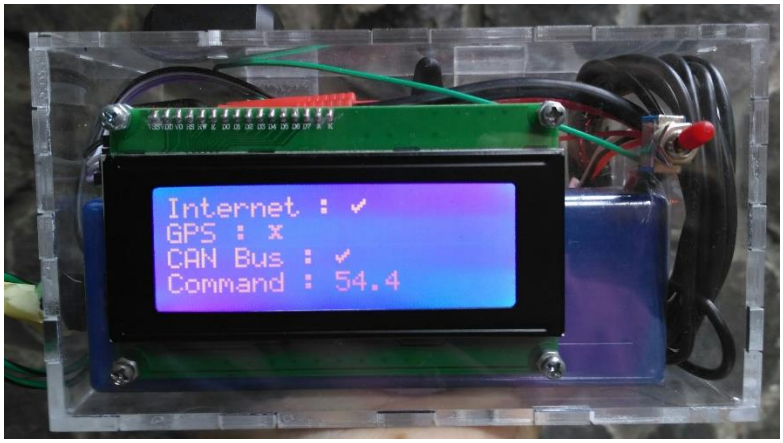
Gambar 4. 5 Plot Rute yang Ditampilkan GUI

Dari perbandingan kedua rute diatas, dapat disimpulkan GUI sudah dapat menghandle simpangan yang dikirim GPS. Meskipun terdapat penyimpangan pada peta geografis, lokasi yang ditampilkan tetap berada pada lajur tengah rute seharusnya.

4.4 Pengujian Pengiriman Komando Kecepatan

Pengujian dilakukan bersamaan dengan pengujian di lapangan. Test dilakukan dengan cara membawa *fleet hardware* dengan menggunakan motor. Data yang dikirimkan *fleet hardware* berupa data posisi akan diolah oleh GUI algoritma penjadwalan. GUI

kemudian mengirimkan instruksi kecepatan setiap kali terdapat perubahan pada instruksi kecepatan. Hasilnya *fleet hardware* sudah dapat menerima instruksi kecepatan dari GUI.



BAB V

KESIMPULAN & SARAN

5. 1 Kesimpulan

Dari hasil pengujian yang telah dilakukan, GUI FMCS sudah dapat menampilkan 40 posisi fleet secara bersamaan dan 10 data lengkap armada berupa koordinat, rpm, fault, dan persentase baterai secara bersamaan dan tanpa menyebabkan lag/error pada PC. Selain itu GUI FMCS juga memiliki UX dan UI Design yang sudah memenuhi *user needs* sehingga dapat disimpulkan GUI FMCS memiliki tampilan yang cukup *user friendly*.

5. 2 Saran

Pada segmentasi di peta geografis GUI, ketelitiannya tidak perlu hingga mencapai 12 meter. Dengan memperhitungkan panjang bis 12 meter dan bis sedang melakukan pergerakan, ketelitian dapat dikurangi hingga 15 meter (12 meter panjang bis + 6 meter setengah panjang bis). Array diusahakan untuk dibuat seminimal mungkin agar proses kalkulasi lebih cepat dan sistem semakin ringan.

DAFTAR PUSTAKA

Ardisasmita, Adam. 2015. “Lima Elemen Dari User Experience Design”. https://ardisaz.com/2015/03/24/lima-elemen-dari-user-experience-design, 24 Maret 2015.

Wikipedia. 2016. “Microsoft Visual Studio”. https://id.wikipedia.org/wiki/Microsoft_Visual_Studio, 20 Oktober 2016.

Filus, Teo. 2017. “Pengenalan Bahasa Pemrograman C#”. https://www.codepolitan.com/pengenalan-bahasa-pemrograman-c-587effa1cb95b, 18 Januari 2017.

Amaliah, Fatiha. 2013. “Pengertian & Konsep Oop (Object Oriented Programming)”. https://fatihamaliah.wordpress.com/2013/04/02/pengertian-konsep-oop-object-oriented-programming/, 2 April 2013.

Multimedia Indonesia. 2016. “Teori Warna dalam Desain”. http://www.multimediaindonesia.net/2016/03/teori-warna-dalam-desain.html, Maret 2016.