

SISTEM OPTIMASI PENENTUAN RUTE TERDEKAT UNTUK PENGIRIMAN HEWAN KURBAN DENGAN MENGUNAKAN METODE *FLOYD WARSHALL*

(Studi Kasus: Program Kurban LAZISMU)



Disusun Oleh:

N a m a : Dzaky Muhammad T

NIM : 13523080

**PROGRAM STUDI TEKNIK INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA**

2018

HALAMAN PENGESAHAN DOSEN PEMBIMBING

**SISTEM OPTIMASI PENENTUAN RUTE TERDEKAT
UNTUK PENGIRIMAN HEWAN KURBAN DENGAN
MENGUNAKAN METODE *FLOYD WARSHALL***

(Studi Kasus: Program Kurban LAZISMU)



Pembimbing,

(Aridhanyati Arifin, S.T., M.Cs.)

HALAMAN PENGESAHAN DOSEN PENGUJI

SISTEM OPTIMASI PENENTUAN RUTE TERDEKAT UNTUK PENGIRIMAN HEWAN KURBAN DENGAN MENGUNAKAN METODE *FLOYD WARSHALL*

(Studi Kasus: Program Kurban LAZISMU)

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Teknik Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 2 Agustus 2018

Tim Penguji

Aridhanyati Arifin, S.T., M.Cs.

Anggota 1

Taufiq Hidayat, S.T., M.Cs.

Anggota 2

Elyza Gustri, S.T., M.Cs.

Mengetahui,

Ketua Program Studi Teknik Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Dzaky Muhammad T

NIM : 13523080

Tugas akhir dengan judul:

SISTEM OPTIMASI PENENTUAN RUTE TERDEKAT UNTUK PENGIRIMAN HEWAN KURBAN DENGAN MENGUNAKAN METODE *FLOYD WARSHALL*

(Studi Kasus: Program Kurban LAZISMU)

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 2 Agustus 2018

(Dzaky Muhammad T)

HALAMAN PERSEMBAHAN

Tugas akhir ini kupersembahkan untuk keluarga besarku dan teman seperjuanganku

HALAMAN MOTO

Jika kau tak suka sesuatu, maka ubahlah! Jika tak bisa, maka ubahlah cara pandangmu tentangnya! – Maya Angelou

Balas dendam terbaik adalah dengan memperbaiki dirimu – Ali Bin Abi Thalib

Jauhilah dengki, karena dengki memakan amal kebaikan seperti api yang memakan kayu bakar.

Kamu calon konglomerat ya, Kamu harus rajin belajar dan membaca, jangan ditelan sendiri. Berbagilah dengan teman-teman yang tak mendapat pendidikan. – Wiji Thukul

KATA PENGANTAR



Assalamu'alaikum Wr. Wb

Alhamdulillahirobbil'alamiin. Puji dan syukur kehadiran Allah SWT atas limpahan dan rahmatNYA, sehingga penulis dapat menyelesaikan tugas akhir yang berjudul sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban dengan menggunakan metode *Floyd warshall*.

Tugas akhir ini merupakan syarat yang harus ditempuh untuk memperoleh gelar sarjana informatika di Universitas Islam Indonesia. Tugas akhir ini juga merupakan implementasi dan penerapan dari ilmu yang telah didapatkan di bangku kuliah. Laporan tugas akhir ini berisi 6 bab yang merupakan dasar, isi, serta solusi dari permasalahan yang diangkat dalam penelitian ini.

Penulis menyampaikan ucapan terimakasih kepada pihak-pihak yang telah mendoakan, membimbing, dan membantu dalam penyelesaian tugas akhir ini. Ucapan terimakasih diberikan kepada:

1. Bapak Mustafa dan Ibu Siti Nurbaya sebagai orang tua penulis yang selalu sabar dan tidak henti-hentinya memberikan dukungan berupa doa dan materi kepada penulis.
2. Hendrik, S.T., M.Eng. selaku Ketua Program Studi Informatika Universitas Islam Indonesia.
3. Aridhanyati Arifin, S.T., M.Cs. selaku Dosen Pembimbing tugas akhir yang telah mengarahkan dan memberikan bantuan, membimbing dengan penuh kesabaran serta mengajarkan banyak ilmu kepada penulis.
4. Bapak dan Ibu dosen Informatika Universitas Islam Indonesia yang telah mengajarkan ilmunya kepada penulis.
5. Abang Reza dan adikku Nabila, yang membantu memberikan doa dan semangat.
6. Keluarga besar yang memberikan doa dan dukungan.
7. Teman-teman informatika UII 2013 yang selalu *sharing* dan saling membantu dalam berbagai hal selama kuliah.
8. Teman-teman bodoamat traveller yang selalu bersama meniti ilmu sejak awal kuliah, saling membantu dalam suka dan duka, menemani jalan-jalan ke berbagai tempat, membuat kenangan serta saling memberikan dukungan.

9. Teman-teman KKN yang pernah mengisi hariku dan saling membantu dalam mengerjakan program kerja selama satu bulan di desa Jrasah, kabupaten Boyolali.
10. Teman-teman di Balikpapan yang saling memberikan dukungan.
11. Pihak terkait lainnya yang tidak dapat penulis sebutkan satu persatu

Yogyakarta, 2 Agustus 2018

(Dzaky Muhammad T)

SARI

Pada hari raya Idul Adha, umat muslim melakukan penyembelihan hewan kurban. Hewan kurban dapat dikirimkan dari berbagai tempat ke berbagai tempat. Permasalahan rute tempuh seringkali membuat perjalanan menjadi lama dan tidak pasti. Dari banyaknya area yang harus didistribusikan hewan kurban, distributor membutuhkan waktu yang tepat dan kendaraan yang tepat pula. Oleh karena itu, rute terdekat dibutuhkan dalam pengiriman hewan kurban, supaya hewan kurban dapat tiba tepat waktu di tempat tujuan.

Untuk mengatasi permasalahan tersebut, maka dibuatlah sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban dengan metode *Floyd warshall*. Algoritma *Floyd warshall* membantu penentuan rute terdekat dari tempat asal ke tempat tujuan, yang salah satu manfaatnya agar dapat meminimumkan biaya. Untuk membuat sistem ini, dibutuhkan data pendistribusian hewan kurban tahun sebelumnya. Data didapatkan dari Lembaga Amil Zakat Infaq Sedekah Muhammadiyah Yogyakarta Cabang Umbulharjo (LAZISMU). Data tersebut diolah ke dalam sistem sehingga menampilkan keluaran berupa data hewan yang akan didistribusikan beserta rute terdekat dari tempat pengambilan hewan ke tempat tujuan yang divisualisasikan dalam bentuk peta.

Sistem diuji dengan melakukan pengujian *User Acceptance Test (UAT)*, pengujian *black box* dan pengujian keoptimalan sistem. Dari beberapa pengujian tersebut, telah didapatkan hasil bahwa sistem telah optimal karena dapat menampilkan data pengiriman hewan dan peta dengan gambaran rute terdekat dari tempat pengambilan hewan ke tempat tujuan, sehingga pengiriman hewan kurban dapat berjalan baik dan tiba tepat waktu di tempat tujuan.

Kata kunci: hewan kurban, rute terdekat, metode *Floyd warshall*, optimal, peta

GLOSARIUM

Node	titik-titik pada sebuah graf.
Graf	sekumpulan simpul bidang dua dimensi yang saling terhubung dengan sekumpulan garis
Framework	kerangka kerja yang berisi plugin, fungsi serta konsep yang memudahkan pembuat program untuk membuat website
<i>Floyd warshall</i>	algoritma pencarian rute terpendek

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN PENGESAHAN DOSEN PEMBIMBING	ii
HALAMAN PENGESAHAN DOSEN PENGUJI	iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR.....	iv
HALAMAN PERSEMBAHAN	v
HALAMAN MOTO.....	vi
KATA PENGANTAR.....	vii
SARI	ix
GLOSARIUM	x
DAFTAR ISI	xi
DAFTAR TABEL	xiii
DAFTAR GAMBAR.....	xiv
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah	3
1.4 Tujuan	3
1.5 Manfaat	3
1.6 Sistematika Penulisan	4
BAB II TINJAUAN PUSTAKA	5
2.1 Review Penelitian Sejenis.....	5
2.2 Landasan Teori.....	6
2.2.1 Pengertian Sistem dan Karakteristiknya.....	6
2.2.2 Pengertian Optimasi dan Sistem Optimasi	6
2.2.3 Rute Terdekat (<i>Shortest Path</i>)	7
2.2.4 Hewan Kurban dan Hari Raya Idul Adha	7
2.2.5 Google Maps API (<i>Application Programming Interface</i>).....	8
2.2.6 Metode <i>Floyd warshall</i>	8
2.2.7 Framework Laravel	10
2.2.8 Jenis Pengujian	10
BAB III ANALISIS SISTEM	11
3.1 Identifikasi Masalah	11
3.2 Analisis Penyebab Masalah	11
3.3 Alternatif Solusi	11
3.4 Contoh Kasus Algoritma <i>Floyd warshall</i>	12
3.5 Proses Bisnis	18
BAB IV PERANCANGAN SISTEM	20
4.1 Proses Pengambilan Data dengan <i>Google Maps API</i>	20
4.2 Perancangan ERD	21
4.3 Tabel Relasi Basis Data	21
4.4 Perancangan <i>Use Case Diagram</i>	22
4.5 Perancangan <i>Activity Diagram</i>	23
4.5.1 <i>Activity Diagram Login</i>	23
4.5.2 <i>Activity Diagram Register</i>	24
4.5.3 <i>Activity Diagram</i> Kelola Data Pengiriman Hewan	24
4.5.4 <i>Activity Diagram</i> Melihat Data Pengiriman	25
4.5.5 <i>Activity Diagram</i> Melihat Rute Terdekat	25

4.5.6	<i>Activity Diagram</i> Konfirmasi Pengiriman Selesai	26
4.5.7	<i>Activity Diagram</i> Melihat Hasil Pengiriman	26
4.6	Perancangan Antarmuka	27
4.7	Perancangan Teknik Pengujian Perangkat Lunak.....	34
BAB V IMPLEMENTASI DAN PENGUJIAN.....		36
5.1	Implementasi	36
5.2	Pengujian Rute Terpendek	44
5.3	Pengujian Keoptimalan dan Fungsionalitas	61
BAB VI PENUTUP.....		79
6.1	Kesimpulan	79
6.2	Saran.....	79
DAFTAR PUSTAKA		80

DAFTAR TABEL

Tabel 3.1 Hasil tabel matriks dari graf awal	13
Tabel 3.2 Hasil tabel matriks dari graf kedua	15
Tabel 3.3 hasil tabel matriks kedua.....	16
Tabel 3.4 hasil tabel matriks ketiga	16
Tabel 3.5 hasil tabel matriks keempat.....	17
Tabel 3.6 hasil tabel matriks kelima	18
Tabel 4.1 Contoh Tabel Pengujian <i>Black Box</i>	34
Tabel 4.2 Contoh Tabel Pengujian UAT Admin	34
Tabel 4.3 Contoh Tabel Pengujian UAT Kurir.....	35
Tabel 5.1 hasil tabel matriks dari graf awal	46
Tabel 5.2 hasil tabel matriks kedua.....	47
Tabel 5.3 hasil tabel matriks ketiga	47
Tabel 5.4 hasil tabel matriks keempat.....	48
Tabel 5.5 hasil tabel matriks kelima	49
Tabel 5.6 Tabel Pengujian <i>Black box</i>	74
Tabel 5.7 Tabel Pengujian UAT untuk Admin.....	76
Tabel 5.8 Tabel Pengujian UAT untuk Kurir	77

DAFTAR GAMBAR

Gambar 3.1 Graf Awal.....	13
Gambar 3.2 Pembuatan Graf Kedua	14
Gambar 3.3 Perhitungan iterasi pertama.....	15
Gambar 3.4 Perhitungan iterasi kedua	16
Gambar 3.5 Perhitungan iterasi ketiga.....	17
Gambar 3.6 Perhitungan iterasi keempat	17
Gambar 3.7 Urutan Proses Bisnis Sistem	19
Gambar 4.1 Rancangan ERD.....	21
Gambar 4.2 Tabel yang telah direlasikan	22
Gambar 4.3 <i>Use case diagram</i>	22
Gambar 4.4 <i>Activity diagram login</i>	23
Gambar 4.5 <i>Activity diagram</i> daftar akun.....	24
Gambar 4.6 <i>Activity diagram</i> kelola data pengiriman	24
Gambar 4.7 <i>Activity diagram</i> melihat data pengiriman	25
Gambar 4.8 <i>Activity Diagram</i> melihat rute terdekat.....	25
Gambar 4.9 <i>Activity Diagram</i> konfirmasi selesai pengiriman.....	26
Gambar 4.10 <i>Activity Diagram</i> melihat hasil pengiriman	26
Gambar 4.11 Rancangan Antarmuka Register.....	27
Gambar 4.12 Rancangan Antarmuka Login	28
Gambar 4.13 Rancangan Antarmuka Atur Ulang Kata Sandi	28
Gambar 4.14 Rancangan Antarmuka Tambah Data Pengiriman.....	29
Gambar 4.15 Rancangan Antarmuka Melihat Data Pengiriman	29
Gambar 4.16 Rancangan Antarmuka Melihat Data Pengguna	30
Gambar 4.17 Rancangan Antarmuka Melihat Rincian Data Pengiriman Sisi Kurir	30
Gambar 4.18 Rancangan Antarmuka Tabel Aksi pada Data Pengiriman.....	31
Gambar 4.19 Rancangan Antarmuka Melihat Peta.....	32
Gambar 4.20 Rancangan Antarmuka Melihat Rute Terdekat.....	33
Gambar 4.21 Rancangan Antarmuka Tracking Posisi Kurir	33
Gambar 5.1 Antarmuka halaman utama	36
Gambar 5.2 Antarmuka halaman daftar akun	37
Gambar 5.3 Antarmuka halaman <i>login</i>	37
Gambar 5.4 Antarmuka halaman beranda admin	38
Gambar 5.5 Antarmuka halaman tambah data pengiriman	38

Gambar 5.6 Antarmuka halaman <i>maps</i> merepresentasikan peta	39
Gambar 5.7 Antarmuka halaman data pengiriman yang akan dikirimkan	40
Gambar 5.8 Antarmuka halaman <i>tracking</i>	40
Gambar 5.9 Antarmuka halaman yang menunjukkan lokasi saat ini	41
Gambar 5.10 Antarmuka halaman data pengguna	41
Gambar 5.11 Antarmuka halaman data pengiriman	42
Gambar 5.12 Antarmuka halaman ubah jabatan	42
Gambar 5.13 Antarmuka halaman tentang	43
Gambar 5.14 Antarmuka halaman layanan	43
Gambar 5.15 Antarmuka halaman kontak	44
Gambar 5.16 Pembuatan Graf Awal	45
Gambar 5.17 Perhitungan iterasi pertama	46
Gambar 5.18 Perhitungan iterasi kedua	47
Gambar 5.19 Perhitungan iterasi ketiga	48
Gambar 5.20 Perhitungan iterasi keempat	49
Gambar 5.21 Kesimpulan	49
Gambar 5.22 Graf akhir	50
Gambar 5.23 Nilai 1	50
Gambar 5.24 Nilai 2	51
Gambar 5.25 Nilai 3	51
Gambar 5.26 <i>Index array</i>	52
Gambar 5.27 Nilai hasil	52
Gambar 5.28 Rute awal	52
Gambar 5.29 Rute akhir	53
Gambar 5.30 <i>Source code</i> menampilkan canvas peta	53
Gambar 5.31 <i>Source code</i> Geolokasi	54
Gambar 5.32 <i>Source code</i> penandaan titik rute	55
Gambar 5.33 <i>Source code</i> pembentukan vertex	56
Gambar 5.34 <i>Source code</i> perhubungan jalur jalan	56
Gambar 5.35 <i>Source code</i> pengolahan data dengan <i>floyd warshall</i>	58
Gambar 5.36 <i>Source code</i> aturan index <i>floyd warshall</i>	58
Gambar 5.37 <i>Source code</i> perhitungan	59
Gambar 5.38 <i>Source code</i> penyimpanan nilai terkecil ke dalam <i>array</i>	60
Gambar 5.39 <i>Source code</i> pembuatan jalur terdekat	60

Gambar 5.40 <i>Source code</i> perhubungan jalur.....	61
Gambar 5.41 <i>Source code</i> pemberian nilai total jarak.....	61
Gambar 5.42 Penentuan Rute Terdekat	62
Gambar 5.43 Antarmuka halaman <i>output</i> rute terdekat.....	62
Gambar 5.44 Halaman Perkiraan Bahan Bakar	63
Gambar 5.45 <i>Source code</i> perkiraan dan kalkulasi bahan bakar	63
Gambar 5.46 Kesalahan Password.....	65
Gambar 5.47 Kesalahan alamat email	65
Gambar 5.48 Kolom Password kosong.....	66
Gambar 5.49 Pelanggaran hak akses	66
Gambar 5.50 Kolom email kosong	67
Gambar 5.51 Email tidak terdaftar.....	68
Gambar 5.52 Password tidak memiliki lebih dari sama dengan 6 karakter.....	68
Gambar 5.53 Password awal dan password konfirmasi berbeda	69
Gambar 5.54 Akun telah terdaftar	69
Gambar 5.55 Kolom alamat kosong	70
Gambar 5.56 Kolom berat kosong	71
Gambar 5.57 Kolom jumlah jamaah kosong	71
Gambar 5.58 Tidak memberikan titik koordinat.....	72
Gambar 5.59 Lokasi tidak dapat terbaca.....	72
Gambar 5.60 Berhasil membaca lokasi	73
Gambar 5.61 Hapus akun staff.....	73

BAB I

PENDAHULUAN

1.1 Latar Belakang

Hari raya Idul Adha adalah puncak pelaksanaan ibadah Haji bagi umat muslim dan pada hari yang sama, umat muslim melakukan penyembelihan hewan kurban. Penyembelihan hewan kurban dilakukan untuk mengingatkan umat muslim akan perintah Allah kepada Nabi Ibrahim yang menyembelih domba sebagai pengganti putranya, yaitu Ismail. Dari sejarah itu, penyembelihan hewan menjadi tradisi lazim dilakukan umat muslim di dunia untuk berkorban atau mengorbankan hewan yang menjadi sebagian hartanya untuk kegiatan sosial (Islam Cendekia, 2014).

Menurut data yang dihimpun oleh Global Aksi Cepat Tanggap (ACT), 90% hewan kurban masih didistribusikan di daerah perkotaan (sekitar tempat tinggal pekurban yang relatif sudah mampu). Penyaluran hewan kurban kebanyakan hanya fokus pada daerah-daerah masjid dan sekitar tempat pemotongan hewan kurban. Pendistribusian hewan kurban belum banyak menyentuh korban bencana dan daerah-daerah miskin lainnya (Aksi Cepat Tanggap, 2015). Hal tersebut juga dikarenakan kurangnya pemberdayaan peternak lokal terdekat dari daerah penyaluran. Dengan memberdayakan peternak lokal terdekat dengan daerah penyaluran, maka dapat mengefektifkan pemerataan penyaluran hewan kurban. Pemerataan yang dimaksud adalah penyaluran hewan kurban yang tepat sasaran dan adil dibagikan kepada daerah yang membutuhkan.

Lembaga Amil Zakat, Infaq, Shodaqoh Muhammadiyah atau biasa disingkat Lazismu memiliki banyak program layanan masyarakat, salah satunya adalah program pendistribusian hewan kurban yang dilaksanakan tiap tahun. Dalam pendistribusian hewan kurban, Lazismu ingin menghindari resiko keterlambatan pengiriman hewan kurban ke daerah tujuan pendistribusian. Salah satu faktor yang dapat membuat keterlambatan dalam pengiriman hewan kurban adalah jarak yang tidak diperkirakan antara tempat peternakan atau tempat pengambilan hewan kurban menuju lokasi daerah tujuan pendistribusian hewan kurban. Jika jarak dari tempat pengambilan hewan ke tempat pendistribusian tidak dapat diperkirakan, maka dapat terjadi keterlambatan pengiriman hewan kurban.

Lazismu memiliki dua tempat berlangganan dalam membeli atau menitipkan hewan kurban untuk dirawat sebelum dikurbankan, yaitu pada peternakan Angkatan Muda Muhammadiyah (AMM) Glagah dan AMM Kotagede. Lazismu membutuhkan rute

pengiriman yang dekat atau rute terdekat dari tempat pengambilan hewan menuju ke daerah tujuan pendistribusian, supaya hewan kurban dapat tiba tepat waktu kepada penerima sehingga penerima dapat melakukan kurban dengan baik. Jika salah menentukan rute, maka hewan kurban belum tentu dapat tiba tepat waktu ke desa tujuan.

Berdasarkan hasil wawancara kepada Pimpinan Lazismu unit layanan Umbulharjo Yogyakarta, cakupan area yang didistribusikan hewan kurban oleh Lazismu tidak pasti dan dapat berubah tiap tahunnya, dan kadang berbeda dari tahun sebelumnya, karena daerah yang ditentukan adalah tergantung yang mana yang lebih membutuhkan. Meskipun Lazismu memiliki dua tempat peternakan hewan kurban langganan tiap tahun, tetapi lokasi peternakan tersebut belum tentu selalu dekat dari daerah – daerah tujuan pendistribusiannya. Jika tempat peternakannya jauh dari tempat pendistribusiannya, maka membuat waktu dan biaya terpakai lebih banyak.

Untuk menentukan rute terdekat, selama ini kurir pengantar hewan kurban bersama staf Lazismu hanya berdiskusi mengenai rute tersebut untuk pengambilan keputusannya. Distributor atau kurir pengantar hewan kurban juga kadang tidak mengetahui bagaimana kondisi perjalanan di lapangan, sehingga harus memilih rute terdekat dari tempat pengambilan hewan ke daerah tujuan supaya kendala jalan yang rusak atau macet tidak membuat hewan kurban tiba dengan terlambat kepada penerima. Hal ini tentu saja tidak efektif karena pengambilan keputusan hanya berdasarkan hasil diskusi.

Berdasarkan masalah di atas, maka diperlukan upaya untuk mengoptimalkan pengiriman hewan kurban yaitu dengan cara mengoptimalkan penentuan rute terdekat dari tempat pengambilan hewan menuju ke daerah tujuan pendistribusian serta dapat mengetahui perkiraan bahan bakar dan transportasi yang akan digunakan dengan tepat. Sistem ini akan menampilkan fitur yang dapat digunakan untuk menentukan jarak antara daerah tujuan pendistribusian dan tempat pengambilan hewan.

Metode yang digunakan untuk penentuan rute terdekat menggunakan metode perhitungan *Floyd warshall*. Metode *Floyd warshall* menghitung bobot terkecil dari semua jalur yang menghubungkan sebuah pasangan titik, dan melakukannya sekaligus untuk semua pasangan titik. Pengiriman hewan kurban tidak hanya dikirimkan ke satu tempat saja, tapi juga ke berbagai tempat. Oleh karena itu, metode ini merupakan metode yang tepat digunakan untuk penelitian ini.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya, maka rumusan masalahnya yaitu:

- a. Bagaimana merancang sebuah sistem yang memudahkan pengguna untuk melakukan pemantauan pada pengantaran hewan kurban dengan menggunakan peta?
- b. Bagaimana menentukan rute terdekat pengiriman hewan kurban ke setiap tujuan pendistribusian secara optimal?

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini, yaitu:

- a. Rekomendasi rute terdekat minimal satu rute dalam satu area tujuan pengiriman.
- b. Rute pengiriman hewan kurban dikatakan optimal jika jarak yang ditempuh adalah jarak terdekat.
- c. Pengiriman hewan kurban dikatakan optimal jika biaya transportasi yang dikeluarkan minimal.
- d. Pemantauan/ *tracking* pengiriman dikatakan optimal ketika sesuai dengan posisi sebenarnya orang yang dipantau.
- e. Hanya akun yang terdaftar sebagai kurir yang dapat menggunakan sistem *Floyd warshall*.
- f. Hanya akun yang terdaftar sebagai admin yang dapat melakukan *tracking*.

1.4 Tujuan

Tujuan dari penelitian yang dilakukan ini adalah membuat sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban dari tempat pengambilan hewan menuju lokasi pendistribusian, serta rekomendasi transportasi dan perkiraan total bahan bakar yang digunakan.

1.5 Manfaat

Beberapa manfaat dari penelitian ini adalah:

- a. Membantu kurir pengantar untuk menentukan rute terdekat dalam proses pengiriman hewan kurban ke tempat pendistribusian.
- b. Membantu meminimalisir biaya bahan bakar dan meminimalisir waktu dalam proses pengiriman hewan kurban ke tempat pendistribusian.
- c. Memudahkan pengguna dalam memilih transportasi yang harus digunakan untuk mengantarkan hewan kurban berdasarkan jumlahnya.

1.6 Sistematika Penulisan

Sistematika penulisan bertujuan untuk memudahkan dalam memahami laporan penelitian. Secara garis besar, sistematika laporan ini dibagi menjadi 6 bab. Berikut sistematika penulisan dalam laporan ini:

BAB I PENDAHULUAN

Merupakan informasi yang menjadi dasar mengenai penelitian ini. Pada bab ini, diuraikan latar belakang mengapa penentuan rute terdekat pengiriman hewan kurban perlu dilakukan, beserta rumusan masalah, batasan masalah, tujuan, manfaat, metodologi penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Berisi informasi serta teori mengenai sistem, optimasi, rute terdekat, pengiriman hewan kurban, *Google Maps API* dan metode *Floyd warshall*.

BAB III ANALISIS SISTEM

Berisi tahapan-tahapan penyelesaian masalah, contoh kasus dengan perhitungan menggunakan metode *Floyd warshall*, dan proses bisnis sistem.

BAB IV PERANCANGAN SISTEM

Memuat hasil rancangan sistem berupa *use case diagram* sistem optimasi penentuan rute terdekat pengiriman hewan kurban, *activity diagram* berdasarkan *use case*, relasi tabel basis data, dan rancangan antarmuka sistem yang telah dirancang sesuai kebutuhan.

BAB V IMPLEMENTASI DAN PENGUJIAN

Berisi *screenshot* hasil implementasi dari sistem yang telah dibuat, *screenshot* kode program inti, hasil pengujian sistem yang berupa pengujian *black box*, *user acceptance test*, dan pengujian keoptimalan sistem serta pembahasan penggunaan metode *Floyd warshall* pada sistem.

BAB VI PENUTUP

Merupakan uraian kesimpulan dari proses dan hasil penelitian, uraian kelebihan dan kekurangan sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban yang telah dibuat dan saran yang dibuat berdasarkan penelitian yang telah dilakukan.

BAB II

TINJAUAN PUSTAKA

2.1 Review Penelitian Sejenis

Review penelitian sejenis bertujuan untuk mengetahui, mempelajari dan menjadikan kekurangan pada penelitian sebelumnya untuk menjadi kelebihan pada penelitian yang dilakukan saat ini. Adapun penelitian yang sejenis dengan penelitian ini ialah pada penelitian yang dilakukan oleh Ichsan Indra Wahyudi pada tahun 2016. Konsep dari penelitian tersebut adalah pencarian rute terdekat untuk pengiriman barang menggunakan algoritma *Dijkstra*. Algoritma *Dijkstra* bertujuan untuk menemukan jalur terdekat berdasarkan bobot terkecil dari satu titik ke titik lainnya. Hasil dari penelitian tersebut adalah sistem yang memiliki fitur pengelolaan data konsumen dan kurir pengantar, ada pula fitur yang ditujukan untuk konsumen agar memudahkan dalam *trace and tracking* barang, dan untuk kurir pengantar barang agar kurir mengetahui data pengiriman barang dan rute terdekat menuju lokasi pengiriman barang (Wahyudi, 2016).

Terdapat pula penelitian sejenis lainnya yang dilakukan oleh Ahyar Rivai Hasibuan pada tahun 2016. Penelitian tersebut membahas tentang sistem pencarian rute terdekat dalam pengiriman barang, dengan menggunakan algoritma *Floyd warshall*. Pada sistem tersebut, pengguna hanya perlu menginputkan nomor pengirim, tujuan, ID mobil dan ID supir pada form data barang. Setelah itu, pengguna perlu memasukkan asal dan tujuan pada *form* rute, kemudian sistemnya akan menampilkan rute terdekatnya dalam bentuk graf (Hasibuan, 2016).

Judul penelitian ini adalah sistem optimasi penentuan rute terdekat pengiriman hewan kurban. Penelitian ini dilakukan dengan menggunakan metode *Floyd warshall*. Algoritma *Floyd warshall* menghitung bobot terkecil dari semua jalur yang menghubungkan sebuah pasangan titik, dan melakukannya sekaligus untuk semua pasangan titik. Algoritma *Floyd warshall* lebih menjamin keberhasilan penemuan solusi optimum untuk penentuan rute terdekat, sedangkan algoritma *Dijkstra* tidak selalu berhasil memberikan solusi optimum. Solusi optimum jelas dibutuhkan untuk sistem ini karena solusi tersebut adalah dasar untuk kurir pengantar mengambil keputusan dengan cepat dan tepat sehingga kurir pengantar tidak salah memilih rute yang akhirnya dapat mengakibatkan keterlambatan atau terhalangnya pengiriman hewan kurban. Hasil pada kedua penelitian tersebut, sistemnya tidak memiliki *output* anggaran biaya perjalanan atau biaya bahan bakar serta jenis kendaraannya, dan tidak

menampilkan waktu tempuh dan jarak tempuhnya. Algoritma *Floyd warshall* membantu penentuan rute terdekat yang salah satu manfaatnya agar dapat meminimumkan biaya, maka anggaran biaya perjalanan atau biaya bahan bakarnya lebih baik ditampilkan pada sistem. Fitur tersebut seperti yang akan ada pada hasil penelitian ini yaitu sistem optimasi penentuan rute terdekat pengiriman hewan kurban, yang akan menampilkan *output* anggaran biaya bahan bakar untuk sampai ke tempat tujuan, beserta jarak tempuh dan waktu tempuhnya. Sebelum sistem menampilkan output anggaran biaya bahan bakar, sistem akan menampilkan fitur yang dimana pengguna harus memasukkan jenis kendaraannya, setelah itu akan keluar kapasitas muatannya dan jenis bahan bakarnya. Pada penelitian ini, rute terdekat yang dihasilkan akan ditampilkan dalam bentuk *map*. Kemudian dalam pengiriman hewan kurban, hewan kurban harus tiba tepat waktu ke daerah tujuan, jika mengalami keterlambatan maka daerah tujuan distribusi hewan kurban tersebut tidak dapat melakukan kurban dengan baik. Oleh karena itu, untuk mengantisipasinya, maka sistem ini tidak hanya menampilkan rute terdekatnya, namun jarak tempuh serta waktu tempuhnya.

2.2 Landasan Teori

2.2.1 Pengertian Sistem dan Karakteristiknya

Sistem adalah kumpulan elemen atau komponen yang saling berkaitan untuk mencapai suatu tujuan tertentu. Sistem juga merupakan kesatuan bagian-bagian saling berkaitan yang berada dalam satu wilayah serta memiliki penggerak. Sistem adalah sekelompok elemen yang terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan (McLeod, 2001). Sistem memiliki beberapa karakteristik, yakni:

- a. Komponen Sistem
- b. Batasan Sistem
- c. Lingkungan Luar Sistem
- d. Penghubung Sistem
- e. Masukan Sistem
- f. Keluaran Sistem
- g. Pengolah Sistem

2.2.2 Pengertian Optimasi dan Sistem Optimasi

Optimasi biasa diartikan dengan suatu cara untuk mencapai hasil yang optimal, menguntungkan dan efektif. Dalam sebagian masalah *engineering*, banyak terdapat berbagai

alternatif pemecahan masalah. Proses penentuan penyelesaian masalah yang baik dinamakan optimasi.

Masalah optimasi sering kita jumpai dalam kehidupan sehari-hari. Proses penyelesaian masalah optimasi dengan terkomputerisasi dinamakan sistem optimasi. Sistem optimasi memudahkan dalam penyelesaian masalah keterbatasan yang dinyatakan dalam kuantitas tertentu dengan menggunakan sistem. Terdapat tiga komponen dalam persoalan optimasi, yaitu fungsi tujuan, variabel keputusan, dan kendala (Komaruddin, 2014).

2.2.3 Rute Terdekat (*Shortest Path*)

Shortest Path dapat diartikan lintasan terdekat. Pencarian lintasan/rute terdekat adalah bertujuan menghasilkan *cost* yang minimum. Persoalan lintasan terdekat yaitu menemukan lintasan terdekat antara dua atau beberapa simpul lebih yang berhubungan. Persoalan lintasan terdekat bisa direpresentasikan dalam bentuk graf yang memiliki bobot.

Dalam penyelesaian permasalahan rute terdekat dapat diselesaikan dengan beberapa algoritma yakni algoritma dijkstra, algoritma *Floyd warshall* dan lain-lain. Rute terdekat yang terkomputerisasi dapat memberi kemudahan dalam menentukan permasalahan rute terdekat (Apriadi, 2018).

2.2.4 Hewan Kurban dan Hari Raya Idul Adha

Hewan kurban adalah hewan tertentu yang disembelih oleh manusia (umat muslim) dengan tujuan mendekatkan diri kepada Allah. Penyembelihan hewan kurban dilaksanakan pada saat hari raya Idul Adha. Hewan kurban yang telah disembelih, dagingnya akan dibagikan kepada seluruh umat islam khususnya yang kurang mampu. Hukum berkurban adalah sunnah muakkadah yang berarti sunnah yang sangat dianjurkan. Sebelum melakukan hari raya kurban, terdapat daerah-daerah yang membutuhkan pendistribusian hewan kurban karena minimnya jumlah hewan kurban yang akan dibagikan. Oleh karena itu, berbagai lembaga ingin mengirimkan hewan kurban ke daerah yang membutuhkan.

Dalam pengiriman hewan kurban, terdapat syarat-syarat pengiriman agar hewan tetap sehat. Adapun syarat-syaratnya antara lain tidak mengangkut hewan dengan berdempetan karena hewan ternak bisa stres. Oleh karena itu, dalam pengangkutan hewan ternak biasanya dikirimkan menggunakan mobil *pick up* atau truk. Contoh hewan ternaknya ialah sapi. Pada umumnya, sapi di Indonesia dijual dari yang beratnya 220 kg hingga 1 ton. Kemudian, batas maksimal muatan pada truk ialah 14 ton, dan batas maksimal muatan pada *pick up* adalah 2 ton. Oleh karena itu, jika dihitung dengan rata-rata berat persapi mencapai 1 ton, maka truk

dapat mengangkut 8 sampai 10 sapi, sedangkan pick up hanya dapat mengangkut 2 sampai 4 sapi (Kargo, 2018).

2.2.5 Google Maps API (*Application Programming Interface*)

Google Maps API adalah *interface* yang digunakan *programmer* untuk berinteraksi dengan *library* milik *Google maps* guna mendapatkan data maupun pemetaan yang dilakukan secara *online*. *Google Maps API* menyediakan fungsi yang sangat banyak yang dan pembagiannya adalah *Google maps Javascript API v3*, *Google Maps API Webservice*, *Google Place API*. Sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban ini menggunakan *Google Maps API* dengan *driving mode*.

Google Maps selalu menunjukkan direksi yang memungkinkan dapat dilalui *driving mode*. Adapun mode-mode perjalanan yang disediakan *google maps*, diantaranya adalah *driving mode*, *bus transit*, *walking mode*, *cycling mode*, dan lain-lain. Dalam sistem ini, sistem menggunakan *driving mode* untuk mendapatkan data *feedback* yang spesifik yang sesuai dengan *driving mode*. *Driving mode* tidak memperhitungkan kendaraan roda dua, empat, atau enam. *Driving mode* juga tidak memperhitungkan jalur bus atau gang. *Driving mode* mengeksekusi *default* nilai jalur cepat saja. Kemudian dalam pemetaannya, *driving mode* menampilkan peta jalan raya, jalan sempit, jalan gang, dan lain-lain. *Google Maps* mengambil setiap belokan secara default (Google Developer, 2015).

2.2.6 Metode *Floyd warshall*

Floyd warshall adalah salah satu metode untuk memecahkan kasus permasalahan pencarian rute terdekat. Kelebihan penggunaan metode *Floyd warshall* dibandingkan dengan salah satu metode lainnya yaitu metode *Dijkstra* ialah dalam memecahkan masalah, metode *Floyd warshall* memandang solusi yang akan diperoleh sebagai keputusan yang terkait. Artinya, metode *Floyd warshall* memandang resiko dari solusi yang akan diambil dan lebih menjamin keberhasilan penemuan solusi optimum untuk kasus penentuan lintasan terdekat, sehingga solusi yang dihasilkan adalah solusi terbaik (Achmad Arsori, 2013).

Algoritma *Floyd warshall* adalah salah satu varian dari pemrograman dinamis, yaitu suatu metode yang melakukan pemecahan dengan memandang solusi sebagai keputusan yang saling terkait. Solusi tersebut dibentuk dari solusi yang berasal dari tahap sebelumnya dan ada kemungkinan solusi lebih dari satu. Dalam usaha untuk mencari rute terdekat, algoritma *Floyd warshall* memulai iterasi dari titik awalnya kemudian memperpanjang *path* dengan

mengevaluasi titik demi titik hingga mencapai tujuan dengan jumlah bobot yang seminimum mungkin (Siang, 2011).

Floyd warshall adalah sebuah algoritma matematis yang mengalkulasikan beberapa *node node* dari satu atau lebih jalur yang tersedia untuk mendapatkan jalur terdekat yang efisien dengan menghitung jarak dari *node* pertama hingga ke *node* terakhir ke semua *node* yang ada secara satu persatu tanpa mengabaikan *node* yang lain hingga di dapatkan nilai jumlah tiap *node* yang sangat rendah. Kalkulasi ini dilakukan dengan menghitung total penjumlahan dan jika didapatkan nilai yang lebih rendah dari nilai sebenarnya, kemudian akan digantikan dengan nilai penjumlahan terendah tiap node dari iterasi hingga di iterasi tersebut tidak lagi terjadi perubahan nilai terendah. Adapun algoritma *Floyd warshall* sebagai berikut.

1. $R = R_1$
2. r^n
3. Untuk $k = 1$ hingga n , lakukan :
 Untuk $i = 1$ hingga n , lakukan :
 Untuk $j = 1$ hingga n , lakukan :
 Jika $R[i,j] > R[i,k] + R[k,j]$ maka tukar $R[i,j]$ dengan $R[i,k] + R[k,j]$.
4. $R^* = R$

Keterangan :

R_1 = matriks keterhubungan graf berarah berbobot awal

R^* = matriks keterhubungan minimal

$R_{i,j}$ = nilai dari titik v_i ke v_j

Penjelasan :

- a. Banyaknya iterasi (R^n) tergantung dari banyaknya *node* yang akan dihitung
- b. Selanjutnya akan dilakukan penjumlahan dari *node* ke *node* dimulai dari iterasi pertama
- c. Iterasi pertama menjumlahkan tiap *node* pada semua indeks baris pertama
- d. Jika didapatkan nilai sebenarnya pada suatu indeks lebih besar dari nilai hasil penjumlahan di *node* indeks tersebut, maka nilai hasil penjumlahan akan disimpan dan nilai sebenarnya akan diganti
- e. Setelah semua indeks dalam satu baris selesai dihitung, kemudian akan dilanjutkan ke indeks baris selanjutnya
- f. Lakukan terus hingga sejumlah iterasi dari sejumlah *node* telah dihitung

- g. Iterasi akhir akan menunjukkan hasil yang tetap dengan nilai terendah dari penjumlahan tiap-tiap *node*.

2.2.7 Framework Laravel

Laravel adalah sebuah *framework* PHP yang bersifat *open source*. Laravel ditulis oleh Taylor Otwell dengan lisensi di bawah *MIT Lisence*. Dengan menggunakan laravel, dapat memudahkan penggunanya untuk menuliskan sintaks dengan lebih sederhana namun tetap elegan dan fungsional. Laravel memudahkan dalam pembuatan kode program untuk pembuatan situs web. Laravel menyediakan langsung *routing*, *session*, dan *caching*. Adapun kebutuhan dalam instalasinya adalah *text editor*, *webserver* dan *database*, dan *composer*.

Dengan menggunakan framework, khususnya Laravel, maka akan mengenal yang namanya *Model*, *View*, *Controller* (*MVC*). *MVC* adalah sebuah metode pengembangan aplikasi yang memisahkan data dari tampilan dan *logic* dari aplikasi yang akan dibuat. *Model* digunakan untuk proses query data ke/dari *database*. *Controller* merupakan kunci logika untuk mengatur alur program dari sebuah web dan menjadi jembatan antara *view* dan *model*. Sedangkan *view* adalah tampilan/*frontend* dari sebuah web. (Nugraha, 2015)

2.2.8 Jenis Pengujian

Terdapat beberapa jenis pengujian pada perangkat lunak. Pengujian dilakukan untuk melihat dan menilai kelayakan sistem. Dari beberapa pengujian yang dilakukan, akan diketahui jaminan kualitas perangkat lunak. Penilaian dapat dilihat dari fungsionalitas perangkat lunak, desain perangkat lunak, *error handling* dan *fix bugging*, dan penilaian lainnya pada perangkat lunak. Adapun jenis pengujian perangkat lunak yang digunakan dalam penelitian ini antara lain:

a. Black box testing

Black box testing adalah melakukan pengujian berdasarkan apa yang dilihat, terfokus dengan fungsionalitas sistem serta outputnya. Kemudian, yang diuji lebih ditujukan pada desain software dan *bugging*.

b. User Acceptance Test (UAT)

UAT adalah pengujian yang dilakukan oleh pengguna meliputi kepuasan atas kelayakan sistem untuk menghasilkan dokumen yang kemudian akan dijadikan bukti bahwa sistem telah diterima oleh pengguna.

BAB III

ANALISIS SISTEM

3.1 Identifikasi Masalah

Algoritma *Floyd warshall* dapat diterapkan pada sebuah aplikasi pencari rute terdekat dari satu daerah ke daerah lainnya. Terdapat beberapa proses untuk menentukan rute terdekat dengan menggunakan algoritma *Floyd warshall*. Dalam pengiriman hewan kurban yang dilakukan oleh Lazismu, terdapat beberapa proses yang terjadi dalam proses pengirimannya.

Prosedur awalnya adalah staf Lazismu melakukan survey untuk menemukan daerah yang membutuhkan penyaluran hewan kurban. Lazismu melakukan rapat untuk mendiskusikan tiga daerah yang layak menerima penyaluran hewan kurban tersebut. Dari tiga pilihan daerah tersebut akan diseleksi untuk menemukan satu daerah yang paling layak. Dua daerah lainnya sebagai cadangan jika daerah pertama ternyata mendadak mendapat kiriman hewan kurban dari tempat lain.

Pada satu hari sebelum perayaan hari raya idul adha, tepat pada malam harinya Lazismu mulai mengirimkan hewan kurban ke daerah yang telah ditentukan dengan pasti. Sebelum pengiriman hewan, Lazismu berdiskusi mengenai rute menuju ke daerah pendistribusian dan berdiskusi mengenai penentuan muatan dan angkutannya. Dari hasil diskusi tersebut, kurir mengantarkan hewan kurban ke daerah pendistribusian. Dari hasil diskusi tersebut juga didapatkan keputusan mengenai pembelian bahan bakar kendaraan yang digunakan untuk mengantarkan hewan kurban.

3.2 Analisis Penyebab Masalah

Beberapa permasalahan yang didapatkan adalah sebagai berikut:

- a. Efisiensi waktu yang dibutuhkan kurir untuk mengantarkan hewan kurban.
- b. Pengeluaran biaya bahan bakar kendaraan yang tidak dapat dihitung dengan pasti.
- c. Staf dan kurir yang kurang mengetahui rute terdekat menuju tempat pendistribusian.
- d. Penentuan muatan dan angkutan yang tidak dapat dipastikan dengan cepat.

3.3 Alternatif Solusi

Penelitian ini memberikan sebuah solusi yaitu sistem optimasi penentuan rute terdekat dalam pengiriman hewan kurban. Sistem ini tidak hanya menentukan rute terdekat, namun

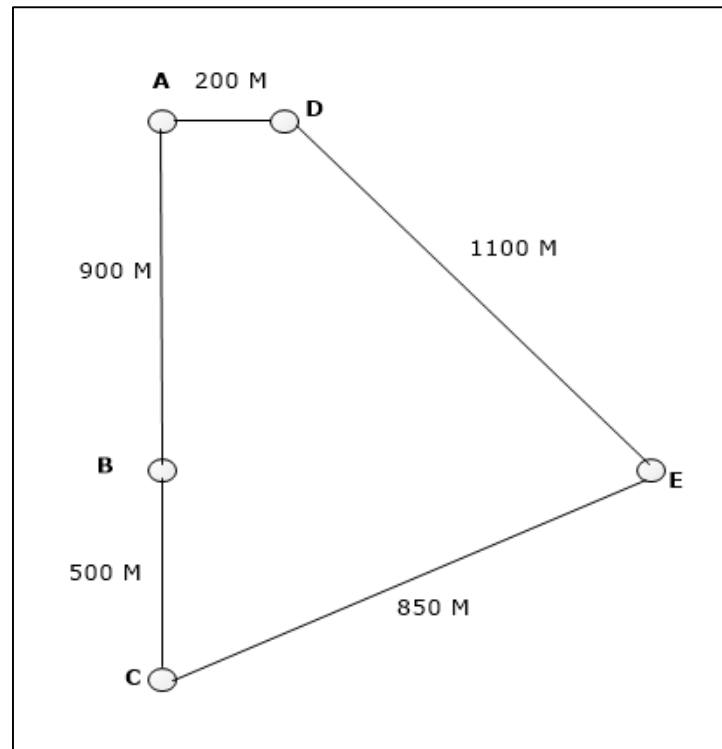
memberikan solusi kendaraan yang digunakan berdasarkan muatannya, serta solusi penentuan bahan bakar kendaraan yang pasti. Dalam penentuan rute terdekatnya, sistem ini menggunakan algoritma *Floyd warshall*. Dengan algoritma ini, hasil yang didapatkan bisa lebih optimum, namun memerlukan *resource* yang besar jika dipakai untuk pencarian yang kompleks. Pencarian rute dikatakan optimum jika telah didapatkan nilai jarak terendahnya.

Dalam pengiriman hewan kurban, terdapat syarat-syarat pengiriman agar hewan tetap sehat. Adapun syarat-syaratnya antara lain tidak mengangkut hewan dengan berdempetan karena hewan ternak bisa stress. Oleh karena itu, dalam pengangkutan hewan ternak biasanya dikirimkan menggunakan mobil *pick up* atau truk. Contoh hewan ternaknya ialah sapi. Pada umumnya, sapi di Indonesia dijual dari yang beratnya 220 kg hingga 1 ton. Kemudian, batas maksimal muatan pada truk ialah 14 ton, dan batas maksimal muatan pada *pick up* adalah 2 ton. Sehingga, jika dihitung dengan rata-rata berat persapi mencapai 1 ton, maka truk dapat mengangkut 8 sampai 10 sapi, sedangkan *pick up* hanya dapat mengangkut 2 sampai 4 sapi. Adapun bahan bakar yang digunakan untuk transportasi pengiriman akan dihitung berdasarkan jarak daerah yang ditempuh dan harga bahan bakar khusus daerah DIY saat ini.

3.4 Contoh Kasus Algoritma *Floyd warshall*

Diketahui tempat peternakan langganan Lazismu ialah AMM Glagah dan AMM Kotagede. Berikut ini adalah perhitungan untuk pencarian rute terdekat dari tempat peternakan langganan Lazismu yaitu AMM Glagah dan AMM Kotagede menggunakan algoritma *Floyd warshall*.

- a. Langkah pertama ialah membuat graf yang akan dijadikan sebuah tabel atau matriks, kemudian mengambil setiap nilai jarak tiap titik yang telah terhubung. Berikut ini adalah penggambaran graf awal sebelum ditemukannya rute terdekat.



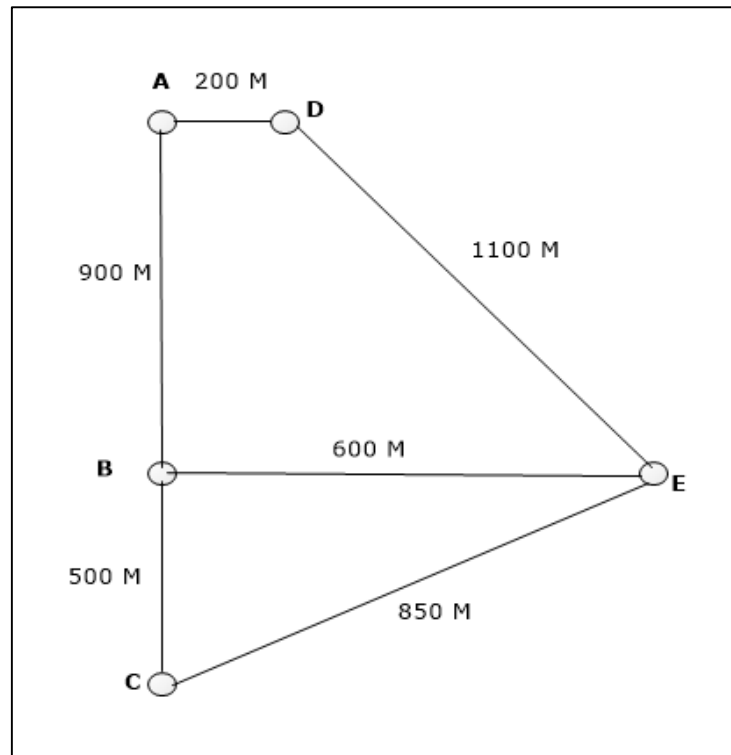
Gambar 3.1 Graf Awal

- b. Nilai dimasukkan ke dalam *array* sementara yang selanjutnya akan seperti pada Tabel 3.1.

Tabel 3.1 Hasil tabel matriks dari graf awal

Dari, Ke	A	B	C	D	E
A	0	900	∞	200	∞
B	900	0	500	∞	∞
C	∞	500	0	∞	850
D	200	∞	∞	0	1100
E	∞	∞	850	1100	0

- c. Untuk setiap R atau titik yang bernilai tak terhingga akan dilakukan pengecekan keterhubungannya dengan titik yang lain apakah masih bisa terhubung atau tidak, dan jika terhubung, maka ganti nilai tak terhingga dengan nilai sebenarnya.



Gambar 3.2 Pembuatan Graf Kedua

- d. Jika semua nilai tak terhingga telah dilakukan pengecekan, maka perhitungan *floyd warshall* dapat dilakukan.

Algoritma:

1. $R = R_1$
2. r^n
3. Untuk $k = 1$ hingga n , lakukan :
 Untuk $i = 1$ hingga n , lakukan :
 Untuk $j = 1$ hingga n , lakukan :
 Jika $R[i,j] > R[i,k] + R[k,j]$ maka tukar $R[i,j]$ dengan $R[i,k] + R[k,j]$.
4. $R^* = R$

Keterangan :

R_1 = matriks keterhubungan graf berarah berbobot awal

R^* = matriks keterhubungan minimal

$R_{i,j}$ = nilai dari titik v_i ke v_j

- e. Berdasarkan graf di atas, maka selanjutnya didapatkan hasil tabel matriks dalam penentuan rute terdekat. Berikut ini ialah tabelnya.

Tabel 3.2 Hasil tabel matriks dari graf kedua

Dari, Ke	A	B	C	D	E
A	0	900	∞	200	∞
B	900	0	500	∞	600
C	∞	500	0	∞	850
D	200	∞	∞	0	1100
E	∞	600	850	1100	0

Huruf A sampai E pada bagian yang ke bawah merupakan titik awal. Huruf A sampai E pada bagian atas ialah titik tujuan. Jika dalam tabel tersebut terdapat lima buah titik yaitu A, B, C, D, E, maka akan ada enam proses yang akan dilewati yaitu R0, R1, R2, R3, R4, dan R5 sebagai hasil akhirnya. Hasil dari sebuah proses akan digunakan untuk proses berikutnya.

R = iterasi

r = indeks dari node A, B, C, D, E

R1, R2, R3, R4, dimana R4= hasil akhir dan R5=kesimpulan

- f. Selanjutnya didapatkan hasil perhitungan berdasarkan rumus. Berikut ini ialah perhitungan iterasi pertama.

R1:			
Ar = 0	Br = 900	Cr = ∞	Dr = 200
<u>A</u> (A) = 0	B(A) = 900	C(A) = ∞	D(A) = 200
0 + 0 = 0	900 + 0 = 900	∞ + 0 = ∞	200 + 0 = 200
<u>A</u> (B) = 900	B(B) = 0	C(B) = 500	D(B) = ∞ diganti 1100
0 + 900 = 900	900 + 900 = 1800	∞ + 900 = ∞	200 + 900 = 1100
<u>A</u> (C) = ∞	B(C) = 500	C(C) = 0	D(C) = ∞
0 + ∞ = ∞	900 + ∞ = ∞	∞ + ∞ = ∞	200 + ∞ = ∞
<u>A</u> (D) = 200	B(D) = ∞ diganti 1100	C(D) = ∞	D(D) = 0
0 + 200 = 200	900 + 200 = 1100	∞ + 200 = ∞	200 + 200 = 400
<u>A</u> (E) = ∞	<u>B</u> (E) = 600	C(E) = 850	D(E) = 1100
0 + ∞ = ∞	900 + ∞ = ∞	∞ + ∞ = ∞	200 + ∞ = ∞
Er = ∞			
<u>E</u> (A) = ∞			
∞ + 0 = ∞			
<u>E</u> (B) = 600			
∞ + 900 = ∞			
E(C) = 850			
∞ + ∞ = ∞			
<u>E</u> (D) = 1100			
∞ + 200 = ∞			
<u>E</u> (E) = 0			

R1 = 1.1=0, 1.2=900, 1.3= ∞ , 1.4=200, 1.5= ∞

Ini adalah iterasi pertama R1, dan di temukan bahwa dari node B ke node D jarak sebenarnya lebih besar dari hasil penjumlahan pada node tersebut, maka nilai sebenarnya di ganti dengan nilai hasil penjumlahan.

Gambar 3.3 Perhitungan iterasi pertama

- g. Dari perhitungan tersebut, didapatkanlah lagi tabel matriks yang kedua. Berikut ini ialah tabelnya.

Tabel 3.3 hasil tabel matriks kedua

Dari, Ke	A	B	C	D	E
A	0	900	∞	200	∞
B	900	0	500	1100	600
C	∞	500	0	∞	850
D	200	1100	∞	0	1100
E	∞	600	850	1100	0

- h. Selanjutnya, dilakukan lagi perhitungan berdasarkan rumus. Dan didapatkanlah hasil iterasi kedua.

<p>R2:</p> <p>Ar = 900</p> <p>A(A) = 0</p> <p>A(B) = 900</p> <p>900 + 0 = 900</p> <p>A(C) = ∞ diganti 1400</p> <p>900 + 500 = 1400</p> <p>A(D) = 200</p> <p>900 + 1100 = 2000</p> <p>A(E) = ∞ diganti 1500</p> <p>900 + 600 = 1500</p>	<p>Br = 0</p> <p>B(A) = 900</p> <p>900 + 900 = 1800</p> <p>B(B) = 0</p> <p>B(C) = 500</p> <p>900 + 500 = 1400</p> <p>B(D) = 1100</p> <p>900 + 1100 = 2000</p> <p>B(E) = 600</p> <p>900 + 600 = 1500</p>	<p>Cr = 500</p> <p>C(A) = ∞ diganti 1400</p> <p>500 + 900 = 1400</p> <p>C(B) = 500</p> <p>500 + 0 = 500</p> <p>C(C) = 0</p> <p>C(D) = ∞ diganti 1600</p> <p>500 + 1100 = 1600</p> <p>C(E) = 850</p> <p>500 + 600 = 1100</p>
<p>Dr = 1100</p> <p>D(A) = 200</p> <p>1100 + 900 = 2000</p> <p>D(B) = 1100</p> <p>1100 + 0 = 1100</p> <p>D(C) = ∞ diganti 1600</p> <p>1100 + 500 = 1600</p> <p>D(E) = 1100</p> <p>1100 + 600 = 1700</p>	<p>Er = 600</p> <p>E(A) = ∞ diganti 1500</p> <p>600 + 900 = 1500</p> <p>E(B) = 600</p> <p>600 + 0 = 600</p> <p>E(C) = 850</p> <p>600 + 500 = 1100</p> <p>E(D) = 1100</p> <p>600 + 1100 = 1700</p> <p>E(E) = 0</p>	<p>R2: 2.1=900, 2.2=0,</p> <p>2.3=500, 2.4=1100, 2.5=600</p> <p>Semua indeks yang nilai jarak sebenarnya lebih besar dari hasil penjumlahan di ganti dengan nilai hasil penjumlahan</p>

Gambar 3.4 Perhitungan iterasi kedua

- i. Dari perhitungan tersebut, didapatkanlah lagi tabel matriks yang ketiga. Tabel matriks yang ketiga dapat dilihat pada tabel 3.4.

Tabel 3.4 hasil tabel matriks ketiga

Dari,Ke	A	B	C	D	E
A	0	900	1400	200	1500
B	900	0	500	1100	600
C	1400	500	0	1600	850
D	200	1100	1600	0	1100
E	1500	600	850	1100	0

- j. Selanjutnya, dilakukan lagi perhitungan berdasarkan rumus. Dan didapatkanlah hasil iterasi ketiga.

R3:		
Ar = 1400	Br = 500	Cr = 0
<u>A(A)</u> = 0	B(A) = 900	C(A) = 1400
<u>A(B)</u> = 900	500 + 1400 = 1900	0 + 1400 = 1400
1400 + 500 = 1900	<u>B(B)</u> = 0	C(B) = 500
A(C) = 1400	B(C) = 500	0 + 500 = 500
1400 + 0 = 1400	500 + 0 = 500	C(C) = 0
<u>A(D)</u> = 200	B(D) = 1100	C(D) = 1600
1400 + 1600 = 3000	500 + 1600 = 2100	0 + 1600 = 1600
A(E) = 1500	<u>B(E)</u> = 600	C(E) = 850
1400 + 850 = 2250	500 + 850 = 1350	
Dr = 1600	Er = 850	
<u>D(A)</u> = 200	E(A) = 1500	
<u>D(B)</u> = 1100	E(B) = 600	
D(C) = 1600	E(C) = 850	
<u>D(D)</u> = 0	E(D) = 1100	
<u>D(E)</u> = 1100	E(E) = 0	

R3 : 3.1=1400, 3.2=500, 3.3=0,
3.4=1600, 3.5=850

Tidak ada perubahan nilai di iterasi ini

Gambar 3.5 Perhitungan iterasi ketiga

- k. Dari perhitungan tersebut, didapatkanlah lagi tabel matriks yang keempat. Berikut ini ialah tabelnya.

Tabel 3.5 hasil tabel matriks keempat

Dari,Ke	A	B	C	D	E
A	0	900	1400	200	1500
B	900	0	500	1100	600
C	1400	500	0	1600	850
D	200	1100	1600	0	1100
E	1500	600	850	1100	0

- l. Selanjutnya, dilakukan lagi perhitungan berdasarkan rumus. Dan didapatkanlah hasil iterasi keempat.

R4:		
Ar = 200	Br = 1100	Cr = 1600
<u>A(A)</u> = 0	B(A) = 900	C(A) = 1400
<u>A(B)</u> = 900	B(B) = 0	C(B) = 500
A(C) = 1400	B(C) = 500	C(C) = 0
<u>A(D)</u> = 200	B(D) = 1100	C(D) = 1600
A(E) = 1500 diganti 1300	<u>B(E)</u> = 600	C(E) = 850
200 + 1100 = 1300		
Dr = 0	Er = 1100	
<u>D(A)</u> = 200	E(A) = 1500 diganti 1300	
<u>D(B)</u> = 1100	1100 + 200 = 1300	
D(C) = 1600	<u>E(B)</u> = 600	
<u>D(D)</u> = 0	E(C) = 850	
<u>D(E)</u> = 1100	E(D) = 1100	
	<u>E(E)</u> = 0	

R4 : 4.1=200, 4.2=1100, 4.3=1600,
4.4=0, 4.5=1100

Semua indeks yang nilai jarak sebenarnya lebih besar dari hasil penjumlahan di ganti dengan nilai hasil penjumlahan, maka nilai sebenarnya di ganti dengan nilai hasil penjumlahan.

Gambar 3.6 Perhitungan iterasi keempat

m. Selanjutnya, didapatkanlah lagi tabel matriks yang kelima. Berikut ini ialah tabelnya.

Tabel 3.6 hasil tabel matriks kelima

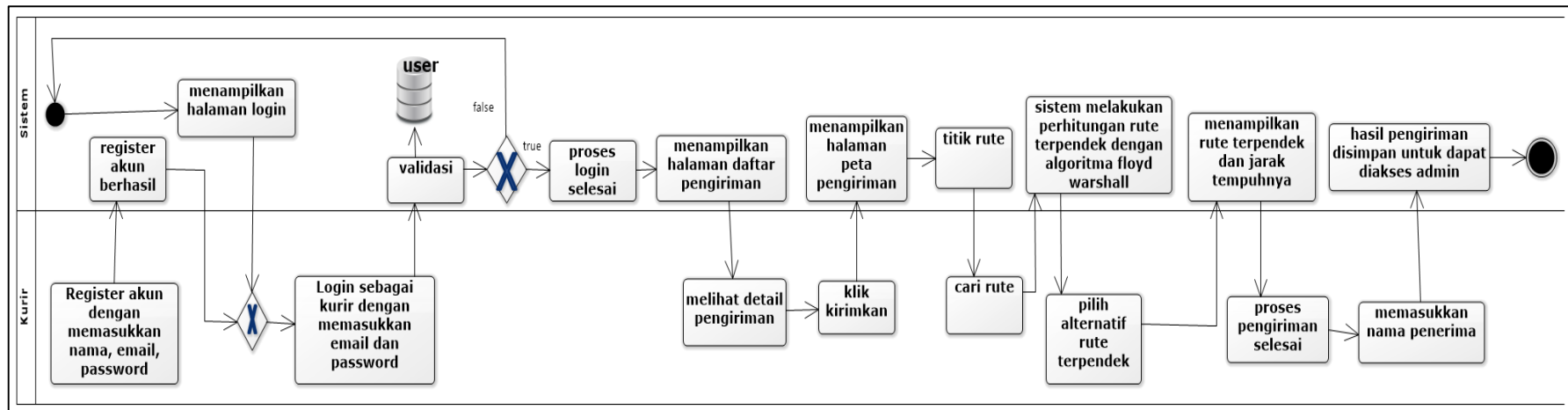
Dari, Ke	A	B	C	D	E
A	0	900	1400	200	1300
B	900	0	500	1100	600
C	1400	500	0	1600	850
D	200	1100	1600	0	1100
E	1300	600	850	1100	0

R5 (Kesimpulan):

Jika titik awal adalah A dan titik akhir adalah E, maka A ke A = 0, A ke B = 900, A ke C = 1400, A ke D = 200, A ke E = 1300. Dari iterasi ini tak lagi ditemukan perubahan nilai jarak sebenarnya, jadi telah didapatkan semua kemungkinan nilai terendah dari semua *node* ke semua *node*.

3.5 Proses Bisnis

Proses bisnis merupakan sekumpulan aktifitas terstruktur untuk mencapai hasil dan tujuan tertentu. Adapun proses bisnis pada sistem pengiriman hewan kurban, berawal dari admin yang mulai memasukkan data pengiriman berupa jenis hewan, lokasi pengiriman, dan berat pengiriman. Dari data yang telah dimasukkan oleh admin, akan tampil berupa informasi tempat pengambilan hewan dan angkutan yang cocok untuk mengangkut dan mengantarkan hewan-hewan tersebut. Ketika kurir telah menerima data pengiriman, kurir akan memilih urutan pengiriman jika tujuan pengiriman lebih dari satu dan kemudian klik data yang akan dikirim. Sistem akan memeriksa jarak tempuh standar antara peternakan dan lokasi pengiriman melalui *Google Maps* dan membandingkan jarak terendahnya. Setelah itu, sistem akan melakukan perhitungan rute terdekat dengan algoritma *Floyd warshall*. Saat kurir mulai melakukan pengiriman, secara otomatis admin dapat mengetahui status pengiriman. Ketika kurir mulai bergerak untuk mengirimkan hewan kurban, admin dapat memantau posisi kurir secara realtime. Pengiriman dikatakan selesai jika kurir telah memasukkan nama penerima, *last odometer*, *total bahan bakar* dan menekan tombol selesai. Kemudian data tersebut akan diterima oleh admin. Proses bisnis sistem dapat dilihat pada Gambar 3.7.



Gambar 3.7 Urutan Proses Bisnis Sistem

BAB IV

PERANCANGAN SISTEM

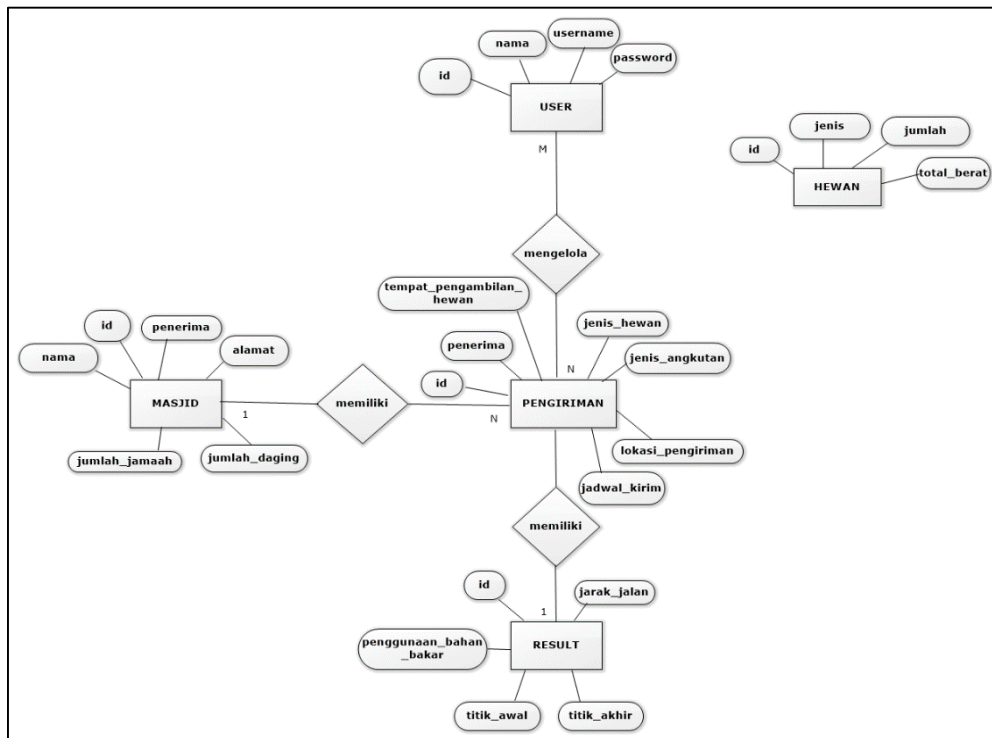
4.1 Proses Pengambilan Data dengan Google Maps API

Sistem ini adalah sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban. Dalam menentukan rute menuju ke tempat pengiriman, maka sistem ini menggunakan *Google Maps API* untuk penentuan jalurnya dan perhitungan rute terdekat menggunakan algoritma *floyd warshall*. Adapun prosesnya ialah sebagai berikut:

- a. Sistem ini menggunakan *Google Maps API* untuk mendapatkan jalur yang memungkinkan untuk dilewati tanpa mempertimbangkan kondisi jalan, cuaca, kepadatan lalu lintas, dan dan kendaraan apa yang akan digunakan.
- b. Sistem ini menggunakan *Google Maps API* dengan *driving mode*. *Driving mode* tidak tidak memperhitungkan jalur bus atau gang, namun rutenya sesuai kebutuhan masukan dari kurir saat memilih berbagai rute pilihannya sendiri.
- c. Setelah mendapatkan jalur dari *Google Maps*, sistem melakukan *request* nilai jarak dari masing-masing jalur yang telah diberikan.
- d. Nilai dari jarak tiap titik ke titik yang diberikan oleh *Google Maps*, dimasukkan ke dalam sebuah *array*, dan jika sistem melewati lebih dari satu titik, maka nilai di set *infinity*.
- e. Sistem kembali melakukan *request* terhadap titik yang bernilai *infinity* untuk memastikan apakah masih ada jalur langsung yang dapat didahului tanpa melewati lebih dari satu titik. Jika ada jalur langsung yang dapat didahului, maka sistem akan mengganti nilai *infinity* dengan nilai jarak yang ada.
- f. Setelah semua titik di cek keterhubungannya, kemudian akan dilakukan penghitungan metode *floyd warshall*.
- g. Sistem melakukan perhitungan jarak yg diberikan menggunakan algoritma *floyd warshall*.
- h. Algoritma *floyd warshall* menghasilkan nilai jarak terendah dari masing-masing jalur yg memungkinkan.
- i. Kemudian, jarak terendah yang telah didapatkan tersebut akan dipetakan kembali oleh *Google Maps API*.
- j. Sistem ini lebih berguna di jalan yang memiliki banyak rute.

4.2 Perancangan ERD

Entity relationship diagram (ERD) merupakan diagram yang menggambarkan relasi antar tabel basis data yang akan dibuat. Pada ERD terdapat entitas yang akan menjadi nama tabel pada basis data, kemudian entitas memiliki atribut yang merupakan nama kolom dari tabel tersebut. Terdapat kardinalitas pada setiap entitas yang terhubung dan *relationship* yang menghubungkan tiap entitas. Rancangan ERD sistem ini dapat dilihat pada Gambar 4.1.

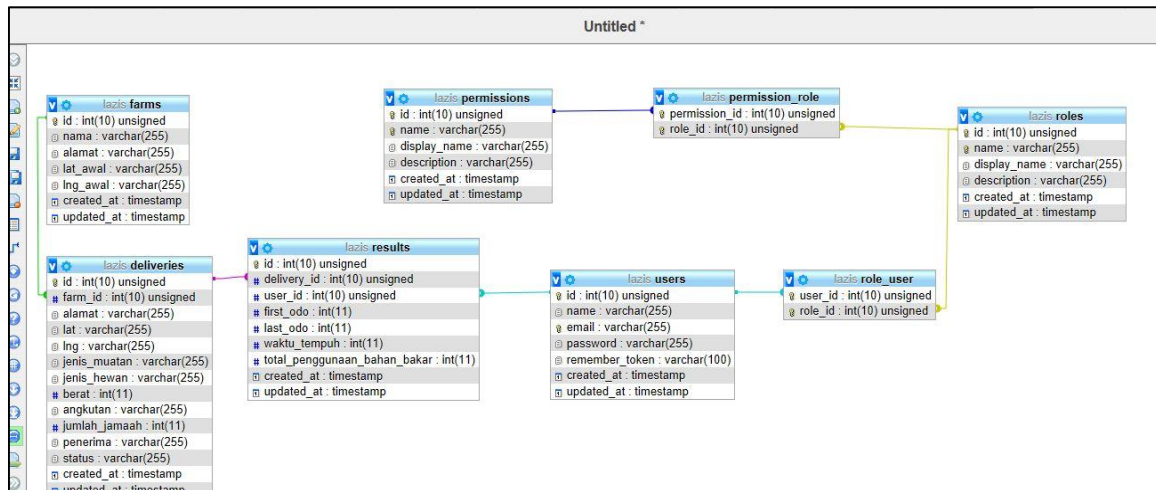


Gambar 4.1 Rancangan ERD

Pada ERD tersebut terdapat entitas *user*, masjid, pengiriman, result dan entitas hewan yang berdiri sendiri. Masing-masing entitas tersebut memiliki atribut yang akan menjadi kolom pada tabel-tabel basis data yang akan dibuat. Selanjutnya, terdapat *relationship* yang menghubungkan tiap entitasnya, *relationship* tersebut berupa atribut memiliki dan mengelola. Terdapat pula kardinalitas pada tiap entitas yang terhubung.

4.3 Tabel Relasi Basis Data

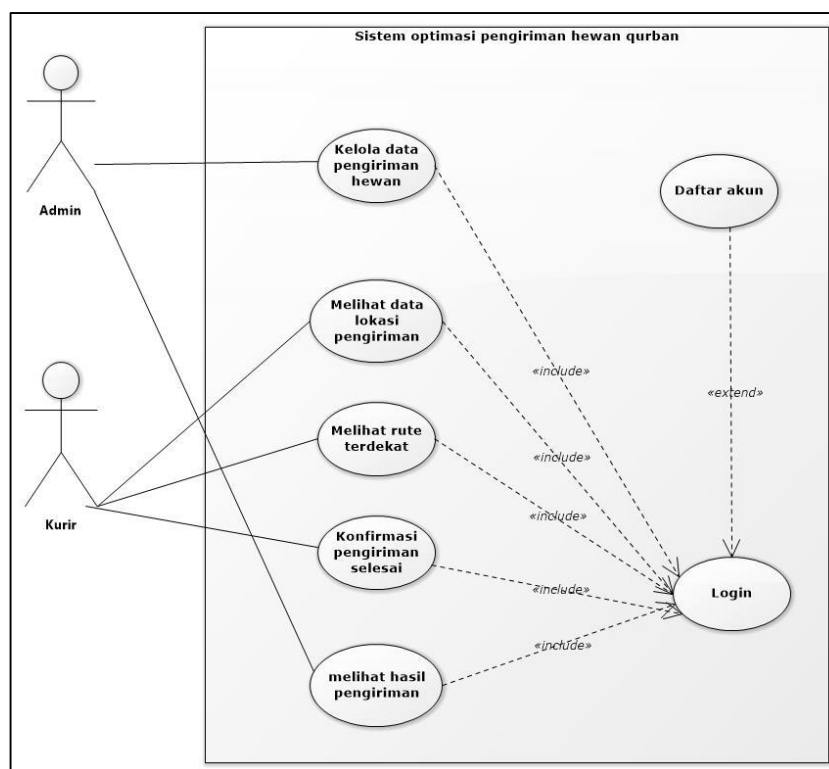
Setelah membuat ERD, dibuatlah basis datanya dengan berdasar kepada ERD yang telah dibuat sebelumnya. Pada tabel relasi ini akan terlihat mana saja yang memiliki *foreign key* pada tabelnya. Tabel ini telah direlasikan pada *phpMyAdmin*. Tabel relasi dapat dilihat pada Gambar 4.2.



Gambar 4.2 Tabel yang telah direlasikan

4.4 Perancangan Use Case Diagram

Use case merupakan diagram yang menjelaskan interaksi antara aktor (pengguna) dengan sistem. *Use case diagram* pada sistem optimasi penentuan rute terdekat pengiriman hewan kurban dapat dilihat pada Gambar 4.3.



Gambar 4.3 Use case diagram

Terdapat dua aktor yang ada pada diagram ini yaitu admin dan kurir. Admin merupakan pengguna yang memiliki akses untuk mengelola data pengiriman hewan kurban. Kurir adalah orang yang mengambil hewan kurban ke tempat peternakan dan mengantarnya ke tempat pendistribusian. Kurir merupakan pengguna yang dapat melihat data lokasi pengiriman, melihat rute terdekat ke tempat pengiriman, dan mengonfirmasi data selesai pengiriman. Data konfirmasi selesai pengiriman dari kurir akan diterima dan dilihat oleh admin.

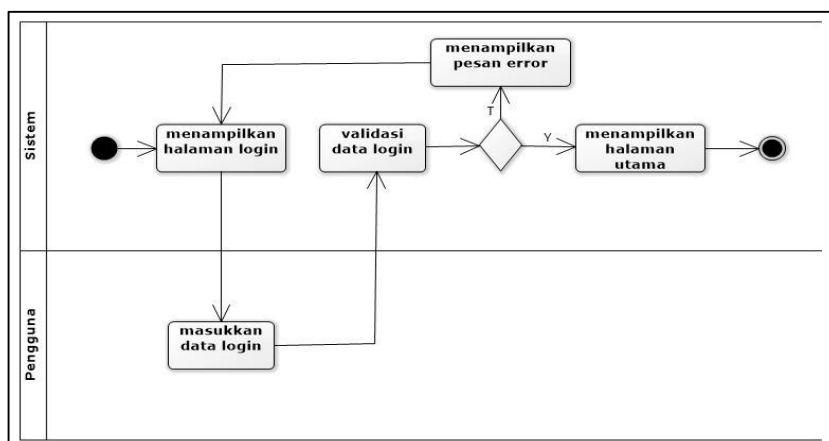
Untuk dapat melakukan semua proses pada sistem, admin maupun kurir harus melakukan *login* terlebih dahulu. Sebelumnya, pengguna harus melakukan daftar akun terlebih dahulu, supaya dapat melakukan *login*.

4.5 Perancangan Activity Diagram

Activity diagram adalah diagram yang menggambarkan aktifitas-aktifitas dan alur yang ada pada sistem yang dirancang. Berikut ini adalah beberapa aktifitas yang ada pada sistem:

4.5.1 Activity Diagram Login

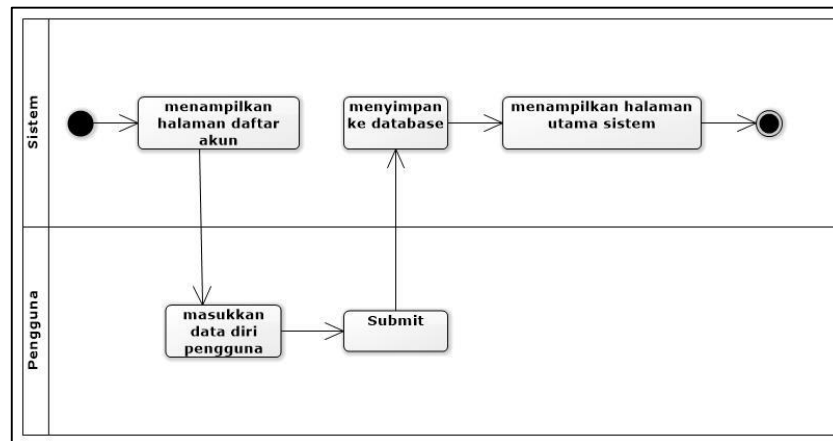
Pada aktifitas ini, sistem menampilkan halaman *login*, kemudian pengguna memasukkan data *login* yang berupa *username* dan *password*. Setelah itu sistem melakukan validasi data sesuai dengan yang terdaftar pada *database*. Jika data *login* yang dimasukkan oleh pengguna adalah benar, maka sistem menampilkan halaman utama sistem. Jika data *login* yang dimasukkan oleh pengguna adalah salah, maka sistem akan menampilkan pesan *error* dan sistem akan kembali menampilkan halaman *login*. Rancangannya dapat dilihat pada Gambar 4.4.



Gambar 4.4 Activity diagram login

4.5.2 Activity Diagram Register

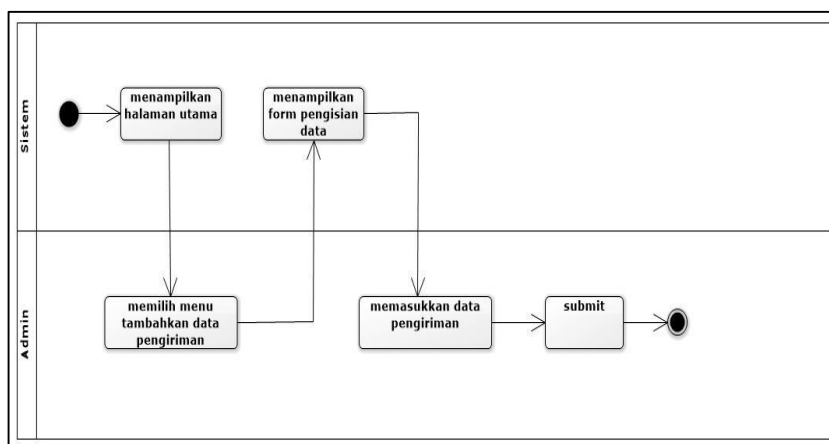
Pada aktifitas ini, sistem menampilkan halaman daftar akun, kemudian pengguna memasukkan data diri untuk pembuatan akun dan melakukan submit. Setelah itu sistem akan menyimpan data tersebut ke dalam *database*. Sistem akan langsung menampilkan halaman utama sistem setelah pengguna selesai melakukan daftar akun. *Activity diagram* daftar akun dapat dilihat pada Gambar 4.5.



Gambar 4.5 Activity diagram daftar akun

4.5.3 Activity Diagram Kelola Data Pengiriman Hewan

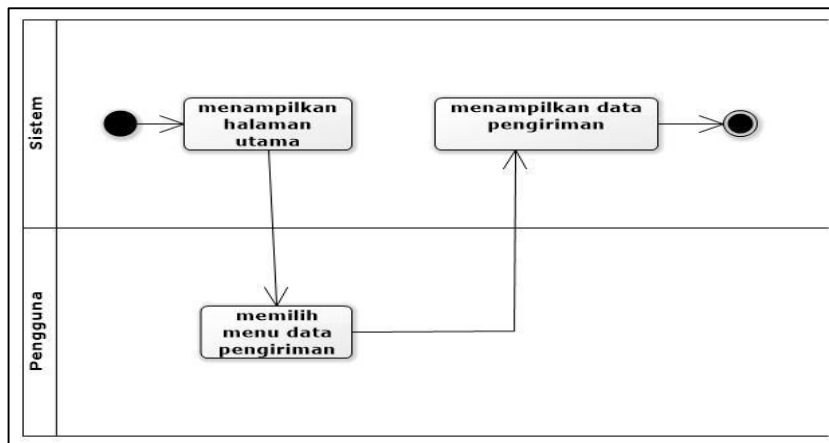
Pada aktifitas ini, sistem menampilkan halaman utama sistem setelah login. Pengguna (admin) memilih menu tambahkan data pengiriman, dan sistem akan menampilkan form pengisian data pengiriman hewan kurban. Setelah selesai memasukkan data, admin hanya perlu menekan tombol simpan. Rancangannya dapat dilihat pada Gambar 4.6.



Gambar 4.6 Activity diagram kelola data pengiriman

4.5.4 Activity Diagram Melihat Data Pengiriman

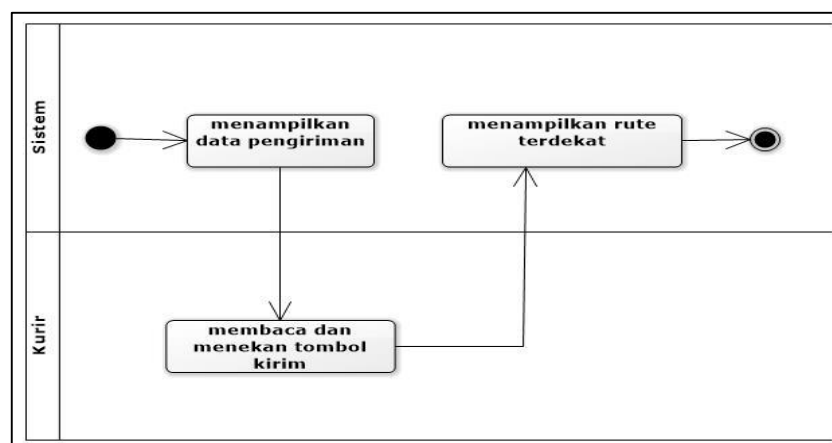
Pada aktifitas ini, sistem menampilkan halaman utama. Pada menu yang tersedia, pengguna (admin maupun kurir) memilih menu data pengiriman. Sistem akan menampilkan data pengiriman yang telah dimasukkan sebelumnya oleh admin. Rancangannya dapat dilihat pada Gambar 4.7.



Gambar 4.7 Activity diagram melihat data pengiriman

4.5.5 Activity Diagram Melihat Rute Terdekat

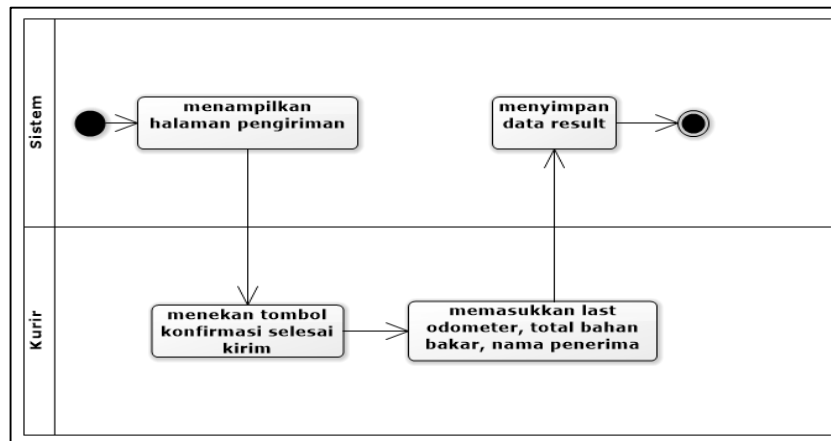
Pada aktifitas ini, sistem menampilkan data pengiriman. Kurir melihat data pengiriman dan menekan tombol kirim. Setelah kurir menekan tombol kirim, sistem akan menampilkan rute terdekat menuju ke tempat tujuan sesuai dengan data pengiriman. Rancangannya dapat dilihat pada Gambar 4.8.



Gambar 4.8 Activity Diagram melihat rute terdekat

4.5.6 Activity Diagram Konfirmasi Pengiriman Selesai

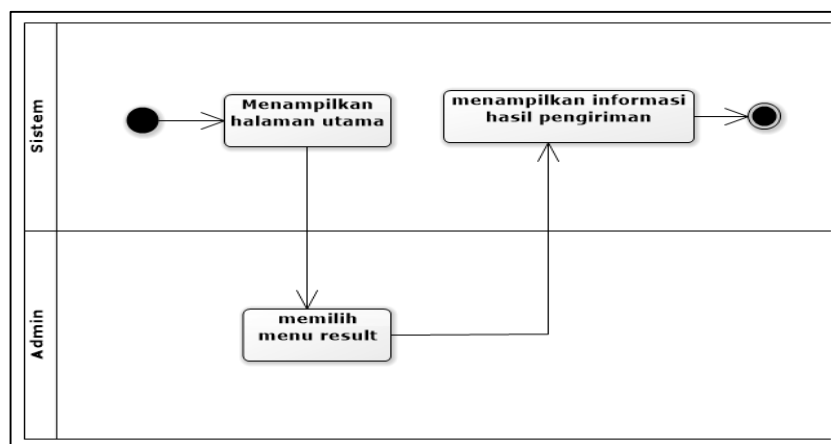
Pada aktifitas ini, kurir akan melakukan konfirmasi selesai kirim jika pengiriman telah selesai dilakukan. Awalnya sistem akan menampilkan halaman pengiriman kemudian kurir akan menekan tombol selesai kirim dan memasukkan data *last odometer*, total bahan bakar, dan nama penerima. Selanjutnya, data pengiriman tersebut akan dikirim ke admin. Rancangannya dapat dilihat pada Gambar 4.9.



Gambar 4.9 Activity Diagram konfirmasi selesai pengiriman

4.5.7 Activity Diagram Melihat Hasil Pengiriman

Pada aktifitas ini, sistem menampilkan menu utama. Untuk melihat hasil pengiriman, admin memilih menu *result*. Selanjutnya, sistem akan menampilkan informasi hasil pengiriman yang telah dilakukan dan dikirimkan oleh kurir. Rancangannya dapat dilihat pada Gambar 4.10.



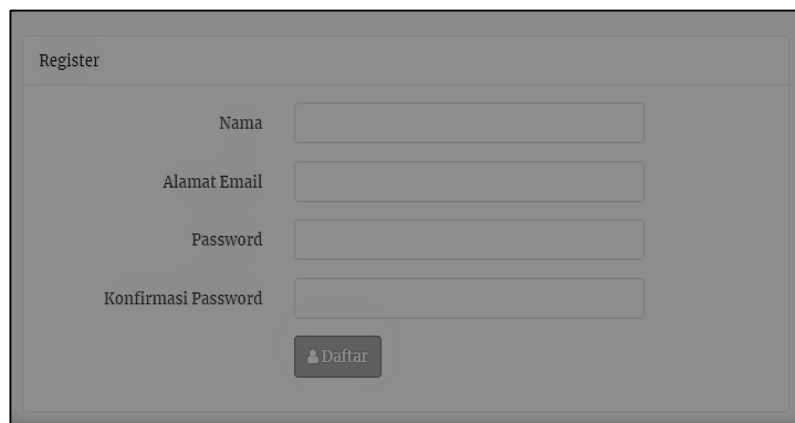
Gambar 4.10 Activity Diagram melihat hasil pengiriman

4.6 Perancangan Antarmuka

Perancangan antarmuka dibutuhkan untuk memudahkan dalam pembuatan *interface* sistem. Dari perancangan ini akan membantu untuk penggambaran seperti apakah tampilan dari *interface* pengguna. Berikut ini adalah rancangan antarmuka sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban.

4.6.1 Rancangan Antarmuka Register

Rancangan antarmuka register adalah rancangan halaman untuk pengguna membuat akun dengan cara mendaftarkan data seperti nama, alamat email, dan password yang nantinya akan digunakan untuk login ke sistem. Rancangannya dapat dilihat pada Gambar 4.11.

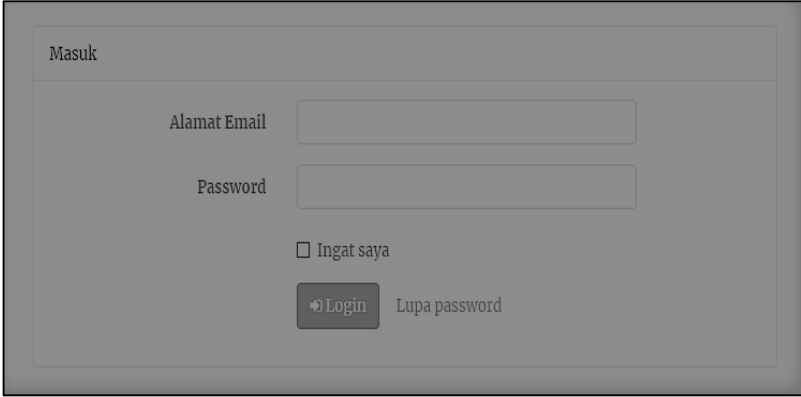


The image shows a wireframe of a registration form. At the top left, the word "Register" is written. Below it, there are four input fields arranged vertically. The first field is labeled "Nama", the second "Alamat Email", the third "Password", and the fourth "Konfirmasi Password". Below these fields is a button labeled "Daftar" with a small user icon to its left.

Gambar 4.11 Rancangan Antarmuka Register

4.6.2 Rancangan Antarmuka Login

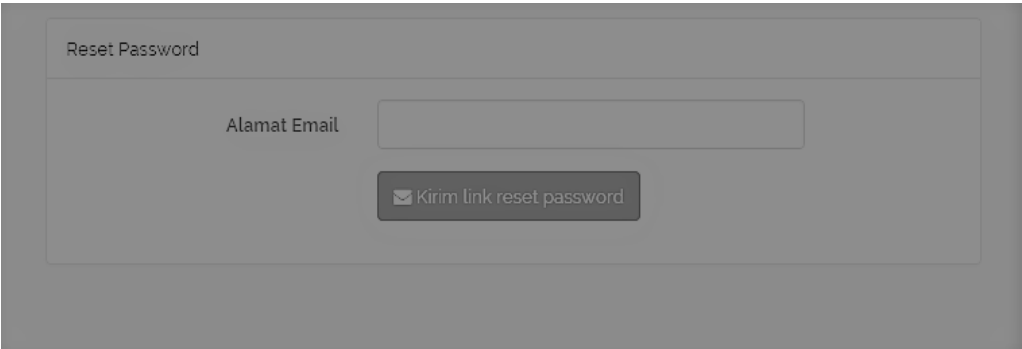
Rancangan antarmuka login adalah rancangan halaman untuk pengguna memasukkan data berupa alamat email dan password yang telah didaftarkan pada halaman register. Rancangannya dapat dilihat pada Gambar 4.12.

A login form titled "Masuk" (Login) with a light gray background. It contains two input fields: "Alamat Email" (Email Address) and "Password". Below the password field is a checkbox labeled "Ingat saya" (Remember me). At the bottom, there is a "Login" button with a right-pointing arrow icon, and a "Lupa password" (Forgot password) link to its right.

Gambar 4.12 Rancangan Antarmuka Login

4.6.3 Rancangan Antarmuka Atur Ulang Kata Sandi

Rancangan antarmuka *reset password* adalah rancangan halaman untuk mengatur ulang kata sandi jika pengguna ingin mengganti kata sandi dengan yang baru. Rancangannya dapat dilihat pada Gambar 4.13.

A "Reset Password" form with a light gray background. It features an "Alamat Email" (Email Address) input field. Below the input field is a button with an envelope icon and the text "Kirim link reset password" (Send reset password link).

Gambar 4.13 Rancangan Antarmuka Atur Ulang Kata Sandi

4.6.4 Rancangan Antarmuka Tambah Data Pengiriman

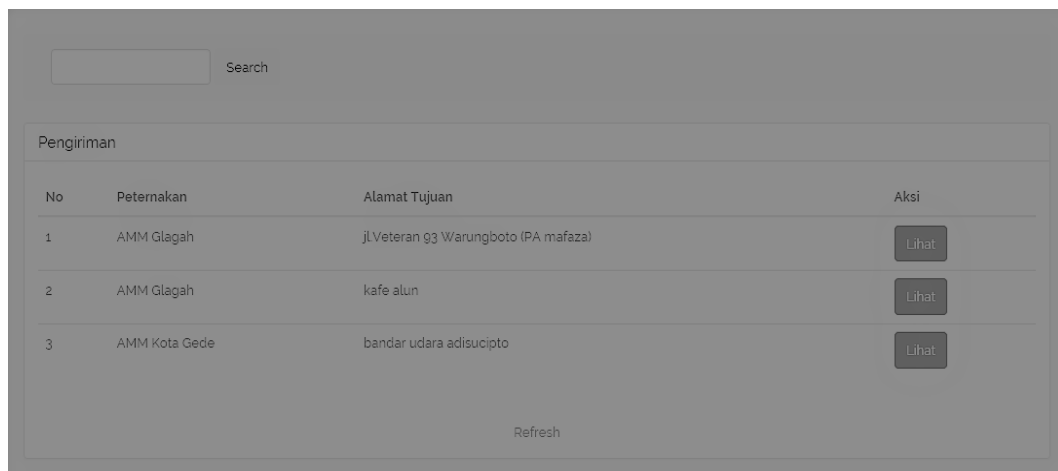
Rancangan antarmuka tambah data pengiriman merupakan halaman untuk pengguna (admin) menambahkan data pengiriman hewan kurban yang nantinya akan diproses oleh kurir. Rancangannya dapat dilihat pada Gambar 4.14.



Gambar 4.14 Rancangan Antarmuka Tambah Data Pengiriman

4.6.5 Rancangan Antarmuka Melihat Data Pengiriman yang telah dimasukkan

Rancangan antarmuka ini merupakan rancangan halaman untuk pengguna melihat data pengiriman yang telah dimasukkan oleh admin. Rancangannya dapat dilihat pada Gambar 4.15.



No	Pernakan	Alamat Tujuan	Aksi
1	AMM Glagah	jl.Veteran 93 Warungboto (PA mafaza)	Lihat
2	AMM Glagah	kafe alun	Lihat
3	AMM Kota Gede	bandar udara adisucipto	Lihat

Refresh

Gambar 4.15 Rancangan Antarmuka Melihat Data Pengiriman

4.6.6 Rancangan Antarmuka Melihat Data Pengguna

Rancangan antarmuka ini merupakan rancangan halaman untuk admin melihat data pengguna yang merupakan kurir dan member dari sistem ini. Rancangannya dapat dilihat pada Gambar 4.16.

Show entries Search:

ID Pegawai	Nama	Email	Jabatan	Tindakan
2	mahmud	member@lazismu.com	Kurir	Ubah HAPUS
4	babang	kurir2@lazismu.com	Member	Ubah HAPUS
6	mamat	mamat@gmail.com	Kurir	Ubah HAPUS
7	kamto	kamto@gmail.com	Member	Ubah HAPUS
8	dadangku	dadangku@love.com	Member	Ubah HAPUS
9	desy	desyfajarrahayu@gmail.com	Kurir	Ubah HAPUS
10	Kim Soek Jin	jin@bts.com	Kurir	Ubah HAPUS
11	nabila	nabila@gmail.com	Kurir	Ubah HAPUS

Gambar 4.16 Rancangan Antarmuka Melihat Data Pengguna

4.6.7 Rancangan Antarmuka Melihat Detail Pengiriman pada Sisi Kurir

Rancangan antarmuka ini merupakan rancangan halaman dari sisi kurir yang memunculkan data pengiriman hewan kurban yang harus dikirim. Rancangannya dapat dilihat pada Gambar 4.17.

Data Pengiriman

Jenis Muatan

daging

Jenis Hewan

sapi

Total Berat Muatan (Kilogram)

60

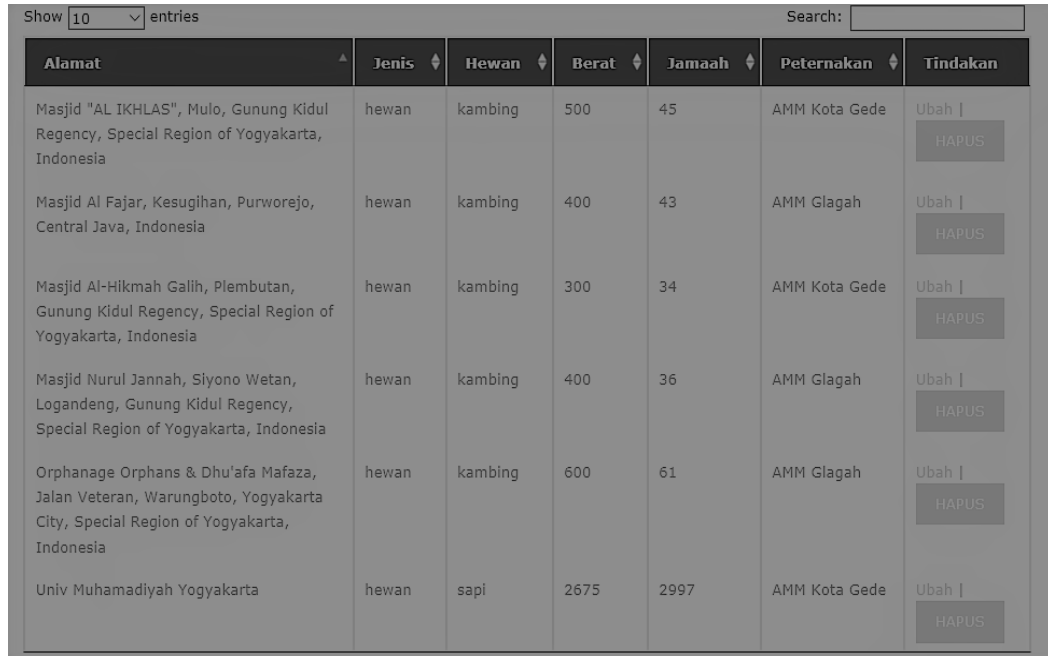
Angkutan

pick-up

Gambar 4.17 Rancangan Antarmuka Melihat Rincian Data Pengiriman Sisi Kurir

4.6.8 Rancangan Antarmuka Melihat Rincian Data Pengiriman dan Melakukan Aksi

Rancangan ini merupakan rancangan halaman rincian data pengiriman yang dari sisi admin. Pada halaman ini, admin dapat melakukan aksi seperti edit dan hapus. Rancangannya dapat dilihat pada Gambar 4.18.

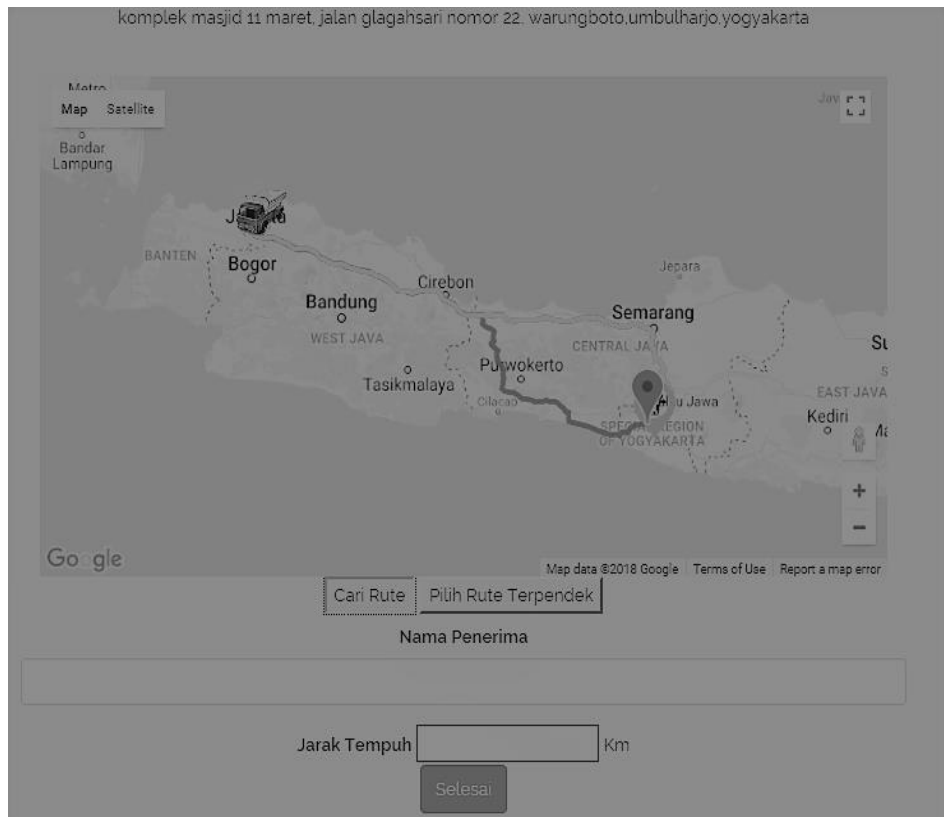


Alamat	Jenis	Hewan	Berat	Jamaah	Peternakan	Tindakan
Masjid "AL IKHLAS", Mulo, Gunung Kidul Regency, Special Region of Yogyakarta, Indonesia	hewan	kambing	500	45	AMM Kota Gede	Ubah HAPUS
Masjid Al Fajar, Kesugihan, Purworejo, Central Java, Indonesia	hewan	kambing	400	43	AMM Glagah	Ubah HAPUS
Masjid Al-Hikmah Galih, Plembutan, Gunung Kidul Regency, Special Region of Yogyakarta, Indonesia	hewan	kambing	300	34	AMM Kota Gede	Ubah HAPUS
Masjid Nurul Jannah, Siyono Wetan, Logandeng, Gunung Kidul Regency, Special Region of Yogyakarta, Indonesia	hewan	kambing	400	36	AMM Glagah	Ubah HAPUS
Orphanage Orphans & Dhu'afa Mafaza, Jalan Veteran, Warungboto, Yogyakarta City, Special Region of Yogyakarta, Indonesia	hewan	kambing	600	61	AMM Glagah	Ubah HAPUS
Univ Muhamadiyah Yogyakarta	hewan	sapi	2675	2997	AMM Kota Gede	Ubah HAPUS

Gambar 4.18 Rancangan Antarmuka Tabel Aksi pada Data Pengiriman

4.6.9 Rancangan Antarmuka Melihat Peta

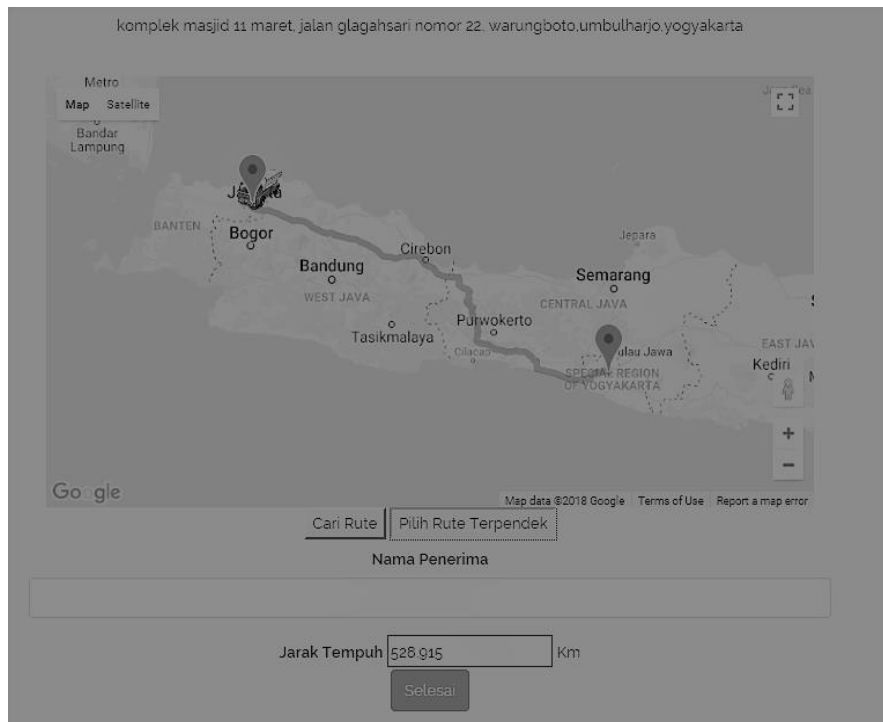
Rancangan antarmuka ini merupakan rancangan halaman yang menampilkan peta yang menunjukkan rute mana saja yang bisa dilewati untuk menuju ke tempat tujuan pengiriman. Rancangannya dapat dilihat pada Gambar 4.19.



Gambar 4.19 Rancangan Antarmuka Melihat Peta

4.6.10 Rancangan Antarmuka Melihat Rute Terdekat pada Peta

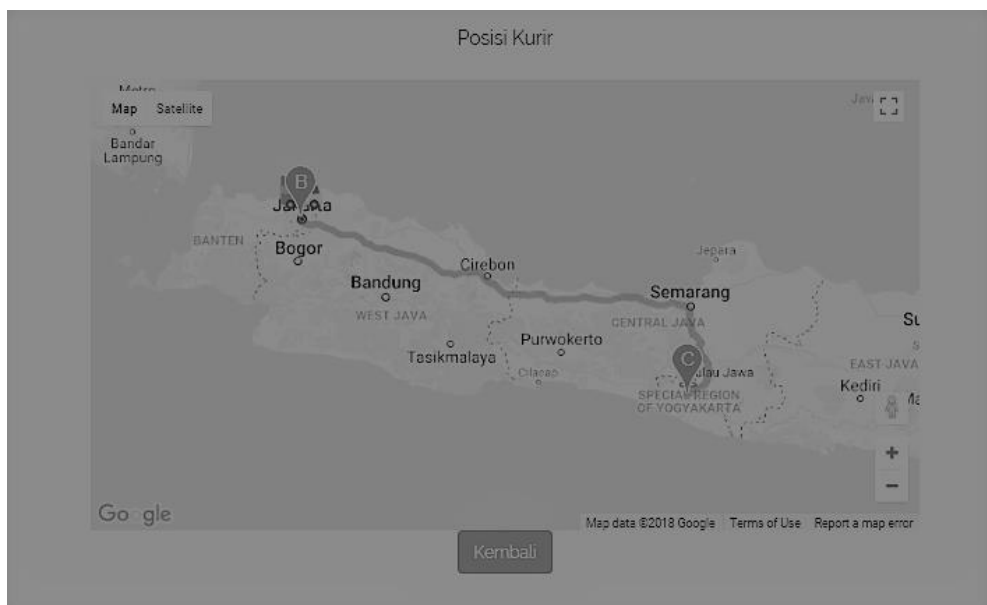
Rancangan ini merupakan halaman yang menampilkan peta yang hanya menunjukkan rute terdekat untuk menuju ke tempat tujuan pengiriman. Rancangannya dapat dilihat pada Gambar 4.20.



Gambar 4.20 Rancangan Antarmuka Melihat Rute Terdekat

4.6.11 Rancangan Antarmuka melakukan *Tracking* Posisi Kurir

Rancangan antarmuka ini merupakan rancangan halaman sisi admin yang melakukan *tracking* posisi kurir yang mengantarkan hewan kurban. Rancangannya dapat dilihat pada Gambar 4.21.



Gambar 4.21 Rancangan Antarmuka Tracking Posisi Kurir

4.7 Perancangan Teknik Pengujian Perangkat Lunak

Perancangan ini merupakan perancangan teknik pengujian yang memudahkan saat tahap pengujian. Awalnya menentukan terlebih dahulu macam-macam pengujian yang akan dilakukan pada sistem. Selanjutnya, dibuatlah perancangannya satu persatu. Pengujian ini sangat penting untuk mengetahui dan menilai kelayakan sistem. Berikut adalah macam-macam perancangan pengujian yang akan dilakukan pada penelitian ini:

a. Black box Testing

Tabel 4.1 Contoh Tabel Pengujian *Black Box*

No	Skenario Pengujian	Test Case	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan

Tabel 4.1 merupakan tabel pengujian *black box* pada sistem. Tabel ini berisi nomor, skenario pengujian, *test case*, hasil yang diharapkan, hasil pengujian, kesimpulan.

b. User Acceptance Test (UAT)

Tabel 4.2 Contoh Tabel Pengujian UAT Admin

No	Pernyataan	Penilaian				
		STS	TS	N	S	SS
1.	Website siskur (sistem kurban) mudah digunakan					
2.	Website siskur (sistem kurban) memiliki tampilan yang menarik					
3.	Penataan dan penyajian informasi memudahkan untuk membaca data dan mencari data					
4.	Website siskur memudahkan dalam penambahan dan penyimpanan data					
5.	Fitur " <i>search</i> " yang ada pada siskur memudahkan dalam pencarian data					
6.	Fitur <i>tracking</i> dapat memudahkan dalam memantau posisi kurir dalam pengiriman hewan kurban					
7.	Fitur <i>maps</i> /peta dapat memudahkan untuk melihat informasi rute-rute se-DIY					

Pada tabel 4.2 dan tabel 4.3 berisi nomor, pernyataan dan penilaian. Penilaian diisi oleh pengguna berdasarkan pengujian pada sistem yang telah dilakukan. Penilaian dibagi menjadi lima, yakni STS (Sangat Tidak Setuju), TS (Tidak Setuju), N (Normal), S (Setuju), SS (Sangat Setuju).

Tabel 4.3 Contoh Tabel Pengujian UAT Kurir

No	Pernyataan	Penilaian				
		STS	TS	N	S	SS
1.	Website siskur (sistem kurban) mudah digunakan					
2.	Website siskur (sistem kurban) memiliki tampilan yang menarik					
3.	Penataan dan penyajian informasi memudahkan untuk membaca data dan mencari data					
4.	Fitur “ <i>search</i> ” yang ada pada siskur memudahkan dalam pencarian data					
5.	Fitur “cari rute” memudahkan untuk melihat rute-rute mana saja yang dapat diakses dari/ke tempat pengiriman					
6.	Fitur “cari rute terdekat” dapat membantu pencarian rute terdekat menuju tempat pengiriman hewan					
7.	Fitur <i>maps</i> /peta dapat memudahkan untuk melihat informasi rute-rute se-DIY					

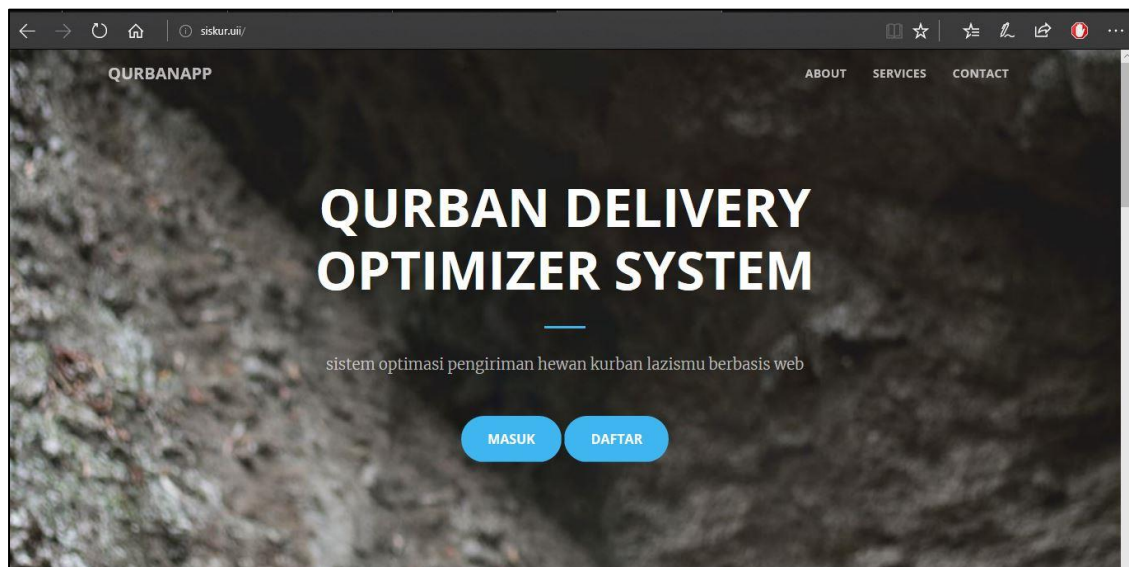
BAB V

IMPLEMENTASI DAN PENGUJIAN

5.1 Implementasi

Hasil penelitian ini berupa sistem optimasi penentuan rute terdekat untuk pengiriman hewan kurban. Sistem ini menggunakan algoritma *floyd warshall* untuk memudahkan dalam pencarian rute terdekat. Pada bagian ini akan dibahas mengenai implementasi dari perancangan sistem yang telah dibahas pada bab sebelumnya. Adapun yang akan dibahas antara lain ialah antarmuka sistem beserta penjelasannya.

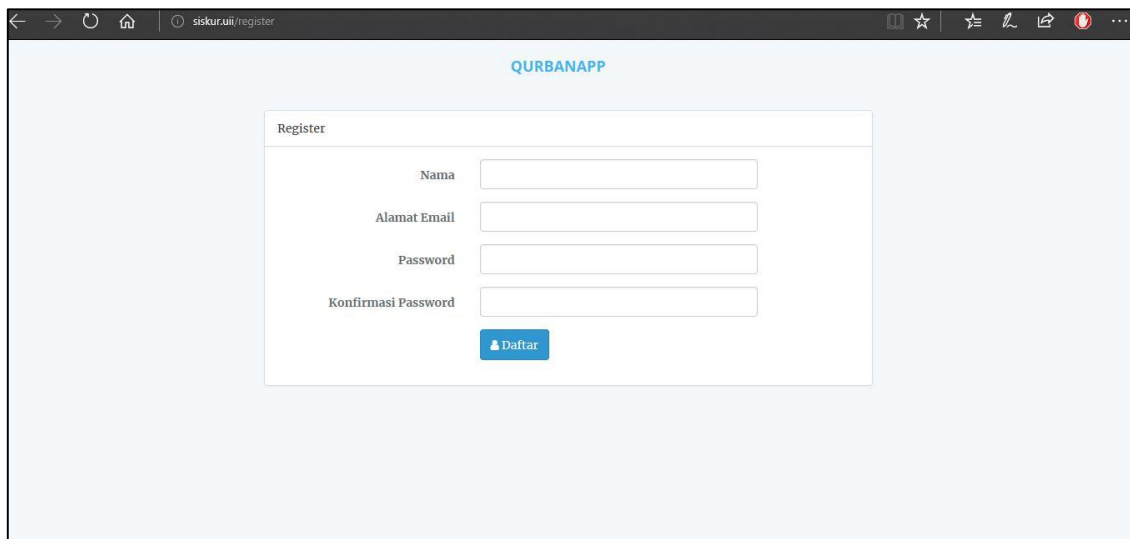
5.1.1 Antarmuka Halaman Home



Gambar 5.1 Antarmuka halaman utama

Gambar 5.1 adalah halaman yang ditampilkan oleh sistem ketika pengguna (admin ataupun kurir) pertama mengakses sistem. Halaman tersebut akan menampilkan pilihan pada pengguna yaitu daftar dan masuk. Menu daftar untuk pengguna melakukan pendaftaran akun baru, kemudian menu masuk adalah untuk pengguna melakukan *login* dengan memasukkan *email* dan *password*.

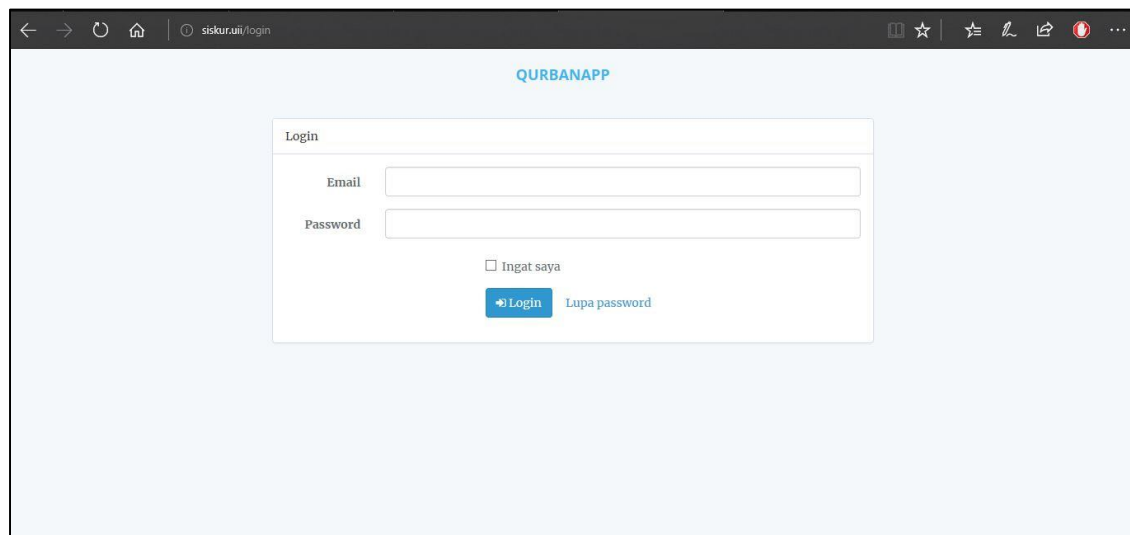
5.1.2 Antarmuka Halaman Daftar



Gambar 5.2 Antarmuka halaman daftar akun

Gambar 5.2 merupakan halaman yang ditampilkan sistem ketika pengguna memilih menu daftar pada halaman utama. Pada halaman ini, pengguna perlu memasukkan data berupa nama, *email*, dan *password* untuk pembuatan akun baru. Pengguna menekan tombol daftar untuk melakukan *submit* data yang telah didaftarkan untuk kemudian masuk ke *database*.

5.1.3 Antarmuka Halaman Masuk

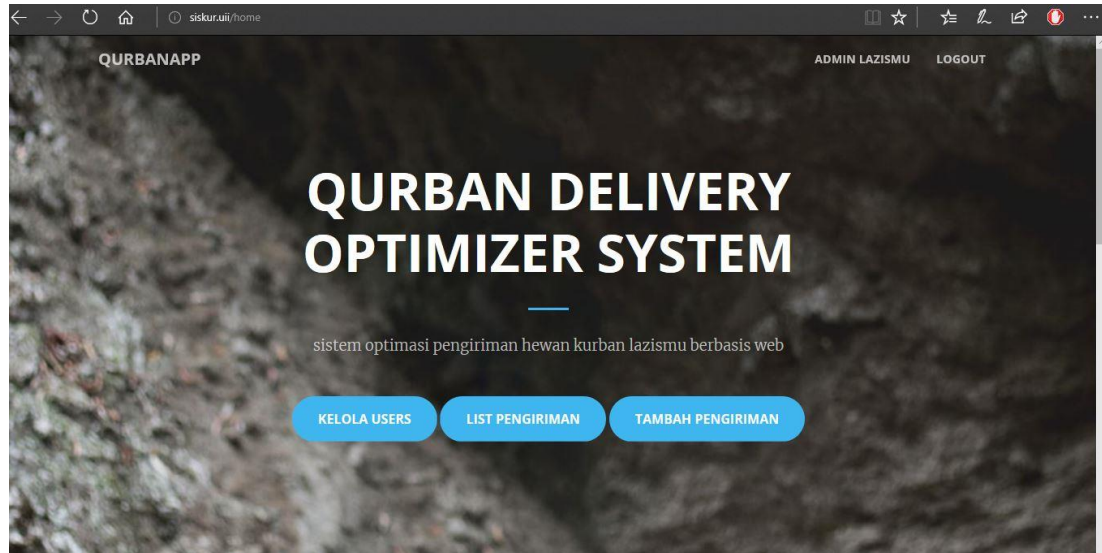


Gambar 5.3 Antarmuka halaman *login*

Gambar 5.3 merupakan halaman yang ditampilkan sistem ketika pengguna memilih menu masuk pada halaman utama. Pada halaman ini, pengguna hanya perlu memasukkan

email dan *password* yang sudah didaftarkan sebelumnya dan kemudian menekan tombol *login* untuk masuk dan mengakses fitur-fitur yang ada pada sistem.

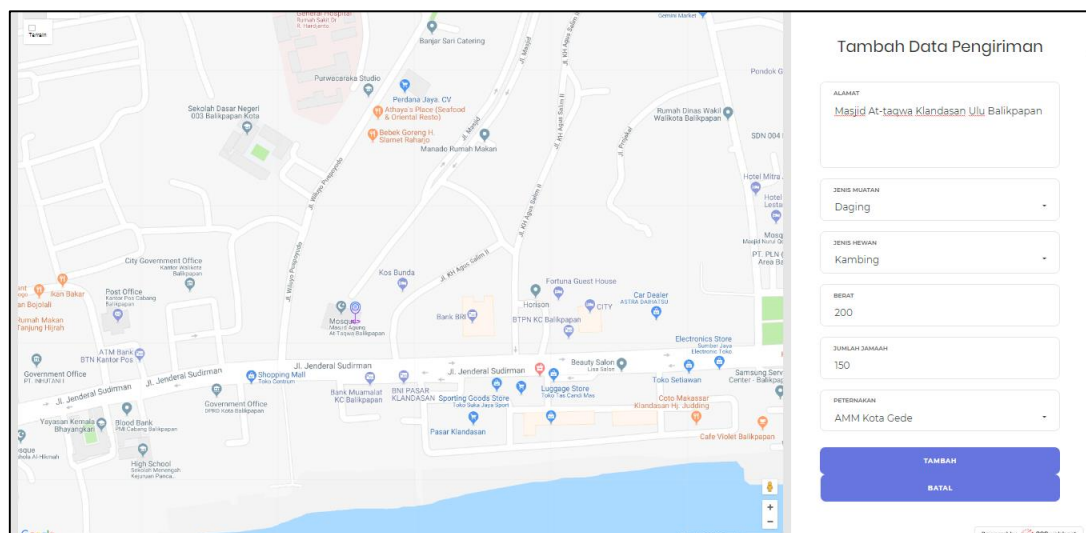
5.1.4 Antarmuka Halaman Beranda Admin



Gambar 5.4 Antarmuka halaman beranda admin

Gambar 5.4 merupakan halaman beranda dari sisi admin yang menampilkan pilihan untuk admin. Pilihan tersebut terdapat pada tiga menu, yakni kelola *users*, *list* pengiriman dan tambah pengiriman.

5.1.5 Antarmuka Halaman Tambah Data

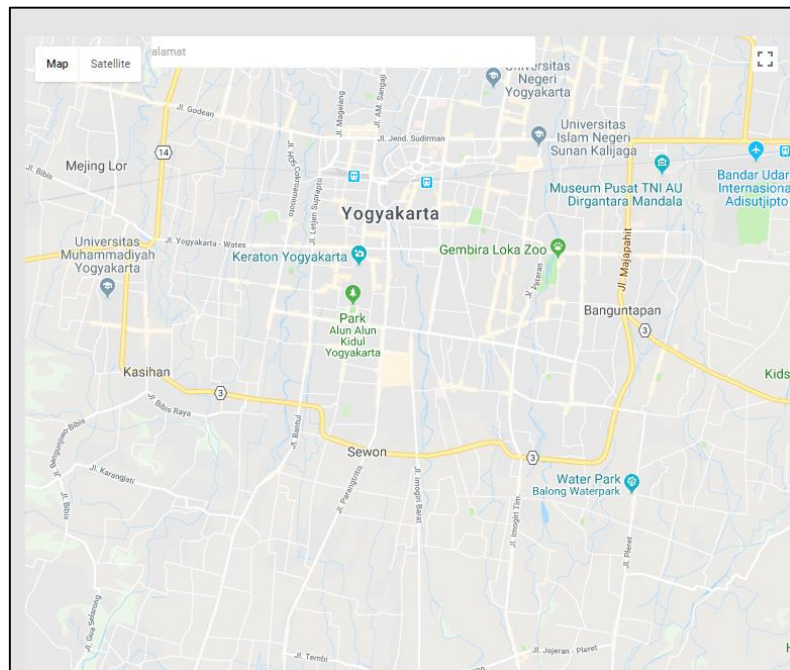


Gambar 5.5 Antarmuka halaman tambah data pengiriman

Gambar 5.5 merupakan halaman untuk admin menambahkan data pengiriman hewan untuk nantinya diproses oleh kurir. Pada halaman ini terdapat form pengiriman yang isinya

berupa alamat pengiriman, jenis muatan yang berupa daging atau hewaan, jenis hewan yang berupa kambing atau sapi, berat muatan, jumlah jamaah di tempat pengiriman, serta dari peternakan mana seharusnya muatan itu diambil oleh kurir. Adminpun juga memasukkan titik koordinat yang akan menjadi acuan agar sistem dapat mengolah pengiriman yang sesuai untuk kurir.

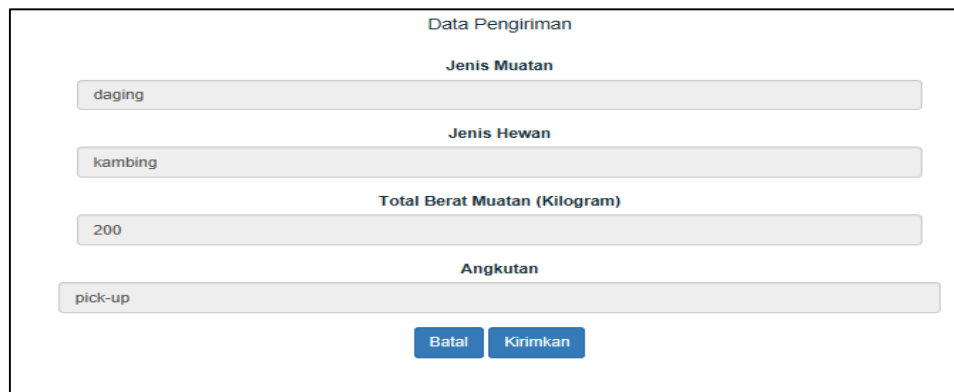
5.1.6 Antarmuka Halaman *Maps*



Gambar 5.6 Antarmuka halaman *maps* merepresentasikan peta

Gambar 5.6 merupakan halaman *maps* ditampilkan oleh sistem dalam *form* penambahan data. Hal ini memudahkan untuk penambahan titik koordinat ketika memasukkan alamat pada form yang tersedia pada bagian atas *maps*.

5.1.7 Antarmuka Halaman Data Pengiriman yang akan Dikirimkan

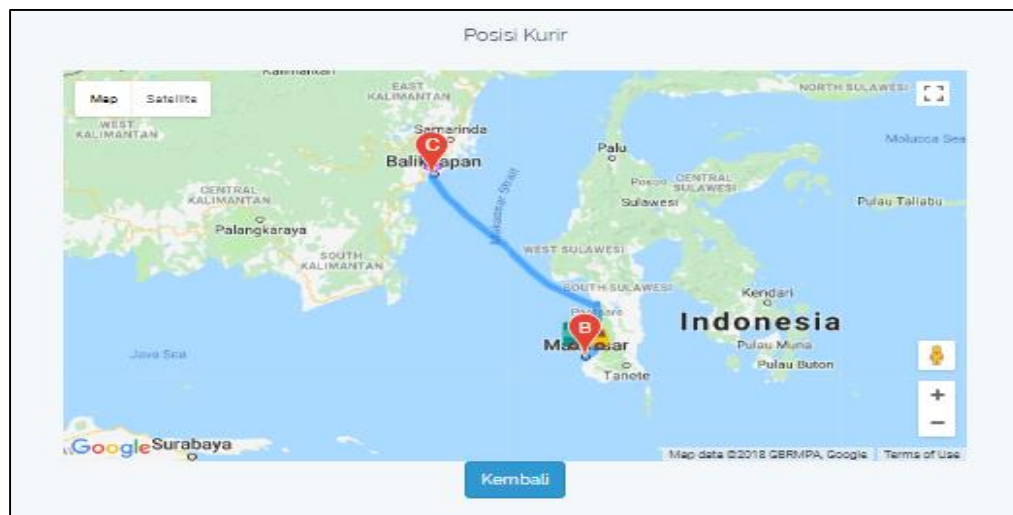


The screenshot shows a web form titled "Data Pengiriman". It contains four input fields, each with a label above it: "Jenis Muatan" (containing "daging"), "Jenis Hewan" (containing "kambing"), "Total Berat Muatan (Kilogram)" (containing "200"), and "Angkutan" (containing "pick-up"). At the bottom of the form are two buttons: "Batal" and "Kirimkan".

Gambar 5.7 Antarmuka halaman data pengiriman yang akan dikirimkan

Gambar 5.7 merupakan halaman yang menampilkan data pengiriman yang telah dimasukkan oleh admin. Data pengiriman ini akan muncul ketika kurir telah memilih pengiriman mana yang akan dikirim dari banyaknya pengiriman yang harus dikirimkan pada daftar pengiriman yang telah dimasukkan oleh admin. Sistem memberikan detail pengiriman kepada kurir dan secara otomatis sistem juga memberikan jenis kendaraan yang harus digunakan yang disesuaikan dengan berat pengiriman.

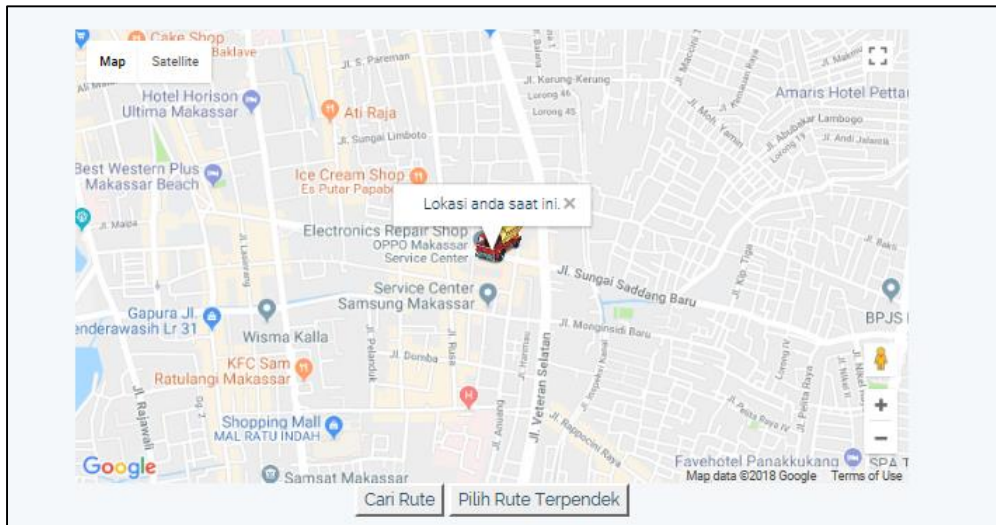
5.1.8 Antarmuka Halaman Tracking



Gambar 5.8 Antarmuka halaman *tracking*

Gambar 5.8 merupakan halaman yang ditampilkan sistem untuk admin melakukan *tracking* posisi kurir yang mengirimkan. Hal ini memudahkan kurir untuk mengetahui posisi kurir secara *real time* dalam pengantaran hewan kurban.

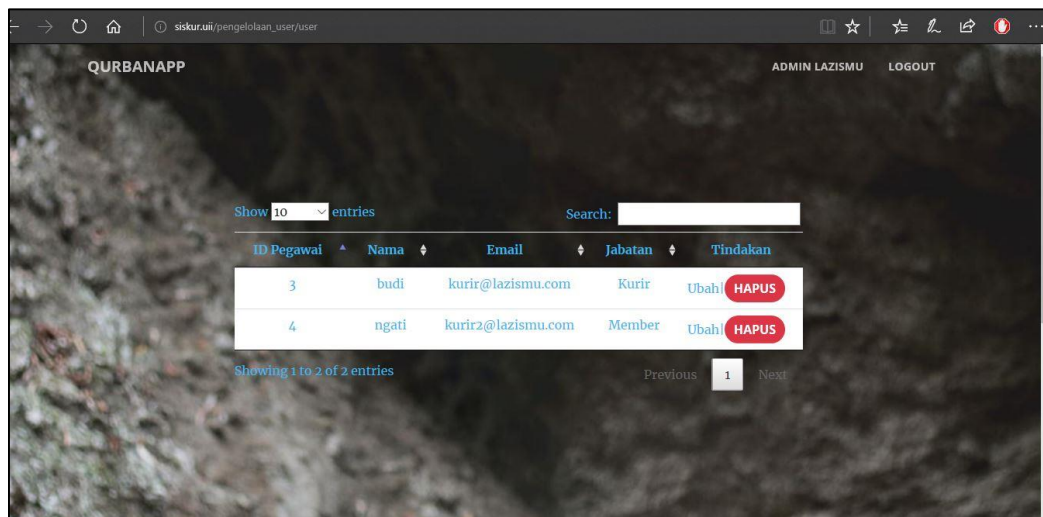
5.1.9 Antarmuka Halaman Maps Posisi Saat Ini (sisi kurir)



Gambar 5.9 Antarmuka halaman yang menunjukkan lokasi saat ini

Gambar 5.9 merupakan halaman yang menampilkan *maps* yang menunjukkan posisi kurir saat ini. Halaman ini ditampilkan dari sisi kurir ketika kurir hendak melakukan pengiriman.

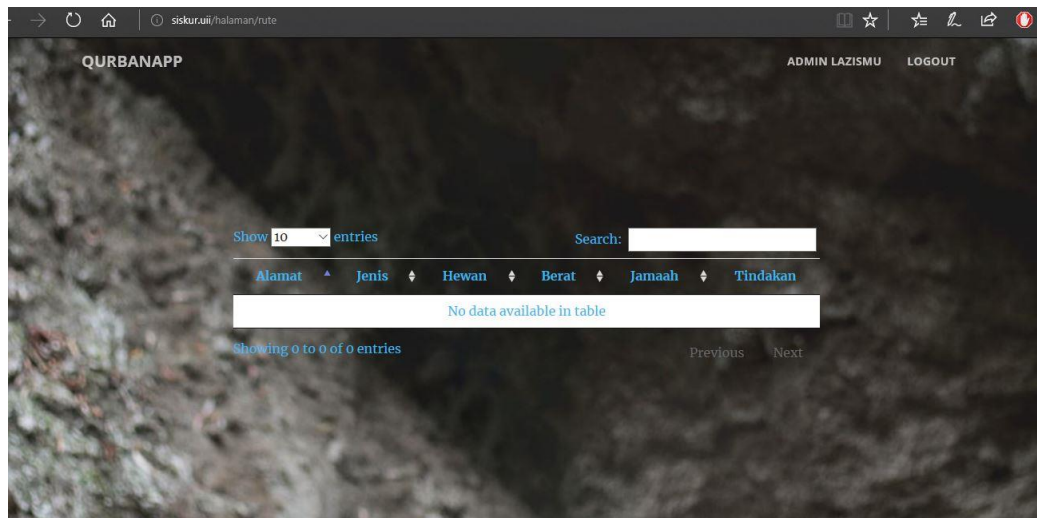
5.1.10 Antarmuka Halaman Data Pengguna (Sisi Admin)



Gambar 5.10 Antarmuka halaman data pengguna

Gambar 5.10 merupakan halaman yang menampilkan data pengguna yang akunnnya terdaftar pada sistem ini. Halaman ini hanya diakses oleh admin. Jadi, hanya admin yang dapat melakukan aksi pada halaman ini.

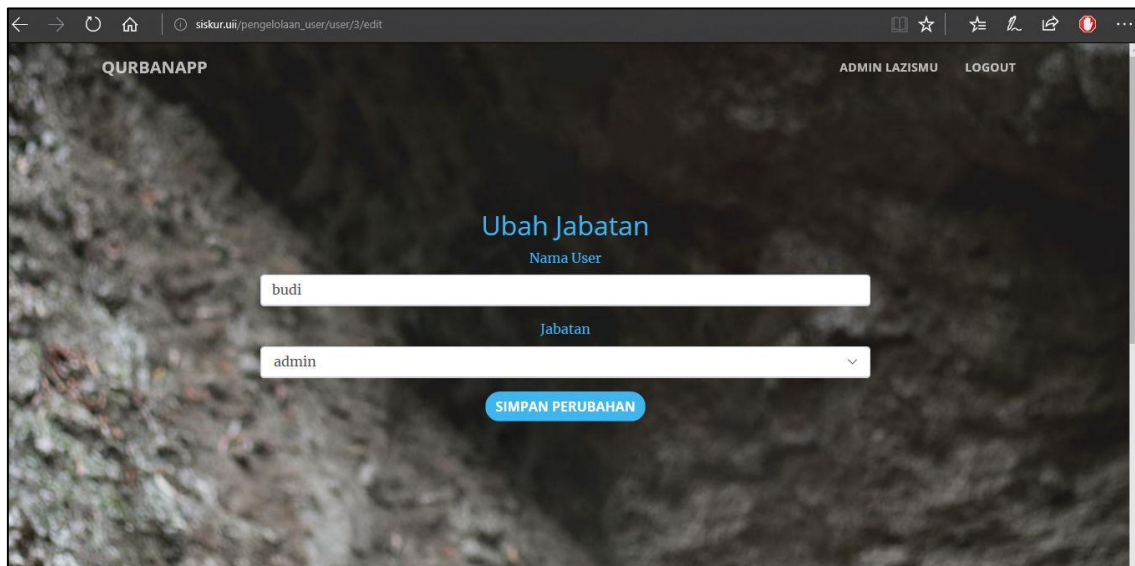
5.1.11 Antarmuka Halaman Data Pengiriman



Gambar 5.11 Antarmuka halaman data pengiriman

Gambar 5.11 merupakan halaman yang menampilkan daftar pengiriman yang telah dimasukkan oleh admin dan akan diproses kurir nantinya. Pada halaman ini, pengguna dapat melakukan pencarian data pengiriman pada kolom *search*.

5.1.12 Antarmuka Halaman Ubah Jabatan (Sisi Admin)

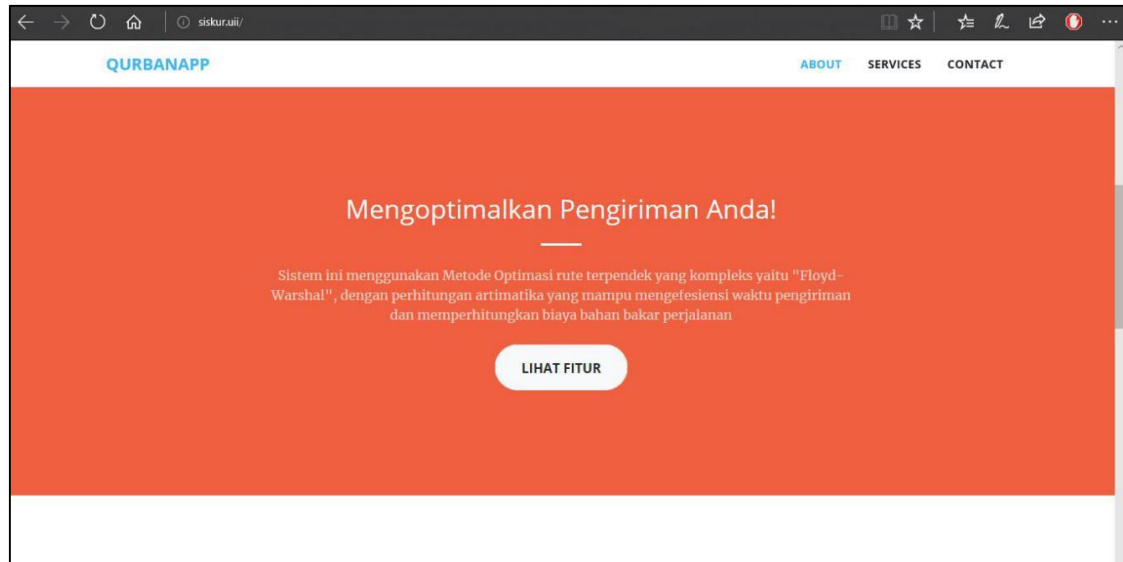


Gambar 5.12 Antarmuka halaman ubah jabatan

Gambar 5.12 merupakan halaman untuk admin mengubah jabatan pada akun yang telah terdaftar. Jika terdapat jabatan yang salah atau terdapat perubahan jabatan, maka admin akan

mengubahnya pada sistem ini sehingga pengguna dapat mengakses sistem sesuai *rolenya* masing-masing.

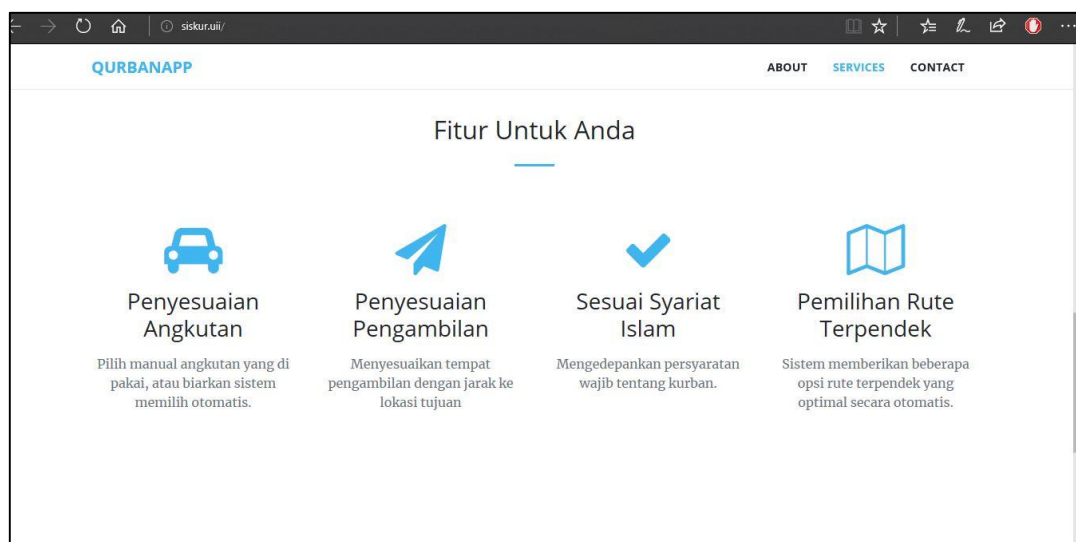
5.1.13 Antarmuka Halaman Tentang



Gambar 5.13 Antarmuka halaman tentang

Gambar 5.13 merupakan halaman yang menampilkan tentang sistem ini. Halaman ini dapat dibaca semua pengguna. Pengguna dapat menekan tombol lihat fitur untuk melihat fitur-fitur apa saja yang ada pada sistem ini.

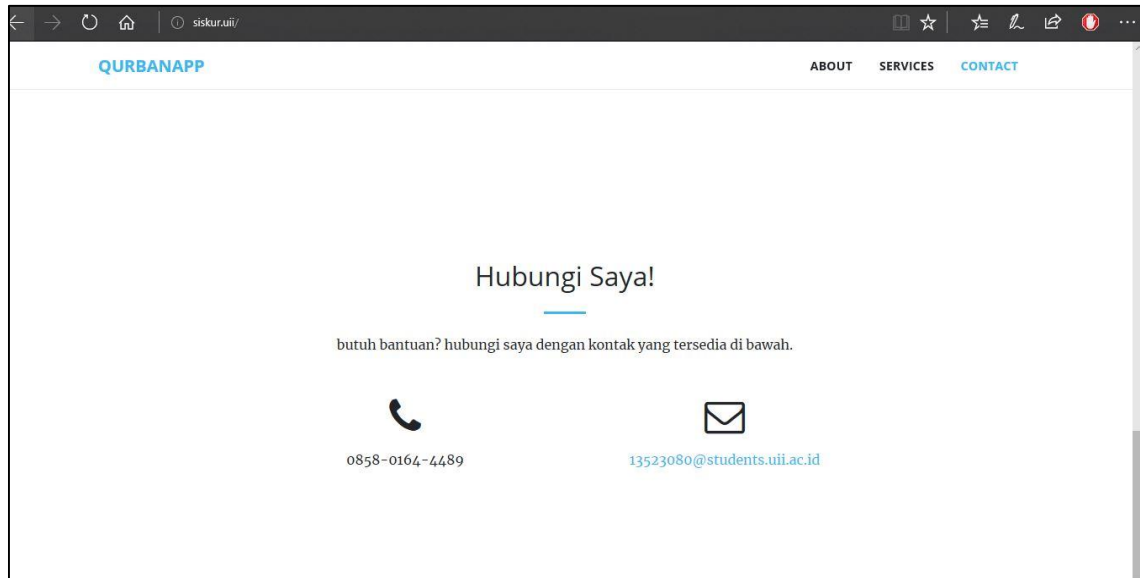
5.1.14 Antarmuka Halaman Layanan



Gambar 5.14 Antarmuka halaman layanan

Gambar 5.14 merupakan halaman yang menampilkan fitur atau layanan apa saja yang terdapat pada sistem ini. Halaman ini dapat dibaca oleh semua pengguna.

5.1.15 Antarmuka Halaman Kontak



Gambar 5.15 Antarmuka halaman kontak

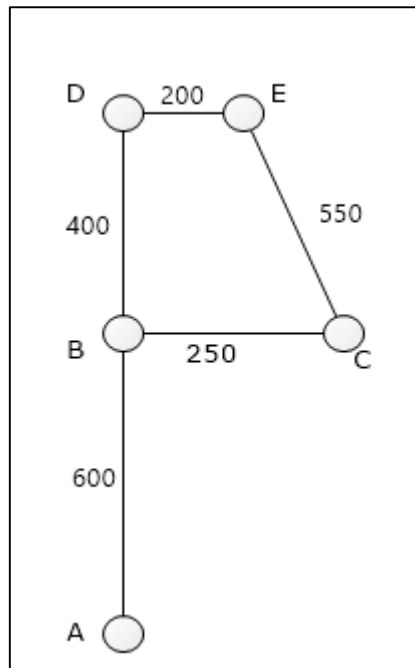
Gambar 5.15 merupakan halaman yang menampilkan kontak yang dapat dihubungkan ketika ada yang perlu diketahui mengenai pengiriman hewan kurban pada sistem ini. Halaman ini dapat diakses oleh semua pengguna.

5.2 Pengujian Rute Terpendek

Berikut ini merupakan pembahasan perhitungan *Floyd warshall* yang manual dan akan dibahas juga mengenai beberapa *source code* dan *user interface* yang menggunakan algoritma *Floyd warshall*. Di bawah ini adalah pengujian komputasi untuk pencarian rute terdekat dari jalan Cendrawasih Yogyakarta menuju jalan Laksda Adisutjipto Yogyakarta menggunakan algoritma *Floyd warshall*.

- a. Langkah pertama ialah membuat grafik yang akan dijadikan sebuah tabel atau matriks.

Berikut ini adalah penggambaran graf awal sebelum ditemukannya rute terdekat.



Gambar 5.16 Pembuatan Graf Awal

Algoritma:

1. $R = R_1$
2. r^n
3. Untuk $k = 1$ hingga n , lakukan :
 Untuk $i = 1$ hingga n , lakukan :
 Untuk $j = 1$ hingga n , lakukan :
 Jika $R[i,j] > R[i,k] + R[k,j]$ maka tukar $R[i,j]$ dengan $R[i,k] + R[k,j]$.
4. $R^* = R$

Keterangan :

R_1 = matriks keterhubungan graf berarah berbobot awal

R^* = matriks keterhubungan minimal

$R_{i,j}$ = nilai dari titik v_i ke v_j

- b. Berdasarkan graf awal yang telah dibuat, maka selanjutnya didapatkan hasil tabel matriks dalam penentuan rute terdekat. Tabel matriks dapat dilihat pada Tabel 5.1.

Tabel 5.1 hasil tabel matriks dari graf awal

Dari,Ke	A	B	C	D	E
A	0	600	∞	∞	∞
B	600	0	250	400	∞
C	∞	250	0	∞	550
D	∞	400	∞	0	200
E	∞	∞	550	200	0

Huruf A sampai E pada bagian yang ke bawah merupakan titik awal. Kemudian huruf A sampai E pada bagian atas ialah titik tujuan. Jika dalam tabel tersebut terdapat lima buah titik yaitu A, B, C, D, E, maka akan ada enam proses yang akan dilewati yaitu R0, R1, R2, R3, R4, dan R5 sebagai hasil akhirnya. Hasil dari sebuah proses akan digunakan untuk proses berikutnya.

R = iterasi

r = indeks dari node A, B, C, D, E

R1, R2, R3, R4, dimana R4= hasil akhir dan R5=kesimpulan

- c. Selanjutnya didapatkan hasil perhitungan berdasarkan rumus. Berikut ini ialah perhitungan iterasi pertama.

R1:			
Ar = 0	Br = 600	Cr = ∞	Dr = ∞
<u>A</u> (A) = 0	B(A) = 600	C(A) = ∞	D(A) = ∞
<u>A</u> (B) = 600	600 + 0 = 600	∞ + 0 = ∞	∞ + 0 = ∞
0 + 600 = 600	<u>B</u> (B) = 0	C(B) = 250	D(B) = 400
A(C) = ∞	600 + 600 = 1200	∞ + 600 = ∞	∞ + 600 = ∞
0 + ∞ = ∞	B(C) = 250	C(D) = ∞	D(C) = ∞
<u>A</u> (D) = ∞	600 + ∞ = ∞	∞ + ∞ = ∞	∞ + ∞ = ∞
0 + ∞ = ∞	<u>B</u> (D) = 400	C(C) = 0	D(D) = 0
<u>A</u> (E) = ∞	600 + ∞ = ∞	∞ + ∞ = ∞	∞ + ∞ = ∞
A(E) = ∞	<u>B</u> (E) = ∞	C(E) = 550	D(E) = 200
0 + ∞ = ∞	600 + ∞ = ∞	∞ + ∞ = ∞	∞ + ∞ = ∞
Er = ∞			
<u>E</u> (A) = ∞			
∞ + 0 = ∞			
<u>E</u> (B) = ∞			
∞ + 600 = ∞			
E(C) = 550			
∞ + ∞ = ∞			
<u>E</u> (D) = 200			
∞ + ∞ = ∞			
<u>E</u> (E) = 0			
∞ + ∞ = ∞			

R1 = 1.1=0, 1.2=600, 1.3= ∞ , 1.4= ∞ , 1.5= ∞

Tidak terjadi perubahan karena di iterasi ini antara hasil hitung dan jarak sebenarnya tidak didapatkan nilai terendah.

Gambar 5.17 Perhitungan iterasi pertama

- d. Dari perhitungan tersebut, didapatkanlah lagi tabel matriks yang kedua. Tabel matriks kedua dapat dilihat pada Tabel 5.2.

Tabel 5.2 hasil tabel matriks kedua

Dari, Ke	A	B	C	D	E
A	0	600	∞	∞	∞
B	600	0	250	400	∞
C	∞	250	0	∞	550
D	∞	400	∞	0	200
E	∞	∞	550	200	0

- e. Selanjutnya, dilakukan lagi perhitungan berdasarkan rumus. Dan didapatkanlah hasil iterasi kedua.

<p>R2:</p> <p>Ar = 600</p> <p>A(A) = 0</p> <p>600+600= 1200</p> <p>A(B) = 600</p> <p>600+0 = 600</p> <p>A(C) = ∞ diganti 850</p> <p>600+250 = 850</p> <p>A(D) = ∞ diganti 1000</p> <p>600+400 = 1000</p> <p>A(E) = ∞</p> <p>600 + ∞ = ∞</p>	<p>Br = 0</p> <p>B(A) = 600</p> <p>0 + 600 = 600</p> <p>B(B) = 0</p> <p>0+0 = 0</p> <p>B(C) = 250</p> <p>0+250 = 250</p> <p>B(D) = 400</p> <p>0 + 400 = 400</p> <p>B(E) = ∞</p> <p>0+∞ = ∞</p>	<p>Cr = 250</p> <p>C(A) = ∞ diganti 850</p> <p>250 + 600 = 850</p> <p>C(B) = 250</p> <p>250 + 0 = 250</p> <p>C(C) = 0</p> <p>250+250 = 500</p> <p>C(D) = ∞ diganti 650</p> <p>250+ 400= 650</p> <p>C(E) = 550</p> <p>250 + ∞ = ∞</p>
<p>Dr = 400</p> <p>D(A) = ∞ diganti 1000</p> <p>400 + 600 = 1000</p> <p>D(B) = 400</p> <p>400 + 0 = 400</p> <p>D(C) = ∞ diganti 650</p> <p>400 + 250 = 650</p> <p>D(D) = 0</p> <p>400 + 400 = 800</p> <p>D(E) = 200</p> <p>400+∞=∞</p>	<p>Er = ∞</p> <p>E(A) = ∞</p> <p>∞ + 600 = ∞</p> <p>E(B) = ∞</p> <p>∞ + 0 = ∞</p> <p>E(C) = 550</p> <p>∞ + 250 = 550</p> <p>E(D) = 200</p> <p>∞ + 400 = ∞</p> <p>E(E) = 0</p> <p>∞+∞=∞</p>	<p>R2: 2.1=600, 2.2=0, 2.3=<u>250</u>, 2.4=400, 2.5=∞</p> <p>Setelah iterasi kedua dilakukan, didapatkan nilai terendah untuk A ke C, A ke D, C ke D, D ke A, D ke C.</p>

Gambar 5.18 Perhitungan iterasi kedua

- f. Dari perhitungan tersebut, didapatkanlah lagi tabel matriks yang ketiga. Berikut ini ialah tabelnya.

Tabel 5.3 hasil tabel matriks ketiga

Dari,Ke	A	B	C	D	E
A	0	600	850	1000	∞
B	600	0	250	400	∞
C	850	250	0	650	550
D	1000	400	650	0	200
E	∞	∞	550	200	0

- g. Selanjutnya, dilakukan lagi perhitungan berdasarkan rumus. Dan didapatkanlah hasil iterasi ketiga.

R3:		
Ar = 850	Br = 250	Cr = 0
A(B) = 600	B(A) = 600	C(A) = 850
850+250 = 1100	250 + 850 = 1100	0 + 850 = 850
A(A) = 0	B(B) = 0	C(B) = 250
850+850 = 1700	250+250 = 500	0+250 = 250
A(C) = 850	B(C) = 250	C(C) = 0
850+0 = 850	250+0 = 250	0+0 = 0
A(D) = 1000	B(D) = 400	C(D) = 650
850+650 = 1500	250+650 = 900	0+650 = 650
A(E) = ∞ diganti 1400	B(E) = ∞ diganti 800	C(E) = 550
850+550=1400	250+550=800	0+550=550
Dr = 650	Er = 550	
D(A) = 1000	E(A) = ∞ diganti 1400	
650+850 = 1500	550+850 = 1400	
D(B) = 400	E(B) = ∞ diganti 800	
650+250 = 900	500+250 = 800	
D(C) = 650	E(C) = 550	
650+0=650	550+0=550	
D(D)=0	E(D)=200	
650+650=1300	550+650=1200	
D(E)=200	E(E)=0	
650+550=1200	550+550=1100	

R3: 3.1=850, 3.2=250, 3.3=0, 3.4=650, 3.5=550

Setelah iterasi ketiga dilakukan, didapatkan nilai terendah untuk A ke E, B ke E, E ke A, E ke B.

Gambar 5.19 Perhitungan iterasi ketiga

- h. Dari perhitungan tersebut, didapatkanlah lagi tabel matriks yang keempat. Berikut ini ialah tabelnya.

Tabel 5.4 hasil tabel matriks keempat

Dari,Ke	A	B	C	D	E
A	0	600	850	1000	1400
B	600	0	250	400	800
C	850	250	0	650	550
D	1000	400	650	0	200
E	1400	800	550	200	0

- i. Selanjutnya, dilakukan lagi perhitungan berdasarkan rumus, dan didapatkanlah hasil iterasi keempat.

R4: $A_r = 1000$ $A(A) = 0$ $1000+1400 = 2400$ $A(B) = 600$ $1000+400 = 1400$ $A(C) = 850$ $1000 + 650 = 1650$ $A(D)=1000$ $1000+0=1000$ $A(E)=1400$ diganti 1200 $1000+200=1200$ $D_r = 0$ $D(A) = 1000$ $0+1000 = 1000$ $D(B) = 400$ $1000+400 = 1400$ $D(C) = 650$ $1000+650=1650$ $D(D)=0$ $1000+0=1000$ $D(E)=200$ $1000+200=1200$	$B_r = 400$ $B(A) = 600$ $400+1000 = 1400$ $B(B) = 0$ $400+400 = 800$ $B(C) = 250$ $400+650=1050$ $B(D)=400$ $400+0=400$ $B(E)=800$ diganti 600 $400+200=600$ $E_r = 200$ $E(A) = 1400$ diganti 1200 $200 + 1000 = 1200$ $E(B) = 800$ diganti 600 $200+400 = 600$ $E(C) = 550$ $200+650 = 850$ $E(D)=200$ $200+0=200$ $E(E)=0$ $200+200=400$	$C_r = 650$ $C(A) = 850$ $650+1000 = 1650$ $C(B) = 250$ $650+400 = 1050$ $C(C) = 0$ $650+650=1300$ $C(D)=650$ $650+0=650$ $C(E)=550$ $650+200=850$
--	---	--

R4 : 4.1=1000, 4.2=400, 4.3=650
 ,4.4=0, 4.5=200

Setelah iterasi keempat dilakukan, didapatkan nilai terendah untuk A ke E, B ke E, E ke A, E ke B.

Gambar 5.20 Perhitungan iterasi keempat

j. Selanjutnya, didapatkanlah lagi tabel matriks yang kelima. Berikut ini ialah tabelnya.

Tabel 5.5 hasil tabel matriks kelima

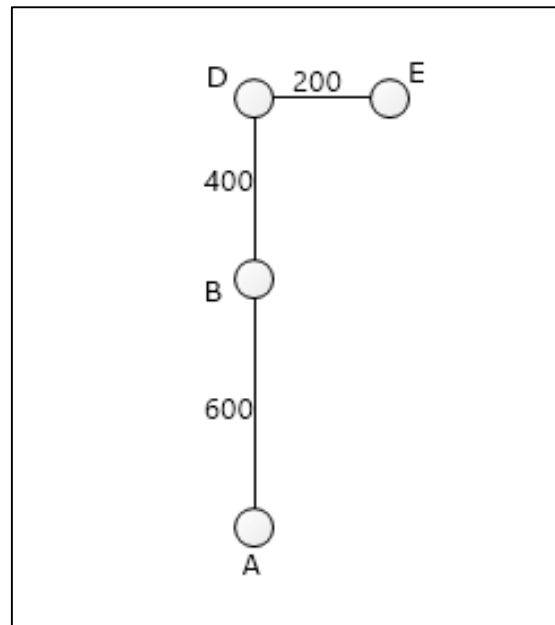
Dari,Ke	A	B	C	D	E
A	0	600	850	1000	1200
B	600	0	250	400	600
C	850	250	0	650	550
D	1000	400	650	0	200
E	1200	600	550	200	0

k. R5 (Kesimpulan):

R5: $A_r = 1200$ $A(A) = 0$ $1200+1200 = 2400$ $A(B) = 600$ $1200+600 = 1800$ $A(C) = 850$ $1200 + 550 = 1750$ $A(D)=1000$ $1200+200=1400$ $A(E)=1200$ $1200+0=1200$ $D_r = 200$ $D(A) = 1000$ $200+1200 = 1400$ $D(B) = 400$ $200+600 = 800$ $D(C) = 650$ $200+550=750$ $D(D)=0$ $200+200=400$ $D(E)=200$ $200+0=1200$	$B_r = 600$ $B(A) = 600$ $600+1200 = 1800$ $B(B) = 0$ $600+600 = 1200$ $B(C) = 250$ $600+550=1150$ $B(D)=400$ $600+200=800$ $B(E)=600$ $600+0=600$ $E_r = 0$ $E(A) = 1200$ $0 + 1200 = 1200$ $E(B) = 600$ $0+600 = 600$ $E(C) = 550$ $0+550 = 550$ $E(D)=200$ $0+200=200$ $E(E)=0$ $0+0=0$	$C_r = 550$ $C(A) = 850$ $550+1200 = 1750$ $C(B) = 250$ $550+600 = 1150$ $C(C) = 0$ $550+550=1100$ $C(D)=650$ $550+200=750$ $C(E)=550$ $550+0=550$
--	---	--

Pada iterasi ini tidak ada perubahan, jadi perhitungan *floyd warshall* dinyatakan selesai.

Gambar 5.21 Kesimpulan



Gambar 5.22 Graf akhir

Gambar 5.22 merupakan graf akhir yang didapatkan dari perhitungan *floyd warshall*. Jika titik awal adalah A dan titik akhir adalah E, maka $A \text{ ke } E = 600 + 400 + 200 = 1200$. Dari iterasi ini tak lagi ditemukan perubahan nilai jarak sebenarnya, jadi telah didapatkan semua kemungkinan nilai terendah dari semua *node* ke semua *node*. Setelah perhitungan manual, selanjutnya akan dilakukan pembahasan pada sistem.

1. Pembahasan pada Sistem

Dari perhitungan manual yang telah dilakukan sebelumnya, selanjutnya akan dilakukan pembuktian ke sistem seperti berikut:

▷ value
matrix 0 1 1234
▷ value ,1234
matrix 0 2 0
▷ value ,1234,0
matrix 0 3 870
▷ value ,1234,0,870,
matrix 1 2 1234
▷ value ,1234,0,870,,,1234
matrix 1 3 506

Gambar 5.23 Nilai 1

▸ value ,1234,0,870,,,1234,506
matrix 1 4 0
▸ value ,1234,0,870,,,1234,506,0,
matrix 2 3 870
▸ value ,1234,0,870,,,1234,506,0,,,870
matrix 2 4 1234
▸ value ,1234,0,870,,,1234,506,0,,,870,1234,
matrix 3 4 506

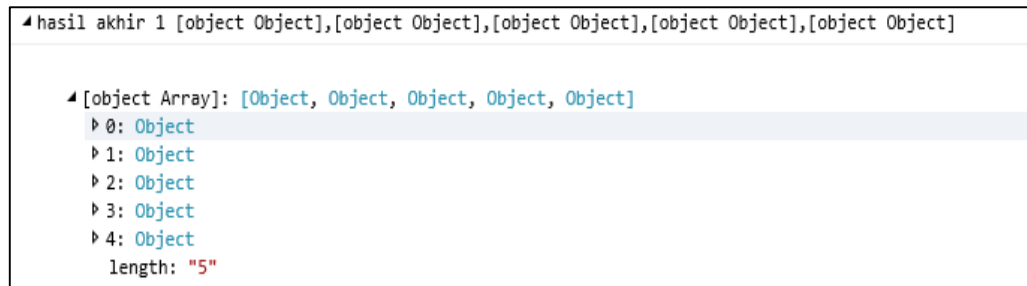
Gambar 5.24 Nilai 2

Gambar 5.23 dan gambar 5.24 merupakan penjelasan tentang sistem yang mendaftarkan nilai tiap jarak ke dalam sebuah index *array* secara menyeluruh dan menghitungnya, lalu jika ditemukan nilai nol maka sistem akan mengabaikan titik itu karena nilai hitung antara suatu titik dengan dirinya akan tetap bernilai nol. Selanjutnya, algoritma *floyd warshall* hanya mengganti nilai jika kalkulasi sistem menunjukan angka lebih kecil dari pada nilai jarak sebenarnya.

▲ nilai matrix [object Object],[object Object],[object Object],[object Object],[object Object]
▸ [object Array]: [Object, Object, Object, Object, Object]
▲ nilai matrix [object Object],[object Object],[object Object],[object Object],[object Object]
▸ [object Array]: [Object, Object, Object, Object, Object]
▲ nilai matrix [object Object],[object Object],[object Object],[object Object],[object Object]
▸ [object Array]: [Object, Object, Object, Object, Object]

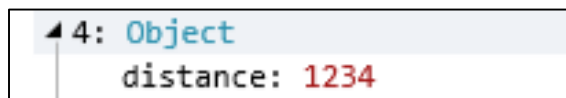
Gambar 5.25 Nilai 3

Gambar 5.25 merupakan penjelasan tentang *Google Maps API*. Dikarenakan keterbatasan penggunaan *Google Maps API* pada akun gratis, untuk menghemat *query Google Maps API*, sistem tidak merinci secara keseluruhan nilai yang dihitung dan hanya akan ditampilkan object dari index *array*.



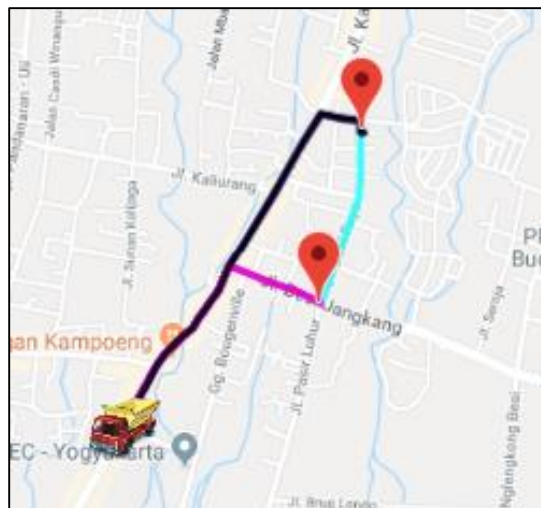
Gambar 5.26 Index array

Gambar 5.26 menjelaskan bahwa sistem menunjukkan penghitungan hingga iterasi ke 5, dan iterasi ke 5 adalah hasil akhir dari penghitungan jalur terdekat, sama dengan penghitungan manual yang juga berakhir di iterasi ke 5.

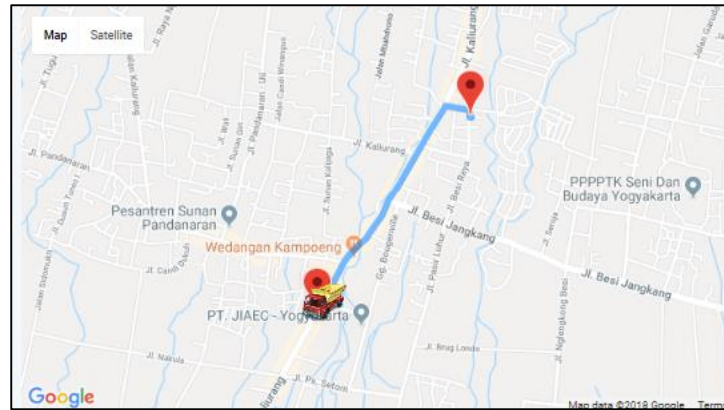


Gambar 5.27 Nilai hasil

Gambar 5.27 menjelaskan tentang didapatkannya nilai hasil "1234 meter" yang berbeda 34 meter dari penghitungan manualnya, ini terjadi karena sedikit pergeseran lokasi *GPS user* yang *update* lokasi setiap 5 detik.



Gambar 5.28 Rute awal



Gambar 5.29 Rute akhir

Gambar 5.28 merupakan rute awal sebelum perhitungan dan gambar 5.29 merupakan rute akhir setelah dilakukan perhitungan. Dari gambar tersebut telah dilihat perbedaannya sebelum dan setelah ditemukannya rute terdekat dengan algoritma *floyd warshall*.

2. Pembahasan Source Code dan Skema Sistem

Pada bagian ini akan dibahas mengenai skema sistem dan *source code* inti yang digunakan dalam sistem ini. *Source code* tersebut menunjukkan beberapa kode program yang dapat menjalankan perintah-perintah pada sistem terutama dalam penggunaan *Google Maps API*. *Source code* akan dijelaskan pada gambar 5.30 sampai 5.28.

```
1. map = new google.maps.Map(document.getElementById('map'), {
2.     center: {lat: -7.795580, lng: 110.369490},
3.     zoom: 15,
4.     mapTypeId:google.maps.MapTypeId.ROADMAP
5. });
```

Gambar 5.30 *Source code* menampilkan canvas peta

```
1. if (navigator.geolocation) {
2.     navigator.geolocation.getCurrentPosition(function(position) {
3.         pos = {
4.             lat: position.coords.latitude,
5.             lng: position.coords.longitude
6.         };
7.     var truck = {
8.         url: "/images/truck.png",
9.         size: new google.maps.Size(71, 71)
10.        origin: new google.maps.Point(0,0)
11.        anchor: new google.maps.Point(17,34)
12.        scaledSize: new google.maps.Size(45,45)
```

```

13.     };
14.     mark1 = new google.maps.Marker({
15.         position: pos,
16.         map: map,
17.         icon: truck
18.     });
19.     map.setCenter(pos);
20.     create_direction();
21.     }, function() {
22.         handleLocationError(true, infoWindow, map.getCenter());
23.     });
24. } else {
25.     handleLocationError(false, infoWindow, map.getCenter());
26. }
27. click_drag_Marker();
28. }

```

Gambar 5.31 *Source code* Geolokasi

- Pada gambar 5.30, baris 1: menginisialisasi *API google maps* untuk memunculkan canvas peta.
- Pada gambar 5.30, baris 2 sampai 4: memusatkan gambar peta pada koordinat Yogyakarta dengan skala gambar 15 dan memunculkan atribut jalan.
- Pada gambar 5.31 baris 1 sampai 6: menggunakan fungsi geolokasi milik Google Maps API untuk mengambil titik koordinat kurir saat menggunakan aplikasi dan menyimpan posisi tersebut ke variabel "pos".
- Pada gambar 5.31 baris 14 sampai 18: membuat *marker* untuk *interface* informasi posisi dirinya saat ini.

```

1. function click_drag_Marker() {
2.     google.maps.event.addListener(map, 'click', function(event) {
3.         marker = new google.maps.Marker({
4.             map: map,
5.             draggable: true,
6.             position: event.latLng,
7.             id: counter,
8.             animation: google.maps.Animation.DROP
9.         });
10.         counter++;
11.         markers.push(marker);
12.         var tes=[];

```

```

13.         tes[marker.id] = {
14.             lat: event.latLng.lat(),
15.             lng: event.latLng.lng()
16.         };
17.         console.log(marker.id);
18.         console.log(tes[marker.id]);
19.         var infowindow = new google.maps.InfoWindow({
20.             content: '' + counter
21.         });
22.         infowindow.open(map, marker);

23.         add_waypoint();
24.     });

```

Gambar 5.32 *Source code* penandaan titik rute

- Pada gambar 5.32 baris 1 sampai 24 : membuat sebuah fungsi yang akan aktif sebagai metode untuk melakukan penandaan titik rute yang ingin dilalui dan juga fungsi untuk memindahkan *marker* dan menyimpan titik koordinatnya.
- Pada gambar 5.32 baris 11 : memasukan nilai *array* dari titik *marker* yang dimasukkan ke dalam variabel *markers* yang selanjutnya akan diolah di tabel *array*.

```

1. function create_direction(){
2.     var ltLokal = mark1.position.lat();
3.     var lgLokal = mark1.position.lng();
4.     var name_generator = 0;
5.     var verteks = {
6.         'lat'    : ltLokal,
7.         'lng'    : lgLokal,
8.         'name'   : name_generator++
9.     }
10.    titik_array.push(verteks);
11.    for(var j = 0; j < titik_rute.length; j++) {
12.        var verteks = {
13.            'lat' : titik_rute[j].lat,
14.            'lng' : titik_rute[j].lng,
15.            'name' : name_generator++
16.        };
17.        titik_array.push(verteks);

```

```

18.     }
19.     var verteks = {
20.         'lat'    : Lt,
21.         'lng'    : Lg,
22.         'name'   : name_generator
23.     }
24.     titik_array.push(verteks);
25.     console.log("titik_array",titik_array);
26.     var counter_requestArray = 0;
27.     var requestArray = [ ];

```

Gambar 5.33 Source code pembentukan vertex

- Pada gambar 5.33 baris 1-24 : membentuk sebuah *vertex* dari *node* ke *node array*, dari titik awal hingga titik tujuan. *Vertex* ini dimuat dari titik koordinat.

```

1. for(var i = 0; i < titik_array.length; i++) {
2.   for(var j = i+1; j < titik_array.length; j++) {
3.     if (i != 0 || j != titik_array.length-1) {
4.       var request = {
5.         origin : {lat:titik_array[i].lat, lng: titik_array[i].lng},
6.         destination:{lat:titik_array[j].lat,lng:titik_array[j].lng},
7.         travelMode : google.maps.TravelMode.DRIVING,
8.       };
9.       requestArray[counter_requestArray++] = {
10.        'request' : request,
11.        'index1' : i,
12.        'index2' : j
13.      };
14.    };
15.  };
16. };
17. console.log('requestArray',requestArray);
18. direction_request(requestArray);
19. }

```

Gambar 5.34 Source code perhubungan jalur jalan

- Pada gambar 5.34 baris 1-16 : perulangan dibentuk untuk persiapan penyatuan node dari tiap rute lalu menyimpannya nilai ke dalam sebuah index
- Pada gambar 5.34 baris 4-8 : inisialisasi variabel yang dibentuk sistem untuk meminta penghubungan jalur jalan yang kemudian permintaan tersebut akan dikembalikan oleh *Google Maps API* dalam bentuk hubungan jalur jalan dengan mode mengemudi
- Pada gambar 5.34 baris 9-13 : nilai dari tiap *node* yang diterima sistem akan disimpan ke dalam index *array*

```

1. function direction_request(requestArray){
2.     var directionsService = new google.maps.DirectionsService();
3.     var i = 0;
4.     function directionResults(result, status) {
5.         if (status == google.maps.DirectionsStatus.OK) {
6.             renderArray[i] = new google.maps.DirectionsRenderer();
7.             renderArray[i].setMap(map);
8.             renderArray[i].setDirections(result);
9.             renderArray[i].setOptions({
10.                 suppressMarkers: true,
11.                 polylineOptions: {
12.                     strokeWeight: 4,
13.                     strokeOpacity: 0.8,
14.                     strokeColor: colourArray[i]
15.                 },
16.             });
17.             if(typeof matrix[requestArray[i].index1] == 'undefined')
matrix[requestArray[i].index1] = [];
18.             console.log('value',matrix);
19.             matrix[requestArray[i].index1][requestArray[i].index2] =
result.routes[0].legs[0].distance.value;
20.             console.log('matrix '+requestArray[i].index1+"
"+requestArray[i].index2,matrix[requestArray[i].index1][requestArray[i].index2];
21.             next_request();
22.         }else{
23.             alert(status);
24.         }
25.     }
26.     function next_request(){
27.         i++;
28.         if(i >= requestArray.length){
29.             floydW();
30.             return;
31.         }
32.         var delay = 1600;

```

```

33.         setTimeout(function() {
34.     directionsService.route(requestArray[i].request, directionResults);
35.         }, delay);

```

Gambar 5.35 Source code pengolahan data dengan *floyd warshall*

- Pada gambar 5.35 baris 5-35 : *Google Maps API* melakukan pengolahan data untuk merepresentasikan gambar tiap jalur yang memungkinkan untuk dilewati dengan metode mengemudi dan menyimpan nilai jarak dari tiap kelokan yang ada ke dalam index *array*
- Pada gambar 5.35 baris 26-31 : nilai mentah tiap kelokan jalur yang memungkinkan untuk dilewati yang telah diberikan *Google Maps API* diproses oleh sistem dan diteruskan ke metode *floyd warshall*.

```

1. for(var x = 0; x <= matrix.length; x++){
2.     hitung_floydW[x] = {
3.         'distance' : Infinity ,
4.         'from'      : 0
5.     }
6.     prev_hitung_floydW[x] = {
7.         'distance' : Infinity ,
8.         'from'      : 0
9.     }
10.    }
11. if(typeof matrix[requestArray[i].index1] == 'infinity')
matrix[requestArray[i].index1] = [];
12. console.log('value',matrix);
13. matrix[requestArray[i].index1][requestArray[i].index2] =
result.routes[0].legs[0].distance.value;
14. console.log('matrix '+requestArray[i].index1+"
"+requestArray[i].index2,matrix[requestArray[i].index1][requestArray[i].index2]);
15. next_request();

```

Gambar 5.36 Source code aturan index *floyd warshall*

- Pada gambar 5.36 baris 1-10 : menyiapkan aturan index untuk *Floyd warshall*.
- Pada gambar 5.36 baris 11-15: melakukan *request* ke *Google Maps API* untuk melakukan pengecekan kembali jalur untuk setiap titik yang telah di set ke infinity apakah tersedia jalur antara titik yang bernilai infinity dengan titik yang lainnya. Jika ditemukan jalur, maka nilai infinity akan diubah dengan nilai jarak dari titik yang terhubung.

```

1.while(i != matrix.length) {
2.    for(var j = 0; j <= matrix.length; j++) {
3.        if(i != j) {
4.            if(typeof matrix[i][j] != 'undefined') {
5.                if(prev_hitung_floydW[i].distance == Infinity)
prev_hitung_floydW[i].distance = 0;
6.                var distance = matrix[i][j] + prev_hitung_floydW[i].distance;
7.                if(distance < prev_hitung_floydW[j].distance) {
8.                    hitung_floydW[j] = { 'distance' : distance, 'from' : i};
9.                }
10.            }
11.        };
12.    };
13.    var min = 0;
14.    for(var k = 0; k <= matrix.length; k++) {
15.        if(typeof already_boxed[k] == 'undefined') {
16.            if (hitung_floydW[k].distance <= hitung_floydW[min].distance) {
17.                min = k;
18.            }
19.        }
20.    }
21.    already_boxed[min] = true;
22.    i = min;
23.    console.log('nilai matrix', hitung_floydW);
24.    prev_hitung_floydW = hitung_floydW;
25.    }

```

Gambar 5.37 Source code perhitungan

- Pada gambar 5.37 baris 1-25 : melakukan penghitungan dari tiap index array yang berisi nilai jarak yang telah di berikan oleh *Google Maps API* dengan metode *floyd warshall*
- Pada gambar 5.37 baris 8-17 : menyimpan tiap nilai *array* index dan jika menemukan nilai baru yang lebih kecil, nilai yang lebih besar diganti dengan yang lebih kecil.

```

1. hitung_floydW[0].distance = 0;
2.    route = [];
3.    index_route = matrix.length;
4.    route.push(index_route);
5.    while(index_route!=0){
6.        route.push(hitung_floydW[index_route].from);
7.        index_route = hitung_floydW[index_route].from;
8.    }
9.    console.log('hasil akhir 1',hitung_floydW);
10.    console.log('rute',route);

```

```
11.      alert('Jalur Telah di Hitung');
```

Gambar 5.38 Source code penyimpanan nilai terkecil ke dalam *array*

- Pada gambar 5.38 baris 1-11 : dari tiap index yang nilainya telah didapatkan yang paling kecil, nilai tersebut disimpan ke dalam *array* dengan nama variabel "*route*"

```
1. route.forEach(function(ta) {
2.   if (route.indexOf(ta) == 0) {
3.     start = {lat: titik_array[ta].lat, lng: titik_array[ta].lng}
4.   }else if(route.indexOf(ta) == route.length-1){
5.     end = {lat: titik_array[ta].lat, lng: titik_array[ta].lng}
6.   }else{
7.     data = titik_array[ta];
8.     waypts.push({
9.       location: {lat: data.lat, lng: data.lng},
10.      stopover: true });  }
11.   var kur = new google.maps.Marker({
12.     position:{
13.       lat: titik_array[ta].lat,
14.       lng: titik_array[ta].lng  },
15.     map: map, });
16. });
17. var request = {
18.   origin: start,
19.   destination: end,
20.   waypoints: waypts,
21.   travelMode: google.maps.TravelMode.DRIVING};
22. console.log(start,end,waypts);
23. requestArray.push({request});
```

Gambar 5.39 Source code pembuatan jalur terdekat

- Pada gambar 5.39 baris 1-23 : nilai hasil terkecil dari perhitungan *Floyd warshall* disiapkan dan sistem membuat jalur yang kemudian akan dipresentasikan oleh *Google Maps API*.

```

1. function directionResults(result, status) {
2.     if (status == google.maps.DirectionsStatus.OK) {
3.         renderArray[i] = new google.maps.DirectionsRenderer();
4.         renderArray[i].setMap(map);
5.         renderArray[i].setDirections(result);
6.         renderArray[i].setOptions({
7.             suppressMarkers: true,
8.         });

```

Gambar 5.40 Source code perhubungan jalur

- Pada gambar 5.40 baris 1-8 : sistem melakukan penghubungan jalur yang selanjutnya akan diproses oleh *Google Maps API* menjadi direksi jalur dan jalur tersebut dipresentasikan ke peta.

```

1. function next_request(){
2.     i++;
3.     if(i >= requestArray.length){
4.         return;
5.     }
6.     directionsService.route(requestArray[i].request, directionResults); }
7.     directionsService.route(requestArray[i].request, directionResults);
8.     directionsService.route(requestArray[i].request, function(response, status) {
9.         if ( status == google.maps.DirectionsStatus.OK ) {
10.             var total = 0;
11.             var myroute = response.routes[0];
12.             for (var i = 0; i < myroute.legs.length; i++) {
13.                 total += myroute.legs[i].distance.value;
14.             }
15.             total = total / 1000;
16.             document.getElementById('total').value = total;
17.             console.log('total jarak:',total);
18.         }

```

Gambar 5.41 Source code pemberian nilai total jarak

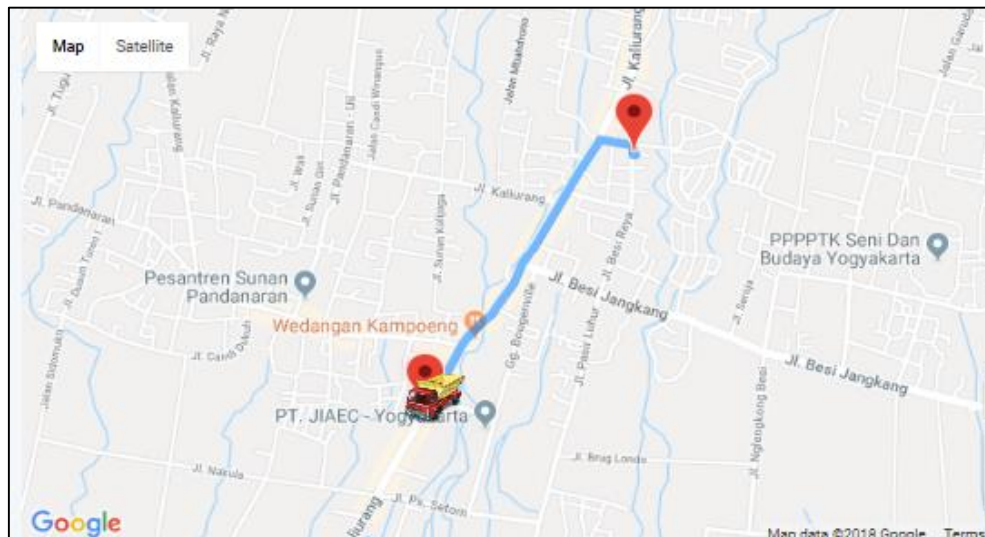
- Pada gambar 5.41 baris 8-18 : memberikan nilai total jarak yang akan dilalui kurir.

5.3 Pengujian Keoptimalan dan Fungsionalitas

Pengujian yang dilakukan adalah pengujian fungsionalitas sistem. Pengujian ini bertujuan untuk menemukan kekurangan atau *bug* yang terdapat pada sistem. Jenis pengujian yang digunakan pada penelitian ini yaitu pengujian keoptimalan sistem, *black box testing*, dan pengujian UAT. Adapun hasil pengujian yang telah dilakukan ialah:

a. Pengujian Keoptimalan

Sistem dikatakan optimal ketika dapat menentukan rute terdekat dari tempat pengambilan hewan ke tempat pendistribusian. Sistem juga dapat memberikan perkiraan perhitungan bahan bakar yang akan digunakan oleh transportasi pengiriman hewan. Pengujian keoptimalan sistem terdapat pada gambar yang merupakan antarmuka halaman jarak tempuh pada Gambar 5.42, Gambar 5.43, dan Gambar 5.44.



Gambar 5.42 Penentuan Rute Terdekat

Nama Penerima

Jarak Tempuh Km

Selesai

Gambar 5.43 Antarmuka halaman *output* rute terdekat

Gambar 5.42 merupakan *form* peta yang menunjukkan rute terdekat dari satu daerah ke daerah tujuan. Sistem menghitung jalur mana yang lebih pendek yang dapat dilalui kurir dan menampilkan total jarak yang akan di tempuh oleh kurir. Dari rute terdekat yang telah tampil tersebut, jarak tempuh yang menunjukkan rute terdekat akan tampil pada halaman seperti yang terlihat pada gambar 5.43. Pada Gambar 5.43 juga terdapat kolom penerima hewan kurban yang harus diisi oleh kurir.



Gambar 5.44 Halaman Perkiraan Bahan Bakar

Gambar 5.44 merupakan antarmuka halaman sistem yang menunjukkan sebuah detail dari hasil salah satu pengiriman. Pada gambar tersebut, salah satu contoh pengiriman dengan perkiraan bahan bakar sebesar 1,07 liter serta mengalkulasikannya dalam bentuk rupiah yaitu 8383,27 rupiah. Harga ini telah disesuaikan dengan harga bahan bakar SPBU (Stasiun Pengisian Bahan Bakar Umum) Pertamina regional Jogja. Adapun source code kalkulasi bahan bakar berdasarkan rute dapat dilihat pada Gambar 5.45.

```

1. if($angkutan == 'pick-up'){
2.     $bb = $jarak/9;
3.     $harga = $bb*7800;
4.     $data = [
5.         'delivery_id' => $request->delivery_id,
6.         'jarak_tempuh' => $request->jarak_tempuh,
7.         'total_biaya' => $harga,
8.         'bahan_bakar' => 'pertalite',
9.         'total_penggunaan_bahan_bakar' => $bb ];
10.    $hasil = Result::create($data);    }
11. elseif ($angkutan == 'truck') {
12.     $bb = $jarak/3.3;
13.     $harga = $bb*8100;
14.     $data = [
15.         'delivery_id' => $request->delivery_id,
16.         'jarak_tempuh' => $request->jarak_tempuh,
17.         'total_biaya' => $harga,
18.         'bahan_bakar' => 'dexlite',
19.         'total_penggunaan_bahan_bakar' => $bb ];
20.    $hasil = Result::create($data);    }

```

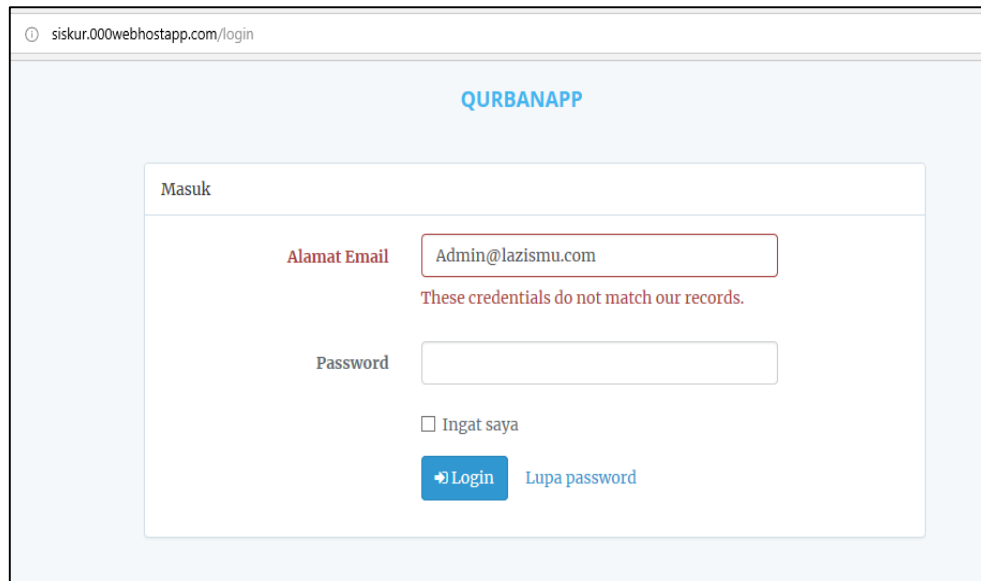
Gambar 5.45 Source code perkiraan dan kalkulasi bahan bakar

- Karena jenis kendaraan yang dipakai untuk pengiriman hanya jenis pick-up dan truk, sistem membuat percabangan untuk mengalkulasikan berapa liter bahan bakar yang akan keluar serta berapa rupiah bahan bakar tersebut.
- Pada Gambar 5.45 baris 2: Untuk mobil *pick up*, angkutan tersebut menggunakan 1 liter bahan bakar untuk berjalan sejauh 9 kilometer, jadi sistem membagi jarak tempuh yang dilalui kurir dengan 9 kilometer lalu didapatkan berapa liter bahan bakar keluar sekali jalan.
- Pada Gambar 5.45 baris 3: Sistem melakukan *default* pada mobil *pick up* menggunakan bahan bakar *pertalite*, harga *pertalite* tahun 2018 regional Jogja adalah 7800 rupiah dengan mengalkulasikan hasil dari proses hitung baris 89 maka didapatkan hasil biaya berapa rupiah biaya yang keluar sekali jalan.
- Pada Gambar 5.45 baris 12: Untuk truk, angkutan tersebut menggunakan 1 liter bahan bakar untuk berjalan sejauh 3,3 kilometer, jadi sistem membagi jarak tempuh yang dilalui kurir dengan 3,3 kilometer lalu didapatkan berapa liter bahan bakar keluar sekali jalan.
- Pada Gambar 5.45 baris 13: Sistem melakukan *default* pada truk menggunakan bahan bakar *dexlite*, harga *dexlite* tahun 2018 regional Jogja adalah 8100 rupiah dengan mengalkulasikan hasil dari proses hitung baris 104 maka didapatkan hasil biaya berapa rupiah yang keluar sekali jalan.

b. Black box testing

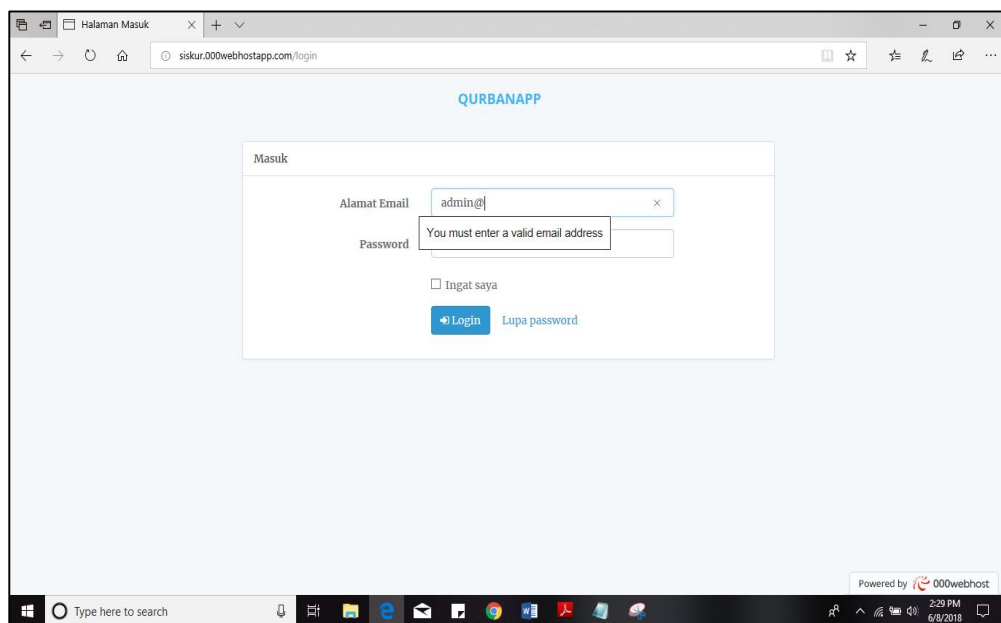
1. Proses *Login* Pengguna

Untuk melakukan *login*, pengguna harus memasukkan *email* dan *password* dengan benar sesuai dengan yang telah terdaftar pada *database*. Jika pengguna memasukkan *email* dan *password* yang salah, maka sistem akan menampilkan notifikasi kesalahan. Notifikasi kesalahan *login* terdapat pada gambar 5.46 sampai 5.48.



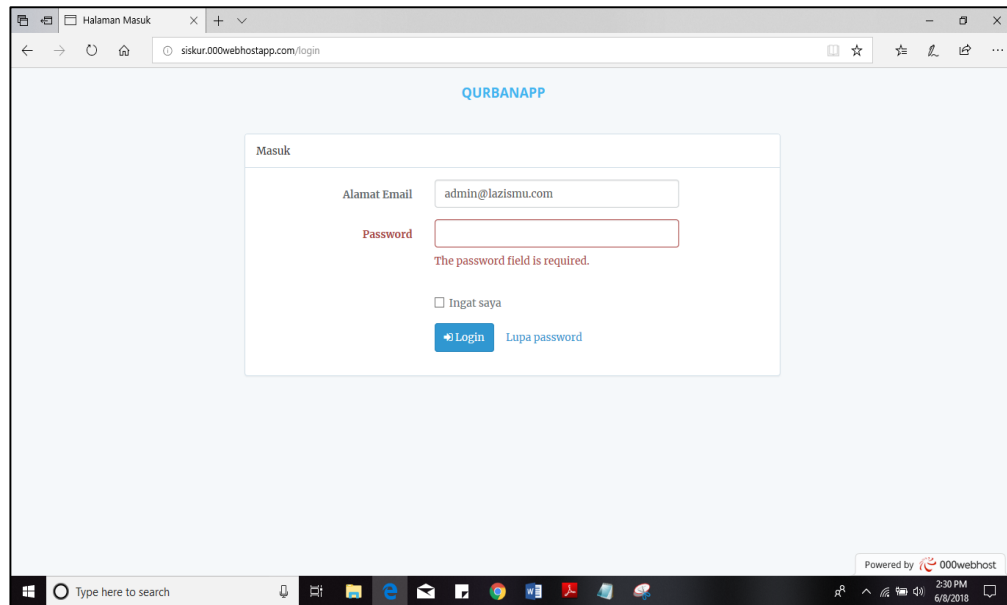
Gambar 5.46 Kesalahan Password

Pada gambar 5.46 merupakan halaman yang menampilkan kegagalan *login*. Kegagalan *login* tersebut disebabkan oleh kesalahan *password* yang dimasukkan pengguna.



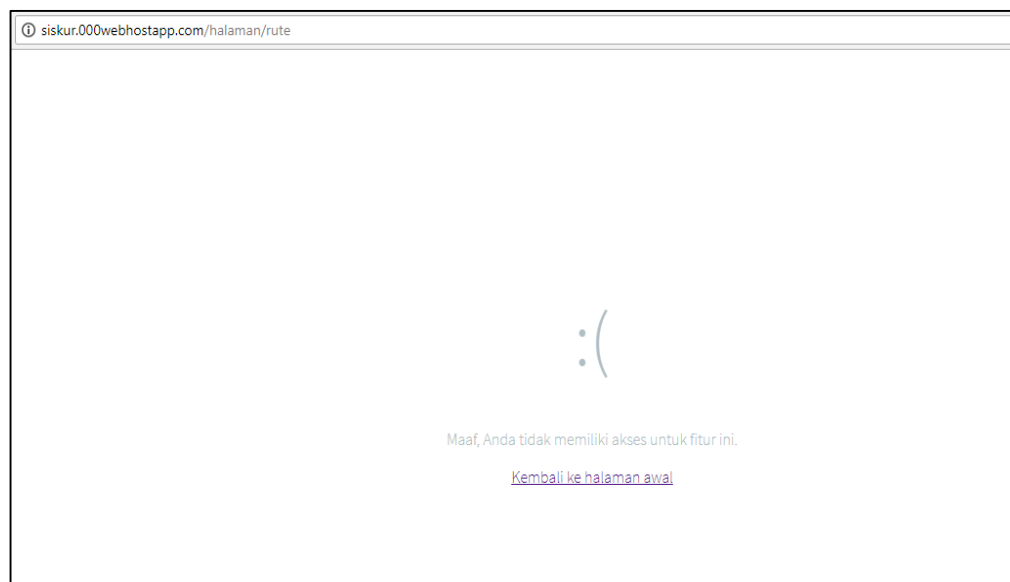
Gambar 5.47 Kesalahan alamat email

Pada gambar 5.47 menampilkan notifikasi kegagalan *login* yang disebabkan oleh kesalahan *email* yang dimasukkan oleh pengguna. Pengguna harus memasukkan *email* yang valid agar dapat melakukan *login*.



Gambar 5.48 Kolom Password kosong

Gambar 5.48 menampilkan kegagalan *login* ketika pengguna belum memasukkan *password*. Kolom *password* dan *email* harus diisi dengan benar agar pengguna dapat melakukan *login*.



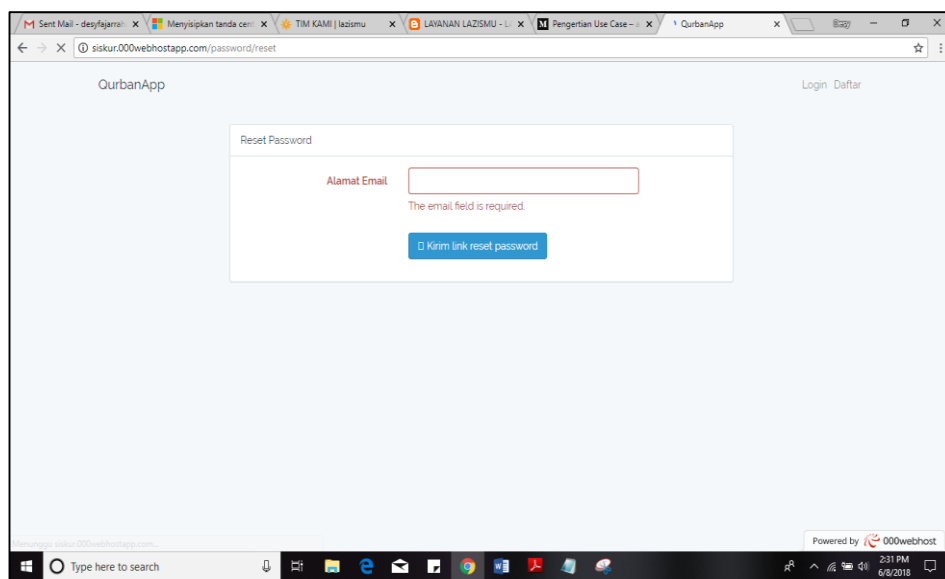
Gambar 5.49 Pelanggaran hak akses

Pada gambar 5.49 menampilkan halaman *404 not found*. Halaman ini menunjukkan bahwa pengguna telah melakukan pelanggaran hak akses. Admin dan kurir memiliki hak

aksesnya masing-masing. Ketika kurir ingin mengakses halaman admin, maka notifikasi tersebut akan muncul pada halaman kurir, begitupun sebaliknya.

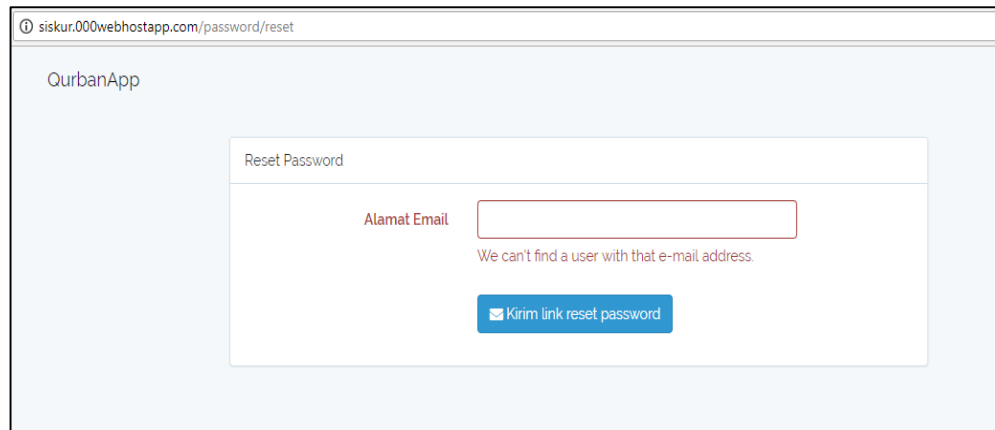
2. Proses *Reset Password*

Ketika pengguna ingin mengubah atau mengganti *password*, maka pengguna perlu masuk ke halaman *reset password* dengan klik link *reset password* pada halaman *login*. Untuk melakukan *reset password*, pengguna harus memasukkan *email* yang telah terdaftar sebelumnya atau *email* yang ingin diubah *password*nya. Berikut kesalahan-kesalahan ketika pengguna melakukan *reset password*.



Gambar 5.50 Kolom email kosong

Pada gambar 5.50 menampilkan halaman *reset password* pengguna jika pengguna ingin mengubah *password*. Ketika pengguna belum memasukkan *email* pada kolom tersebut, kemudian akan muncul notifikasi bahwa kolom *email* harus diisi agar pengguna dapat melakukan *reset password*.

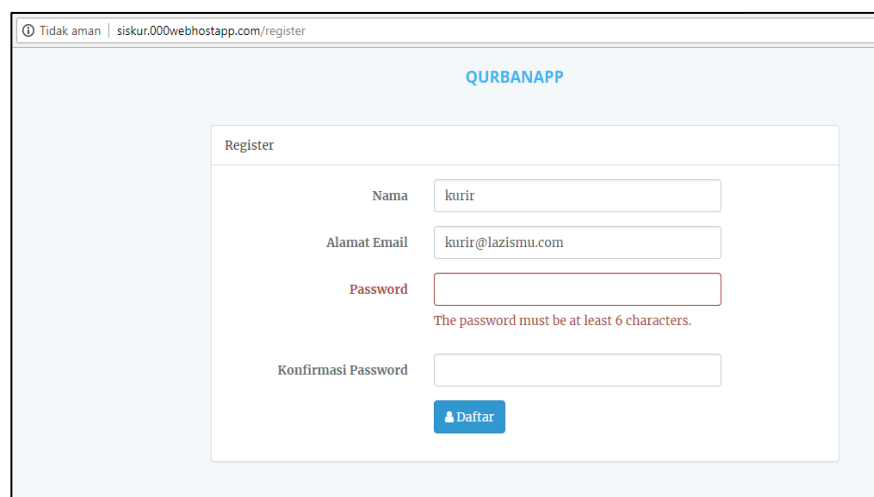


Gambar 5.51 Email tidak terdaftar

Pada gambar 5.51 menampilkan halaman *reset password*. Ketika pengguna ingin merubah *password* namun salah memasukkan *email* atau memasukkan *email* yang tidak terdaftar pada *database*, kemudian akan muncul notifikasi yang menunjukkan bahwa sistem tidak mengenali *email* yang dimasukkan.

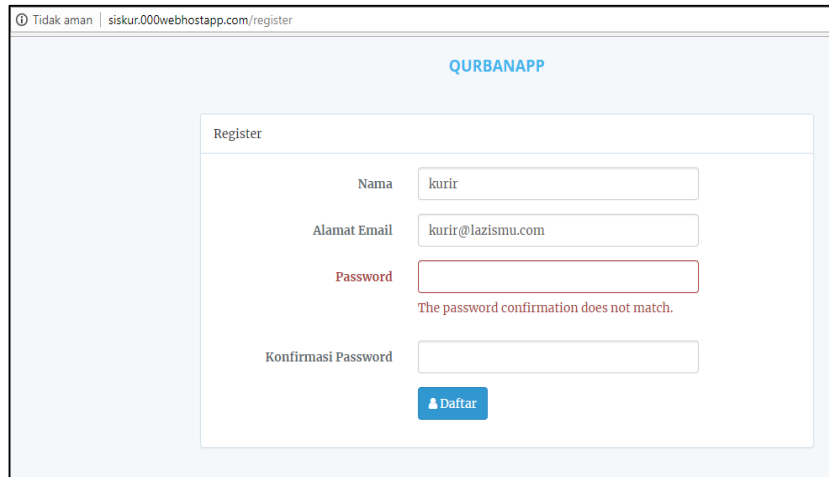
3. Proses Daftar Akun

Ketika pengguna ingin membuat atau mendaftarkan akun baru, maka pengguna harus mengakses menu daftar pada halaman awal sistem. Pengguna harus mendaftarkan nama, *email* dan *password* untuk kemudian disimpan ke *database*. Berikut notifikasi dari kesalahan-kesalahan yang terjadi ketika pengguna melakukan daftar akun.



Gambar 5.52 Password tidak memiliki lebih dari sama dengan 6 karakter

Pada gambar 5.52 menampilkan halaman register atau daftar akun. Jika pengguna mengisi *password* kurang dari 6 karakter, maka sistem akan menampilkan notifikasi yang menyebutkan bahwa pengguna harus memasukkan *password* lebih dari sama dengan enam karakter.



Gambar 5.53 Password awal dan password konfirmasi berbeda

Pada gambar 5.53 menampilkan halaman daftar akun. Ketika pengguna telah memasukkan *password* namun mengisikan konfirmasi *password* yang berbeda dari *password* sebelumnya, maka sistem akan menampilkan notifikasi yang menunjukkan bahwa *password* konfirmasi tidak sesuai dengan *password* yang dimasukkan di awal.



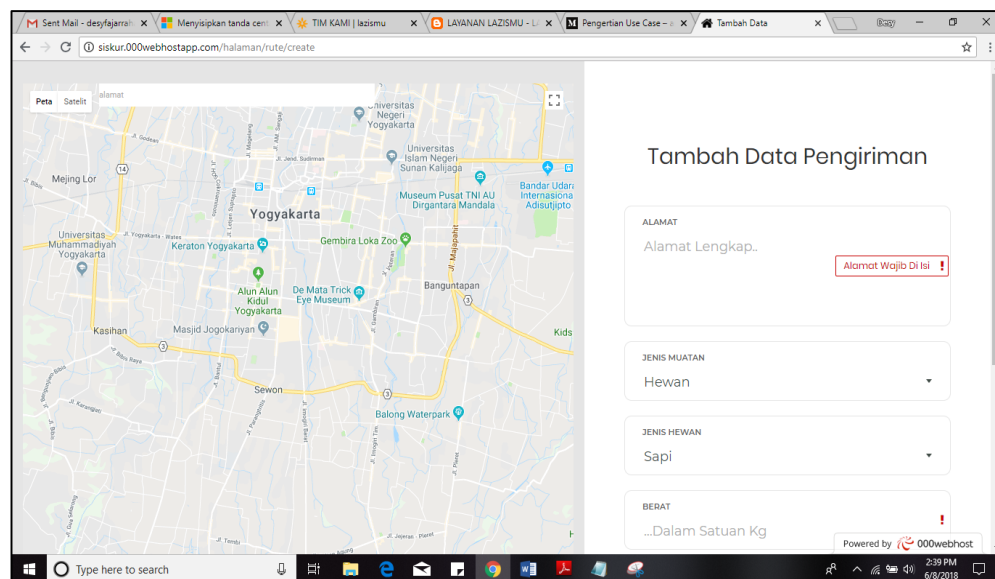
Gambar 5.54 Akun telah terdaftar

Pada gambar 5.54 menampilkan halaman awal atau beranda ketika pengguna telah berhasil mendaftarkan akun pada sistem. Ketika telah berhasil daftar, kemudian akan muncul

notifikasi bahwa pengguna masih berstatus member dan belum berstatus kurir atau admin yang masing-masing memiliki batasan akses sistem.

4. Proses Penambahan Data

Pada halaman tambah data pengiriman hanya dapat dilakukan oleh admin. Pada halaman ini admin perlu memasukkan data pengiriman dengan lengkap agar pengiriman akan lancar diproses oleh kurir. Adapun beberapa notifikasi kesalahan-kesalahan pada tambah data pengiriman ialah sebagai berikut:



Gambar 5.55 Kolom alamat kosong

Pada gambar 5.55 menampilkan halaman tambah data pengiriman yang dilakukan oleh admin. Admin perlu memasukkan data-data dengan lengkap. Jika admin tidak mengisi kolom alamat atau kolom alamat kosong, maka sistem akan menampilkan notifikasi bahwa kolom alamat wajib diisi.

The screenshot shows a web browser window with a map of Bantul, Indonesia, on the left. On the right is a form titled 'Tambah Data'. The form contains the following fields:

- JENIS MUATAN:** Dropdown menu with 'Hewan' selected.
- JENIS HEWAN:** Dropdown menu with 'Sapi' selected.
- BERAT:** Text input field containing '...Dalam Satuan Kg'. A red error message 'Harap Masukkan Berat !' is displayed next to it.
- JUMLAH JAMAAH:** Text input field containing '....Jamaah'. A red error message 'Harap Masukkan Jumlah Jamaah !' is displayed next to it.
- PETERNAKAN:** Dropdown menu with 'AMM Clagah' selected.

At the bottom of the form are two buttons: 'TAMBAH' (Add) and 'BATAL' (Cancel). The browser's address bar shows 'sisur.000webhostapp.com/halaman/rute/create'.

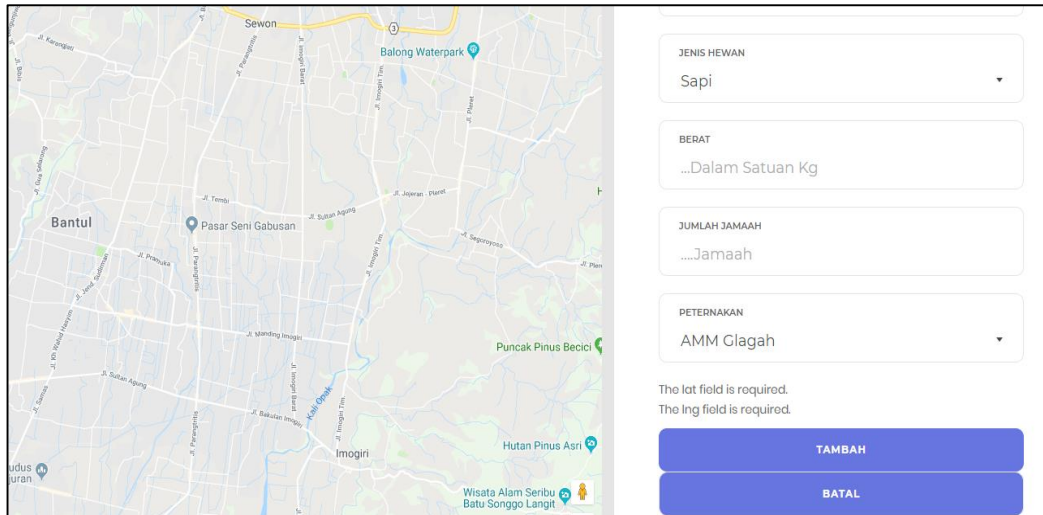
Gambar 5.56 Kolom berat kosong

Pada gambar 5.56 menampilkan halaman tambah data pengiriman. Pada halaman tersebut, jika pengguna mengosongkan kolom berat, maka terdapat notifikasi yang menunjukkan bahwa kolom berat wajib diisi dan tidak boleh kosong.

This screenshot shows the same web application interface as Gambar 5.56. The map on the left is the same. In the form on the right, the 'BERAT' field now contains '..Dalam Satuan Kg' and no longer has an error message. The 'JUMLAH JAMAAH' field is now empty, and a red error message 'Harap Masukkan Jumlah Jamaah !' is displayed next to it. The other fields remain the same as in the previous image.

Gambar 5.57 Kolom jumlah jamaah kosong

Pada gambar 5.57 menampilkan halaman tambah data pengiriman. Jika pengguna mengosongkan kolom jumlah jamaah, maka sistem akan menampilkan notifikasi “harap masukkan jumlah jamaah” karena kolom ini wajib diisi.

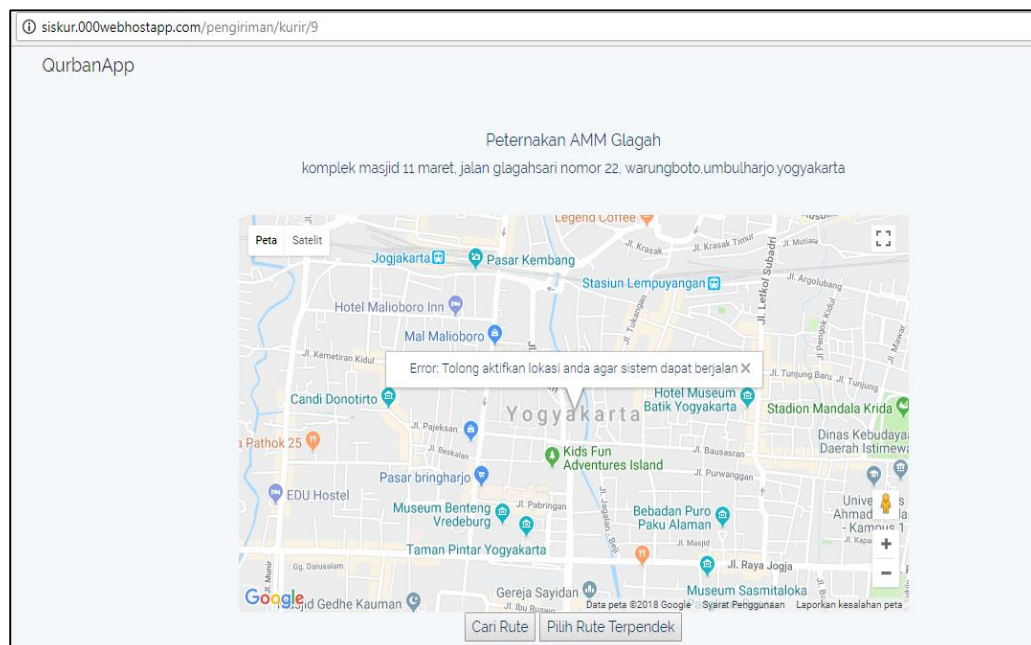


Gambar 5.58 Tidak memberikan titik koordinat

Pada gambar 5.58 menampilkan halaman yang memunculkan notifikasi ketika admin tidak mengisi titik koordinat alamat yang dituju. Titik koordinat harus diisi untuk memudahkan kurir memproses data pengiriman tersebut.

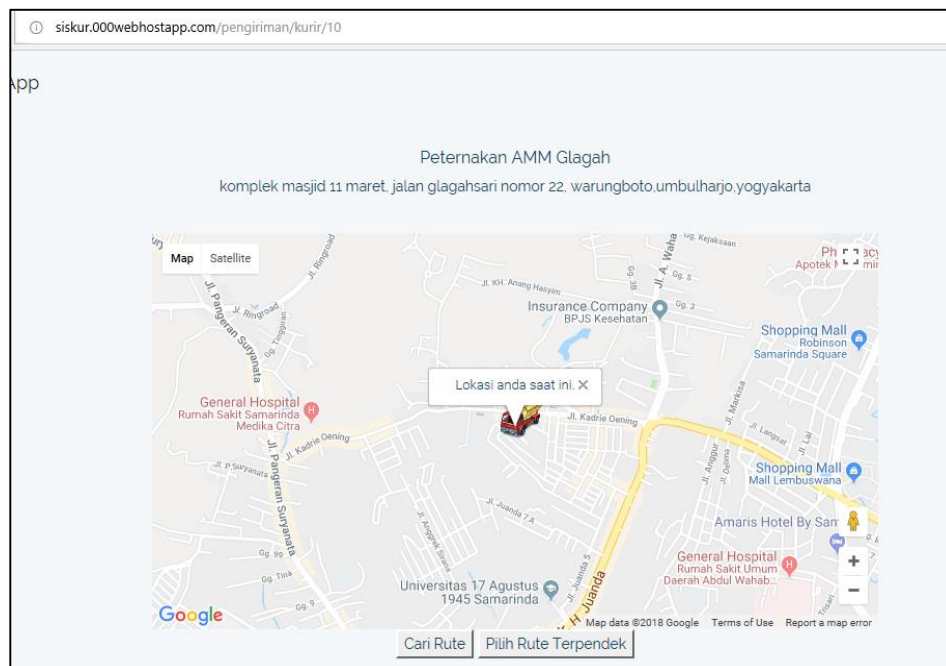
5. Proses Pembacaan Lokasi

Sistem ini menggunakan *Google Maps API* yang hanya dapat diakses secara online. Pada halaman-halaman tertentu terdapat visualisasi peta di sisi kurir maupun admin. Pengguna juga harus mengaktifkan lokasi *device* untuk dapat terbaca oleh *Google maps*.



Gambar 5.59 Lokasi tidak dapat terbaca

Pada gambar 5.59 menampilkan visualisasi peta. Jika pengguna tidak mengaktifkan lokasi perangkat, maka sistem akan menampilkan notifikasi seperti di atas.



Gambar 5.60 Berhasil membaca lokasi

Pada gambar 5.60 menampilkan halaman visualisasi *maps*. Jika pengguna telah mengaktifkan lokasi pada perangkat dan telah terhubung internet (*online*), maka lokasi pengguna akan terbaca dan sistem akan menampilkan halaman seperti gambar 5.42.

6. Proses Kelola Akun Staff



Gambar 5.61 Hapus akun staff

Pada gambar 5.61 menampilkan halaman kelola staff dan halaman ini hanya dapat diakses oleh admin. Ketika admin berhasil menghapus akun staff yang telah terdaftar, maka sistem akan menampilkan notifikasi berhasil hapus akun. Jika akun telah terhapus pada sistem, maka akun tersebut juga telah terhapus pada *database*.

Tabel 5.6 Tabel Pengujian *Black box*

No	Skenario Pengujian	Test Case	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
1.	Melakukan login dengan memasukkan email dan password	Email: admin@lazismu.com Password: 123123	Sistem menolak status login dikarenakan password yang dimasukkan pengguna salah atau tidak valid dan memunculkan notifikasi kesalahan	Sesuai harapan	Valid
2.	Melakukan login dengan memasukkan email dan password	Email: admin@ Password: 123qweasd	Sistem menolak status login dikarenakan alamat email yang dimasukkan tidak valid dan memunculkan notifikasi kesalahan	Sesuai harapan	Valid
3.	Melakukan login dengan memasukkan email dan password	Email: admin@lazismu.com Password:	Sistem menolak status login karena kolom password kosong dan memunculkan notifikasi kesalahan	Sesuai harapan	Valid
4.	Mengakses halaman yang di luar batas hak akses	Kurir mengakses halaman /halaman/rute akses	Sistem menampilkan notifikasi <i>404 not found</i>	Sesuai harapan	Valid
5.	Melakukan reset password dan menekan tombol reset	Email:	Sistem menolak reset password dan memunculkan notifikasi bahwa kolom email wajib diisi	Sesuai harapan	Valid
6.	Melakukan reset password dan menekan tombol reset	Email: admiin@lazismu.Com	Sistem menolak reset password karena email yang dimasukkan tidak terdaftar dan sistem menampilkan notifikasi alamat email tidak valid	Sesuai harapan	Valid
7.	Melakukan daftar akun dan memasukkan password	Nama: kurir Email: kurir@lazismu.com Password:123	Sistem menampilkan notifikasi bahwa password yang dimasukkan kurang dari 6 karakter	Sesuai harapan	Valid
8.	Melakukan daftar akun dan memasukkan konfirmasi	Password: kurir123 Konfirmasi password: kurir	Sistem menampilkan notifikasi kesalahan konfirmasi password karena password	Sesuai harapan	Valid

	password		konfirmasi yang dimasukkan tidak sesuai dengan password sebelumnya		
9.	Berhasil mendaftarkan akun baru	Redirect halaman beranda	Sistem menampilkan notifikasi bahwa pengguna masih berstatus member dan belum dapat mengakses fitur-fitur lainnya sebelum admin mengesahkan jabatan pengguna.	Sesuai harapan	Valid
10.	Melakukan penambahan data dengan memasukkan data pengiriman secara lengkap dan menekan tombol tambah data	Alamat: Jenis muatan: hewan Jenis hewan: sapi Berat: 5 Jumlah jamaah: 50	Sistem menampilkan notifikasi gagal tambah data karena alamat kosong dan tidak diisi.	Sesuai harapan	Valid
11.	Melakukan penambahan data dengan memasukkan data pengiriman secara lengkap dan menekan tombol tambah data	Alamat: jalan kadrie oening samarinda Jenis muatan: hewan Jenis hewan: sapi Berat: 5 Jumlah jamaah: 50	Sistem menampilkan notifikasi kolom berat wajib diisi	Sesuai harapan	Valid
12.	Melakukan penambahan data dengan memasukkan data pengiriman secara lengkap dan menekan tombol tambah data	Alamat: jalan kadrie oening samarinda Jenis muatan: hewan Jenis hewan: sapi Berat: 5 Jumlah jamaah:	Sistem menampilkan notifikasi jumlah jamaah wajib diisi	Sesuai harapan	Valid
13.	Melakukan penambahan data dengan memasukkan data pengiriman secara lengkap dan menekan tombol tambah data	Alamat: jalan kadrie oening samarinda Jenis muatan: hewan Jenis hewan: sapi Berat: 5 Jumlah jamaah: 50 Lat: Lng:	Sistem menampilkan notifikasi titik koordinat tujuan wajib diisi	Sesuai harapan	Valid
14.	Membuka halaman yang terdapat <i>maps</i>	Lokasi perangkat tidak aktif	Sistem menampilkan notifikasi aktifkan lokasi perangkat untuk dapat	Sesuai harapan	Valid

			menggunakan sistem		
15.	Membuka halaman yang terdapat <i>maps</i>	Lokasi perangkat aktif	Sistem menunjukkan lokasi pengguna saat ini	Sesuai harapan	Valid
16.	Menghapus akun staff pada halaman kelola staff	Klik aksi hapus pada akun yang ingin dihapus	Sistem menampilkan notifikasi berhasil hapus	Sesuai harapan	Valid

Tabel 5.6 merupakan tabel yang berisi hasil pengujian *black box* pada sistem. Pada tabel tersebut, dijelaskan tentang beberapa batasan sistem beserta notifikasinya. Dari *form login* hingga *form* tambah data, masing-masing memiliki batas dan notifikasinya. Salah satu contohnya ialah pada *form login*, pengguna tidak dapat masuk ke dalam sistem bila memasukkan *email* dan *password* yang tidak tepat. Adapun batasan lainnya yaitu pada form setelah register dan login. Pengguna tidak dapat mengakses sistem lebih dalam sebelum admin menentukan status pengguna. Status tersebut berupa kurir dan admin yang kemudian pengguna akan dapat mengakses sistem berdasarkan status tersebut. Terdapat banyak batasan dan notifikasi lainnya yang telah dijelaskan secara rinci pada tabel 5.6. Dari tabel 5.6, dapat diketahui bahwa fungsionalitas sistem berjalan dengan baik.

c. User Acceptance Test

Pengujian UAT Admin.

Tabel 5.7 Tabel Pengujian UAT untuk Admin

No	Pernyataan	Penilaian				
		ST S	TS	N	S	SS
1.	Website siskur (sistem kurban) mudah digunakan				✓	
2.	Website siskur (sistem kurban) memiliki tampilan yang menarik				✓	
3.	Penataan dan penyajian informasi memudahkan untuk membaca data dan mencari data					✓
4.	Website siskur memudahkan dalam penambahan dan penyimpanan data					✓
5.	Fitur " <i>search</i> " yang ada pada siskur memudahkan dalam pencarian data					✓

6.	Fitur <i>tracking</i> dapat memudahkan dalam memantau posisi kurir dalam pengiriman hewan kurban					✓
7.	Fitur <i>maps</i> /peta dapat memudahkan untuk melihat informasi rute-rute se-DIY				✓	

Dari penilaian pengguna terhadap sistem pada tabel 5.7, didapatkan hasil pengujian khusus admin. Dari penilaian tersebut telah dibuktikan bahwa penilaian terhadap sistem khusus admin telah optimal dan memudahkan pengguna mengakses sistemnya sebagai admin.

Pengujian UAT Kurir.

Tabel 5.8 Tabel Pengujian UAT untuk Kurir

No	Pernyataan	Penilaian				
		STS	TS	N	S	SS
1.	Website siskur (sistem kurban) mudah digunakan					✓
2.	Website siskur (sistem kurban) memiliki tampilan yang menarik				✓	
3.	Penataan dan penyajian informasi memudahkan untuk membaca data dan mencari data					✓
4.	Fitur " <i>search</i> " yang ada pada siskur memudahkan dalam pencarian data					✓
5.	Fitur "cari rute" memudahkan untuk melihat rute-rute mana saja yang dapat diakses dari/ke tempat pengiriman				✓	
6.	Fitur "cari rute terdekat" dapat membantu pencarian rute terdekat menuju tempat pengiriman hewan					✓
7.	Fitur <i>maps</i> /peta dapat memudahkan untuk melihat informasi rute-rute se-DIY					✓
8.	Perkiraan bahan bakar memudahkan untuk penggunaan bahan bakar kendaraan dalam pengantaran hewan kurban ke tempat tujuan				✓	

Dari penilaian pengguna terhadap sistem pada tabel 5.8, didapatkan hasil pengujian khusus kurir. Dari penilaian tersebut telah dibuktikan bahwa penilaian terhadap sistem khusus kurir telah optimal karena sistem dapat memudahkan pengguna menentukan rute terdekat serta memudahkan pengguna mengakses sistemnya sebagai kurir.

Dengan sistem yang telah diuji, jadi didapatkanlah kelebihan dan kekurangan sistem ini. Adapun kelebihanannya adalah:

- a. Secara otomatis, sistem ini dapat memberikan kemungkinan jalur yang akan dilewati dan dapat memilih jalur yang belum diolah (jalur mentah) secara manual.
- b. Sistem ini dapat melakukan *tracking* secara *real time*.
- c. Secara otomatis, sistem ini juga dapat memberikan saran kendaraan yang dapat digunakan berdasarkan berat muatan yang sesuai.
- d. Sistem dapat memberikan status pengiriman secara *real time*.
- e. Dapat melakukan pengiriman ke tempat yang sama lebih dari sekali dikarenakan bisa saja ada *accepted* baru dari admin dan penyumbang.
- f. Sistem dapat memberikan perkiraan penggunaan bahan bakar yang akan digunakan untuk tiap pengiriman

Selanjutnya, kekurangan pada sistem ini setelah dilakukan pengujian ialah:

- a. Kecepatan pengolahan query dipengaruhi oleh banyaknya titik rute yang diberikan pengguna, jika semakin banyak titik rute yang diberikan, maka akan semakin lambat pengolahannya.
- b. Terdapat *limit callback data* (batasan pengolahan data antara sistem ini dengan API) dari *Google maps* dikarenakan diperlukannya paket premium untuk membuka batasan tersebut. Kemungkinan ini terjadi karena banyaknya penggunaan *API Google* dalam hal komersil.
- c. Tingkat keakurasian posisi dipengaruhi oleh jaringan provider, browser yang digunakan, dan GPS (*Global Positioning System*) perangkat yang digunakan.

BAB VI

PENUTUP

6.1 Kesimpulan

Ada beberapa kesimpulan yang terdapat pada penelitian yang telah dilakukan. Sistem pengiriman hewan kurban ini dapat dikatakan optimal ketika:

1. Sistem dapat menampilkan peta yang digunakan oleh pengguna untuk dapat melakukan pemantauan / *tracking* atas posisi kurir yang melakukan pengiriman hewan kurban.
2. Sistem dapat menunjukkan rute terdekat dari tempat pengambilan hewan ke tempat pendistribusian.

6.2 Saran

Adapun saran untuk penelitian ini supaya penelitian selanjutnya lebih baik lagi:

1. Mengembangkan sistem dengan penggunaan paket premium *Google Maps API* supaya tidak ada batasan pengolahan data antara sistem dengan *API*.
2. Mengembangkan sistem dengan fitur perhitungan rute terdekat dengan *Floyd warshall* dan algoritma lain sehingga pengguna dapat membandingkan dan memilih secara langsung rute yang lebih optimal tiap kali memulai pengiriman.
3. Menggunakan citra mandiri yang dilengkapi dengan *database* alamat dan jarak dari *Google Maps*.

DAFTAR PUSTAKA

- Achmad Arsori. (2013, January 19). *Algoritma Floyd Warshall*, 1.5. (perbandingan) Retrieved October 07, 2017, from ilmu komputer: <https://achmad-arsori.blogspot.co.id/2013/01/algoritma-floyd-warshall.html>
- Aksi Cepat Tanggap. (2015, September 4). *Global Qurban*, 1.5. (ACT) Retrieved January 11, 2018, from Care For Humanity: www.act.id
- Apriadi, H. (2018, may 25). *scribd*, 1.5. (scribd) Retrieved july 8, 2018, from scribd: <https://id.scribd.com/doc/266043481/LINTASAN-TERPENDEK>
- Google Developer. (2015, January 2). *Google Maps Platform*, 1.5. (Google Developer) Retrieved March 14, 2018, from Google Maps Documentation: <https://developers.google.com/maps/documentation/>
- Hasibuan, A. R. (2016). Penerapan Algoritma Floyd Warshall untuk Menentukan Jalur Terpendek dalam Pengiriman Barang. *III*(6).
- Islam Cendekia. (2014, October 04). *Artikel, sejarah islam, teologi islam*, 1.5. (Islam Cendekia) Retrieved October 05, 2017, from Media Islam Populer: <http://www.islamcendekia.com/2014/10/arti-makna-dan-hakikat-idul-adha-qurban.html>
- Kargo. (2018, january 5). *Penghubung jasa truk seluruh Indonesia*. Retrieved from Kargo Indonesia: <https://www.kargo.co.id/kapasitas-truk/>
- Komaruddin. (2014, June 22). *Mengenal Persoalan Optimasi*, 1.0. (KIPMI) Retrieved March 2, 2018, from KIPMI: <https://kipmi.or.id/mengenal-persoalanoptimasi.html>
- McLeod, R. (2001). *Sistem Informasi Manajemen* (Vol. I). Jakarta: Prenhallindo.
- Nugraha, T. (2015). Tutorial Dasar Laravel. *I*(12).
- Siang, J. J. (2011). *Riset Operasi dalam Pendekatan Algoritmis* (Vol. I). Penerbit Andi.
- Wahyudi, I. I. (2016). *Sistem Monitoring Pengiriman Barang menggunakan Algoritma Dijkstra* (Vol. I). Yogyakarta: Ichsan Indra Wahyudi.