ABSTRAK

Chanafi, Imam. 2010. *Aplikasi Perhitungan Pembangunan Jalan Berbasis Java Menggunakan Global Positioning System*. Skripsi, Jurusan Elektro Fakultas Teknik Universitas Brawijaya. Pembimbing (I) M. Aswin Ir. MT.; Pembimbing (II) Moch. Rif'an ST. MT.

Kata Kunci : Aplikasi, Pembangunan Jalan, Java, GPS

Indonesia merupakan salah satu dari negara yang sedang berkembang, dimana pembangunan terjadi hampir pada semua sektor kehidupan masyarakat. Pelaksanaan pembangunan tidak hanya berpusat pada wilayah kota saja, namun juga ke daerah-daerah secara merata. Bidang konstruksi merupakan salah satu sektor yang sangat menunjang pembangunan nasional, hasil-hasil dari bidang konstruksi ini sangat dirasakan manfaatnya oleh masyrakat, misalnya pembangunan sarana dan prasarana masyarakat terutama pada sarana transportasi dan perhubungan.

Salah satu bentuk pembangunan sarana transportasi adalah perencanaan dan pembangunan jalan, dimana di dalamnya terdapat tiga proses utama yakni perencanaan titik stasiun, perencanaan bentuk *alignment* lengkung jalan, dan perencanaan volume pekerjaan secara keseluruhan. Bila perencanaan dan pembangunan jalan diperhitungkan secara manual maka akan memakan waktu yang cukup lama dan memungkinkan terjadinya penurunan tingkat kejelian, karena jika kedua hal tersebut terjadi maka akan menyebabkan keterlambatan dalam tahap pembangunannya dan lebih fatal lagi menyebabkan terjadinya kesalahan dalam perhitungan yang sangat berdampak pada tingkat keamanan pengemudi, serta pembengkakan dalam pendanaannya.

Guna mendukung keberhasilan dalam tahapan perencanaan jalan maka dikembangkan suatu sistem aplikasi yang dapat menggantikan manusia dalam melakukan perhitungan dan perencanaannya. Pada sistem ini akan mempergunakan *device bluetooth* GPS sebagai masukan dari sistem, dari data streaming *device* tersebut nantinya dapat ditentukan titik-titik stasiunnya, diperhitungkan *alignment-alignment* lengkung jalannya, dan pada tahapan akhirnya diperhitungkan volume pekerjaannya. Setelah semua tahapan selesai maka hasil perhitungannya akan ditampilkan kepada user dalam bentuk map dan laporan-laporan berformat html.

KATA PENGANTAR

Puji syukur Alhamdulillah penulis panjatkan ke hadirat ALLAH SWT, atas segala rahmat taufik dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Aplikasi Perhitungan Pembangunan Jalan Berbasis Java Menggunakan *Global Positioning Sistem*".

Penyusunan skripsi ini dimaksudkan untuk memenuhi salah satu persyaratan untuk memperoleh gelar Sarjana Teknik dalam program S1 Teknik Elektro Fakultas Teknik Universitas Brawijaya.

Keberhasilan penyusunan skripsi ini tentu tidak lepas dari bantuan, motivasi, bimbingan dan kerjasama yang baik dari berbagai pihak. Untuk itu, penulis menyampaikan terima kasih kepada :

1. Bapak M. Aswin, Ir, MT selaku Dosen Pembimbing I yang telah memberikan bimbingan, arahan, serta motivasi selama penyusunan skripsi ini.
2. Bapak Moch. Rif'an ST, MT selaku Dosen Pembimbing II juga telah memberikan bimbingan, arahan, serta motivasi selama penyusunan skripsi ini.
3. Segenap Bapak/Ibu dosen Jurusan Teknik Elektro Fakultas Teknik Brawijaya yang telah memberikan bimbingan, arahan, serta motivasi selama kuliah.
4. Ayah ibu dan keluarga tercinta yang selalu memberikan cinta kasih yang tulus, doa yang tidak pernah putus, semangat dan dorongan baik secara spiritual maupun material kepada penulis.
5. Bapak Soleh dan Bapak Nanang selaku staf Dinas Bina Marga yang berkenan untuk memberikan informasi kepada penulis.

6. Bapak Hadi Sucipto, selaku direktur utama CV. Jaya Utama yang bersedia meluangkan segenap waktu dan ilmunya kepada penulis mengenai dunia perkonstruksian di Kabupaten Pasuruan.
7. Pendamping setiaku, Mayang atas ketulusannya dalam menemani, memotivasi dan memanjatkan doa tanpa henti kepada penulis.
8. Sahabat-sahabat seperjuangan selama penulis menempuh bangku perkuliahan, Nanda, Anang, Rizal, Putu, Triaji, Riza, Tulus, Jatayu, willy.
9. Keluarga besar Serang 14, Anam, Dina, Ecox, Humble, Memed, Fau, Tomo, Bebex, Galih.
10. Mas Roy dan Ibu kos, yang telah memberikan tempat berteduh selama penulis menempuh kuliah.
11. Sahabat-sahabat pendamping setiaku, yang memberikan dorongan untuk segera lulus dan mencari pekerjaan, Astrid, Tika, Rendra.
12. Sahabat-sahabat SMAN I Pasuruan yang selalu memberikan semangat dan kebersamaan selama ini, Emon, Gembel, Avi, Ndower, dkk.
13. Semua teman-teman mahasiswa Fakultas Teknik Elektro Universitas Brawijaya Malang.

Malang, 30 November 2010

Penulis

DAFTAR ISI

ABSTRAK i

KATA PENGANTAR ii

DAFTAR ISI iv

DAFTAR TABEL vi

DAFTAR GAMBAR viii

BAB I **PENDAHULUAN**

A. Latar Belakang 1

B. Rumusan Masalah 5

C. Batasan Masalah 6

D. Tujuan Penelitian 6

E. Manfaat Penelitian 7

BAB II **LANDASAN TEORI**

A. Pengertian dan Definisi Jalan 9

B. Karakteristik Konstruksi Jalan 13

C. Tahapan Perencanaan Jalan 17

D. *Alignment* Jalan 23

E. Bahasa Pemrograman Berorientasi Objek 44

F. *Global Positioning System* 59

BAB III **METODELOGI PENELITIAN**

A. Jenis Penelitian 63

B. Variabel Penelitian 63

C. Teknik Pengumpulan Data 66

D. Analisis Data 67

BAB IV **PERANCANGAN**

A. Analisa Kebutuhan Sistem 68

B. Perancangan Perangkat Lunak 98

BAB V **IMPLEMENTASI**

A. Spesifikasi Sistem 135

B. Implementasi Klas pada Program 138

C. Implementasi Algoritma 139

D. Implementasi Antarmuka Aplikasi 160

BAB VI **PENGUJIAN DAN ANALISA**

A. Pengujian Unit 187

B. Pengujian Validasi 205

BAB VII **PENUTUP**

A. Kesimpulan 241

B. Saran 242

DAFTAR RUJUKAN 243

DAFTAR TABEL

2.1 Perhitungan Volume Pekerjaan 22

2.2 Kelandaian Maksimum Jalan, Bina Marga 25

2.3 Panjang Kritis Untuk Kelandaian Melebihi Kelandaian Maksimum..... 25

2.4 Kelandaian Relative Maksimum menurut Bina Marga 37

2.5 Ls dan e yang dibutuhkan (e maksimum = 10%) Bina Marga 38

2.6 Ls dan e yang dibutuhkan (e maksimum = 8%) Bina Marga 39

2.7 Hak akses kelas dan subkelas 53

4.1 Tabel Utama Volume 100

4.2 Tabel Utama Vertikal 100

4.3 Tabel Utama Horisontal 100

4.3 Tabel Volume 103

4.4 Tabel Preparation 103

4.5 Tabel Stationing 104

4.6 Tabel Vertikal 104

4.7 Tabel Horisontal 104

4.8 Tabel Horisontal Circle 104

4.9 Tabel Horisontal Spiral-Circle-Spiral 104

4.10 Tabel Horisontal Spiral-Spiral 105

4.11 Entitas Volume 106

4.12 Entitas Stationing 106

4.13 Entitas Preparation 106

4.14 Entitas Vertikal 107

4.15 Entitas Horisontal 108

4.16 Entitas Horisontal Circle 108
4.17 Entitas Horisontal Spiral-Circle-Spiral 109
4.18 Entitas Horisontal Spiral-Spiral 110
4.19 Entitas GPS 111
4.20 Entitas R Minimum 111
4.21 entitas Ls Minimum dan elevasi 112
4.22 Entitas Jarak henti 112
4.23 Entitas Landai relative 112
4.24 Entitas operator 113
5.1 Spesifikasi Perangkat Keras 135
5.2 Spesifikasi Perangkat Lunak 136
5.3 Implementasi Klas dan File program 138
5.4 Penjelasan tombol login 161
5.5 Penjelasan tombol koneksi 164
5.6 Penjelasan tombol koordinat 165
5.7 Penjelasan tombol stationing 168
5.8 Penjelasan tombol preparation 169
5.9 Penjelasan tombol vertikal 170
5.10 Penjelasan tombol horisontal 172
5.11 Penjelasan tombol horisontal circle 174
5.12 Penjelasan tombol horisontal spiral-circle-spiral 176
5.13 Penjelasan tombol horisontal spiral-spiral 177
5.14 Penjelasan tombol volume 179
5.15 Penjelasan tombol operator 180
6.1 Hasil Pengujian Validasi 221

DAFTAR GAMBAR

2.1 Contoh Skema Penentuan *Station* 19

2.2 Lengkung Vertikal 26

2.3 Lengkung Peralihan menurut Bina Marga 36

2.4 Diagram superelevasi dengan sumbu jalan sebagai sumbu putar 40

2.5 Lengkung busur lingkaran 40

2.6 Lengkung spiral-circle-spiral 41

2.7 Lengkung spiral-spiral 43

4.1 Context diagram 75

4.2 DFD level 1 77

4.3 DFD level 2 Login proses 81

4.4 DFD level 2 *Preparation* proses 82

4.5 DFD level 2 Konek proses 84

4.6 DFD level 2 Data Koordinat 86

4.7 DFD level 2 *Stationing* proses 87

4.8 DFD level 2 *Alignment* vertikal 89

4.9 DFD level 2 *Alignment* horisontal 90

4.10 DFD level 2 *Alignment* horisontal *circle* 92

4.11 DFD level 2 *Alignment* horisontal *spiral-circle-spiral* 93

4.12 DFD level 2 *Alignment* horisontal *spiral-spiral* 95

4.13 DFD level 2 volume kerja 97

4.14 Entitas Sistem Perhitungan Pembangunan Jalan 113

4.15 Site map sistem 115

4.16 Tampilan Form Login 116

4.17 Tampilan Form Utama 117
4.18 Tampilan Form Preparation 117
4.19 Tampilan Form Koneksi tab bluetooth 118
4.20 Tampilan Form Koneksi tab data 119
4.21 Tampilan Form Koneksi tab koordinat 119
4.22 Tampilan Form koordinat 120
4.23 Tampilan Form Stationing 121
4.24 Tampilan Form Vertikal 122
4.25 Tampilan Form Horisontal 123
4.26 Tampilan Form Horisontal circle 124
4.27 Tampilan Form Horisontal spiral-circle-spiral 125
4.28 Tampilan Form Horisontal spiral-spiral 126
4.29 Tampilan Form Volume 127
4.30 Tampilan Map kerja 128
4.31 Tampilan Laporan Koordinat 128
4.32 Tampilan Laporan Preparation 129
4.33 Tampilan Laporan Horisontal 130
4.34 Tampilan Laporan Horisontal circle 130
4.35 Tampilan Laporan Horisontal spiral-circle-spiral 131
4.36 Tampilan Laporan Horisontal spiral-spiral 132
4.37 Tampilan Laporan Vertikal 132
4.38 Tampilan Laporan Volem kerja 133
4.39 Tampilan operator 134
4.40 Tampilan Form Laporan operator 134
5.1 Tampilan Login 161
5.2 Tampilan Form Konek panel bluetooth 162
5.3 Tampilan Form Konek panel data 163

5.4 Tampilan Form Konek panel koordinat 163
5.5 Tampilan Form Koordinat 165
5.6 Tampilan Form Stationing 167
5.7 Tampilan Form Preparation 168
5.8 Tampilan Form Vertikal 170
5.9 Tampilan Form Horisontal 171
5.10 Tampilan Form Horisontal circle 173
5.11 Tampilan Form Horisontal spiral-circle-spiral 175
5.12 Tampilan Form Horisontal spiral-spiral 176
5.13 Tampilan Form Volume kerja 178
5.14 Tampilan Form Operator 179
5.15 Tampilan Form Map kerja 181
5.16 Tampilan Laporan koordinat 182
5.17 Tampilan Laporan properti jalan 182
5.18 Tampilan Laporan *alignment* vertikal 183
5.19 Tampilan Laporan *alignment* horisontal 183
5.20 Tampilan Laporan *alignment* horisontal circle 184
5.21 Tampilan Laporan *alignment* horisontal spiral-circle-spiral 184
5.22 Tampilan Laporan *alignment* horisontal spiral-spiral 185
5.23 Tampilan Laporan volume kerja 186
5.24 Tampilan Laporan operator 186
6.1 Pemodelan metode pencarian dalam flow graph 188
6.2 Pemodelan konek dalam flow graph 191
6.3 Pemodelan login dalam flow graph 194
6.4 Pemodelan penyimpanan dalam flow graph 196
6.5 Pemodelan penghapusan dalam flow graph 199
6.6 Pemodelan penampilan map dalam flow graph 202

BAB I

PENDAHULUAN

1.1. Latar Belakang

Indonesia termasuk sebagai salah satu dari negara yang sedang berkembang, dimana pembangunan terjadi hampir pada semua sektor kehidupan masyarakat baik terjadi pada sektor ekonomi, pendidikan, pertanian, pariwisata, perhubungan maupun sektor – sektor yang lainnya. Pelaksanaan pembangunan tidak dipusatkan pada wilayah perkotaan saja, namun juga ke daerah – daerah bahakan pemerintah lebih mengupayakan pembangunan di pedesaan dan *sub urban* untuk lebih memeratakan peningkatan kesejahteraan masyarakat. Dengan semakin meningkat dan kompleksnya pembangunan, berbagai upaya dilakukan pemerintah untuk menunjang kegiatan tersebut. Bidang konstruksi merupakan salah satu sektor yang sangat menunjang pembangunan nasional. Hasil dari jasa bidang konstruksi sangat dirasakan manfaatnya oleh masyarakat, misalnya pembangunan sekolah, pembangunan rumah sakit dan puskesmas, perbaikan dan peningkatan jaringan irigasi, sarana transportasi dan sebagainya.

Perkembangan dalam pembangunan sarana transportasi dimulai dengan sejarah perkembangan manusia yang selalu memenuhi kebutuhan hidup dan kebutuhan berkomunikasi satu dengan lainnya. Di dalam sejarah Indonesia tercatat perkemabangan tekhnologi pembangunan sarana transportasi yang pertama kali terlaksana adalah pembangunan jalan raya dari Anyer ke Panarukan yang dilakukan pada jaman penjajahan Balanda. Namun secara teknis menurut teori rakayasa pembangunan jalan raya yang ada sekarang ini, kualitas dari pembangunan jalan

raya tersebut masih jauh dari yang diharapkan karena tidak menggunakan perencanaan secara geometri dan perkerasan jalannya secara matang. Salah satu bentuk metode perencanaan matang secara teknis baik geometri maupun perkerasannya adalah dengan menggunakan methode perhitungan perekayasaan jalan raya. Di dalam perekayasaan jalan raya diperhitungkan kebutuhan luasan penampang melintang dari agregat (batu – batu pecah penyusun jalan), luas pelebaran jalan pada tikungan, serta kadar intensitas aspal yang berfungsi sebagai perekat dalam campuran aspal dengan agregat, dari ketiga jenis perhitungan diatas kesemuanya diperlukan sistem perhitungan yang rumit da kompleks serta memerlukan waktu yang cukup lama dalam proses perhitungannya.

Di wilayah kabupaten Pasuruan, salah satu bidang yang sedang digalakkan pembangunannya adalah bidang sarana transportasi baik jalan raya maupun pembangunan sarana pendukung lainnya. Proses pelaksanaan pembangunan pada bidang jalan raya dapat dikategorikan menjadi tiga kelompok yakni kelompok pembangunan jalan, peningkatan jalan dan pemeliharaan jalan. Pembangunan jalan berhubungan dengan proses pengadaan jalan yang semula berupa lahan kosong kemudian dijadikan sebagai jalan *macadam* atau *telford*, peningkatan jalan berhubungan dengan peningkatan level jalan yang semula berasal dari jalan lingkungan yang berupa jalan *macadam* atau *telford* menjadi jalan lapis penetrasi, buras (laubran pasir), burtu (laburan batu) atau pun *hotmix* (campuran antara aspal panas dengan batu koral), sedangkan proses yang terjadi dalam pemeliharaan jalan adalah penutupan lubang – lubang yang ada, serta pembangunan sarana dan prasarana pendukung jalan. Diantara ketiga proses pembangunan jalan tersebut yang paling membutuhkan metode perekayasaan jalan dengan perhitungan yang

cukup rumit adalah proses pembangunan jalan raya. Karena didalamnya selain diperhitungkan mengenai luas dan geografi jalan, kebutuhan resource yang akan dipakai nantinya, juga diperhitungkan kadar intentsitas aspal yang berfungsi sebagai perekat dalam campuran aspal dengan batu. Ketiga perhitungan tersebut membutuhkan ketelitian dan waktu yang cukup lama dalam menyelesaikannya, sedangkan menurut Keputusan Presiden no. 08 tahun 2003 mengenai pedoman pengadaan barang dan jasa, di dalam pasal 12 dijelaksan bahwa pengguna barang atau jasa wajib mengalokasikan waktu yang cukup untuk penayangan pengumunan, kesempatan pengambilan dokumen, kesempatan untuk mempelajari dokumen, dan penyiapan dokumen penawaran, berdasarkan pada data di lapangan rata – rata waktu yang diberikan oleh panitia pengadaan terhadap penyedia jasa pemborongan dalam mempelajari dan menyusun dokumen penawarannya paling lambat adalah dalam kurun waktu lima sampai enam hari. Dalam kurun waktu tersebutlah penyedia jasa pemborongan nantinya diwajibkan untuk menentukan harga penawaran atas pelelangan tersebut, dengan sempitnya kurun waktu yang diberikan maka akan mempersulit dalam melakukan perhitungan atas perencanaan jalan. Sedangkan dalam pasal 19 disebutkan di dalamnya bahwa system penilaian biaya adalah evaluasi penilaian penawaran dengan cara memberikan nilai pada unsur – unsur teknis dan harga yang dinilai menurut ekonomis barang yang ditawarkan berdasarkan kriteria dan nilai unsur tersebut dikonversikan ke dalam suatu mata uang tertentu, dan dibandingkan dengan jumlah nilai dari setiap penawaran peserta dengan penawaran peserta lain, pengertian dari pasal tersebut bahwa nilai penawaran dari penyedia jasa harus memenuhi nilai – nilai ekonomis dari unsur – unsur teknis di dalam pengadaan jalan semisal batu dan aspal serta dapat

dipertanggung-jawabkan, kemudian bila memenuhi persyaratan maka nilai penawaran tersebut akan dibandingkan dengan penawaran lain untuk menentukan pemenang pengadaan jalan, hal ini lah yang menimbulkan persaingan harga antara penyedia jasa dalam pelelangan pengadaan barang, namun jika nilai penawaran terlalu rendah maka penyedia jasa akan dinyatakan gugur. Oleh karena kedua hal tersebutlah maka para penyedia jasa memerlukan suatu tools atau alat bantu dalam melakukan perhitungan – perhitungan rumit pada perencanaan jalan dan penyusunan dokumen penawaran secara cepat dan dapat dipertanggung jawabkan.

Dengan semakin pesatnya perkembangan dunia tekhnologi, maka akan memberikan pilihan bantuan kepada para penyedia jasa untuk mengatasi permasalahan yang sedang dihadapi. Dengan menggunakan komputerisasi maka akan mempermudah dalam melakukan perhitungan perencanaan jalan. Karena komputer adalah perangkat yang mampu melakukan komputasi dan membuat keputusan logis dengan kecepatan cahaya, miliyaran kali lebih cepat dibandingkan dengan kemampuan dan skill yang dimiliki manusia. Komputer mengolah data dibawah kendali sekumpulan instruksi yang disebut program atau aplikasi komputer. Aplikasi ini menutut komputer malakukan serangkaian aksi – aksi yang dispesifikkan, aplikasi ini disusun berdasarkan atas bahasa yang dikenal oleh komputer. Bahasa yang dikenal oleh komputer dikelompokkan menjadi tiga tingkatan yakni antara lain bahasa pemrograman tingkat rendah dimana sintaks – sintaksnya condong mendekati bahasa mesin, kedua bahasa pemrograman tingkat menengah dimana sintaks – sintaksnya berada diantara bahasa mesin dan manusia, ketiga adalah bahasa pemrograman tingkat atas dimana sintaks – sintaksnya

menggunakan bahasa manusia dan diperlukan suatu *compiler* untuk mengkonversinya menjadi bahasa mesin.

Salah satu bahasa pemrograman tingkat atas yang marak digunakan saat ini adalah bahasa pemrograman yang berorientasi objek (bahasa java) dimana memiliki perbedaan dengan bahasa – bahasa tingkat tinggi lainnya. Di dalam bahasa pemrigraman berorientasi objek ditawarkan fleksibilitas, kegunaulangan dan kemudahan dalam melakukan perhitungan – perhitungan yang rumit.

Oleh karena adanya persaingan ketat yang timbul di dalam proses pelelangan pada pengadaan barang dan jasa serta ganjalan – ganjalan yang timbul atas kekompleksitasan dan kerumitan dalam perhitungan perencanaan penyedian jasa pemborongan jalan. Untuk itu penulis mencoba untuk memberikan solusi permasalahan tersebut melalui proses komputerisasi dengan mengaplikasikan perhitungan perencanaan pemborongan jalan dan penyusunan atas penawaran yang sangat diperlukan dalam proses pelelangan pengadaan barang dan jasa khususnya pada konstruksi jalan.

1.2. Rumusan Masalah

Berdasarkan uraian latar belakang diatas, maka penulis dapat menguraikan permasalahan – permasalah yang timbul sebagai berikut :

1. Bagaimanakah sistem kerja dari aplikasi yang dapat mewakili perhitungan – perhitungan di dalam perencanaan pembangunan jalan terutama pada proses aliran logika yang terjadi di dalamnya?
2. Bagaimanakah proses perancangan interface program aplikasi perhitungan pembangunan jalan berbasis java menggunakan gps?

1.3. Batasan Masalah

Untuk membatasi pembahasan pada penilitian agar tidak melebar dari tujuan yang hendak dicapai, maka ditentukan batasan – batasan permasalahannya. Adapun batasan tersebut antara lain :

1. Penilitian ditetapkan pada ruang lingkup penyusunan aplikasi perhitungan pada proses perencanaan pemborongan jasa konstruksi pembangunan jalan raya.
2. Penyusunan program aplikasi menggunakan bahasa pemrograman java dan bahasa *database mySql* sebagai pendukung dari bekerjanya aplikasi.
3. Aplikasi hanya terbatas pada perhitungan pengerjaan jalan tanpa menyangkut pautkan sarana dan prasarana pendukung jalan seperti bahu jalan, pelengsengan pengalir air hujan dari jalan, maupun paving trotoal pejalan kaki.
4. Aplikasi hanya memperhitungkan *alignment* secara individu bukan dalam bentuk campuran, bentuk campuran semisal *alignment* vertikal yang diikuti dengan horisontal.

1.4. Tujuan Penelitian

Tujuan dari penelitian yang dilakukan penulis dapat dikategorikan menjadi dua, yakni :

1.4.1. Tujuan Umum

1. Memberikan gambaran umum mengenai proses perencanaan pembangunan jalan.
2. Memberikan kesempatan kepada penulis untuk mengimplementasikan ilmu dan teori yang telah didapat.

3. Memberikan solusi permasalahan kepada para penyedia jasa dalam melakukan perhitungan dan perencanaan jalan.

1.4.2. Tujuan Khusus

1. Mengetahui system perhitungan yang dipergunakan dalam perencanaan pada pengadaan jasa konstuksi pembangunan jalan raya.
2. Mengetahui dan memahami bahasa pemrograman Java yang digunakan sebagai aplikasi perhitungan perencanaan tersebut.

1.5. Manfaat Penelitian

Berdasarkan dari permasalah yang diangkat dalam penelitian ini, maka manfaat yang ingin dicapai dalam penelitian ini adalah sebagai berikut :

1. Bagi perkembangan dunia konstruksi dan penyedia jasa pemborongan.

   Penelitian ini merupakan sumbangan wawasan untuk memberikan solusi pemecahan masalah atas persaingan ketat yang terjadi di dalam pengadaan barang dan jasa, karena dengan aplikasi ini maka penyedia jasa lebih mampu untuk menganalisa total kebutuhan dan mencegah terjadinya kerugian akibat penurunan harga penawaran yang berlebihan.

2. Bagi lembaga.

   Hasil penelitian ini dapat digunakan sebagai referensi mahasiswa yang akan meneliti lebih dalam mengenai pembuatan aplikasi perhitungan konstruksi jalan khususnya pada peningkatan jalan raya.

3. Bagi peneliti.

Dapat memberikan pengalaman dan menambah wawasan tentang bentuk dan gambaran umum mengenai dunia perkonstruksian di Indonesia, khususnya pada pengadaan jasa konstruksi pembangunan jalan raya.

BAB II

LANDASAN TEORI

Pada landasan teori berikut dikemukakan mengenai teori tentang pengertan dan karakteristik jalan, tahapan dasar perencanaan jalan baik dari segi geometric maupun perkerasan jalan, bahan lapis perkerasan dan *subgrade*-nya, pondasi permukaan jalan, serta teori tentang bahasa pemerograman java yang digunakan untuk menyusun dan merancang aplikasi perhitungan kebutuhan konstruksi jalan.

A. Pengertian dan Definisi Jalan Raya

Hendra (2008 : 1) mengemukakan bahwa " Jalan Raya adalah suatu lintasan yang bermanfaat untuk melewatkan lalu lintas dari suatu tempat ke tempat yang lain". Di dalam bukunya Hendra menambahkan bahwa "lintasan adalah jalur tanah yang diperkuat atau diperkeras dan jalur tanpa perkerasan tergantung volume lalu lintas" serta "lalu lintas adalah semua benda dan makhluk yang melewati jalan tersebut, baik kendaraan bermotor, manusia dan hewan". Menurut Undang-Undang Jalan Raya No.13 tahun 1980 bahwa " Jalan adalah suatu prasarana perhubungan darat dalam bentuk apapun, meliputi segala bagian jalan termasuk bangunan pelangkap dan perlengkapannya yang diperuntukkan bagi lalu lintas". Dari beberapa pengertian yang dikemukakan diatas maka dapat ditarik kesimpulan bahwa jalan adalah suatu jalur berupa tanah tanah yang mengalami perkerasan serta jalur tanah tanpa perkerasan yang memiliki kegunaan sebagai sarana penghubung bagi semua benda dan makhluk yang melewati baik kendaraan bermotor, manusia dan hewan dari suatu tempat ke tempat yang lain.

Undang-Undang Jalan Raya No.13 tahun 1980 juga mengkategorikan jalan menjadi tiga bentuk, yakni :

1. Jalan Umum

   Jalan umum adalah jalan yang dipergunakan dan difungsikan sebagai sarana dan prasarana bagi lalu lintas umum.

2. Jalan Khusus

   Jalan khusus adalah jalan yang dipergunakan dan difungsikan sebagai sarana dan prasarana kepentingan khusus, dengan kata lain tidak diperuntukkan sebagai lalu lintas umum.

3. Jalan Tol

   Jalan tol adalah jalan umum yang dipergunakan dan difungsikan sebagai sarana dan prasarana bagi lalu lintas umum dimana pengguna jalan dikenakan tarif dan biaya pemakaian jalan (tol).

Berdasarkan Peraturan Perencanaan Geometrik Jalan Raya No.13 tahun 1970, jalan dikategorikan berdasarkan fungsi jalan menjadi 3 kategori, yakni sebagai berikut :

1. Jalan utama

   Jalan yang berfungsi melayani lalu lintas dengan volume tinggi antara kota-kota penting, sehingga harus direncanakan untuk dapat melayani lalu lintas cepat dan berat. Jalan utama dikalsifikasikan kedalam kelas I dimana lalu lintas hariannya di atas 20.000 kendaraan.

2. Jalan sekunder

   Jalan yang berfungsi melayani lalu lintas cukup tinggi antara kota-kota penting dan kota-kota yang lebih kecil serta daerah sekitarnya. Jaln sekunder

diklasifikasikan kedalam kelas II, di dalam kelas II tersebut masih diklasifikasikan lagi menjadi 3 yakni kelas IIA dimana lalu lintas hariannya mencapai 6.000 – 20.000, kelas IIB dimana lalu lintas hariannya mencapai 1.500 – 2.000, kelas IIC dimana lalu lintas hariannya kurang dari 2.000 kendaraan.

3. Jalan penghubung

   Jalan yang berfungsi untuk keperluan aktivitas daerah yang juga dipakai sebagai penghubung antara jalan-jalan dari golongan yang sama atau berlainan. Jalan penghubung diklasifikasikan kedalam kelas III.

Adapun karakteristik dari klasifikasi-klasifikasi kelas yang telah dijabarkan sebelumnya adalah sebagai berikut :

Kelas I :

a. Melayani lalu lintas cepat dan berat.
b. Tidak terdapat kendaraan lambat atau tidak bermotor.
c. Berupa jalan raya berlajur banyak.
d. Jenis kontruksi perkerasan baik.
e. Tingkat pelayanan tinggi.

Kelas II :

a. Mencakup semua jalan sekunder dua jalur atau lebih.
b. Konstruksi aspal beton atau setaraf.
c. Terdapat kendaraan lambat, tidak ada kendaraan tak bermotor.
d. Untuk kendaraan lambat disediakan jalur sendiri.

Kelas IIA :

a. Dua jalu atau lebih dengan permukaan aspal beton atau setaraf.
b. Terdapat kendaraan lambat tanpa kendaraan tak bermotor.
c. Ada 3 kelas yang berlainan sifat lalu lintasnya.

Kelas IIB :

a. Dua jalur dengan konstruksi permukaan penetrasi berganda atau yang setaraf.
b. Terdapat kendaraan lambat tanpa kendaraan tak bermotor.

Kelas IIC :

a. Dua jalur dengan konstruksi permukaan penetrasi tunggal.
b. Terdapat kendaraan lambat dan kendaraan tak bermotor.

Kelas III :

a. Mencakup semua jalan penghubung.
b. Konstruksi jalan berjalur tunggal atau dua.
c. Jenis konstruksi paling tinggi adalah pelaburan dengan aspal.

Sedangkan jika berdasarkan menurut jenis pengelolaannya, jalan dapat dikategorikan menjadi :

a. Jalan arteri

   Jalan arteri adalah jalan yang terletak di luar pusat area perdagangan (*out lying business district*)

b. Jalan kolektor

   Jalan kolektor adalah jalan yang terletak di pusat area perdagangan (*central business district*)

c. Jalan local

Jalan local adalah jalan yang terletak pada area pemukiman penduduk.

d. Jalan negara

Jalan negara adalah jalan yang menghubungkan antara ibukota provinsi. Biaya pembangunan dan perawatannya ditanggung oleh pemerintah pusat.

e. Jalan kabupaten

Jalan kabupaten adalah jalan yang menghubungkan antara ibukota provinsi dengan ibukota kabupaten atau ibukota kabupaten dengan ibukota kecamatan, juga antar desa dalam satu kabupaten.

B. Karakteristik Konstruksi Jalan

Karakteristik kontruksi suatu jalan dapat diamati melalui penampang melintang jalan tersebut. Sukirman (1999 : 21) menjelaskan bahwa " Penampang melintang jalan merupakan potongan melintang yang ditarik tegak lurus dari tepi jalan memotong sumbu jalan hingga mencapai tepi jalan sisi yang lain". Pada potongan melintang jalan dapat terlihat bagian-bagian jalan. Bagian-bagian jalan utama dapat dikelompokkan sebagai berikut :

1. Bagian yang langsung berguna untuk lalu lintas :

   a. Jalur lalu lintas.

      Jalur lalu lintas disebut juga dengan *travelled way* atau *carriage way* adalah keseluruhan bagian perkerasan jalan yang diperuntukkan bagi lalu lintas kendaraan yang terdiri atas beberapa lajur kendaraan.

b. Lajur lalu lintas.

Yaitu bagian dan jalur lalu lintas yang khusus diperuntukkan bagi lewatnya satu rangkaian kendaraan beroda empat atau lebih dalam satu arah. Lebar lajur lalu lintas merupakan bagian yang paling menentukan lebar melintang jalan secara keseluruhan. Besarnya lebar lajur hanya dapat ditentukan dengan pengamatan langsung di lapangan, karena lintasan kendaraan tidak mungkin tepat sama dengan lebar kendaraan maksimum dan lintasan kendaraan tidak mungkin tetap sejajar sumbu lajur lalu lintas karena dipengaruhi oleh gaya-gaya menyamping seperti tidak ratanya jalan, gaya *sentrifugal* pada tikungan, dan gaya dorong angin ketika ada kendaraan lain yang menyalip.

Dinas Bina Marga menentukan lebar kendaraan rencana untuk mobil penumpang adalah 1,7 m dan untuk kendaraan rencana truck/bus/semitrailer adalah 2,5 m. lebar lajur lalu lintas merupakan lebar kendaraan ditambah dengan ruang bebas antar kendaraan yang besarnya sangat ditentukan keamanan dan kenyamanan yang diharapkan. Lajur yang dipergunakan untuk kecepatan tinggi memerlukan ruang bebas untuk menyalip dan bergerak yang lebih besar dibandingkan dengan jalan untuk kecepatan rendah. Jumlah lajur lalu lintas yang dibutuhkan tergantung pada volume lalu lintas yang akan memakai jalan tersebut dan tingkat pelayanan yang diharapkan.

Kemiringan melintang jalur lalu lintas di jalan lurus diperuntukkan terutama untuk kebutuhan drainase jalan. Kemiringan melintang bervariasi antara 2 % - 4 % untuk jenis lapisan permukaan dengan mempergunalan bahan pengikat

seperti aspal atau semen. Semakin kedap air lapisan perkerasan semakin kecil kemiringan melintang yang dipergunakan. Besarnya kemiringan melintang akan dibahas lebih lanjut pada bab berikutnya.

c. Bahu jalan.

Adalah jallur yang terletak berdampingan dengan jalur lalu lintas yang berfungsi sebagai ruangan untuk tempat berhenti sementara kendaraan, ruangan untuk menghindarkan diri pada saat darurat, ruangan pembantu pada waktu mengadakan pekerjaan perbaikan atau pemeliharaan jalan, dan ruangan untuk lintasan kendaraan patroli dan ambulan yang sangat dibutuhkan pada keadaan darurat.

d. Trotoar.

Adalah jalur yang terletak berdampingan dengan jalur lalu lintas yang khusus dipergunakan untuk pejalan kaki. Untuk keamanan pejalan kaki maka trotoar ini harus dibuat terpisah dari jalur lalu lintas oleh struktur berupa kereb. Lebar trotoar yang dibutuhkan ditentukan volume pejalan kai dan tingkat pelayanan pejalan kaki yang diinginkan, untuk itu lebar 1,5 – 3 meter merupakan nilai yang umum dipergunakan.

e. Median.

Adalah jalur yang terletak ditengah jalan untuk membagi jalan dalam masing-masing arah. Median berfungsi untuk menyediakan daerah netral yang cukup netral dimana pengemudi masih dapat mengontrol kendaraannya pada saat darurat, menyediakan jarak yangcukup untuk membatasi kesilauan terhadap lampu besar dari kendaraan yang berlawanan arah, mengamankan kebebasan samping dari masing-masing arah arus lalu lintas.

2. Bagian yang berguna untuk drainase jalan :

   a. Saluran samping.

      Saluran samping berguna untuk mengalirkan air dari permukaan perkerasan jalan ataupun dari bagian luar jalan dan menjaga supaya konstruksi ja;an selalu berada dalam keadaan kering tidak terendam air.

      Umumnya bentuk saluran samping trapezium atau empat persegi panjang. Lebar dasar saluran samping bergantung pada debit air yang diperkirakan akan mengalir, minimum sebesar 30 cm.

   b. Kemiringan lereng (talud)

      Talud untuk saluran samping yang berbentuk trapezium dan tidak diperkeras adalah 2:1 atau sesuai dengan kemiringan yang memberikan kestabilan lerengyang aman.

3. Bagian yang pelengkap jalan :

   a. Kereb.

      Adalah penonjolan atau peninggian tepi perkerasan atau bahu jalan, yang terutama dimaksudkan untuk keperluan-keperluan drainase, mencegah keluarnya lintasan kendaraan dari tepi perkerasan dan memberikan ketegasan tepi perkerasan.

   b. Pengaman tepi.

      Pengaman tepi bertujuan untuk memberkan ketegasan tepi badan jalan. Jika terjadi kecelakaan, dapat mencegah kendaraan keluar dari badan jalan. Umumnya dipergunakan disepanjang jalan yang menyusur jurang.

4. Bagian konstuksi jalan :
   a. Lapisan perkerasan jalan.
   b. Lapisan pondasi atas.
   c. Lapisan pondasi bawah.
   d. Lapisan tanah dasar.
5. Daerah manfaat jalan (damaja).

   Daerah manfaat jalan meliputi badan jalan, saluran tepi jalan, dan ambang pengamannya. Badan jalan meliputi jalur lalu lintas dengan atau tanpa jalur pemisah dan bahu jalan.
6. Daerah milik jalan (damija).

   Merupakan ruang sepanjang jalan yang dibatasi oleh lebar dan tinggi tertentu yang dikuasai oleh pembina jalan dengan suatu hak tertentu.
7. Daerah pengawasan jalan (dawasja).

   Adalah suatu jalur tanah tertentu yang terletak di luar damija, yang penggunaannya diawasi oleh pembina jalan, dengan maksud agar tidak mengganggu pandangan pengemudi dan konstruksi bangunan jalan, dalam hal tidak cukup luasnya damija.

C. Tahapan Perencanaan Jalan

Prinsip utama adanya perencanaan jalan adalah untuk memenuhi syarat keamanan, kenyamanan, kecepatan dan ekonomis. Syarat-syarat ini akan terkait dengan jarak pandang, koefisien gesekan ban dengan lapis permukaan serta ruang gerak kendaraan. Tahapan perencanaan jalan memiliki empat tahapan :

1. Penentuan *trase* jalan

Tahapan awal pada perencanaan jalan adalah tahapan penentuan *trase* jalan, tahapan ini sangat bergantung kepada empat Faktor berikut :

a. Faktor Topografi

Faktor ini mempengaruhi kelandaian jalan, jarak pandangan, penampang melintang dan sebagainya. Permasalahannya adalah topografi yang memiliki daerah perbukitan, lembah sungai atau danau sering memberikan pembatasan terhadap lokasi dan perencanaan *trase*.

b. Faktor Geologi

Pada daerah yang rawan seperti patahan atau daerah yang bergerak, merupakan daerah yang tidak baik untuk dibuat *trase* jalan.

c. Faktor Tata Guna Lahan

Penentuan tata guna lahan merupakan hal yang paling mendasar dalam perencanaan lokasi suatu jalan. Apakah jalan yang direncanakan untuk daerah pemukiman, daerah perindustrian atau yang lain. Kesemuanya terkonsentrasi pada tujuan pembuatan jalan ini.

d. Faktor Lingkungan

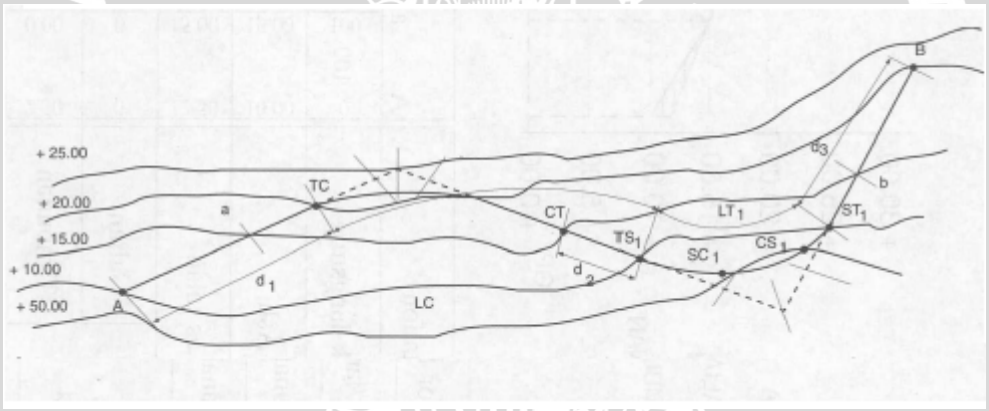
Faktor lingkungan ini menjadi suatu alat penganalisa kelayakan dibuatnya *trase* jalan, apakah akan membawa efek positif ataukan negative pada keadaan di lingkungan sekitarnya.

2. Penentuan Stasiun (*Stationing*)

Tujuan penentuan stasiun untuk menentukan panjang suatu lokasi jalan atau jarak dan suatu tempat ke tempat lain pada suatu lokasi jalan. Titik pada panjang

tertentu dinamakan titik stasiun. Jadi stasiun (Sta) adalah jarak langsung yang diukur dan titik awal berupa Sta 0 + 000 sampai titik yang dicari stasiunnya. Titik awal penting pada suatu rencana *trase* harus ditetapkan atau dihitung stasiunnya. Untuk menghitung stasiun di luar titik penting dilakukan dengan cara yang sama dengan kriteria-kriteria berikut :

a. Untuk daerah datar dibuat jarak patork ± 100 meter.
b. Untuk daerah perbukitan jarak patoknya dibuat ± 50 meter.
c. Untuk daerah pegunungan jarak patoknya dibuat ± 25 meter.
d. Untuk daerah tikungan, jarak patoknya harus dibuat lebih pendek menurut keperluan ketelitian.



Gambar 2.1 Contoh Skema Penentuan Stasiun.

Keterangan dari gambar 2.1 adalah sebagai berikut.

a. Titik A = Sta. 0+000
b. Titik TC = Sta. A + d1
c. Titik CT = Sta. TC + Lc
d. Titik TS1 = Sta. CT + d2
e. Titik ST1 = Sta.TS1 + LT1

f. Titik B = Sta. TS1 + d3

Dimana :

a. Titik A adalah titik awal pengerjaan konstruksi jalan.
b. d1 adalah panjang bagian lurus (*tangent*) dari A sampai TC.
c. Titik TC adalah titik awal lengkung *circle* (*Alignment* vertikal).
d. LC adalah panjang lengkung *circle* (*Alignment* vertikal).
e. Titik CT adalah titik akhir lengkung *circle* (*Alignment* vertikal).
f. d2 adalah panjang bagian lurus (*tangent*) dari CT hingga TS1.
g. Titik TS1 adalah titik awal dari lengkung peralihan dari jalan lurus menuju jalan tikungan.
h. Titik SC1 adalah titik awal dari lengkung pada tikungan.
i. Titik CS1 adalah titik akhir dari lengkung pada tikungan.
j. Titik ST1 adalah titik akhir dari lengkung peralihan dari jalan tikungan menuju jalan lurus.
k. LT1 adalah panjang total dari jalan tikungan.
l. d3 adalah panjang bagian lurus (*tangent*) dari ST1 sampai B.
m. Titik B adalah titik akhir pengerjaan konstruksi jalan.

3. Perencanaan potongan memanjang dan melintang

Perencanaan potongan memanjang dan melintang disusun untuk dipergunakan sebagai skema pembantu dalam perhitungan volume pekerjaan.

a. Potongan Memanjang

Untuk pembuatan potongan memanjang sebaiknnya dibuat menggunakan skala horisontalnya 1 : 1000 atau 1 : 2000 dan skala vertikalnya 1 : 100. Potongan memanjang digambarkan langsung pada hasil pengukuran, sehingga akan diketahui bagian yang harus digali atau ditimbun. Perhitungan gambar nantinya didasarkan pada perhitungan *Alignment* vertikal serta standar-standar yang digunakan.

b. Potongan Melintang

Potongan melintang adalah potongan dari skema jalan pada tiap – tiap titik stasiunnya yang tegak lurus dengan sumbu jalan dan diproyeksikan sejajar menghadap sumbu jalan. Pada potongan melintang ini ditampakkan bagian-bagian dari jalan baik itu lajur jalan, bahu jalan, drainase jalan dan bagian-bagian lain.

4. Perhitungan volume pekerjaan

Di dalam perencanaan jalan raya diusahakan volume galian sama dengan volume timbunan. Sehingga nantinya pada perhitungan keseluruhan volume pekerjaan galian dan timbunan tidak menjadi beban pekerjaan keseluruhan. Dengan menggunakan gambar potongan melintang dan memanjang maka akan memungkinkan menghitung keselurhan pekerjaan konstruksi yang akan dilaksanakan. Langkah-langkah perhitungan volume pekerjaan adalah :

a. Penentuan satitioning sehingga diperoleh panjang horizontal jalan dan *Alignment* horisontalnya.

b. Penggambaran profil memanjang yang memperlihatkan perbedaan tinggi tanah asli dengan tinggi perencanaan perkerasan.

c. Penggambaran profil melintang pada tiap-tiap titik stasiun, sehingga didapat luas penampang luas penampang pekerjaan.

d. Perhitungan volume galian dan timbunan dengan mengalikan luas penampang rerata dan jarak antar stasiun.

Adapun rumus perhitungan volume pekerjaan dapat diselesaikan dengan model table seperti berikut.

No. Sta	Luas Penampang Melintang (m^2)						Jarak (m)	Volume (m^3)		
	Satuan			Rerata				Galian	Timbunan	Perkerasan
	G	T	P	$\bar{G}$	$\bar{T}$	$\bar{P}$				
1.	AG1	AT1	AP1	$\frac{AG1+AG2}{2}$	$\frac{AT1+AT2}{2}$	$\frac{AP1+AP2}{2}$	J1	$J1 \times \bar{G}$	$J1 \times \bar{T}$	$J1 \times \bar{P}$
2.	AG2	AT2	AP2	$\frac{AG2+AG3}{2}$	$\frac{AT2+AT3}{2}$	$\frac{AP2+AP3}{2}$	J2	$J2 \times \bar{G}$	$J2 \times \bar{T}$	$J2 \times \bar{P}$
3.	AG3	AT3	AP3	$\frac{AG3+AG4}{2}$	$\frac{AT3+AT4}{2}$	$\frac{AP3+AP4}{2}$	J3	$J3 \times \bar{G}$	$J3 \times \bar{T}$	$J3 \times \bar{P}$
dst.	dst	dst	dst	dst	dst	dst	dst	dst	dst	dst

Tabel 2.1 Perhitungan Volume Pekerjaan

Keterangan :

- G : Luas penampang melintang galian suatu stasiun
- T : Luas penampang melintang timbunan suatu stasiun
- P : Luas penampang melintang perkerasan suatu stasiun
- $\bar{G}$: Luas penampang rerata galian antara dua stasiun
- $\bar{T}$: Luas penampang rerata timbunan antara dua stasiun

- $\bar{P}$: Luas penampang rerata perkerasan antara dua stasiun

Pada keadaan-keadaan tertentu, seperti terjadinya tikungan horizontal dan vertikal pada jalan maka perhitungan luas penampang dan volume tidak lagi menggunakan metode seperti diatas melainkan menggunalan pendeketan integral.

D. *Alignment* Jalan

Silvia (1999 : 67) mengatakan bahwa "Yang dinamakan sebagai *Alignment* adalah proyeksi sumbu jalan pada suatu bidang tertentu yang dipengaruhi oleh beberapa gaya-gaya dan kecepatan rencana kendaraan". Adapun *Alignment-Alignment* dapat dikategorikan berdasarkan ordinat dan absisnya terhadap sumbu jalan menjadi dua bentuk yakni *Alignment* horizontal dan vertikal.

1. *Alignment* Vertikal

Sukirman (1999 : 153) menjelaskan bahwa "*Alignment* vertikal adalah perpotongan bidang vertikal dengan bidang permukaan perkerasan jalan melalui sumbu jalan untuk jalan 2 jalur 2 arah atau melalui tepi dalam masing-masing perkerasan untuk jalan dengan median".

Perencanaan *Alignment* vertikal dipengaruhi oleh besarnya biaya pembangunan yang tersedia. *Alignment* vertikal yang mengikuti permukaan tanah asli akan mengurangi pekerjaan tanah, akan tetapi juga mengakibatkan jalan memiliki banyak lengkungan yang belum tentu sesuai dengan persyaratan sehubungan dengan fungsi jalan. Penarikan *Alignment* vertikal sangat dipengaruhi oleh berbagai pertimbangan sebagai berikut :

a. Kondisi tanah dasar
b. Keadaan medan
c. Fungsi jalan
d. Muka air banjir
e. Muka air tanah
f. Kelandaian yang masih memungkinkan

Kelandaian yang dimaksud dinyatakan dalam bentuk persen, pada perencanaan *Alignment* vertikal kelandaian memiliki nilai maksimum dan minimum.

a. Landai minimum

   Berdasarkan kepentingan arus lalu lintas, landai ideal adalah datar (0%). Sebaliknya ditinjau dari kepentingan drainase jalan, jalan berlandailah yang ideal. Dalam perencanaan disarankan menggunakan :

   i. Landai datar (0%) untuk jalan di atas tanah timbunan yang tidak mempunyai kereb.
   ii. Landai 0,15% dianjurkan untuk jalan di atas tanah timbunan dengan medan datar dan mempergunakan kereb.
   iii. Landai minimum sebesar 0,3 – 0,5% dianjurkan dipergunkan untuk jalan di daerah galian atau jalan yang memakai kereb.

b. Landai maksimum

   Keladaian dengan nilai 3% mulai memberikan pengaruh kepada gerak kendaraan mobil penumpang, walaupun tidak seberapa dibandingkan dengan gerakan kendaraan truck bermuatan penuh. Pengaruh dari kelandaian ini terlihat dari berkurangnya kecepatan laju kendaraan. Kelandaian tertentu

masih diperbolehkan jika kelandaian tersebut mengakibatkan kecepatan laju yang tetap lebih besar dari setengah kecepatan rencana. Untuk membatasi pengaruh perlambatan kendaraan pada arus lalu lintas, maka ditetapkan landai maksimum untuk kecepatan rencana tertentu. Bina Marga menetapkan kelandaian maksimum seperti pada table berikut.

Kecepatan Rencana Km/Jam	Jalan Antar Kota	
	Kelandaian Maksimum Standar (%)	Kelandaian Maksimum Mutlak (%)
40	7	11
50	6	10
60	5	9
80	4	8

Tabel 2.2 Kelandaian Maksimum Jalan, Bina Marga 1990

Faktor pengaruh lain yang menjadi penentu dalam perencanaan *Alignment* vertikal adalah Faktor panjang dari suatu kelandaian. Semakin panjang suatu lengkungan jalan maka akan semakin besar pula faktor yang menimbulkan penurunan kecepatan kendaraan walaupun pada kelandaian yang sama besarnya. Batas kritis umumnya diambil jika kecepatan kendaraan berkurang mencapai 30 – 75% kecepatan rencana, atau kendaraan terpaksa mempergunakan gigi rendah. Untuk mencegah terjadinya hal tersebut maka Bina Marga memberikan nilai panjang kritis yang boleh dipergunakan sesuai dengan table berikut.

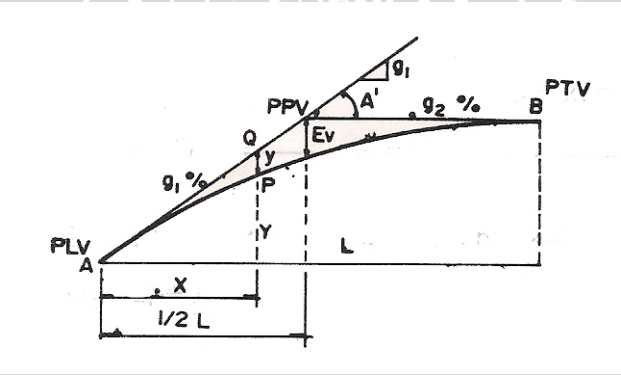
Kecepatan Rencana (Km/Jam)												
80		60		50		40		30		20		
5%	500 m	6%	500 m	7%	500 m	8%	420 m	9%	340 m	10%	250 m	
6%	500 m	7%	500 m	8%	420 m	9%	340 m	10%	250 m	11%	250 m	
7%	500 m	8%	420 m	9%	340 m	10%	250 m	11%	250 m	12%	250 m	
8%	420 m	9%	340 m	10%	250 m	11%	250 m	12%	250 m	13%	250 m	

Tabel 2.3 Panjang Kritis untuk kelandaian melebihi kelandaian maksimum standar

Pergantian dari suatu kelandaian ke kelandaian yang lain dilakukan dengan mempergunakan lengkung vertikal. Lengkung tersebut direncanakan sedemikian rupa sehingga memenuhi keamanan dan drainase jalan. Jenis lengkungan dilihat dari bentuk perpotongan kedua bagian lurus (*tangent*) adalah :

a. Lengkung vertikal cekung, dimana titik perpotongan antara kedua *tangent* berada di bawah permukaan jalan.
b. Lengkung vertikal cembung, dimana titik perpotongan antara kedua *tangent* berada di atas permukaan jalan.

Bentuk lengkung vertikal yang umum dipergunakan adalah bentuk lengkung parabola sederhana. Besarnya kelandaiaan bagian *tangent* dinyatakan dengan g1% dan g2%, kelandaian diberi tanda positif (+) jika mendaki dan negative (-) jika menurun yang ditinjau dari bagian kiri sketsa jalan.



Gambar 2.2 Lengkung vertikal

Adapun rumus lengkung vertikal dapat dicari melalui rumus umum parabola, yakni.

$$dY/dX = rX + C$$

$$x = 0 \quad \rightarrow \quad dY/dX = g_1 \quad \rightarrow \quad C = g_1$$

$$x = L \quad \rightarrow \quad dY/dX = g_2 \quad \rightarrow \quad rL + g_1 = g_2$$

$$r = (g_2 - g_1)/L$$

$$dY/_{dX} = \left\{ (g_2 - g_1)/_L \right\} X + g_1$$

$$Y = \frac{(g_2 - g_1)}{L} \frac{X^2}{2} + g_1 X \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.1)$$

Dari sifat segitiga sebangun didapat :

$$\frac{(Y + y)}{X} = \frac{g_1 \cdot \frac{1}{2}L}{\frac{1}{2}L}$$

$$(Y + y) = g_1 \cdot X$$

$$g_1 \cdot X = Y + y$$

$$Y - (g_2 - g_1) \cdot x^2/2L = Y + y$$

$$y = -(g_2 - g_1) \cdot x^2/2L$$

$$y = (g_1 - g_2) \cdot x^2/2L$$

Jika dinyatakan dalam persen maka :

$$y = \frac{A}{200L} \cdot x^2$$

Dimana :

y : adalah jarak antara bagian *tangent* jalan pada absis x dengan lengkung vertikalnya

A : adalah perbedaan aljabar landai ($g_1 - g_2$)

L : adalah panjang lengkung vertikal pada bidang horisontal

x : adalah absis yang dipergunakan dalam lengkung vertikal

Dari persamaan rumus 1.1, maka dapat dicari panjang dari *Alignment* vertikal (L_l) dengan mempergunakan rumus kalkulus sebagai berikut.

$$L_l = \int_0^L \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx$$

$$L_l = \int_0^L \sqrt{1 + \left[\frac{(g_2 - g_1) \cdot X}{L} + g_1\right]^2}\, dx$$

Bentuk lengkung vertikal seperti yang diuraikan di atas, berlaku untuk lengkung vertikal cembung dan cekung. Hanya saja untuk masing-masing lengkung terdapat batasan-batasan. Pada lengkung vertikal cembung yang panjang dan relative datar dapat menyebabkan kesulitan dalam masalah drainase jika disepanjang jalan dipasang kereb. Untuk menghindari hal tersebut maka bina marga memberikan ketentuan bahwa panjang lengkung vertikal tidak melebihi 50A. Sehingga dapat dirumuskan menjadi.

$$L \leq 50 \cdot A$$

Selain didasarkan pada persyaratan drainase, pada lengkung vertikal cembung juga diperhatikan mengenai kenyamanan perjalanan. Untuk perbedaan aljabar yang kecil, maka panjang lengkung vertikal yang dibutuhkan pendek, sehingga *Alignment* tampak melengkung. Oleh karena itu disyaratkan panjang

lengkung yang diambil untuk perencanaan tidak kurang dari 3 detik perjalanan. Sehingga dapat dirumuskan menjadi.

$$L_l \geq V \cdot \frac{3 \cdot 1000}{3600}\ meter$$

$$L_l \geq V \cdot 0{,}83333\ meter$$

Sedangkan pada bentuk lengkung vertikal cekung, adanya gaya *sentrifugal* dan gaya gravitasi pada lengkung menimbulkan ketidak nyamanan pengemudi. Untuk menghindari hal tersebut maka bina marga memberikan ketentuan bahwa panjang lengkung vertikal cekung adalah tidak melebihi dari perkalian A dengan V^2 dibagi 3480.

$$L \leq \frac{A \cdot V^2}{3480}$$

Selain prasyarat di atas, pada lengkung vertikal cekung yang kelandaian kecil menyebabkan panjang lengkungan menjadi pendek, sehingga akan tampak curam dan berbahaya bagi segi keamanan. Untuk mengatasi hal tersebut bina marga menentukan panjang lengkung vertikal cekung diambil $\geq$ dari 3 detik perjalanan.

$$L_l \geq V \cdot 0{,}833333\ meter$$

Dari penjelasan-penjelasan diatas, maka dapat diketahui bahwa *variable*-variabel yang menjadi *input*an perhitungan adalah :

a. Kelandaian pada sisi kiri lengkungan vertikal (g_1)

b. Kelandaian pada sisi kanan lengkungan vertikal (g_2)

c. Panjang lengkung vertikal yang diproyeksikan pada bidang horizontal (L)

d. Kecepatan rencana suatu kendaraan (V)

Proses berikutnya adalah *input*an diperhitungkan dengan rumus-rumus diatas dan berdasarkan pada persyaratan-persyaratan yang telah dijabarkan.

Untuk lengkung vertikal cembung, pemilihan panjang lengkung pada bidang horizontal haruslah merupakan panjang terpanjang yang dibutuhkan dengan mempertimbangkan persyaratan drainase dan bentuk *visual* lengkung.

Untuk lengkung vertikal cekung, pemilihan panjang lengkung pada bidang horizontal haruslah merupakan panjang terpanjang yang dibutuhkan setelah mempertimbangkan kenyamanan pengendara dan faktor keamanan.

2. *Alignment* Horisontal

Hendra (1999 : 22) mengatakan bahwa "*Alignment* horizontal adalah proyeksi dari sumbu jalan pada bidang horizontal atau proyeksi horizontal dari sumbu jalan secara tegak lurus. *Alignment* horizontal terdiri dari garis-garis lurus yang dihubungkan dengan garis-garis lengkung. Garis lengkung tersebut terdiri dari busur lingkaran ditambah dengan busur peralihan dari lurus ke busur lingkaran.

Terdapat beberapa faktor yang mempengaruhi laju kendaraan pada *Alignment* horizontal, yakni gaya-gaya yang bekerja pada kendaraan maupun berat kendaraan itu sendiri. Apabila kendaraan melaju dengan kecepatan v pada bidang datar atau miring dengan lintasan berbentuk lengkung seperti lingkaran, maka pada kendaraan tersebut bekerja gaya kecepatan v dan gaya *sentrifugal*. Gaya *sentrifugal*

mendorong kendaraan keluar dari jalur secara radial dari lajurnya dengan arah tegak lurus terhadap kecepatan v. gaya *sentrifugal* dirumuskan sebagai.

$$F = m \times a$$

Dimana :

F : Gaya *sentrifugal*

m : masa kendaraan dengan nilai G/g

G : berat kendaraan

g : gaya gravitasi bumi yakni (9,81 m/det^2)

a : percepatan *sentrifugal* dengan nilai v^2/R

v : kecepatan kendaraan

R : jari-jari lingkaran lintasan

Dengan demikian besarnya gaya *sentrifugal* dapat ditulis sebagai berikut :

$$F = \frac{G}{g}\frac{v^2}{R}$$

Untuk menjaga agar kendaraan tersebut bergerak tetap pada lajurnya maka perlu adannya suatu gaya yang dapat mengimbangi gaya tersebut sehingga terjadi suatu kesetimbangan.

Gaya yang mengimbangi gaya *sentrifugal* tersebut dapat berasal dari :

a. Gaya gesekan melintang antara ban kendaraan dengan permukaan jalan (Fs).
b. Komponen berat akibat kemiringan permukaan jalan.

Gaya gesekan melintang adalah besarnya gesekan yang timbul antara ban dan permukaan jalan dalam arah melintang jalan yang berfungsi untuk mengimbangi gaya *sentrifugal*. Perbandingan antara gaya gesekan melintang dengan gaya normal yang bekerja disebut juga sebagai koefisien gesekan melintang.

Besarnya koefisien gesekan melintang dipengaruhi oleh beberapa Faktor seperti jenis dan kondisi ban, tekanan ban, kekasaran permukaan perkerasan, keceaptan kendaraan dan keadaan cuaca. Koefisien gesekan melintang untuk perencanaan atau perancangan jalan secara matematis dapat dihitung dengan :

a. Untuk Vrencana < 80 km/jam : $f = -0{,}00065\ V\ + 0{,}192$
b. Untuk Vrencana antara 80-112 km/jam : $f = -0{,}00125\ V + 0{,}24$

Sedangkan komponen berat kendaraan untuk mengimbangi gaya *sentrifugal* dapat diperoleh dengan membuat kemiringan melintang permukaan jalan. Kemiringan melintang tersebut dapat juga disebut sebagai *superelevasi*. Semakin besar *superelevasi* maka semakin besar pula komponen berat kendaraan yang diperoleh.

Superelevasi makasimum yang dapat dipergunakan pada suatu jalan raya dibatasi oleh beberapa keadaan seperti :

a. Keadaan cuaca, *superelevasi* maksimum yang dipilih dipengaruhi juga oleh sering dan banyaknya curah hujan yang turun.
b. Jalan yang berada di daerah berkabut memiliki *superelevasi* yang lebih rendah dari pada daerah yang bercuaca baik.
c. Keadaan medan, pada daerah data *superelevasi* maksimum dapat dipilih lebih tinggi dari daerah daerah yang berbukit-bukit. Dalam hal ini pemilihan

superelevasi maksimum dipengaruhi oleh faktor kerumitan dalam sisi pembuatan dan pengerjaan jalan.

d. Keadaan lingkungan, di wilayah perkotaan kendaraan lebih bergerak perlahan-lahan, banyak teradapat persimpangan, arus lalu lintas yang lebih padat sehingga sebaiknya *superelevasi* maksimum diperkotaan dipilih serendah mungkin.
e. Komposisi jenis kendaraan dari arus lalu lintas. Banyaknya kendaraan berat yang bergerak lebih lambat mengakibatkan gerak arus lalu lintas tidak menentu. Pada kondisi ini sebaiknya dipilih *superelevasi* maksimum yang lebih rendah.

Terdapatnya faktor-faktor yang membatasi seperti yang disebutkan di atas menyebabkan keragaman nilai *superelevasi* maksimum yang diperbolehkan untuk setiap tempat. Dinas Bina Marga menganjurkan *superelevasi* maksimum 10% pada kecepatan rencana >30 km/jam dan 8% pada kecepatan rencana 30 km/jam untuk dipergunakan di wilayah luar perkotaan sedangkan untuk wilayah perkotaan sendiri menggunakan *superelevasi* maksimum 6%.

Oleh karena adanya gaya-gaya penyeimbang gaya *sentrifugal* yang terjadi tersebut maka rumus umum pada lengkung horizontal dapat dijelaskan sebagai berikut.

Gaya-gaya yang bekerja digambarkan seperti pada gambar 2.2 yaitu gaya *sentrifugal*, berat kendaraan G, dan gaya gesekan antara ban dan permukaan perkerasan Fs.

Gambar 2.2 Gaya-gaya yang bekerja pada lengkung horizontal

$$G.sin\alpha + Fs = \frac{G}{g}\frac{v^2}{R}cos\alpha$$

$$G.sin\alpha + f\left(G.cos\alpha + \frac{G}{g}\frac{v^2}{R}sin\alpha\right) = \frac{G}{g}\frac{v^2}{R}cos\alpha$$

$$G.sin\alpha + f.G.cos\alpha = \frac{G}{g}\frac{v^2}{R}(cos\alpha - f.sin\alpha)$$

$$G\frac{sin\alpha}{cos\alpha} + f.G = \frac{G}{g}\frac{v^2}{R}(1 - f.tan\alpha) \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ e = tan\alpha$$

$$G(e + f) = \frac{G}{g}\frac{v^2}{R}(1 - f.e)$$

$$\frac{e + f}{1 - e.f} = \frac{v^2}{g.R}$$

Karena nilai $e.f$ terlalu kecil, maka dapat diabaikan. Dengan demikian diperoleh rumus umum untuk lengkung horizontal sebagai berikut.

$$e + f = \frac{v^2}{g.R}$$

Jika v dinyatakan dalam km/jam dan g = 9,81 m/det^2 serta R dalam m, maka diperoleh :

$$e + f = \frac{v^2}{127.R}$$

Ketajaman lengkung horizontal dinyatakan dengan besarnya radius dari lengkung tersebut atau dengan besarnya derajat busur lengkung. Derajat lengkung adalah besarnya sudut lengkung yang menghasilkan panjang busur 25 meter. Semakin besar nilai R semakin kecil D dan semakin tumpul lengkung horizontal

rencana, semakin kecil R semkain besar D dan semakin tajam lengkung horizontal yang direncanakan. Sesuai dengan penjelasan dari gambar berikut.

Berarti :

$$D = \frac{25}{2\pi R} \times 360°$$

$$D = \frac{1432{,}29}{R}$$

Dari persamaan $e + f = v^2/127.R$ terlihat bahwa besarnya radius lengkung horizontal dipengaruhi oleh nilai e dan f serta kecepatan rencana yang ditetapkan. Ini berarti bahwa terdapat nilai radius minimum atau derajat lengkung maksimum untuk nilai *superelevasi* maksimum dan koefisien gesekan melintang maksimum. Lengkung tersebut dinamakan lengkung tertajam yang dapat direncanakan untuk satu nilai kecepatan tertentu. R minimum dapat ditentukan dengan mempergunakan rumus :

$$R\,\text{min} = \frac{v^2}{127(\,e\,maks + f\,maks\,)}$$

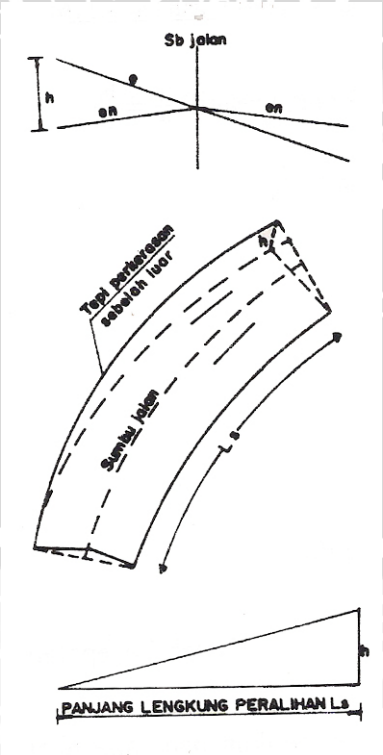
$$D\,maks = \frac{181913{,}53\,(\,e\,maks + f\,maks\,)}{v^2}$$

Nantinya nilai dari Rmin dan Dmaks tersebut digunakan bersamaan dengan nilai *superelevasi* maksimum untuk menentukan nilai panjang lengkung peralihan dan *superelevasi* yang dipergunakan dalam perancangan *Alignment* horizontal.

Pada lengkung horizontal yang tumpul dengan jari-jari besar, lintasan kendaraan masih dapat tetap berada pada lajur jalannya, tetapi pada tikungan tajam kendaraan akan menyimpang dari lajur yang disediakan mengambil lajur lain disampingnya. Guna menghindari hal tersebut maka dibuatkan sebuah lengkung dimana lengkung tersebut merupakan peralihan dari bagian *tangent* ke bagian lengkung horizontal. Lengkung ini disebut dengan lengkung peralihan.

Bentuk lengkung peralihan yang memberikan bentuk yang sama dengan jejak kendaraan ketika beralih dari jalan lurus ke tikungan busur dan sebaliknya, dipengaruhi oleh kecepatan rencana kendaraan, radius lengkung dan kemiringan melintang jalan.

Panjang lengkung peralihan menurut bina marga diperhitungkan sepanjang mulai dari penampang melintang berbentuk crown sampai penampang melintang dengan kemiringan sebesar *superelevasi*.



Gambar 2.3 lengkung peralihan menurut Bina Marga.

Proses pencapaian kemiringan melintang dari kemiringan melintang normal jalan hingga kemiringan melintang sebesar *superelevasi* pada lengkung berbentuk busur lingkaran, menyebabkan peralihan tinggi perkerasan sebelah luar dari jalan. Peralihan tinggi tersebut disebut sebagai landai relative $(1/m)$. Menurut bina marga landai relative dirumuskan sebagai berikut.

$$\frac{1}{m} = \frac{h}{L_s}$$

$$\frac{1}{m} = \frac{(e + e_n)B}{L_s}$$

Dimana :

$1/m$: Landai Relatif

L_s : Panjang lengkung peralihan

B : Lebar jalur 1 arah (m)

e : *Superelevasi*

e_n : Kemiringan melintang normal jalan

Bina marga memberikan nilai kelandaian relative maksimum berdasarkan empiris sesuai dengan table berikut.

Kecepatan rencana km/jam	Kelandaian relatif maksimum
20	1/50
30	1/75
40	1/100
50	1/115
60	1/125
80	1/150

Table 2.4 kelandaian relative maksimum menurut Bina Marga

Menurut Bina Marga, dengan menggunakan nilai-nilai dari Dmaks, Rmin dan *superelevasi* maksimum maka dapat ditentukan panjang dari lengkung peralihan (Ls) yang diperkenankan melalui table berikut.

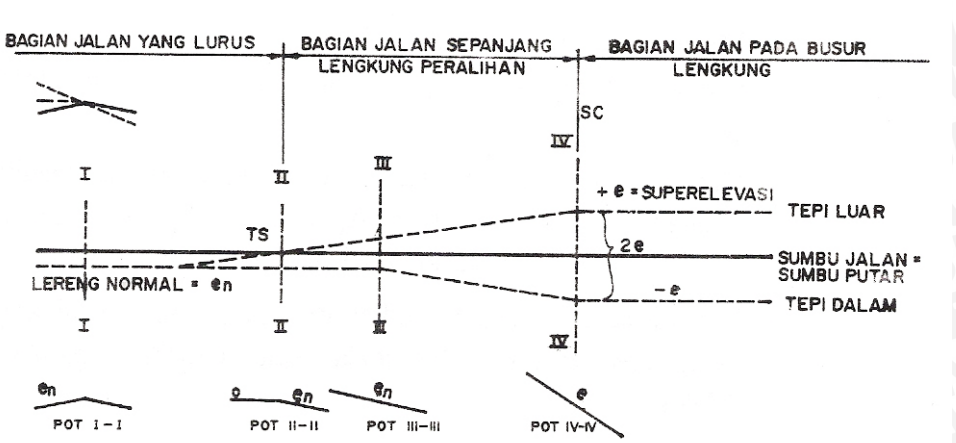
D (o)	R (m)	V = 50 km/jam e	Ls	V = 60 km/jam e	Ls	V = 70 km/jam e	Ls	V = 80 km/jam e	Ls	V = 90 km/jam e	Ls
0,250	5730	LN	0	LN	0	LN	0	LN	0	LN	0
0,500	2865	LN	0	LN	0	LP	60	LP	70	LP	75
0,750	1910	LN	0	LP	50	LP	60	0,020	70	0,025	75
1,000	1432	LP	45	LP	50	0,021	60	0,027	70	0,033	75
1,250	1146	LP	45	LP	50	0,025	60	0,033	70	0,040	75
1,500	955	LP	45	0,023	50	0,030	60	0,038	70	0,047	75
1,750	819	LP	45	0,026	50	0,035	60	0,044	70	0,054	75
2,000	716	LP	45	0,029	50	0,039	60	0,049	70	0,060	75
2,500	573	0,026	45	0,036	50	0,047	60	0,059	70	0,072	75
3,000	477	0,030	45	0,042	50	0,055	60	0,068	70	0,081	75
3,500	409	0,035	45	0,048	50	0,062	60	0,076	70	0,089	75
4,000	358	0,039	45	0,054	50	0,068	60	0,082	70	0,095	75
4,500	318	0,043	45	0,059	50	0,074	60	0,088	70	0,099	75
5,000	286	0,048	45	0,064	50	0,079	60	0,093	70	0,100	75
6,000	239	0,055	45	0,073	50	0,088	60	0,098	70	Dmaks = 5,12	
7,000	205	0,062	45	0,080	50	0,094	60	Dmaks = 6,82			
8,000	179	0,068	45	0,086	50	0,098	60				
9,000	159	0,074	45	0,091	50	0,099	60				
10,000	143	0,079	45	0,095	60	Dmaks = 9,12					
11,000	130	0,083	45	0,098	60						
12,000	119	0,087	45	0,100	60						
13,000	110	0,091	50	Dmaks = 12,79							
14,000	102	0,093	50								
15,000	95	0,096	50								
16,000	90	0,097	50								
17,000	84	0,099	60								
18,000	80	0,099	60								
19,000	75	Dmaks = 18,85									

Table 2.5 Ls dan e yang dibutuhkan (e maksimum = 10%) metode Bina Marga

D (o)	R (m)	V = 50 km/jam e	Ls	V = 60 km/jam e	Ls	V = 70 km/jam e	Ls	V = 80 km/jam e	Ls	V = 90 km/jam e	Ls
0,25	5730	LN	0	LN	0	LN	0	LN	0	LN	0
0,50	2865	LN	0	LN	0	LN	60	LP	70	LP	75
0,75	1910	LN	0	LP	50	LP	60	LP	70	0,025	75
1,00	1432	LP	45	LP	50	LP	60	0,026	70	0,032	75
1,25	1146	LP	45	LP	50	0,025	60	0,031	70	0,038	75
1,50	955	LP	45	0,022	50	0,029	60	0,036	70	0,045	75
1,75	819	LP	45	0,025	50	0,033	60	0,041	70	0,050	75
2,00	716	LP	45	0,028	50	0,037	60	0,046	70	0,055	75
2,50	573	0,025	45	0,034	50	0,044	60	0,054	70	0,064	75
3,00	477	0,029	45	0,040	50	0,050	60	0,060	70	0,070	75
3,50	409	0,033	45	0,045	50	0,056	60	0,065	70	0,075	75
4,00	358	0,037	45	0,049	50	0,061	60	0,071	70	0,079	75
4,50	318	0,041	45	0,053	50	0,064	60	0,074	70	0,080	75
5,00	286	0,044	45	0,057	50	0,068	60	0,077	70	Dmaks = 4,67	
6,00	239	0,050	45	0,063	50	0,074	60	0,080	70		
7,00	205	0,056	45	0,068	50	0,078	60	Dmaks = 6,25			
8,00	179	0,060	45	0,073	50	0,080	60				
9,00	159	0,064	45	0,076	50	Dmaks = 8,43					
10,00	143	0,068	45	0,078	50						
11,00	130	0,071	45	0,079	50						
12,00	119	0,074	45	Dmaks = 11,74							
13,00	110	0,076	45								
14,00	102	0,078	45								
15,00	95	0,079	45								
16,00	90	0,080	45								
17,00	84	0,080	45								
		Dmaks = 17,47									

Table 2.6 Ls dan e yang dibutuhkan (e maksimum = 8%) metode Bina Marga

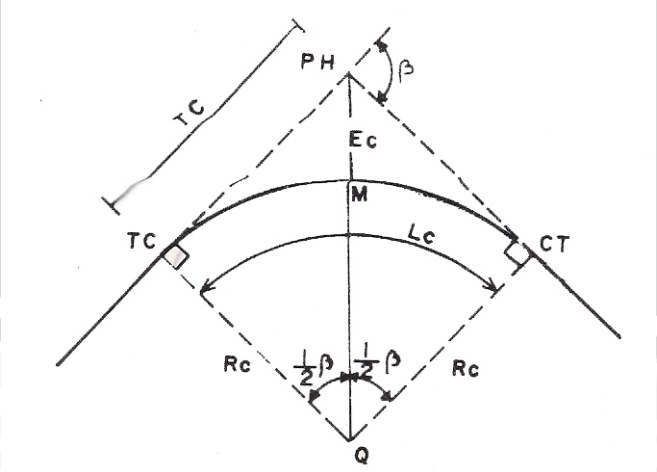
Diagram yang mengambarkan *superelevasi* dari pencapaian *superelevasi* dari lereng normal ke *superelevasi* penuh disebut juga sebagai diagram *superelevasi*. Dengan menggunakan diagram *superelevasi* dapat ditentukanbentuk penampang melintang pada suatu titik di lengkung horizontal yang direncanakan. Diagram *superelevasi* digambar berdasarkan elevasi sumbu jalan sebagai garis nol. Elevasi tepi perkerasan diberi tanda positif atau negative ditinjau dari ketinggian sumbu jalan.



Gambar 2.4 diagram *superelevasi* dengan sumbu jalan sebagai sumbu putar.

Pada perencanaan lengkung horizontal, terdapat tiga macam bentuk yang dapat dipergunakan. Yakni bentuk busur lingkaran (*circle*), bentuk busur lingkaran dengan lengkung peralihan (spiral-*circle*-spiral) dan bentuk lengkung peralihan saja (spiral-spiral).

Pada bentuk busur lingkaran, hanya lengkung dengan radius saja yang diperbolehkan menggunakan bentuk ini. Selain menggunakan radius yang besar, bentuk ini juga membutuhkan *superelevasi* yang bernilai kurang atau sama dengan 3%. Berikut gambar bentuk lengkungan busur lingkaran.



Gambar 2.5 lengkung busur lingkaran

Berikut adalah rumus-rumus yang dipergunakan dalam perhitungan pada lengkung busur lingkaran.

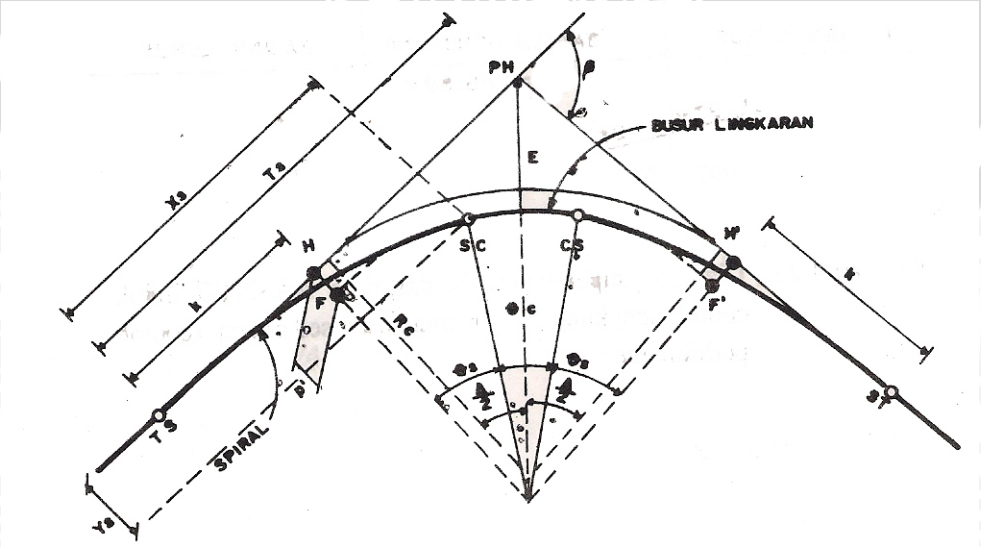
$$T_c = R_c \cdot \tan\frac{1}{2}\beta$$

$$E_c = \frac{R_c\left(1 - cos\frac{1}{2}\beta\right)}{cos\frac{1}{2}\beta}$$

$$L_c = \frac{\beta\pi}{180} R_c$$

Karena lengkung hanya berbentuk busur lingkaran saja, maka pencapaian *superelevasi* dilakukan sebagian pada jalan lurus dan sebagian pada bagian lengkung. Bina marga menempatkan ¾ L_s' dibagian lurus dan $1/4L_s'$ dibagian lengkung.

Bentuk umum yang sering digunakan dalam perencanaan *Alignment* horizontal adalah bentuk spiral-*circle*-spiral yang simetris.



Gambar 2.6 lengkung spiral-*circle*-spiral

Lengkung TS-SC adalah lengkung peralihan berbentuk spiral yang menghubungkan bagian lurus dengan bagian *circle*. Guna membuat ruangan untuk spiral sehingga lengkung lingkaran ditempatkan di ujung lengkung spiral, maka lengkung lingkaran tersebut digeser ke dalam pada posisi FF', dimana HF'=FF'=P terletak sejauh k dari awal lengkung peralihan.

Berikut rumus-rumus yang dipergunakan pada lengkung spiral-*circle*-spiral.

$$\theta_s = \frac{L_s \cdot 90}{\pi \cdot R}$$

$$\theta_c = \beta - 2\theta_s$$

$$L_c = \frac{\theta_c}{360} \cdot 2\pi R_c$$

$$L = L_c + 2L_s$$

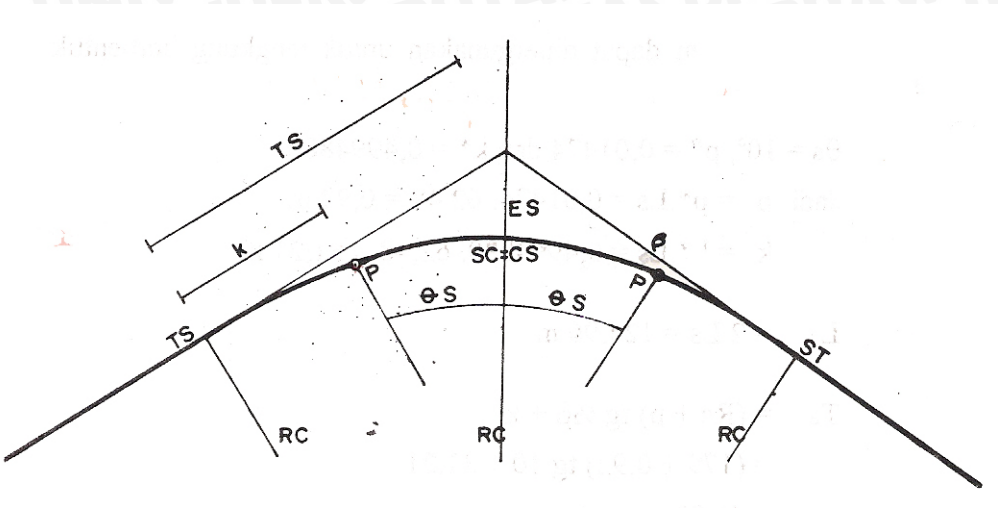
$$p = \frac{{L_s}^2}{6R_c} - R_c(1 - cos\theta_s)$$

$$k = L_s - \frac{{L_s}^3}{40{R_c}^2} - \qquad sin\theta_s$$

$$E_s = (R_c + p)sec\tfrac{1}{2}\beta - R_c$$

$$T_s = (R_c + p)tan\tfrac{1}{2}\beta + k$$

Pada bentuk lengkung spiral-spiral, tidak terdapat lengkungan yang berupa busur lingkaran, sehingga titik SC berimppit pada titik CS. Rumus-rumus yang dipergunakan dalam perhitungan bentuk ini serupa dengan bentuk spiral-*circle*-spiral dengan ketentuan nilai $L_c = 0$ dan $\theta_s = \frac{1}{2}\beta$. Berikut gambar dari bentuk spiral-spiral.



Gambar 2.7 lengkung spiral-spiral

Untuk kesemua bentuk, dalam perhitungan dipergunakan table 2.5 dan 2.6 untuk menentukan *Superelevasi* (e) dan panjang lengkung peralihan (L_s). Sesuai dengan rumus-rumus perhitungan yang dijabarkan diatas, maka *variable* yang menjadi *input*an dalam perencanaan *Alignment* horizontal adalah :

a. Kecepatan rencana kendaraan
b. e maksimum
c. sudut β
d. lebar jalan
e. kemiringan melintang normal jalan
f. dan Rc

serta dalam proses perhitungannya dipergunakan persyaratan-persyaratan yang ditetapkan oleh bina marga.

E. Bahasa Pemrograman Berorientasi Objek

Setelah era pemrograman tersturktur, bahasa pemrograman mengalami perkembangan hingga mencapai pemrograman berorientasi objek. Perograman berorientasi objek menggantikan pemrograman terstruktur karena mempunyai banyak keunggulan dalam menangani proyek yang luar biasa kompleknya. Pemrograman ini menawarkan fleksibilitas, kegunaulangan dan kemudahan perawatan. Segala sesuatu di java kecuali sedikit tipe dasar (int, float, double, char) adalah objek. Keunggulan bahasa java ketimbang bahasa C++ adalah :

a. Perancang java menghilangkan keperluan dealokasi manual. Java dilengkapi garbage collector yang bertugas mendealokasi *memori* yang tidak terpakai.
b. Java menerapkan array sebenarnya, menghilangkan keperluan aritmatika *pointer* yang berbahaya dan berpeluang besar menyebabkan kesalahan.
c. Menghilangkan ketidak – mungkinan operasi penugasan (*assigment*) sebagai pengujian atas kesamaan di kalimat bersyarat.
d. Menghilangkan pewarisan jamak (*multiple inheritance*) diganti menjadi fasilitas *interface*.

Adapun fitur penting utama yang ditawarkan oleh java adalah bahasa ditujukan untuk membuat beragam aplikasi secara seragam, yaitu :

a. Pemrograman di lingkungan *web browser*, pemrograman ini menggunakan bahasa java *applet*. Program tersebut dieksekusi di *web browser* dari halaman *web* yang memuat java *applet*. *Web browser* kemudian menugaskan Java *Interpreter* (JRE, *Java Runtime Environtment*) untuk mengeksekusi java *applet* yang diterima. Java *applet* memungkinkan *web* menjadi sarana interaktif di mana halaman *web* dapat berinteraksi terhadap masukan.

b. Pemrograman di lingkungan *web server*, terdapat dua jenis bahasa java yang digunakan dalam aplikasi *server*, yakni java *server* pages (JSP) dan java *Servlet*. JSP adalah *web sripting* yang serupa dengan ASP, PHP dan sebagainya. Program tersebut ditempelkan pada halaman html. Html ini tidak langsung dikirim ke *web browser* melainkan diolah dulu oleh *web server* dan hasilnya yang berupa dokumen html dikirim *web server* ke *web browser*. Sedangkan Java *Servlet* adalah semacam modul di *web server*. JSP akan diterjemahkan menjadi *servlet* agar mempercepat proses eksekusi.

c. Program mandiri, java merupakan bahasa yang bermaksud umum (*general – purpose language*) untuk mengembangkan semua jenis program yang dapat dijalankan di komputer bersistem operasi apapun asalkan ada java *interpreter* di dalam *platform* itu. Sebuah program java yang diciptakan di-*setting* menghasilkan *file* yang bertipe teks dan berekstensi .java. Program ini dikompilasi menghasilkan satu *file bytecode* berektensi .*Class* atau lebih.

d. Bahasa untuk pengembangan aplikasi objek – objek tersebar skala *enterprise*. Terdapat tekhnologi java untuk mengembangkan komponen yaitu EJB (*Enterprise* Java *Beans*).

Keunggulan – keunggulan dari penggunaan bahasa pemrograman java adalah sebagai berikut :

a. Bahasa yang digunakan adalah bahasa sederhana.

b. Bahasa berorientasi objek.

c. Bahasa *statically typed*, merupakan bahasa yang harus mendeklerasikan lebih dahulu sebelum digunakan. Hal ini memungkinkan kompilator untuk

menentukan dan melaporkan terjadinya pertentangan tipe sehingga menjadi barikade awal pencegahan kesalahan.

d. Bahasa dikompilasi, sebelum program dijalankan terlebih dahulu dikompilasi menggunakan java *compiler* yang menghasilkan *file bytecode* yang serupa dengan *file* kode mesin. Sehingga program akan lebih berjalan cepat mendekati bahasa mesin.
e. Bahasa yang aman.
f. Bahasa independen terhadap berbagai *platform.*
g. Bahasa *multithreading*.
h. Bahasa yang didukung *garbage collector*.
i. Bahasa yang tangguh, tidak menyebabkan *crash* pada sistem.
j. Bahasa yang mampu diperluas, mampu untuk ditulis dalam bahasa lain.
k. Bahasa yang memberikan dukungan berupa berbagai macam *library*, baik itu dari segi GUI ataupun dari segi logika dan aritmatiknya.

Dari kelebihan – kelebihan tersebutlah maka penulis lebih memilih bahasa java dalam penggunaan pembuatan aplikasi perhitungan ini.

1. Kelas dan Objek

Kelas dapat didefinisikan sebagai cetak biru (*blue print*) atau *prototipe* yang mendefinisikan variabel – veriabel dan *method – method* umum dari sebuah objek tertentu. Variabel – variabel yang dimaksud adalah atribut – atribut yang digunakan objek di dalam kelas sebagai penunjuk dan pembeda dengan kelas lainnya, sedangkan *method* adalah fungsi – fungsi di dalam kelas tersebut. Pada saat pembuatan kelas baru, kelas tersebut masih bersifat abstrak. Karena masih belum

berisi variabel dan *method* untuk mendeskripsikan suatu objek. Dengan kata lain kelas adalah pola (*template*) untuk pembuatan objek, sedangkan objek adalah wujud nyata (*instance*) dari sebuah kelas. Dapat dimisalkan bahwa manusia adalah sebuah kelas, sedangkan contoh objek atau wujud nyata dari kelas manusia adalah si Udin, Tono, Dewi, dan lainnya.

Dalam bahasa java, kelas didefiniskan dengan menggunakan kata kunci *Class*. Berikut bentuk umum yang digunakan untuk mendefinisikan sebuah kelas.

```
Class NamaKelas {
    Tipe data1;
    Tipe data2;
    ...
    Tipe dataN;
    Tipe method1(daftar-parameter) {
        //kode untuk method1
    }
    Tipe method2(daftar-parameter) {
        //kode untuk method2
    }
    ...
    Tipe method1(daftar-parameter) {
        //kode untuk method1
    }
}
```

Data atau variabel yang di definisikan di dalam kelas sering disebut dengan *instance variable*. Nilai dari data – data tersebut selanjutnya akan diakses eoleh *method* – *method* yang ada. Dengan demikian *method* sebenarnya digunakan sebagai antar muka (*interface*) antara *user* (pemakai) dengan data – data yang ada di dalam kelas. Data dan *method* yang tergabung di dalam suatu kelas sering disebut sebagai *Class* members (anggota kelas).

Objek adalah suatu kelas dimana di dalamnya sudah terdapat *method* – *method* dan variabel – variabel sebagai pembeda dengan kelas lainnya.

Seperti yang telah diutarakan sebelumnya, ketika kelas baru dideklarasikan, maka yang terjadi hanyalah terbentuknya tipe data baru, bukan membuat objek baru. Untuk mendeklarasikan objek baru dari tipe kelas yang telah dideklarasikan sebelumnya, secara *eksplisit* perlu dilakukan dua tahap, yakni :

a. Pendeklarasian *variable* yang digunakan sebagai referensi ke objek dari kelas bersangkutan.
b. Penginstansian kelas dengan menggunakan operator *new* dan memasukkan *instance*-nya ke dalam variabel referensiyang baru saja dideklarasikan. Operator *new* secara dinamis akan mengalokasikan ruang *memori* untuk menyimpan suatu objek tertentu dan mengembalikan nilai berupa referensi ke objek bersangkutan. Referensi objek yang dimaksud adalah berupa alamat *memori* dari objek tersebut. Berikut adalah *method* pendeklarasian dari dua tahapan diatas.

//Mendeklarasikan variabel k bertipe objek

Objek k;

//Melakukan instansiasi dan memasukkan referensi ke variabel k

K = new Objek( );

2. Pewarisan Sifat Objek

Di dalam java diperbolehkan pembuatan kelas baru yang bersifat umum dan kemudia dari kelas tersebut diturunkan menjadi kelas baru yang bersifat lebih spesifik. Kelas induk yang diturunkan disebut sebagai super*Class*s, sedangkan kelas baru hasil turunan dari kelas induk adalah *subClass*.

Pada proses penurunan ini, kelas turunan akan mewarisi sifat – sifat yang terdapat pada kelas induknya dan dapat pula memiliki sifat – sifat lain yang tidak dimiliki induknya tanpa meninggalkan sifat warisan induknya. Dari kelas – kelas baru turunan tersebut dapat diturunkan lagi menjadi kelas yang lebih spesifik lagi.

Java menyediakan *sintaks extend* untuk melakukan proses penurunan terhadap suatu kelas. Bentuk umum dari penggunaan *sintaks* tersebut adalah :

Class nama-subClass extends nama-superClass {

//badan kelas

}

3. Paket dan *Interface*

Java menyediakan suatu sarana untuk pengelompokan kelas – kelas dan *interface* yang saling berkaitan menjadi satu unit tunggal yakni paket. Dengan paket

konflik yang timbul akibat penamaan kelas dapat terhindar. Fasilitas paket merupakan mekanisme penamaan dan kendali akses, yang dimaksud dengan kendali akses adalah kelas – kelas dalam paket dapat didefinisikan tidak boleh diakses kode di luar peket, kelas – kelas tersebut juga dapat didefinisikan hanya boleh diakses oleh kelas lain di dalam paket yang sama. Dengan cara ini memungkinkan kelas – kelas di satu paket untuk saling kenal, akan tetapi tidak untuk kelas – kelas di luar paket.

Penciptaan paket dilakukan dengan penyertaan statement *package* di kalimat pertama *file* sumber. Kelas – kelas yang dideklarasikan di *file* akan menjadi milik paket tersebut. *Sintaks* untuk pernyataan paket adalah sebagai berikut.

package Identifier;

pernyataan paket harus diletakkan di awal unit kompilasi. Setiap keals dan *interface* di unit kompilasi dalam pernyataan paket akan dianggap sebagai bagian paket. Pemilihan nama paket harus dilakukan secara hati – hati karena penggantian nama paket tidak bisa dilakukan tanpa mengganti direktori di mana kelas – kelas itu disimpan.

Penggunaan kelas atau *interface* di paket berbeda dapat dilakukan dengan terlebih dahulu menyatakan :

a. Kalimat *import* kelas.

   Kelas – kelas dalam paket dapat di-*import* dari paket lain secara individu ataupun secara keseluruhan. *Sintaks* untuk statement *import* adalah.

Import Identifier;

Identifier adalah nama kelas atau paket yang ingin di-*import*. Peng*import*an kelas secara keseluruhan dapat dilakukan dengan menambahkan * kepada *sintaks import*. Misalkan untuk melakukan *import* semua kelas di paket *java.awt* maka *sintaks*nya dituliskan sebagai berikut.

Import java.awt.;*

Bentuk *import* dengan * dapat meningkatkan waktu kompilasi, khususnya ketika melakukan *import* beberapa paket yang berukuran besar. Bentuk *import* dengan * tidak akan berdampak pada kinerja waktu proses program berjalan nantinya ataupun pada ukuran hasil kompilasi.

b. Atau dengan mengacu letak kelas atau *interface* yang digunakan secara lengkap.

   Cara lain peng*import*an kelas atau *interface* di paket adalah dengan pengacuan paket secara *eksplisit* mengacu pada nama paket setiap menggunakan kelas atau *interface* yang di*import*. Statement yang digunakan dalam peng*import*an bentuk ini adalah.

Identifier nama_kelas;

Misalkan penggunaan kelas color yang di-*import* dari paket *java.awt*, maka *sintaks*nya yang diketikkan menjadi.

java.awt.color color;

Pernyataan diatas berarti, kelas *color* yang digunakan mengacu pada kelas *color* di paket *java.awt*. Cara ini sangat tidak efisien karena terlalu panjang, namun cara tersebut mutlak dilakukan bila terdapat dua paket atau lebih mempunyai nama kelas yang sama.

Java menyediakan banyak level proteksi untuk memungkinkan beragam kendali pengaksesan bagi variabel dan metode di dalam kelas, subkelas dan paket. Level proteksi yang diberikan kepada variabel dan *method* (anggota kelas) relatif baik terhadap kelas itu sendiri maupun kelas lain yang mengaksesnya. Adapun hak – hak akses yang dapat diberikan pada suatu kelas dan subkelas adalah sebagai berikut :

default *private* *protected* *public*

Yang di maksud dengan hak akses default disini adalah pemberian level proteksi pada variabel atau *method* tanpa mengetikkan *sintaks* apapun pada variabel atau *method* tersebut.

Di dalam paket, setiap kelas dan subkelas diberikan hak akses yang beragam sesuai dengan kebutuhan. Adapaun hak – hak akses tersebut dijabarkan melalui tabel berikut.

Tabel 2.7 Hak Akses Kelas dan Subkelas

no	Kategori Kelas	Default	Private	Protected	Public
1	Kelas yang sama	Ya	Ya	Ya	Ya
2	Subkelas di paket yang sama	Ya	Tidak	Ya	Ya
3	Bukan subkelas di paket yang sama	Ya	Tidak	Ya	Ya
4	Subkelas di paket berbeda	Tidak	Tidak	Ya	Ya
5	Bukan subkelas di paket berbeda	Tidak	Tidak	Tidak	Ya

Interface adalah *prototipe* untuk kelas dan berguna ditinjau dari perspektif rencana logis. *Interface* hampir sama dengan kelas abstrak namun berbeda. *Interface* adalah kelas abstrak yang sepenuhnya tidak diimplementasikan, yang berarti tidak ada metode yang diimplementasikan dan hanya terdapat data anggota yang bersifat final static yang berarti konstanta murni.

Interface menyediakan sarana mendefinisikan protokol kelas tanpa khawatir terhadap rincian – rincian implementasi. Begitu *interface* telah dirancang, maka pengembangan kelas dapat dilakukan tanpa khawatir mengenai komunikasi antar kelas.

Interface berisi kumpulan nama metode tanpa perlu mengimplementasikan metode tersebut. *Interface* selalu hanya mendeklarasikan metode abstrak dan *variable* final, serta tidak mendefinisikan metode – metode ini. Ketika sebuah metode ditulis ke dalam *interface* maka badan metode kosong. Kelas – kelas yang mengimplementasikan *interface* ini bertanggung jawab untuk menspesifikasikan implementasi metode – metode ini. Secara garis besar *interface* memungkinkan pemrogram mendefinisikan sekumpulan fungsionalitas tanpa merinci cara fungsionalitas didefinisikan. Contoh dari bentuk *interface* adalah sebagai berikut.

Jika satu kelas mengimplemetnasikan *interface java.lang.Runnable* maka kelas itu pasti harus mengimplementasikan metode *run ( )*. Karena itu, Java *Virtual Machine* dapat memastikan kelas yang mengimplementasikan *Runnable* mempunyai metode *run ( )*, maka JVM memastikan dapat memanggil metode *run()*. Bagi JVM, tidak perlu mengetahui apapun mengenai apa yang terjadi di metode *run()*. Yang terpenting bagi JVM adalah kelas mengimplementasikan *interface Runnabel*.

4. Pemrograman Grafik

Java telah menyediakan sejumlah objek-objek *visual* di dalam pemrograman, akan tetapi masih banyak bentuk-bentuk geometri yang di[erlukan oleh *user* yang tidak dimiliki java. Oleh karena itu java menyediakan beberapa macam tipe kelas yang mendukung dalam pembuatan objek sesuai dengan kebutuhan. *Class* tersebut antara lain :

a. *Class Canvas*

Adalah kelas yang disediakan oleh java untuk melakukan penggambaran objek geometri, semisal menggambar garis, kotak, lingkaran dan sebagainya. Java mendefinisikan *Class canvas* dengan kelas abstrak sehingga untuk menggunakannya diperlukan penurunan kelas menjadi kelas baru yan berisi instruksi penggambaran dengan cara override *method paint(graphics)* terlebih dahulu, baru kemudian kelas baru tersebut digunakan dalam aplikasi. *Sintaks* yang digunakan adalah.

```
[public|private] Class NamaClass extends Canvas
{
        Public void paint (Graphics g)
        {
          Instruksi penggambaran
        }
}
```

Selanjutnya turunan dari kelas *canvas* tersebut dipanggil dalam kelas utama program.

b. *Class* Grafik

Kelas ini berisi sejumlah instruksi yang digunakan untuk menggambar objek geometri. Instruksi-instruksi yang ada di dalam kelas ini bias dianggap sebagai instruksi dasar untuk proses penggambaran. Terdapat berbagai macam fitur yang disediakan dalam kelas ini, yang antara lain adalah *void drawArc*,

void drawLine, void drawOval, void drawPolygon, void drawPolyline, void drawRect, void draw roundRect, dan berbagai macam lainnya.

5. Penjebakan Eksepsi

Eksepsi adalah bentuk representasi dari kesalahan – kesalahan yang terjadi pada saat program sedang berjalan. Eksepsi juga dapat didefinisikan sebagai suatu objek yang dibuat pada saat program mengalami suatu kondisi yang tidak wajar. Pada saat eksepsi terjadi maka dapat dikatakan eksepsi telah dibangkitkan (*thrown*), untuk mengatasi hal tersebut maka perlu dilakukan proses penangkapan eksepsi untuk mencegah terjadi kegagalan proses pada program.

Terdapat lima buah kata kunci yang disediakan java untuk menangani eksepsi, yakni *try, catch, throw, throws,* dan *finally*. Kata kunci *try* digunakan untuk membuat blok yang berisi statement – statement yang mungkin menimbulkan eksepsi. Apabila dalam proses eksekusi runtunan statement tersebut terjadi sebuah eksepsi, maka eksepsi akan dilempar ke bagian blok penangkap yang dibuat dengan kata kunci *catch*. Pada kasus – kasus tertentu, terkadang dipergunakan pelemparan eksepsi secara manual. Untuk itu dapat dipergunakan kata kunci *throw*. Apabila apabila diinginkan pembangkitan sebuah eksepsi tanpa menuliskan blok *try,* maka kita perlu menambahkan kata kunci *throws* pada saat pendeklarasian *method*s. Dalam mendefinisikan blok *try* kita juga diperkenankan untuk menambahkan statement tambahan dengan mempergunakan kata kunci *finally* yang nantinya pasti akan dieksekusi baik terjadi eksepsi maupun tidak. Berikut adalah bentuk umum penanganan eksepsi dalam java.

Try {

// kumpulan statement yang mungkin menimbulkan eksepsi

*} catch (*TipeEksepsi1 objekEksepsi1*) {*

*//*penanganan untuk tipe eksepsi1

*} catch (*TipeEksepsi2 objekEksepsi2*) {*

*//*penanganan untuk tipe eksepsi2

Finally {

*//*statement tambahan yang pasti akan dieksekusi

}

6. *Input* dan *Output*

Proses *input* dan *output* adalah hal yang paling sering dijumpai pada sebagian besar program yang membutuhkan data – data eksternal. Java memberikan dukungan terhadap proses *input* dan *output* dengan menghadirkan paket java.io. Di dalam paket tersebut tersimpan banyak kelas dan *interface* siap pakai yang memudahkan kita sebagai programer, dalam pengambilan dan penyimpanan informasi dari atau ke media lain.

Program java melakukan proses I/O melalui *stream. Stream* adalah sebuah abstraksi yang dapat memberikan atau mendapatkan informasi. Sebuah *stream input* dapat mengabstraksikan beberapa tipe peralatan fisik seperti *keyboard*, *file*, atau *socket* jaringan. Begitu pula dengan *stream output* yang dapat dihubungkan dengan layar *console, file,* maupun koneksi jaringan. Dengan demikian *stream* akan memudahkan kita dalam melakukan proses I/O, karena kode program yang kita tulis

akan sama untuk masing – masing peralatan fisik yang dihubungkan dengan *stream* yang bersangkutan.

7. Pembuatan Aplikasi *Visual*

Dalam dunia nyata sebagian besar program – program yang dipergunakan oleh *user* adalah program berbentuk *visual*, atau dengan kata lain dapat juga disebut sebagai program berbentuk GUI (Grafik *User Interface*) yang lebih memudahkan penggunanya ketimbang program yang berbentuk *console*. Dalam java, terdapat dua paket yang dapat digunakan untuk mengembangkan program – program GUI, yakni AWT dan *Swing*. AWT (*Abstract Window Toolkit*) adalah sekumpulan *library* yang tidak tergantung pada *platform* dan digunakan untuk menyederhanakan implementasi *user-interface*. Sedangkan *Swing* adalah bentuk implementasi selanjutnya yang menambahakan komponen – komponen dalam sistem GUI, tetapi masih didasarkan pada arsitektur AWT. Jika dibandingkan dengan penggunaan AWT, penggunaan komponen – komponen *swing* lebih memakan resource yang lebih sedikit. Pada kenyataannya di lapangan, program – program GUI yang ada saat ini banyak dikembangkan dengan menggunakan komponen – komponen *swing*, karena kelebihannya dalam penghematan sistem resource.

Pada program GUI, setiap *user* melakukan pengisian karakter melalui keyboard maupun klik terhadap tombol *mouse*, hal ini dikatakan bahwa suatu *event* – *event* semacam ini dapat direspon atau ditangani oleh program untuk kemudian diproses sesuai dengan kebutuhan yang diinginkan. Sebagai contoh, pada saat *user* melakukan klik terhadap suatu tombol yang terdapat di dalam sebuah program, program akan menampilkan sebuah dialog atau form lain. Teknik atau cara untuk

melakukan pekerjaan tertentu pada saat suatu *event* terjadi sering dinamakan dengan *event-handling*.

Mekanisme *event-handling* yang terdapat pada java didasarkan pada delegation *event* model (model pendelegasian even), yang mendefinisikan secara standart dan konsisten untuk membangkitkan dan memproses suatu *event*. Konsepnya sederhana, *event* source akan membangkitkan suatu *event* tertentu dan mengirimkannya ke satu atau lebih *event listener*. Contoh *event* source adalah objek – objek yang terdapat di dalam *user interface* seperti *button*, *listbox*, *combobox*, dan sebagainya. Sedangkan contoh *event listener* adalah *interface keylistener*, *actionlistener*, *mousemotionlistener*, *itemlistener*, dan sebagainya.

F. Global Positioning System

Global possitioning system adalah suatu sistem navigasi satelite yang menggunakan setidaknya 24 satelit yang mengirimkan sinyal gelombang mikro ke bumi. Sinyal ini diterima oleh suatu alat penerima di permukaan bumi, dan digunakan untuk menentukan posisi kecepatan, koordinat, ketinggian, arah dan waktu.

Cara kerja sistem ini mempergunakan sejumlah satelit yang berada di orbit bumi yang memancarkan sinyalnya dan ditangkap oleh alat penerima di permukaan bumi. Terdapat tiga bagian penting dari sistem ini, yakni :

a. Bagian Kontrol

   Bagian ini berfungsi sebagai pengontrol, setiap satelit dapat berada sedikit di luar orbit, sehingga bagian ini melacak orbit satelit, lokasi, ketinggian dan kecepatan. Sinyal-sinyal diterima oleh bagian kontrol, dikoreksi dan dikirimkan

kembali ke satelit. Koreksi data lokasi yang tepat dari satelit ini disebut data *ephemeris* yang nantinya akan dikirimkan ke alat navigasi.

b. Bagian Angkasa

Bagian ini terdiri dari kumpulan satelit-satelityang berada di orbit bumi, sekitar 12.000 mil diatas permukaan bumi. Kumpulan satelit-satelit ini diatur sedemikian rupa sehingga alat navigasi setiap saat dapat menerima paling sedikit sinyal dari empat buah satelit. Sinyal satelit dapat melewati kaca, plastik dan awan tetapi tidak dapat melewati gedung atau gunung. Satelit mempunyai jam atom dan juga akan memancarkan informasi mengenai waktu / jam.

c. Bagian Pengguna

Bagian ini terdiri dari alat navigasi yang dipergunakan. Satelit akan memancarkan data almanak dan ephemeris yang akan diterima oleh alat navigasi secara teratur. Data almanak berisikan perkiraan lokasi satelit yang dipancarkan secara terus menerus oleh satelit. Data ephemeris dipancarkan oleh satelit dan valid untuk sekitar 4 – 6 jam. Untuk menunjukkan koordinat sebuah titik (dua dimensi) alat navigasi memerlukan sinyal paling sedikit dari 3 buah satelit. Untuk menunjukkan data ketinggian sebuah titik maka diperlukan satu buah satelit tambahan.

Dari sinyal-sinyal yang dipancarkan oleh kumpulan satelit tersebut, alat navigasi akan melakukan perhitungan-perhitungan dan hasil akhirnya adalah koordinat posisi alat tersebut. Makin banyak jumlah sinyal satelit yang diterima oleh sebuah alat, maka akan membuat alat tersebut menghitung koordinat posisinya lebih akurat.

Perhitungan suatu posisi koordinat oleh gps tidak dipengaruhi oleh topologi medan yang akan diukur. Apapun jenis medan yang dilalui oleh sebuah koordinat menuju koordinat lainnya tidak akan mengubah jarak antar titik koordinat tersebut, karena pada dasarnya gps hanya memperhitungkan jarak udara dari dua buah koordinat yang diperhitungkan. Hal ini menjadi suatu kelemahan dari penggunaan gps untuk pengukuran dan perencanaan pembangunan jalan, semakin bergelombang suatu dataran maka akan menyebabkan semakin besar perbedaan perhitungan jarak di lapangan dengan jarak pada gps.

Selain faktor perbedaan perhitungan tersebut, gps juga memiliki kelemahan dalam penentuan suatu titik koordinat, gps akan mengambil titik koordinat secara acak dengan area yang berbentuk lingkaran dengan jari-jarinya adalah besar kesalahan akurasi dari gps tersebut. Jadi semakin kecil tingkat akurasi gps maka akan semakin acak gps tersebut mengambil suatu posisi. Pada sistem yang akan dirancang ini, untuk meminimalisir pengambilan acak suatu titik koordinat oleh gps, maka sistem akan mengambil empat posisi koordinat dari pengambilan oleh gps yang sama, nantinya dari keempat posisi tersebut dirata-rata untuk diambil nilai yang mendekati koordinat sebenarnya. Untuk pengukuran yang memerlukan akurasi tinggi disarankan menggunakan gps dengan tingkat akurasi hingga satuan centimeter ataupun milimeter.

Dari kelemahan-kelemahan penggunaan gps dalam sistem, terdapat kelebihan yang menjadi kompensasi untuk menutupi kekurangan tersebut. Dengan mempergunakan gps maka direncanakan pengguna tidak dipersulit untuk proses pemasukan data, karena sebagian besar data-data yang menjadi masukan diambil dan diperhitungkan dari data koordinat gps. Sistem dirancang sedemikian rupa

sehingga sistem tidak terpaku oleh penggunaan suatu gps tertentu, pengguna diperbolehkan untuk mempergunakan gps jenis apapun guna mendukung tingkat akurasi sistem asalkan gps tersebut memiliki media koneksi berupa bluetooth. Sistem juga dirancang agar pengguna dapat memasukkan data koordinat dan ketinggian secara manual yang dirasa pengguna lebih presisi tanpa memerlukan data dari gps.

Jika dibandingkan pengukuran secara konvensional, penggunaan gps dalam sistem dapat mempercepat waktu penentuan data-data koordinat dari suatu titik. Pada pengukuran konvensional, untuk menentukan sudut, ketinggian dan jarak suatu titik ke titik lain diperlukan data-data dari alat theodolit, kemudian dari data-data theodolit tersebut dikonversi dan diperhitungkan untuk mendapatkan data-data koordinatnya, hal ini memerlukan waktu untuk perhitungannya. Sedangkan dengan mempergunakan gps maka pengguna cukup berada pada posisi koordinat yang diinginkan, kemudian gps akan memperhitungkan posisinya secara otomatis, nantinya akan didapatkan data koordinat, jarak dan ketinggian dari posisi tersebut.

Dengan mempergunakan gps, maka pengguna dapat menentukan kecepatan geraknya dari satu station ke station lainnya. Dalam sistem, hal ini diperlukan guna menentukan kecepatan rencana dari kendaraan yang akan melewati jalan tersebut.

Pada penelitian ini, peneliti menggunakan GPS yang disediakan oleh Holux M241, dimana spesifikasi yang dimiliki oleh GPS adalah sebagai berikut :

a. Menggunakan dual interface (bluetooth dan usb cable)
b. Caompatible dengan bluetooth serial port profile
c. Menggunakan 32 paralel satelit
d. Berfrekuensi 1.023 MHz

e. Sensitivitas penermiaan sinyal satelit sebesar -159dBm

f. Akurasi posisi sebesar 3 M CEP

g. Velositas sebesar 0.1 M/detik

h. Deviasi posisi untuk horisontal dan vertikal sebesar 5 meter.

BAB III

METODELOGI PENELITIAN

A. Jenis Penelitian

Untuk menemukan model *alternative* system dalam perhitungan yang dipergunakan dalam proses pengadaan barang dan jasa, dengan unsur-unsur pokok yang harus dipecahkan sesuai dengan butir-butir rumusan masalah, tujuan dan manfaat penelitian, maka digunakan metode kuantitatif.

Sugiyono (2009:13) mengemukakan bahwa "penelitian pada prinsipnya digunakan untuk menjawab permasalahan, masalah merupakan penyimpangan dari apa yang seharusnya dengan apa yang terjadi sesungguhnya". Sugiyono juga menjelaskan bahwa "penelitian kuantitatif bertolak dari studi pendahuluan dari obyek-obyek yang diteliti untuk mendapatkan apa yang betul-betul menjadi permasalahan. Agar peneliti dapat menggali masalah dengan baik maka peneliti harus menguasai teori melalui membaca referensi, selanjutnya berdasarkan referensi tersebut masalah dapat dijawab dengan baik".

B. Variabel Penelitian

Arikunto (2002:126) menyebutkan bahwa "Variabel adalah gejala yang bervariasi yang menjadi objek penelitian". Dalam penelitian ini terdapat 3 variabel yaitu :

1. Variabel bebas

   Variable bebas pada penilitian ini adalah variable-variabel yang menjadi data inputan yang nantinya diproses dalam program aplikasi untuk menghasilkan outputan. Variable tersebut antara lain :

   a. Kecepatan Rencana kendaraan
   b. Superelevasi (e) maksimum
   c. sudut perpotongan antara dua bagian lurus jalan yang membentuk lengkungan (β)
   d. lebar jalan
   e. kemiringan melintang normal jalan
   f. Jari-jari lingkaran pada lengkungan
   g. Kelandaian pada sisi kiri lengkungan vertikal (g_1)
   h. Kelandaian pada sisi kanan lengkungan vertical (g_2)
   i. Panjang lengkung vertical yang diproyeksikan pada bidang horizontal (L)

2. Variabel terikat

   Variable terikat pada penelitian ini adalah variable-variabel konstan yang digunakan dalam perhitungan program aplikasi, yaiut antara lain :

   a. Variabel konstanta-kontanta seperti π.
   b. Variable yang sudah ditentukan nilainya dan menjadi ketetapan, seperti kelandaian relative ($1/m$) dan kemiringan melintang jalan (e_n) yang bernilai 2%.

3. Variable control

Variable control pada penelitian ini adalah variable yang menjadi outputan atau hasil akhir pada program aplikasi, antara lain :

a. Volume pekerjaan galian, timbunan dan perkerasan.
b. Panjang keseluruhan jalan.
c. Perbandingan hasil akhir antara output aplikasi dengan output perhitungan.

C. Teknik Pengumpulan Data

Dalam penelitian kuantitatif yang menjadi data penelitian adalah data-data kuantitatif. Sugiyono (2007:14) mengemukakan "Data kuantitatif adalah data yang berbentuk bilangan, angka-angka, variable atau data kualitatif yang diangkakan".

Adapun jenis data yang dipergunakan antara lain :

a. Data Primer.

   Data yang dikumpulkan langsung dari lokasi penelitian, yaitu dengan melakukan pengamatan di lapangan mengenai data-data yang berhubungan dengan ukuran dan kelengkapan jalan.

b. Data Sekunder

   Data yang dikumplkan dari dokumen dan arsip dari dokumen penawaran yang telah dilakukan CV. Jaya Utama kepada Bina Marga atas pelelangan pengadaan jalan.

Sumber data yang dipergunakan dalam penelitian ini berasal dari :

a. Pengamatan langsung pada lokasi pekerjaan.
b. Dokumen penawaran CV. Jaya Utama untuk pelelangan pengadaan jalan.

c. Referensi-referensi dari Bina Marga Kabupaten Pasuruan

Data-data yang dipergunakan tersebut dikumpulkan melalui beberapa teknik pengumpulan data, antara lain :

a. Teknik wawancara

Yaitu proses memperoleh keterangan atau data untuk tujuan penelitian dengan Tanya jawab secara langsung dengan pihak yang terkait, yaitu staf administrasi CV.Jaya Utama.

b. Teknik dokumentasi

Yaiut pengumpulan data dengan menggali arsip dan dokumentasi dari pihak manajemen CV. Jaya Utama, misalnya dokumen penawaran harga.

D. Analisis Data

Yang dimaksud dengan analisis data adalah proses mencari dan menghitung serta mengolah seluruh data yang telah diperoleh dari hasil penelitian dan dilakukan secara sistematis melalui prosedur dan sesuai rencana yang telah ditetapkan.

Pertama, data yang telah terkumpul dianalisis untuk mengetahui kelayakan data inputan berdasarkan atas persyaratan-persyaratan yang telah ada dan dijabarkan pada bab 2.

Kedua, teknik analisi data yang digunakan untuk melihat pengaruh dari bermacam-macam variable bebas terhadap variable terikat baik secara terpisah maupun gabungan, disebut juga sebagai teknik analisis komparatif. Penggunaannya yaitu dengan menguji tingkat kebenaran dari perhitungan program aplikasi yang disusun dengan cara membandingkan output aplikasi dengan data hasil perhitungan pada dokumen penawaran.

BAB IV

PERANCANGAN

4.1. Analisa Kebutuhan Sistem

Analisa sistem dibutuhkan untuk menjabarkan kebutuhan-kebutuhan pengguna dalam pemakaian perangkat lunak tersebut. Disamping itu dengan melakukan analisa terhadap perangkat lunak yang akan disusun maka memberikan kemudahan dalam perencanaan dan perancangan perangkat lunak. Sehingga pengimplementasian perangkat lunak nantinya sesuai dengan yang dibutuhkan oleh pengguna akhir.

Proses analisa dalam pembuatan sistem ini meliputi analisa kebutuhan perangkat lunak dan implementasinya.

4.1.1. Analisa Kebutuhan Perangkat Lunak

Dari hasil pengamatan di lapangan, pada tahap perencanaan pembangunan jalan baru diperlukan suatu penentuan letak jalan (*Stasiuning*) yang benar-benar strategis serta perhitungan atas perkiraan bentuk jalan yang telah jadi nantinya secara matang baik itu berbentuk bidang lurus, lengkung vertikal ataupun horisontal. Untuk itu diperlukan suatu sistem yang dapat memberikan hasil perhitungan yang akurat dan dalam tempo yang singkat, sehingga mempermudah dalam proses perencanaan dan perancangan pembangunan jalan. Dengan mengkonversikan titik-titik letak jalan (*Stasiuning*) ke dalam bentuk koordinat dan mengo*to*matiskan segala variabel-variabel yang menjadi nilai masukan maka

didapatkan suatu analisa kebutuhan perangkat lunak yang dispesifikkan sebagai berikut.

4.1.1.1. Spesifikasi Pengguna Perangkat Lunak

Perangkat lunak tidak dapat lepas dari interuksi-interuksi yang diberikan oleh pengguna. Pada sistem ini pengguna perangkat lunak dibedakan menjad dua bentuk, yakni pengguna yang berhak mengakses dan memanipulasi data serta pegguna yang tidak berhak untuk memanipulasi data tetapi berhak untuk melihat laporan perhitungan data yang telah tersimpan saja.

4.1.1.2. Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak merupakan hal-hal yang dapat dilakukan oleh perangkat lunak untuk menanggapi permintaan dari pengguna ataupun dari dalam sistem perangkat lunak itu sendiri. Adapun spesifikasi-spesifikasi perangkat lunak tersebut antara lain :

1. Perangkat lunak harus dapat memproses data *login* yang telah dilakukan oleh pengguna, dan menentukan apakah pengguna tersebut berhak mengakses sistem atau tidak, jika tidak maka pengguna hanya dapat mengakses sebagian menu yang disediakan untuk guest. Serta memberikan pesan kesalahan jika data masukan pengguna tidak sesuai.
2. Perangkat lunak harus dapat melakukan pencarian terhadap *device-device* yang terintegrasi dengan blue*to*oh, dan menampilkannya kepada pengguna. Kemudian melakukan proses koneksi jika pengguna menekan *t*ombol konek.

3. Perangkat lunak harus dapat membaca data masukan yang dari *device* gps *bluetooth* jika sudah terkoneksi, kemudian memparsing dan menampilkannya kepada pengguna kedalam bentuk *latitude*, *longitude* dan *altitude.* Serta menyimpan data tersebut ke dalam *storage*.
4. Perangkat lunak harus dapat membaca data koordinat dari *storage* yang bertipe *degree minute second* dan mengkonversinya menjadi *form*at *decimal degree* serta menyimpannya kedalam *storage* untuk dipergunakan dalam perhitungan berikutnya.
5. Perangkat lunak harus dapat melakukan penghapusan data koordinat yang telah tersimpan pada *storage*.
6. Perangkat lunak harus dapat membaca data konversi koordinat dan mempergunakannya dalam perhitungan *stasiuning* untuk menentukan jarak interval dua titik dan mengeset tipe jalan dari data tersebut, kemudian menyimpan data *stasiuning* tersebut ke dalam *storage*.
7. Perangkat lunak harus dapat melakukan penghapusan data *stasiuning* yang telah tersimpan pada *storage*.
8. Perangkat lunak harus dapat membaca data masukan mengenai properti jalan yang di*input*kan oleh pengguna dan menyimpan data properti ke dalam *storage*.
9. Perangkat lunak harus dapat melakukan penghapusan data properti jalan yang telah tersimpan pada *storage*.
10. Perangkat lunak harus dapat membaca data *stasiuning* dari *storage* untuk yang bertipe jalan *alignment* horisontal saja dan membaca data *properti jalan* untuk nama jalan yang ditentukan pengguna, kemudian membaca masukan dari pengguna mengenai arah tikungan dan tipe tikungan untuk diperhitungkan

sudut perpo*to*ngan antara bidang lurus (tangen) sebelah kiri koordinat dengan sebelah kanan koordinat, elevasi normal jalan dan elevasi maksimum bidang lengkungnya, setelah itu perangkat lunak meyimpan data horisontal tersebut ke dalam *storage*.

11. Perangkat lunak harus dapat melakukan penghapusan data horisontal yang telah tersimpan pada *storage*.
12. Perangkat lunak harus dapat membaca data horisontal yang memiliki tipe tikungan *circle* dan data properti jalan untuk nama jalan yang telah ditentukan pengguna. Kemudian data tersebut dipergunakan untuk memperhitungkan elevasi jalan, landai *relative*, jarak awal dan akhir lengkung horisontal dari *stasiuning* sebelumnya, serta panjang lengkung horisontalnya. Setelah itu data hasil perhitungan horisontal tipe *circle* disimpan pada *storage.*
13. Perangkat lunak harus dapat melakukan penghapusan data horisontal tipe *circle* yang telah tersimpan pada *storage.*
14. Perangkat lunak harus dapat membaca data horisontal yang memiliki tipe tikungan *spiral-circle-spiral* dan data properti jalan untuk nama jalan yang telah ditentukan pengguna. Kemudian data tersebut dipergunakan untuk memperhitungkan elevasi jalan, landai *relative*, jarak awal dan akhir lengkung horisontal dari *stasiuning* sebelumnya, serta panjang lengkung horisontalnya. Setelah itu data hasil perhitungan horisontal tipe *spiral-circle-spiral* disimpan pada *storage.*
15. Perangkat lunak harus dapat melakukan penghapusan data horisontal tipe *spiral-circle-spiral* yang telah tersimpan pada *storage.*

16. Perangkat lunak harus dapat membaca data horisontal yang memiliki tipe tikungan *spiral-spiral* dan data properti jalan untuk nama jalan yang telah ditentukan pengguna. Kemudian data tersebut dipergunakan untuk memperhitungkan elevasi jalan, landai *relative*, jarak awal dan akhir lengkung horisontal dari *stasiuning* sebelumnya, serta panjang lengkung horisontalnya. Setelah itu data hasil perhitungan horisontal tipe *spiral-spiral* disimpan pada *storage.*
17. Perangkat lunak harus dapat melakukan penghapusan data horisontal tipe *spiral-spiral* yang telah tersimpan pada *storage*.
18. Perangkat lunak harus dapat membaca data *stasiuning* yang bertipe jalan *alignment* vertikal dan data properti jalan untuk nama jalan yang telah ditentukan pengguna. Kemudian data tersebut dipergunakan untuk memperhitungkan gradien bagian lurus jalan, panjang lengkung vertikal, besarnya perpindahan tinggi dari bagian lurus jalan menjadi bagian lengkung vertikal pada titik perpo*t*ongannya, jarak pandang kendaraan, serta jarak awal dan akhir lengkung dari *statioing* sebelumnya. Kemudian menyimpan data-data vertikal tersebut kedalam *storage*.
19. Perangkat lunak harus dapat melakukan penghapusan data vertikal yang telah tersimpan pada *storage*.
20. Perangkat lunak harus dapat membaca data vertikal, data horisontal baik yang bertipe *circle*, *spiral-circle-spiral* dan *spiral-spiral*, serta data properti jalan untuk nama jalan yang telah ditentukan pengguna. Kemudian mempergunakan data-data tersebut untuk memperhitungkan jarak antar *stasiuning* setelah adanya *alignment* horisontal dan vertikal, volume untuk pekerjaan lapis

penetrasi jalan dan volume pekerjaan makadam. Kemudian menyimpan data-data tersebut kedalam *storage*.

21. Perangkat lunak harus dapat melakukan penghapusan data volume kerja yang telah tersimpan pada *storage*.
22. Perangkat lunak harus dapat membaca masukan dari pengguna untuk informasi mengenai penggun-pengguna yang berhak mengakses sistem perangkat lunak.
23. Perangkat lunak harus dapat melakukan penghapusan data pengguna otentik yang telah tersimpan pada *storage*.
24. Perangkat lunak harus dapat membaca data pengguna otentik untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.
25. Perangkat lunak harus dapat membaca data map kerja untuk ditampilkan ke pengguna dalam bentuk laporan bersamaa dengan gambaran bentuk jalan dari penampang map yang disediakan oleh *Google*, dan mencetak laporan tersebut jika pengguna menginginkannya.
26. Perangkat lunak harus dapat membaca data koordinat untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.
27. Perangkat lunak harus dapat membaca data properti jalan untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.
28. Perangkat lunak harus dapat membaca data *alignment* horisontal untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.

29. Perangkat lunak harus dapat membaca data *alignment* horisontal tipe *circle* untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.
30. Perangkat lunak harus dapat membaca data *alignment* horisontal tipe *spiral-circle-spiral* untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.
31. Perangkat lunak harus dapat membaca data *alignment* horisontal tipe *spiral-spiral* untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.
32. Perangkat lunak harus dapat membaca data *alignment* vertikal untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.
33. Perangkat lunak harus dapat membaca data volume kerja untuk ditampilkan ke pengguna dalam bentuk laporan, dan mencetak laporan tersebut jika pengguna menginginkannya.

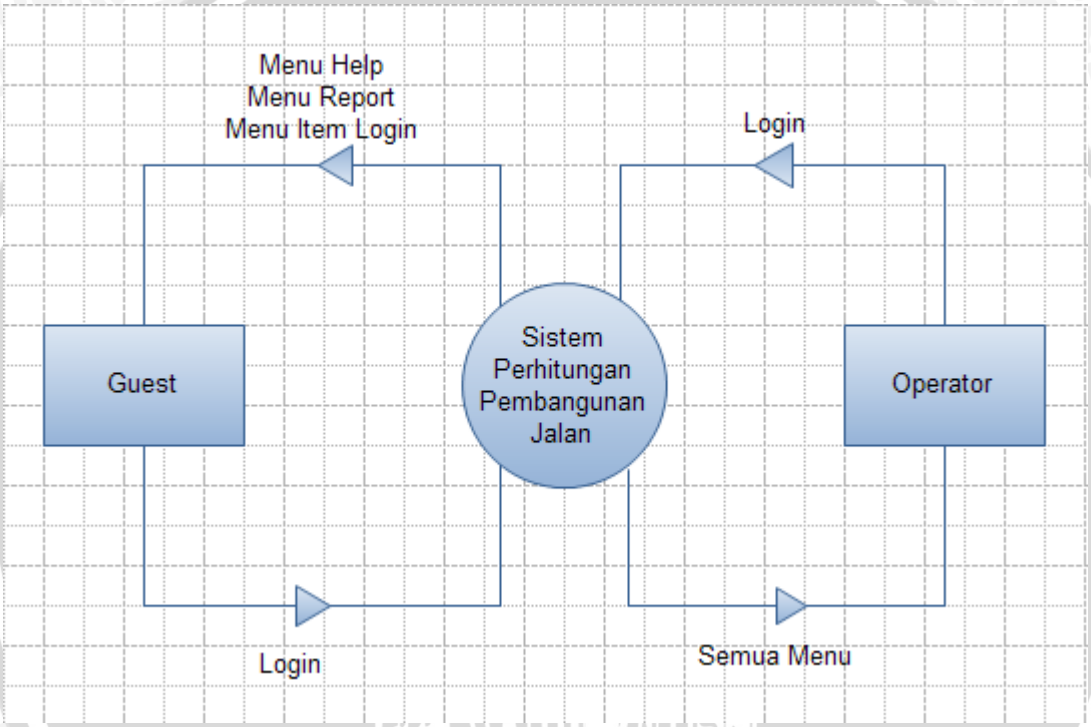
4.1.2. Data Flow Diagram

Data flow diagram adalah suatu diagram yang menggambarkan model komponen sistem yang akan dibuat dalam hal ini aplikasi perhitungan pembangunan jalan berbasis java.

Data flow diagram yang direncanakan pada aplikasi ini terbagi menjadi tiga bagian, yang antara lain :

1. Context Diagram

Context diagram menjelaskan hubungan sistem dengan lingkungan. Pada sistem ini, diagram context yang dibuat melibatkan dua fak*t*or luar, yakni guest (*user* yang tidak memiliki hak akses) dengan *operator* (*user* yang memiliki hak akses penuh). Context diagram sistem ini ditunjukkan pada gambar berikut.



Gambar 4.1 Context Diagram

Berdasarkan context diagram diatas, sistem yang dihasilkan akan mempunyai masukan sebagai berikut :

a. *Login* guest

Login guest dilakukan oleh *user* yang tidak memiliki akses, sehingga harus melakukan cancel pada *form login*. Parameter yang digunakan dalam proses *login* adalah :

i. *User*name

ii. *Password*

b. *Login operator*

Login operator dilakukan oleh *user* yang memiliki hak akses, sehingga *user* harus meng*input*kan *user* name dan *password* yang valid pada *form login.* Parameter yang dipergunakan adalah :

i. *User*name

ii. *Password*

Dan berdasarkan pada context diagram diatas, sistem yang dihasilkan tersebut mempunyai keluaran berupa :

a. Data menu *report*

Data menu *report* adalah data-data laporan dari hasil perhitungan yang telah dilakukan sebelumnnya. Data *report* terbagi atas *report* map kerja, *report* koordinat, *report* properti jalan, *report alignment* horisontal, *report alignment* vertikal, *report* volume kerja.

b. Data menu help

Data menu help adalah data-data mengenai menu bantuan dan data-data pembuat program.

c. Data menu perhitungan

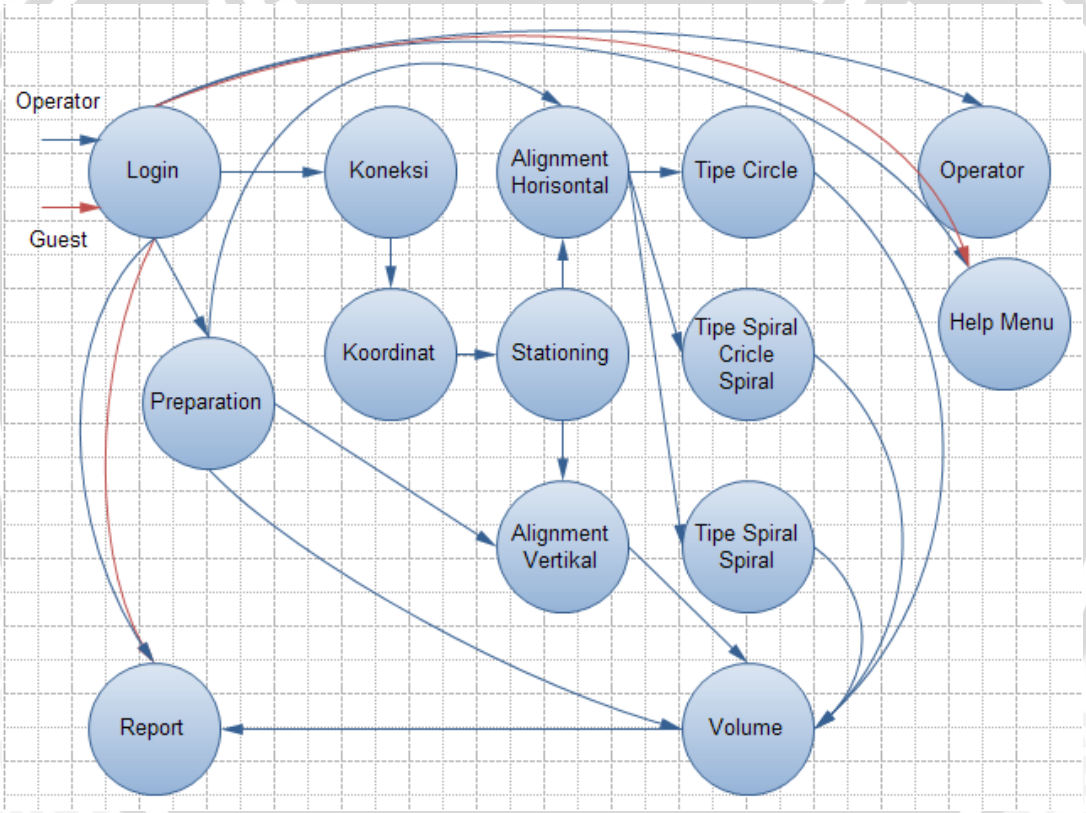
Data menu perhitungan adalah data-data perhitungan dalam perencanaan pembangunan jalan. Data perhitungan tersebut antara lain adalah perhitungan koordinat, perhitungan *stasiuning*, perhitungan properti jalan, perhitungan *alignment* horisontal, perhitungan *alignment* vertikal, perhitungan volume kerja.

d. Data menu admin

Data menu admin adalah data-data mengenai pengguna-pengguna yang berhak mengakses program.

2. DFD Level 1

Tahapan selanjutnya yang merupakan penjabaran dari context diagram adalah pembuatan DFD level 1. DFD level 1 tersebut dapat digambarkan sebagai berikut.



Gambar 4.2 DFD Level 1

Dari penggambaran DFD level 1 diatas dapat diketahui bahwa dalam sistem perhitungan pembangunan jalan terdapat 14 proses untuk *user operator* yang

berhak mengakses dan 2 proses untuk *user* guest yang tidak memiliki hak akses sistem. Penjabaran dari proses-proses tersebut adalah sebagai berikut.

a. Proses *login*

   Proses *login* terdiri dari dua jenis yakni proses *login* yang dilakukan oleh *operator* dan proses *login* yang dilakukan oleh guest. Atribut yang dipergunakan dalam proses ini adalah *user*name dan *password*. Proses *login* tervalidasi jika terdapat kecocokan antara *user*name dan *password* yang di*input*kan *user* dengan *user*name dan *password* yang talah tersimpan pada *storage*.

b. Proses olah data prerparation

   Proses olah data *preparation* dipergunakan untuk mengolah data-data masukan dari *user* mengenai properi jalan yang direncanakan dan kendaraan rencana yang akan melewatinya.

c. Proses koneksi

   Proses koneksi dipergunakan untuk mengambil data streaming dari gps *device* dengan menggunakan koneksi *bluetooth*. Data streaming akan diolah lagi menjedi bentuk koordinat *longitude, latitude* dan *altitude*.

d. Proses olah data koordinat

   Proses ini merupakan kelanjutan dari proses koneksi, dimana data koordinat dari proses koneksi dikonversi yang semula berbentu *degree minute second* menjadi betuk *decimal degree*.

e. Proses *stasiuning*

   Proses *stasiuning* adalah kelanjutan dari proses olah data koordinat, dimana data konversi koordinat diolah dan diperhitungkan sehingga ditemukan jarak

interval antar titik koordinatnya, kemudian titik-titik koordinat tersebut diset dengan nomer stasiun dan tipe jalannya.

f. Proses olah data *alignment* vertikal

Proses ini dipergunakan untuk memperhitungkan lengkung jalan yang berbentuk turunan ataupun tanjakan agar memenuhi standart keamanan dan kenyamanan bagi pengguna jalan. Data masukan dari proses ini diperoleh dari data *stasiuning* dan data properti jalan. Dari data-data tersebut nantinya akan didapatkan panjang lengkungan vertikal dan jarak pergeseran vertikal badan jalan dari keadaan awal lokasi.

g. Proses olah data *alignment* horisontal

Proses ini dipergunakan untuk memperhitungkan lengkung jalan yang berbentuk tikungan ke kanan atau ke kiri agar memenuhi standar keamanan dan kenyamanan bagi pengguna jalan. Data masukan dari proses ini diperoleh dari data *stasiuning* dan data *preparasi*, data keluaran dari proses ini adalah panjang jari-jari lengkung, besar sudut beta, elevasi normal dan elevasi maksimum. Data output tersebut nantinya akan diolah kembali untuk menjadi masukan pada perhitungan selanjutnya sesuai dengan jenis *alignment* horisontal yang telah diset.

h. Proses olah data *alignment* horisontal tipe *circle*

Proses ini adalah kalenjutan dari proses olah data horisontal, dimana proses ini memperhitungkan *alignment* horisontal yang bertipe *circle*. Masukan dari proses ini adalah data koordinat dan data *preparasi*, sedangkan keluaran dari proses ini adalah panjang tikungan, tinggi elevasi badan jalan, kelandaian relatif jalan serta jarak awal tikungan dengan jarak perpo*to*ngan jalan.

i. Proses olah data *alignment* horisontal tipe spiral-*circle*-spiral

Proses ini adalah kalenjutan dari proses olah data horisontal, dimana proses ini memperhitungkan *alignment* horisontal yang bertipe spiral-*circle*-spiral. Masukan dari proses ini adalah data koordinat dan data *preparasi*, sedangkan keluaran dari proses ini adalah panjang tikungan, tinggi elevasi badan jalan, kelandaian relatif jalan serta jarak awal tikungan dengan jarak perpo*to*ngan jalan.

j. Proses olah data *alignment* horisontal tipe spiral-spiral

Proses ini adalah kalenjutan dari proses olah data horisontal, dimana proses ini memperhitungkan *alignment* horisontal yang bertipe spiral-spiral. Masukan dari proses ini adalah data koordinat dan data *preparasi*, sedangkan keluaran dari proses ini adalah panjang tikungan, tinggi elevasi badan jalan, kelandaian relatif jalan serta jarak awal tikungan dengan jarak perpo*to*ngan jalan.

k. Proses olah data volume

Proses ini adalah proses perhitungan yang terakhir di dalam perencanaan pembangunan jalan, masukan dari proses ini adalah data-data dari *alignment* vertikal maupun horisontal, data *stasiuning* serta data *preparasi* jalan. Keluaran dari proses ini adalah panjang jarak jalan setelah dilakukan perhitungan *alignment* vertikal dan horisontal serta besarnya volume pekerjaan lapen dan makadam yang dipergunakan untuk membangun jalan.

l. Proses olah data *report*

Proses ini adalah proses yang dipergunakan untuk memberikan tampilan berupa laporan kepada *user*. Dalam proses ini juga diperkenankan untuk melakukan penyimpanan file laporan dalam bentuk html dan pencetakan laporan.

m. Proses olah data *operator*

Proses ini adalah proses pengolahan data-data yang berhubungan dengan data *user* yang berhak mengakses program.

n. Proses menu bantuan

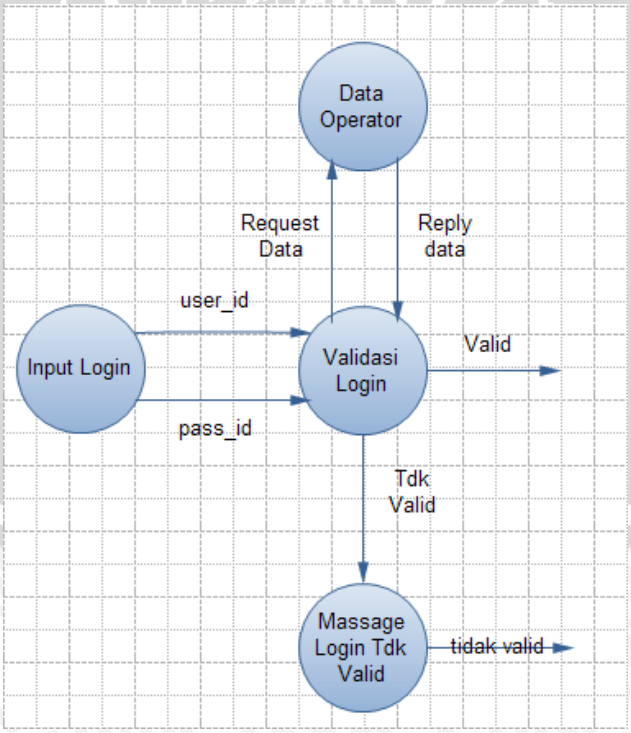
Proses ini dipergunakan untuk menampilkan menu bantuan dan menu tentang pembuat program.

3. DFD Level 2

DFD level 2 merupakan diagram yang menguraikan proses-proses pada DFD level 1 dan merinci tiap prosesnya. DFD level 2 dari sistem ini dapat dijelaskan sebagi berikut.

a. Proses *login*

Level 2 pada proses *login* digambarkan sesuai dengan gambar berikut.



Gambar 4.3 DFD Level 2 *Login* Proses

Gambar diatas menunjukkan perincian proses *login* menjadi 4 proses yang saling beruntunan.

i. Proses *input login*

Merupakan proses untuk memasukkan data *login* dari *user* yang berupa *user*nam dan *password*.

ii. Proses validasi *login*

Merupakan proses untuk mengecek validasi dari data *login* yang di-*input*-kan *user* dengan data *operator* pada tabel *operator*.

iii. Proses data *operator*

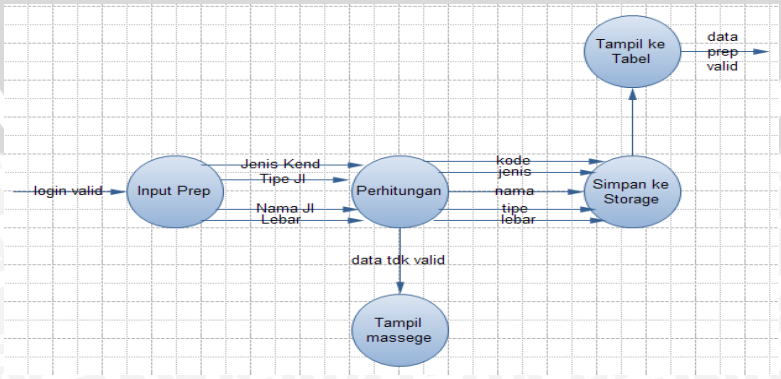
Merupakan proses pembacaan data *user*name dan *password* pada tabel *operator*, untuk proses validasi data *login user*.

iv. Proses *message* tidak valid

Merupakan proses menampilkan pesan kepada *user* jika data yang telah di-*input*-kan tidak valid.

b. Proses olah data *preparation*

Level 2 proses olah data *preparation* digambarkan sebagai berikut.



Gambar 4.4 DFD Level 2 *Preparation* Proses

Rincian dari DFD level 2 dari *preparation* proses digolongkan menjadi 5 proses. Yang diantaranya adalah sebagai berikut.

i. Proses *input preparasi*

Proses ini digunakan untuk membaca masukan yang diberikan oleh *user*. Data masukan tersebut berupa jenis kendaraan, tipe jalan, nama jalan dan lebar jalan.

ii. Proses perhitungan

Proses ini digunakan untuk mengola data masukan menjadi data yang lebih diperlukan dalam proses perhitungan selanjutnya. Dalam hal ini data jenis kendaraan akan diolah menjadi kecepatan rencana sesuai dengan jenis kendaraan yang dipilih.

iii. Proses tampil *message*

Proses ini digunakan untuk menampilkan *message* jika data yang di-*input*-kan tidak valid.

iv. Proses simpan *storage*

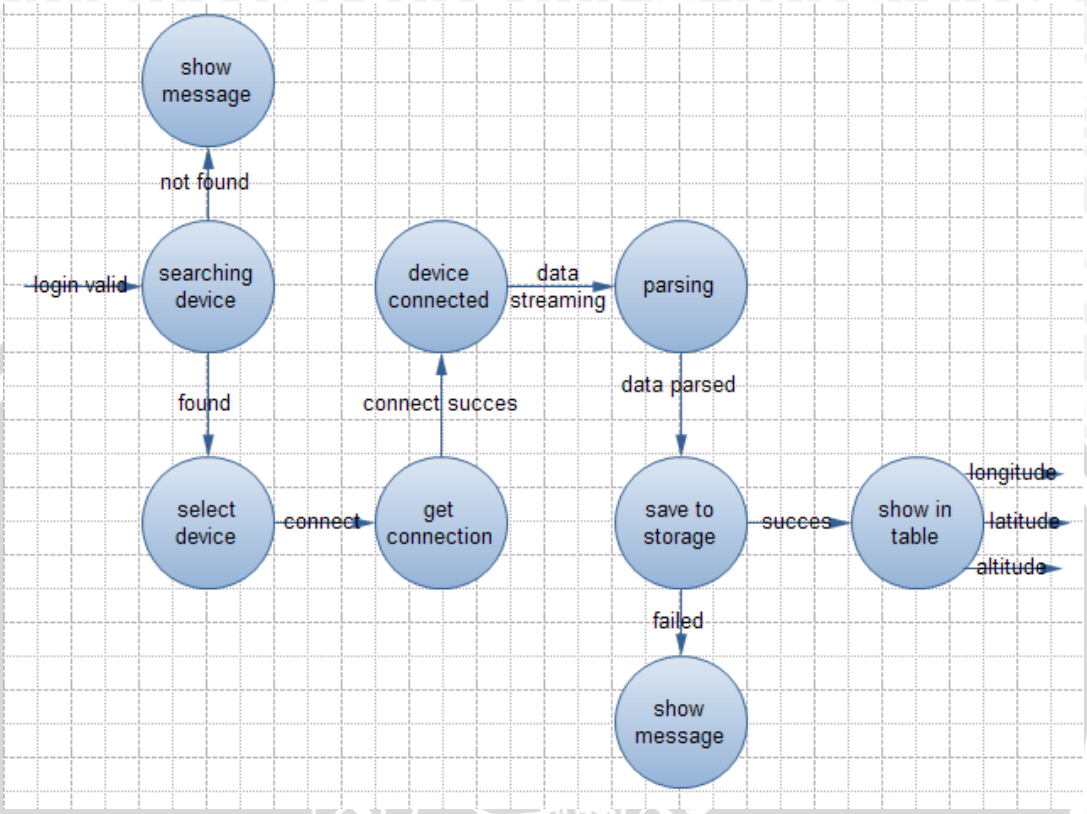
Proses ini digunakan untuk menyimpan data-data hasil pengolahan ke dalam database pada tabel *preparation*.

v. Proses tampil ke tabel

Proses ini digunakan untuk memberikan tampilan kepada *user* mengenai data-data yang telah tersimpan dalam tabel *operator*.

c. Proses koneksi

Level 2 dari proses koneksi dapat digambarkan sebagai berikut.



Gambar 4.5 DFD Level 2 Konek Proses

Rincian dari proses koneksi sesuai dengan DFD level 2 digolongkan menjadi 9 proses. Yang antara lain :

i. *Searching device*

Merupakan proses pencarian *device-device* yang mengkatifkan *bluetooth*-nya, kemudian menampilkannya dalam bentuk *list* kepada *user*.

ii. *Show message*

Proses pemberitahuan sistem kepada *user* bahwa proses pencarian telah selesai.

iii. *Select device*

Proses penampilan nama-nama *bluetooth device* yang terdeteksi oleh sistem kepada *user*, agar *user* dapat menyeleksi *device* mana yang akan dikoneksi.

iv. *Get connection*

Proses pengkonekan sistem dengan *device* melalui *bluetooth* menggunakan nama *bluetooth device* yang diseleksi oleh *user*.

v. *Device connected*

Proses penampilan data-data streaming dari *device* yang terkoneksi kepada *user*.

vi. *Parsing*

Proses pemarsingan data streaming *device* menjadi bentuk text-text yang lebih berguna dalam proses perhitungan.

vii. *Save to storage*

Proses penyimpanan data-data hasil parsingan ke dalam *storage* pada tabel gps.

viii. *Show message*

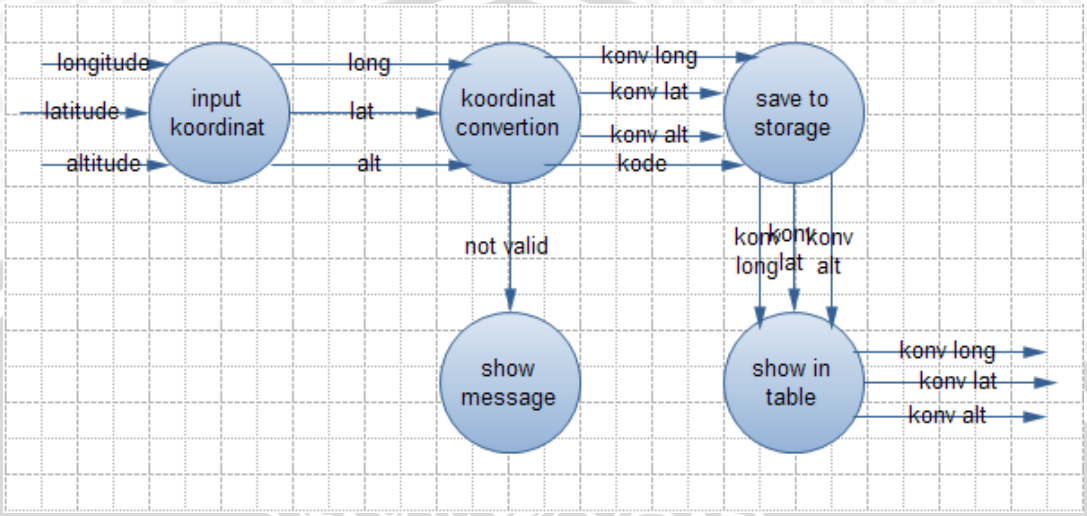
Proses penampilan pemberitahuan kepada *user* jika data yang akan disimpan tidak valid.

ix. *Show in* tabel

Proses penampilan data-data yang telah tersimpan pada tabel gps kepada *user*.

d. Proses olah data koordinat

Level 2 dari proses olah data koordinat dapat digambarkan sebagai berikut.



Gambar 4.6 DFD Level 2 Proses Olah Data Koordinat

Dari gambar diatas dapat dirinci proses olah data koordinat menjadi 5 proses, yang antara lain adalah :

i. *Input* koordinat

Merupakan proses pembacaan data yang telah di*input*kan oleh *user*, data yang menjadi masukan dari proses ini adalah *longitude*, *latitude* dan *altitude* dari proses koneksi.

ii. Koordinat *convertion*

Merupakan proses pengkonversian *form*at data koordinat dari bentuk *degree minute second* menjadi bentu *decimal degree*.

iii. *Show message*

Merupakan proses pemberitahuan kepada *user* apabila data masukan tidak valid.

iv. *Save to storage*

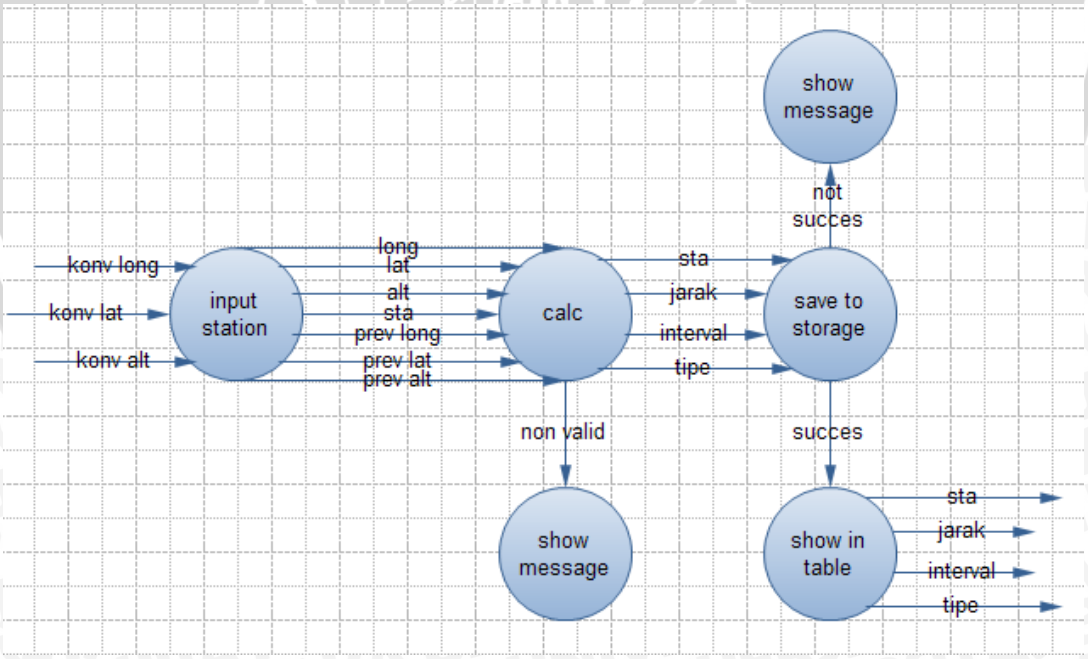
Merupakan proses penyimpanan data-data konversi koordinat ke dalam *storage* pada tabel koordinat.

v. *Show in* tabel

Merupakan proses penampilan data-data yang telah tersimpan pada tabel koordinat kepada *user*.

e. Proses *stasiuning*

Level 2 dari proses *stasiuning* dapat digambarkan sebagai berikut.



Gambar 4.7 DFD Level 2 Proses *Stasiuning*

Dari gambar diatas dapat diketahui bahwa proses *stasiuning* dirinci menjadi 6 proses, penjabaran tiap-tiap proses adalah sebagai berikut.

i. *Input* stasiun

Merupakan proses pembacaan data dari *user*, yang berupa data konversi *longitude*, *latitude* dan *altitude*.

ii. *Calc*

Merupakan proses perhitungan data dari masukan untuk menentukan jarak interval antar titik koordinat.

iii. *Show message*

Merupakan proses penampilan pemberitahuan kepada *user* karena data tidak valid.

iv. *Save to storage*

Merupakan proses penyimpanan data hasil perhitungan ke dalam *storage* pada tabel stasiun.

v. *Show message*

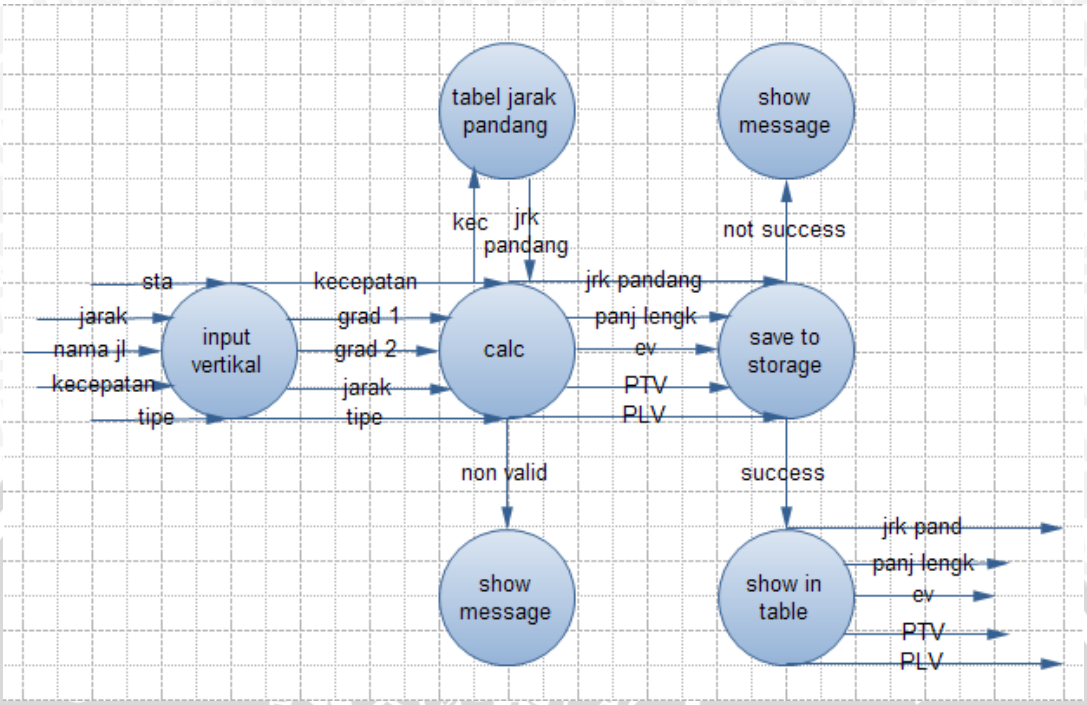
Merupakan proses pemberitahuan kepada *user* bahwa data yang akan disimpan tidak valid.

vi. *Show in* tabel

Merupakan proses penampilan data yang telah disimpan pada tabel stasiun kepada *user*.

f. Proses olah data *alignment* vertikal

Level 2 dari proses oleh data *alignment* vertikal dapat digambarkan sebagai berikut.



Gambar 4.8 DFD Level 2 Proses Olah Data *Alignment* Vertikal

Dari gambar diatas dapat diketahui bahwa proses olah data *alignment* vertikal dirinci menjadi 7 proses, yakni sebagai berikut.

i. *Input* vertikal

Merupakan proses pembacaan data masukan dari *user*, data masukan tersebut antara lain data stasiun dan data *preparation*.

ii. *Calc*

Merupakan proses perhitungan data-data masukan menjadi data panjang lengkung, jarak pandang, pergeseran ketinggian badan jalan.

iii. Tabel jarak pandang

Merupakan proses pembacaan data jarak pandang pada tabel jarak pandang berdasarkan pada kecepatan rencana.

iv. *Show message*

Proses pemberitahuan kepada *user* bahwa data yang akan diperhitungkan tidak valid.

v. *Save to storage*

Proses penyimpanan data hasil perhitungan ke dalam *storage* pada tabel vertikal.

vi. *Show message*

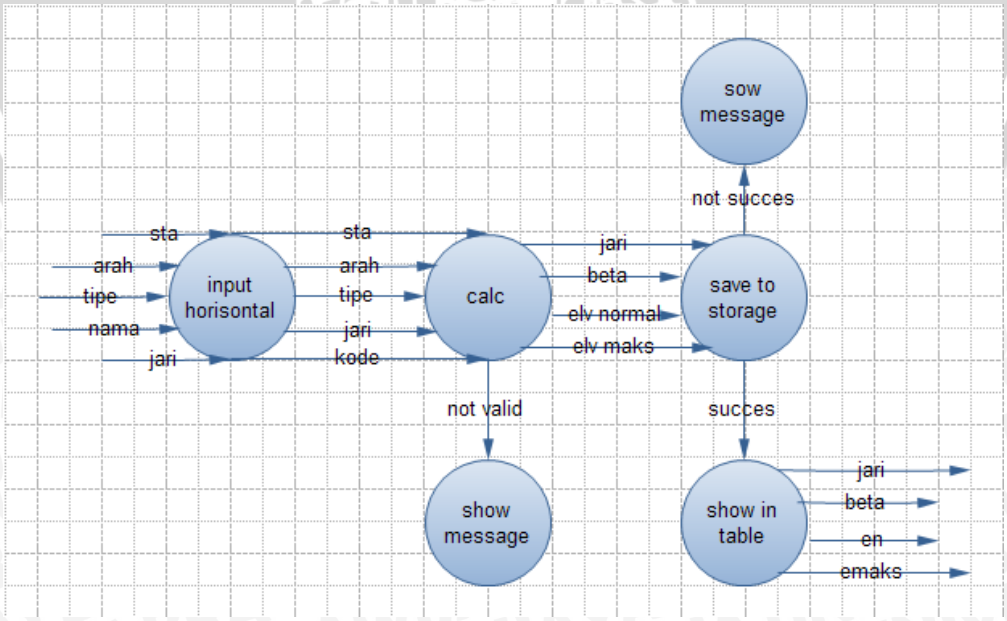
Proses pemberitahuan kepada *user* bahwa data yang akan disimpan tidak valid.

vii. *Show in* tabel

Proses penampilan data-data yang telah disimpan pada tabel vertikal kepada *user*.

g. Proses olah data *alignment* horisontal

Level 2 pada proses olah data dapat digambarkan sebagai berikut.



Gambar 4.9 DFD Level 2 Proses Olah Data *Alignment* Horisontal

Dari gambar diatas, dapat diketahui bahwa proses olah data *alignment* horisontal dirinci menjadi 6 proses, yakni sebagai berikut.

i. *Input* horisontal

Merupakan proses pembacaan data *input* dari *user*, dimana data *input* tersebut adalah data *stasiuning* dan data *preparation*.

ii. *Calc*

Merupakan proses perhitungan data masukan menjadi data jari-jari lengkung, sudut beta, elevasi normal dan elevasi maksimal. Untuk nantinya diolah kembali sesuai dengan tipe *alignment* horisontalnya.

iii. *Show message*

Proses pemberitahuan kepada *user* bahwa data yang akan diperhitungkan tidak valid.

iv. *Save to storage*

Proses penyimpanan data hasil perhitungan ke dalam *storage* pada tabel horisontal.

v. *Show message*

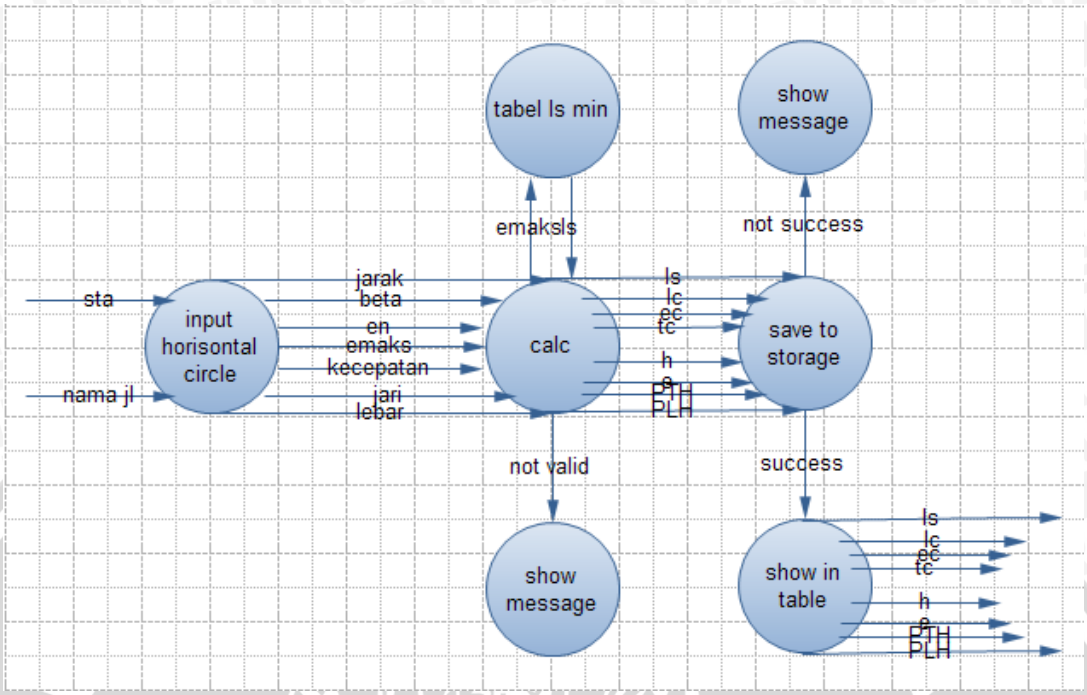
Proses pemberitahuan bahwa data yang akan disimpan tidak valid.

vi. *Show in* tabel

Proses penampilan data-data yang telah disimpan pada tabel horisontal.

h. Proses olah data *alignment* horisontal tipe *circle*

Level 2 proses olah data *alignment* horisontal tipe *circle* dapat digambarkan sebagai berikut.



Gambar 4.10 DFD Level 2 Proses Olah Data *Alignment* Horisontal tipe *Circle*

Dari gambar diatas dapat diketahui bahwa proses olah data *alignment* horisontal tipe *circle* dirinci menjadi 7 proses, yakni sebagai berikut.

i. *Input* horisontal *circle*

Merupakan proses pengambilan data masukan dari *user*, data yang menjadi masukan adalah data *stasiuning* dan data *preparation*.

ii. *Calc*

Proses perhitungan *alignment* horisontal tipe *circle* dari data masukan menjadi data panjang lengkung, kelandaian jalan, dan elevasi badan jalan.

iii. Tabel ls minimum

Proses pembacaan data panjang lengkung minimum pada tabel ls minimum berdasarkan kepada data kecepatan rencana.

iv. *Show message*

Proses pemberitahuan kepada *user* bahwa data yang diperhitungkan tidak valid.

v. *Save to storage*

Proses penyimpanan data hasil perhitungan ke dalam *storage* pada tabel horisontal *circle*.

vi. *Show message*

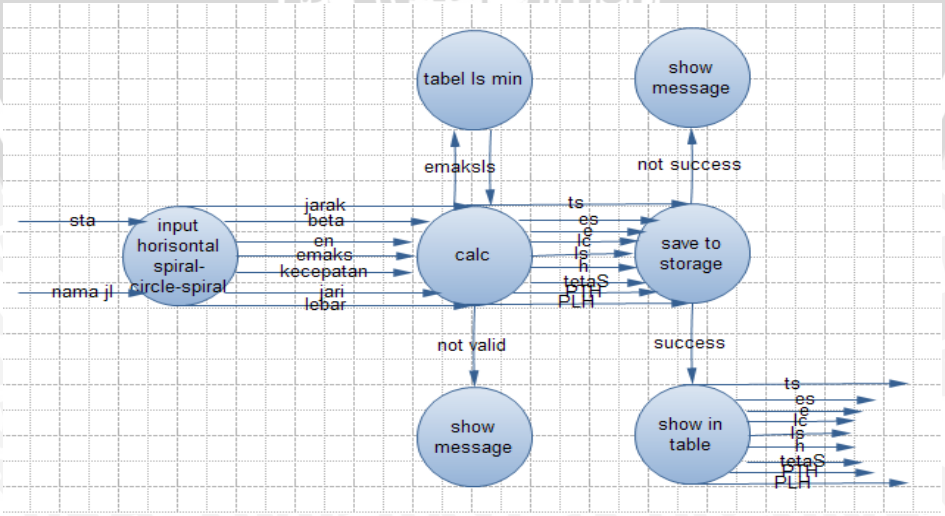
Proses pemberitahuan kepada *user* bahwa data yang akan disimpan tidak valid.

vii. *Show in* tabel

Prose penampilan data-data yang telah disimpan pada tabel horisontal *circle* kepada *user*.

i. Proses olah data *alignment* horisontal tipe spiral-*circle*-spiral

Level 2 proses olah data alignmetn horisontal tipe spiral-*circle*-spiral dapat digambarkan sebagai berikut.



Gambar 4.11 DFD Level 2 Proses Olah Data *Alignment* Horisontal tipe Spiral-*Circle*-Spiral

Dari gambar diatas dapat diketahui bahwa proses olah data spiral-cirlce-spiral dirinci menjadi 7 proses, yakni sebagai berikut.

i. *Input* horisontal spiral-*circle*-spiral

   Merupakan proses pembacaan data masukan dari *user*, data masukan tersebut antara lain data-data *stasiuning* dan data-data *preparation*.

ii. *Calc*

   Proses perhitungan data-data masukan menjadi data panjang lengkung, data pergeseran tikungan jalan, data kelandaian relatif dan data panjang titik awal lengkung hingga titik perpo*to*ngannya.

iii. Tabel ls minimum

   Proses pembacaan data lengkung minimum pada tabel ls minimum berdasarkan kepada kecepatan rencana.

iv. *Show message*

   Proses pemberitahuan kepada *user* bahwa data yang akan diperhitungkan tidak valid.

v. Savae *to storage*

   Proses penyimpanan data-data hasil perhitungan ke dalam *storage* pada tabel spiral-*circle*-spiral.

vi. *Show message*

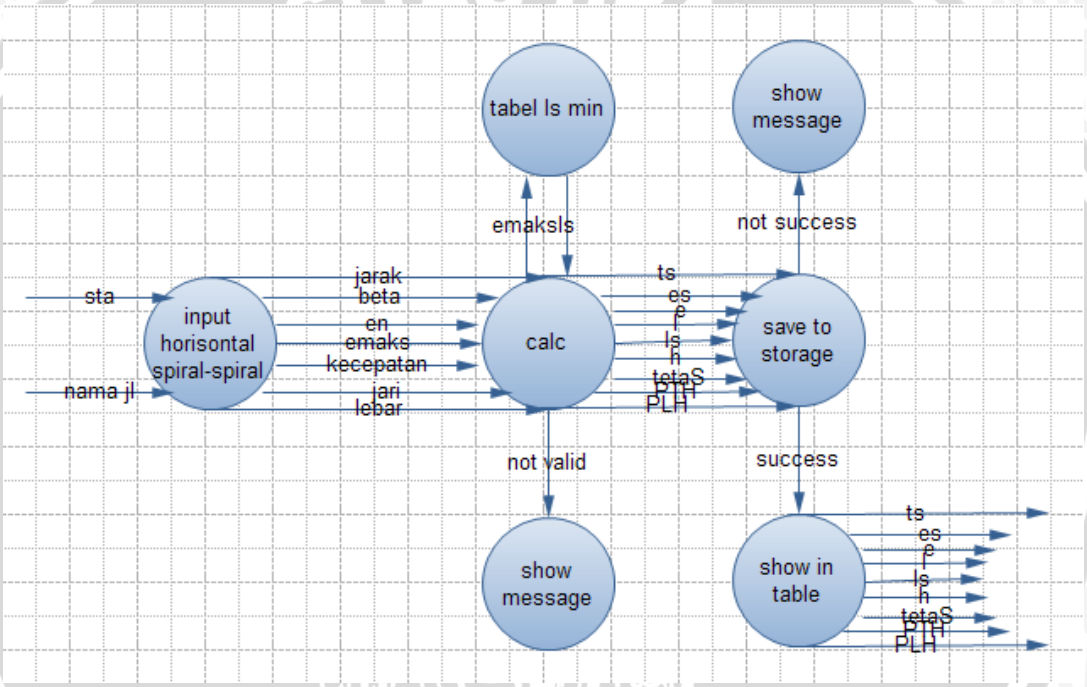
   Proses pemberitahuan kepada *user* bahwa data yang akan disimpan tidak valid.

vii. *Show in* tabel

Proses penampilan data-data yang telah disimpan pada tabel spiral-*circle*-spiral kepada *user*.

j. Proses olah data *alignment* horisontal tipe spiral-spiral

Level 2 proses olah data *alignment* horisontal tipe spiral-spiral dapat digambarkan sebagai berikut.



Gambar 4.12 DFD Level 2 Proses Olah Data *Alignment* Horisontal tipe Spiral -Spiral

Dari gambar diatas dapat diketahui bahwa proses olah data spiral -spiral dirinci menjadi 7 proses, yakni sebagai berikut.

i. *Input* horisontal spiral-*circle*-spiral

Merupakan proses pembacaan data masukan dari *user*, data masukan tersebut antara lain data-data *stasiuning* dan data-data *preparation*.

ii. *Calc*

Proses perhitungan data-data masukan menjadi data panjang lengkung, data pergeseran tikungan jalan, data kelandaian relatif dan data panjang titik awal lengkung hingga titik perpo*to*ngannya.

iii. Tabel ls minimum

Proses pembacaan data lengkung minimum pada tabel ls minimum berdasarkan kepada kecepatan rencana.

iv. *Show message*

Proses pemberitahuan kepada *user* bahwa data yang akan diperhitungkan tidak valid.

v. Savae *to storage*

Proses penyimpanan data-data hasil perhitungan ke dalam *storage* pada tabel spiral-spiral.

vi. *Show message*

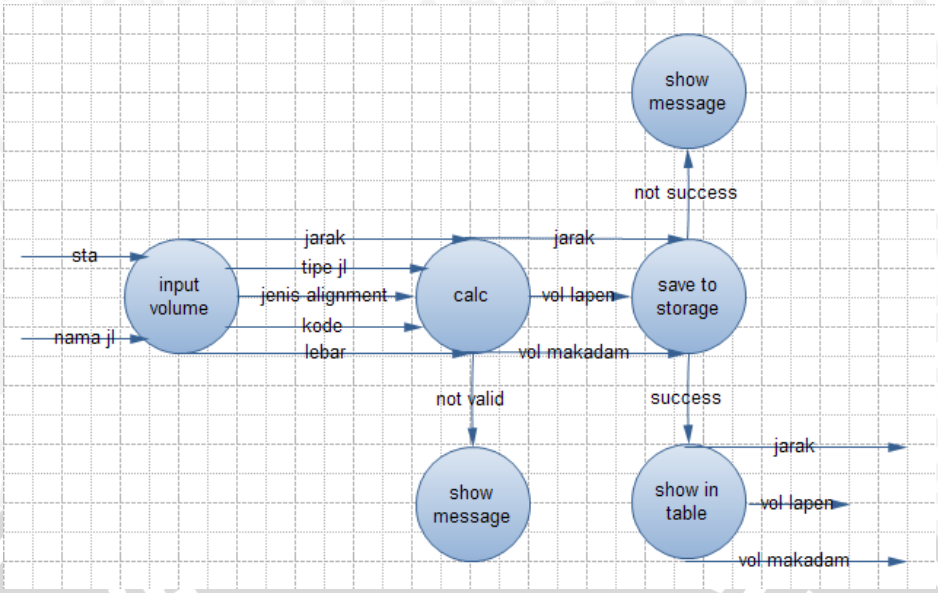
Proses pemberitahuan kepada *user* bahwa data yang akan disimpan tidak valid.

vii. *Show in* tabel

Proses penampilan data-data yang telah disimpan pada tabel spiral-spiral kepada *user*.

k. Proses olah data volume

Level 2 dari prose olah data volume dapat digambarkan sebagai berikut.



Gambar 4.13 DFD Level 2 Proses Olah Data Volume

Dari gambar diatas dapat diketahui bahwa proses olah data volume dirinci menjadi 6 proses, yakni sebagai berikut.

i. *Input* volume

Merupakan proses pembacaan data masukan dari *user*, dimana yang menjadi data masukan adalah data *stasiuning*, data *alignment* vertikal, data *alignment* horisontal berbagai tipe dan data *preparation*.

ii. *Calc*

Proses perhitungan data-data masukan menjadi hasil keluaran terakhir dari sistem ini, yakni data jarak pembangunan jalan yang direncanakan, volume pekerjaan lapis penetrasi dan volume pekerjaan makadam.

iii. *Show message*

Proses pemberitahuan kepada *user* bahwa data yang akan diperhitungkan tidak valid.

iv. *Save to storage*

Proses penyimpanan data-data hasil perhitungan ke dalam *storage* pada tabel volume.

v. *Show message*

Proses pemberitahuan kepada *user* bahwa data yang akan disimpan tidak valid.

vi. *Show in* tabel

Proses penampilan data-data yang telah disimpan pada tabel volume.

4.2. Perancangan Perangkat Lunak

Berdasarkan data dan informasi yang telah dijabarkan sebelumnya. Perancangan sistem aplikasi perhitungan pembangunan jalan berbasis java meliputi perancangan database, site map dan *user* interface.

4.2.1. Perancangan Basis Data

Perancangan basis data dilakukan agar dapat mendukung terimplementasinya sistem aplikasi perhitungan pembangunan jalan ini, yang nantinya basis data digunakan sebagai tempat penyimpanan dari variabel-variabel hasil perhitungan sehingga variabel tersebut dapat diakses dan dimanipulasi untuk perhitungan lebih lanjut. Perancangan basis data meliputi normalisasi dan pembuatan ER (*Entity Relationship*) Diagram.

1. Normalisasi

Normalisasi data merupakan proses pengelompokan data ke dalam tabel yang menyatakan entitas dan hubungan antar tabel sehingga terwujud bentuk sistem basis data yang mudah untuk dimodifikasi. Normalisasi juga digunakan untuk mengurangi pengulangan data di setiap tabel dan mempercepat proses pengaksesan data dalam database.

Pada sistem basis data yang akan dibuat, pada mulanya disusun suatu tabel utama yang berisi variabel-variabel hasil akhir dari perhitungan secara keseluruhan. Di dalam sistem ini tabel utama tersebut dikategorikan menjadi 3 bentuk tabel, yakni tabel utama volume, tabel utama vertikal dan tabel utama horisontal, adapun tabel-tabel tersebut dapat ditunjukkan sebagai berikut :

no_sta	tipe	jenis_lengk	jenis_kend	kecepatan	nama_jl	type_jl	lebar	interval	jarak	vol_lpn	vol_mkdm
Sta_1	Tangen	-	Kendaraan Angkut	40	Raya Raci	Sub Urban	4	200	220	44	61.6
Sta_2	Tangen	-	Kendaraan angkut	40	Raya Raci	Sub Urban	4	200	220	44	61.6
Sta_3	Alginement Vertikal	Cembung	Kendaraan angkut	40	Raya Raci	Sub Urban	4	230	250	50	70

Tabel 4.1 Tabel Utama Volume

PPV	tipe	PLV	PTV	g1	g2	L	S	Ev	H	jrk_kanan	jrk_kiri
Sta_5	Cembung	Sta TV1	Sta LV1	0.8	0.6	210	100	0.5	6	115	115

Tabe; 4.2 Tabel Utama Vertikal

PH	PTH	PLH	no_sta	arah	jenis	beta	emaks	Rc	en	nama_jl	tetaS	tetaC	Tc	Ts	ec	es	e	L
Sta 5	Sta TC1	Sta CT	Sta_5	Kanan	Circle	60	0.08	300	0.02	Raya Raci	0	0	126.25	0	8	0	0.0029	249.8

Tabel 4.3 Tabel Utama Horisontal

Lc	Ls	p	k	H	jrk_kanan	jrk_kiri
249.8	50	0	0	0.003675	124.9	124.9

Lanjutan Tabel Utama Horisontal

Dari ketiga bentuk tabel utama diatas dapat dilakukan proses normalisasi data sebagai berikut :

a. Bentuk Normalisasi Pertama

Pada proses normalisasi pertama, yang dilakukan adalah menghilangkan elemen data berulang. Bentuk ini terpenuhi jika sebuah tabel tidak memiliki atribut bernilai banyak atau lebih dari satu atribut dengan domain yang sama. Dari ketiga tabel diatas dapat diketahui elemen data yang berulang adalah :

- Tipe, jenis lengkung, jenis kendaraan, kecepatan, nama jalan, tipe jalan, lebar, interval, jarak, volume lapen dan volume makadam pada tabel utama volume.

Untuk jenis tabel utama yang lain dimungkinkan terjadi elemen data berulang, akan tetapi data berulang tersebut masih didasarkan pada satu atribut no_sta yang berbeda-beda, sehingga pada tabel tersebut tidak terdapat atribut yang bernilai banyak dan tabel-tabel utam tersebut memenuhi normalisasi pertama.

b. Bentuk Normalisasi Kedua

Bentuk normal kedua dilakukan dengan cara menghilangkan ketergantungan fungsional sepenuhnnya. Bentuk ini terpenuhi jika semua atribut yang tidak termasuk dalam *primary key* memiliki ketergantungan fungsional pada *primary key*. *Primary key* berbentuk unik dan dapat mewakili atribut lain yang menjadi anggota serta lebih sering dipergunakan dalam tabel/relasi.

Adapun hasil normalisasi kedua dari tabel-tabel utama di atas adalah :

Primary key	Atribut
id_vol	vol_lapen, vol_makadam
id_prep	tipe_kend, kecepatan, nama_jl, lebar
id_vert	jenis, g1, g2, l, s, ev, h, PPV, PTV, PLV, jrk_knn, jrk_kr
id_hor	jenis, beta, e_maks, arah, rc, en
id_horc	tc, ls, lc, e, ec, h, PPH, PTH, PLH, jrk_knn, jrk_kr
id_horscs	tetaS,tetaC,ls,lc,l,es,e,ts,h,p,k,PPH,PTH,PLH,jrk_knn,jrk_kr
id_horss	tetaS,es,e,l,ls,ts,p,k,h,PPH,PTH,PLH,jrk_knn,jrk_kr
id_stasioning	no_sta, jarak, interval, tipe

Dengan demikian, maka dari tabel-tabel utama tersebut dapat dikelompokkan masing-masing berdasar *primary key*-nya menjadi tabel normal kedua sebagai berikut :

kd_vol	no_sta	tipe	jenis	lebar	interval	jarak	vol_lpn	vol_mkdm
1	Sta 1	Tangen		4	0	0	0	0
2	Sta 2	Alignment Horisontal	Circle	4	0	0	0	0
3	Sta 3	Tangen		4	0	0	0	0
4	Sta 4	Alignment Vertikal	Cembung	4	0	0	0	0
5	Sta 5	Tangen		4	0	0	0	0

Tabel 4.3 Tabel Vol

kode_prep	type_kend	kecepatan	nama_jl	type_jl	lebar
3	Kend angkut barang	40	raci	Perkotaan	4

Tabel 4.4 Tabel *Preparation*

kd_sta	no_sta	jarak	jarak1	tipe
5	Sta 5	0	0	Tangen
4	Sta 4	0	0	Alignment Vertikal
3	Sta 3	0	0	Tangen
2	Sta 2	0	0	Alignment Horisontal
1	Sta 1	0	0	Tangen

Tabel 4.5 Tabel *Stasiuning*

id_ver	tipe	g1	g2	L	S	jrk_kanan	jrk_kiri	Ev	h	PPV	PTV	PLV
1	Cembung	0	0	0	99.06	0	0	0	508.7	Sta 4	Sta TC4	Sta CT4

Tabel 4.6 Tabel Vertikal

id_hor	jenis	beta	e_maks	arah	Rc	En	no_sta	nama_jl
1	Circle	0	0.08	Kanan	318	0.02	Sta 2	raci
2	Spiral-Circle-Spiral	0	0.08	Kiri	409	0.02	Sta 2	raci
3	Spiral-Spiral	0	0.08	Kiri	477	0.02	Sta 2	raci

Tabel 4.7 Tabel Horisontal

id_hor_c	TC	LS	EC	e	Lc	H	PH	PTH	PLH	jrk_kanan	jrk_kiri
1	0	45	0	0.043	0	0.252	Sta 2	Sta TC2	Sta CT2	0	0

Tabel 4.8 Tabel Horisontal *Circle*

id_hor_csc	tetaS	tetaC	Es	Ts	Lc	L	e	H	Ls	p	k
1	3.15357	-6.30713	-817.146	49.8838	-45	45	0.035	0.00488889	45	-817.146	49.8838

Tabel 4.9 Tabel Horisontal Spiral-*Circle*-Spiral

PH	PLH	PTH	jrk_kanan	jrk_kiri
Sta 2	Sta TC2	Sta CT2	49.8838	-49.8838

Tabel 4.10 Lanjutan Tabel Horisontal Spiral-*Circle*-Spiral

2. ER Diagram

Entity Relation Diagram adalah suatu model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak. ER Diagram digunakan untuk menggambarkan hubungan antara entitas satu dengan entitas lain, yang ditandai dengan adanya *key* dalam sebuah tabel yang digunakan untuk berelasi antar tabel.

Sebelum menggambarkan ER Diagram, entitas-entitas pembentuk sistem harus ditentukan terlebih dahulu beserta tribut entitas itu sendiri. Entitas yang ditentukan merupakan tabel yang akan dipakai pada database. Entitas dan atribut yang akan dibentuk berikut ini merupakan tabel secara umum yang telah dinormalisasi sebelumnya. Entitas dan atribut pembentuk sistem aplikasi perhitungan pembangunan jalan berbasis java ditunjukkan sebagai berikut :

Entitas volume, merupakan tabel volume yang berisi hasil perhitungan akhir yang merupakan data akhir yang dibutuhkan *user* dalam merencanakan pemabangunan jalan.

Field	Type	Collation	Attributes	Null	Default	Extra
kd_vol	int(2)			No		auto_increment
no_sta	varchar(10)	utf8_general_ci		No		
tipe	text	utf8_general_ci		No		
jenis	text	utf8_general_ci		No		
lebar	float			No		
interval	float			No		
jarak	float			No		
vol_lpn	float			No		
vol_mkdm	float			No		

Tabel 4.11 Entitas Volume

Entitas *stasiuning*, merupakan tabel *stasiuning* yang berisi informasi mengenai titik-titik *stasiuning* yang dipergunakan dalam perhitungan.

Field	Type	Collation	Attributes	Null	Default	Extra
kd_sta	int(1)			No		auto_increment
no_sta	varchar(5)	utf8_general_ci		No		
jarak	float			No		
jarak1	float			No		
tipe	text	utf8_general_ci		No		

Tabel 4.12 Entitas *Stationing*

Entitas *preparation*, merupakan tabel *preparasi* yang berisi informasi mengenai properti jalan dan kendaraan yang melaluinya.

Field	Type	Collation	Attributes	Null	Default	Extra
kode_prep	int(1)			No		auto_increment
type_kend	varchar(50)	utf8_general_ci		No		
kecepatan	float			No		
nama_jl	varchar(50)	utf8_general_ci		No		
type_jl	varchar(50)	utf8_general_ci		No		
lebar	float			No		

Tabel 4.13 Entitas *Preparasi*

Entitas vertikal merupakan tabel vertikal yang berisi informasi mengenai variabel-variabel yang dipergunakan dalam perhitungan lengkung vertikal beserta hasilnya.

Field	Type	Collation	Attributes	Null	Default	Extra
id_ver	int(3)			No		auto_increment
tipe	text	utf8_general_ci		No		
g1	float			No		
g2	float			No		
L	float			No		
S	float			No		
jrk_kanan	float			No		
jrk_kiri	float			No		
Ev	float			No		
h	float			No		
PPV	varchar(10)	utf8_general_ci		No		
PTV	varchar(10)	utf8_general_ci		No		
PLV	varchar(10)	utf8_general_ci		No		

Tabel 4.14 Entitas Vertikal

Entitas Horisontal merupakan tabel horisontal yang berisi informasi mengenai variabel-variabel masukan dalam perhitungan *alignment* horisontal, yang nantinya dipergunakan dalam perhitungan sesuai dengan tipe lengkungan yang direncanakan.

Field	Type	Collation	Attributes	Null	Default	Extra
id_hor	int(3)			No		auto_increment
jenis	text	utf8_general_ci		No		
beta	float			No		
e_maks	float			No		
arah	text	utf8_general_ci		No		
Rc	float			No		
En	float			No		
no_sta	varchar(10)	utf8_general_ci		No		
nama_jl	text	utf8_general_ci		No		

Tabel 4.15 Entitas Horisontal

Entitas *circle* merupakan tabel *circle* yang berisi informasi mengenai variabel-variabel perhitungan lengkung horisontal yang bertipe *circle*.

Field	Type	Collation	Attributes	Null	Default	Extra
id_hor_c	int(3)			No		auto_increment
TC	float			No		
LS	float			No		
EC	float			No		
e	float			No		
Lc	float			No		
H	float			No		
PH	varchar(10)	utf8_general_ci		No		
PTH	varchar(10)	utf8_general_ci		No		
PLH	varchar(10)	utf8_general_ci		No		
jrk_kanan	float			No		
jrk_kiri	float			No		

Tabel 4.16 Entitas Horisontal tipe *Circle*

Entitas spiral-*circle*-spiral merupakan tabel spiral-*circle*-spiral yang berisi informasi mengenai variabel-variabel perhitungan lengkung horisontal bertipe spiral-*circle*-spiral.

Field	Type	Collation	Attributes	Null	Default	Extra
id_hor_csc	int(3)			No		auto_increment
tetaS	float			No		
tetaC	float			No		
Es	float			No		
Ts	float			No		
Lc	float			No		
L	float			No		
e	float			No		
H	float			No		
Ls	float			No		
p	float			No		
k	float			No		
PH	varchar(10)	utf8_general_ci		No		
PLH	varchar(10)	utf8_general_ci		No		
PTH	varchar(10)	utf8_general_ci		No		
jrk_kanan	float			No		
jrk_kiri	float			No		

Tabel 4.17 Entitas Horisontal tipe Spiral-*Circle*-Spiral

Entitas spiral-spiral merupakan tabel spiral-spiral yang berisi informasi mengenai variabel-variabel perhitungan *alignment* horisontal tipe spiral-spiral.

Field	Type	Collation	Attributes	Null	Default	Extra
id_hor_ss	int(3)			No		auto_increment
tetaS	float			No		
Es	float			No		
Ts	float			No		
L	float			No		
e	float			No		
Ls	float			No		
p	float			No		
k	float			No		
h	float			No		
jrk_kiri	float			No		
jrk_kanan	float			No		
PH	varchar(10)	utf8_general_ci		No		
PLH	varchar(10)	utf8_general_ci		No		
PTH	varchar(10)	utf8_general_ci		No		

Tabel 4.18 Entitas Horisontal tipe Spiral-Spiral

Selain dari entitas-entitas penyimpan variabel diatas, di dalam sistem aplikasi perhitungan pembangunan jalan juga dipergunakan entitas-entitas pendukung guna mempermudah dalam proses perhiungannya. Yakni antara lain adalah :

Entitas koordinat merupakan tabel koordinat yang berguna sebagai tempat penyimpanan informasi koordinat yang telah dikonversi dari *degree minute second* menjadi *decimal degree*.

Field	Type	Collation	Attributes	Null	Default	Extra
no_sta	varchar(5)	utf8_general_ci		No		
kon_long	float			No		
kon_lat	float			No		
ko_elev	float			No		
kd_sta	int(1)			No		auto_increment

Tabel 4.18 Entitas Koordinat

Entitas gps merupakan tabel gps yang berguna sebagai tempat penyimpanan informasi data-data yang bertipe NMEA dari *device* gps *bluetooth*.

Field	Type	Collation	Attributes	Null	Default	Extra
no_sta	varchar(5)	utf8_general_ci		No		
latitude	double			No		
longitude	double			No		
altitude	double			No		

Tabel 4.19 Entitas Gps

Entitas r minimum merupakan tabel r yang berguna sebagai data paten (tidak bisa dirubah maupun diupdate), data paten tersebut merupakan data ketetapan bina marga Indonesia mengenai jari-jari minimum lengkung yang diperkenankan dalam perencanaan pembangunan jalan.

Field	Type	Collation	Attributes	Null	Default	Extra
v	float			No		
r_min	float			No		

Tabel 4.20 Entitas R Minimum

Entitas ls merupakan tabel ls yang berguna sebagai data pa*to*kan yang paten (tidak bisa diupdate), data pa*to*kan paten tersebut merupakan data ketetapan bina marga Indonesia mengenai panjang lengkung minimum dan elevasi yang diperkenankan dalam perencanaan pembangunan jalan berdasarkan pada nilai jari-jari yang dipergunakan dalam lengkung horisontal.

Field	Type	Collation	Attributes	Null	Default	Extra
id_ls_e	int(3)			No		auto_increment
R	float			No		
e1	float			Yes	*NULL*	
ls1	float			Yes	*NULL*	
e2	float			Yes	*NULL*	
ls2	float			Yes	*NULL*	
e3	float			Yes	*NULL*	
ls3	float			Yes	*NULL*	
e4	float			Yes	*NULL*	
ls4	float			Yes	*NULL*	

Tabel 4.21 Entitas Ls Minimum dan Elevasi

Entitas jarak henti merupakan tabel yang berisi mengenai informasi data pa*t*okan jarak henti yang didasarkan pada kecepatan rencana, data pa*t*okan tersebut merupakan data yang ditetapkan bina marga untuk menentukan jarak henti kendaraan.

Field	Type	Collation	Attributes	Null	Default	Extra
kecepatan	float			No		
jarak_henti	float			No		

Tabel 4.22 Entitas Jarak Henti

Entitas landai relatif merupakan tabel yang berisi informasi mengenai data pa*t*okan ketinggian relatif yang diijinkan berdasarkan kecepatan rencana, data pa*t*okan ditetapkan oleh bina marga Indonesia.

Field	Type	Collation	Attributes	Null	Default	Extra
v	float			No		
lr	float			No		

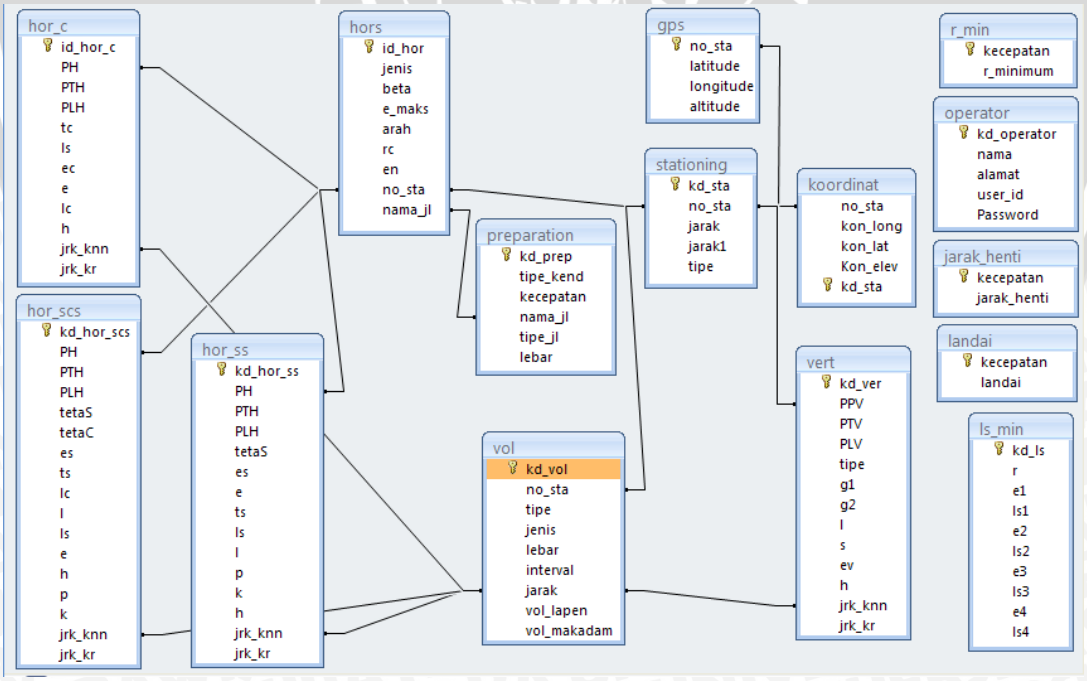
Tabel 4.23 Entitas Landai Relatif

Entitas *operator* merupakan tabel yang berisi mengenai data informasi pengguna-pengguna yang berhak mengakses sistem aplikasi ini.

Field	Type	Collation	Attributes	Null	Default	Extra
kd_operator	char(4)	utf8_general_ci		No		
nm_operator	varchar(25)	utf8_general_ci		No		
alamat	varchar(100)	utf8_general_ci		No		
user_id	varchar(25)	utf8_general_ci		No		
pass_id	varchar(25)	utf8_general_ci		No		

Tabel 4.24 Entitas *Operator*

Hubungan antar entitas pada tabel yang sudah dijabarkan sebelumnya dapat ditunjukkan pada gambar berikut.



Gambar 4.14 Entitas Sistem Perhitungan Pembangunan Jalan

4.2.2. Perancangan Site Map Sistem

Site map merupakan penggambaran dari menu-menu yang diterapkan pada sistem yang akan dibuat. Dari menu-menu yang akan dibuat maka dapat diketahui alur pengaksesan dari tiap-tiap *form* pada aplikasi ini.

Pada aplikasi berbasis java ini, tampilan utamanya berbentuk suatu *form* yang berisi main panel, menu bar, *to*ol bar dan status panel. *Form* ini merupakan *form* induk yang digunakan sebagai container dari semua *form-form* yang akan disusun. *Form-form* lain yang telah dipanggil melalui proses pemilihan pada menu bar ataupun *to*ol bar akan ditampilkan di dalam main panel.

Namun sebelum *form* utama muncul pada permulaan aplikasi dijalankan akan tampil *form login* yang berfungsi sebagai authentikasi data pengguna yang berhak mengakses. Apabila proses *login* berhasil maka *form* utama akan ditampilkan lengkap dengan semua menu-menunya, namun apabila proses *login* gagal maka *form* utama tetap ditampilkan tetapi dengan menu-menu yang terbatas saja.

Site map sistem yang dibuat dapat ditunjukkan pada gambar berikut.

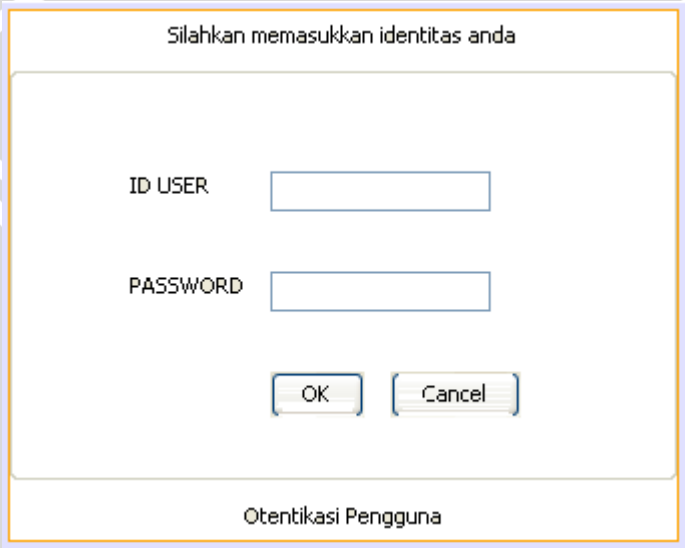
Gambar 4.15 Site Map Sistem

4.2.3. Perancangan *User* Interface

Interface merupakan penghubung antara sistem dengan *user* dan berfungsi untuk memudahkan interaksi antara keduanya. Dalam hal ini, perancangan *user* interface pada aplikasi perhitungan pembangunan jalan berbasiskan pada java swing, yang mendukung pemrograman *graphic user interface*. Adapun interface dari tiap-tiap *form* direncanakan sesuai dengan bentuk berikut.

1. Tampilan *Form Login*

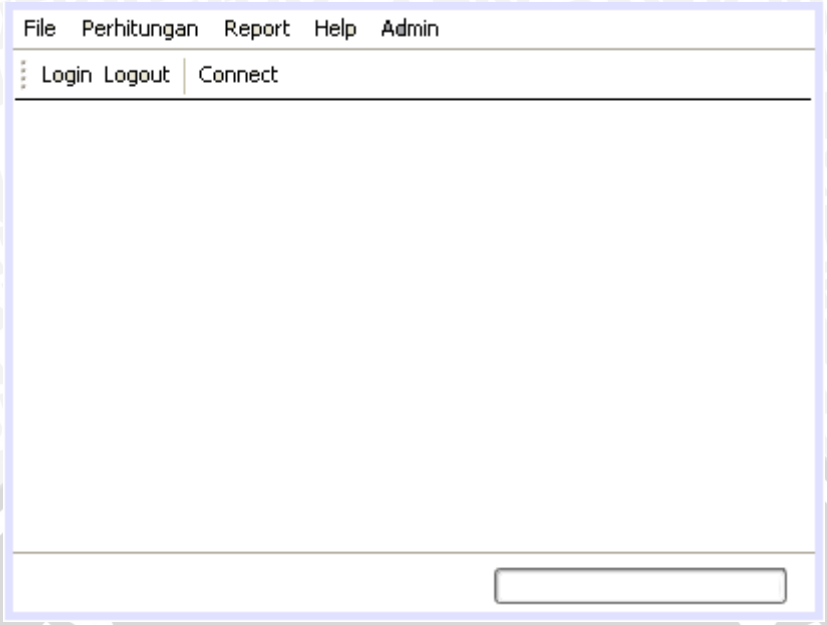
Form ini merupakan *form* pertama kali yang tampil ketika program dijalankan. *Form* ini juga merupakan *form* pengautentikasi pengguna-pengguna yang berhak mengakses aplikasi. Tampilan *form* tersebut digambarkan sebagai berikut.



Gambar 4.16 Tampilan *Form Login*

2. Tampilan *Form* Utama

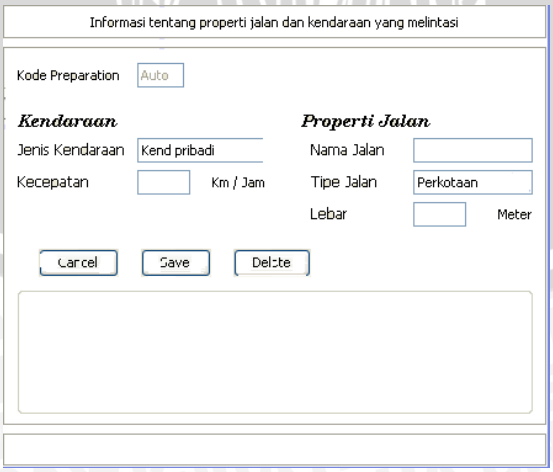
Form ini merupakan *form* induk, berfungsi sebagai container dari semua *form-form* yang ada di dalam aplikasi ini. *Form* ini berisikan 2 panel yakni panel utama sebagai container *form-form* dan panel status sebagai penunjuk aplikasi sedang melakukan suatu proses, dan 2 bar yakni menu bar sebagai tempat menu-menu utama sistem dan *to*ol bar sebagai tempat menu-menu pendukung yang berbentuk ikon dan huruf. Tampilan *form* utama digambarkan sebagai berikut.



Gambar 4.17 Tampilan *Form* Utama

3. Tampilan *Form Preparation*

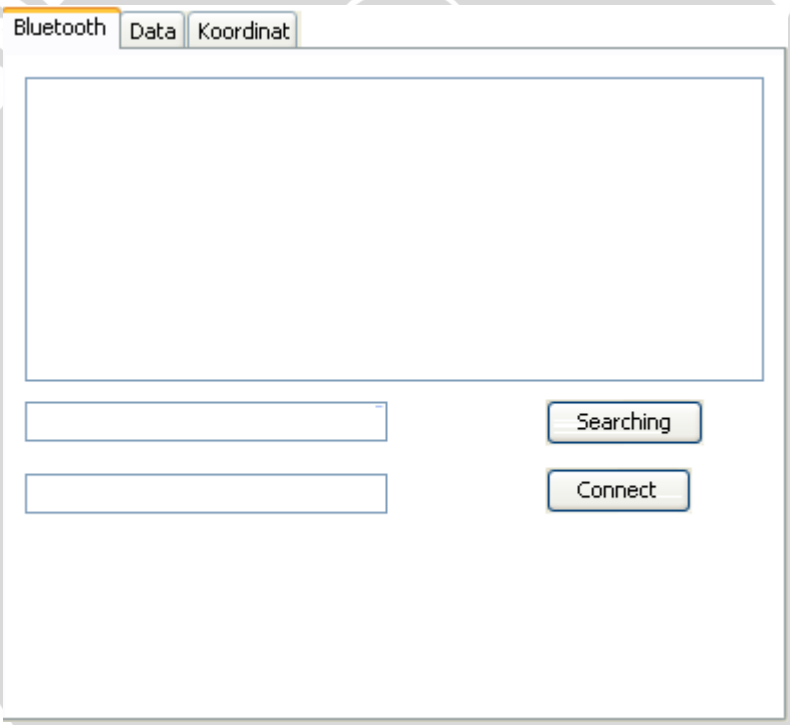
Pada *form* ini, *user* dapat memberikan informasi mengenai properti-properti tipe jalan dan kendaraan-kendaraan yang direncanakan akan melaluinya. Yang kemudian nantinya informasi tersebut akan disimpan dalam *storage* untuk diolah dalam perhitungan selanjutnya. Tampilan *form* ini digambarkan sebagai berikut.



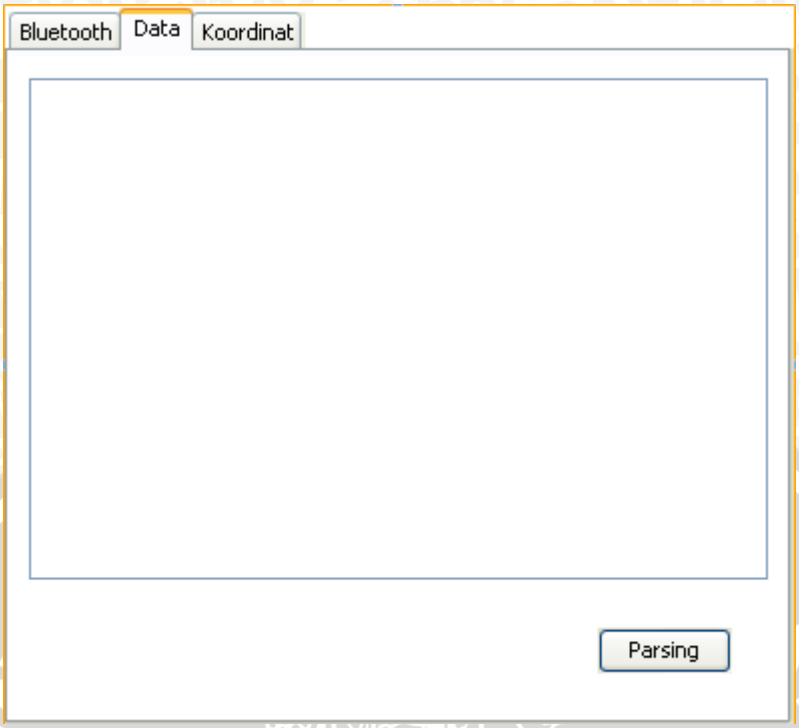
Gambar 4.18 Tampilan *Form Preparation*

4. Tampilan *Form* Koneksi

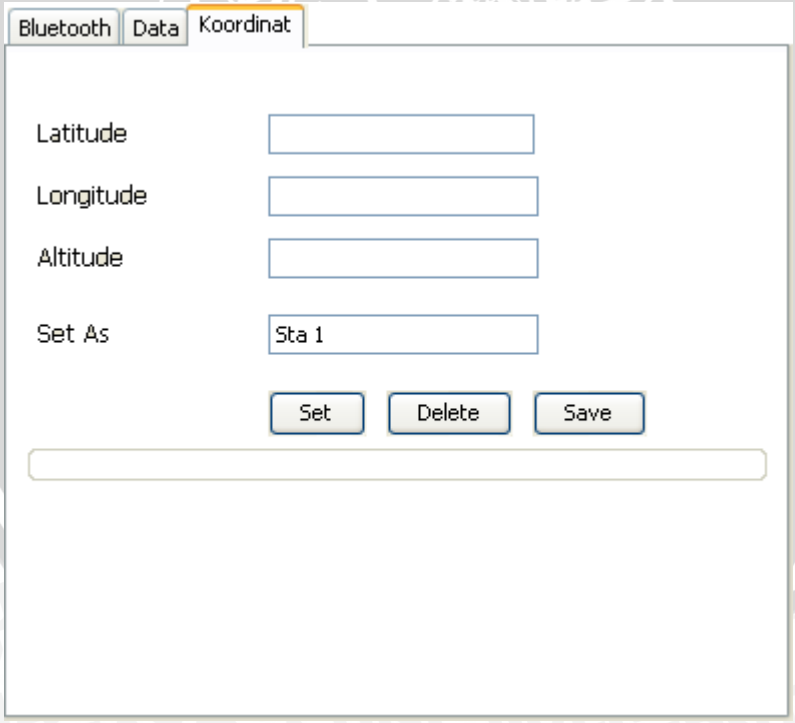
Form koneksi digunakan sebagai tampilan aplikasi dalam melakukan konetivitas terhadap *device* melalui *bluetooth* semisal gps ataupun *device* lainnya. Kemudian dari gps tersebut akan dibaca data masukannya yang berbentuk text NMEA, dan akhirnya text tersebut diparsing menjadi data koordinat *latitude*, *longitude* dan *altitude*nnya kemudian disimpan dalam *storage*. Tampilan *form* koneksi digambarkan sebagai berikut.



Gambar 4.19 Tampilan *Form* Koneksi tab *Bluetooth*



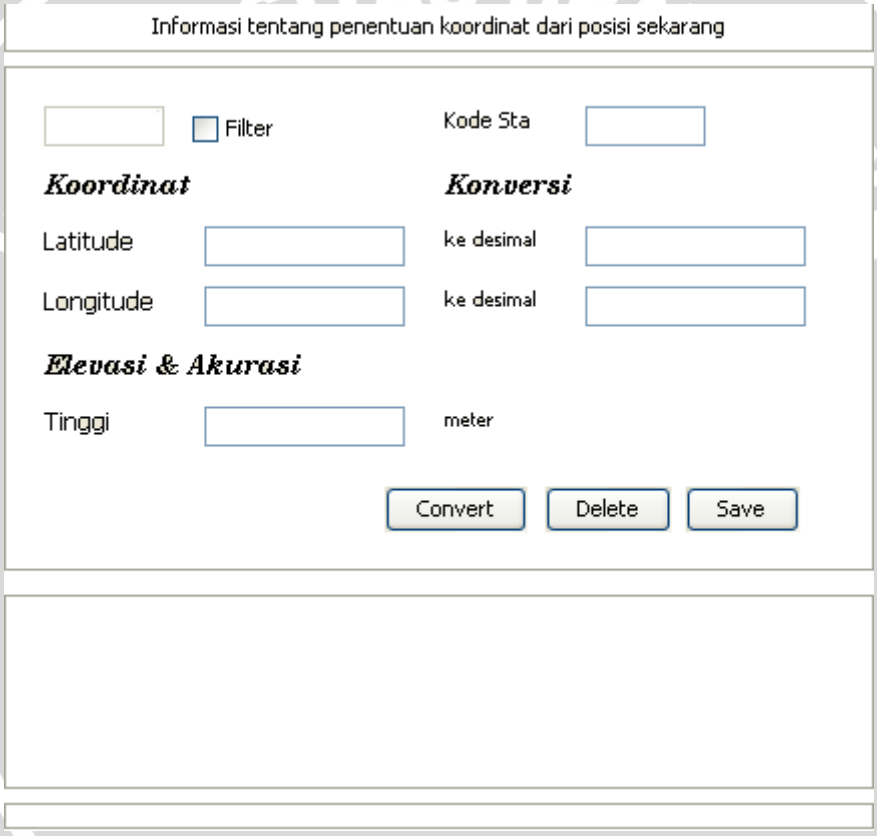
Gambar 4.20 Tampilan Koneksi tab Data



Gambar 4.21 Tampilan Koneksi tab Koordinat

5. Tampilan *Form* Koordinat

Form koordinat digunakan untuk mengambil nilai-nilai koordinat dari *form* koneksi yang bertipe *degree minute second* dan mengkonversinya ke bentuk *decimal degree*, kemudian data konversi tersebut diset satu-persatu nama *stasiuning*nya dan disimpan dalam *storage*. Tampilan *form* ini digambarkan sebagai berikut.

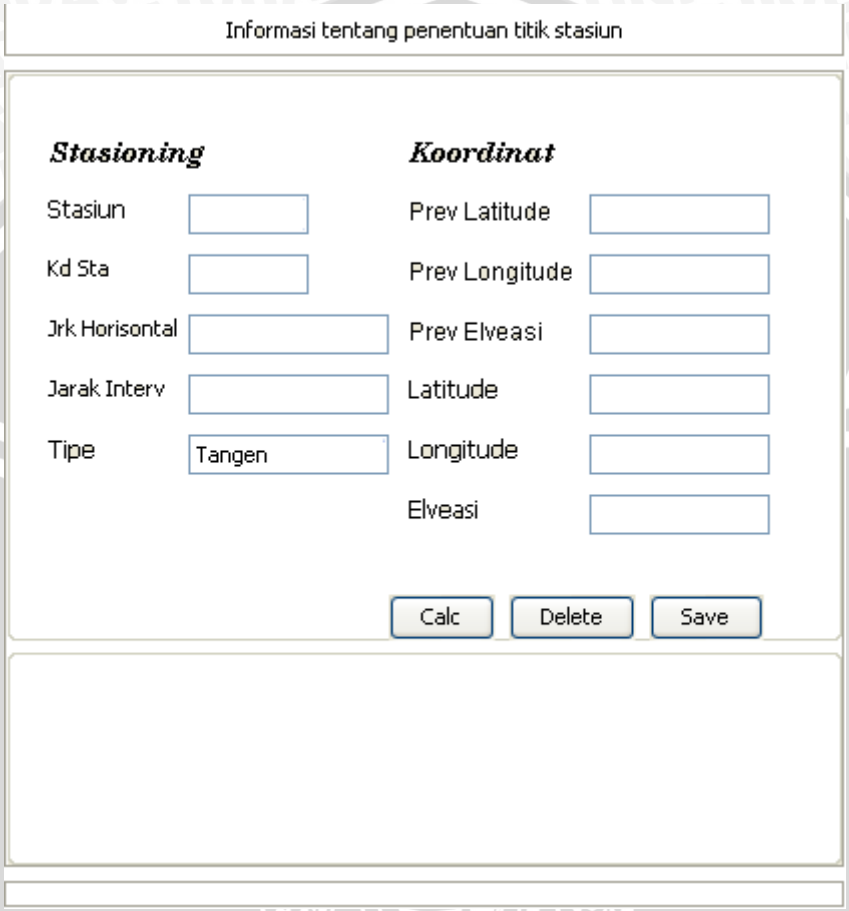


Gambar 4.22 Tampilan *Form* Koordinat

6. Tampilan *Form Stasiuning*

Form ini digunakan untuk mengambil data konversi koordinat dari *form* koordinat dan mempergunakannya sebagai masukan dalam perhitungan jarak titik stasiun saat ini dengan sebelumnya. Sekaligus mengeset jenis fisik dari lahan kerja

yang direncanakan apakah berbentu tangen (garis lurus) ataukah berbentuk lengkung. Dan menyimpannya dalam *storage*. Tampilan *form* ini dapat digambarkan sebagai berikut.

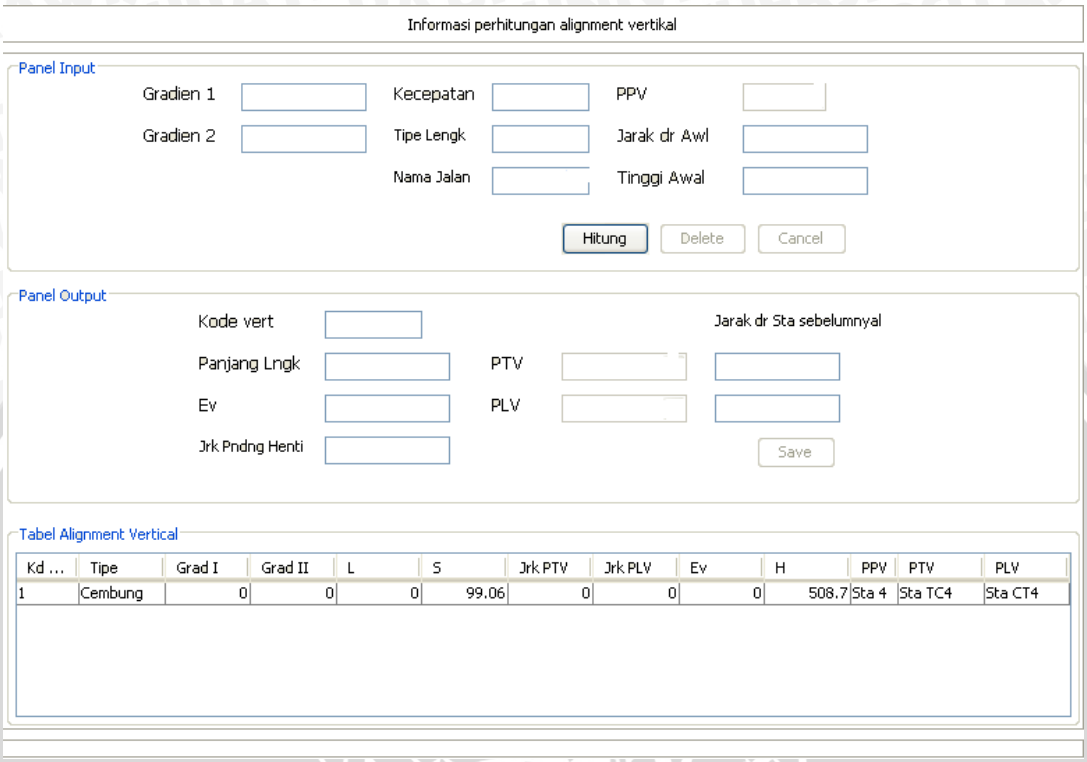


Gambar 4.23 Tampilan *Form Stationing*

7. Tampilan *Form Alignment* Vertikal

Form ini digunakan untuk mengambil data informasi properti jalan dan informasi *stasiuning* yang bertipe *alignment* vertikal dari *form-form* sebelumnya, kemudian dari data-data tersebut diperhitungkan *alignment* vertikalnya sehingga didapatkan keluaran yang berbentuk variabel-variabel baru. Kemudian variabel-

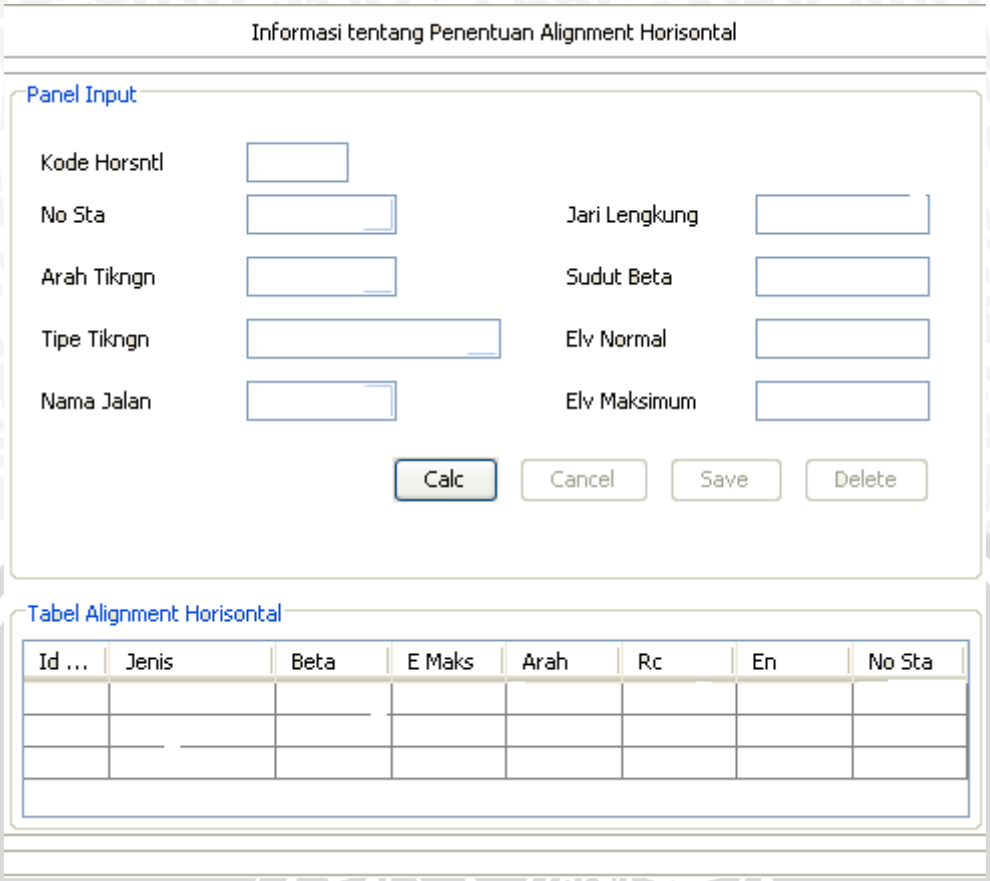
variabel tersebut disimpan dalam *storage*. Tampilan *form* ini digambarkan sebagai berikut.



Gambar 4.24 Tampilan *Form* Vertikal

8. Tampilan *Form Alignment* Horisontal

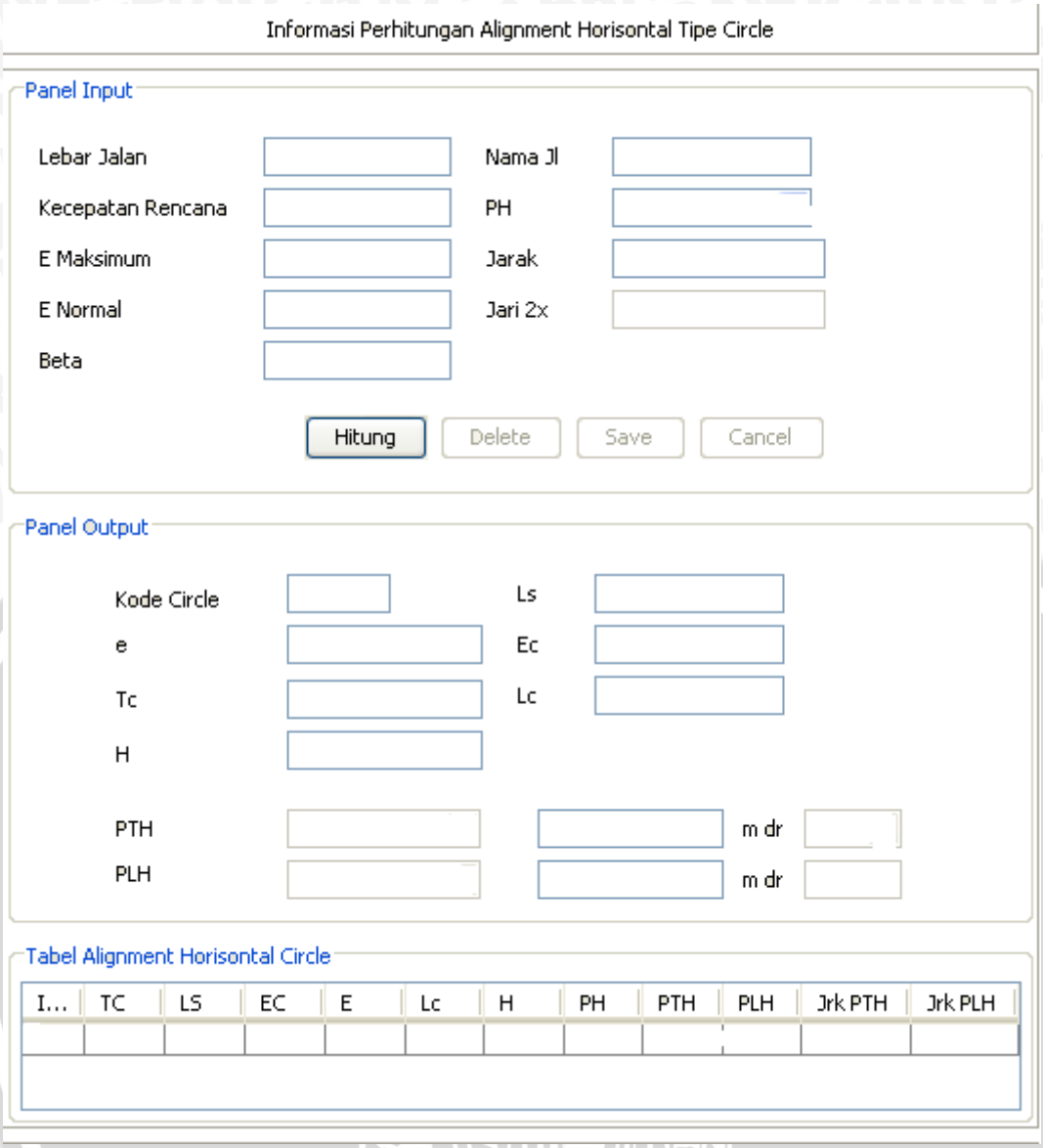
Form ini digunakan untuk mengambil data informasi properti jalan dan informasi *stasiuning* yang bertipe *alignment* horisontal dari *form-form* sebelumnya, kemudian dari data-data tersebut diperhitungkan *alignment* horisontalnya sehingga didapatkan keluaran yang berbentu variabel-variabel baru yang dapat dipergunakan untuk perhitungan selanjutnya. Kemudian variabel tersebut disimpan dalam *storage*. Tampilan *form* ini digambarkan sebagai berikut.



Gambar 4.25 Tampilan *Form* Horisontal

9. Tampilan *Form Alignment* Horisontal tipe *Circle*

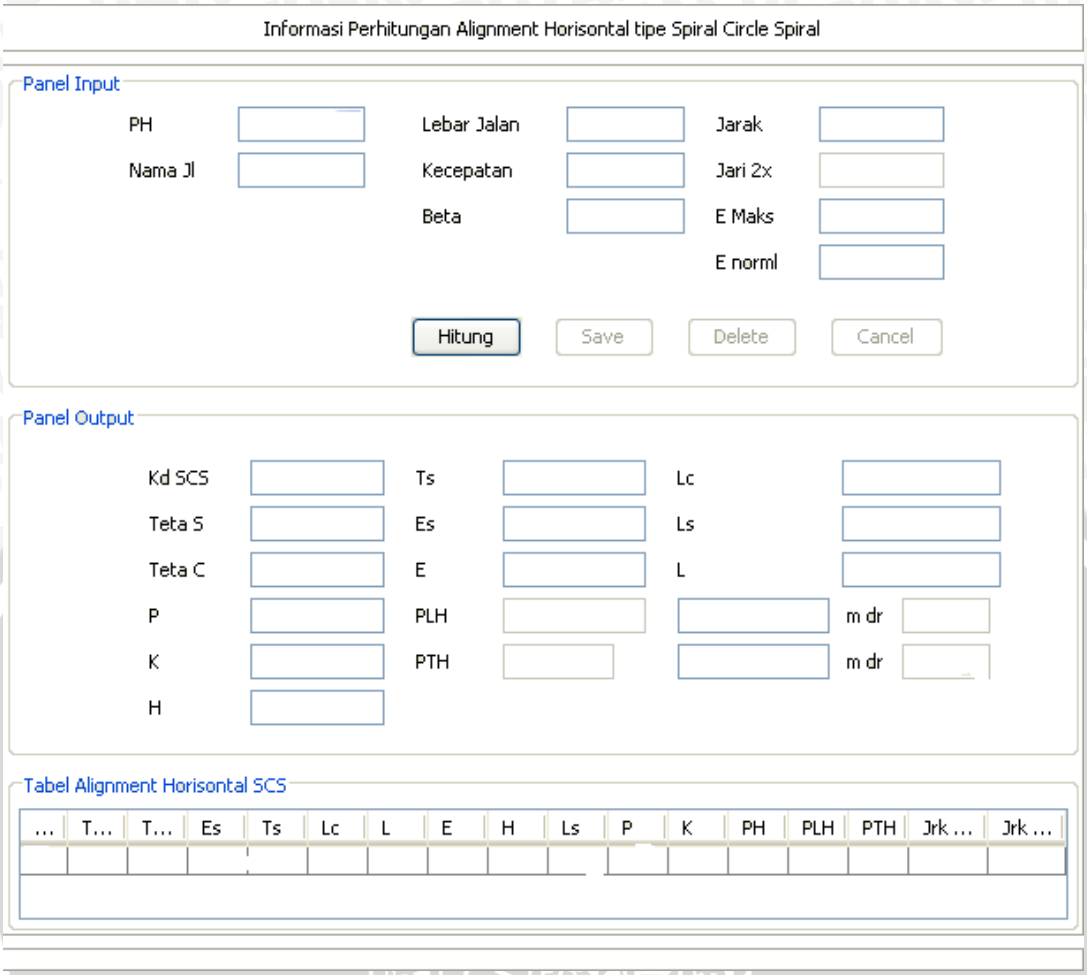
Form ini merupakan perpanjangan *form* horisontal, *form* ini digunakan untuk mengambil variabel-variabel yang berjenis *circle* dari *form* horisontal untuk diperhitungkan *alignment* horisontal tipe *circle*nya, dari perhitungan didapatkan variabel-variabel baru dan disimpan dalam *storage*. *Form* ini digambarkan sebagai berikut.



Gambar 4.26 Tampilan *Form* Horisontal tipe *Circle*

10. Tampilan *Form Alignment* Horisontal tipe Spiral-*Circle*-Spiral

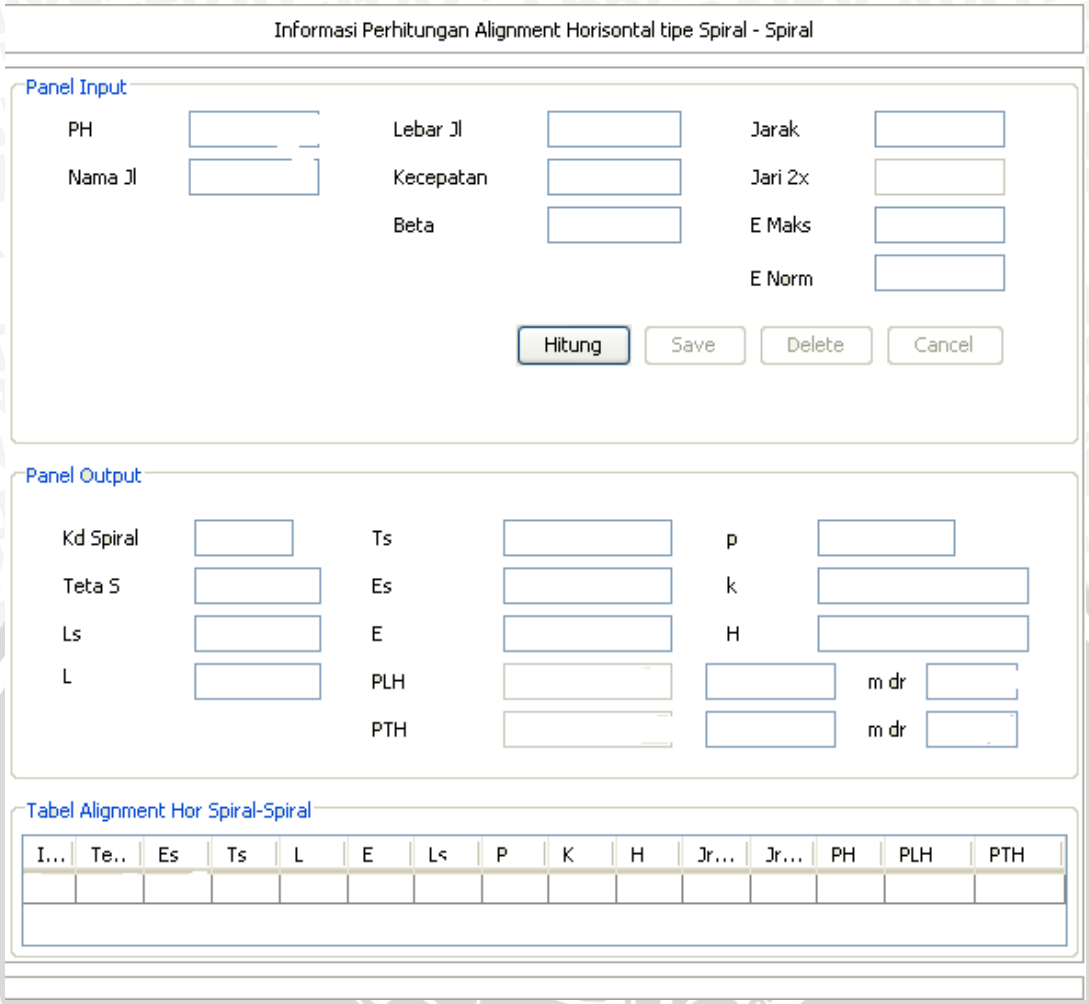
Form ini merupakan perpanjangan *form* horisontal, *form* ini digunakan untuk mengambil variabel-variabel yang berjenis spiral-*circle*-spiral dari *form* horisontal untuk diperhitungkan *alignment* horisontal tipe spiral-*circle*-spiralnya, dari perhitungan didapatkan variabel-variabel baru dan disimpan dalam *storage*. *Form* ini digambarkan sebagai berikut.



Gambar 4.27 Tampilan *Form* Horisontal tipe Spiral-*Circle*-Spiral

11. Tampilan *Form Alignment* Horisontal tipe Spiral-Spiral

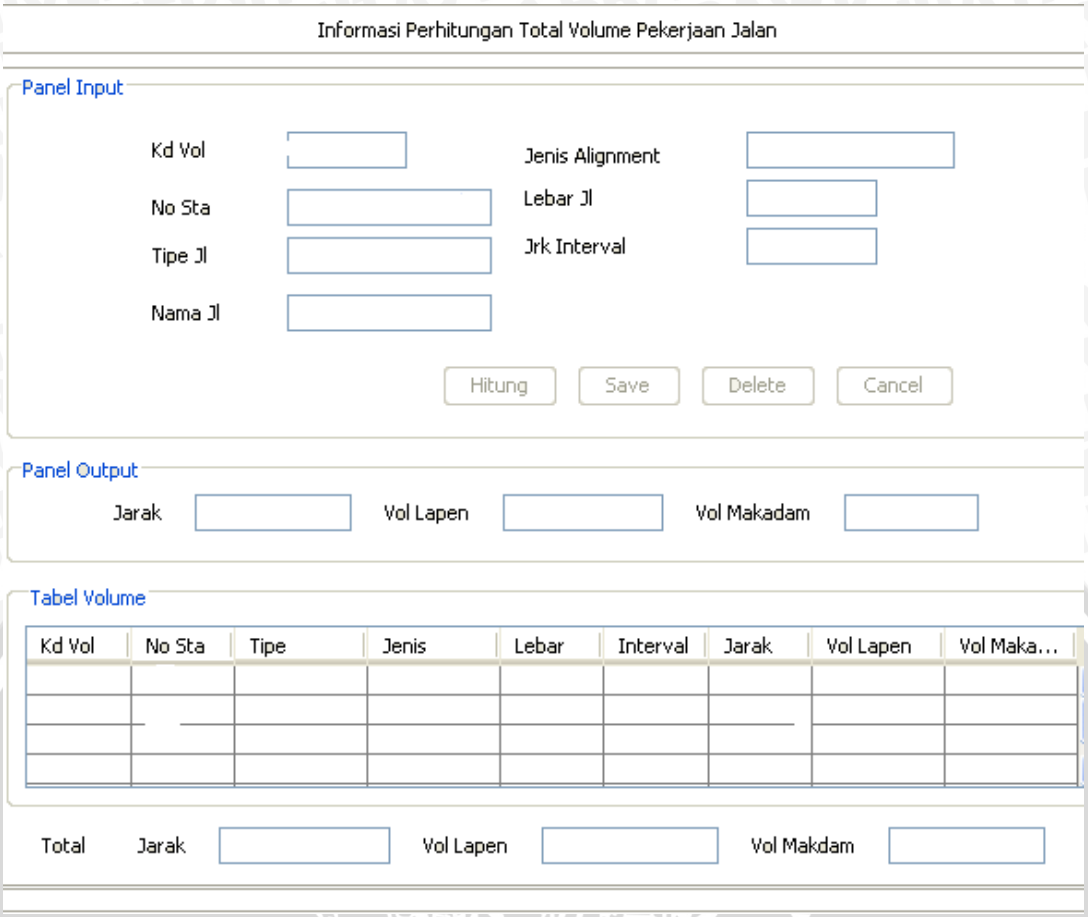
Form ini merupakan perpanjangan *form* horisontal, *form* ini digunakan untuk mengambil variabel-variabel yang berjenis spiral-spiral dari *form* horisontal untuk diperhitungkan *alignment* horisontal tipe spiral-spiralnya, dari perhitungan didapatkan variabel-variabel baru dan disimpan dalam *storage*. *Form* ini digambarkan sebagai berikut.



Gambar 4.28 Tampilan *Form* Horisontal tipe Spiral-Spiral

12. Tampilan *Form* Volume

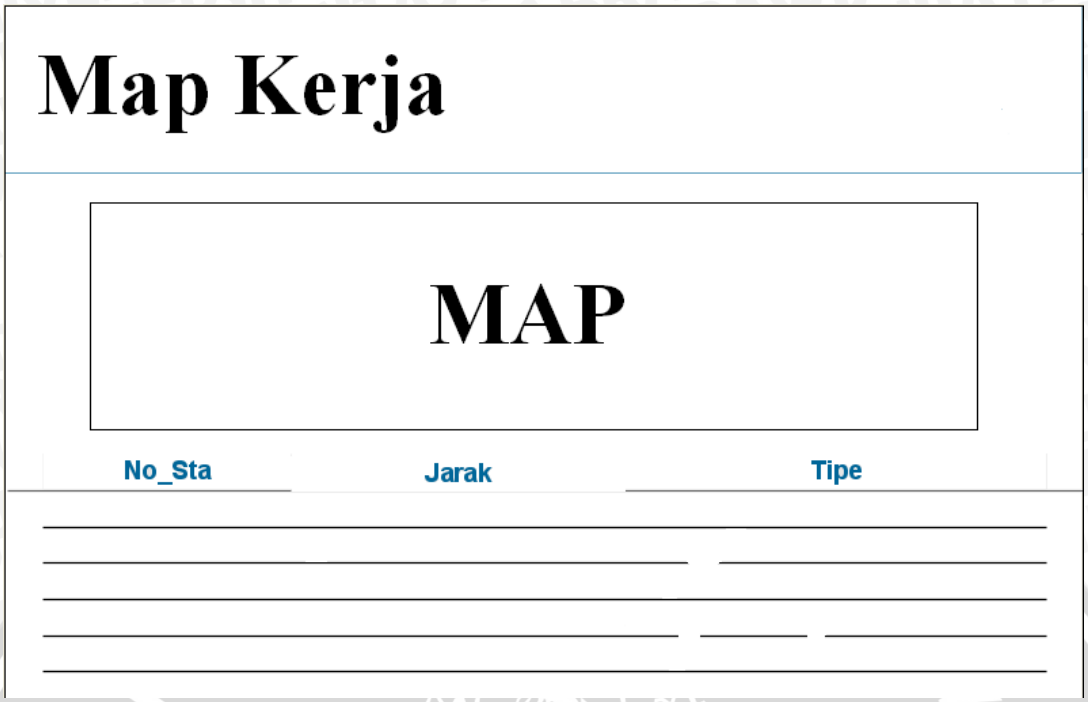
Form ini merupakan *form* perhitungan terakhir di dalam aplikasi ini, karena di dalam *form* ini terdapat variabel-variabel keluaran terakhir yang dibutuhkan oleh *user* dalam merencanakan pekerjaan pembangunan jalan. Tampilan dari *form* ini dapat digambarkan sebagai berikut.



Gambar 4.29 Tampilan *Form* Volume

13. Tampilan *Form* Laporan Map Kerja

Form ini merupakan *form* daftar laporan dari data koordinat yang dilengkapi dengan sebuah map untuk menggambarkan lokasi pekerjaan pembangunan jalan nantinya. Tampilan dari *form* ini digambarkan sebagai berikut.



Gambar 4.30 Tampilan Map Kerja

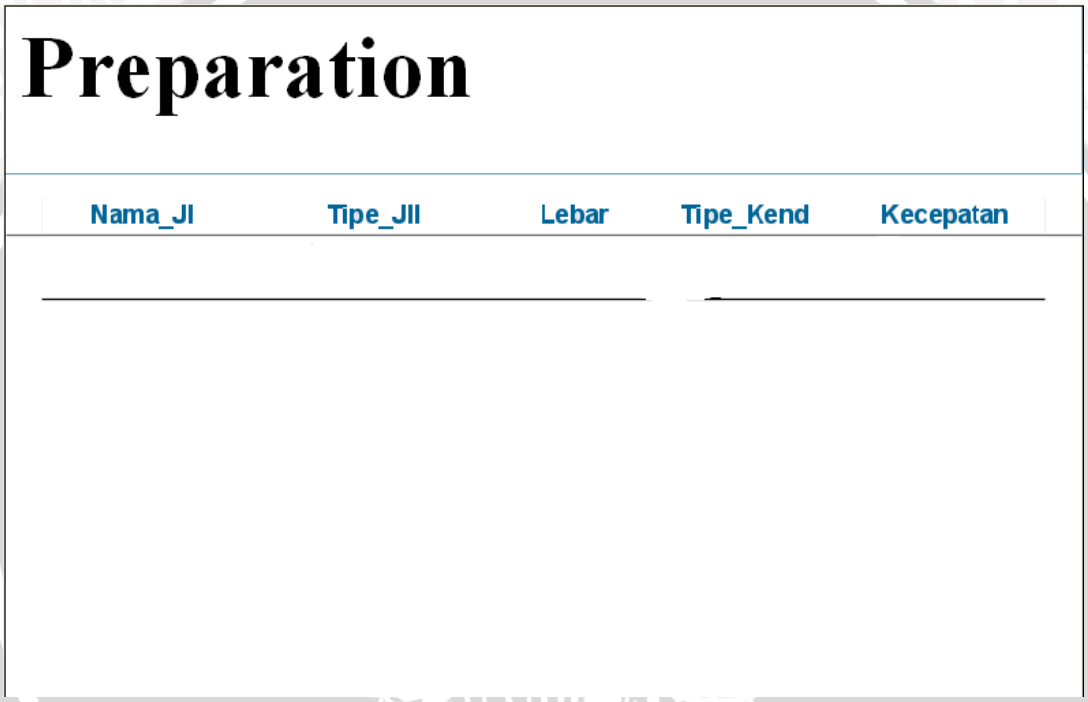
14. Tampilan *Form* Laporan Koordinat

Form ini merupakan *form* daftar laporan dari data konversi koordinat yang berbentu *decimal degree*. Dari *form* ini nantiny dapat dilakukan proses printing dan penyimpanan halaman *form* laporan. Tampilan *form* ini dapat digambarkan sebagai berikut.

Gambar 4.31 Tampilan Laporan Koordinat

15. Tampilan *Form* Laporan Properti Jalan

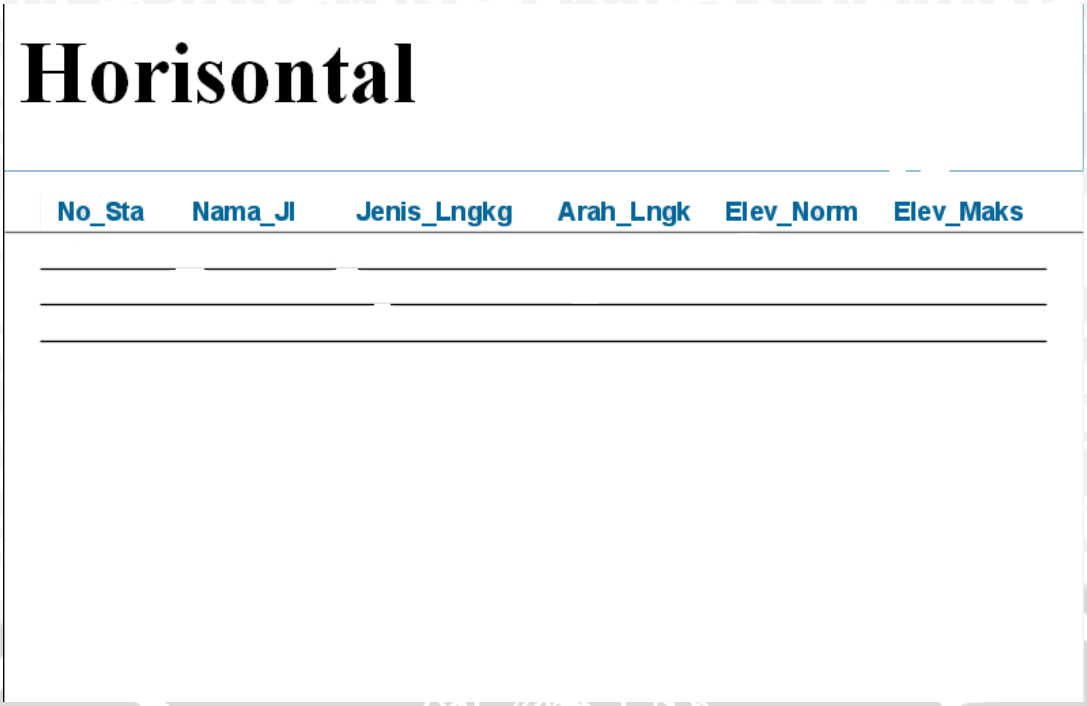
Form ini merupakan *form* daftar laporan dari data properti jalan. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.



Gambar 4.32 Tampilan Laporan *Preparation*

16. Tampilan *Form* Laporan *Alignment* Horisontal

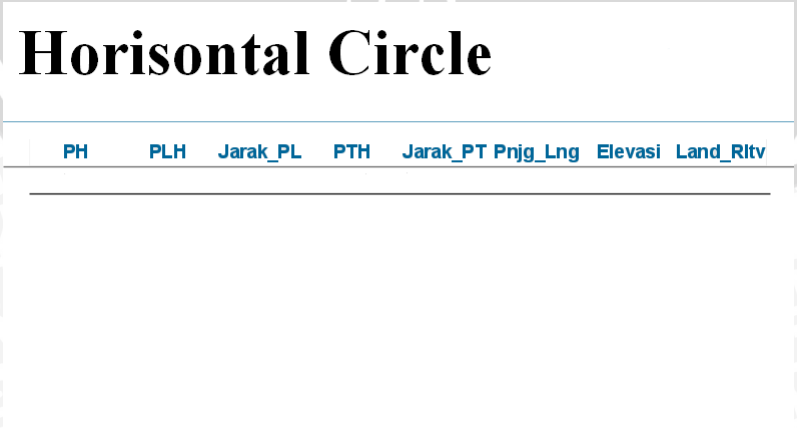
Form ini merupakan *form* daftar laporan dari data *alignment* horisontal. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.



Gambar 4.33 Tampilan Laporan Horisontal

17. Tampilan *Form* Laporan *Alignment* Horisontal tipe *Circle*

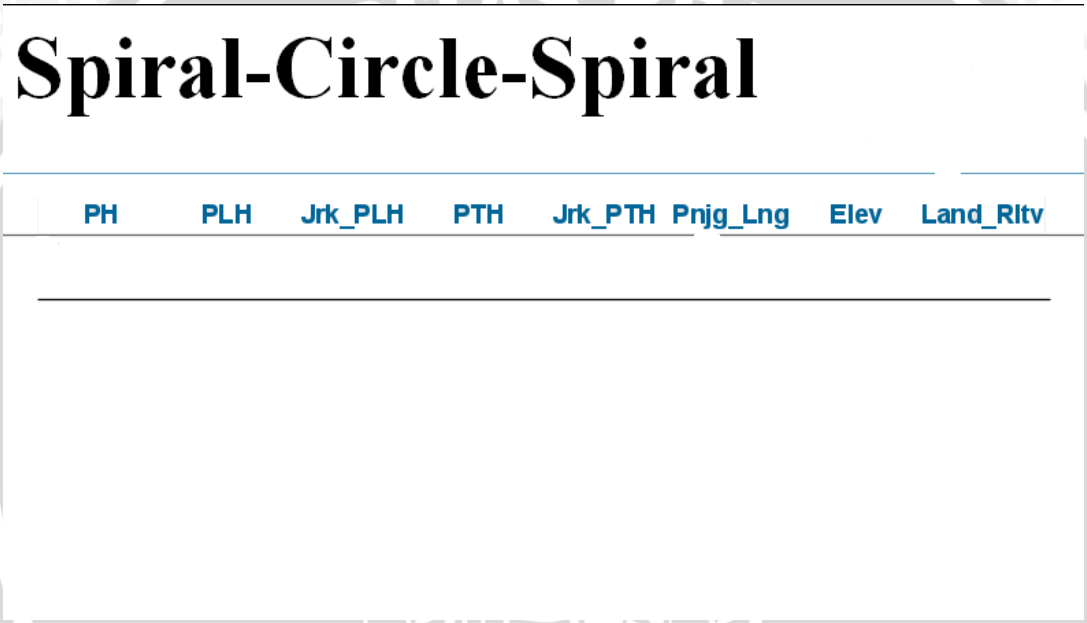
Form ini merupakan *form* daftar laporan dari data *alignment* horisontal tipe *circle*. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.



Gambar 4.34 Tampilan Laporan Horisontal tipe *Circle*

18. Tampilan *Form* Laporan *Alignment* Horisontal tipe Spiral-*Circle*-Spiral

Form ini merupakan *form* daftar laporan dari data *alignment* horisontal tipe spiral-*circle*-spiral. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.



Gambar 4.35 Tampilan Laporan Horisontal tipe Spiral-*Circle*-Spiral

19. Tampilan *Form* Laporan *Alignment* Horisontal tipe Spiral-Spiral

Form ini merupakan *form* daftar laporan dari data *alignment* horisontal tipe spiral-spiral. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.

Gambar 4.36 Tampilan Laporan Horisontal tipe Spiral-Spiral

20. Tampilan *Form* Laporan *Alignment* Vertikal

Form ini merupakan *form* daftar laporan dari data *alignment* vertikal. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.

Gambar 4.37 Tampilan Laporan Vertikal

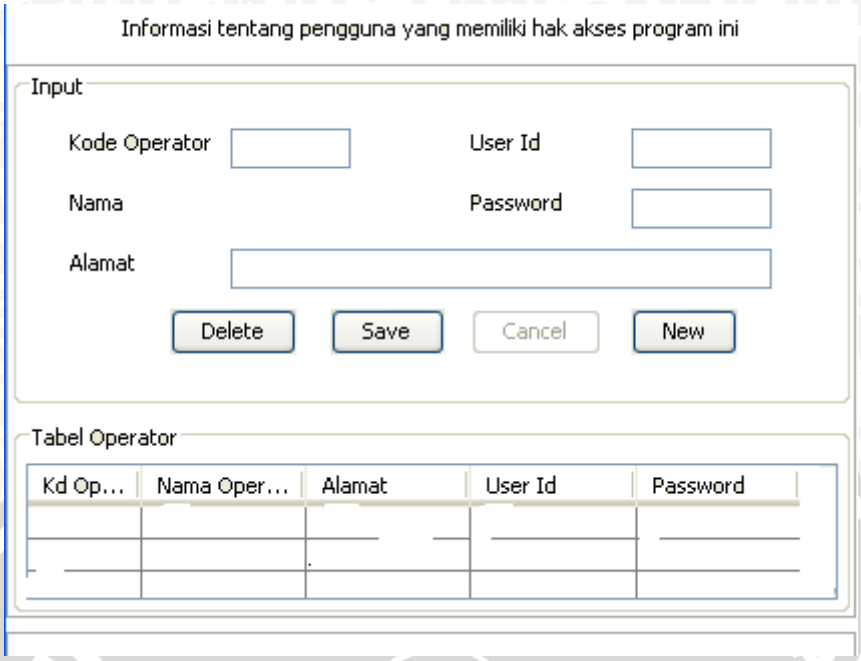
21. Tampilan *Form* Laporan Volume Kerja

Form ini merupakan *form* daftar laporan dari data volume kerja. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.

Gambar 4.38 Tampilan Laporan Volume Kerja

22. Tampilan *Form Operator*

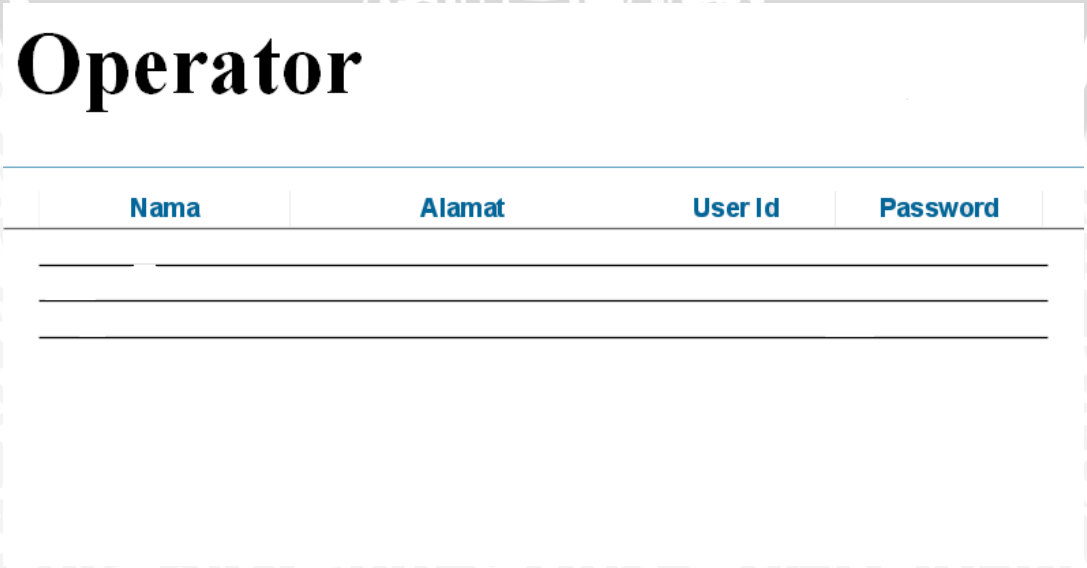
Form ini merupakan *form* yang digunakan untuk menentukan siapa-siapa saja yang berhak mengakses aplikasi ini. Salah satu data-data pada *form* ini adalah data id *user* dan *password* yang dipergunakan untuk proses autentikasi pada permulaan aplikasi berjalan. Data-data yang ada dalam *form* ini disimpan dalam *storage*. Tampilan dari *form* ini dapat digambarkan sebagai berikut.



Gambar 4.39 Tampilan *Operator*

23. Tampilan *Form* Laporan *Operator*

Form ini merupakan *form* daftar laporan dari data *operator*. Dari *form* ini nantinya *user* dapat melakukan proses printing dan penyimpanan halaman *form* laporana berbentuk html. Tampilan *form* ini dapat digambarkan sebagai berikut.



Gambar 4.40 Tampilan Laporan *Operator*

BAB V

IMPLEMENTASI

Bab ini membahas tentang implementasi perangkat lunak berdasarkan hasil yang didapatkan dari analisis kebutuhan dan proses perancangan perangkat lunak sebelumnya. Pembahasan terdiri dari penjelasan tentang spesifikasi lingkungan (spesifikasi perangkat keras dan perangkat lunak) di mana sistem diimplementasikan, implementasi tiap kelas pada *file* program, implementasi algoritma dan implementasi antarmuka aplikasi.

5.1. Spesifikasi Sistem

Dari hasil analisis kebutuhan dan perancangan yang diuraikan pada bab IV, dilakukan implementasi agar sistem ini bisa berfungsi sesuai dengan kebutuhan. Sistem diimplementasikan pada suatu lingkungan dengan spesifikasi perangkat keras dan lunak seperti ditunjukkan pada sub bab 5.1.1 dan sub bab 5.1.2.

5.1.1. Spesifikasi Perangkat Keras

Spesifikasi minimum perangkat keras yang digunakan untuk implementasi aplikasi perhitungan pembangunan jalan ini ditunjukkan berikut.

Spesifikasi Minimum Perangkat Keras	
Prosesor	133 Mhz Pentium III
Memori (RAM)	512 Mb
Disk Space	2 GB
VGA	On Board
Bluetooth	Versi 2.0+EDR
GPS *Bluetooth*	Holux M 241

Tabel 5.1 Spesifikasi Minimum Perangkat Keras

5.1.2. Spesifikasi Perangkat Lunak

Spesifikasi dari perangkat lunak berikut adalah sebagai berikut :

Spesifikasi Perangkat Lunak	
Grafick Jalan	*Alignment* jalan diproyeksikan dalam bentuk grafik untuk mempermudah pengolahan data oleh user
Login autentikasi	Pada awal program dilakukan autentikasi user yang berhak mengakses sistem, guna mendukung keamanan data
Report HTML	Hasil laporan berbentuk HTML sehingga mempermudahkan user melakukan penyimpanan dan pencetakan data
Mapping	Gambaran jalan ditampilkan dalam bentuk map
Bluetooth supported	Mendukung pemasukan data melalui GPS *device* melalui media bluetooth

Tabel 5.2 spesifikasi perangkat lunak

Untuk membangun sistem ini dibutuhkan beberapa perangkat lunak yang dapat dilihat pada tabel berikut.

Peralatan Perangkat Lunak	
Sistem Operasi	Windows Xp version 2002
Bahasa Pemerograman	Java 2 Standart *Edit*ion
Tools Program	JDK 1.4.2_03 Jasper Report-3.7.1 BlueCove 2.1.0 Swingx-ws-2010_03_21 JXMapKit
IDE (Integrated Development Kit)	NetBeans 6.8

Adapun sistem ini menggunakan library-library yang sudah disediakan oleh JDK dan library-library lain yang bersifat *open source*, library tersebut ditunjukkan sebagai berikut :

a. Jasper Report, adalah library yang disediakan oleh *Jasper Company* yang bersifat open source, library ini dipergunakan untuk mengolah data-data hasil perhitungan untuk ditampilkan dalam bentuk HTML, tampilan dari HTML tersebut diolah dalam aplikasi iReport, kemudian dari sistem

dilakukan pengkoneksian data-data jasper report tersebut dengan aplikasi, setelah koneksi selesai maka dilakukan pemanggilan-pemanggilan form HTML hasil olahan dari iReport dari dalam sistem.

b. BlueCove, adalah library yang dipergunakan untuk mengkoneksikan suatu sistem dengan *device* disekitar sistem melalui media *bluetooth*. Proses pengkoneksian diawali dengan pendefinisian DiscoveryListener yang memiliki 4 fungsi untuk pencarian *device bluetooth.*

c. JDK, adalah library standart yang disediakan oleh sun java guna mendukung pengolahan objek. Library-library didalam JDK yang dipergunakan dalam sistem ditunjukkan sebagai berikut :

   i. Library Math, dipergunakan untuk melakukan proses perhitungan matematik dan numerik yang hanya tersedia untuk J2SE.

   ii. Library Swing, dipergunakan untuk membuat tampilan dari interface sistem kepada user.

   iii. Library IO, dipergunakan untuk melakukan pembacaan data hasil *methode* pengkoneksian *device* sebelumnya.

   iv. Library SQL, dipergunakan untuk penyimpanan dan pemanggilan data-data yang telah disimpan pada database sistem.

d. JXMapKit, adalah library yang dipergunakan untuk melakukan pengolahan pada proses mapping, penampilan map diawali dengan download dari openstreetmap.com, kemudian penentuan titik pusat dari tampilan map, penentuan waypoint dari setiap titik *station*, dan diakhiri dengan penggambaran bentuk jalan ke dalam map.

e. Model, adalah library yang dibuat oleh perancang guna mengkoneksikan sistem dengan database yang dibuat pada MySql, serta menampilkan database tersebut ke dalam report dan tiap-tiap form pada sistem.

5.2. Implementasi Klas pada *File* Program

Klas-klas yang telah didesain pada proses perancangan, masing-masing direalisasikan pada sebuah *file* program yang berekstensikan *.java. pada tabel 5.3 dijabarkan pasangan antara klas dengan *file* program yang digunakan untuk implementasinya.

No.	Nama Kelas	Nama *File* Program
1	Hitung Jalan	HitungJalan.java
2	*Login*	*LoginForm*.java
3	Koneksi	*Connect*.java
4	Koordinat	Coordinate.java
5	*Stationing*	*Stationing*.java
6	*Preparation*	*Preparation*.java
7	*Alignment* vertikal	Vertikal.java
8	*Alignment* horisontal	Horisontal.java
9	Horisontal tipe *circle*	Hor*Circle*.java
10	Horisontal tipe spiral-*circle*-spiral	HorSCS.java
11	Horisontal tipe spiral-spiral	HorSS.java
12	Volume kerja	Volume.java
13	*Operator*	*Operator*.java
14	Map	Map.java

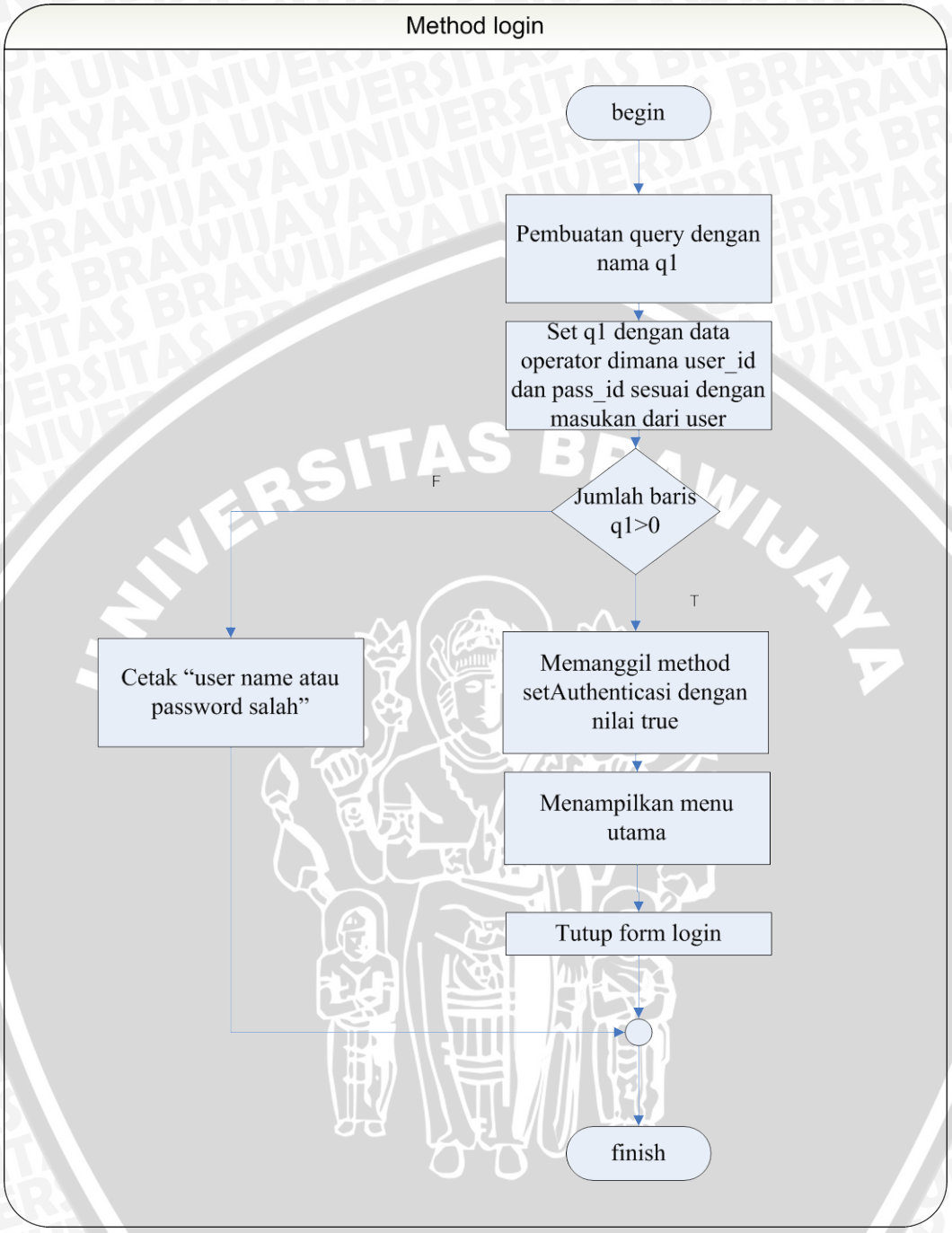
Tabel 5.3 Implementasi Klas dan *File* Program

5.3. Implementasi Algoritma

Pada sistem ini, terdapat beberapa proses utama (pokok). Diantara proses utama tersebut adalah proses *authentikasi user*, proses komunikasi data melalui jaringan koneksi *bluetooth*, proses perhitungan, proses penyimpanan data ke dalam *storage*, proses penghapusan data dan proses penampilan data ke dalam bentuk tabel.

5.3.1. Implementasi Algoritma *Login*

Proses *login* ini merupakan proses awal berjalannya program yang dipergunakan untuk *authentikasi* pengguna yang berhak mengakses sistem sepenuhnya. Adapun algoritma yang digunakan adalah membandingkan antara inputan yang diketikkan oleh *user* dengan data *operator* yang tersimpan di dalam *database*. Implementasi dari algoritma *login* ini terdapat pada klas *loginForm*.java seperti yang ditunjukkan dalam algoritma sebagai berikut.

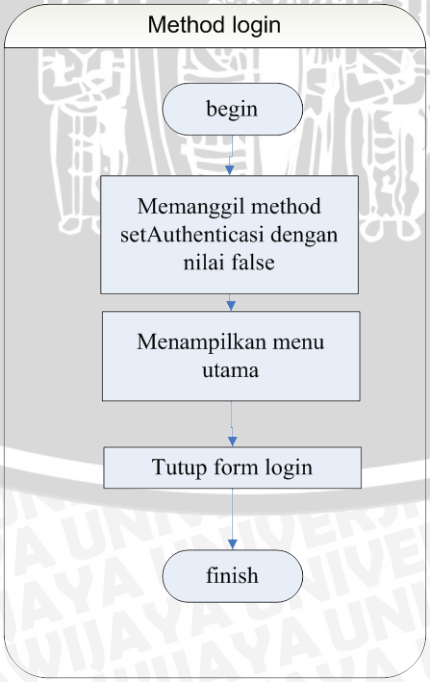


Algoritma 5.1 Proses *Login*

Algoritma *login* diatas digunakan jika *user* telah menginputkan data *user*nam dan *password* kemudian menekan tombol ok, Penjelasan dari algoritma *login* diatas adalah sebagai berikut :

a. Pada baris pertama adalah inisialisasi proses *login* ketika tombol ok ditekan dengan menggunakan eksepsi-eksepsi.
b. Pada baris ke 2 hingga 4 adalah proses pembuatan query dengan data yang diambil dari tabel *operator* dimana datanya sesuai dengan inputan *user* pada *field* txt*User* dan txt*Password*.
c. Pada baris ke 6 dan 7 adalah proses aktifasi dari query yang telah didefinisikan sebelumnnya.
d. Pada baris ke 8 hingga 14 dituliskan bahwa jika jumlah baris pada query yang telah disusun lebih besar dari 0 maka set *authentikasi* pada HitungJalanApp sebagai true dan tampilkan semua *main* menu pada HitungJalanApp, jika tidak maka tampilkan pemberitahuan bahwa *user*name atau *password* yang dientrikan salah.

Jika pada *form login user* menekan tombol *cancel* maka algoritma yang dijalankan adalah sebagai berikut.



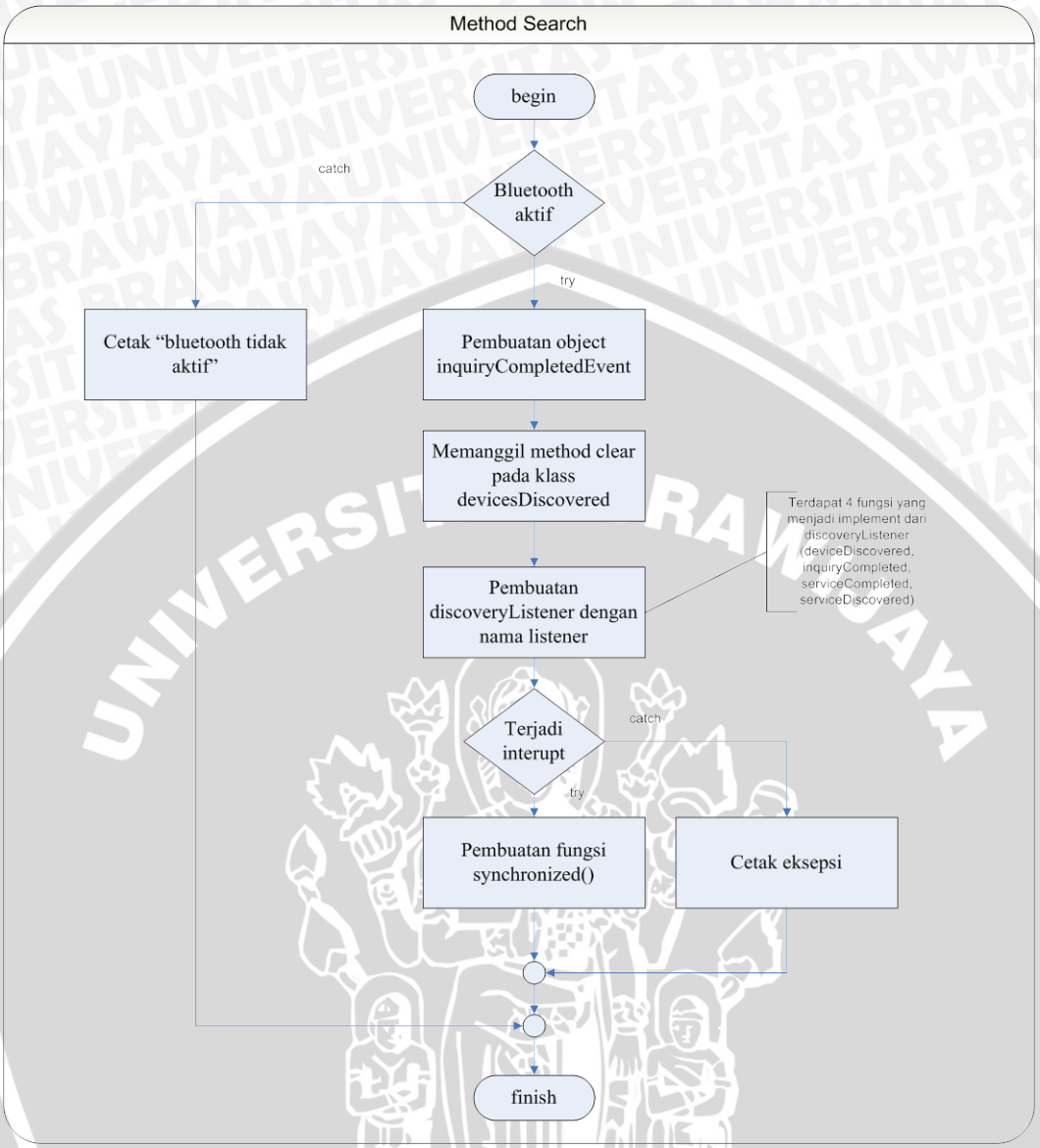
Algoritma 5.2 Proses *Cancel* pada *Login*

Penjelasan algoritma diatas adalah ketika tombol *cancel* ditekan maka set *authentucation* pada HitungJalanApp sebagai *false* dan tampilkan menu-menu yang disediakan untuk *authentikasi false*, kemudian tutup *form login*.

5.3.2. Implementasi Algoritma Koneksi *Bluetooth*

Koneksi yang dilakukan dalam sistem ini adalah koneksi satu arah dimana sistem hanya melakukan pembacaan data dari data *streaming device* yang telah terkoneksi dengan sistem. Dengan kata lain urutan proses yang terjadi dalam algoritma koneksi *bluetooth* ini adalah pengaktifan mode pencarian pada *device bluetooth* komputer, penampilan *device-device bluetooth* yang terdeteksi oleh komputer, pengkoneksian *device* sesuai dengan pilihan *user*, penampilan data *streaming device*, pemarsingan data *streaming* tersebut menjadi bentuk *latitude*, *longitude* dan *altitude*, kemudian pengesetan koordinat dengan nomer stasiunnya oleh *user*, dan terakhir adalah proses penyimpanan data ke dalam *storage*.

Implementasi dari algoritma proses koneksi *bluetooth* dapat ditunjukkan sebagai berikut.

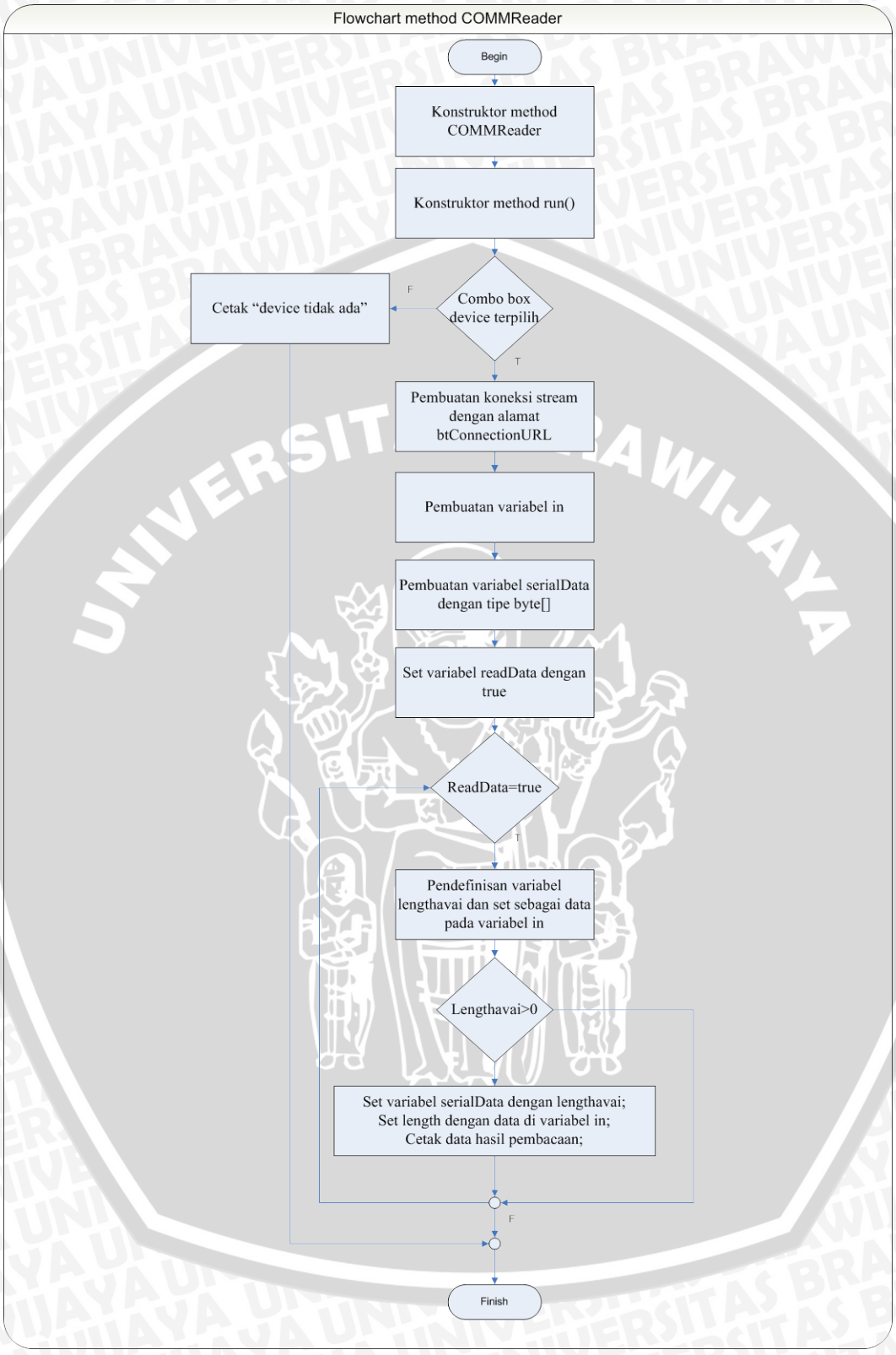


Algoritma 5.3 Proses Searching *Device* pada Konek

Algoritma diatas akan berjalan ketika tombol search pada *form* konek ditekan, prosesnya diawali dengan pendefinisian DiscoveryListener sebagai object baru, kemudian pemanggilan method clear pada kelas devicesDiscovered, dilanjut dengan pendeskripsian dari discoveryListener yang didalamnya terdapat 4 fungsi, yakni fungsi *deviceDiscovered* yang berguna memanggil nama dan alamat *bluetooth* yang terdeteksi, diikuti dengan pemanggilan fungsi *inquiryCompleted* yang berguna

sebagai pemberitahuan kepada *user* bahwa pendeteksian *device bluetooth* sedang berjalan, setelah itu fungsi *serviceCompleted* yang berguna untuk memberitahukan user bahwa pencarian telah selesai kemudian diakhiri dengan pendefinisian *method serviceDiscovered* yang berguna untuk menampilkan alamat dari *device* yang diseleksi oleh *user* serta menyinkronisasi *method*e *inquiryCompleted* dengan menampilkan pemberitahuan proses dan jumlah *device* yang diketemukan jika berhasil melakukan koneksi dan menampilkan eksepsi jika gagal dalam melakukan koneksi.

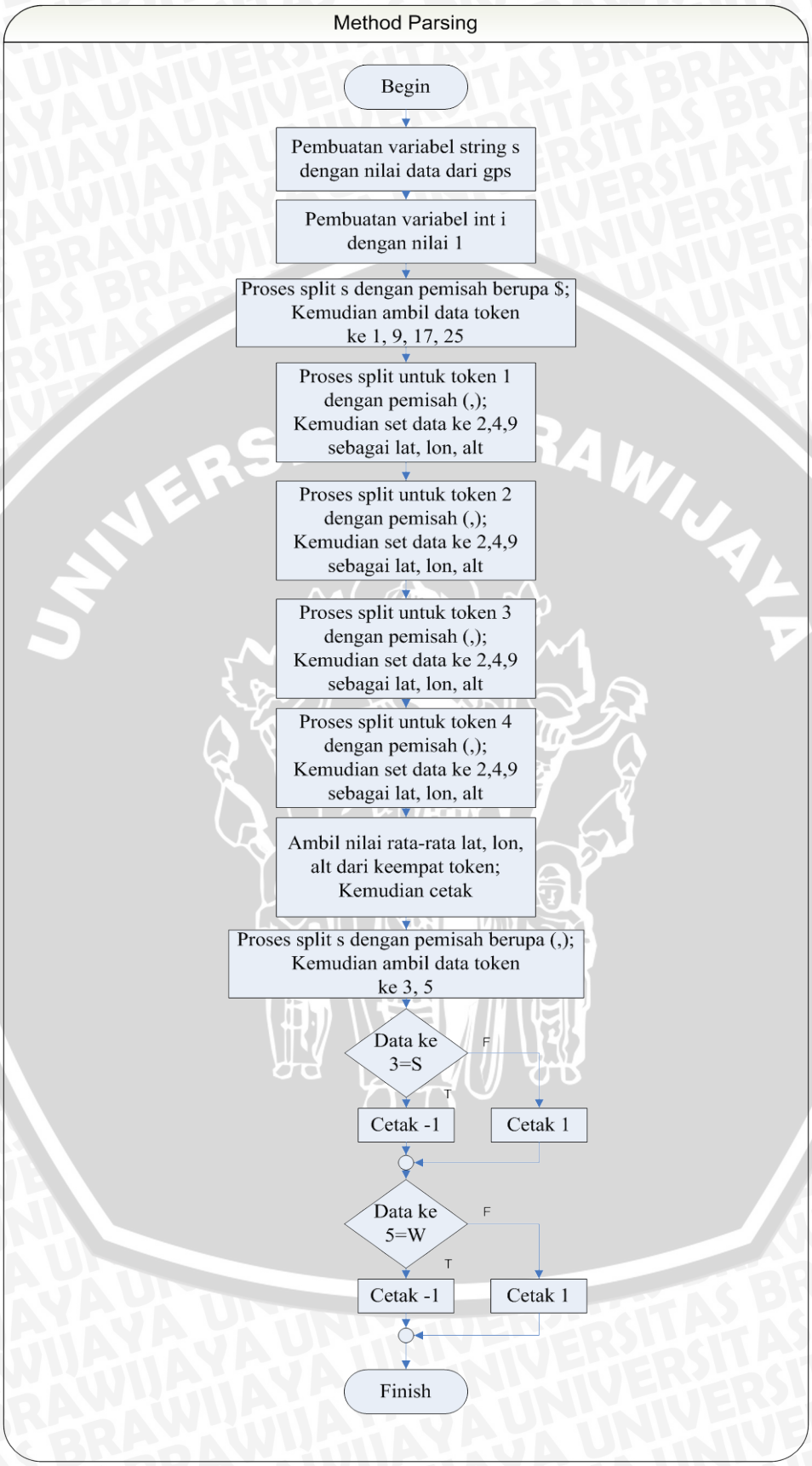
Untuk proses pembacaan data *streaming* dari *device* gps *bluetooth*nya digunakan algoritma sebagai berikut.



Algoritma 5.4 Proses Pembacaan Data *Streaming*

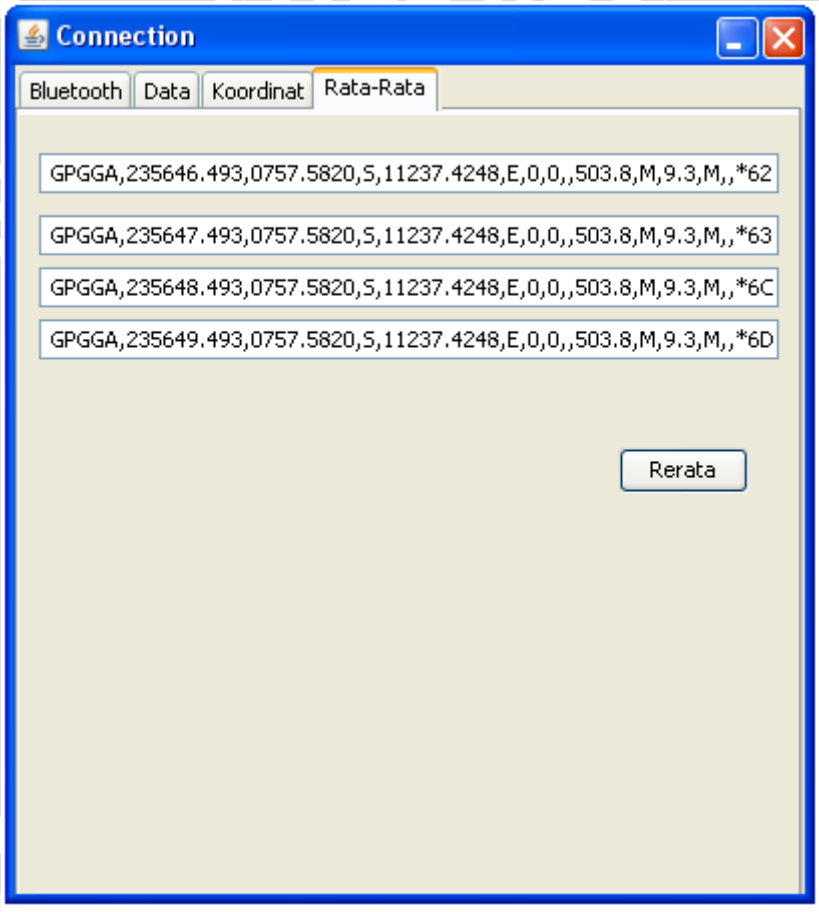
Algoritma *commreader* diatas aktif ketika *user* menekan tombol konek pada *form connect*, adapun urutan proses yang terjadi dari algoritma diatas adalah pendefinisian *method run* dimana didalamnya terdapat pendeskripsian stream*Connect*ion dengan alamat *bluetooth* url adalah *btConnectionURL* yang telah didefinisikan pada algoritma sebelumnya, kemudian diikuti dengan proses pembacaan data per*byte*nya dan mencetaknya pada *text area*, proses pembacaan berlanjut hingga koneksi dengan *device bluetooth* terputus.

Untuk proses pemarsingan data *streaming* tersebut maka dipergunakan algoritma sebagai berikut.



Algoritma 5.5 Proses *Parsing*

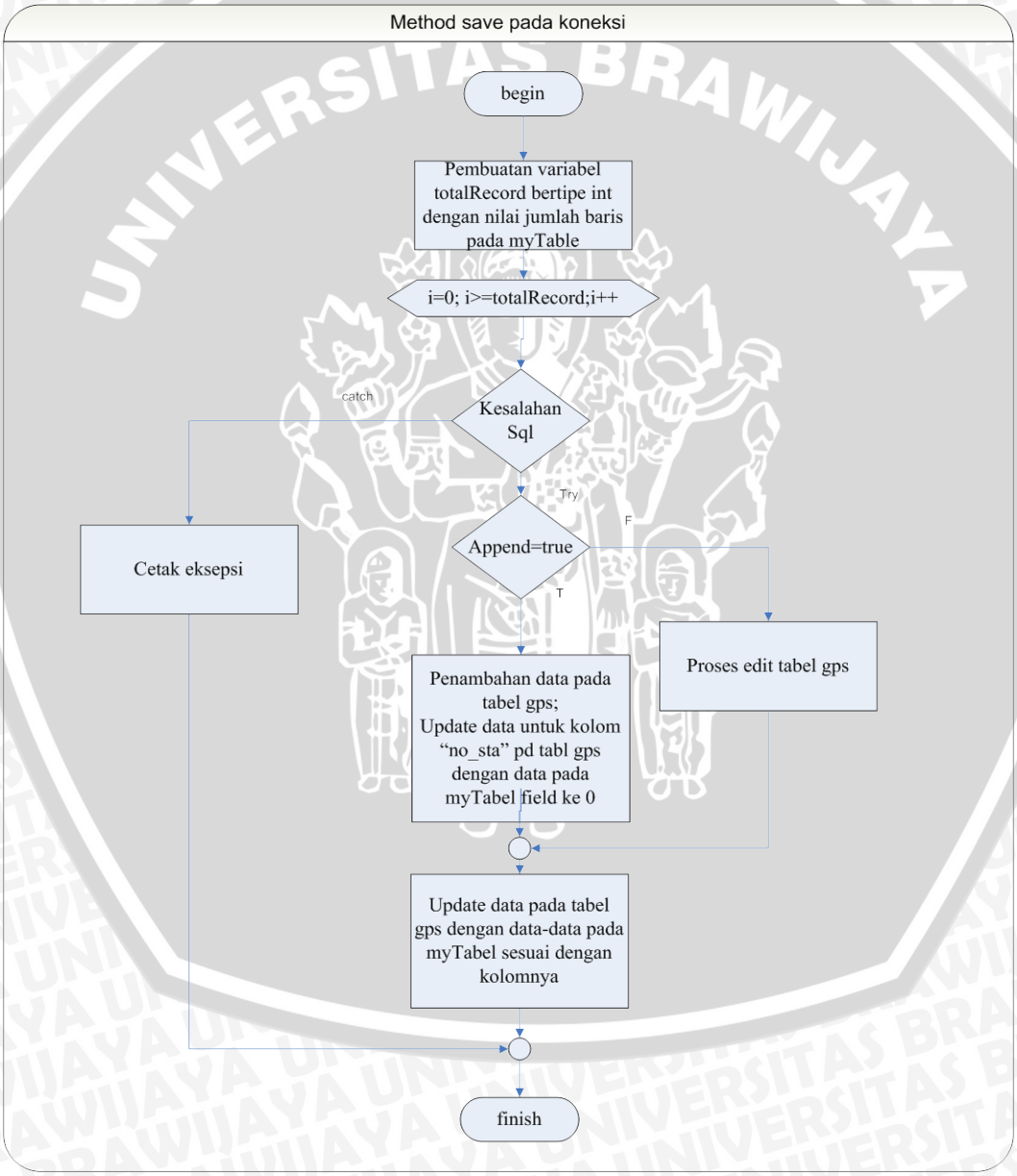
Algoritma *parsing* aktif ketika *user* menekan tombol *parsing* pada *form connect* pada *tabed pane* data, proses *parsing* berlangsung dua kali, yang pertama dengan pembacaan data inputan yang berupa *text* dengan *form*at NMEA pada *text area* hasil pembacaan data *GPS*, kemudian data-data yang terbaca dipilah-pilah dengan menggunakan *delimieter* berupa *dollar* (“$“), setelah dipilah dengan *dollar* data tersebut dikelompokkan mejadi 4 bagian untuk diambil data koordinatnya yang ditunjukkan dalam form berikut.



Dimana bagian-bagian yang diambil adalah data yang dimulai dengan “$GPGGA”. Setelah pengambilan data pertama selesai maka proses selanjutnya adalah data tersebut dipilah lagi dengan *delimiter* yang berupa koma (“,”), dari parsing kedua ini diambil data latitude, longitude dan altitude dari setiap keempat data hasil

parsing pertama, kemudian data dirata-rata. Dan terakhir adalah proses pencetakan data pada *form* koordinat, serta pembacaan data arah koordinat.

Tahapan terakhir dalam proses koneksi ini adalah tahapan penyimpanan data koordinat ke dalam *storage*. Adapun algoritma dari tahapan ini adalah sebagai berikut.

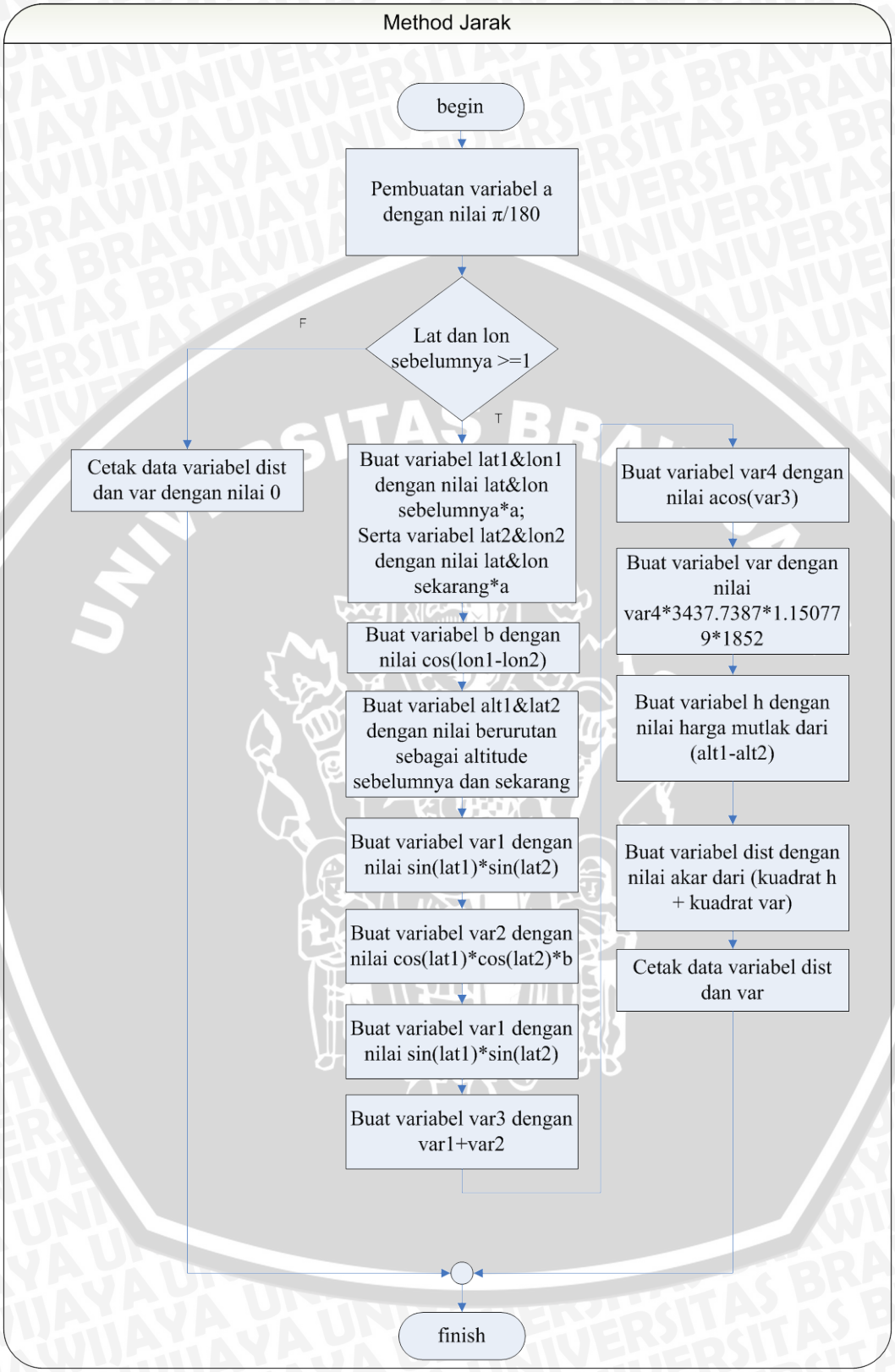


Algoritma 5.6 Proses *Save* pada Koneksi

Algoritma *save* aktif ketika *user* menekan tombol *save* pada *form connect* pada *tabbed pane* koordinat, proses *saving* diawali dengan pembacaan jumlah baris pada tabel yang telah disediakan, kemudian melakukan perulangan untuk membaca tiap-tiap *record* pada tabel dan menyimpannya kedalam tabel gps di *database*, dan untuk benar-benar mengakhiri proses penyimpanan digunakan fungsi *post()*.

5.3.3. Implementasi Algoritma Perhitungan

Algoritma perhitungan ini merupakan algoritma utama yang dipergunakan dalam menjalankan aplikasi ini. Algoritma perhitungan memperhitungkan semua variabel-variabel di dalam aplikasi untuk didapatkan satu atau beberapa variabel baru yang lebih berguna. Algoritma perhitungan di dalam sistem ini dibedakan berdasarkan perhitungan matematis yang dipergunakan untuk merencanakan pembangunan jalan. Yang diantaranya adalah perhitungan *stationing*, perhitungan *konversi* koordinat, perhitungan *alignment* vertikal dan horisontal dengan segala macam tipenya serta perhitungan volume pekerjaan. Berikut diberikan gambaran mengenai salah satu algoritma perhitungan dalam sistem ini, yakni algoritma perhitungan jarak yang terimplementasikan dalam klas *stationing*.java.



Algoritma 5.7 Proses Perhitungan Jarak pada *Stationing*

Algoritma perhitungan jarak akan aktif ketika tombol *Calc* pada *form stationing* ditekan oleh *user*. Penjelasan dari algoritma diatas adalah jika *previus latitude* dan *longitude* lebih besar dari nol maka set lat1 sebagai *previus latitude*, lat2 sebagai *latitude* sekarang, long1 sebagai previous *longitude*, long2 sebagai *longitude* sekarang, alt1 sebagai previous elevasi dan alt2 sebagai elevasi sekarang. Kemudian perhitungkan nilai variable 1 yang memiliki rumus

$$var1 = \sin lat1 \times \sin lat2$$

Dan perhitungkan nilai dari variabel 2 yang memiliki rumus

$$var\ 2 = (\cos lat1 \times \cos lat2 \times \cos(long1 - long2))$$

Kemudian var1 dan var2 dikombinasikan kedalam perhitungan var3 yang memiliki rumus

$$var3 = (\text{acos}(var1 + var2) \times 3437.7387 \times 1.150779 \times 1852)$$

Dari ketiga perhitungan diatas didapatkan nilai jarak horisontal dari dua titik koordinat yang bernilai var3. Untuk mencari perbedaan ketinggian dari kedua titik tersebut maka dipergunakan variabel h yang memiliki rumus.

$$h = \sqrt[2]{(alt1 - alt2)^2}$$

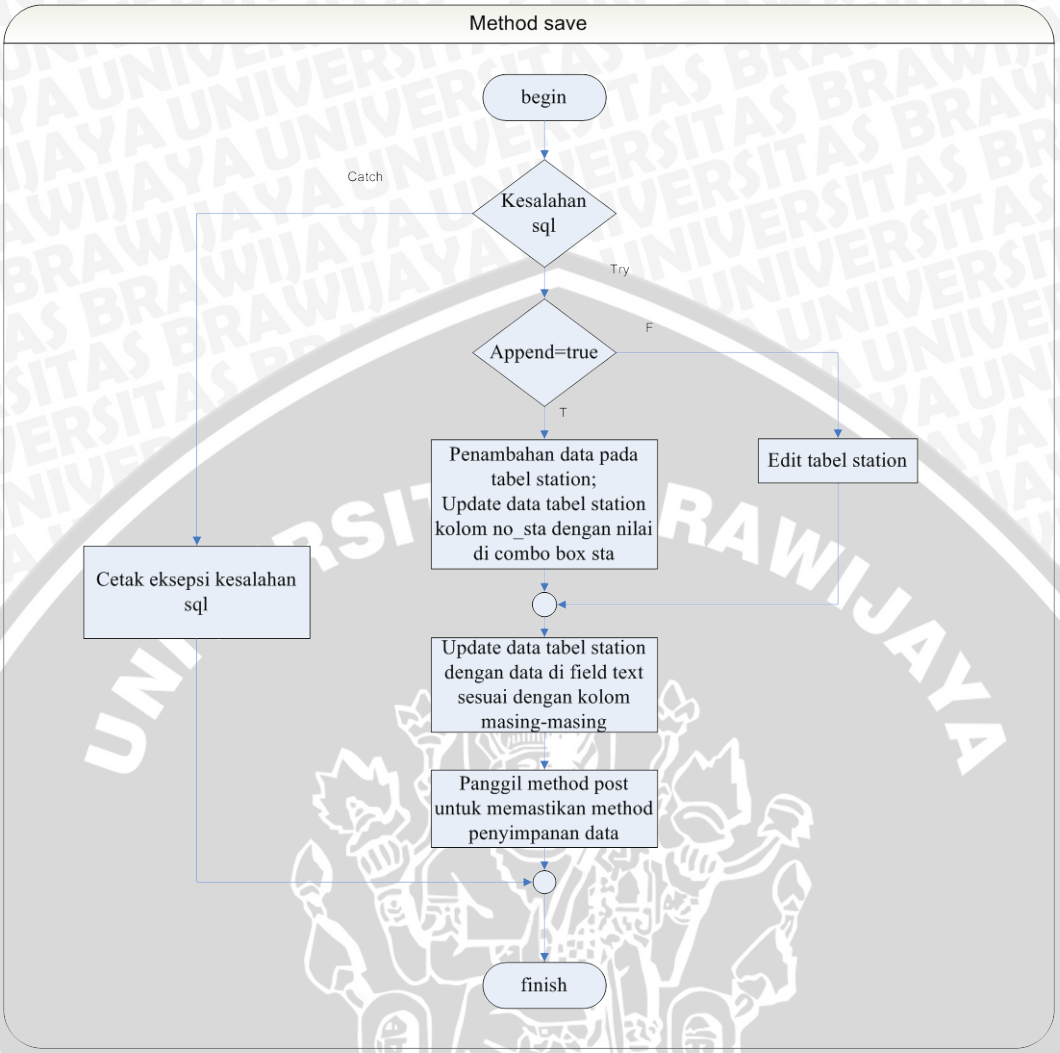
kemudian dari variabel h dan var3 maka dapat diperhitungkan jarak sebenarnya dari dua titik koordinat tersebut dengan mempergunakan rumus

$$dist = \sqrt[2]{(var3^2 + h^2)}$$

Sehingga jarak interval sebenarnya dari dua titik koordinat tersebut sebesar dist.

5.3.4. Implementasi Algoritma *Save*

Algoritma *save* dipergunakan sebagai proses penyimpanan variabel-variabel hasil perhitungan ke dalam sebuah *storage* berupa *database* untuk nantinya dipergunakan dalam perhitungan pada tingkatan selanjutnya. Alur proses dari algoritma *save* ini disesuaikan dengan tabel tujuan penyimpanan data pada *database*. Berikut adalah salah satu bentuk algoritma *save* pada proses penyimpanan data *stationing*.



Algoritma 5.8 Proses Penyimpanan pada *Stationing*

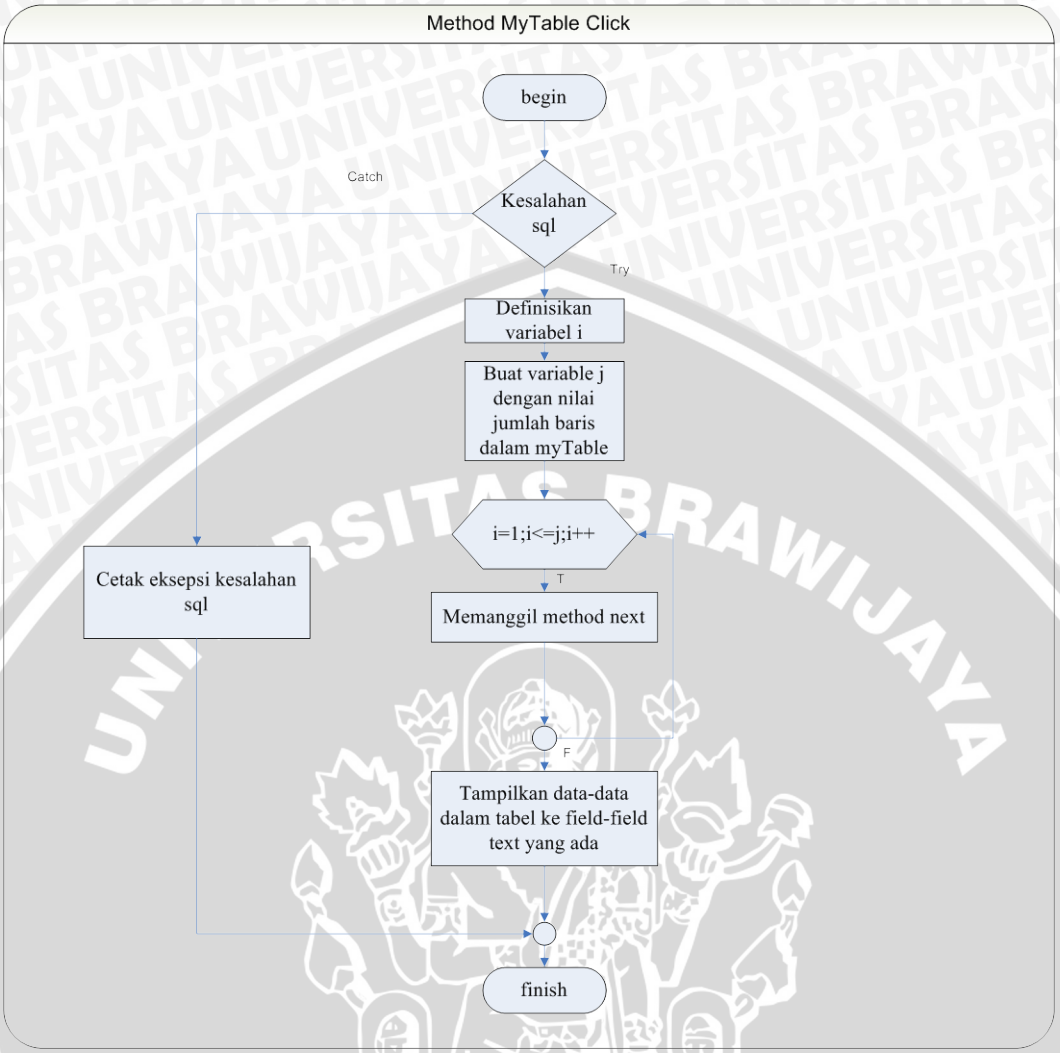
Algoritma *save* diatas aktif ketika *user* melakukan penekanan terhadap tombol *save* pada *form stationing*. Alur proses dari algoritma ini adalah jika *append* bernilai true, maka tabel *stationing* pada *database* akan diset dengan *append* (penambahan data) kemudian kolom no_sta di*update* dengan nilai yang tertera pada *combo box* sta, jika bukan *append* maka lakukan proses *edit*. Selanjutnya lakukan proses *update* untuk kolom jarak dengan nilai yang tertera pada *distTxt*, kolom jarak1 dengan nilai *dist1Txt*, kolom tipe dengan nilai yang tertera pada *combo box type*, dan kolom kd_sta dengan nilai yang tertera pada kdTxt, untuk mengakhiri proses *saving* pada

tabel *stationing database* hitungjalan maka dipanggil *method post* dari *package* model.

5.3.5. Implementasi Algoritma *Delete*

Algoritma *delete* dipergunakan untuk menghapus data pada tabel *form* yang sedang aktif. Proses penghapusan data pada sistem ini dipisahkan kedalam 2 tahapan yang berbeda, yakni tahapan pemilihan data pada yang akan dihapus pada tampilan tabel di *form* dan tahapan penghapusan data dari tabel pada *database* hitungjalan.

Adapun algoritma pemilihan data yang akan dihapus ditunjukkan sebagai berikut.

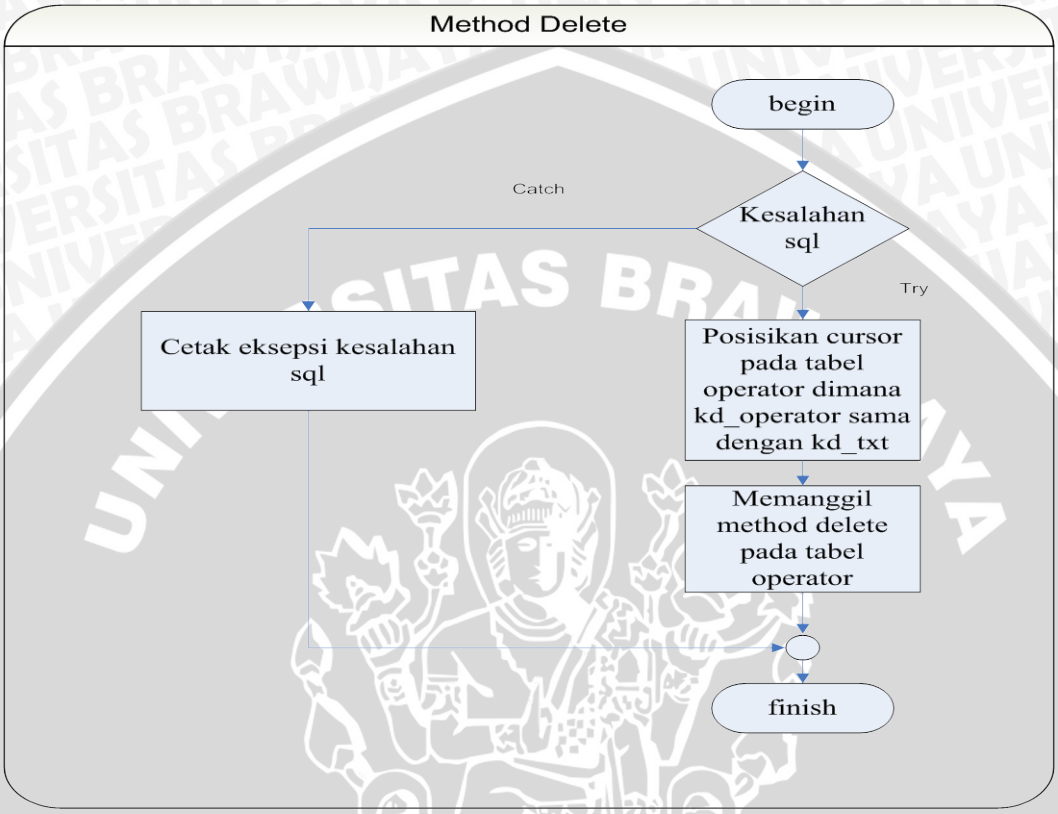


Algoritma 5.9 Proses Pengklikan *Mouse* pada *MyTable*

Algoritma diatas akan aktif ketika *myTable* pada *form* yang sedang aktif mengalami penekanan *mouse*. Algoritma tersebut dipergunakan untuk mengambil data dari tabel *operator* untuk ditampilkan pada *field-field* dari *form operator*. Alur tahapan proses yang terjadi adalah pendefinisian *integer* i dan j, pengesetan posisi *record* tabel *operator* pada permulaan posisi, perulangan mulai dari 1 hingga baris yang terseleksi pada *myTabel*, untuk setiap kali perulangan maka posisi *record* akan maju satu langkah hingga ditemukan data tabel *operator* yang telah diseleksi *user* pada

myTable, terakhir adalah menampilkan setiap kolom pada tabel *operator* kedalam *field-field* yang telah disediakan pada *form operator*.

Untuk algoritma penghapusan data dapat ditunjukkan sebagai berikut.



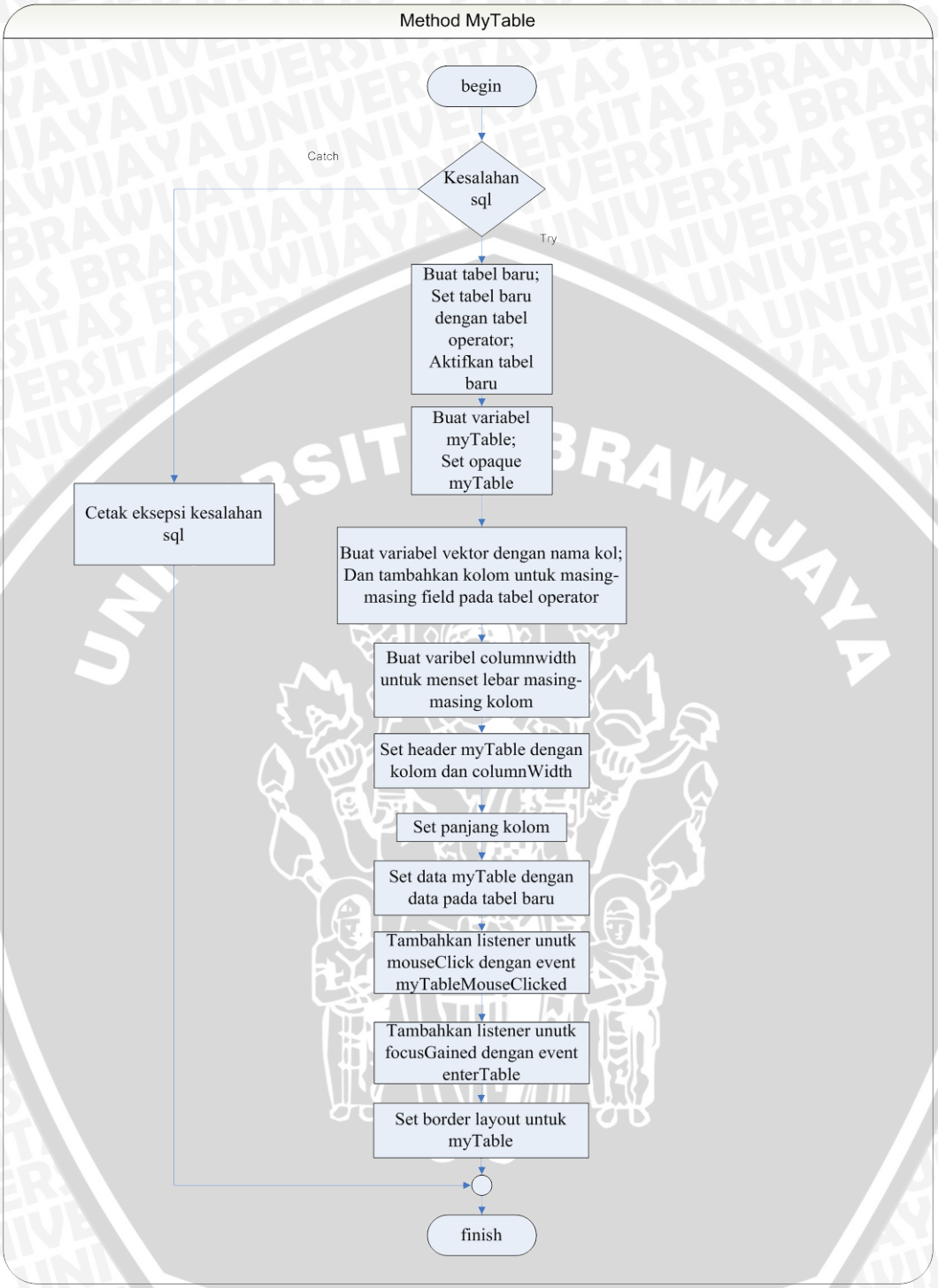
Algoritma 5.10 Proses Penghapusan Data yang terseleksi

Algoritma diatas aktif ketika *user* menekan tombol *delete* pada *form* yang sedang aktif. Alur dari proses penghapusan tersebut adalah pencarian data pada tabel *operator* kolom kode *operator* yang sesuai dengan *field* kodeTxt, setelah ditemukan maka memanggil *method delete*() dari *package* model. Kemudian datapun sudah terhapus baik pada *myTabel form operator* ataupun pada tabel *operator*nya.

5.3.6. Implementasi Algoritma *myTabel*

Algoritma *myTabel* digunakan untuk membaca data yang telah tersimpan pada *database* hitungjalan kemudian menampilkannya pada *form* yang sedang aktif, algoritma ini aktif ketika *form* mengalami proses inisialisasi, algoritma ini terdapat dalam setiap klas yang dibuat didalam sistem kecuali klas *loginForm*.

Adapun bentuk algoritma *myTabel* tersebut adalah sebagai berikut.



Algoritma 5.11 Proses Penampilan *MyTable*

Urutan algoritma diatas adalah pendefinisian t*Operator* dimana datanya diambil dari tabel *operator*, kemudian pendefinisian *myTable* dengan kolom Kd *Operator*,

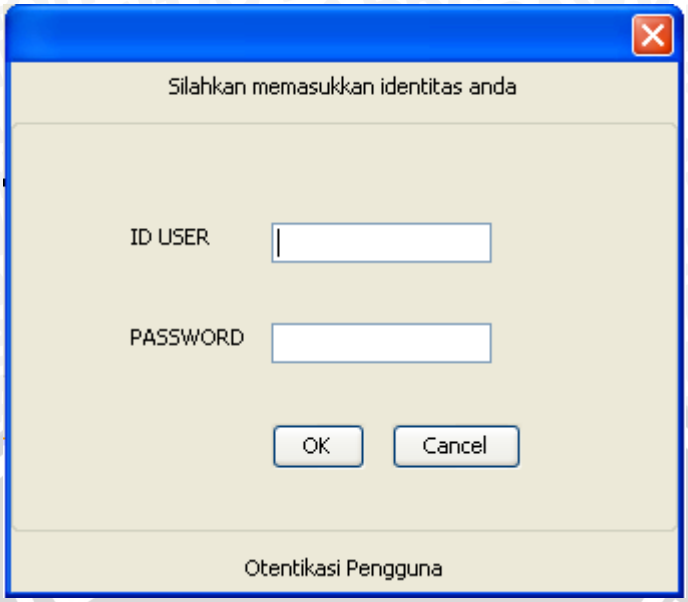
Nama *Operator*, Alamat, *User* Id dan *Password* dengan lebar dan ketinggian yang sedemikian rupa, kemudian data dalam *myTable* diset sebagai data dari t*Operator*, setelah pengesetan maka dilakukan pemberian event ketika *myTable* mengalami pengklikkan *mouse* dan pemfokusan, terakhir *myTable* diletakkan pada *panel* pGrid dengan *borderLayout center*.

5.4. Implementasi Antarmuka Aplikasi

Dalam aplikasi ini, antarmuka dibuat dengan kebutuhan fungsionalitas yang harus disediakan oleh sistem. Tujuan utama pengembangan antarmuka adalah untuk memberikan kemudahan bagi pengguna (*user*) dalam menggunakan aplikasi yang telah dibangun.

5.4.1. Implementasi Antarmuka *Login*

Antarmuka *login* ini akan ditampilkan ketika penggunan (*user*) menggunakan aplikasi. *Form* ini menampilkan *field-field user* id dan *password* yang disertai dengan berbagai label penunjuk dan tombol-tombol. Antarmuka *login* ditunjukkan pada gambar berikut.



Gambar 5.1 Tampilan *Login*

Penjelasan masing-masing tombol dari antarmuka diuraikan pada tabel berikut.

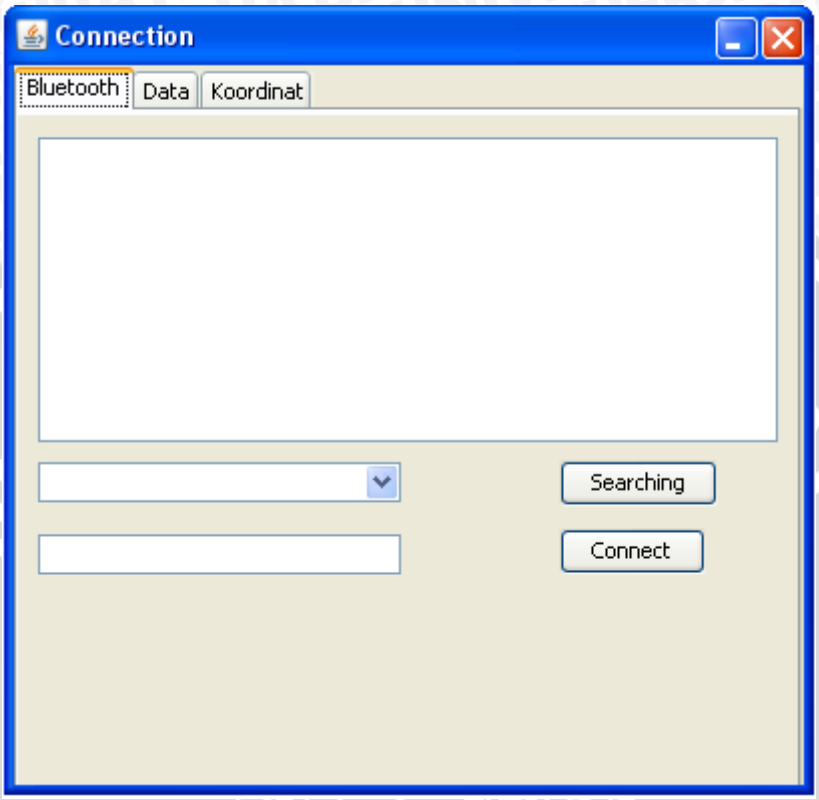
No	Tombol	Keterangan
1	OK	Tombol ini digunakan untuk meng*authentikasi* data inputan serta masuk ke sistem dengan dukungan menu sepenuhnya.
2	*Cancel*	Tombol ini digunakan untuk membatalkan proses *authentikasi* dan langsung masuk ke sistem dengan menu yang minim.

Tabel 5.4 Penjelasan Tombol *Login*

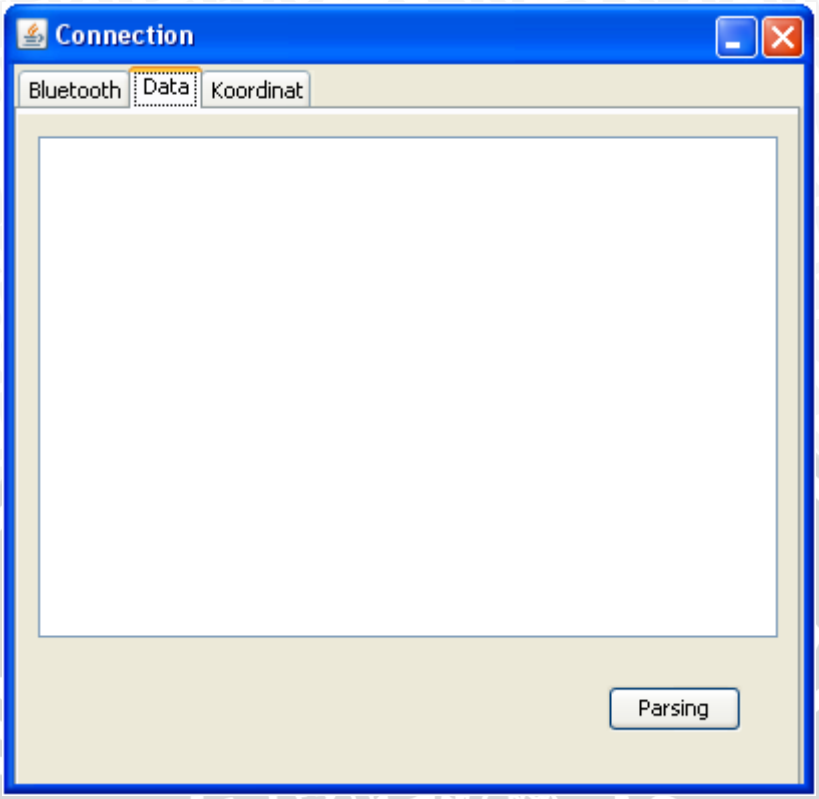
5.4.2. Implementasi Antarmuka Koneksi

Antarmuka ini dipergunakan untuk mengantarmuka proses koneksi sistem dengan *device-device* eksternal *bluetooth* kepada *user*. Pada *form* ini antarmuka

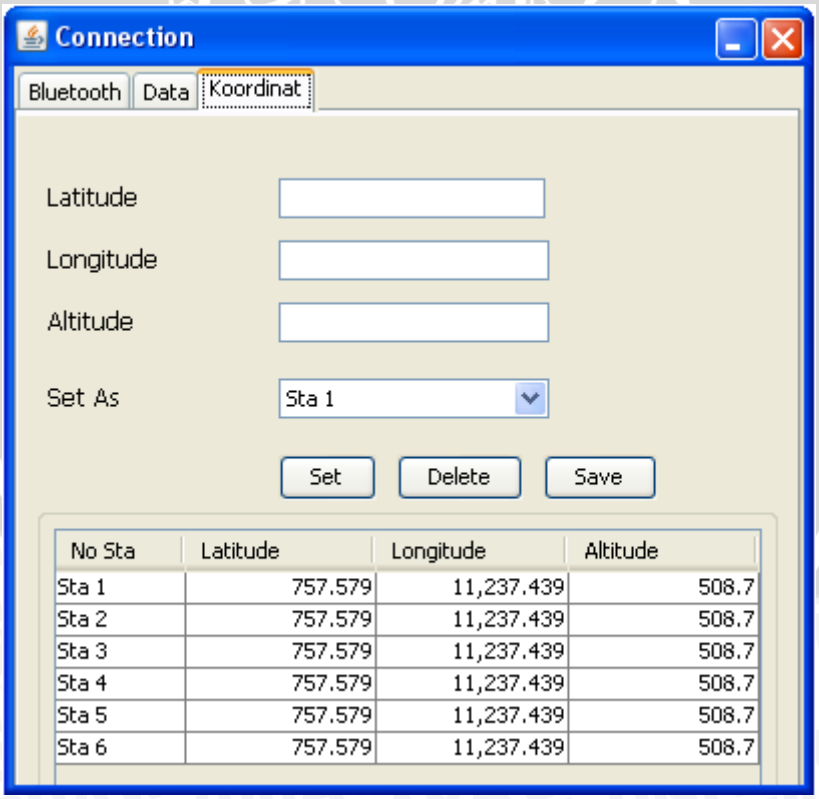
berbentuk *tabbed pane* yang dibagi menjadi tiga *panel*, yakni *panel bluetooth*, data dan koordinat. Gambar masing-masing *panel* ditunjukkan sebagai berikut.



Gambar 5.2 Tampilan *Form Connect Panel Bluetooth*



Gambar 5.3 Tampilan *Form Connect Panel* Data



Gambar 5.4 Tampilan *Form Connect Panel* Koordinat

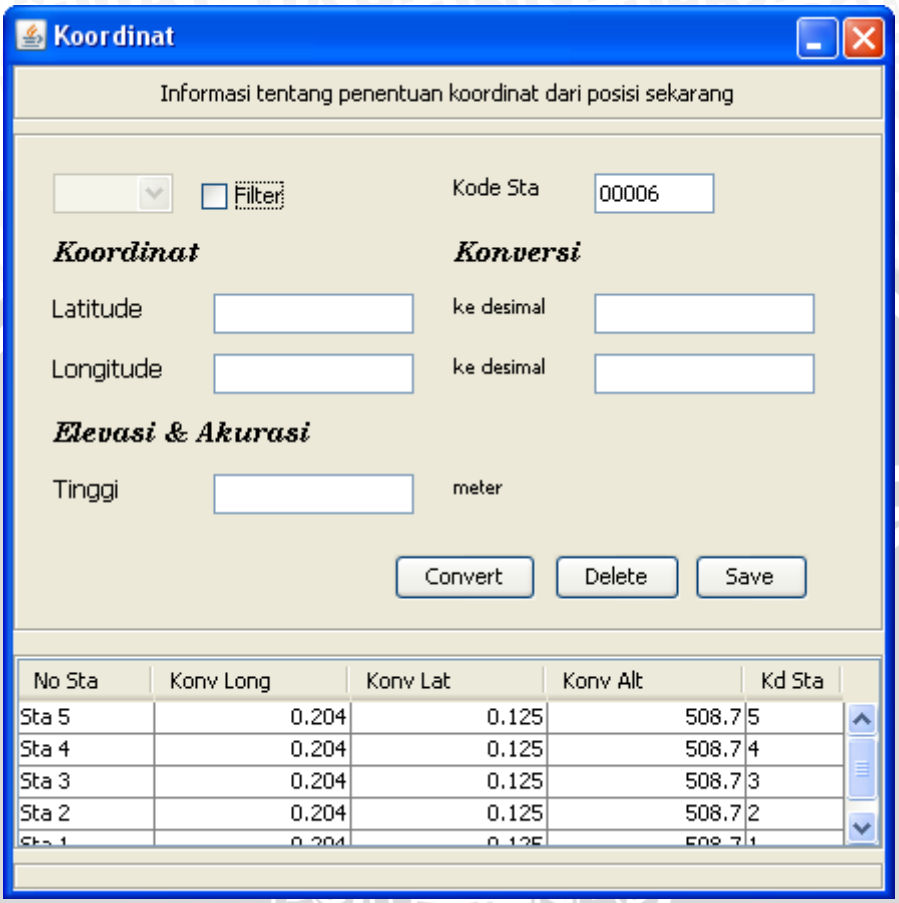
Penjelasan masing-masing tombol dari antarmuka koneksi dijelaskan pada tabel berikut.

No	Tombol	Keterangan
1	Searching	Digunakan untuk melakukan proses pencarian *device-device* yang memiliki *bluetooth* dan menampilkannya ke dalam *text area panel bluetooth*
2	*Connect*	Digunakan untuk melakukan koneksi dengan *device* yang diseleksi *user* melalui *combo box* pada *panel bluetooth*
3	*Parsing*	Digunakan untuk melakukan proses *parsing* data pada *text area*, dengan ketentuan bahwa data tersebut ber*form*at NMEA
4	Set	Digunakan untuk mengeset data koordinat pada *field-field panel* koordinat ke dalam *myTabel*
5	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
6	*Save*	Digunakan untuk menyimpan semua data pada *myTabel* ke dalam *database*

Tabel 5.5 Penjelasan Tombol Koneksi

5.4.3. Implementasi Antarmuka Koordinat

Antarmuka koordinat merupakan *form* yang digunakan untuk meng*konversi* data-data gps yang ber*form*at *degree-minute-second* menjadi *decimal-degree*. Gambar dari antarmuka tersbut ditunjukkan sebagai berikut.



Gambar 5.5 Tampilan *Form* Koordinat

Penjelasan dari masing-masing tombol disajikan dalam tabael berikut.

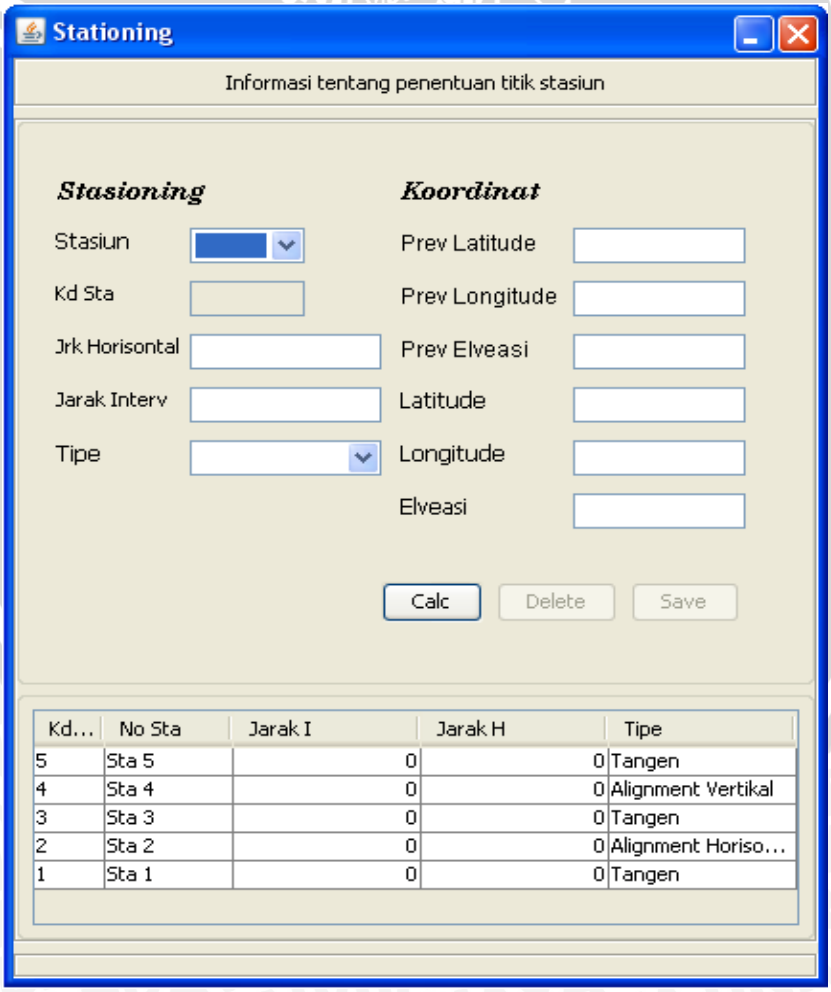
No	Tombol	Keterangan
1	Convert	Digunakan untuk meng*konversi* data *degree-minute-second* menjadi data *decimal degree*
2	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*

3	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.6 Penjelasan Tombol Koordinat

5.4.4. Implementasi Antarmuka *Stationing*

Antarmuka *stationing* merupakan *form* yang digunakan untuk memperhitungkan data jarak titik koordinat yang diseleksi *user* dengan titik koordinat sebelumnya. Gambar tampilan dari antarmuka *stationing* ditunjukkan sebagai berikut.



Gambar 5.6 Tampilan *Form Stationing*

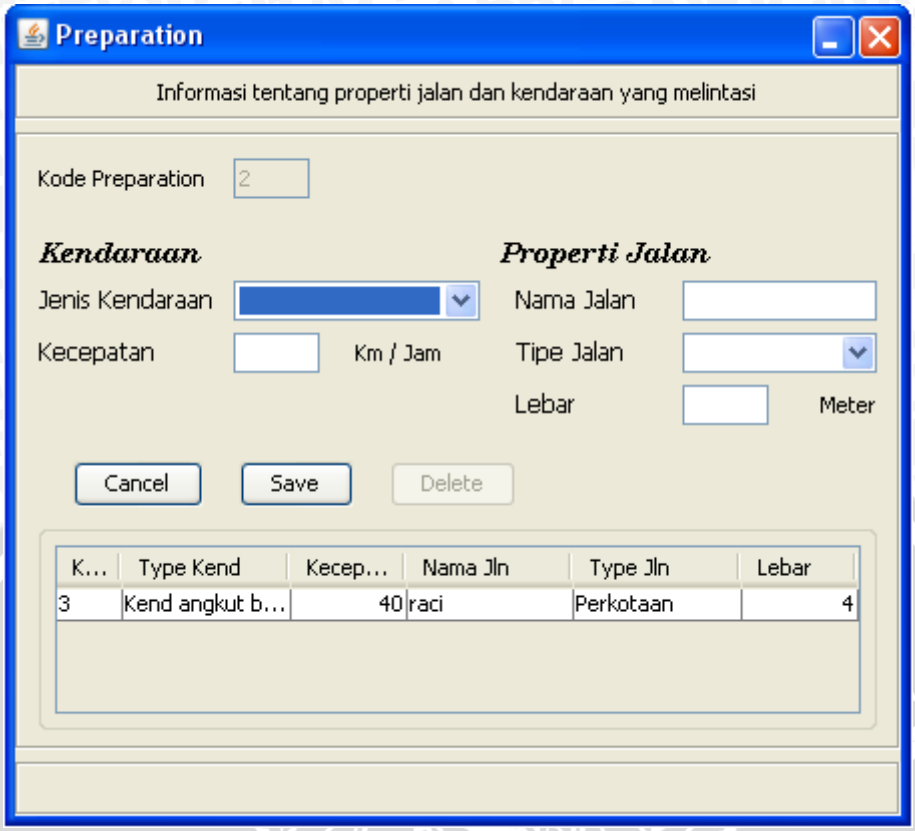
Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

No	Tombol	Keterangan
1	*Calc*	Digunakan untuk menghitung jarak dan interval dari suatu titik koordinat dengan titik koordinat sebelumnya
2	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
3	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.7 Penjelasan Tombol *Stationing*

5.4.5. Implementasi Antarmuka *Preparation*

Antarmuka *preparation* merupakan *form* preparasi yang digunakan untuk mengolah data properti jalan dan jenis kendaraan rencana. Gambar tampilan dari antarmuka *preparation* ditunjukkan seperti berikut.



Gambar 5.7 Tampilan *Form Preparation*

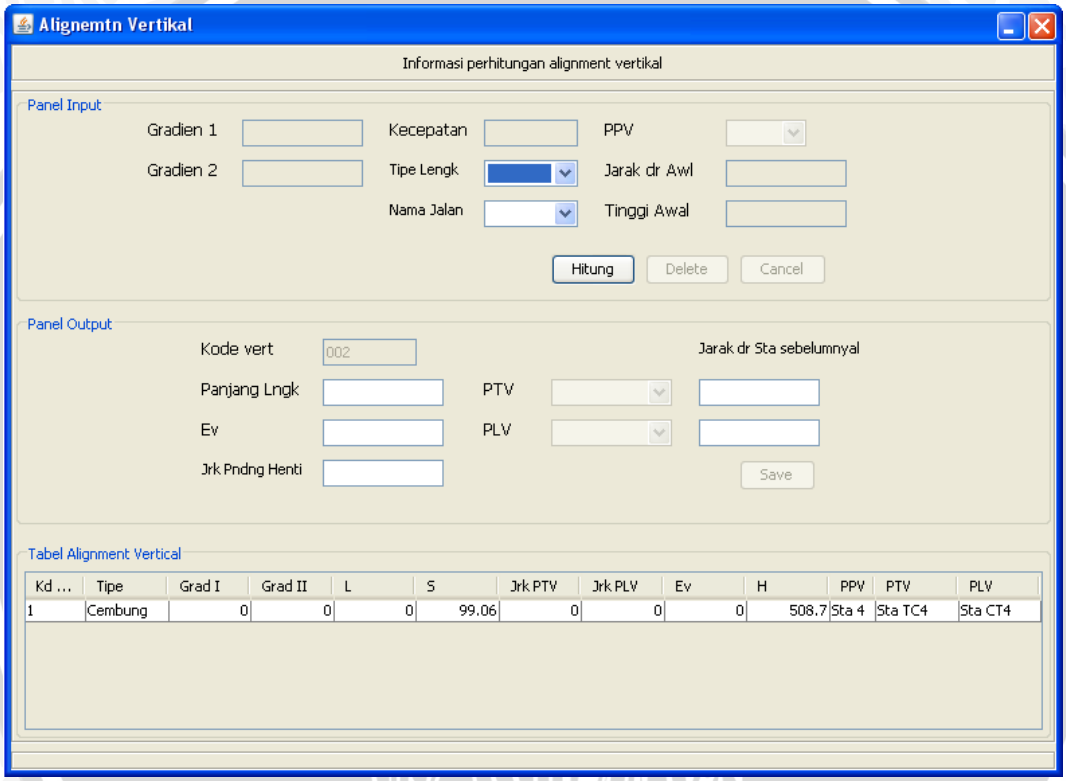
Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

No	Tombol	Keterangan
1	*Cancel*	Digunakan untuk membatalkan proses dengan mengosongkan *field-field*
2	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
3	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.8 Penjelasan Tombol *Stationing*

5.4.6. Implementasi Antarmuka *Alignment* Vertikal

Antarmuka *alignment* vertikal merupakan *form* vertikal yang dipergunakan untuk melakukan perhitungan *alignment* vertikal. Gambar dari tamplan antarmuka ini adalah.



Gambar 5.8 Tampilan *Form* Vertikal

Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

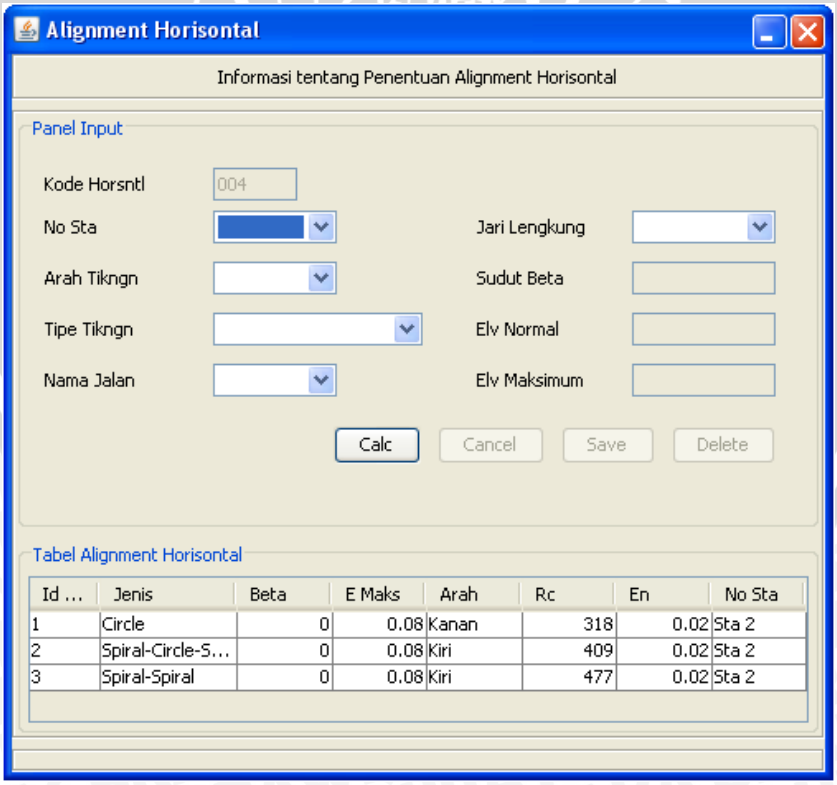
No	Tombol	Keterangan
1	Hitung	Digunakan untuk memperhitungkan *alignment* vertikal dengan inputan grad1, grad2, kecepatan,

		jarak, tinggi dan PPV
2	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
3	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.9 Penjelasan Tombol Vertikal

5.4.7. Implementasi Antarmuka *Alignment* Horisontal

Antarmuka horisontal merupakan *form* horisontal yang dipergunakan untuk perhitungan *alignment* horisontal, nantinya data hasil perhitungan dari antarmuka ini akan dipergunakan lagi dalam perhitungan horisontal dengan tipe yang sesuai setting dari *user*. Gambar tampilan dari antarmuka ini adalah.



Gambar 5.9 Tampilan *Form* Horisontal

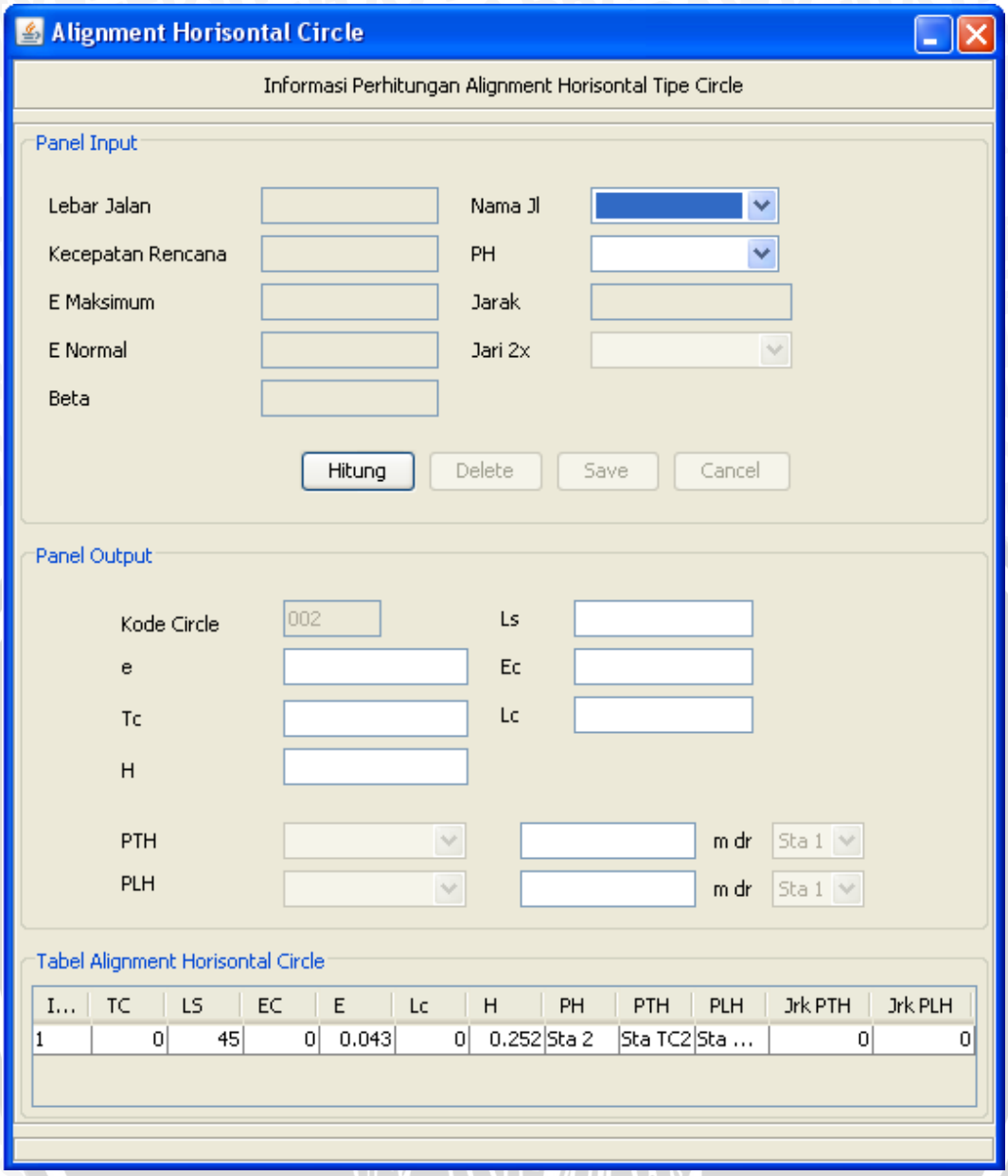
Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

No	Tombol	Keterangan
1	Hitung	Digunakan untuk memperhitungkan *alignment* horisontal dengan inputan jari lengkung dan no_stationig
2	*Cancel*	Digunakan untuk membatalkan proses dengan mengosongkan tiap-tiap *field*
3	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
4	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.10 Penjelasan Tombol Horisontal

5.4.8. Implementasi Antarmuka *Alignment* Horisontal tipe *Circle*

Antarmuka horisontal tipe *circle* merupakan *form* horisontal yang bertipe *circle* yang dipergunakan untuk memperhitungkan *alignment* horisontal tipe *circle*. Gambar tampilan dari antarmuak ini adalah sebagai berikut.



Gambar 5.10 Tampilan *Form* Horisontal tipe *Circle*

Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

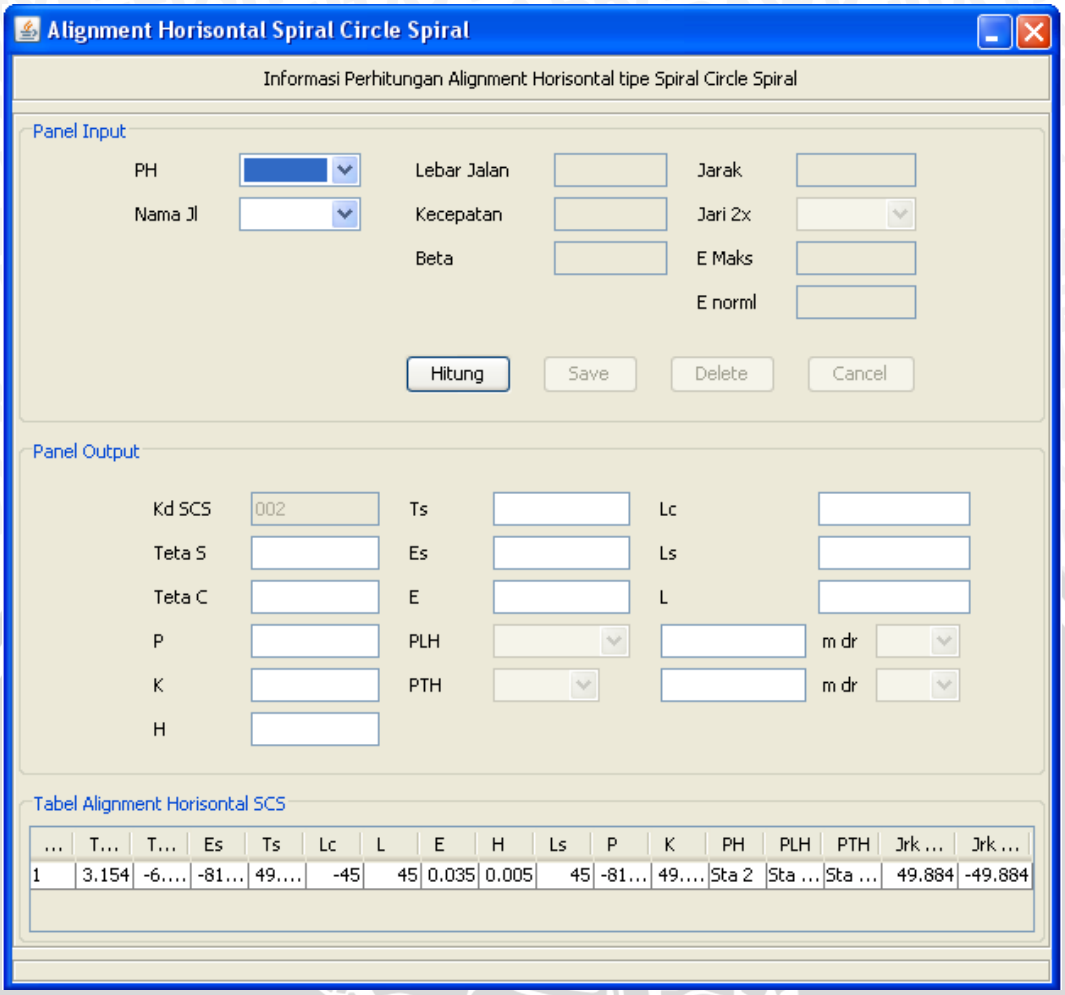
No	Tombol	Keterangan
1	Hitung	Digunakan untuk memperhitungkan *alignment* horisontal tipe *circle*

2	*Cancel*	Digunakan untuk membatalkan proses dengan mengosongkan tiap-tiap *field*
3	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
4	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.11 Penjelasan Tombol Horisontal tipe *Circle*

5.4.9. Implementasi Antarmuka *Alignment* Horisontal tipe Spiral-*Circle*-Spiral

Antarmuka horisontal tipe spiral-*circle*-spiral merupakan *form* horisontal scs yang dipergunakan untuk memperhitungkan *alignment* horisontal tipe spiral-*circle*-spiral. Gambar tampilan dari antarmuak ini adalah sebagai berikut.



Gambar 5.11 Tampilan *Form* Horisontal tipe Spiral-*Circle*-Spiral

Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

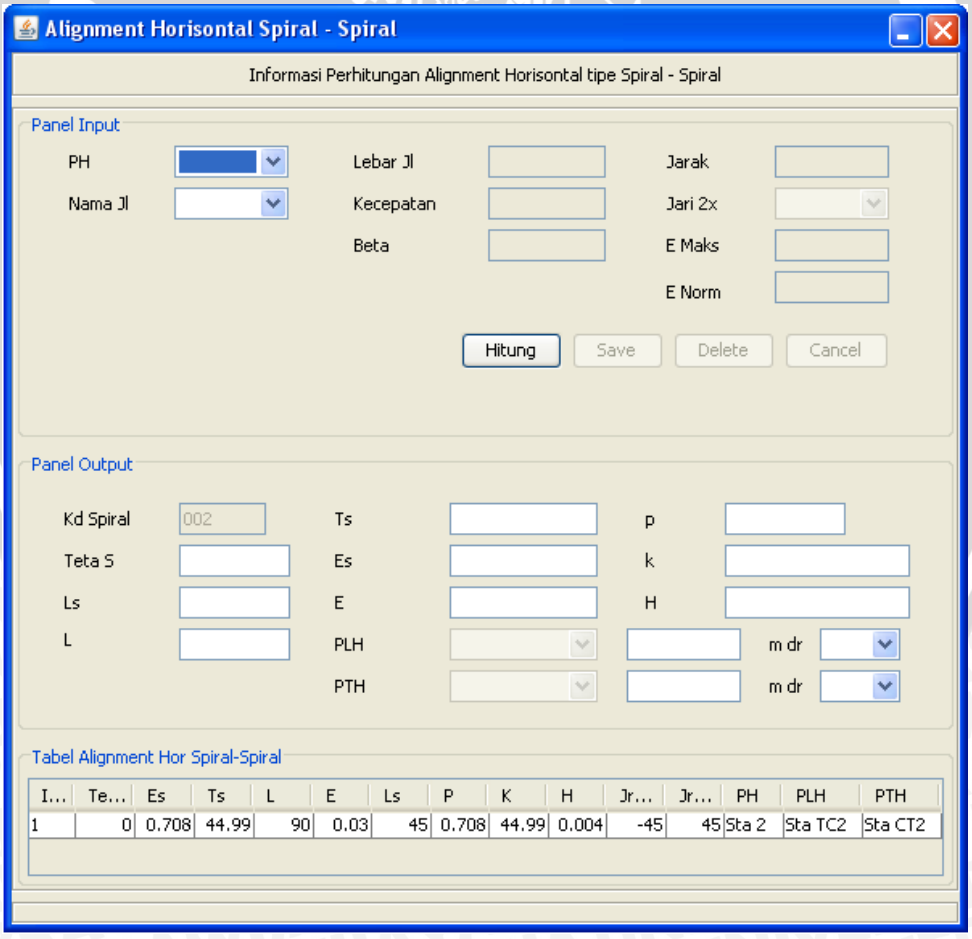
No	Tombol	Keterangan
1	Hitung	Digunakan untuk memperhitungkan *alignment* horisontal tipe spiral-*circle*-spiral
2	*Cancel*	Digunakan untuk membatalkan proses dengan mengosongkan tiap-tiap *field*
3	*Delete*	Digunakan untuk menghapus data baik pada

		myTable dan pada *database*
4	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.12 Penjelasan Tombol Horisontal tipe Spiral-*Circle*-Spiral

5.4.10. Implementasi Antarmuka *Alignment* Horisontal tipe Spiral-Spiral

Antarmuka horisontal tipe spiral- spiral merupakan *form* horisontal ss yang dipergunakan untuk memperhitungkan *alignment* horisontal tipe spiral- spiral. Gambar tampilan dari antarmuak ini adalah sebagai berikut.



Gambar 5.12 Tampilan *Form* Horisontal tipe Spiral- Spiral

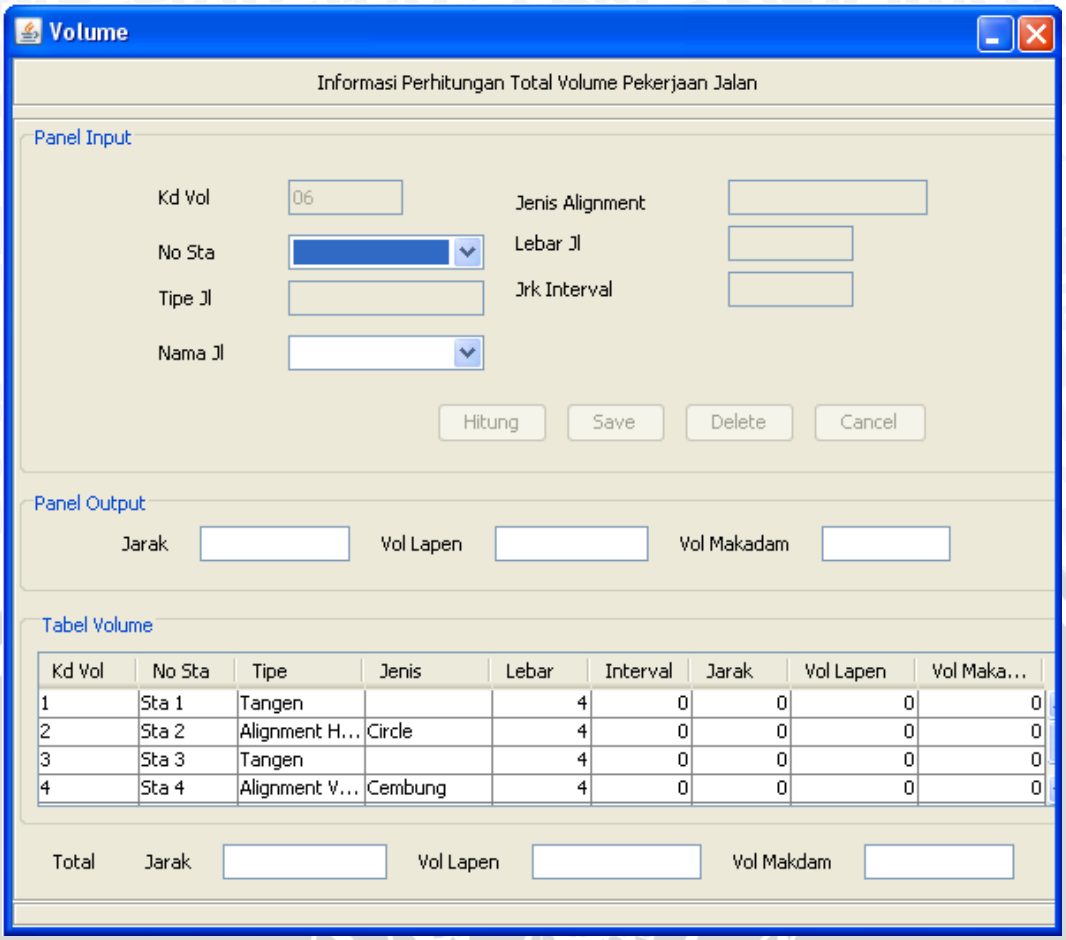
Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

No	Tombol	Keterangan
1	Hitung	Digunakan untuk memperhitungkan *alignment* horisontal tipe spiral-spiral
2	*Cancel*	Digunakan untuk membatalkan proses dengan mengosongkan tiap-tiap *field*
3	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
4	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.13 Penjelasan Tombol Horisontal tipe Spiral-Spiral

5.4.11. Implementasi Antarmuka Volume

Antarmuka volume merupakan *form* yang paling bermanfaat bagi *user*, karena dari antarmuka ini dapat diketahui tujuan dibuatnya sistem aplikasi perhitungan pemabangunan jalan ini. Adapun tampilan dari antarmuka ini adalah.



Gambar 5.13 Tampilan *Form* Volume Kerja

Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

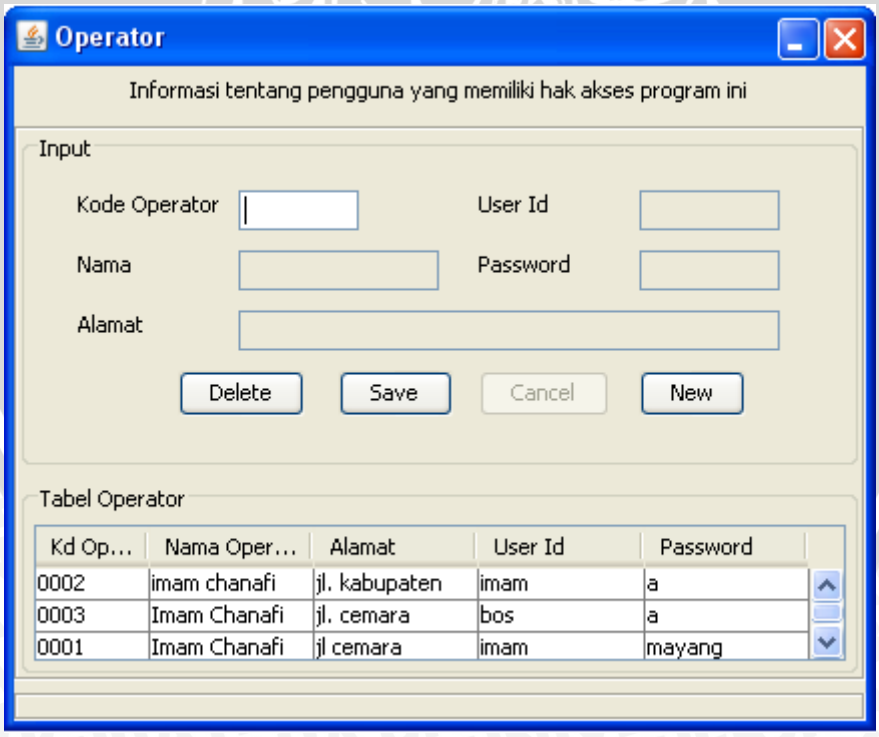
No	Tombol	Keterangan
1	Hitung	Digunakan untuk memperhitungkan volume total pekerjaan lapen, makadam dan panjang jalan berdasarkan kepada perhitungan-perhitungan sebelumnya.
2	*Cancel*	Digunakan untuk membatalkan proses dengan mengosongkan tiap-tiap *field*

3	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
4	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.14 Penjelasan Tombol Volume

5.4.12. Implementasi Antarmuka *Operator*

Antarmuka *operator* digunakan untuk melakukan pengolahan data pengguna-pengguna yang berhak mengakses program. Tampilan dari antarmuka ini adalah sebagai berikut.



Gambar 5.14 Tampilan *Form Operator*

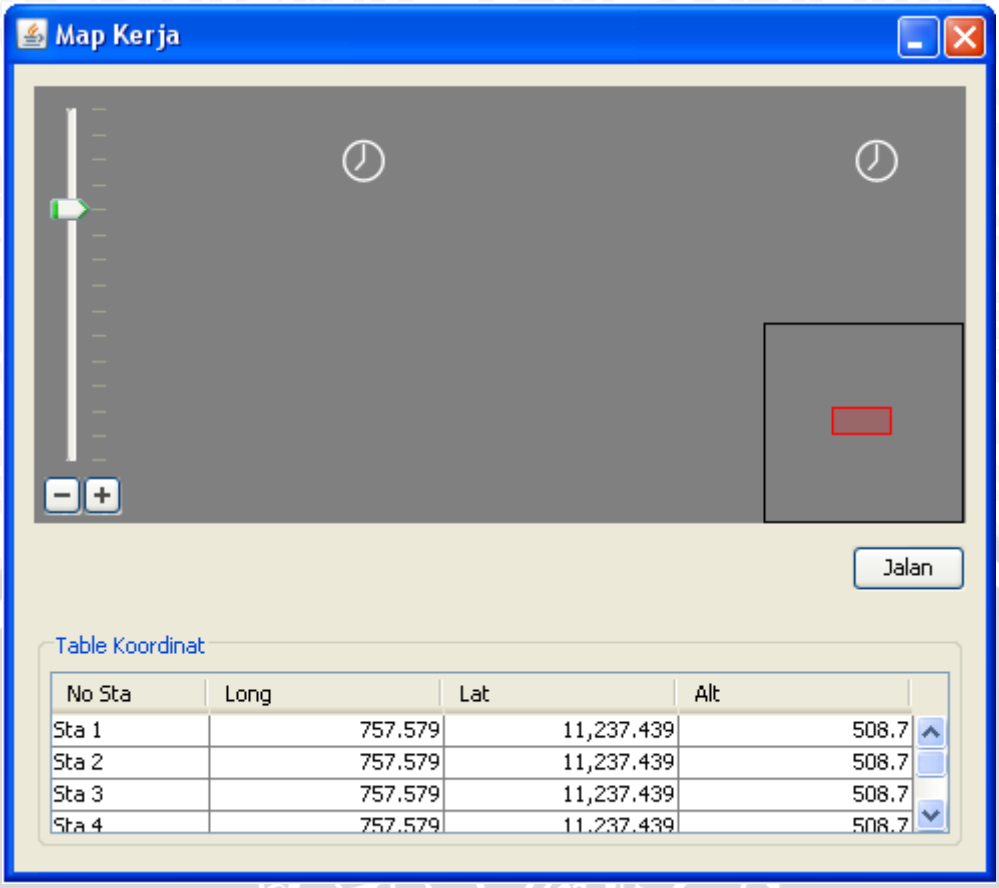
Penjelasan dari masing-masing tombol ditunjukkan pada tabel berikut.

No	Tombol	Keterangan
1	New	Digunakan untuk mengosongkan tiap-tiap *field* dan mengeset boolean *append* menjadi true
2	*Cancel*	Digunakan untuk membatalkan proses dengan mengosongkan tiap-tiap *field*
3	*Delete*	Digunakan untuk menghapus data baik pada *myTable* dan pada *database*
4	*Save*	Digunakan untuk menyimpan semua data pada *field-field* ke dalam *database* kemudian menampilkannya ke dalam *myTabel*

Tabel 5.15 Penjelasan Tombol *Operator*

5.4.13. Implementasi Antarmuka Map Kerja

Antarmuka ini dipergunakan untuk menampilkan bentuk jalan rencana dari atas permukaan bumi menggunakan sistem mapping (pemetaan). Tampilan dari antarmuka ini adalah sebagai berikut.

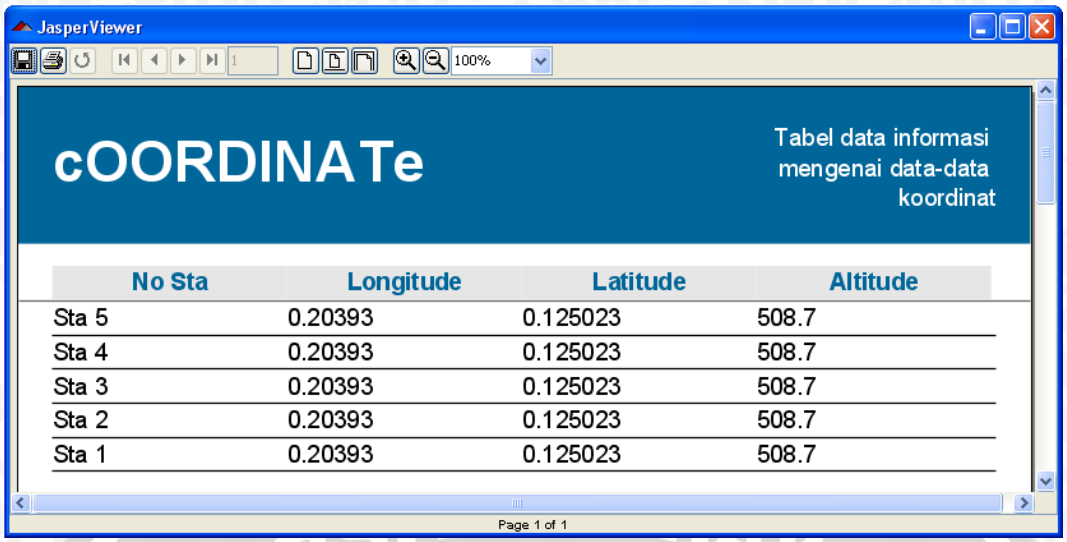


Gambar 5.15 Tampilan *Form* Map Kerja

Tombol jalan pada antarmuka ini dipergunakan untuk menampilkan titik-titik koordinat pengambilan data dari gps dan membentuk garis lurus yang menghubungkan antar titik-titik tersebut.

5.4.14. Implementasi Antarmuka Laporan Koordinat

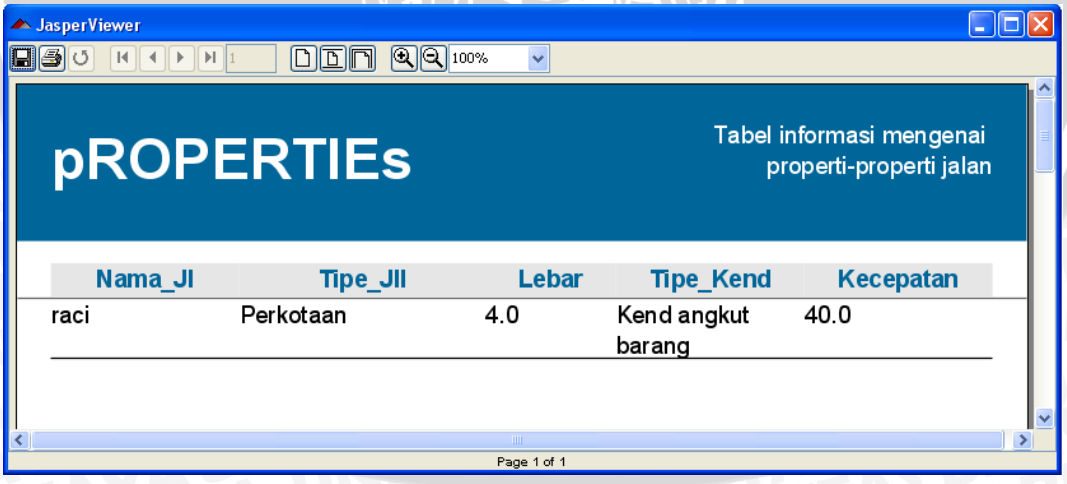
Antarmuka laporan koordinat digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel koordinat. Tampilan antarmuka ini adalah sebagai berikut.



Gambar 5.16 Tampilan Laporan Coordinat

5.4.15. Implementasi Antarmuka Laporan Properti Jalan

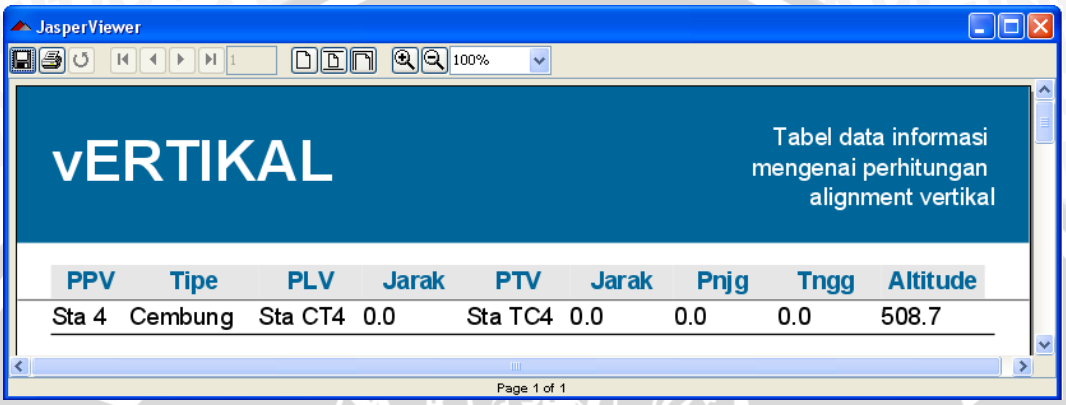
Antarmuka laporan properti jalan digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel *preparation*. Tampilan antarmuka ini adalah sebagai berikut.



Gambar 5.17 Tampilan Laporan Properti Jalan

5.4.16. Implementasi Antarmuka Laporan *Alignment* Vertikal

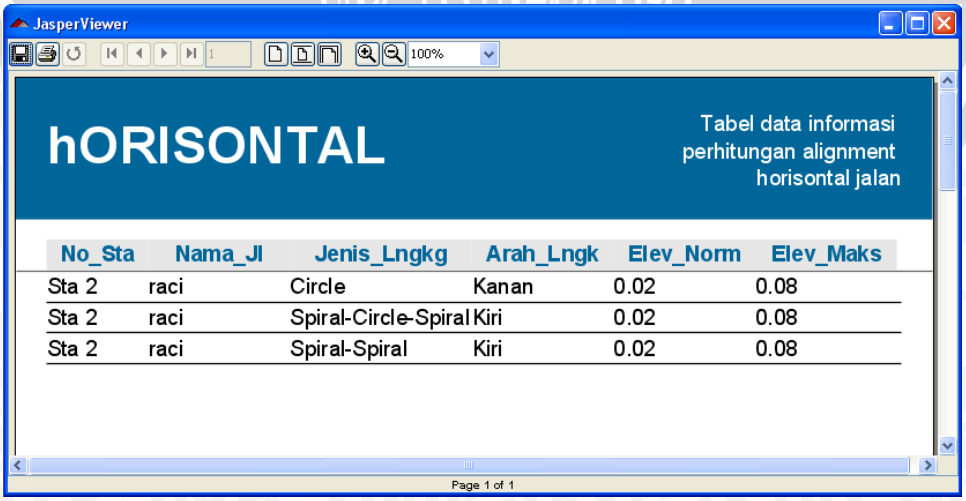
Antarmuka laporan *alignment* vertikal digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel vertikal. Tampilan antarmuka ini adalah sebagai berikut.



Gambar 5.18 Tampilan Laporan *Alignment* Vertikal

5.4.17. Implementasi Antarmuka Laporan *Alignment* Horisontal

Antarmuka laporan *alignment* horisontal digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel horisontal. Tampilan antarmuka ini adalah sebagai berikut.



Gambar 5.19 Tampilan Laporan *Alignment* Horisontal

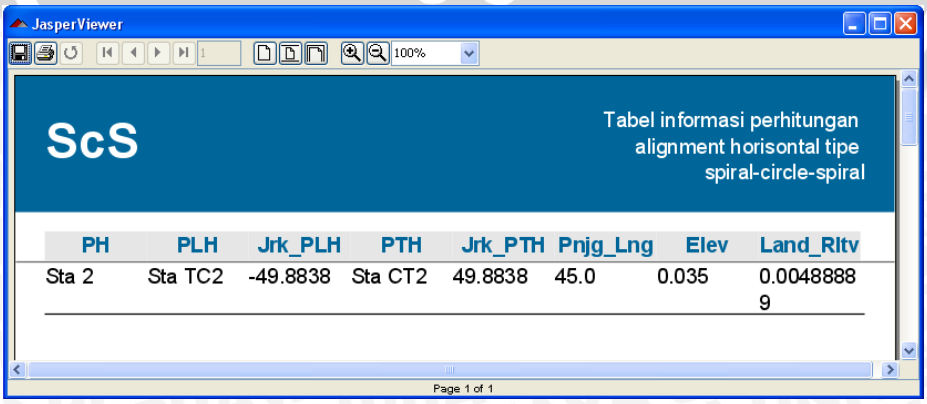
5.4.18. Implementasi Antarmuka Laporan Horisontal tipe Cicle

Antarmuka laporan horisontal tipe *circle* digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel horC. Tampilan antarmuka ini adalah sebagai berikut.

Gambar 5.20 Tampilan Laporan *Alignment* Horisontal tipe *Circle*

5.4.19. Implementasi Antarmuka Laporan Horisontal tipe Spiral-*Circle*-Spiral

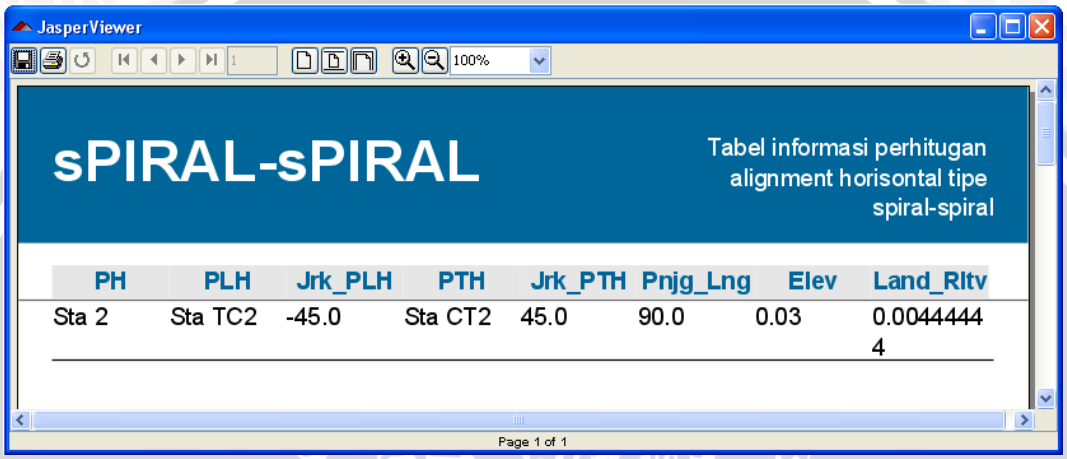
Antarmuka laporan horisontal tipe spiral-*circle*-spiral digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel horSCS. Tampilan antarmuka ini adalah sebagai berikut.



Gambar 5.21 Tampilan Laporan *Alignment* Horisontal tipe Spiral-*Circle*-Spiral

5.4.20. Implementasi Antarmuka Laporan Horisontal tipe Spiral-Spiral

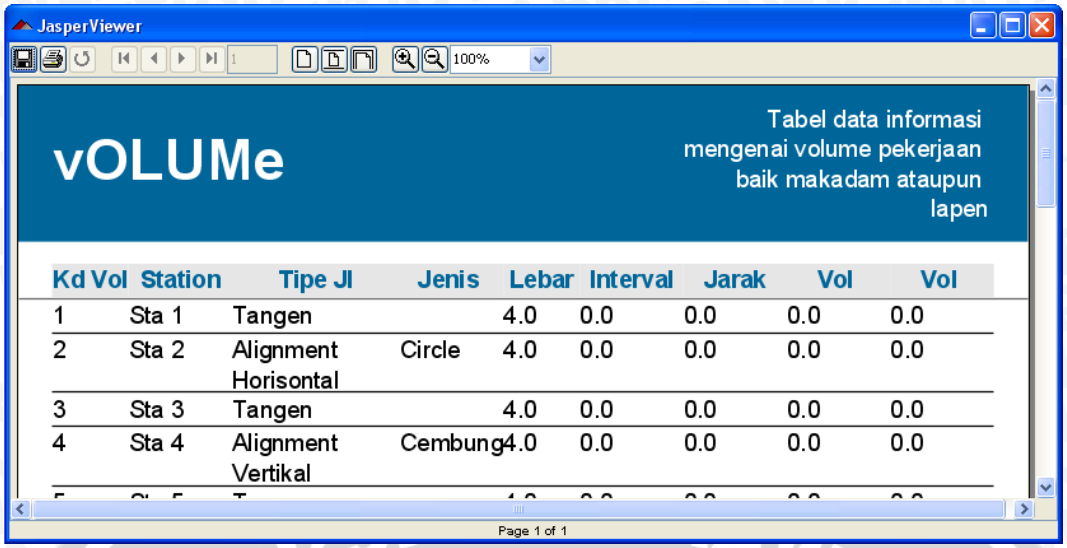
Antarmuka laporan horisontal tipe spiral-spiral digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel horSS. Tampilan antarmuka ini adalah sebagai berikut.



Gambar 5.22 Tampilan Laporan *Alignment* Horisontal tipe Spiral-Spiral

5.4.21. Implementasi Antarmuka Laporan Volume Kerja

Antarmuka laporan volume kerja digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel volume. Tampilan antarmuka ini adalah sebagai berikut.



Gambar 5.23 Tampilan Laporan Volume Kerja

5.4.22. Implementasi Antarmuka Laporan Data *Operator*

Antarmuka laporan *operator* digunakan untuk menampilkan laporan data-data yang telah tersimpan pada tabel *operator*. Tampilan antarmuka ini adalah sebagai berikut.

Gambar 5.24 Tampilan Laporan *Operator*

BAB VI

PENGUJIAN DAN ANALISA

Pada bab ini berisi pembahasan tentang pengujian dan analisa terhadap sistem yang telah dibangun. Dalam proses pengujian digunakan tiga buah metode pengujian, yaitu pengujian unit, pengujian validasi dan pengujian akurasi sistem. Pada pengujian unit akan digunakan teknik pengujian *white box*, sedangkan pada pengujian validasi akan digunakan metode pengujian *black box*. Proses analisis digunakan untuk mengetahui kinerja dari sistem perangkat lunak yang telah dibangun.

6.1. Pengujian Unit

Dalam sistem yang dibangun dengan pemrograman berorientasi objek, pengujian unit diterapkan untuk suatu metode dari suatu klas. Pada pengujian unit ini, digunakan teknik pegujian *white box* dengan teknik *basis path testing.* Pada teknik ini, proses pengujian dilakukan dengan memodelkan algoritma pada suatu *flow graph,* menentukan jumlah kompleksitas siklomatis, menentukan sebuah basis set dari jalur independen dan memberikan kasus uji pada setiap basis set yang telah ditentukan.

6.1.1. Pengujian Unit Untuk Operasi Pencarian *Device* Gps

Operasi pencarian merupakan salah satu implementasi dari algoritma klas konneksi *bluetooth*. Dalam gambar berikut diperlihatkan proses pemodelan dalam *flow graph* pada operasi pencarian klas konneksi *bluetooth*.

Gambar 6.1 Pemodelan Metode Pencarian Klas Konek ke dalam *flow graph*

6.1.1.1. Tujuan Pengujian

Pengujian terhadap operasi pencarian yang terdapat dalam klas konneksi *bluetooth* ini bertujuan untuk mengetahui kerja dari metode pencarian tersebut secara independen dalam menjalankan fungsinya untuk mengubah sistem ke dalam mode *discoverable* dan menampilkannya kepada *user* untuk memberikan pilihan *device* yang akan dikoneksikan, serta berguna untuk mempertahankan koneksi hingga aplikasi dihentikan. Pengujian dilakukan dalam beberapa uji kasus untuk mengetahui per*form*a dari metode pencarian dalam menjalankan fungsinya melalui berbagai kemungkinan kondisi proses yang akan terjadi pada saat sistem dijalankan.

6.1.1.2. Prosedur Pengujian

Dari hasil pemodelan *flow graph* yang telah dilakukan, dapat ditentukan kompleksitas siklomatis melalui persamaan V(G) = E – N + 2, dimana V(G) merupakan jumlah kompleksitas siklomatis, E merupakan sisi dan N merupakan jumlah simpul.

$$V(G) = E - N + 2$$

$$= 8 - 7 + 2$$

$$= 3$$

Dari nilai kompleksitas siklomatik diatas, dapat ditentukan satu basis set dari jalur independen yang akan terjadi, yaitu :

Jalur 1 : 1, 6, 7

Jalur 2 : 1, 2, 3, 5, 7

Jalur 3 : 1, 2, 3, 4, 5, 7

Pengujian kasus dan hasil eksekusi untuk masing-masing jalur adalah sebagai berikut:

a. Kasus Jalur 1

   Menjalankan klas konneksi *bluetooth* dengan menonaktifkan atau tidak mengkoneksikan *bluetooth device*.

   Hasil yang diharapkan : metode pencarian dari klas koneksi *bluetooth* tidak dapat dijalankan dan memberikan pemberitahuan kepada *user* atas permasahan yang terjadi.

   Hasil pengujian : metode pencarian tidak dapat dilakukan dan muncul *dialogue message box* yang memberitahukan bahwa *bluetooth device* tidak aktif.

b. Kasus Jalur 2

Menjalankan klas koneksi *bluetooth* dengan *bluetooth device* yang aktif akan tetapi gps *device* bluetooh tidak diaktifkan, atau tidak terdapat satupun perangkat yang terdiscovery.

Hasil yang diharapkan : metode pencarian dapat berjalan dengan baik dan menampilkan pemberitahuan bahwa tidak terdapat *bluetooth* dari perangkat lain disekitar sistem yang aktif.

Hasil pengujian : metode pencarian berjalan baik dan menampilkan bahwa hasil dari proses pencarian adalah kosong dan tidak ditemukan *device* lainnya.

c. Kasus Jalur 3

Menjalankan klas koneksi *bluetooth* dengan *bluetooth device* aktif dan gps *device* yang aktif pula.

Hasil yang diharapkan : metode pencarian dapat berjalan dengan baik dan menampilan nama *device* yang terdiscovery beserta dengan alamatnya.

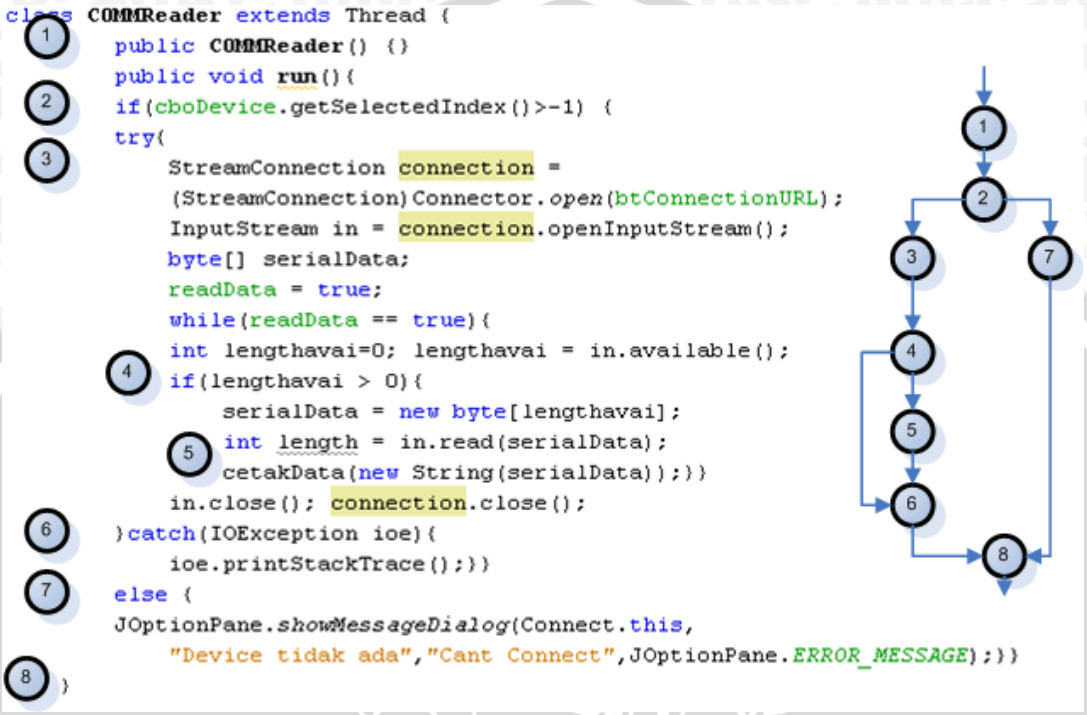
Hasil pengujian : metode pencarian berjalan dengan baik dan menampilkan jumlah perangkat yang terdeteksi beserta list nama dan alamat *device*.

6.1.1.3. Analisis Pengujian

Pengujian unit untuk metode pencarian dalam klass koneksi *bluetooth* memberikan hasil pengujian yang sesuai dengan hasil yang diharapkan berdasarkan kasus uji. Hasil pengujian metode pencarian ini membuktikan bahwa sistem yang bertindak sebagai pendiscovery perangkat lain mampu untuk menjalankan fungsinya dengan baik.

6.1.2. Pengujian Unit Untuk Operasi Koneksi

Operasi koneksi merupakan fungsi utama dari implementasi klass koneksi *bluetooth*. Melalui gambar berikut diperlihatkan proses pemodelan dalam *flow graph* pada operasi koneksi klas koneksi *bluetooth*.



Gambar 6.2 Pemodelan Operasi Konek ke dalam *flow graph*

6.1.2.1. Tujuan Pengujian

Pengujian terhadap operasi konek yang terdapat dalam klas koneksi *bluetooth*, bertujuan untuk mengetahui kinerja dari operasi konek secara independen dalam menjalankan fungsinya untuk mengkoneksikan perangkat eksternal sistem dengan sistem itu sendiri. Pengujian dilakukan dalam beberapa uji kasus dalam berbagai kemungkinan kondisi yang akan terjadi pada saat sistem menjalan operasi tersebut.

6.1.2.2. Prosedur Pengujian

Dari hasil pemodelan *flow graph* yang telah dilakukan terhadap operasi konek, dapat ditentukan jumlah kompleksitas siklomatis melalui persamaan berikut.

$$V(G) = E - N + 2$$
$$= 9 - 8 + 2$$
$$= 3$$

Dari nilai kompleksitas siklomatik diatas, dapat ditentukan satu basis set dari jalur independen yang akan terjadi, yaitu :

Jalur 1 : 1, 2, 7, 8

Jalur 2 : 1, 2, 3, 4, 6, 8

Jalur 3 : 1, 2, 3, 4, 5, 6, 8

Pengujian kasus dan hasil eksekusi untuk masing-masing jalur adalah sebagai berikut:

a. Kasus Jalur 1

   Menjalankan klas konneksi *bluetooth* dengan tanpa ada perangkat yang ditemukan.

   Hasil yang diharapkan : metode koneksi dari klas koneksi *bluetooth* tidak dapat dijalankan dan memberikan pemberitahuan kepada *user* atas permasalahan yang terjadi.

   Hasil pengujian : metode konek tidak dapat dilakukan dan muncul *dialogue message box* yang memberitahukan bahwa *device* tidak ada.

b. Kasus Jalur 2

Menjalankan klas koneksi *bluetooth* dengan terjadi kegagalan koneksi disebabkan karena terjadi gangguan terhadap *streaming connection*.

Hasil yang diharapkan : metode konek tidak dapat berjalan dan klass koneksi *form* berada pada posisi *stand by* menunggu hingga kegagalan koneksi dapat diatasi.

Hasil pengujian : metode konek tidak dapat berjalan dan klass koneksi *bluetooth* pada posisi *idle*.

c. Kasus Jalur 3

Menjalankan klas koneksi *bluetooth* dengan *device* yang telah diketemukan sebelumnya dan tanpa terjadi ganguan terhadap *streaming connection*.

Hasil yang diharapkan : metode konek dapat berjalan dengan baik dan menampilan data hasil *streaming connection* pada tab selanjutnya.

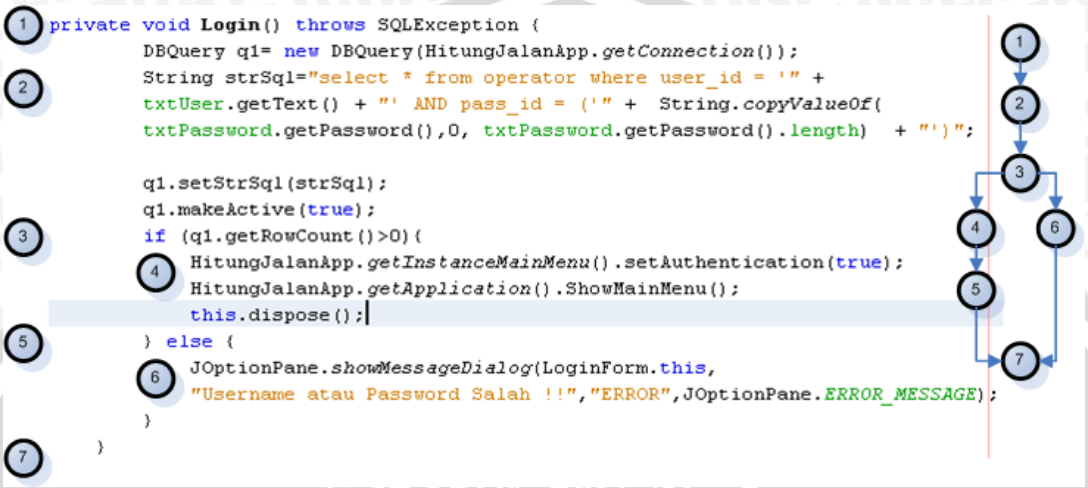
Hasil pengujian : metode konek berjalan dengan baik dan menampilkan tab selanjutnya yang berisi data hasil *streaming connection*.

6.1.2.3. Analisis Pengujian

Pengujian unit untuk metode konek dalam klass koneksi *bluetooth* memberikan hasil pengujian yang sesuai dengan hasil yang diharapkan berdasarkan kasus uji. Hasil pengujian metode konek ini membuktikan bahwa sistem yang bertindak sebagai penghubung dengan perangkat lain mampu untuk menjalankan fungsinya dengan baik.

6.1.3. Pengujian Unit Untuk Operasi Login

Operasi login merupakan implementasi dari klass login sebagai proses validasi *user* yang berhak mengakses program aplikasi. Dalam gambar berikut diperlihatkan proses pemodelan dalam *flow graph* pada operasi login.



Gambar 6.3 Pemodelan Operaso Login ke dalam *flow graph*

6.1.3.1. Tujuan Pengujian

Pengujian terhadap operasi login ini bertujuan untuk mengetahui kinerja dari operasi login secara independen dalam menjalankan fungsinya untuk memvalidasi *user* yang akan mengakses aplikasi. Pengujian dilakukan dalam beberapa uji kasus dengan berbagai kemungkinan yang akan terjadi pada saat aplikasi dijalankan.

6.1.3.2. Prosedur Pengujian

Dari hasil pemodelan ke dalam *flow graph* yang telah dilakukan, dapat ditentukan jumlah kompleksitas siklomatis melalui persamaan sebagai berikut.

$$V(G) = E - N + 2$$

$$= 7 - 7 + 2$$

= 2

Dari hasil perhitungan kompleksitas siklomatis, terdapat dua basis set dari jalur independen yaitu :

Jalur 1 : 1, 2, 3, 6, 7

Jalur 2 : 1, 2, 3, 4, 5, 7

Pengujian kasus dan hasil eksekusi untuk masing-masing jalur adalah sebagai berikut:

a. Kasus Jalur 1

Menjalankan klas login dengan menginputkan nama dan password yang salah ataupun mengkosongkan *field* nama dan password.

Hasil yang diharapkan : metode login dari klas login *form* akan menampilkan *message dialogue box* yang memberitahukan bahwa nama dan password yang dimasukkan salah, dan kembali pada *form* login lagi.

Hasil pengujian : metode login memunculkan *dialogue message box* yang memberitahukan bahwa nama dan password salah.

b. Kasus Jalur 2

Menjalankan klas login dengan menginputkan nama dan password yang benar.

Hasil yang diharapkan : metode login berjalan dengan baik kemudian mengalihkan *form* kepada tampilan utama aplikasi.

Hasil pengujian : metode login berjalan dengan baik dan menampilkan halaman utama aplikasi dengan menu-menu dan submenu yang lengkap.

6.1.3.3. Analisis Pengujian

Pengujian unit untuk operasi login dalam klas login *form* memberikan hasil yang sesuai dengan yang diharapkan untuk masing-masing jalur uji sesuai dengan pemodelan *flow graph* yang ditampilkan pada gambar 6.3. Hasil pengujian membuktikan bahwa sistem validasi *user* dapat berjalan dengan baik.

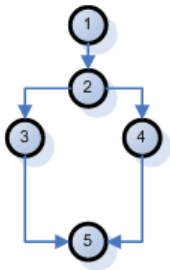
6.1.4. Pengujian Unit Untuk Operasi Penyimpanan Data ke *Storage*

Operasi penyimpanan data ke *storage* merupakan implementasi dari berbagai klas yang terdapat dalam aplikasi sebagai proses penyimpanan data-data sementara untuk diperhitungkan. Operasi ini berada pada sebagaian besar klass-klass sistem, terutama pada klas-klas yang mempergunakan perhitungan-perhitungan. Dalam gambar berikut diperlihatkan proses pemodelan dalam *flow graph* pada operasi penyimpanan data ke *storage* dalam klass *preparation*.

```
1  private void save() {
       try{
2        if(append==true){
3          tPrepare.append();
           tPrepare.updateString("kode_prep", kdTxt.getText());
         }else{
4          tSta.edit();
         }

5  tPrepare.updateString("type_kend", cboCar.getSelectedItem().toString());
   tPrepare.updateFloat("kecepatan", Float.parseFloat(speedTxt.getText()));
   tPrepare.updateString("nama_jl", namaTxt.getText());
   tPrepare.updateString("type_jl", cboRoad.getSelectedItem().toString());
   tPrepare.updateFloat("lebar", Float.parseFloat(widthTxt.getText()));
   tPrepare.post();
   }
       catch(SQLException sqle) {System.err.println(sqle);

   }
   }
```



Gambar 6.4 Pemodelan Operasi Penyimpanan ke dalam *flow graph*

6.1.4.1. Tujuan Pengujian

Pengujian terhadap operasi penyimpanan ini bertujuan untuk mengetahui kinerja dari operasi penyimpanan secara independen dalam menjalankan fungsinya untuk melakukan penyimpanan data-data hasil perhitungan untuk dipergunakan dalam perhitungan selanjutnya. Pengujian dilakukan dalam beberapa uji kasus dengan berbagai kemungkinan yang akan terjadi pada saat aplikasi dijalankan.

6.1.4.2. Prosedur Pengujian

Dari hasil pemodelan ke dalam *flow graph* yang telah dilakukan, dapat ditentukan jumlah kompleksitas siklomatis melalui persamaan sebagai berikut.

$$V(G) = E - N + 2$$
$$= 5 - 5 + 2$$
$$= 2$$

Dari hasil perhitungan kompleksitas siklomatis, terdapat dua basis set dari jalur independen yaitu :

Jalur 1 : 1, 2, 4, 5

Jalur 2 : 1, 2, 3, 5

Pengujian kasus dan hasil eksekusi untuk masing-masing jalur adalah sebagai berikut:

a. Kasus Jalur 1

Menjalankan klas penyimpanan dengan pemasukan data data-data yang benar tanpa ada *field-field* yang kosong.

Hasil yang diharapkan : metode penyimpanan data dari klas *preparation* akan menyimpan data-data pada *field* yang berada dalam *form* ke dalam *storage* dan menampilkan *message dialogue box* yang memberitahukan bahwa data telah tersimpan.

Hasil pengujian : metode penyimpanan data menyimpan data pada *field* ke dalam *storage* dan memunculkan *dialogue message box* yang memberitahukan bahwa data telah tersimpan.

b. Kasus Jalur 2

Menjalankan klas penyimpanan dengan pemasukan data data-data yang tidak benar atau terdapat *field* kosong.

Hasil yang diharapkan : metode penyimpanan data tidak dapat berjalan kemudian menampilkan *message dialogue box* bahwa terdapat kesalahan pada data.

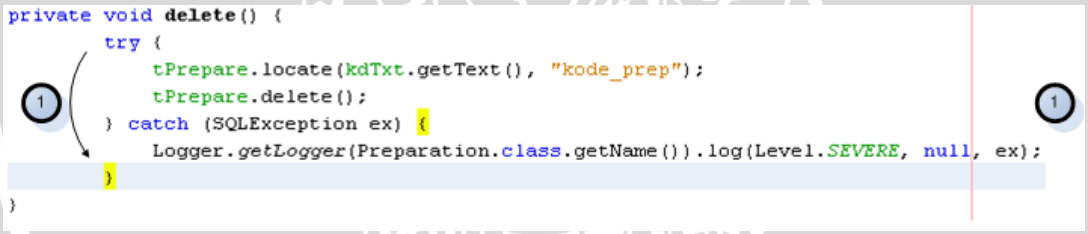
Hasil pengujian : metode penyimpanan data tidak berjalan dan muncul *message dialogue box message dialogue box* yang berisi pemberitahuan bahwa data salah.

6.1.4.3. Analisis Pengujian

Pengujian unit untuk operasi penyimpanan data dalam klas *preparation* memberikan hasil yang sesuai dengan yang diharapkan untuk masing-masing jalur uji sesuai dengan pemodelan *flow graph* yang ditampilkan pada gambar 6.4. Hasil pengujian membuktikan bahwa sistem penyimpanan data dapat berjalan dengan baik.

6.1.5. Pengujian Unit Untuk Operasi Penghapusan Data dari *Storage*

Operasi penghapusan data dari *storage* merupakan implementasi dari berbagai klas yang terdapat dalam aplikasi sebagai proses penghapusan data-data baik yang salah maupun yang tidak diperlukan dalam *storage*. Operasi ini berada pada sebagaian besar klass-klass sistem, terutama pada klas-klas yang memerlukan proses penyimpanan data. Dalam gambar berikut diperlihatkan proses pemodelan dalam *flow graph* pada operasi penghapusan data dari *storage* dalam klass *preparation*.



Gambar 6.5 Pemodelan Operasi Penyimpanan ke dalam *flow graph*

6.1.5.1. Tujuan Pengujian

Pengujian terhadap operasi penghapusan ini bertujuan untuk mengetahui kinerja dari operasi penghapusan data secara independen dalam menjalankan fungsinya untuk melakukan penghapusan data-data yang berada di dalam tempat penyimpanan data. Pengujian dilakukan dalam beberapa uji kasus dengan berbagai kemungkinan yang akan terjadi pada saat aplikasi dijalankan.

6.1.5.2. Prosedur Pengujian

Dari hasil pemodelan ke dalam *flow graph* yang telah dilakukan, dapat ditentukan jumlah kompleksitas siklomatis melalui persamaan sebagai berikut.

$$V(G) = E - N + 2$$

$$= 0 - 1 + 2$$

$$= 1$$

Dari hasil perhitungan kompleksitas siklomatis, terdapat satu basis set dari jalur independen yaitu :

Jalur 1 : 1

Pengujian kasus dan hasil eksekusi untuk masing-masing jalur adalah sebagai berikut:

a. Kasus Jalur 1

Menjalankan klas penghapusan data dengan memberikan pilihan pada tabel data sementara di dalam *form*.

Hasil yang diharapkan : metode penghapusan data dapat berjalan dengan baik dan menampilan *message dialogue box* yang memberitahukan bahwa data telah terhapus.

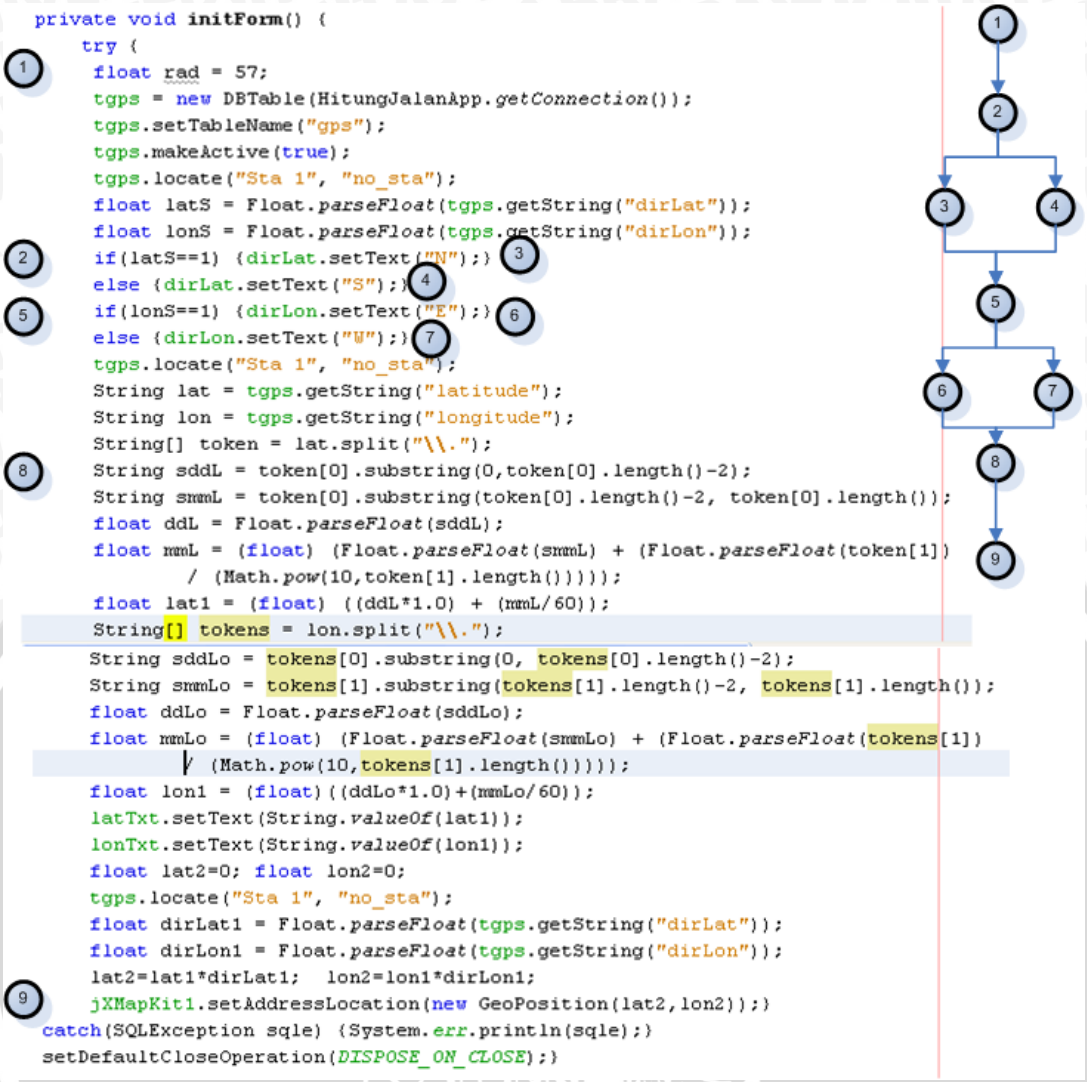
Hasil pengujian : metode penghapusan data memunculkan *dialogue message box* yang memberitahukan bahwa data telah terhapus, dan data pada *storage* telah benar-benar terhapus sehingga yang ditampilkan pada tabel dalam *form* adalah data-data saat ini.

6.1.5.3. Analisis Pengujian

Pengujian unit untuk operasi penghapusan data dalam klas *preparation* memberikan hasil yang sesuai dengan yang diharapkan untuk masing-masing jalur uji sesuai dengan pemodelan *flow graph* yang ditampilkan pada gambar 6.5. Hasil pengujian membuktikan bahwa sistem penghapusan data yang berada di dalam *storage* dapat berjalan dengan baik.

6.1.6. Pengujian Unit Untuk Operasi Penampilan Hasil pada Map

Operasi penampilan map merupakan fungsi utama dari implementasi klass koneksi mapping. Melalui gambar berikut diperlihatkan proses pemodelan dalam *flow graph* pada operasi penampilan map klas mapping.



Gambar 6.6 Pemodelan Operasi Penampilan Map ke dalam *flow graph*

6.1.6.1. Tujuan Pengujian

Pengujian terhadap operasi penampilan map yang terdapat dalam klas *maping*, bertujuan untuk mengetahui kinerja dari operasi penampilan map secara independen dalam menjalankan fungsinya untuk menampilkan hasil perhitungan ke dalam suatu objek gambar proyeksi dua dimensi berdasarkan skala peta. Pengujian dilakukan dalam beberapa uji kasus dalam berbagai kemungkinan kondisi yang akan terjadi pada saat sistem menjalan operasi tersebut.

6.1.6.2. Prosedur Pengujian

Dari hasil pemodelan *flow graph* yang telah dilakukan terhadap operasi konek, dapat ditentukan jumlah kompleksitas siklomatis melalui persamaan berikut.

$$V(G) = E - N + 2$$
$$= 10 - 8 + 2$$
$$= 4$$

Dari nilai kompleksitas siklomatik diatas, dapat ditentukan satu basis set dari jalur independen yang akan terjadi, yaitu :

Jalur 1 : 1, 2, 3, 5, 6, 8, 9

Jalur 2 : 1, 2, 4, 5, 6, 8, 9

Jalur 3 : 1, 2, 3, 5, 7, 8, 9

Jalur 4 : 1, 2, 4, 5, 7, 8, 9

Pengujian kasus dan hasil eksekusi untuk masing-masing jalur adalah sebagai berikut:

d. Kasus Jalur 1

Menjalankan klas penampilan map apabila data koordinat berada pada arah NE (Timur Laut).

Hasil yang diharapkan : metode penampilan map dari klas *maping* menampilkan grafik peta hasil perhitungan yang berada pada posisi timur laut dihitung dari letak koordinat lintang 0 dan bujur 0.

Hasil pengujian : grafik peta hasil perhitungan dapat ditampilkan yang dilengkapi dengan atribut-atributnya.

e. Kasus Jalur 2

Menjalankan klas penampilan map apabila data koordinat berada pada arah SE (Tenggara).

Hasil yang diharapkan : metode penampilan map dari klas *maping* menampilkan grafik peta hasil perhitungan yang berada pada posisi tenggara dihitung dari letak koordinat lintang 0 dan bujur 0.

Hasil pengujian : grafik peta hasil perhitungan dapat ditampilkan yang dilengkapi dengan atribut-atributnya.

f. Kasus Jalur 3

Menjalankan klas penampilan map apabila data koordinat berada pada arah NW (Barat Laut).

Hasil yang diharapkan : metode penampilan map dari klas *maping* menampilkan grafik peta hasil perhitungan yang berada pada posisi barat laut dihitung dari letak koordinat lintang 0 dan bujur 0.

Hasil pengujian : grafik peta hasil perhitungan dapat ditampilkan yang dilengkapi dengan atribut-atributnya.

g. Kasus Jalur 4

Menjalankan klas penampilan map apabila data koordinat berada pada arah SW (Barat Daya).

Hasil yang diharapkan : metode penampilan map dari klas *maping* menampilkan grafik peta hasil perhitungan yang berada pada posisi barat daya dihitung dari letak koordinat lintang 0 dan bujur 0.

Hasil pengujian : grafik peta hasil perhitungan dapat ditampilkan yang dilengkapi dengan atribut-atributnya.

6.1.6.3. Analisis Pengujian

Pengujian unit untuk metode penampilan map dalam klass *maping* memberikan hasil pengujian yang sesuai dengan hasil yang diharapkan berdasarkan kasus uji. Hasil pengujian metode penampilan map ini membuktikan bahwa sistem yang bertindak sebagai pemroyeksi hasil perhitungan ke dalam bentuk peta mampu untuk menjalankan fungsinya dengan baik.

6.2. Pengujian Validasi

Tujuan dari pengujian validasi sistem ini adalah untuk menguji kesesuaian antara kebutuhan akan penyusunan sistem dengan kinerja dari sistem itu sendiri. Pada pengujian validasi digunakan metode pengujian *Black Box*, dimana pengujian tidak memerlukan konsentrasi terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan kesesuaian kinerja sistem dengan daftar kebutuhan. Objek dari pengujian validasi adalah item-item kebutuhan yang digunakan sebagai acuan dalam menyusun program. Kasus pengujian terhadap item-item tersebut akan dijabarkan sebagai berikut.

6.2.1. Kasus Uji Validasi

Untuk mengetahui kesesuaian kinerja sistem dengan kebutuhan *user*, maka pada setiap kebutuhan dilakukan proses pengujian dengan masing-masing kasus uji.

a. Kasus Uji Login

Objek Uji	:	Kebutuhan Otorisasi *User*
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat

melakukan proses validasi terhadap *user* yang berhak mengakses atau tidak.

Prosedur uji : 1. Menjalankan aplikasi untuk pertama kalinya
2. Memasukkan nama dan password *user*

Hasil yang diharapkan : Sistem akan membaca data masukan *user* dan melakukan pengecekan dengan *database operator*, jika cocok maka sistem akan menampilkan menu utama jika tidak maka sistem akan menampilkan pemberitahuan

b. Kasus Uji Pencarian *Device* dengan *Bluetooth*

Objek Uji : Kebutuhan *Discovery Device*

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses pencarian terhadap *device-device* di sekitar sistem melalui perangkat koneksi *bluetooth*.

Prosedur uji : 1. Memilih tombol konek pada *toolbar*
2. Kemudian menekan tombol *searching* pada *form* konek

Hasil yang diharapkan : Sistem akan mencari *device-device* yang dapat terhubung melalui media *bluetooth* dan menampilkan jumlah *device*, list nama dan alamat dari *device* tersebut.

c. Kasus Uji *Streaming* Data *Device*

Objek Uji	:	Kebutuhan *Streaming Data Device*
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pembacaan data terhadap *gps bluetooth device* yang telah ter*discovery*.
Prosedur uji	:	1. Menekan tombol konek pada *form* konek 2. Memindah tab *form* pada tab data
Hasil yang diharapkan	:	Sistem akan melakukan koneksi terhadap *device* yang telah dipilih oleh *user*, kemudian menampilkan data *streaming*nya pada tab data.

d. Kasus Uji Pengolahan Data Koordinat

Objek Uji	:	Kebutuhan Data Koordinat
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pembacaan dan pengolahan data koordinat.
Prosedur uji	:	1. Memilih menu Perhitungan 2. Memilih sub menu koordinta 3. Melakukan entri data pada *field* 4. Menekan tombol *convert* 5. Menekan tombol *save*

Hasil yang diharapkan : Sistem akan menampilkan *form* koordinat, setelah itu *user* dapat menginputkan data-data hasil parsing *streaming* untuk dikonversi kedalam bentuk *decimal degree*, kemudian disimpan ke dalam *storage*.

e. Kasus Uji Penghapusan Data Koordinat

Objek Uji : Kebutuhan Penghapusan Data Koordinat

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses pembacaan data koordinat di dalam *storage* dan menghapus data yang dipilih *user* dari dalam *storage*.

Prosedur uji :
1. *User* memilih data pada tabel yang tersedia
2. Memilih tombol *delete* pada *form* koordinat

Hasil yang diharapkan : Sistem akan membaca data dari tabel pada *form* koordinat untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan terhapus.

f. Kasus Uji Pengolahan Data *Stationing*

Objek Uji : Kebutuhan Data *Stationing*

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat

melakukan pembacaan data konversi koordinat untuk ditentukan letak *stationing*nya.

Prosedur uji : 1. Memilih menu perhitungan
2. Memilih sub menu *stationing*
3. Proses masukan dari *user*
4. Menekan tombol hitung
5. Menekan tombol *save*

Hasil yang diharapkan : Sistem akan menampilkan *form stationing*, setelah itu *user* dapat menginputkan data konversi koordinat untuk diset titik statioingnya, kemudian sistem akan melakukan proses perhitungan diikuti dengan penyimpanan data hasil perhitungan pada *storage*.

g. Kasus Uji Penghapusan Data *Stationing*

Objek Uji : Kebutuhan Penghapusan Data *Stationing*

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data *stationing* dari *storage*.

Prosedur uji : 1. *User* memilih data yang hendak di hapus
2. Menekan tombol *delete*

Hasil yang diharapkan : Sistem akan membaca data pada tabel yang

telah terseleksi oleh *user* kemudian dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus dari *storage*.

h. Kasus Uji Pengolahan Data Properti jalan

Objek Uji	:	Kebutuhan Data Properti Jalan
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pengolahan data-data properti jalan.
Prosedur uji	:	1. Memilih menu perhitungan 2. Memilih sub menu *preparation* 3. Memasukkan data 4. Menekan tombol *save*
Hasil yang diharapkan	:	Sistem akan menampilkan *form preparation*, ksetelah itu *user* dapat melakukan pemasukan data data, kemudian data tersebut disimpan ke dalam *storage*.

i. Kasus Uji Penghapusan Data Properti Jalan

Objek Uji	:	Kebutuhan Penghapusan Data Properti Jalan
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data properti jalan pada *storage*.

Prosedur uji	:	1. Memilih data pada tabel 2. Menekan tombol *delete*
Hasil yang diharapkan	:	Sistem akan membaca data pada tabel yang terseleksi oleh *user* untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus.

j. Kasus Uji Pengolahan Data *Alignment* Horisontal

Objek Uji	:	Kebutuhan Data *Alignment* Horisontal
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pengolahan data-data *alignment* horisontal.
Prosedur uji	:	1. Memilih menu perhitungan 2. Memilih sub menu *alignment* horisontal 3. *User* melakukan pemasukan data 4. Menekan tombol hitung 5. Menekan tombol *save*
Hasil yang diharapkan	:	Sistem akan menampilkan *form alignment* horisontal, setelah itu *user* dapat menginputkan data-data *alignment* horisontal, kemudian sistem akan melakukan proses perhitungan diikuti dengan penyimpanan data-data hasil perhitungan ke dalam *storage*.

k. Kasus Uji Penghapusan Data *Alignment* Horisontal

Objek Uji	:	Kebutuhan Penghapusan Data *Alignment* Horisontal
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data *alignment* horisontal pada *storage*.
Prosedur uji	:	1. Memilih data pada tabel 2. Menekan tombol *delete*
Hasil yang diharapkan	:	Sistem akan membaca data pada tabel yang terseleksi oleh *user* untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus.

l. Kasus Uji Pengolahan Data *Alignment* Horisontal tipe *Circle*-Cricel

Objek Uji	:	Kebutuhan Data *Circle-Circle*
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pengolahan data-data *alignment* horisontal tipe *circle-circle*.
Prosedur uji	:	1. Memilih menu perhitungan 2. Memilih sub menu *circle-circle* 3. *User* melakukan pemasukan data 4. Menekan tombol hitung 5. Menekan tombol *save*

Hasil yang diharapkan : Sistem akan menampilkan *form alignment* horisontal tipe *circle-circle*, setelah itu *user* dapat menginputkan data-data, kemudian sistem akan melakukan proses pengitungan diikuti dengan penyimpanan data-data hasil perhitungan ke dalam *storage*.

m. Kasus Uji Penghapusan Data *Alignment* Horisontal tipe *Circle*-Cricle

Objek Uji : Kebutuhan Penghapusan Data *Alignment* Horisontal Tipe *Circle-Circle*

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data *alignment* horisontal tipe *circle-circle* pada *storage*.

Prosedur uji :
1. Memilih data pada tabel
2. Menekan tombol *delete*

Hasil yang diharapkan : Sistem akan membaca data pada tabel yang terseleksi oleh *user* untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus.

n. Kasus Uji Pengolahan Data *Alignment* Horisontal tipe *Circle*-Spiral-*Circle*

Objek Uji : Kebutuhan Data *Circle*-Spiral-*Circle*

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat

melakukan proses pengolahan data-data *alignment* horisontal tipe *circle*-spiral-*circle*.

Prosedur uji : 1. Memilih menu perhitungan
2. Memilih sub menu *circle*-spiral-*circle*
3. *User* melakukan pemasukan data
4. Menekan tombol hitung
5. Menekan tombol *save*

Hasil yang diharapkan : Sistem akan menampilkan *form alignment* horisontal tipe *circle*-spiral-*circle,* setelah itu *user* dapat menginputkan data-data, kemudian sistem akan melakukan proses perhitungan diikuti dengan penyimpanan data-data hasil perhitungan ke dalam *storage*.

o. Kasus Uji Penghapusan Data *Alignment* Horisontal tipe *Circle*-Spiral-*Circle*

Objek Uji : Kebutuhan Penghapusan Data *Alignment* Horisontal tipe *Circle*-Spiral-*Circle*

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data *alignment* horisontal tipe *circle*-spiral-*circle* pada *storage*.

Prosedur uji : 1. Memilih data pada tabel
2. Menekan tombol *delete*

Hasil yang diharapkan : Sistem akan membaca data pada tabel yang

terseleksi oleh *user* untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus.

p. Kasus Uji Pengolahan Data *Alignment* Horisontal tipe Spiral-Spiral

Objek Uji	:	Kebutuhan Data Spiral-Spiral
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pengolahan data-data *alignment* horisontal tipe spiral-spiral.
Prosedur uji	:	1. Memilih menu perhitungan 2. Memilih sub menu spiral-spiral 3. *User* melakukan pemasukan data 4. Menekan tombol hitung 5. Menekan tombol *save*
Hasil yang diharapkan	:	Sistem akan menampilkan *form alignment* horisontal tipe spiral-spiral, setelah itu *user* dapat menginputkan data-data, kemudian sistem akan melakukan penghitungan data diikuti dengan penyimpanan data-data hasil perhitungan ke dalam *storage*.

q. Kasus Uji Penghapusan Data Alignemtn Horisontal tipe Spiral-Spiral

Objek Uji	:	Kebutuhan Penghapusan Data *Alignment* Horisontal tipe Spiral-Spiral

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data *alignment* horisontal tipe spiral-spiral pada *storage*.

Prosedur uji :
1. Memilih data pada tabel
2. Menekan tombol *delete*

Hasil yang diharapkan : Sistem akan membaca data pada tabel yang terseleksi oleh *user* untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus.

r. Kasus Uji Pengolahan Data *Alignment* Vertikal

Objek Uji : Kebutuhan Data *Alignment* Vertikal

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses pengolahan data-data *alignment* vertikal.

Prosedur uji :
1. Memilih menu perhitungan
2. Memilih sub menu *alignment* vertikal
3. *User* melakukan pemasukan data
4. Menekan tombol hitung
5. Menekan tombol *save*

Hasil yang diharapkan : Sistem akan menampilkan *form alignment* vertikal, setelah itu *user* dapat menginputkan data-data, kemudian sistem akan melakukan

proses perhitungan diikuti dengan proses penyimpanan data-data hasil perhitungan ke dalam *storage*.

s. Kasus Uji Penghapusan Data *Alignment* Vertikal

Objek Uji	:	Kebutuhan Penghapusan Data *Alignment* Vertikal
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data *alignment* vertikal pada *storage*.
Prosedur uji	:	1. Memilih data pada tabel 2. Menekan tombol *delete*
Hasil yang diharapkan	:	Sistem akan membaca data pada tabel yang terseleksi oleh *user* untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus.

t. Kasus Uji Pengolahan Data Volume Kerja

Objek Uji	:	Kebutuhan Data Volume Kerja
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pengolahan data-data volume kerja.
Prosedur uji	:	1. Memilih menu perhitungan 2. Memilih sub menu volume kerja

3. *User* melakukan pemasukan data
4. Menekan tombol hitung
5. Menekan tombol *save*

Hasil yang diharapkan : Sistem akan menampilkan *form* volume kerja, setelah itu *user* dapat menginputkan data-data, kemudian sistem akan melakukan proses perhitungan diikuti dengan proses penyimpanan data-data hasil perhitungan ke dalam *storage*.

u. Kasus Uji Penghapusan Data Volume Kerja

Objek Uji : Kebutuhan Penghapusan Data Volume Kerja

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data volume kerja pada *storage*.

Prosedur uji :
1. Memilih data pada tabel
2. Menekan tombol *delete*

Hasil yang diharapkan : Sistem akan membaca data pada tabel yang terseleksi oleh *user* untuk dibandingkan dengan data pada *storage*, jika terdapat kesamaan maka data akan dihapus.

v. Kasus Uji Penampilan Data Koordinat ke Dalam Map

Objek Uji : Kebutuhan Data Koordinat dalam Map

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses penampilan data koordinat dalam proyeksi peta dua dimensi.

Prosedur uji :
1. Memilih menu report
2. Memilih sub menu map kerja
3. Menekan tombol jalan

Hasil yang diharapkan : Sistem akan menampilkan *form* map kerja dengan tampilan map dimana letak titik tengahnya adalah pada *station* pertama, kemudian setelah tombol jalan ditekan maka sistem akan menampilkan bentuk proyeksi dari perencanaan jalan pada peta.

w. Kasus Uji Penampilah Data Laporan

Objek Uji : Kebutuhan Data Laporan

Tujuan pengujian : Untuk memastikan bahwa aplikasi dapat melakukan proses penampilan data-data laporan dalam bentuk html agar mudah untuk disimpan dan dicetak.

Prosedur uji :
1. Memilih menu report
2. Memilih sub menu laporan

Hasil yang diharapkan : Sistem akan menampilkan *form* laporan dalam bentuk html dilengkapi dengan

beberapa macam *tools* untuk penyimpanan dan pencetakan data.

x. Kasus Uji Pengolahan Data *Operator*

Objek Uji	:	Kebutuhan Data *Operator*
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses pengolahan data-data *operator*.
Prosedur uji	:	1. Memilih menu admin 2. Memilih sub menu *operator* 3. *User* melakukan pemasukan data 4. Menekan tombol *save*
Hasil yang diharapkan	:	Sistem akan menampilkan *form* opertor, setelah itu *user* dapat menginputkan data-data, kemudian sistem akan melakukan penyimpanan data-data masukan ke dalam *storage*.

y. Kasus Uji Penghapusan Data *Operator*

Objek Uji	:	Kebutuhan Penghapusan Data *Operator*
Tujuan pengujian	:	Untuk memastikan bahwa aplikasi dapat melakukan proses penghapusan data-data *operator* yang tidak dipergunakan.
Prosedur uji	:	1. Memilih menu admin

2. Memilih sub menu reprot opertor
3. *User* melakukan pemilihan data pada tabel
4. Menekan tombol *delete*

Hasil yang diharapkan : Sistem akan membaca data pada tabel yang telah terseleksi oleh *user* untuk dicocokkan dengan data pada *storage*, jika terjadi persamaan maka data tersebut dihapus.

6.2.2. Hasil Pengujian Validasi

Dari kasus uji yang telah dilaksanakan sesuai dengan prosedur pengujian pada sub bab sebelumnya, didapatkan hasil pengujian seperti ditunjukkan pada tabel berikut.

Tabel 6.1 Hasil Pengujian Validasi

No	Kasus Uji	Hasil yang didapatkan	status
1	kasus uji login	sistem membaca data masukan *user* untuk dibandingkan dengan data pada *storage*, jika serupa maka sistem akan menampilkan halaman utam, jika salah maka sistem akan menampilkan pemberitahuan	valid
2	kasus uji searching	sistem mencari *device-device* disekitar sistem dengan mempergunkan koneksi *bluetooth*, setelah pencarian selesai sistem menampilkan jumlah *device* yang	valid

		ditemukan beserta list nama dan alamatnya	
3	kasus uji konek	sistem melakukan koneksi dengan *device* yang telah dipilih *user*, kemudian menampilkan data *streaming* pada tab data	valid
4	kasus uji koordinat	sistem menampilkan *form* koordinat dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan konversi data masukan dan menyimpan ke dalam *storage*	valid
5	kasus uji penghapusan koordinat	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid
6	kasus uji *stationing*	sistem menampilkan *form stationing* dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan penghitungan data masukan dan menyimpan ke dalam *storage*	valid
7	kasus uji penghapusan *stationing*	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid
8	kasus uji properti jalan	sistem menampilkan *form preparation* dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan penghitungan data masukan dan menyimpan ke dalam *storage*	valid

9	kasus uji penghapusan properti jalan	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid
10	kasus uji *alignment* horisontal	sistem menampilkan *form alignment* horisontal dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan penghitungan data masukan dan menyimpan ke dalam *storage*	valid
11	kasus uji penghapusan horisontal	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid
12	kasus uji *circle*-cirlce	sistem menampilkan *form circle-circle* dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan perhitungan data masukan dan menyimpan ke dalam *storage*	valid
13	kasus uji penghapusan *circle-circle*	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid
14	kasus uji spiral-*circle*-spiral	sistem menampilkan *form* spiral-*circle*-spiral dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan perhitungan data masukan dan menyimpan ke dalam *storage*	valid
15	kasus uji penghapusan	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika	valid

	spiral-*circle*-spiral	sebanding maka data akan terhapus	
16	kasus uji spiral-spiral	sistem menampilkan *form* spiral-spiral dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan perhitungan data masukan dan menyimpan ke dalam *storage*	valid
17	kasus uji penghapusan spiral-spiral	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid
18	kasus uji *alignment* vertikal	sistem menampilkan *form alignment* vertikal dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan perhitungan data masukan dan menyimpan ke dalam *storage*	valid
19	kasus uji penghapusan vertikal	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid
20	kasus uji volume kerja	sistem menampilkan *form* volume kerja dengan *field-field* yang masih kosong, jika pemasukan data oleh *user* selesai, maka sistem akan melakukan perhitungan data masukan dan menyimpan ke dalam *storage*	valid
21	kasus uji penghapusan volume kerja	sistem membaca data pada tabel yang telah diseleksi *user* dan membandingkan dengan data *storage*, jika sebanding maka data akan terhapus	valid

22	kasus uji penampilan map	sistem membaca data koordinat pada *storage* kemudian menampilkannya dalam proyeksi map yang menampilkan wilayah dari koordinat tersebut, disertai dengan bentuk rencana jalannya	valid
23	kasus uji penampilan laporan	sistem menampilkan hasil-hasil perhitungan yang terdapat dalam *storage* ke dalam bentuk html	valid
24	kasus uji pengolahan *operator*	sistem menampilkan *form operator* dengan *field-field* yang masih kosong, jika *user* selesai melakukan pemasukan data maka sistem akan menyimpan data tersebut ke *storage*	valid
25	kasus uji penghapusan *operator*	sistem membaca data di tabel yang telah terseleksi *user* untuk dibandingkan dengan data pada *storage*, jika data serupa maka sistem akan menghapus data dalam *storage* tersebut dan menampilkan kembali data yang baru pada tabel.	valid

6.2.3. Analisa Pengujian Validasi

Data hasil pengujian yang teradpat dalam tabel 6.1 merupakan data yang didapat dengan membandingkan per*form*a sistem dengan kebutuhan sistem.

Kasus uji 1 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional login yang memvalidasi akses masuk dari *user* sebelum mempergunakan sistem. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 2 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pencarian *bluetooth device*, sistem akan mendiscovery setiap *device-device bluetooth* di sekitar sistem dan menampilkannya dalam bentuk list kepada *user*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 3 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengkoneksian *bluetooth device* yang diketemukan dan diseleksi oleh *user*, setelah terkoneksi sistem melakukan pembacaan data *streaming* dan ditampilkan pada tab data. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 4 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data koordinat, sistem membaca masukan data *streaming* untuk ditampilkan kepada *user*, kemudian sistem mengkonversi data koordinat menjadi bentuk decimal degree, sistem menyimpan data konversi tersebut kedalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 5 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data koordinat, sistem membaca data pada tabel di *form* koordinat yang telah dipilih oleh *user* untuk dibandingkan dengan data koordinat di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data koordinat yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 6 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data *stationing*, sistem membaca masukan data koneversi koordinat yang telah diperhitungkan sebelumnya dan menampilkannya kepada *user*. *User* mensetting tiap-tiap titik data konversi koordiant sesuai dengan *station*nya, sistem memperhitungkan jarak tiap-tiap titik *station* dan menyimpannya ke dalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 7 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data *stationing*, sistem membaca data pada tabel di *form stationing* yang telah dipilih oleh *user* untuk dibandingkan dengan data *stationing* di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data *stationing* yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 8 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data properti jalan, sistem menampilakn *form preparation* dengan *field-field* yang masih kosong, *user* menginputkan data-data properti jalan kemudian sistem menyimpan data properti jalan tersebut kedalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 9 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data properti jalan, sistem membaca data pada tabel di *form* properti jalan yang telah dipilih oleh *user* untuk dibandingkan dengan data properti jalan di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus

oleh sistem dan sistem menampilkan data properti jalan yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 10 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data alignmetn horisontal, sistem membaca data *stationing* dan data properti jalan dari perhitungan sebelumnya, *user* memberikan data dari variabel-variabel yang menjadi masukan, setelah itu sistem memperhitungkan sudut lengkung, elevasi normal dan maksimum jalan kemudian sistem menyimpan data hasil perhitungan tersebut kedalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 11 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data *alignment* horisontal, sistem membaca data pada tabel di *form* horisontal yang telah dipilih oleh *user* untuk dibandingkan dengan data horisontal di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data horisontal yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 12 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data alignmetn horisontal tipe *circle-circle*, sistem membaca data horisontal dari perhitungan sebelumnya, *user* memberikan data dari variabel-variabel yang menjadi masukan, setelah itu sistem memperhitungkannya dan menyimpan data hasil perhitungan tersebut kedalam *storage*. Pengujian validasi

berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 13 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data horisontal *circle-circle*, sistem membaca data pada tabel di *form circle-circle* yang telah dipilih oleh *user* untuk dibandingkan dengan data *circle* di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data *circle* yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 14 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data *alignment* horisontal tipe spiral-*circle*-spiral, sistem membaca data horisontal dari perhitungan sebelumnya, *user* memberikan data dari variabel-variabel masukan, setelah itu sistem memperhitungkannya dan menyimpan data hasil perhitungan tersebut ke dalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 15 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data horisontal spiral-*circle*-spiral, sistem membaca data pada tabel di *form* spiral-*circle*-spiral yang telah dipilih oleh *user* untuk dibandingkan dengan data *circle* di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data spiral-*circle*-spiral yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 16 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data *alignment* horisontal tipe spiral- spiral, sistem membaca data horisontal dari perhitungan sebelumnya, *user* memberikan data dari variabel-variabel masukan, setelah itu sistem memperhitungkannya dan menyimpan data hasil perhitungan tersebut ke dalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 17 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data horisontal spiral-spiral, sistem membaca data pada tabel di *form* spiral-*circle*-spiral yang telah dipilih oleh *user* untuk dibandingkan dengan data *circle* di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data spiral-*circle*-spiral yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 18 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data *alignment* vertikal, sistem membaca data *stationing* dan data properti jalan dari perhitungan sebelumnya, *user* memberikan data dari variabel-variabel masukan, setelah itu sistem memperhitungkannya dan menyimpan data hasil perhitungan tersebut ke dalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 19 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data *alignment* verikal, sistem membaca data pada tabel di *form* vertikal yang telah dipilih oleh *user* untuk dibandingkan dengan data vertikal

di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data vertikal yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 20 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data volume kerja, sistem membaca data *alignment* vertikal dan vertikal dari perhitungan sebelumnya, *user* memberikan data dari variabel-variabel masukan, setelah itu sistem memperhitungkannya dan menyimpan data hasil perhitungan tersebut ke dalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 21 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data volume kerja, sistem membaca data pada tabel di *form* volume yang telah dipilih oleh *user* untuk dibandingkan dengan data volume di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data volume yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 22 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penampilan data hasil-hasil perhitungan ke dalam proyeksi pemetaan dua dimensi, sistem membaca data koordinat, data volume kerja dan data-data perhitungan lainnya, kemudian dari pembacaan data tersebut sistem menampilkannya dalam bentuk titik-titk *waypoint* untuk setiap *station* yang ada, dan menghubungkannya dengan sebuah garis proyeksi. Pengujian validasi berjalan

dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 23 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penampilan laporan-laporan hasil perhitungan, sistem membaca data-data hasil perhitungan dari *storage* dan menampilkannya dalam bentuk tabel dengan *form*at html. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 24 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional pengolahan data *operator*, sistem menampilkan *form operator* dengan *field-field* yang masih kosong, *user* dapat memberikan masukan pada tiap *field* dan menyimpan data hasil perhitungan tersebut ke dalam *storage*. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

Kasus uji 25 merupakan kasus uji yang diberikan untuk menguji kebutuhan fungsional penghapusan data *operator*, sistem membaca data pada tabel di *form operator* yang telah dipilih oleh *user* untuk dibandingkan dengan data *operator* di dalam *storage*, jika terdapat persamaan maka data di dalam *storage* dihapus oleh sistem dan sistem menampilkan data *operator* yang baru pada tabel. Pengujian validasi berjalan dengan lancar dan sistem memberikan hasil pengujian yang valid sesuai dengan yang diharapkan.

6.3. Pengujian akurasi dan presisi sistem

Pengujian ini dilakukan untuk mengetahui tingkat akurasi dan presisi sistem dibandingkan dengan perhitungan secara konvensional, adapun langkah-

langkah pengujian adalah melakukan pengujian awal dengan mengukur akurasi dan presisi dari gps yang dipergunakan, untuk mengukur tingkat presisi dari gps maka peneliti melakukan pengujian sampling dari data gps di suatu titik tertentu sebenyak 8 kali dan diasumsikan tidak terjadi perubahan posisi selama pengambilan sampling. Adapun data yang diperoleh adalah sebagai berikut.

No Uji	Latitude (dd mm.mmmm)	Longitude (ddd mm.mmmm)
1	07 57.3497	112 37.2585
2	07 57.3493	112 37.2582
3	07 57.3503	112 37.2581
4	07 57.3501	112 37.2583
5	07 57.3500	112 37.2582
6	07 57.3500	112 37.2584
7	07 57.3500	112 37.2578
8	07 57.3502	112 37.2579

Diasumsikan bahwa data dengan nilai yang keluar terbanyak adalah data yang presisi maka dapat diketahui bahwa untuk data latitude nilainya adalah 07 57.3500 sedangkan untuk data longitude nilainya adalah 112 37.2582.

No Uji	Latitude (dd mm.mmmm)	Longitude (ddd mm.mmmm)	Perbedaan jarak (dibndkn dgn koordinat asumsi)
1	07 57.3497	112 37.2585	3,6 meter
2	07 57.3493	112 37.2582	1,5 meter
3	07 57.3503	112 37.2581	0,5 meter
4	07 57.3501	112 37.2583	0,2 meter
5	07 57.3500	112 37.2582	0 meter
6	07 57.3500	112 37.2584	3,2 meter
7	07 57.3500	112 37.2578	0 meter
8	07 57.3502	112 37.2579	0,4 meter

Sehingga dari data diatas dapat diperhitungkan tingkat kesalahan pembacaan gps maksimum mencapai 3,6 meter dalam 8 kali pembacaan. Sedangkan minimumnya mencapai 0 meter, yakni mendekati presisi dari koordinat asumsi awal.

Sedangkan untuk mengukur tingkat akurasi gps yang dipergunakan sistem, maka dilakukan kasus uji pengukuran jarak, dengan patokan mempergunakan alat ukur berupa meteran yang dibandingkan dengan hasil pengukuran jarak oleh gps. Pengujian dilakukan pada jarak sepanjang 25 meter, langkah awal adalah dengan mengukur jarak mempergunakan meteran sejauh 25 meter, kemudian dari titik awal diambil koordinat melalui gps, setelah itu baru diambil koordinat pada titik akhirnya, terakhir diperhitungkan jarak tempuh gpsnya. Langkah tersebut diulang sebanyak 5 kali, dan berikut hasil yang diperoleh melalui pengukuran.

No.	Latitude (dd mm.mmmm)	Longitude (dd mm.mmmm)	Jarak gps (meter)	Perbedaan dgn alat ukur meter
1	07 57.3682	112 37.2689	24,198	0,802
	07 57.3601	112 37.2769		
2	07 57.3687	112 37.2701	23,617	1,383
	07 57.3595	112 37.2775		
3	07 57.3685	112 37.2700	23,148	1,852
	07 57.3600	112 37.2770		
4	07 57.3691	112 37.2702	22,279	2,721
	07 57.3601	112 37.2765		
5	07 57.3693	112 37.2705	24,171	0,829
	07 57.3598	112 37.2774		

Dari data pengukuran diatas dan dengan mengasumsikan alat ukur meteran akurat maka dapat disimpulkan bahwa tingkat kesalahan akurasi maksimum dari gps tersebut adalah sejauh 2,721 meter (10,88%) dan tingkat akurasi minimumnya sejauh 0,802 meter (3,20%).

Setelah pengujian akurasi dan presisi alat ukur sistem yang berupa gps dilakukan, kemudian dilakukan pengujian sistem secara keseluruhan dengan melakukan penghitungan jarak sesungguhnya di lapangan, memperhitungkan *alignment-alignment*nya serta perhitungan jarak total. Adapun perhitungan tersebut dijabarkan sebagai berikut.

a. Penghitungan jarak sesungguhnya di lapangan.

Dari hasil perhitungan di lapangan, didapatkan nilai jarak antar *station* sebagai berikut.

Titik Sta	Perhitungan konvensional (meter)	Perhitungan GPS (meter)	Perbedaan (%)
Sta 2	145	142.715	1.575
Sta 3	305	307.093	0.686
Sta 4	75	78.352	4.469
Sta 5	245	243.624	0.561
Sta 6	75	74.432	0.757
Jarak Total	845	846.216	0.143

Tabel 6.2 hasil perrhitungan jarak dilapangan dengan GPS

Dari tabel diatas diketahui bahwa terdapat perbedaan yang terjadi akibat kesalahan pembacaan dari GPS senilai 1.216 meter dengan persentase sebesar 0.143 persen.

b. Perhitungan *alignment-alignment* jalan.

Adapun alignment yang dipergunakan dalam pengujian ini adalah *alignment* horisontal saja karena data masukan sistem tidak mencantumkan data vertikal. Perhitungan a*lignment* horisontal tersebut ditunjukkan sebagai berikut :

i. Sta 2 bertipe horisontal *circle*.

Yang menjadi data masukan perhitungan adalah sebagai berikut :

V = 40 km/jam E maksimum = 0.08

Sudut beta = 11. 7777 E normal jalan = 0.02

R = 130 m Lebar jalan = 4 m

Setelah diketahui data masukan maka akan diperhitungkan nilai Tc, Ec, e, Lc serta Lsnya.

$$Tc = R\,.\ \tan\frac{1}{2}\beta = 130\,.\tan\frac{1}{2}11.7777$$

$$Tc = 13.408\ meter$$

$$Ec = Tc \,.\tan\frac{1}{4}\beta = 13.408 \,.\tan 2.9444$$

$$Ec = 0.689\ meter$$

$$Lc = 0.01745\,.\ \beta\,.\ R = 0.01745\,.\ 11.777\,.\ 130$$

$$Lc = 26.7177\ meter$$

ii. Sta 3 bertipe horisontal *spiral-circle-spiral.*

Yang menjadi data masukan perhitungan adalah sebagai berikut :

V = 40 km/jam	E maksimum = 0.08
Sudut beta = 3.825	E normal jalan = 0.02
R = 110 m	Lebar jalan = 4 m
Ls = 45	E = 0.076

Setelah diketahui data masukan maka akan diperhitungkan nilai θc, θs, Ts, Lc, serta Lnya.

$$\theta s = \frac{Ls\,.\ 90}{\pi\,.R} = \frac{45\,.90}{\pi\,.\ 110}$$

$$\theta s = \ 0.204°$$

$$\theta c = \ \beta - 2\theta s = \ 3.825 - 0.408$$

$$\theta c = \ 3.417$$

$$Lc = \frac{\theta c}{360}\ x\ 2\pi R = \frac{3.417}{360}\ x\ 2.\pi.110$$

$$Lc = 6.556\ meter$$

$$L = Lc + 2Ls = 6.556 + 2.45 = 96.556\ meter$$

$$p = \frac{Ls^2}{6.R} - \ R(1 - \cos\theta s) = \frac{45^2}{6.110} - \ 110(1 - \cos 0.204)$$

$$p = \ 0.775\ m$$

$$k = \ Ls - \frac{Ls^3}{40.R^2} - \ R.\sin\theta s = \ 45 - \frac{45^3}{40.110^2} - \ 110.\sin 0.204$$

$$k = 22.468\ m$$

$$Ts = (R + p).\ \tan\frac{1}{2}\beta + k = (110 + 0.775).\tan\frac{1}{2}3.825 + 22.468$$

$$Ts = 26.167\ meter$$

iii. Sta 5 bertipe horisontal *spiral-spiral.*

Yang menjadi data masukan perhitungan adalah sebagai berikut :

V = 40 km/jam	E maksimum = 0.08
Sudut beta = 0.054	E normal jalan = 0.02
R = 102 m	Lebar jalan = 4 m
Lstabel = 45	E = 0.078

Setelah diketahui data masukan maka akan diperhitungkan nilai θc, θs, Ts, Lc, serta Lnya.

$$\theta s = \frac{1}{2}\beta$$

$$\theta s = 0.027°$$

$$Ls = \frac{\theta s\ .\ \pi\ .R}{90} = \frac{0.027\ .\ \pi\ .102}{90}$$

$$Ls = 0.096\ meter$$

Ls minimum berdasarkan landai relative menurut Bina Marga adalah $m(e + en)B$. Sehingga

$$Lsmin = 100(0.078 + 0.02)4 = 39.2\ meter$$

Berdasarkan ketentua dari Bina Marga nilai Ls yang diambil adalah nilai Ls yang terbesar, dan karena Ls < Lsmin < Ls tabel maka nilai Ls yang dipergunakan adalah ls tabel. Sehingga

$$Ls = Ls\ tabel = 45\ meter$$

$$L = 2Ls = 2.45 = 90\ meter$$

$$p = \frac{Ls^2}{6.R} - R(1 - \cos\theta s) = \frac{45^2}{6.102} - 102(1 - \cos 0.027)$$

$$p = 3.271\ m$$

$$k = Ls - \frac{Ls^3}{40.R^2} - R.\sin\theta s = 45 - \frac{45^3}{40.102^2} - 102.\sin 0.027$$

$$k = 42.022\ m$$

$$Ts = (R + p).\ \tan\frac{1}{2}\beta + k = (102 + 3.271).\tan\frac{1}{2}0.027 + 42.022$$

$$Ts = 44.872\ meter$$

c. Perhitungan jarak total.

Sesuai dengan perhitungan diatas maka untuk data perhitungan lapangan dapat ditentukan sebagai berikut :

Jarak antara titik Sta 1 dengan Sta 2 adalah :

$$L = Interval - Tc + \left(\frac{1}{2}Lc\right) = 145 - 13.408 + (\frac{26.717}{2})$$

$$L = 144.9505\ meter$$

Jarak antara titik Sta 2 dengan Sta 3 adalah :

$$L = Interval - Tc + \left(\frac{1}{2}Lc\right) - Ts + \left(\frac{1}{2}Ls\right)$$

$$L = 305 - 13.408 + \left(\frac{26.717}{2}\right) - 26.167 + (\frac{96.556}{2})$$

$$L = 327.061\ meter$$

Jarak antara titik Sta 3 dengan Sta 4 adalah :

$$L = Interval - Ts + \left(\frac{1}{2}Ls\right)$$

$$L = 75 - 26.167 + (\frac{96.556}{2})$$

$$L = 97.111\ meter$$

Jarak antara titik Sta 4 dengan Sta 5 adalah :

$$L = Interval - Ts + \left(\frac{1}{2}Ls\right)$$

$$L = 245 - 44.872 + (\frac{90}{2})$$

$$L = 245.128\ meter$$

Jarak antara titik Sta 5 dengan Sta 6 adalah :

$$L = Interval - Ts + \left(\frac{1}{2}Ls\right)$$

$$L = 75 - 44.872 + (\frac{90}{2})$$

$$L = 75.128\ meter$$

Dari perhitungan diatas maka dapat dibandingkan antara perhitungan di lapangan dengan sistem, yang ditunjukkan dalam tabel berikut.

Titik Sta	Perhitungan konvensional (meter)	Perhitungan sistem (meter)	Perbedaan (%)
Sta 2	144.95	142.665	1.576
Sta 3	327.061	329.154	0.639
Sta 4	97.111	100.463	3.451
Sta 5	245.128	243.752	0.561
Sta 6	75.128	74.562	0.753
Jarak Total	889.378	892.596	0.361

Tabel 6.3 hasil jarak total perhitungan dilapangan dengan sistem

Dari tabel diatas dapat diketahui bahwa sistem memiliki perbedaan sebesar 3.218 meter dan 0.361% dari perhitungan di lapangan.

BAB VII

PENUTUP

7.1. Kesimpulan

Berdasarkan hasil pengujian dan analisis sitem yang telah dilakukan, baik pengujian per unitnya maupun pengujian validasi dari sistemnya secara keseluruhan diperoleh kesimpulan sebagai berikut :

1. Sistem kerja dari aplikasi dimulai dengan pengambilan titik koordinat dari GPS bluetooth *device* sebagai titik stasioning, berikutnya adalah perhitungan *alignment-alignment* yang mempergunakan variabel-variabel batasan dari dinas bina marga, dari hasil perhitungan *alignment* dapat ditentukan jarak dan bentuk lengkung dari jalan yang direncanakan sehingga dapat ditentukan pula volume pekerjaan baik volume pekerjaan tiap titik stationnya maupun volume kerja keseluruhan.
2. *Interface* yang dipergunakan sistem mempergunakan metode *graphick user interface* yang mempermudahkan pengguna aplikasi dalam mengoperasikannya, *interface* utama dari sistem adalah *form main* menu yang didalamnya terdapat menu-menu untuk menuju ke *form-form* internal perhitungannya, menu utama dibagi menjadi 5 jenis yakni menu *file*, perhitungan, *report, admin* dan bantuan. Menu file dipergunakan untuk login dan logut serta keluar dari program. Menu perhitungan dipergunakan untuk perhitungan *stationing*, *perparation*, koordinat, *alignment* horisontal dan vertikal, serta perhitungan volume kerja. Menu laporan dipergunakan untuk menapilkan laporan-laporan dalam bentuk html dan menampilkan map. Menu

admin digunakan untuk mengolah data *admin*. Menu bantuan dipergunakan untuk menampilkan data tentang pembuat aplikasi. Selain menu-menu tersebut di dalam *main form* juga terdapat *toolbar* yang merupakan tombol *shortcut* untuk menuju *login, logut* dan koneksi.

7.2. Saran

Dalam perencanaan dan pembuatan aplikasi ini terdapat beberapa hal yang bisa dikembangkan untuk kesempurnaan sistem ini. Hal-hal tersebut adalah :

1. Untuk data *streaming* dari koneksi gps dengan sistem masih terdapat *error* yang disebabkan oleh tingkat keakuratan dari gps *device* yang dipergunakan, untuk lebih mendukung akurasi perhitungan sistem maka dapat dipergunakan *device* gps yang memiliki keakuratan hingga tingkat centimeter, ataupun dapat juga direkayasa mengenai error pemabacaan gps *device* sehingga dapat meminimalisir kesalahan pembacaan.
2. Untuk tampilan *mapping* dari hasil perhitungan dapat dipergunakan situs-situs penyedia jasa penampil map yang *open source* dan lebih baik dalam memberikan gambaran mapnya. Dalam aplikasi ini masih mempergunakan situs *openstreetmap* sebagai penyedia jasa penampil mapnya, sehingga tampilannya tampak minimal.