ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - KI141502

RANCANG BANGUN APLIKASI ANGKUTAN TRANS SARBAGITA PROVINSI BALI BERBASIS PERANGKAT BERGERAK

I MADE ADITYA PRADNYADIPA MUSTIKA
NRP 5112 100 184

Dosen Pembimbing I
Dr. Tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

Dosen Pembimbing II
Abdul Munif, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR – KI141502

RANCANG BANGUN APLIKASI ANGKUTAN TRANS SARBAGITA PROVINSI BALI BERBASIS PERANGKAT BERGERAK

I MADE ADITYA PRADNYADIPA MUSTIKA
NRP 5112 100 184

Dosen Pembimbing I
Dr. Tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

Dosen Pembimbing II
Abdul Munif, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESES – KI141502

DESIGN AND IMPLEMENTATION OF MOBILE BASED APPLICATION FOR TRANS SARBAGITA IN BALI PROVINCE

I MADE ADITYA PRADNYADIPA MUSTIKA
NRP 5112 100 184

Supervisor I
Dr. Tech. Ir. Raden Venantius Hari Ginardi, M.Sc.

Supervisor II
Abdul Munif, S.Kom., M.Sc.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2017

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI ANGKUTAN TRANS SARBAGITA PROVINSI BALI BERBASIS PERANGKAT BERGERAK

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Manajemen Informasi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :
I MADE ADITYA PRADNYADIPA MUSTIKA
NRP : 5112 100 184

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Dr. Tech. Ir. RADEN VENANTIUS HARI GINARDI, M.Sc. ..... (pembimbing 1)
NIP: 19650518 199203 1 003

ABDUL MUNIF, S.Kom., M.Sc. ..... (pembimbing 2)
NIP: 19860823 201504 1 004

SURABAYA
JANUARI, 2017

[Halaman ini sengaja dikosongkan]

RANCANG BANGUN APLIKASI ANGKUTAN TRANS SARBAGITA PROVINSI BALI BERBASIS PERANGKAT BERGERAK

Nama Mahasiswa	**: I Made Aditya Pradnyadipa Mustika**
NRP	**: 5112100184**
Jurusan	**: Teknik Informatika FTIF-ITS**
Dosen Pembimbing 1	**: Dr. Tech. Ir. Raden Venantius Hari Ginardi, M.Sc.**
Dosen Pembimbing 2	**: Abdul Munif, S.Kom., M.Sc.**

Abstrak

Trans Sarbagita merupakan salah satu upaya pemerintah Bali dalam menyediakan fasilitas angkutan umum untuk mengurangi kemacetan. Trans Sarbagita adalah sebuah fasilitas BRT (Bus Rapid Transit) yang memiliki angkutan pengumpan tersendiri untuk mengangkut penumpang ke halte-halte bus terdekat. Trans Sarbagita sendiri sudah memiliki rencana rute untuk armada bus dan pengumpan mereka. Dari sebagian rute yang sudah beroperasi, informasi seputar rute, dan halte yang sudah beroperasi dirasa belum terpublikasi dengan baik sehingga dapat menjadi masalah bukan hanya untuk masyarakat Bali, tapi juga untuk para wisatawan yang ingin menggunakan fasilitas tersebut. Salah satu cara mengatasi permasalahan tersebut adalah dengan membangun sebuah aplikasi yang berisikan informasi mengenai halte dan rute-rute yang dapat ditempuh untuk mencapai tujuan yang dapat diakses melalui perangkat bergerak.

Aplikasi Trans Sarbagita dapat memberikan rute terpendek yang dapat dilalui trayek pengumpan berdasarkan lokasi awal dan tujuan pengguna yang dikalkulasikan dengan algoritma Dijkstra. Selain menampilkan rute terpendek, aplikasi ini juga dapat menampilkan rute trayek serta lokasi halte Trans Sarbagita yang sudah beroperasi.

Hasil uji coba sistem sudah menunjukkan berjalannya sistem dengan baik sesuai dengan perancangan awal. Meskipun demikian pengembangan lebih lanjut masih dibutuhkan untuk

mengimbangi berkembangnya teknologi, serta berubahnya kebutuhan pengguna yang dapat diaplikasikan pada aplikasi ini.

Kata kunci: Android, Dijkstra, Halte, Perangkat Bergerak, Rute, Transportasi.

DESIGN AND IMPLEMENTATION OF MOBILE BASED APPLICATION FOR TRANS SARBAGITA IN BALI PROVINCE

Student's Name	**: I Made Aditya Pradnyadipa Mustika**
Student's ID	**: 5112100184**
Department	**: Teknik Informatika FTIF-ITS**
First Advisor	**: Dr. Tech. Ir. Raden Venantius Hari Ginardi, M.Sc**
Second Advisor	**: Abdul Munif, S.Kom., M.Sc.**

Abstract

Trans Sarbagita is one of the Balinese government effort in providing public transportation facility to reduce traffic. Trans Sarbagita is a BRT (Bus Rapid Transit) facility that have its own feeder to transport passenger to the closest bus stops. Trans Sarbagita already have their own route plan for the buses and the feeders. From all of the operating route, the information of route and stops considered not well-informed which can be a problem to Balinese people and also the tourist who would like to use this kind of transportation system. one of the ways to solved the problems are by developing an information system that can informed the user about the routes and stops of Trans Sarbagita that can be accesed trough mobile device.

Trans Sarbagita application provide shortest path of Trans Sarbagita feeder route based on origin location and destination of user that calculated using Dijkstra algorithm. Other than visualizing shortest route, this information system also provide user with visualization of feeder's route and bus stop location of active Trans Sarbagita fleet.

Testing results shows how the system works well according to the initial design. However, further development is needed to complement the technology development and the changes of user needs that can be apply to the system.

Keyword: Android, Dijkstra, Mobile, Route, Stop, Transportation.

KATA PENGANTAR

Puji syukur setinggi-tingginya bagi Tuhan yang Maha Esa, yang telah memberikan berkah dan kelancaran sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "Rancang Bangun Aplikasi Angkutan Trans Sarbagita Provinsi Bali Berbasis Perangkat Bergerak" dengan baik.

Pengerjaan Tugas Akhir ini merupakan suatu kesempatan yang sangat berharga bagi penulis, karena dengan mengerjakan Tugas Akhir ini penulis dapat memperdalam, meningkatkan serta mengimplementasikan ilmu yang didapat selama penulis menempuh perkuliahan di jurusan Teknik Informatika ITS.

Terselesaikannya buku Tugas Akhir ini, tidak lepas dari bantuan dan dukungan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan rasa terima kasih kepada:

1. Tuhan yang Maha Esa atas berkah yang tiada habisnya sehingga penulis dapat menyelesaikan Tugas Akhir ini dengan baik.
2. Orang tua serta saudara, Pradnyanita Mustika dan Pradnyatiwi Mustika, yang telah memberikan dukungan moral dan material serta doa yang tak terhingga untuk penulis. Serta selalu memberikan semangat dan motivasi pada penulis dalam mengerjakan Tugas Akhir ini.
3. Bapak Dr. Tech. Ir. Raden Venantius Hari Ginardi, M.Sc selaku pembimbing I yang telah membantu, membimbing, dan memotivasi penulis dalam menyelesaikan Tugas Akhir ini dengan sabar.
4. Bapak Abdul Munif, S.Kom., M.Sc. selaku pembimbing II yang juga telah membantu, membimbing, dan memotivasi penulis dalam mengerjakan Tugas Akhir ini.
5. Bapak Darlis Herumurti, S.Kom., M.Kom. selaku Kepala Jurusan Teknik Informatika ITS, Bapak Radityo Anggoro, S.Kom., M.Sc selaku koordinator TA, dan segenap dosen Teknik Informatika yang telah memberikan ilmunya.

6. Teman-teman terdekat, Surya, Adiartha, Agus, Pur, Wik, Antara, Diksi, Erwin, Dera, Depik , Erta, Sita, Pete, Bobo, Indra, dan Bagus Permata yang senantiasa menghibur dan mendukung penulis dalam mengerjakan tugas akhir ini serta menemani penulis saat penulis membutuhkan semangat.
7. Teman-teman Teknik Informatika ITS angkatan 2012 yang sudah bersama-sama jatuh bangun menjalani kuliah di kampus sejak pertama bertemu hingga akhir kuliah.
8. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Sebagai manusia biasa, penulis menyadari Tugas Akhir ini masih jauh dari kesempurnaan dan memiliki banyak kekurangan. Sehingga dengan segala kerendahan hari penulis mengharapkan saran dan kritik yang membangun dari pembaca.

Surabaya, Januari 2017

I Made Aditya Pradnyadipa Mustika

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dijelaskan mengenai beberapa hal dasar dalam Tugas Akhir ini yang meliputi latar belakang, perumusan masalah, batasan, tujuan dan manfaat pembuatan Tugas Akhir serta metodologi dan sistematika pembuatan buku Tugas Akhir ini. Dari uraian di bawah ini diharapkan gambaran Tugas Akhir secara umum dapat dipahami dengan baik.

1.1 Latar Belakang

Transportasi adalah pemindahan barang dari satu tempat ke tempat lainnya dengan menggunakan sebuah kendaraan yang digerakkan oleh manusia atau mesin. Transportasi digunakan untuk memudahkan manusia dalam melakukan aktivitas sehari-hari. Dewasa ini transportasi menyebabkan pertumbuhan pariwisata yang sangat pesat sekali. Kemajuan fasilitas transportasi mendorong kemajuan kepariwisataan dan sebaliknya ekspansi yang terjadi dalam industri pariwisata dapat menciptakan permintaan akan transportasi yang dapat memenuhi kebutuhan wisatawan. Tidak dapat disangkal lagi bahwa fungsi utama transportasi sangat erat hubungannya dengan "*accessibility*". Maksudnya, frekuensi penggunaannya, kecepatan yang dimilikinya dapat mengakibatkan jarak yang jauh seolah-olah menjadi lebih dekat. Hal ini berarti mempersingkat waktu dan tentunya akan lebih meringankan biaya perjalanan. Dengan demikian transportasi dapat memudahkan orang untuk mengunjungi suatu daerah tertentu, seperti misalnya daerah tujuan wisata. Bila kita adakan sedikit analisa secara umum, hubungan antara pariwisata dan transportasi, maka secara kualitatif kita dapat mengasumsikan bahwa pariwisata tidak dapat berkembang tanpa tersedianya sarana transportasi. Dengan menerapkan kemajuan teknologi diharapkan dunia transportasi akan jauh lebih baik, baik dalam sistem yang hendak dikembangkan, informasi yang dapat

diakses pengguna, serta bagaimana mencari solusi alternatif bila ditemukan hambatan dari pengguna jasa transportasi.

Bali merupakan pulau yang memiliki banyak objek wisata yang terkenal baik di dalam maupun di luar negeri. Hal ini selain memiliki banyak dampak positif terhadap masyarakatnya juga membawa dampak negatif, salah satunya adalah mengenai kemacetan. kemacetan terjadi karena pertumbuhan angkutan pribadi yang tidak terkendali dan minimnya angkutan umum. Di Bali tercatat jumlah sepeda motor mencapai 2,5 juta dan mobil berjumlah 375 ribu. Dilihat dari jenisnya, sepeda motor mendominasi hingga 86 persen, disusul mobil 12 persen dan sisanya jenis kendaraan yang lain [1]. Tingkat kemacetan yang makin parah di Bali, khususnya di kawasan wisata dapat menjadi ancaman bagi kualitas pariwisata daerah ini.

Pemerintah Bali sudah berupaya untuk mengurangi tingkat kemacetan dengan membenahi infrastruktur jalan dan menyediakan fasilitas transportasi umum untuk menunjang pemerataan pertumbuhan ekonomi antar wilayah dan kelancaran lalu lintas. Trans Sarbagita merupakan salah satu upaya pemerintah dalam menyediakan fasilitas angkutan umum. Trans Sarbagita adalah sebuah fasilitas BRT (Bus Rapid Transit) yang memiliki angkutan pengumpan tersendiri untuk mengangkut penumpang ke halte-halte bus terdekat. Nama Sarbagita sendiri merupakan akronim dari Denpasar, Badung, Gianyar dan Tabanan yang juga menunjukkan bahwa armada ini beroperasi di daerah Denpasar, Badung, Gianyar, dan Tabanan untuk armada busnya, dan untuk armada pengumpannya hanya beroperasi di kota Denpasar dan Badung.

Trans Sarbagita sendiri sudah memiliki rencana rute untuk armada bus dan pengumpan mereka, namun hingga tahun ini rute yang aktif sebanyak 2 dari 17 koridor yang direncanakan untuk armada bus dan 4 dari 10 koridor untuk angkutan pengumpan. Dari sebagian rute yang sudah beroperasi, informasi seputar rute dan halte yang sudah beroperasi dirasa belum terpublikasi dengan baik. Sehingga dapat menjadi masalah bukan hanya untuk masyarakat

Bali, tapi juga untuk para wisatawan yang ingin menggunakan fasilitas tersebut. Salah satu cara mengatasi permasalahan tersebut adalah dengan membangun sebuah aplikasi yang berisikan informasi mengenai halte dan rute-rute yang dapat ditempuh untuk mencapai tujuan yang dapat diakses melalui perangkat bergerak.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana membuat desain peta digital untuk rute angkutan Trans Sarbagita untuk divisualisasikan pada perangkat Android?
2. Bagaimana menampilkan rute angkutan Trans Sarbagita yang akan dilalui sesuai dengan lokasi keberangkatan dan tujuan yang ditentukan pengguna?
3. Bagaimana membuat aplikasi yang dapat mengkalkulasikan rute terpendek yang dapat dilalui pengguna dengan menerapkan algoritma Dijkstra?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Aplikasi *client* yang akan dibangun berbasis Android dan kompatibel dengan versi 4.2 (Jelly Bean) ke atas.
2. Aplikasi *server* yang akan dibangun berbasis web.
3. Data yang digunakan ialah data dari Dinas Perhubungan dan Komunikasi Informasi Provinsi Bali.
4. Algoritma yang digunakan dalam mencari jarak terdekat adalah algoritma Dijkstra.

1.4 Tujuan

Adapun Tujuan dari pembuatan Tugas Akhir ini adalah sebagai berikut:

1. Membangun sistem yang berisi informasi seputar Angkutan Trans Sarbagita yang sudah beroperasi di Bali dengan visualisasi peta digital.
2. Menyajikan informasi daring tentang rute angkutan Trayek Pengumpan Trans Sarbagita dan dapat diakses dimanapun dan kapanpun.
3. Membangun aplikasi yang dapat mengakomodir kebutuhan pengguna seperti pencarian rute yang dilalui berdasarkan posisi dan tempat tujuan yang ditentukan pengguna, dan visualisasi rute dan halte yang akan dilalui.

1.5 Manfaat

Beberapa manfaat yang ingin dicapai dalam Tugas Akhir ini antara lain:

1. Memudahkan masyarakat dalam mengetahui ketersediaan angkutan Trans Sarbagita serta jalur yang sudah dapat digunakan.
2. Mempermudah mendapatkan informasi jalur yang akan dilalui berdasarkan lokasi keberangkatan dan kedatangan.
3. Meringankan pekerjaan pihak Dinas Perhubungan dan Komunikasi Informasi Provinsi Bali dalam mempublikasikan rute yang sudah aktif dan dapat dilalui kepada masyarakat.

1.6 Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. **Penyusunan proposal Tugas Akhir**

   Tahap awal untuk memulai pengerjaan Tugas Akhir adalah penyusunan proposal tugas akhir yang berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir

2. **Studi literatur**

   Pada tahap ini akan dilakukan pencarian informasi dan studi literatur yang berhubungan untuk dijadikan referensi dalam melakukan pengerjaan Tugas Akhir. Informasi dan studi literatur ini didapatkan dari buku, *internet*, dan materi-materi kuliah yang berhubungan dengan metode yang akan digunakan. Dimana literatur yang diperlukan dalam pengerjaan tuas akhir ini yaitu mengenai Google Maps API, pemrograman perangkat bergerak, dan algoritma yang akan digunakan.

3. **Analisis dan desain perangkat lunak**

   Tahap ini meliputi perancangan sistem berdasarkan studi literatur dan pembelajaran konsep teknologi dari perangkat lunak yang ada. Tahap ini mendefinisikan alur dari implementasi. Langkah-langkah yang dikerjakan juga didefinisikan pada tahap ini. Pada tahapan ini dibuat prototipe sistem, yang merupakan rancangan dasar dari sistem yang akan dibuat. Serta dilakukan desain fungsi yang akan dibuat yang ditunjukkan melalui diagram alir.

4. Implementasi perangkat lunak

Pada tahap implementasi ini merupakan tahap membangun implementasi rancangan sistem yang telah dibuat. Pada tahapan ini direalisasikan apa yang terdapat pada tahapan sebelumnya, sehingga menjadi sebuah sistem yang sesuai dengan apa yang telah direncanakan. Dimana aplikasi ini akan dibangun menggunakan bahasa pemrograman Java dan php serta dibentuk dengan menggunakan Google Maps API, MySQL sebagai *Relational Database Management Syatem* (RDBMS), dan Android Studio sebagai kakas pemrograman perangkat bergerak.

5. Pengujian dan evaluasi

Pengujian yang dilakukan oleh pengguna terhadap sistem yang baru atau sistem yang telah diubah dengan tujuan memperoleh persetujuan terhadap sistem yang sedang di uji coba dan siap dipakai. Tahapan ini dimaksudkan untuk mengevaluasi kesesuaian data dan program serta mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan pada program.

6. Penyusunan buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat.

1.7 Sistematika Penulisan Laporan Tugas Akhir

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir secara keseluruhan. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini:

Bab I Pendahuluan

Bab yang berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan Tugas Akhir. Selain itu permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

Bab II Tinjauan Pustaka

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan Tugas Akhir ini. Dasar teori yang digunakan adalah informasi mengenai Trans Sarbagita, *Shortest Path Problem*, algoritma Dijkstra, Geocoding, Google Maps API, Android, Android Studio, Java, JSON, SQL, *Web Service*, Apache HTTP Server serta PHP.

Bab III Desain dan Perancangan

Bab ini berisi tentang Perancangan perangkat lunak meliputi spesifikasi kebutuhan, perancangan umum sistem, perancangan arsitektur sistem, perancangan antarmuka, perancangan basis data, serta perancangan *web service*.

Bab IV Implementasi

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode program dan cuplikan hasil dari implementasi umum sistem, basis data, *web service*, dan antarmuka pengguna.

Bab V Uji Coba Dan Evaluasi

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat dengan uji coba

fungsionalitas menggunakan metode *blackbox*. Uji coba pengguna juga akan diterapkan untuk dapat mengetahui umpan balik tentang aplikasi dari sudut pandang pengguna.

Bab VI Kesimpulan Dan Saran

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

BAB II
TINJAUAN PUSTAKA

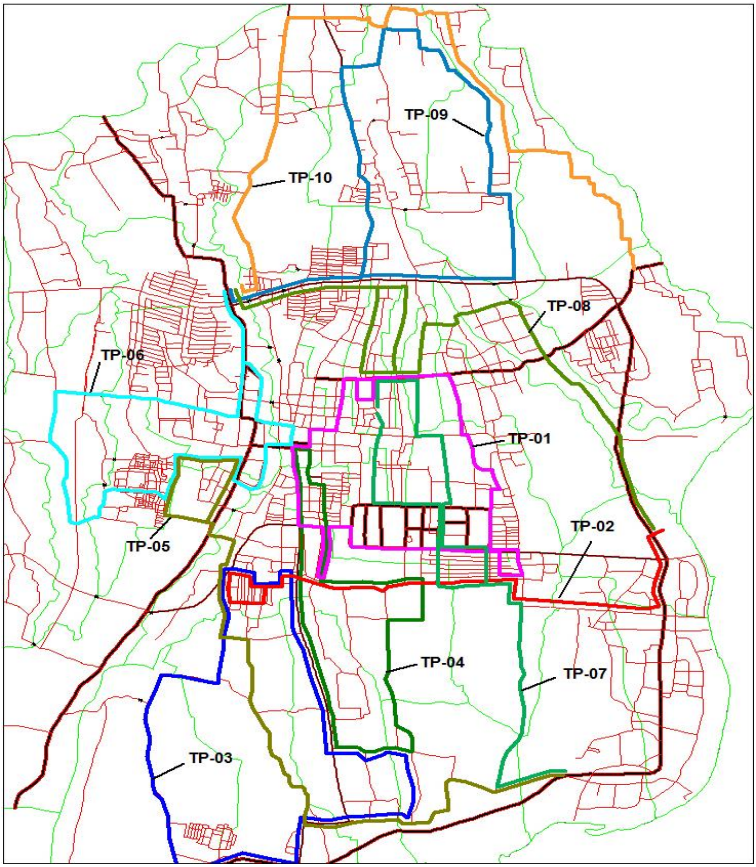
Pada bab ini akan dibahas mengenai teori yang menjadi dasar dari pembuatan Tugas Akhir ini. Teori yang dibahas mencakup elemen-elemen yang terkait dalam topik Tugas Akhir mulai dari sumber dari permasalahan, pendekatan yang digunakan, serta metode dan teknologi yang digunakan untuk pengerjaan Tugas Akhir ini.

2.1 Trans Sarbagita

Trans Sarbagita adalah angkutan umum berjenis *bus rapid transit* (BRT) di Denpasar, Indonesia yang mulai beroperasi pada 18 Agustus 2011. Trans Sarbagita dibuat untuk membangun kembali jaringan angkutan umum di Bali. Pada tahun 2014, Trans Sarbagita mengangkut 5000 penumpang per hari dengan 25 Bus yang beroperasi. Trans Sarbagita beroperasi di 2 koridor yakni koridor 2 Batubulan - Nusa Dua PP dan koridor 1 Kota - Garuda Wisnu Kencana PP. Trans Sarbagita memiliki layanan pengumpan yang terhubung ke koridor utama. Dimana 4 dari 10 rencana trayek pengumpan sudah aktif beroperasi sejak bulan September tahun 2012 [2]. Gambar 2.1 menunjukkan rencana rute 10 trayek pengumpan di Bali.

Trayek pengumpan Trans Sarbagita beroperasi selama 16 jam yang terbagi menjadi dua *shift* yang dimulai *shift* 1 pukul 06.00 sampai pukul 14.00 wita dan *shift* 2 dimulai pada pukul 14.00 sampai pukul 22.00 wita serta rentang waktu keberangkatan setiap 10 menit dengan rute yang dilayani sejumlah 4 trayek yang pergerakan sebagian besar berada diwilayah denpasar timur dan selatan. Kendaraan tiap *shift* berjumlah 24 armada dengan 4

kendaraan cadangan, sehingga total armada yang wajib disiapkan sebanyak 56 armada.



Gambar 2.1 Perencanaan Rute Trayek Pengumpan

Dasar pelaksanaan untuk Program Layanan Angkutan Pengumpan (*Feeder*) Trans Sarbagita Kota Denpasar adalah sebagai berikut:

1. UU no 22 tahun 2009 tentang Lalu lintas dan angkutan jalan *pasal 138* tentang kewajiban pemerintah daerah menyediakan angkutan umm di wilayahnya

2. UU no 22 tahun 2009 tentang Lalu Lintas dan Angkutan Jalan pada *pasal 158 ayat 2* yang menyebutkan bahwa penyediaan angkutan massal (Bus Transarbagita) wajib didukung oleh angkutan pengumpan
3. UU no 22 tahun 2009 tentang Lalu Lintas dan Angkutan Jalan pada *pasal 185* dimana disebutkan angkutan penumpang umum dengan tarif kelas ekonomi dapat diberi subsidi oleh Pemerintah Daerah
4. Peraturan Pemerintah Nomor 70 tahun 2014 *pasal 14 ayat 2* dan *pasal 15* menyebutkan kembali bahwa Pemerintah Daerah bertanggung jawab atas penyelenggaraan dan ketersediaan angkutan umum orang di wilayahnya
5. Peraturan Pemerintah Nomor 70 tahun 2014 pasal 47 menyebutkan bahwa pemerintah daerah wajib menyediakan angkutan massal dengan angkutan pengumpan (feeder) angkutan massal
6. MOU (nota kesepahaman) tahun 2010 antara Pemerintah Pusat, Pemerintah Provinsi Bali, Pemerintah Kota/Kabupaten di kawasan Sarbagita tentang kesanggupan menyediakan sarana dan prasarana pendukung layanan angkutan massal transarbagita.

2.2 *Shortest Path Problem*

Di dalam teori graf, masalah *shortest path* adalah permasalahan dalam mencari jalan antara dua titik di dalam graf dengan total beban dari jumlah jalan yang dilalui adalah yang paling minimal. Masalah *shortest path* dapat didefinisikan untuk graf berarah, tidak berarah, maupun gabungan dari keduanya.

Algoritma pencarian *shortest path* dapat diaplikasikan dalam pencarian arah antar lokasi, seperti rute berkendara dalam situs *web mapping* seperti MapQuest atau Google Map. Selain itu algoritma pencarian *shortest path* dapat juga digunakan untuk menemukan urutan yang optimal untuk mencapai keadaan tujuan tertentu, atau untuk mendapatkan nilai minimum pada waktu yang dibutuhkan untuk mencapai keadaan tertentu. Sebagai contoh, jika

simpul mewakili keadaan dari teka-teki seperti Rubik Cube dan setiap sisi diarahkan sesuai dengan langkah tunggal atau gilirannya, algoritma jalur terpendek dapat digunakan untuk menemukan solusi yang menggunakan jumlah minimum gerakan.

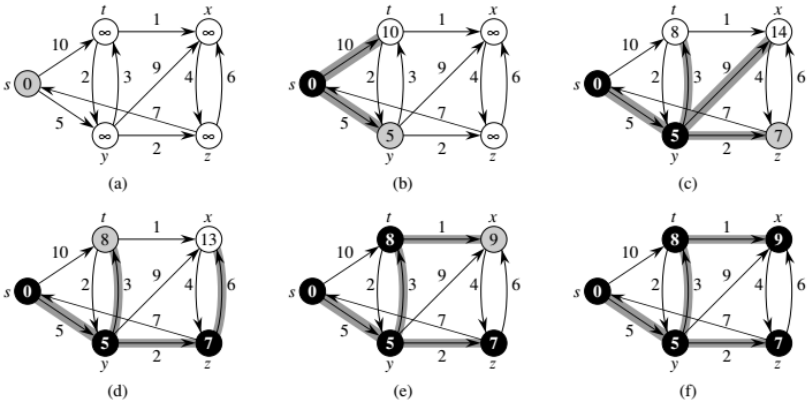
Adapun algoritma yang sering digunakan dalam menyeselesaikan masalah *shortest path* antara lain:

- Algoritma Dijkstra yang mendapatkan beban terkecil dari satu titik awal ke banyak tujuan dengan nilai beban yang positif.
- Algoritma Bellman–Ford yang mendapatkan beban terkecil dari satu titik awal ke banyak tujuan dengan nilai beban yang boleh negatif.
- Algoritma pencarian A*(A-star) menyelesaikan masalah dari satu titik awal ke satu titik tujuan dan menggunakan *heuristic* untuk mempercepat pencarian.
- Algoritma Floyd–Warshall menyelesaikan masalah shortest path untuk setiap pasangan simpul yang ada.
- Algoritma Johnson menyelesaikan masalah shortest path untuk setiap pasangan simpul yang ada dan bisa lebih cepat dari Floyd–Warshall pada kasus *sparse graphs*.

2.3 Algoritma Dijkstra

Algoritma Dijkstra adalah algoritma untuk menemukan jalur terpendek antara node dalam sebuah grafik seperti halnya jaringan jalan. Algoritma ini disusun oleh ilmuwan komputer Edsger W. Dijkstra pada tahun 1956 dan diterbitkan tiga tahun kemudian. Algoritma ini adalah sebuah algoritma *greedy* yang dipakai dalam memecahkan permasalahan jarak terpendek (*shortest path problem)* untuk sebuah graf berarah (*directed graph*) dengan bobot-bobot sisi (*edge weights*) yang bernilai tak-negatif [3]. Berikut merupakan pseudocode dari algoritma Dijkstra.

```
1:    function Dijkstra(Graph, source):
2:       for each vertex v in Graph:
3:          dist[v] := infinity
4:          previous[v] := undefined
5:       dist[source] := 0
6:       Q := the set of all nodes in Graph
7:       while Q is not empty:
8:          u := node in Q with smallest dist[ ]
9:          remove u from Q
10:         for each neighbor v of u:
11:            alt := dist[u] + dist_between(u, v)
12:            if alt < dist[v]
13:               dist[v] := alt
14:               previous[v] := u
15:   return previous[ ]
```



Gambar 2.2 Proses Algoritma Dijkstra

Gambar 2.2 memberikan visualisasi mengenai proses berjalannya algoritma Dijkstra. Titik awal dapat dilihat pada simpul yang diberi penanda s. Simpul berwarna putih memiliki nilai yang belum pasti, karena masih ada kemungkinan nilai simpul yang didapat bisa lebih kecil jika melalui simpul lainnya. Simpul berwarna hitam menandakan bahwa nilai pada simpul merupakan nilai terkecil yang didapat dari iterasi seluruh kemungkinan jalan. Perulangan dilakukan untuk melihat apakah simpul yang

berhubungan dengan simpul yang sedang dikerjakan memiliki nilai yang lebih kecil setelah ditambahkan dengan bobot sisi atau tidak.

Algoritma Dijkstra menerapkan prinsip *greedy* yang mencari solusi optimum pada tiap langkah yang dilalui dengan tujuan mendapatkan solusi terbaik membuat algoritma ini memiliki kompleksitas waktu yang cukup besar yaitu $O(E \log V)$ dimana E adalah jumlah dari sisinya dan V merupakan simpulnya.

Algoritma Dijkstra digunakan sebagai algoritma untuk mencari rute terpendek daripada trayek pengumpan Trans Sarbagita pada kasus tugas akhir ini. Rute trayek yang divisualisasikan pada peta digambarkan sebagai graf berarah dimana simpul merupakan titik temu antar trayek yang terjadi di dalam rute tersebut dan bobot sisi adalah jarak yang memisahkan simpul-simpul tersebut sesuai dengan jalan yang dilalui oleh trayek.

2.4 Geocoding

Geocoding adalah proses konversi alamat (seperti "1600 Amphitheatre Parkway, Mountain View, CA") menjadi koordinat geografis (seperti garis lintang 37,423021 dan garis bujur -122,083739) yang dapat digunakan untuk menempatkan lokasi menurut garis lintang dan garis bujur pada peta. [4]. Geocoding juga bisa mengkonversikan koordinat geografis menjadi sebuah alamat yang dapat dibaca manusia. Layanan Geocoding yang disediakan Google Maps tidak dapat mengkonversikan nama lokasi atau tempat menjadi koordinat geografis.

Geocoding sering digunakan dalam analisis GIS (geographic information system), kartografi, transaksi *mash-up*, atau digunakan dalam proses bisnis yang lebih besar. Bersama dengan GPS, Geocoding juga biasa diaplikasikan untuk keperluan *geotagging* media, seperti foto ataupun *RSS item*.

Pada kasus tugas akhir ini, Geocoding diaplikasikan dengan menggunakan layanan dari Google Maps. Proses Geocoding dijalankan setelah pengguna memasukkan lokasi keberangkatan dan tujuan untuk mengubah masukan pengguna menjadi data

koordinat geografis yang akan digunakan sebagai acuan untuk mencari simpul awal dan tujuan.

2.5 Google Maps API

API (*Application Programming Interface*) atau antarmuka pemrograman aplikasi adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan oleh programmer saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan pengembang perangkat lunak untuk menggunakan fungsi standar dalam berinteraksi dengan sistem operasi. API dapat menjelaskan cara sebuah tugas (task) tertentu dilakukan. Dalam pemrograman prosedural seperti bahasa C, aksi biasanya dilakukan dengan media pemanggilan fungsi. Karena itu, API biasanya menyertakan penjelasan dari fungsi yang disediakannya.

Google Maps API merupakan API yang disediakan oleh Google untuk membuat peta sesuai dengan yang diinginkan [5]. Google Maps API memiliki sejumlah fungsi yang dapat digunakan agar aplikasi dapat berinteraksi dengan Google Maps khususnya dalam fitur menambahkan rute dan halte. Google Maps API memiliki banyak layanan yang dapat digunakan untuk visualisasi peta maupun untuk mendapatkan informasi mengenai lokasi pada peta, adapun beberapa layanan yang disediakan oleh Google Maps adalah sebagai berikut:

- **Google Maps Geocoding API**
  Layanan ini berguna untuk mengkonversi antara alamat menjadi koordinat geografis maupun sebaliknya.
- **Google Maps Distance Matrix API**
  Fungsi dari layanan ini adalah untuk mendapatkan jarak dan waktu tempuh ke beberapa lokasi.
- **Google Maps Road API**
  Menyediakan informasi mengenai jalan seperti batas kecepatan.
- **Google Places API**

Layanan yang memberikan informasi mengenai lokasi-lokasi di sekitar baik itu berupa bisnis, maupun lokasi wisata.

- **Google Maps Time Zone API**
  Layanan ini menyediakan informasi zona waktu dari seluruh lokasi di dunia.
- **Google Maps Geolocation API**
  Layanan yang memberikan lokasi pengguna berdasarkan menara komunikasi dan sinyal *WiFi* yang diterima perangkat pengguna.
- **Google Maps Directions API**
  Layanan yang memberikan informasi berupa arahan antara lokasi yang diinginkan pengguna.
- **Google Maps Elevation API**
  Layanan ini memberikan data elevasi dari suatu lokasi termasuk kedalaman laut.

Layanan-layanan yang disediakan Google sebagian besar bebas untuk digunakan namun dengan kuota pemakaian yang terbatas.

2.6 Android

Android adalah sistem operasi untuk perangkat bergerak yang dikembangkan oleh Google berdasar pada kernel Linux dan didesain untuk perangkat bergerak layar sentuh seperti *smartphones* dan *tablet*. Antarmuka pengguna Android didasarkan pada manipulasi langsung dalam menggunakan gerakan sentuh yang fleksibel sesuai dengan tindakan dunia nyata, seperti *swiping*, *tapping* dan *pinching*, untuk memanipulasi objek di layar, bersama dengan keyboard virtual untuk memasukkan teks. Sistem operasi Android sudah berkembang dari sistem operasi yang hanya ditujukan untuk perangkat mobile menjadi Android TV untuk televisi, Android Auto untuk mobil, dan Android Wear untuk jam tangan.

Android dikembangkan oleh Google secara *private* hingga perubahan terbaru dan update siap untuk dirilis, dan kode sumber

dibuat tersedia untuk umum. Kode sumber ini hanya akan berjalan tanpa modifikasi pada perangkat terpilih yaitu perangkat yang juga diproduksi oleh Google (Seperti Nexus dan Google Pixel). Kode sumber akan diadaptasi oleh *Original Equipment Manufacturers* (OEM) untuk dijalankan pada perangkat keras mereka.

Android memiliki aplikasi yang memperluas fungsionalitas dari perangkat itu sendiri. Aplikasi ini biasanya dibangun menggunakan Android *Software Development Kit* (SDK) dengan menggunakan bahasa pemrograman Java yang memiliki akses lengkap pada API Android [6]. Selain itu bahasa pemrograman Java dapat dikombinasikan dengan bahasa pemrograman C atau C++. Selain Java dan C, bahasa pemrograman Go juga dapat digunakan sejak versi 1.4 namun dengan akses API Android yang terbatas.

Sejarah versi sistem operasi Android dimulai dengan rilis Android Alpha pada 5 November 2007. Versi komersial pertama, Android 1.0, dirilis pada September 2008. Android terus dikembangkan oleh Google dan Open Handset Alliance (OHA). Tabel 2.1 memaparkan versi sistem operasi Android serta API level yang didukung.

Tabel 2.1 Versi Sistem Operasi Android

Nama	Versi	API Level
Alpha	1.0	1
Beta	1.1	2
Cupcake	1.5	3
Donut	1.6	4
Éclair	2.0 – 2.1	5 – 7
Frozen Yogurt	2.2 – 2.2.3	8
Gingerbread	2.3 – 2.3.7	9 – 10
Honeycomb	3.0 – 3.2.6	11 – 13
Ice Cream Sandwich	4.0 – 4.0.4	14 – 15
Jelly Bean	4.1 – 4.3.1	16 – 18
KitKat	4.4 – 4.4.4	19
KitKat for Wearables Only	4.4W	20

Lollipop	5.0 – 5.1.1	21 – 22
Marshmallow	6.0 – 6.0.1	23
Nougat	7.0 – 7.1.1	24 - 25

2.7 Android Studio

Android Studio adalah *Integrated Development Environment* (IDE) ofisial untuk pengembangan aplikasi Android yang dibangun berdasarkan perangkat lunak IntelliJ IDEA [7]. Android Studio dapat dioperasikan pada sistem operasi Linux, Windows, maupun Mac OS X yang dimana Android Studio menggantikan Eclipse *Android Development Tools* (ADT) sebagai IDE utama Google dalam pengembangan *native* Android. Adapun fitur yang disediakan oleh Android Studio antara lain:

- Gradle-*based build system*.
- Emulator Android yang cepat dan kaya fitur.
- Lingkungan yang terintegrasi sehingga dapat digunakan untuk mengembangkan semua perangkat Android.
- Instant Run untuk menyisipkan perubahan ke aplikasi yang sedang berjalan tanpa perlu membangun APK baru.
- Template kode dan integrasi GitHub untuk membantu dalam pembangunan aplikasi dengan fitur dasar dan mengimpor kode sampel.
- Kerangka kerja dan alat pengujian yang komprehensif.
- Lint *tools* untuk mengetahui performa, kegunaan, kompatibilitas versi, and dan masalah lainnya.
- Menunjang C++ dan NDK.
- *Built-in support* untuk Google Cloud Platform, sehingga mudah untuk mengintegrasikan Google Cloud Messaging dan App Engine.

2.8 Java

Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik, dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin. Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana [8]. Karena fungsionalitasnya, aplikasi Java mampu berjalan di beberapa platform sistem operasi yang berbeda. Saat ini Java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak.

2.9 JSON (*JavaScript Object Notation*)

JSON adalah format pertukaran data berbasis teks yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan oleh client maupun server [9]. JSON adalah format bahasa dan dukungan untuk format data JSON tersedia dalam semua bahasa yang populer, beberapa di antaranya adalah C#, PHP, Java, C ++, Python, dan Ruby. JSON biasanya digunakan pada aplikasi website yang meliputi sistem transfer data di dalamnya. Berikut adalah contoh dari format JSON yang diambil dari layanan Google Maps Elevation API.

```
{
   "results":[
      { "elevation": 1608.637939453125,
         "location": {
            "lat": 39.73915360,
            "lng": -104.98470340},
         "resolution": 4.771975994110107}],
   "status":"OK"
}
```

Contoh respon dari layanan elevation API Google Maps memperlihatkan struktur data JSON yang memiliki format data Objek, *Array*, *Value*, *String*, dan *Number*.

Objek JSON adalah sepasang nama dan nilai yang tidak terurut. Objek dimulai dengan '{' dan diakhiri dengan '}'. Setiap nama diikuti dengan ':' dan setiap pasangan nama/nilai dipisahkan oleh ','.

Array JSON adalah kumpulan nilai yang terurut. *Array* dimulai dengan '[' dan diakhiri dengan ']'. Setiap nilai dipisahkan oleh ','.

Nilai (*value*) dapat berupa sebuah string dalam tanda kutip ganda, angka, true, false, null, sebuah objek JSON atau sebuah *array* JSON. Struktur-struktur tersebut dapat disusun bertingkat.

Respon dari layanan elevation API Google Maps memiliki sebuah *array* JSON dengan nama "results" yang berisikan tiga buah objek JSON di dalamnya, yakni "elevation", "location", dan

“resolution” serta sebuah objek JSON dengan nama “status”. Objek JSON “location” yang berada pada *array* JSON “results” juga memiliki dua buah objek JSON dengan nama “lat” dan “lng” yang berisikan nilai *latitude* dan *longitude*. Terlihat nilai latitude dan longitude disusun bertingkat dimana tingkatan di atasnya adalah “location” dan “results” pada tingkatan paling atas.

2.10 MySQL

MySQL adalah *relational database management system* (RDBMS) *open source* [10]. MySQL menggunakan bahasa SQL dalam proses mengakses datanya. Secara umum, SQL terdiri dari dua bahasa, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). Implementasi DDL dan DML berbeda untuk tiap sistem manajemen basis data (SMBD), namun secara umum implementasi tiap bahasa ini memiliki bentuk standar yang ditetapkan ANSI.

Data Definition Language (DDL) berguna untuk mendefinisikan, mengubah, ataupun menghapus basis data dan atau objek yang ada di dalam basis data . Secara umum, DDL yang digunakan adalah CREATE untuk membuat basis data maupun objek-objek basis data baru, USE untuk menggunakan objek basis data, ALTER untuk mengubah objek basis data yang sudah ada, serta DROP untuk menghapus objek basis data.

Data Manipulation Language (DML) digunakan untuk mengubah dan memanipulasi data yang ada dalam suatu tabel. Perintah DML yang umum digunakan adalah SELECT yang berguna untuk menampilkan data, INSERT untuk menambahkan data baru, UPDATE untuk memperbarui data yang sudah ada, dan DELETE untuk menghapus data yang diinginkan.

2.11 *Web Service*

Web service merupakan jasa yang ditawarkan perangkat elektronik untuk berkomunikasi satu sama lain melalui *World Wide Web*. Pada *web service*, teknologi seperti HTTP yang sebenarnya didesain untuk komunikasi antara manusia dan mesin dapat digunakan untuk komunikasi antar mesin, untuk lebih spesifiknya adalah untuk mentransfer file format yang dapat dibaca mesin seperti XML dan JSON [11]. *Web service* biasanya menyediakan antarmuka berbasis web berorientasi objek ke server basis data untuk digunakan oleh *web server* lain atau aplikasi perangkat bergerak. *Web service* dapat diimplementasikan dalam berbagai cara dimana SOAP dan REST adalah beberapa contohnya.

- **SOAP**

  SOAP adalah sebuah spesifikasi protokol untuk pertukaran pesan/informasi terstruktur dalam implementasi *web service* di jaringan komputer. SOAP menggunakan *Extensible Markup Language* (XML) sebagai format pesannya, dan biasanya bergantung pada protokol layer aplikasi lainnya, terutama *Hypertext Transfer Protocol* (HTTP) dan *Simple Mail Transfer Protocol* (SMTP), untuk transmisi dan negosiasi pesan [12].

- **REST**

  Representational State Transfer (REST) adalah sebuah arsitektur software untuk sistem terdistribusi semisal web. REST telah berkembang sebagai model desain *web service* yang dominan saat ini. Istilah *representational state transfer* dikenalkan dan didefinisikan pada tahun 2000 oleh Roy Fielding dalam disertasi doktoralnya. REST biasa disebut dengan "RESTful" [13].

2.12 Apache HTTP Server

Apache HTTP Server atau biasa disebut Apache merupakan perangkat lunak *web server* yang paling banyak digunakan di dunia. Awalnya dibangun berdasar pada server NCSA HTTPd, pengembangan Apache dimulai pada awal tahun 1995 setelah pembangunan pada kode NCSA terhenti. Apache menjadi peran kunci dalam pertumbuhan awal *World Wide Web*, dengan cepat menyalip NCSA HTTPd sebagai server HTTP dominan, dan tetap menjadi yang paling populer sejak April 1996. Pada tahun 2009, menjadi perangkat lunak web server pertama untuk melayani lebih dari 100 juta website [14].

Apache dikembangkan dan dikelola oleh komunitas pengembang terbuka di bawah naungan Apache Software Foundation. Apache sering digunakan pada sistem Unix-like (Linux) [15], dan tersedia untuk berbagai sistem operasi selain Unix, termasuk Microsoft Windows. Dirilis di bawah lisensi Apache, Apache merupakan perangkat lunak yang dapat digunakan secara bebas dan *open source*.

Apache mendukung beragam fitur dan banyak diimplementasikan sebagai *compiled modules* yang memperluas fungsionalitas inti dari apache itu sendiri. Fitur-fitur bervariasi dari bahasa pemrograman *server-side* hingga skema otentikasi.

2.13 PHP

PHP adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP dipakai untuk memrogram situs web dinamis. PHP disebut bahasa pemrograman *server* side karena PHP diproses pada komputer *server* **[16]**. Hal ini berbeda dibandingkan dengan bahasa pemrograman *client-side* seperti JavaScript yang diproses pada *web browser* (*client*).

[Halaman ini sengaja dikosongkan

BAB III
DESAIN DAN PERANCANGAN

Pada bab ini akan dijelaskan mengenai hal-hal berkaitan dengan perancangan sistem yang akan dibangun untuk mempermudah pada tahap implementasi. Perancangan tersebut mencakup deskripsi umum aplikasi, arsitektur sistem, model fungsional, diagram alir aplikasi serta antarmuka aplikasi. Perancangan akan direpresentasikan dengan diagram *Unified Modelling Language* (selanjutnya disebut UML). UML memperlihatkan dan menjelaskan struktur rancangan aplikasi dalam bentuk diagram-diagram.

3.1 Analisis Perangkat Lunak

Pada subbab ini dibahas analisis perancangan dan arsitektur layanan informasi Trans Sarbagita. Analisis sistem meliputi deksripsi dan spesifikasi kebutuhan. Untuk penjelasan lebih detail dapat dilihat pada bagian Bab 3.1.1 dan Bab 3.1.2.

3.1.1 Deskripsi Umum Sistem

Aplikasi yang dibangun pada Tugas Akhir ini adalah Aplikasi Trans Sarbagita yang dapat mengkalkulasikan dan memberikan visualisasi rute terperndek yang dapat ditempuh pengguna berdasarkan titik awal dan tujuan yang ditentukan pengguna dimana pencarian rute terpendek dilakukan dengan algoritma Dijkstra.

3.1.2 Spesifikasi Kebutuhan Perangkat Lunak

Pada layanan informasi yang akan dibangun dibutuhkan spesifikasi perangkat lunak. Spesifikasi perangkat lunak memberikan solusi dari permasalahan yang diberikan sehingga dapat bekerja dengan baik dalam mengakomodasi kebutuhan.

Diharapkan dengan adanya spesifikasi ini dapat menyesuaikan kebutuhan pengguna.

3.1.2.1 Kebutuhan Fungsional

Kebutuhan Fungsional merupakan jenis kebutuhan yang berisikan proses-proses dalam sebuah sistem informasi. Tabel 3.1 menjelaskan kebutuhan fungsional dari aplikasi Trans Sarbagita.

Tabel 3.1 Kebutuhan Fungsional

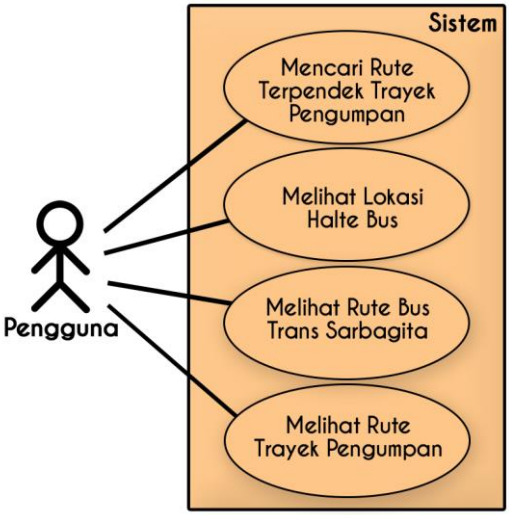
Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
UC-001	Mencari Rute Terpendek Trayek Pengumpan	Pengguna dapat mencari rute terpendek yang dapat dilalui berdasarkan data rute trayek pengumpan dan masukan berupa lokasi awal dan tujuan.
UC-002	Melihat Lokasi Halte Bus	Pengguna dapat melihat informasi lokasi halte bus Trans Sarbagita.
UC-003	Melihat Rute Bus	Pengguna dapat melihat Rute Bus Trans Sarbagita secara keseluruhan.
UC-004	Melihat Rute Trayek Pengumpan	Pengguna dapat melihat Rute Trayek Pengumpan Trans Sarbagita Secara keseluruhan.

3.1.2.2 Aktor

Aktor adalah pihak-pihak, baik manusia maupun sistem atau perangkat lunak lain yang terlibat dan berinteraksi langsung dengan sistem. Pada aplikasi tugas akhir ini hanya memiliki satu aktor yaitu pengguna.

3.1.2.3 Kasus Penggunaan

Pada bagian ini dijelaskan secara rinci kasus penggunaan pada perangkat lunak. Selain itu, spesifikasi kasus penggunaan, diagram aktivitas dan diagram urutan juga dijelaskan di setiap kasus penggunaan. Berdasarkan penjelasan kebutuhan fungsional, maka perangkat lunak memiliki empat kasus penggunaan yang divisualisasikan pada Gambar 3.1.



Gambar 3.1 Diagram Kasus Penggunaan

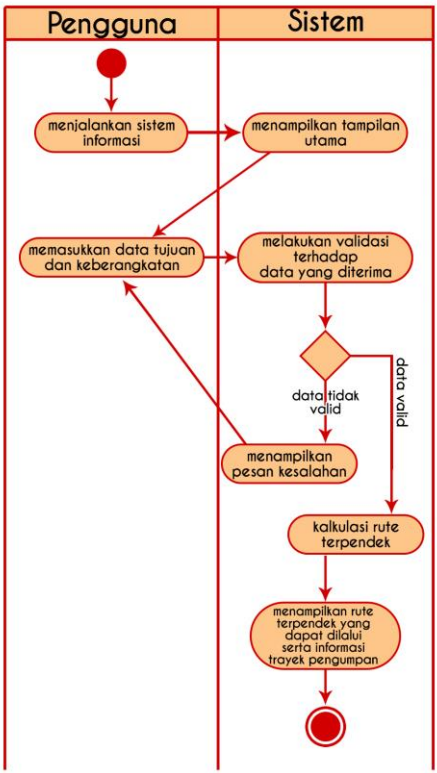
3.1.2.4 Kasus Penggunaan Mencari Rute Terpendek Trayek Pengumpan

Pada kasus penggunaan ini, pengguna dapat mencari rute terpendek berdasarkan data rute trayek pengumpan dan masukan berupa lokasi awal dan tujuan. Layanan ini menampilkan rute dan memberikan informasi angkutan mana yang melewati rute tersebut. Rincian kasus penggunaan dapat dilihat pada Tabel 3.2 dan diagram aktivitas pada Gambar 3.2.

Tabel 3.2 Spesifikasi Kasus Penggunaan Mencari Rute Terpendek Trayek Pengumpan

Komponen	Deskripsi
Nama	Mencari Rute Terpendek Trayek Pengumpan
Kode	UC-001
Deskripsi	Pengguna dapat mencari rute terpendek yang dapat dilalui berdasarkan data rute trayek pengumpan dan masukan berupa lokasi awal dan tujuan.
Tipe	Fungsional
Prasyarat	Aplikasi dijalankan oleh pengguna
Aktor	Pengguna
Aliran:	
- **Kejadian Normal**	1. Pengguna menjalankan aplikasi 2. Sistem menampilkan tampilan utama 3. Pengguna memasukkan data tujuan dan keberangkatan 4. Sistem melakukan validasi terhadap data yang diterima 5. Sistem melakukan kalkulasi rute terpendek 6. Sistem menampilkan rute terpendek yang dapat dilalui oleh pengguna serta informasi trayek pengumpan
- **Kejadian Alternatif**	-
Kondisi Akhir	Rute beserta informasi angkutan ditampilkan
- **Kejadian Pengecualian**	1. Data lokasi tujuan atau keberangkatan berada di luar Bali (setelah langkah 3) a. Sistem menampilkan pesan kesalahan bahwa lokasi berada di luar Bali

	b. Kembali ke langkah 3 2. Masukan salah format (setelah langkah 3) a. Sistem menampilkan pesan kesalahan bahwa data tidak dapat diproses b. Kembali ke langkah 3



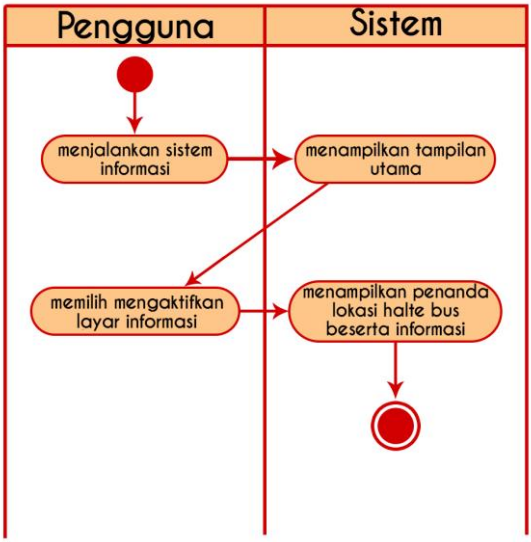
Gambar 3.2 Diagram Aktivitas Kasus Penggunaan Mencari Rute Terpendek Trayek Pengumpan

3.1.2.5 Kasus Penggunaan Melihat Lokasi Halte Bus

Pada kasus penggunaan ini, pengguna dapat melihat seluruh halte Bus Trans Sarbagita. Layanan ini menyediakan dan menampilkan informasi semua halte. Rincian kasus penggunaan dapat dilihat pada Tabel 3.3 dan diagram aktivitas pada Gambar 3.3.

Tabel 3.3 Spesifikasi Kasus Penggunaan Melihat Lokasi Halte Bus

Komponen	Deskripsi
Nama	Melihat Lokasi Halte Bus
Kode	UC-002
Deskripsi	Pengguna dapa melihat informasi lokasi halte bus Trans Sarbagita.
Tipe	Fungsional
Prasyarat	Aplikasi dijalankan oleh pengguna
Aktor	Pengguna
Aliran:	
- **Kejadian Normal**	1. Pengguna menjalankan aplikasi 2. Sistem menampilkan tampilan utama 3. Pengguna memilih mengaktifkan layar informasi 4. Sistem menampilkan penanda lokasi halte bus beserta informasi
- **Kejadian Alternatif**	-
Kondisi Akhir	Sistem menampilkan lokasi halte dan informasinya



Gambar 3.3 Diagram Aktivitas Kasus Penggunaan Melihat Lokasi Halte Bus

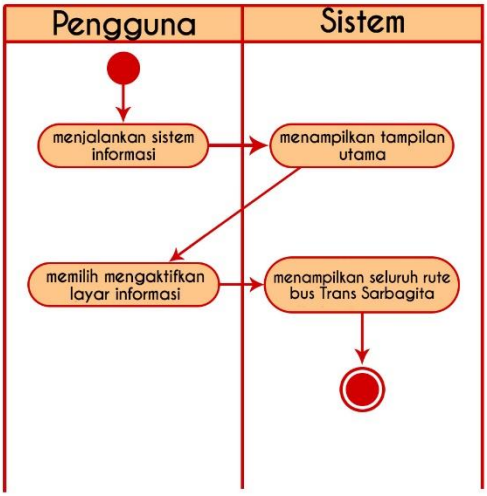
3.1.2.6 Kasus Penggunaan Melihat Rute Bus

Pada kasus penggunaan ini, pengguna dapat melihat seluruh rute dari Bus Trans Sarbagita.. Rincian kasus penggunaan dapat dilihat pada Tabel 3.4 dan diagram aktivitas pada Gambar 3.4.

Tabel 3.4 Spesifikasi Kasus Penggunaan Melihat Rute Buspan s

Komponen	Deskripsi
Nama	Melihat Rute Bus
Kode	UC-003
Deskripsi	Pengguna dapat melihat Rute Bus Trans Sarbagita secara keseluruhan
Tipe	Fungsional
Prasyarat	Aplikasi dijalankan oleh pengguna

Aktor	Pengguna
Aliran:	
- Kejadian Normal	1. Pengguna menjalankan aplikasi 2. Sistem menampilkan tampilan utama 3. Pengguna memilih mengaktifkan layar informasi 4. Sistem menampilkan rute bus Trans Sarbagita secara keseluruhan
- Kejadian Alternatif	-
Kondisi Akhir	Sistem menampilkan rute trayek pengumpan



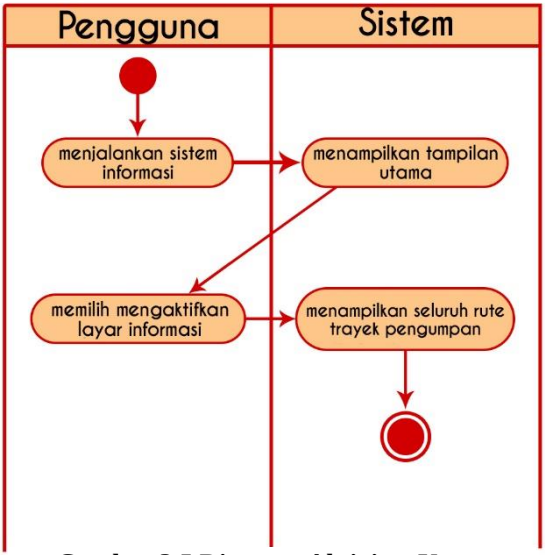
Gambar 3.4 Diagram Aktivitas Kasus Penggunaan Melihat Rute Bus

3.1.2.7 Kasus Penggunaan Melihat Rute Trayek Pengumpan

Pada kasus penggunaan ini, pengguna dapat melihat rute dari Trayek Pengumpan Sarbagita. Rincian kasus penggunaan dapat dilihat pada Tabel 3.5 dan diagram aktivitas pada Gambar 3.5.

Tabel 3.5 Spesifikasi Kasus Penggunaan Melihat Rute Trayek Pengumpan

Komponen	Deskripsi
Nama	Melihat Rute Trayek Pengumpan
Kode	UC-004
Deskripsi	Pengguna dapat melihat Trayek Pengumpan Trans Sarbagita berdasarkan nomor trayek.
Tipe	Fungsional
Prasyarat	Aplikasi dijalankan oleh pengguna
Aktor	Pengguna
Aliran:	
- **Kejadian Normal**	1. Pengguna menjalankan aplikasi 2. Sistem menampilkan tampilan utama 3. Pengguna memilih mengaktifkan layar informasi 4. Sistem menampilkan rute trayek pengumpan secara keseluruhan
- **Kejadian Alternatif**	-
Kondisi Akhir	Sistem menampilkan rute trayek pengumpan

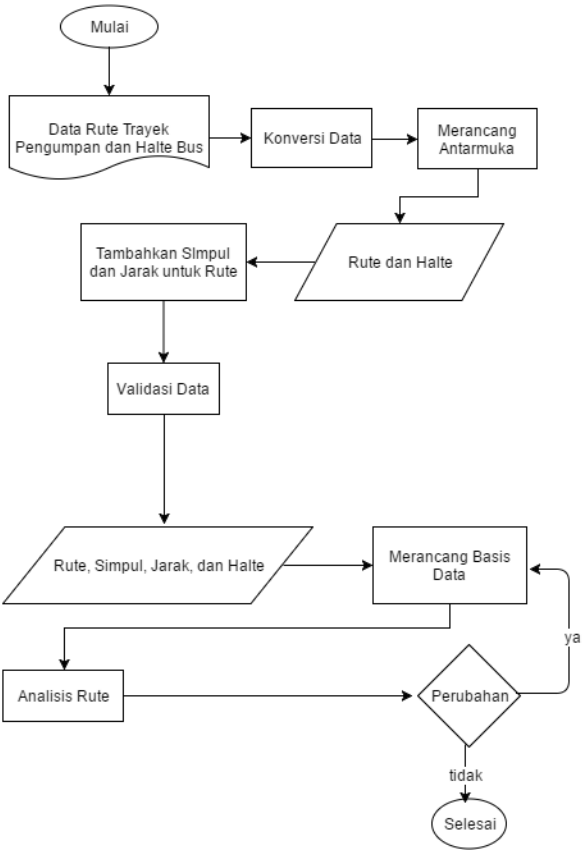


Gambar 3.5 Diagram Aktivitas Kasus Penggunaan Melihat Rute Trayek Pengumpan

3.2 Perancangan Sistem

Subbab perancangan akan membahas garis besar dan detail dari sistem yang dibangun mengenai arsitektur sistem yang digunakan dan perancangan antarmuka pengguna. Garis besar dan detail dari sistem akan dijelaskan menggunakan diagram alir untuk mempermudah dalam memahami alur kerja sistem. Selain itu, proses perancangan sistem serta algoritma juga dijelaskan pada subbab ini.

3.2.1 Perancangan Umum Sistem



Gambar 3.6 Diagram Alir Perancangan Sistem

Secara umum, perancangan sistem yang dibangun terdiri dari beberapa tahapan seperti pada Gambar 3.6. Adapun penjelasan tahapan adalah sebagai berikut:

- Data Rute Trayek Pengumpan dan Halte Bus

  Data ini merupakan data yang didapatkan dari Dinas Perhubungan dan Komunikasi Informasi Provinsi Bali berupa

dokumen berisikan rute trayek dan lokasi halte Trans Sarbagita.

- Konversi Data

  Mengubah data pada dokumen dimana data halte hanya berupa nama halte dan data rute trayek berupa nama-nama jalan yang dilalui trayek. Lokasi halte didapatkan dengan melakukan survei lapangan dan bisa didapatkan *latitude* serta *longitude* melalui Google Maps. Sedangkan rute trayek dikonversikan dengan cara membuat jalur sesuai nama jalan dari data rute pada fitur My Maps di Google Maps yang selanjutnya data jalur di ekspor untuk mendapatkan *polyline* dari sekumpulan *latitude* dan *longitude* jalan.

- Merancang Antarmuka

  Merancang tampilan pada sistem yang berupa tampilan peta, serta visualisasi rute dan halte.

- Tambahkan Simpul dan Jarak untuk Rute

  Mencari titik temu trayek pengumpan yang nantinya akan menjadi simpul pada algoritma yang digunakan dan mencari jarak antar simpul

- Validasi Data

  Validasi dilakukan untuk data halte yang kurang tepat posisinya. Pada validasi dilakukan proses penarikan titik menuju posisi idealnya.

- Merancang Basis Data

  Melakukan proses perancangan untuk data yang terkait dengan sistem.

3.2.2 Perancangan Umum Arsitektur Sistem

Alur informasi diawali dengan pengguna yang mengakses layanan melalui perangkat bergerak. Permintaan dari pengguna dikirim ke *web service* dimana permintaan akan diolah pada *web service* dengan melakukan permintaan kepada Google Maps API. Mendapatkan balasan dari Google Maps API, algoritma untuk mendapat rute dijalankan dan hasil dikirimkan kembali ke perangkat bergerak. Perangkat bergerak mengolah dan memvisualisasikan dengan mengirimkan permintaan ke Google Maps API dalam menjalankan fungsinya.

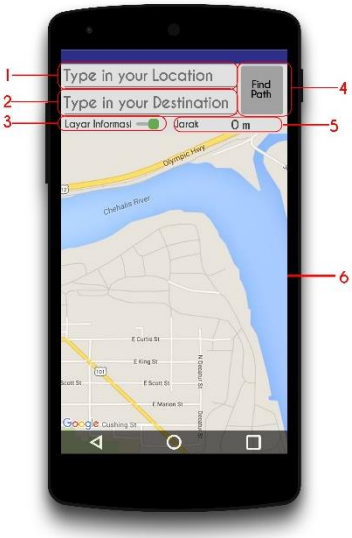
Gambar 3.7 Arsitektur Sistem

Gambar 3.7 menjelaskan alur komunikasi pada arsitektur sistem. Google Maps API dan *web service* bertugas memberikan layanan sesuai dengan permintaan pengguna.

3.2.3 Perancangan Antarmuka

Perancangan antarmuka pengguna merupakan hal yang penting dalam perancangan aplikasi. Antarmuka pengguna yang berhubungan langsung dengan aktor harus memiliki kemudahan bagi penggunanya. Oleh karena itu antarmuka dibuat sederhana agar mempermudah penggunaannya.

Antarmuka ini berisikan kolom masukan lokasi keberangkatan dan lokasi tujuanyang nantinya akan diproses setelah tombol "Find Path" dipilih. Data yang dikirim dan diproses akan diterima dan ditampilkan ke dalam fragmen Google Maps yang berada pada tampilan yang sama. Selain itu antarmuka juga berfungsi untuk memberikan visualisasi rute dari tiap trayek pengumpan dan lokasi halte bus dari Trans Sarbagita. Gambar 3.8 memberikan visualisasi mengenai tampilan antarmuka yang akan dilihat pengguna dan Tabel 3.6 memperjelas fungsi dan peran bagian yang ditandai.



Gambar 3.8 Visualisasi Rancangan Antarmuka

Tabel 3.6 Fungsi Rancangan Antarmuka Pencarian Rute

No	Nama Atribut	Fungsi
1	Data Keberangkatan	Memasukkan data keberangkatan yang diinginkan pengguna.
2	Data Tujuan	Memasukkan data Tujuan yang diinginkan pengguna.
3	Layar Informasi	Memilih untuk menampilkan lapisan layar yang berisikan informasi lokasi halte bus, rute trayek, serta rute bus Trans Sarbagita.
4	Tombol Find Path	Mengirim informasi dan menjalankan fungsi pencarian rute terpendek.
5	Jarak	Menampilkan jarak dari rute trayek pengumpan Trans Sarbagita terpendek sesuai hasil dari proses pencarian rute.
6	Google Maps Fragment	Menampilkan tampilan peta dan hasil fungsi yang dijalankan.

3.3 Perancangan Basis Data

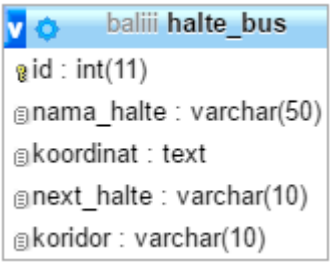
Pada perancangan basis data dibutuhkan analisis data rute trayek pengumpan Trans Sarbagita. Setelah menganalisis dan menambahkan data lainnya, maka penulis dapat membangun basis data yang diperlukan dalam membangun aplikasi ini.

Gambar 3.9 Tabel graph

Gambar 3.9 menjelaskan tabel graph beserta atributnya. Tabel graph merupakan tabel yang berisikan data rute dalam bentuk text dari simpul ke simpul. Tabel ini memiliki atribut “id”, “simpul_awal”, ”simpul_tujuan”, ”jalur”, “bobot”, dan “*temp*”. atribut “*temp*” digunakan untuk menyimpan status dari simpul sementara yang nantinya akan dibuat.

Gambar 3.10 Tabel angkutan_umum

Tabel angkutan_umum yang digambarkan pada gambar 3.10 berisikan beberapa atribut guna mendefinisikan rute yang dilalui oleh trayek pengumpan. Tabel ini berisikan atribut “id”, “no_trayek”, dan “simpul” dimana atribut “simpul” berisikan text yang memuat jalur mana saja yang dilalui oleh trayek berdasarkan simpul awal dan tujuan pada tabel graph.



Gambar 3.11 Tabel halte_bus

Gambar 3.11 merupakan gambaran dari tabel halte_bus dimana tabel ini berisikan data koordinat halte bus yang disimpan dalam bentuk text yang ditempatkan pada atribut “koordinat”. Selain itu tabel ini juga berisi data “nama_halte ” dan “koridor”

yang menunjukkan nama halte dan bus dari koridor mana saja yang melalui halte tersebut. Selain itu tabel ini juga memiliki atribut "next_halte" yang berisi data id halte selanjutnya yang akan dilalui oleh bus.

Gambar 3.12 Tabel rute_bus

Gambar 3.12 merupakan gambaran dari tabel rute_bus yang berisikan data koridor dalam atribut "koridor" yang memiliki rute dari bus Trans Sarbagita yang disimpan dalam kumpulan titik berupa latitude dang longitude pada atribut "jalur".

3.4 Perancangan *Web Service*

Web Service aplikasi Trans Sarbagita akan dibangun menggunakan PHP dan memiliki tiga layanan utama yang akan membantu dalam mengirimkan data dari basis data. Adapun tiga layanan tersebut antara lain layanan pencarian rute terpendek, layanan rite trayek pengumpan, dan layanan lokasi halte bus.

3.4.1 Layanan Pencarian Rute Terpendek

Layanan ini memberikan rute terpendek berdasarkan masukan pengguna berupa lokasi keberangkatan dan lokasi tujuan. Layanan ini diakses melalui antarmuka HTML dengan *request* berupa *string* URL serta *text* yang berisikan infromasi lokasi keberangkatan dan tujuan. Setelah data diterima, kalkulasi jarak terpendek akan dijalankan sesuai dengan perancangan alur sistem pencarian rute yang dijelaskan pada subbab 3.4.1. Contoh *request* yang akan dikirimkan adalah seperti berikut.

```
url/Main.php?koord_user=”lokasi  awal”&koord_destination=”lokasi tujuan”
```

Respon dari *web service* akan memiliki format sebagai berikut.

```
{
  "jalur_shortest_path": [
    {"lat": 'x1',"lng": 'y1'},
    {"lat": 'x2',"lng": 'y2'},
    {"lat": 'x3',"lng": 'y3'}],
  "jarak": 'jarak tempuh',
  "angkot": [
    {"koordinat_angkot": {
        "lat": 'x4',
        "lng": 'y4'},
      "no_angkot": 'nama rute trayek'},
    {"koordinat_angkot": {
        "lat": 'x5',
        "lng": 'y5'},
      "no_angkot": 'nama rute trayek'}],
   "dataawal":
    {"origin":[x6,y6],
    "destination":[x7,y7]}
}
```

Respon di atas memiliki dua *array* JSON dengan nama jalur_shortest_path, dan angkot serta dua buah objek JSON yang bernama jarak dan dataawal. *Array* JSON jalur_shortest_path berisi data kumpulan titik *latitude* yang dilambangkan dengan ‘x1’, ‘x2’, dan ‘x3’ serta *longitude* yang dilambangkan dengan ‘y1’, ‘y2’, dan ‘y3’ yang nantinya akan membentuk sebuah rute terpendek yang dapat dilalui pengguna. Data di dalam jalur_shortest_path dapat menghasilkan lebih dari tiga hasil, tidak seperti dalam contoh respon di atas. Objek JSON jarak berisikan total jarak dalam meter yang dilalui rute didapatkan. Sedangkan *Array* JSON angkot memiliki sekumpulan data untuk mengetahui titik-titik mana saja pengguna perlu berganti angkutan. *Array* tersebut berisikan objek JSON koordinat_angkot yang memiliki titik *latitude* yang dilambangkan dengan ‘x5’ dan *longitude* yang dilambangkan dengan ‘y5’ tempat dimana pergantian angkutan perlu dilakukan. Selain itu *array* JSON angkot berisikan objek JSON no_angkot yang memberikan informasi angkutan dengan nomor pengenal tersebut

dapat digunakan oleh pengguna. Objek dataawal berisikan data titik *latitude* yang dilambangkan dengan ‘x6’ pada *array* JSON origin dan ‘x7’ pada *array* JSON destination serta nilai *longitude* yang dilambangkan dengan ‘y6’ dan ‘y7’ pada array JSON oirigin dan destination.

3.4.2 Layanan Rute Trayek Pengumpan

Layanan Rute Trayek Pengumpan memberikan pengguna rute dari salah satu trayek pengumpan sesuai dengan *request* yang dikirimkan oleh pengguna. Layanan ini diakses melalui antarmuka HTML dengan request berupa *string* URL serta *text* yang berisikan infromasi id trayek pengumpan yang nantinya merupakan sebuah pilihan pada antarmuka perangkat. Contoh *request* yang akan dikirimkan adalah seperti berikut.

```
url/ShowRoute.php?no_trayek=”id trayek”
```

Respon dari *web service* akan memiliki format sebagai berikut.

```
{
  "jalur": [
    {"lat": ‘x1’,"lng": ‘y1’},
    {"lat": ‘x2’,"lng": ‘y2’},
    {"lat": ‘x3’,"lng": ‘y3’},
    {"lat": ‘x4’,"lng": ‘y4’},
    {"lat": ‘x5’,"lng": ‘y5’},
    {"lat": ‘x6’,"lng": ‘y6’}]
  “koordinat”: [
    {"lat": ‘x7’,"lng": ‘y7’},
    {"lat": ‘x8’,"lng": ‘y8’}]
  “no_trayek”: “id trayek”
}
```

Respon *web service* yang berupa objek JSON memperlihatkan rute trayek pengumpan yang berisikan *array* JSON jalur, dimana jalur berisikan sekumpulan titik *latitude* yang dilambangkan dengan ‘x’ dan *longitude* yang dilambangkan dengan ‘y’ yang merepresentasikan rute dari trayek pengumpan itu sendiri. Jumlah data jalur yang diterima tergantung dari titik yang akan membentuk rute trayek. Selain itu terdapat pula *array* JSON

koordinat yang berisikan titik tengah antar simpul yang nantinya akan berfungsi sebagai lokasi penanda trayek mana yang melewati jalur tersebut. Lokasi penanda akan dibentuk pada setiap titik ‘x’ yang berupa *latitude* dan ‘y’ sebagai *longitude*. Sedangkan objek JSON no trayek hanya berisikan ‘id trayek’ dari rute yang diminta oleh pengguna.

3.4.3 Layanan Lokasi Halte Bus

Layanan ini memberikan lokasi halte-halte bus Trans Sarbagita berupa *latitude* dan *longitude* yang nantinya akan ditampilkan sebagai *marker* pada antarmuka pengguna dan berisikan informasi tambahan berupa nama halte, halte selanjutnya yang akan dilalui, serta koridor mana yang melewati halte tersebut. Layanan ini diakses melalui antarmuka HTML dengan *request* berupa *string* URL. Contoh *request* yang akan dikirimkan adalah seperti berikut.

```
url/Showhalte.php
```

Respon dari *web service* akan memiliki format sebagai berikut.

```
[
  {
    "koordinat": "x1,y1",
    "nama": "nama_halte1",
    "haltenext": "halte_next1",
    "halteprev": "halte_prev1",
    "koridor": "koridor_halte1"
  },
  {
    "koordinat": "x2,y2",
    "nama": "nama_halte2",
    "haltenext": "halte_next2",
    "halteprev": "halte_prev2",
    "koridor": "koridor_halte2"
  },
{
    "koordinat": "x3,y3",
```

```
    "nama": "nama_halte3",
    "haltenext": "halte_next3",
    "halteprev": "halte_prev3",
    "koridor": "koridor_halte3"
  }
]
```

Layanan lokasi halte bus memberikan respon yang berisi data berupa *array* JSON dimana *array* tersebut berisikan koordinat halte dengan nama objek JSON koordinat yang berisikan titik latitude dengan lambang 'x' dan longitude dengan lambang 'y', nama halte dengan nama objek JSON nama, halte selanjutnya yang akan dituju dengan nama objek JSON haltenext, halte sebelumnya dengan nama objek JSON halteprev, serta koridor halte tersebut dengan nama objek JSON koridor. Jumlah data pada *array* JSON yang diterima sesuai dengan jumlah halte yang sudah terdata.

3.4.4 Layanan Rute Bus Trans Sarbagita

Layanan rute bus Trans Sarbagita merupakan layanan pelengkap dari layanan lokasi halte bus dimana layanan ini memberikan sekumpulan titik latitude dan longitude yang akan divisualisasikan menjadi rute penghubung antar halte-halte yang ada. Layanan ini diakses melalui antarmuka HTML dengan *request* berupa *string* URL. Contoh *request* yang akan dikirimkan adalah seperti berikut.

```
url/Showallrute.php
```

Respon dari *web service* akan memiliki format sebagai berikut.

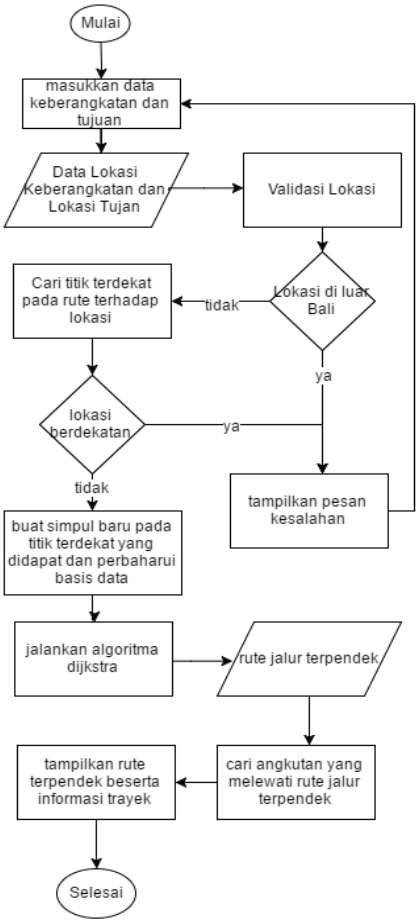
```
[
  {
    "0": [
      {"lat": 'x1',"lng": 'y1'},
      {"lat": 'x2',"lng": 'y2'}]
```

```
  }
  {
    "1": [
      {"lat": 'x3',"lng": 'y3'},
      {"lat": 'x4',"lng": 'y4'},
      {"lat": 'x5',"lng": 'y5'}]
  }
]
```

Layanan ini memberikan respon seluruh rute yang ada pada basis data. Dimana tiap satu data akan ditampilkan sesuai id pada basis ditampilkan sebagai array JSON 0 dan 1 pada contoh respon diatas. Dalam *array* JSON tersebut terdapat kumpulan titik yang akan berjumlah sesuai jumlah data. Dimana titik titik yang dilambangkan dengan 'x' sebagai titik *latitude* dan 'y' sebagai titik *longitude* nantinya akan memberikan visualisasi rute bus Trans Sarbagita.

3.5 Perancangan Alur Sistem Pencarian Rute

Pada Gambar 3.13 ditampilkan alur sistem pencarian rute terpendek yang digunakan pada layanan aplikasi Trans Sarbagita yang dapat mewadahi pencarian rute terpendek berdasarkan jarak rute yang dilalui trayek pengumpan.



Gambar 3.13 Diagram Alir Sistem Pencarian Rute

Adapun penjelasan proses di dalam alur tersebut adalah sebagai berikut:

a. Masukkan Data Keberangkatan dan Tujuan

Proses ini bertujuan untuk mendapatkan data keberangkatan dan tujuan yang akan diproses pada sistem.

b. Validasi Lokasi
   Proses ini bertujuan untuk mengecek apakah lokasi yang dimasukkan berada di area Bali atau tidak. Hal ini diperlukan guna meminimalisir operasi proses yang tidak diperlukan.
c. Cari Titik Terdekat Pada Rute Terhadap Lokasi
   Mencari titik terdekat pada rute terhadap lokasi bertujuan untuk menginformasikan pada pengguna dimana jalur terdekat yang dilewati oleh trayek pengumpan. Titik terdekat diperlukan untuk digunakan dalam pembuatan simpul baru
d. Buat Simpul Baru Pada Titik Terdekat Yang Didapat dan Perbaharui Basis Data
   Membuat simpul baru adalah proses menambahkan simpul untuk digunakan sebagai titik awal dan akhir algoritma beroperasi.
e. Jalankan Algoritma Dijkstra
   Algoritma dijalankan untuk mendapatkan hasil rute terpendek.
f. Cari Angkutan yang Melewati Rute Jalur Terpendek
   Proses ini bertujuan untuk mengetahui angkutan trayek pengumpan mana saja yang melewati jalur terpendek yang sudah didapatkan sehingga nantinya informasi angkutan dapat dilihat oleh pengguna.
g. Tampilkan Rute Terpendek Beserta Informasi Trayek
   Menampilkan rute terpendek dilakukan agar pengguna mendapatkan visualisasi terhadap rute yang dapat dilalui oleh pengguna. Serta menampilkan *marker* yang berisikan

informasi tambahan berupa trayek yang beroperasi pada jalur tersebut.

h. Tampilkan Pesan Kesalahan

Pesan kesalahan akan tampil jika terjadi kesalahan dari masukan yang menyebabkan memiliki keluaran yang memberi tahu pengguna kesalahan apa yang terjadi.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Bab ini membahas mengenai implementasi sistem sesuai dengan analasisis dan perancangan sistem secara umum yang mengacu pada desain dan perancangan yang telah dibahas pada bab sebelumnya.

Implementasi yang dijelaskan meliputi lingkungan pembangunan perangkat lunak dan perangkat keras yang digunakan dalam pengembangan serta implementasi sistem. Implementasi kode sumber dan implementasi antarmuka perangkat lunak juga akan dibahas dimana implementasi menggunakan bahasa pemrograman PHP sebagai *web service*, MySQL sebagai basis data dan Java sebagai pembangunan sistem pada perangkat bergerak.

4.1 Lingkungan Implementasi

Lingkungan implementasi dan pengembangan dibagi menjadi dua bagian, meliputi perangkat lunak dan perangkat keras.

4.1.1 Perangkat Lunak

Lingkungan implementasi dan pengembangan dilakukan menggunakan perangkat lunak sebagai berikut:

- Sistem Operasi Windows 10 64 bit sebagai lingkungan pengembangan sistem secara keseluruhan.
- Sistem Operasi Linux 64 bit sebagai komputer server *web service*.
- Android Studio 2.2.3 sebagai IDE utama pembangunan dan pengembangan sistem.
- Android Platform versi API 23: Android 6.0 (Marshmallow) *revision* 3 sebagai sistem perangkat uji coba dan implementasi sistem.
- PHP versi 5.6.27 dalam implementasi *web service*.

- Apache versi 2.4.23 sebagai server *web service*.
- MySQL versi 5.5.52 untuk basis data pada *web service*.

4.1.2 Perangkat Keras

Lingkungan perangkat keras yang digunakan selama proses pengerjaan Tugas Akhir adalah sebgai berikut:

- Laptop dengan *processor* Intel(R) Core(TM) i7-3517U CPU @ 1.90GHz, *Installed Memory* (RAM) 8.00 GB sebagai lingkungan pengembangan sistem secara keseluruhan.
- *Smartphone* dengan *processor* Qualcomm MSM8994 Snapdragon 810 @ 4x1.56 GHz Cortex-A53 & 4x1.82 GHz Cortex-A57, *Installed Memory* (RAM) 4.00 GB sebagai perangkat uji coba dan implementasi sistem.

4.2 Implementasi Umum Sistem

Pada tahap implementasi ini membutuhkan beberapa proses sehingga peta dan halte dapat divisualisasi. Hasil dari proses-proses pada tahap ini dibutuhkan kembali pada proses selanjutnya. Selain itu, implementasi umum sistem merupakan proses dasar yang dibutuhkan sebelum melakukan proses implementasi lainnya.

4.2.1 Pendataan Halte Bus dan Rute Trayek Trans Sarbagita

Aplikasi ini memiliki data halte bus serta rute trayek pengumpan Trans Sarbagita yang akan disimpan. Data yang diberikan oleh Dinas Perhubungan Informasi dan Komunikasi Provinsi Bali disajikan dalam bentuk titik halte bus dan jalan yang dilalui trayek pengumpan. Dari data yang diterima, diperlukan pendataan lebih lanjut agar bisa digunakan dalam perhitungan dan visualisasi pada perangkat.

4.2.1.1 Pendataan Halte Bus

Data Halte Bus didapatkan melalui survei lapangan, dimana *latitude* dan *longitude* dari halte bus disimpan untuk digunakan dan dimasukkan dalam basis data. Hasil pendataan dapat dilihat pada cuplikan data halte pada Tabel 4.1 dan Tabel 4.2

Tabel 4.1 Cuplikan data halte pada koridor 1

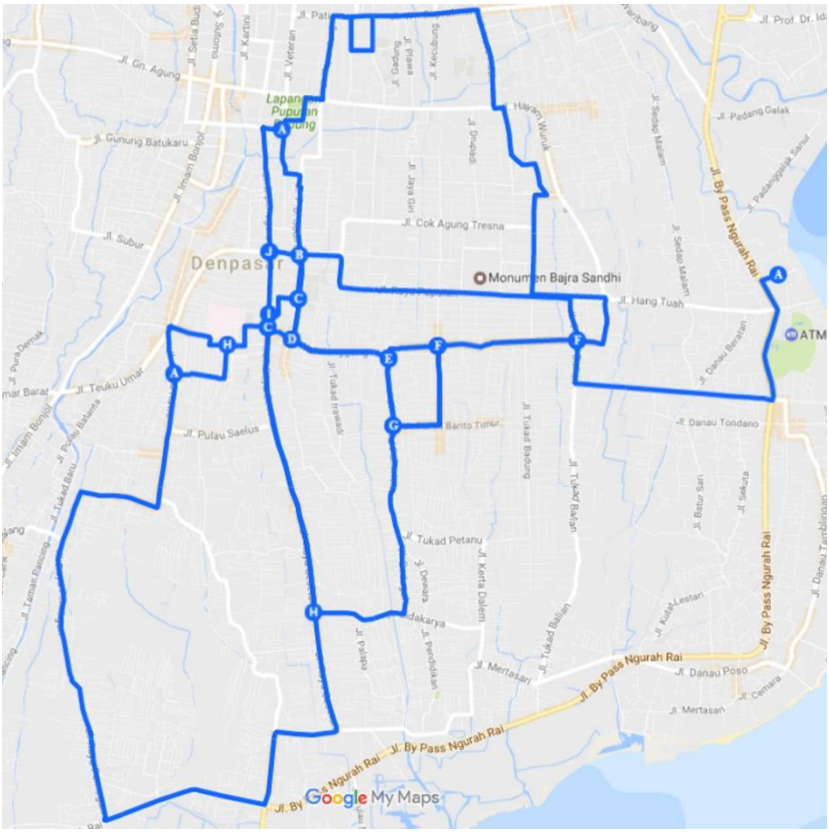
ID	Nama Halte	Koordinat	Halte Selanjutnya
1	Kamboja	-8.6511808,115.2244447	2
2	Surapati	-8.656535, 115.220718	3
3	Sudirman 2	-8.6716668,115.2182199	4
4	Sudirman 3	-8.6755317,115.2177339	5
5	Pesanggaran 1	-8.7168465,115.2133202	6
6	Pedungan 1 a	-8.7170407,115.205237	7
7	Pedungan 2 a	-8.7187963,115.2005534	8

Tabel 4.2 Cuplikan data halte pada koridor 2

ID	Nama Halte	Koordinat	Halte Selanjutnya
42	Batubulan	-8.631351, 115.260876	43
43	Siulan a	-8.635250, 115.255552	44
44	Tohpati 1	-8.6395908,115.2540515	45
45	Prof. I.B. Mantra 1	-8.6498148,115.2548555	46
46	Waribang a	-8.6588496,115.2536035	47
47	Matahari Terbit 1	-8.6717698,115.258211	48
48	Sindhu 1	-8.6811509,115.2593604	49

4.2.1.2 Pendataan Rute Trayek Pengumpan

Data rute trayek pengumpan didapatkan dengan mengekspor *directions* yang dibuat pada fitur My Maps pada Google Maps. Hasil ekspor berupa kumpulan data *latitude* dan *longitude* dari beberapa titik pada jalur trayek pengumpan. Gambar 4.1 menunjukkan *directions* yang dibuat pada fitur My Maps dimana titik-titik yang ada pada gambar nantinya akan menjadi simpul.



Gambar 4.1 Hasil *direction* yang dibuat pada fitur My Maps

4.3 Implementasi Basis Data

Implementasi pembuatan basis data ini menggunakan bahasa SQL. Pada bagian ini, terdapat subbagian yang menjelaskan implementasi setiap tabel beserta hasilnya. Implementasi basis data ini berguna untuk menyimpan dan mengelola data yang dibutuhkan dalam pembentukan layanan informasi ini.

4.3.1 Implementasi Tabel graph

Implementasi tabel graph berguna untuk menyimpan data rute trayek pengumpan serta simpul awal dan tujuan daripada rute tersebut. Pada tabel graph memuat atribut "id", "simpul_awal", "simpul_tujuan", "jalur", "bobot", dan "temp", dimana atribut jalur berisikan sekumpulan data *latitutde* dan *longitude* dari jalur yang dilalui trayek, dan atribut temp merupakan status sementara ketika nantinya dibutuhkan jalur tambahan untuk membuat simpul baru. Implementasi tabel dan hasil dapat dilihat pada Kode Sumber 4.1 dan Gambar 4.2.

```
CREATE TABLE `graph` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `simpul awal` varchar(10) NOT NULL,
  `simpul tujuan` varchar(10) NOT NULL,
  `jalur` text NOT NULL,
  `bobot` double NOT NULL,
  `temp` char(2) NOT NULL DEFAULT 'N',
  PRIMARY KEY (`id`));
```

Kode Sumber 4.1 Membuat Tabel graph

id	simpul_awal	simpul_tujuan	jalur	bobot	temp
1	1	2	{"nodes": ["1-2"], "coordinates": [[-8.67152, 115....	3678	N
2	2	3	{"nodes": ["2-3"], "coordinates": [[-8.67715, 115....	1360	N
3	3	4	{"nodes": ["3-4"], "coordinates": [[-8.67761, 115....	1203	N
4	4	5	{"nodes": ["4-5"], "coordinates": [[-8.68448, 115....	654	N
5	5	6	{"nodes": ["5-6"], "coordinates": [[-8.67866, 115....	961	N
6	6	7	{"nodes": ["6-7"], "coordinates": [[-8.67708, 115....	292	N
7	7	8	{"nodes": ["7-8"], "coordinates": [[-8.67602, 115....	571	N
8	8	9	{"nodes": ["8-9"], "coordinates": [[-8.67754, 115....	810	N
9	9	8	{"nodes": ["9-8"], "coordinates": [[-8.68002, 115....	1882	N

Gambar 4.2 Cuplikan Basis Data Tabel graph

Pada Kode Sumber 4.1 dapat dilihat bahwa untuk “id” bertipe data *integer* serta “simpul_awal” dan “simpul_tujuan” bertipe data *varchar*. Atribut “jalur” memiliki tipe *text* dan “bobot” bertipe *double*. Sedangkan atribut “temp” memiliki tipe data *char* yang akan bernilai default ‘N’ jika tidak ada data yang dimasukkan. Untuk *primary key* tabel ini ialah “id”.

```
INSERT INTO `graph` (`id`, `simpul_awal`, `simpul_tujuan`,
`jalur`, `bobot`, `temp`) VALUES (6, '6', '7', '{"nodes":
["6-7"], "coordinates": [[-8.67708, 115.21744],[-8.67701,
115.21731],[-8.67692, 115.21718],[-8.67684, 115.21704],[-
8.67671, 115.21681],[-8.6767, 115.2168],[-8.6766,
115.2166],[-8.67651, 115.21643],[-8.67642, 115.21626],[-
8.67639, 115.21618],[-8.67638, 115.21611],[-8.67635,
115.21597],[-8.67634, 115.21529],[-8.67608, 115.21529],[-
8.67602, 115.2153]], "distance metres": [292]}', 292,
'N');
```

Kode Sumber 4.2 Menyisipkan Data pada Tabel graph

Kode Sumber 4.2 merupakan salah satu contoh data yang disimpan dalam tabel graph.

4.3.2 Implementasi Tabel angkutan_umum

Implementasi tabel angkutan_umum berguna untuk menyimpan data rute mana saja yang dilalui oleh trayek pengumpan yang sudah beroperasi. Tabel angkutan_umum memiliki atribut "id", "no_trayek", dan "simpul" dimana atribut "simpul" memuat jalur yang dilalui oleh trayek berdasarkan simpul awal dan tujuan pada tabel graph. Untuk implementasi tabel dan hasil dapat dilihat pada Kode Sumber 4.3 dan Gambar 4.3.

```
CREATE TABLE `angkutan_umum` (
  `id` int(10) NOT NULL AUTO INCREMENT,
  `no trayek` varchar(10) NOT NULL,
  `simpul` text NOT NULL,
  PRIMARY KEY (`id`));
```

Kode Sumber 4.3 Membuat Tabel angkutan_umum

id	no_trayek	simpul
0	TP01	,14-2,2-0,0-12,12-13,13-14,
1	TP02	,1-2,2-3,3-4,4-5,5-6,6-7,7-8,8-9,9-8,8-7,7-11,11-1...
2	TP03	,7-8,8-9,9-10,10-7,
3	TP04	,7-11,11-13,13-14,14-0,0-12,12-6,6-5,5-3,3-4,4-10,...

Gambar 4.3 Cuplikan Basis Data Tabel angkutan_umum

Pada Kode Sumber 4.3 dapat dilihat bahwa untuk "id" bertipe data *integer* serta "no_trayek" bertipe *varchar* dan "simpul" bertipe data *text*. Untuk *primary key* tabel ini ialah "id".

```
INSERT INTO `angkutan_umum` (`id`, `no_trayek`, `simpul`)
VALUES (0, 'TP01', ',14-2,2-0,0-12,12-13,13-14,');
```

Kode Sumber 4.4 Menyisipkan Data pada Tabel angkutan_umum

Kode Sumber 4.4 merupakan salah satu contoh data yang disimpan dalam tabel angkutan_umum.

4.3.3 Implementasi Tabel halte_bus

Implementasi tabel halte_bus digunakan untuk menyimpan data halte bus Trans Sarbagita beserta informasi berupa nama halte, koridor yang melalui halte bus dan halte selanjutnya yang akan dilalui. Tabel halte_bus memiliki atribut "id", "nama_halte", "koordinat", "next_halte "dan "koridor". Untuk implementasi tabel dan hasil dapat dilihat pada Kode Sumber 4.5 dan Gambar 4.4.

```
CREATE TABLE `halte_bus` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `nama halte` varchar(50) DEFAULT NULL,
  `koordinat` text,
  `next_halte` varchar(10) DEFAULT NULL,
  `koridor` varchar(10) NOT NULL,
  PRIMARY KEY (`id`));
```

Kode Sumber 4.5 Membuat Tabel halte_bus

id	nama_halte	koordinat	next_halte	koridor
1	Kamboja	-8.6511808,115.2244447	2	1
2	Surapati	-8.656535, 115.220718	3	1
3	Sudirman 2	-8.6716668,115.2182199	4	1
4	Sudirman 3	-8.6755317,115.2177339	5	1
5	Pesanggaran 1	-8.7168465,115.2133202	6	1-2

Gambar 4.4 Cuplikan Basis Data Tabel halte_bus

Pada Kode Sumber 4.5 dapat dilihat bahwa untuk "id" bertipe data *integer* serta "nama_halte", "next_halte", dan "koridor" bertipe *varchar* sedangkan "koordinat" bertipe data *text*. Untuk *primary key* tabel ini ialah "id".

```
INSERT INTO `halte bus` (`id`, `nama halte`, `koordinat`,
`next_halte`, `koridor`) VALUES
(1, 'Kamboja', '-8.6511808,115.2244447', '2', '1');
```

Kode Sumber 4.6 Menyisipkan Data pada Tabel halte_bus

Kode Sumber 4.6 merupakan salah satu contoh data yang disimpan dalam tabel halte_bus.

4.3.4 Implementasi Tabel rute_bus

Implementasi tabel rute_bus berguna untuk menyimpan data rute yang dilalui bus untuk memberikan visualisasi dari halte-halte yang berdekatan. Tabel rute_bus memiliki atribut "id", "kordior", dan "jalur" dimana atribut "jalur" memuat jalur yang dilalui oleh bus berdasarkan koridor. Untuk implementasi tabel dan hasil dapat dilihat pada Kode Sumber 4.7 dan Gambar 4.5.

```
CREATE TABLE `rute bus` (
  `id` int(11) NOT NULL,
  `koridor` varchar(10) NOT NULL,
  `jalur` text NOT NULL
   PRIMARY KEY (`id`));
```

Kode Sumber 4.7 Membuat Tabel rute_bus

id	koridor	jalur
1	1a	{"coordinates":[[115.2139, -8.71685],[115.21416, -...
2	1b	{"coordinates":[[115.18081, -8.78232],[115.18127, ...
3	2a	{"coordinates":[[115.2139, -8.71685],[115.21416, -...
4	2b	{"coordinates":[[115.18081, -8.78232],[115.18127

Gambar 4.5 Cuplikan Basis Data Tabel rute_bus

Pada Kode Sumber 4.7 terlihat atribut "id" bertipe data *integer*, "koridor" bertipe *varchar* dan atribut"jalur" bertipe data *text*. Untuk *primary key* tabel ini ialah "id".

```
INSERT INTO `rute bus` (`id`, `koridor`, `jalur`) VALUES
(1, '1a', '{"coordinates":[[115.2139, -
8.71685],[115.21416, -8.71692],[115.21458, -
8.71699],[115.215, -8.71706],[115.21505, -
8.7169],[115.21516, -8.71654],[115.21516, -
8.71653],[115.21522, -8.7164],[115.21525, -
8.71637],[115.21526, -8.71635],[115.21532,
```

```
.................................................
[115.2151, -8.71709],[115.21507, -8.71721],[115.21497, -
8.71719],[115.21467, -8.71714],[115.21368, -
8.7169],[115.21333, -8.71681]]}')
```

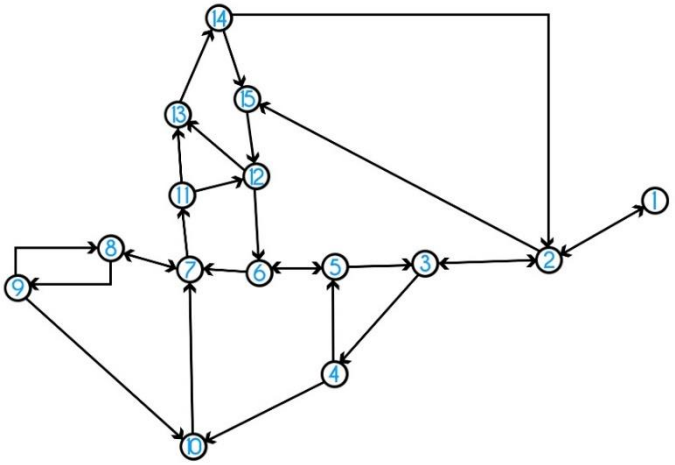
Kode Sumber 4.8 Menyisipkan Data pada Tabel rute_bus

Kode Sumber 4.8 merupakan cuplikan kode sumber dari salah satu data yang dimasukkan ke dalam tabel rute_bus.

4.4 Implementasi Dijkstra

Algoritma Dijkstra diimplementasikan menggunakan PHP, dimana algoritma ini akan dijalankan dalam proses *web service* layanan pencarian rute terpendek. Sebelum Dijkstra dijalankan, sistem akan mencari dan membuat node baru berdasarkan lokasi terdekat dari masukan pengguna pada rute yang dilewati trayek.

Graf yang diproses oleh algoritma Dijkstra didapat dari hasil rute trayek pengumpan pada Gambar 4.1, dimana dapat digambarkan sebagai graf berarah seperti pada Gambar 4.6 dengan beban simpul yang dapat dilihat pada Tabel 4.3.



Gambar 4.6 Visualisasi Graf dari Rute Trayek Pengumpan

Tabel 4.3 Beban antar Simpul pada Graf

Simpul Awal	Simpul Tujuan	Beban
1	2	3678
2	3	1360
3	4	1203
4	5	654
5	6	961
6	7	292
7	8	571
8	9	810
9	8	1882
8	7	571
7	11	121
11	12	456
12	6	393
6	5	961
5	3	558
3	2	1360
2	1	3678
14	2	8497
2	15	3302
15	12	425
12	13	730
13	14	1414
9	10	9361
10	7	2785
11	13	608
14	15	1362
4	10	2722

Kode sumber dari implementasi algoritma Dijsktra dapat dilihat pada lampiran A Kode Sumber A.1.

4.5 Implementasi *Web Service*

Implementasi pembuatan *web service* ini menggunakan PHP. Pada bagian ini, terdapat subbagian yang menjelaskan implementasi tiap layanan beserta penggunaannya. *Web service* ini akan menjadi penghubung antara basis data dan aplikasi yang dibangun pada perangkat bergerak.

4.5.1 Implementasi Layanan Pencarian Rute Terpendek

Layanan pencarian rute terpendek dapat diakses dengan permintaan berupa *string* URL dengan format sebagai berikut:

```
http://yasurya.com/tbo/Main.php?koord_user=jalan%20dewi%20
sartika%20denpasar&koord_destination=jalan%20diponegoro%20
denpasar
```

Dimana koord_user dan koord_destination adalah lokasi keberangkatan dan tujuan yang merupakan variabel berupa *text* yang nantinya akan didapatkan lokasi *latitude* dan *longitude* melalui proses *geocoding*. Dari contoh permintaan di atas akan menghasilkan respon seperti berikut:

```
{
  "jalur_shortest_path": [
    {"lat": -8.66964,"lng": 115.21677},
    {"lat": -8.66959,"lng": 115.2163},
    {"lat": -8.66958,"lng": 115.21617},
    {"lat": -8.66955,"lng": 115.21593},
    {"lat": -8.66948,"lng": 115.21548},
    {"lat": -8.66945,"lng": 115.21546},
    {"lat": -8.66855,"lng": 115.21546},
    {"lat": -8.66854,"lng": 115.21546},
    {"lat": -8.66834,"lng": 115.21546},
    {"lat": -8.66822,"lng": 115.21546}],
  "jarak": 7711.8210003349,
  "angkot": [
    {"koordinat_angkot": {
        "lat": -8.66964,
        "lng": 115.21677},
      "no_angkot": "TP01"}],
   "dataawal":
```

```
        {"origin":[-8.6696404,115.2167727],
        "destination":[-8.668216,115.2154639]}
}
```

Respon dari layanan di atas memberikan kumpulan titik berupa *latitude* dan *longitude* yang terdapat pada *array* JSON jalur_shortest_path, dimana titik-titik tersebut akan divisualisasikan menjadi *polyline* pada tampilan fragmen Google Maps. *Array* JSON angkot akan divisualisasikan sebagai *marker* dimana posisi *marker* ditentukan oleh objek JSON koordinat_angkot dan nama *marker* berisi data yang diambil dari objek JSON no_angkot. Objek JSON dataawal juga akan divisualisasikan sebagai *marker* yang menunjukkan lokasi keberangkatan dan tujuan yang dimasukkan oleh pengguna. Implementasi kode sumber dari layanan pencarian rute terpendek dapat dilihat pada lampiran A Kode Sumber A.2.

4.5.2 Implementasi Layanan Rute Trayek Pengumpan

Layanan rute trayek pengumpan dapat diakses dengan permintaan berupa *string* URL dengan format sebagai berikut:

```
http://yasurya.com/tbo/ShowRoute.php?no_trayek=2
```

Dimana no_trayek yang merupakan variabel berupa *text* yang digunakan untuk mencari data trayek dan mendapatkan jalur yang dilalui trayek tersebut. Dari contoh permintaan di atas akan menghasilkan respon seperti berikut:

```
{
  "jalur": [
    {"lat": -8.67602,"lng": 115.2153},
    {"lat": -8.67599,"lng": 115.21437},
    {"lat": -8.67593,"lng": 115.21404},
    {"lat": -8.67592,"lng": 115.21396},
    {"lat": -8.67592,"lng": 115.21393},
    {"lat": -8.67593, "lng": 115.21348},
    {"lat": -8.67593,"lng": 115.21339},
    {"lat": -8.67593,"lng": 115.2133},
    ..................................
    {"lat": -8.67643,"lng": 115.21528},
```

```
    {"lat": -8.67642,"lng": 115.21528},
    {"lat": -8.67634,"lng": 115.21529},
    {"lat": -8.67608,"lng": 115.21529},
    {"lat": -8.67602,"lng": 115.2153}],
    "koordinat": [
    {"lat": -8.67621, "lng": 115.21328},
    {"lat": -8.6801,  "lng": 115.20975},
    {"lat": -8.70631,  "lng": 115.19819},
    {"lat": -8.68801,  "lng": 115.21687}],
  "no_trayek": 3
}
```

Layanan rute trayek pengumpan memberikan respon berupa objek JSON yang berisikan sebuah *array* JSON dengan nama jalur. *Array* JSON jalur memberikan informasi berupa titik-titik *latitude* dan *longitude* yang akan divisualisasikan menjadi *polyline* pada tampilan fragmen Google Maps sehingga membentuk rute trayek sesuai dengan masukan dan respon yang diterima. Implementasi kode sumber dari layanan pencarian rute terpendek dapat dilihat pada lampiran A Kode Sumber A.3.

4.5.3 Implementasi Layanan Lokasi Halte Bus

Layanan lokasi halte bus dapat diakses dengan permintaan berupa *string* URL dengan format sebagai berikut:

```
http://yasurya.com/tbo/ShowHalte.php
```

Layanan ini memberi semua data halte bus yang ada pada basis data untuk ditampilkan sebagai *marker* yang berisikan informasi halte pada antarmuka perangkat. Dari contoh permintaan di atas akan menghasilkan respon seperti berikut:

```
[
  {
    "koordinat": "-8.6511808,115.2244447",
    "nama": "Kamboja",
    "haltenext": "Surapati",
    "halteprev": "Surapati 1"
    "koridor": "1"
  },
```

```
  {
    "koordinat": "-8.656535, 115.220718",
    "nama": "Surapati",
    "haltenext": "Sudirman 2",
    "halteprev": "Kamboja"
    "koridor": "1"
  },
  {
    "koordinat": "-8.6716668,115.2182199",
    "nama": "Sudirman 2",
    "haltenext": "Sudirman 3",
    "halteprev": "Surapati"
    "koridor": "1"
  },
  ..................
  {
    "koordinat": "-8.6504232,115.2545972",
    "nama": "Prof. I.B. Mantra",
    "haltenext": "Tohpati",
    "halteprev": "Waribang b"
    "koridor": "2"
  },
  {
    "koordinat": "-8.6409781,115.2544506",
    "nama": "Tohpati",
    "haltenext": "Siulan b",
    "halteprev": "Prof. I.B. Mantra"
    "koridor": "2"
  },
  {
    "koordinat": "-8.6337302,115.2569607",
    "nama": "Siulan b",
    "haltenext": "Batubulan",
    "halteprev": "Tohpati"
    "koridor": "2"
  }
]
```

Layanan lokasi halte bus memberikan respon berupa array JSON yang berisikan sekumpulan data halte yang akan divisualisasikan sebagai marker nantinya. Data halte berisikan koordinat untuk memposisikan marker pada Google Maps, nama halte yang menjadi nama marker dan informasi halte lainnya berupa halte selanjutnya yang didapat dari objek JSON dengan nama haltenext, halte sebelumya yang didapat dari objek JSON bernama halteprev, dan koridor dari halte tersebut. Implementasi

kode sumber dari layanan pencarian rute terpendek dapat dilihat pada lampiran A Kode Sumber A.4.

4.5.4 Implementasi Layanan Rute Bus Trans Sarbagita

Layanan rute bus Trans Sarbagita dapat diakses dengan permintaan berupa *string* URL dengan format sebagai berikut:

```
http://yasurya.com/tbo/Showallrute.php
```

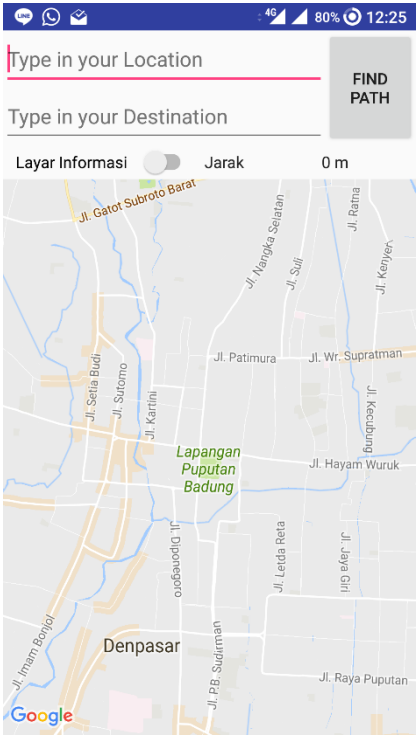
Layanan ini akan memberi semua data rute bus yang ada pada basis data untuk ditampilkan berupa polyline yang membentuk rute bus dan menghubungkan halte-halte bus yang ada. Dari contoh permintaan di atas akan menghasilkan respon seperti berikut:

```
[
  {
    "0": [
      { "lat": -8.71685,  "lng": 115.2139 },
      { "lat": -8.71692,  "lng": 115.21416},
      ..................................
      {"lat": -8.7169, "lng": 115.21368},
      {"lat": -8.71681, "lng": 115.21333}
    ] },
  {
    "1": [
      {"lat": -8.78232, "lng": 115.18081},
      {"lat": -8.78251, "lng": 115.18127},
       ..................................
      {"lat": -8.71681, "lng": 115.2137},
      {"lat": -8.71685, "lng": 115.2139}]
  }
]
```

Layanan rute bus memberikan respon berupa *array* JSON yang berisikan sekumpulan data rute per koridornya dimana data rute merupakan sekumpulan titik di dalam *array* JSON lainnya. Implementasi kode sumber dari layanan pencarian rute terpendek dapat dilihat pada lampiran A Kode Sumber A.5.

4.6 Implementasi Antarmuka

Implementasi antarmuka pada perangkat dapat memberikan visualisasi rute dan informasi mengenai halte Trans Sarbagita. Implementasi antarmuka dibangun sebagai *layout* utama dalam *project* pembangunan aplikasi Trans Sarbagita pada perangkat bergerak, dimana layout ini memiliki format .xml. Cuplikan antarmuka dapat dilihat pada Gambar 4.7 dimana implementasi antarmuka sebagian besar sudah memenuhi kebutuhan dari perancangan antarmuka.



Gambar 4.7 Implementasi Antarmuka

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Pada bab ini akan membahas uji coba dan evaluasi dari sistem yang dibuat. Sistem akan diuji coba fungsionalitasnya dengan menjalankan skenario yang sudah ditentukan. Uji coba dilakukan untuk mengetahui hasil dari sistem ini sehingga menjawab rumusan masalah pada tugas akhir ini. Selain itu Uji coba pengguna juga akan dilakukan untuk mendapatkan umpan balik mengenai sistem dari sudut pandang pengguna.

5.1 Lingkungan Uji Coba

Lingkungan pengujian adalah lingkungan, baik perangkat keras maupun perangkat lunak tempat pengujian sistem dilakukan. Uji coba fungsionalitas untuk layanan informasi Trans Sarbagita dilakukan pada perangkat Android dengan spesifikasi pada Tabel 5.1.

Tabel 5.1 Spesifikasi Sistem Pengujian

Spesifikasi	Deskripsi
CPU	Qualcomm MSM8994 Snapdragon 810 @ 4x1.56 GHz Cortex-A53 & 4x1.82 GHz Cortex-A57
RAM	4.00 GB
Sistem Operasi	Android Platform versi API 23: Android 6.0 (Marshmallow) revision 3

5.2 Uji Coba Fungsionalitas

Pengujian sistem akan dilakukan dengan uji coba fungsionalitas yang menggunakan metode *Black-box*. Metode *Black-box* merupakan metode pengujian yang ditekankan pada pola *input* dan *output* yang sesuai dengan skenario. Selain itu akan dilakukan juga pengujian pengguna yang dimana pengujian ini

dilakukan untuk mendapatkan umpan balik mengenai aplikasi yang sedang dibangun.

5.2.1 Kasus Pengujian Pencarian Rute Terpendek Kondisi Normal

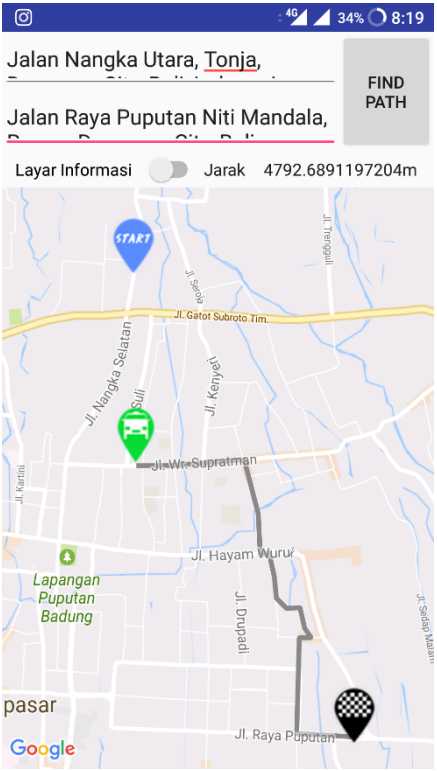
Pada kasus uji ini, pengguna dapat mencari rute terpendek yang dapat dilalui berdasarkan lokasi keberangkatan dan lokasi tujuan. Detail pengujian berupa skenario, kondisi awal, data uji, dan kondisi akhir yang tercantum pada Tabel 5.2.

Tabel 5.2 Skenario Pencarian Rute Terpendek

ID	UJ-001
Referensi Kasus Penggunaan	UC-001
Nama	Pengujian Pencarian Rute Terpendek Kondisi Normal
Tujuan	Menguji fitur pencarian rute terpendek dalam melakukan pencarian rute dengan benar.
Skenario 1	Pengguna melakukan pencarian rute terpendek.
Kondisi Awal	Pengguna belum melakukan pencarian dan disuguhkan tampilan antarmuka aplikasi.
Data Uji	1. Masukan pengguna berupa lokasi keberangkatan. 2. Masukan pengguna berupa lokasi tujuan.
Langkah Pengujian	1. Pengguna mengakses aplikasi Trans Sarbagita. 2. Pengguna memasukkan data berupa *text* lokasi keberangkatan dan lokasi tujuan. 3. Pengguna menekan tombol "*Find Route*" untuk menjalankan proses pencarian rute.
Hasil yang Diharapkan	Rute terpendek serta informasi mengenai trayek yang melalui rute tersebut terlihat.

Hasil yang Didapat	Ditampilkannya rute terpendek sesuai dengan masukan pengguna.
Kondisi Akhir	Pada *fragment* Google Maps ditampilkan *polyline* yang membentuk rute terpendek.

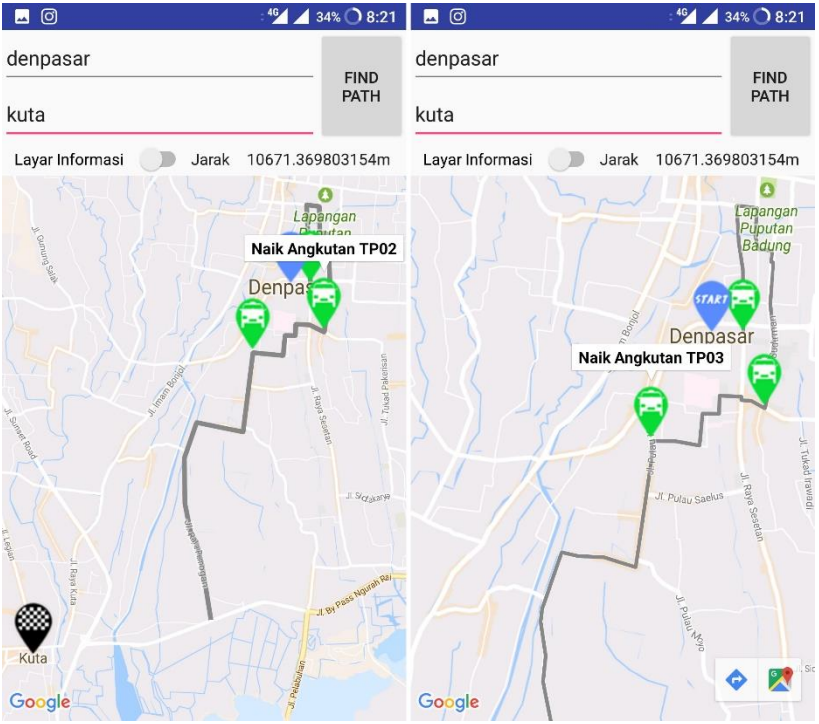
Bentuk pengisian lokasi keberangkatan dan lokasi tujuan yang disediakan ialah kolom teks yang sudah ditambahkan fungsi *autocomplete* yang diambil dari API Google Maps.



Gambar 5.1 Hasil Uji Coba Tanpa Pergantian Trayek

Gambar 5.1 menampilkan rute dari Jalan Nangka Utara yang ditandai dengan marker biru menuju Jalan Raya Puputan yang

ditandai dengan marker hitam. Marker hijau menandakan rute terdekat yang dilewati trayek dari lokasi keberangkatan. Dan marker hijau berisi informasi trayek mana yang bisa digunakan.



Gambar 5.2 Hasil Uji Coba dengan Pergantian Trayek

Gambar 5.2 menampilkan rute terpendek dari Denpasar menuju Kuta dengan marker yang sama dengan hasil uji coba sebelumnya. Namun pada uji coba kali ini terlihat adanya pergantian trayek sebanyak 2 kali dimana ditandai dengan adanya marker hijau lebih dari satu. Setiap marker hijau berisikan informasi berbeda mengenai trayek mana yang harus digunakan.

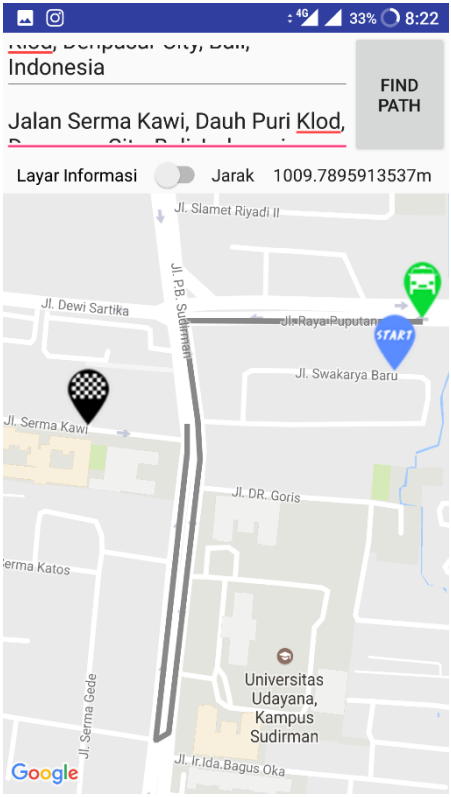
5.2.2 Kasus Pengujian Pencarian Rute Terpendek Kondisi Keberangkatan dan Tujuan Berseberangan

Pada kasus uji ini, pengguna dapat mencari rute terpendek yang dapat dilalui dengan kondisi lokasi keberangkatan dan lokasi tujuan berseberangan. Detail pengujian berupa skenario, kondisi awal, data uji, dan kondisi akhir yang tercantum pada Tabel 5.3.

Tabel 5.3 Skenario Pencarian Rute Terpendek dengan Lokasi Berseberangan

ID	UJ-002
Referensi Kasus Penggunaan	UC-001
Nama	Pengujian Pencarian Rute Terpendek Kondisi Keberangkatan dan Tujuan Berseberangan
Tujuan	Menguji fitur pencarian rute terpendek dalam melakukan pencarian rute dengan benar.
Skenario 1	Pengguna melakukan pencarian rute terpendek.
Kondisi Awal	Pengguna belum melakukan pencarian dan disuguhkan tampilan antarmuka aplikasi.
Data Uji	1. Masukan pengguna berupa lokasi keberangkatan yang berseberangan dengan lokasi tujuan. 2. Masukan pengguna berupa lokasi tujuan yang berseberangan dengan lokasi keberangkatan.
Langkah Pengujian	1. Pengguna mengakses aplikasi Trans Sarbagita. 2. Pengguna memasukkan data berupa *text* lokasi keberangkatan dan lokasi tujuan yang berseberangan. 3. Pengguna menekan tombol "*Find Route*" untuk menjalankan proses pencarian rute.

Hasil yang Diharapkan	Rute terpendek serta informasi mengenai trayek yang melalui rute tersebut ditampilkan.
Hasil yang Didapat	Ditampilkannya rute terpendek sesuai dengan masukan pengguna.
Kondisi Akhir	Pada *fragment* Google Maps ditampilkan *polyline* yang membentuk rute terpendek.



Gambar 5.3 Hasil Uji Coba Rute Terpendek dengan Lokasi Berseberangan

Gambar 5.3 memperlihatkan hasil pencarian rute terpendek dengan lokasi berseberangan. Hasil rute memperlihatkan

bahwa rute tidak berhenti di tengah melainkan harus mengambil jalan memutar dikarenakan lokasi rute terdekat dengan lokasi tujuan adalah pada titik akhir yang dapat dilihat pada Gambar 5.3.

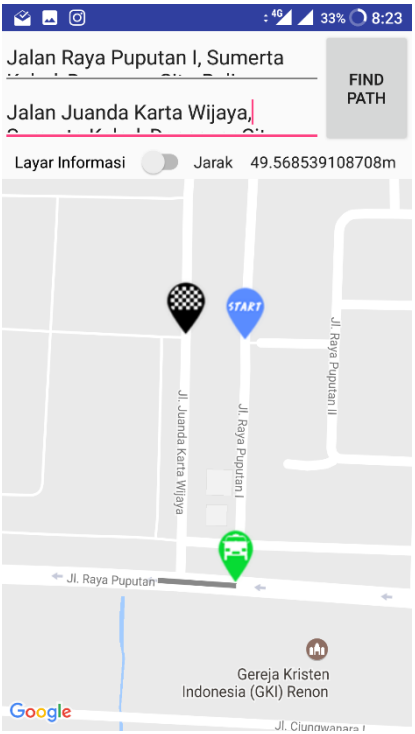
5.2.3 Kasus Pengujian Pencarian Rute Terpendek Kondisi Berdekatan

Kasus Pengujian pencarian rute terpendek dengan lokasi keberangkata dan tujuan saling berdekatan bertujuan untuk menguji fungsi sistem. Dalam kasus ini dibutuhkan masukan pengguna berupa lokasi awal dan lokasi tujuan yang saling berdekatan. Detail pengujian berupa scenario, kondisi awal, data uji, dan kondisi akhir tercantum pada Tabel 5.4.

Tabel 5.4 Skenario Uji Coba Pencarian Rute Terpendek dengan Lokasi yang Berdekatan

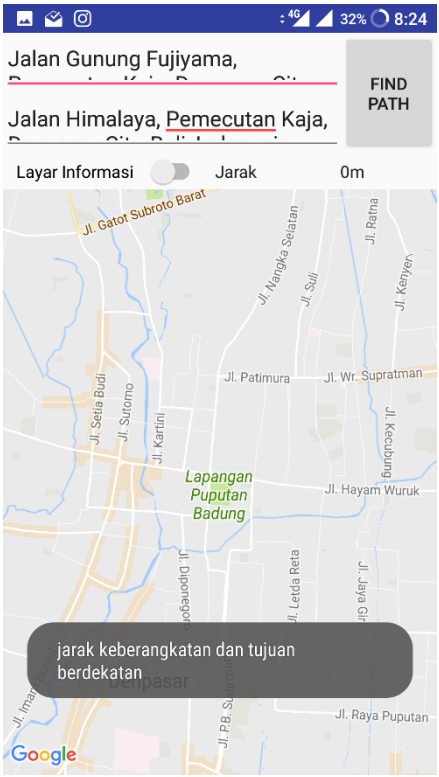
ID	UJ-003
Referensi Kasus Penggunaan	UC-001
Nama	Pengujian Pencarian Rute Terpendek Kondisi Keberangkatan dan Tujuan Berdekatan
Tujuan	Menguji fitur pencarian rute terpendek dalam melakukan pencarian rute dengan benar.
Skenario 1	Pengguna melakukan pencarian rute terpendek.
Kondisi Awal	Pengguna belum melakukan pencarian dan disuguhkan tampilan antarmuka aplikasi.
Data Uji	1. Masukan pengguna berupa lokasi keberangkatan yang berdekatan dengan lokasi tujuan. 2. Masukan pengguna berupa lokasi tujuan yang berdekatan dengan lokasi keberangkatan.

Langkah Pengujian	1. Pengguna mengakses aplikasi Trans Sarbagita. 2. Pengguna memasukkan data berupa *text* lokasi keberangkatan dan lokasi tujuan yang berseberangan. 3. Pengguna menekan tombol "*Find Route*" untuk menjalankan proses pencarian rute.
Hasil yang Diharapkan	1. Rute terpendek serta informasi mengenai trayek yang melalui rute tersebut ditampilkan. 2. Menampilkan peringatan jika titik keberangkatan pada rute sama dengan titik tujuan pada rute.
Hasil yang Didapat	1. Ditampilkannya rute terpendek sesuai dengan masukan pengguna. 2. Dtiampilkannya peringatan pada antarmuka.
Kondisi Akhir	1. Pada fragmen Google Maps ditampilkan *polyline* yang membentuk rute terpendek. 2. Pada tampilan antarmuka ditampilkan peringatan.



Gambar 5.4 Hasil Uji Coba Rute Terpendek dengan Jarak yang Berdekatan

Gambar 5.4 menampilkan hasil pencarian rute terpendek dengan lokasi awal dan tujuan yang berdekatan. Rute ditampilkan karena lokasi keberangkata dan tujuan pada rute adalah berbeda. Sedangkan Gambar 5.5 menampilkan peringatan yang menyebutkan bahwa lokasi berdekatan, hal ini dikarenakan titik awal pada rute sama dengan titik tujuan pada rute.



Gambar 5.5 Peringatan yang ditampilkan jika Titik Awal pada Rute sama dengan Titik Tujuan

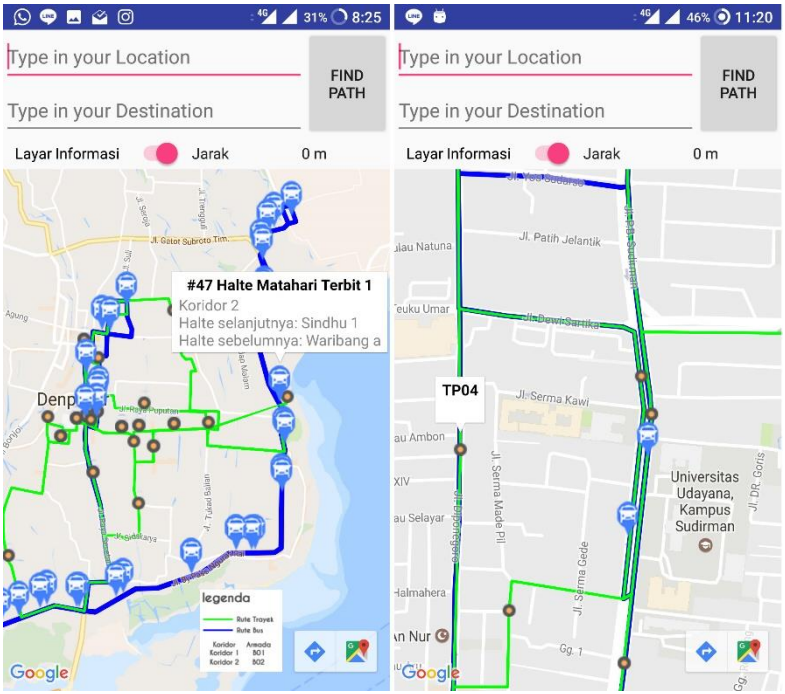
5.2.4 Kasus Pengujian Menampilkan Layar Informasi

Pada kasus uji menampilkan layar informasi, pengguna dapat melihat layar informasi Trans Sarbagita yang berisikan visualisasi rute trayek, rute bus, serta halte bus beserta informasinya. Detail pengujian berupa skenario, kondisi awal, data uji dan kondisi akhir yang tercantum pada Tabel 5.5.

Tabel 5.5 Skenario Menampilkan Layar Halte

ID	UJ-004
Referensi Kasus Penggunaan	UC-002, UC-003, UC-004
Nama	Pengujian Menampilkan Layar Informasi
Tujuan	Menguji fitur menampilkan layar informasi yang berisikan visualisasi rute trayek, rute bus, serta penanda halte.
Skenario 1	Pengguna melihat seluruh informasi Trans Sarbagita.
Kondisi Awal	Pengguna disuguhkan tampilan antarmuka aplikasi.
Data Uji	-
Langkah Pengujian	1. Pengguna mengakses aplikasi Trans Sarbagita. 2. Pengguna menekan *switch* untuk menampilkan layar informasi.
Hasil yang Diharapkan	Seluruh rute trayek, rute bus dan halte yang sudah tersedia ditampilkan.
Hasil yang Didapat	Sistem menampilkan rute trayek, rute bus, dan penanda halte beserta infromasinya.
Kondisi Akhir	Pada fragmen Google Maps ditampilkan *polyline* yang membentuk rute trayek pengumpan dan rute bus. Halte juga ditampilkan berupa *marker* yang berisikan informasi halte tersebut.

Informasi yang ditampilkan didapat dengan mengaktifkan switch yang ada pada antarmuka.



Gambar 5.6 Tampilan Layar Informasi yang Diaktifkan

Gambar 5.6 menampilkan rute trayek pengumpan yang divisualisasikan dengan warna hijau dan rute bus yang divisualisasikan dengan warna biru. Halte beserta informasinya juga ditampilkan sebagai *marker* dimana informasi yang ditampilkan adalah kode halte, nama halte, koridor, halte selanjutnya, dan halte sebelumnya. Pencarian rute terpendek dapat dilakukan saat layar informasi tetap aktif. Layar informasi dapat dihilangkan dengan menonaktifkan *switch* layar informasi.

5.3 Uji Coba Pengguna

Uji coba pengguna dilakukan secara daring. Uji coba yang dilakukan dengan memberikan pengguna tautan untuk mengunduh *Android application package* (apk) agar aplikasi dapat di-install pada perangkat pengguna. Setelah itu pengguna diharapkan untuk mencoba aplikasi dan akan diminta untuk mengisi kuisioner pada tautan yang mengarahkan ke form kuisioner.

Kuisioner yang disiapkan menerapkan kuisioner dalam bentuk skala Likert, dimana kuisioner memiliki empat pernyataan untuk mengetahui performa aplikasi dari sudut pandang pengguna. Berikut merupakan pernyataan yang ada dalam kuisioner uji coba pengguna.

Tabel 5.6 Daftar Pernyataan dalam Uji Coba Pengguna

Pernyataan	Isi Pernyataan	Penilaian
1	Aplikasi memudahkan pengguna dalam mendapatkan informasi mengenai angkutan Sarbagita.	Skala Likert (1-4)
2	Aplikasi memproses permintaan pengguna secara cepat dan tepat.	Skala Likert (1-4)
3	Aplikasi memberikan hasil rute trayek pengumpan secara tepat.	Skala Likert (1-4)
4	Tampilan aplikasi mudah dimengerti.	Skala Likert (1-4)

.

Tipe pernyataan yang bersifat positif akan dinilai dengan memilih antara 4 poin jawaban, yaitu sangat setuju, setuju, tidak setuju, sangat tidak setuju. Poin netral sengaja dihilangkan agar mendapatkan hasil yang tidak bias [17].

Hasil interpretasi dapat dihitung dengan dengan cara membagi total skor dengan skor tertinggi dan kemudian dikalikan dengan 100% untuk mendapat persentase.

$$I = \frac{X}{y \times z} \times 100\%$$

dimana I adalah hasil persentase interpretasi, X merupakan total skor, y merupakan skor tertinggi dan z merupakan jumlah responden yang ada. Interpretasi perhitungan didapatkan dengan memberikan nilai interval. Berikut kriteria interpretasi skor berdasarkan interval:

- Angka 0% - 24,99% = Sangat Tidak Setuju
- Angka 25% - 49,99% = Tidak Setuju
- Angka 50% - 74,99% = Setuju
- Angka 75% - 100% = Sangat Setuju

Nilai skor pada penialian skala Likert adalah sebagai berikut:

- Sangat Setuju = 4
- Setuju = 3
- Tidak Setuju = 2
- Sangat Tidak Setuju = 1

Uji coba pengguna dilakukan dengan 21 responden. Tabel 5.7 menyediakan data pekerjaan responden dan Tabel 5.8 menyediakan data umur dari responden.

Tabel 5.7 Tabel Data Pekerjaan Responden

Pekerjaan	**Jumlah**
Pegawai	5
Mahasiswa	12
Swasta	3
Pengangguran	1

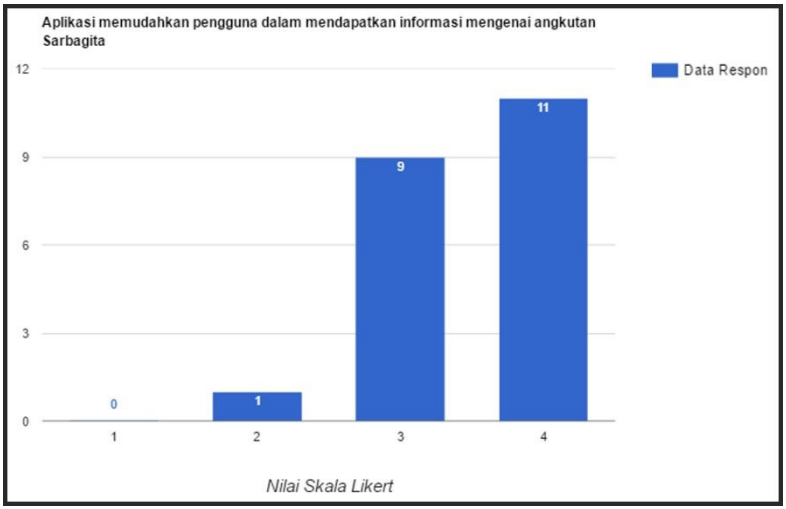
Tabel 5.8 Tabel Data Umur Responden

Umur	Jumlah
21	2
22	11
23	6
24	1
25	1

Data responden selengkapnya dapat dilihat pada lampiran B, Tabel B.1.

5.3.1 Hasil Analisis Pernyataan 1

Pernyataan "Aplikasi memudahkan pengguna dalam mendapatkan informasi mengenai angkutan Sarbagita" mendapatkan respon seperti yang dapat dilihat pada Gambar 5.7.



Gambar 5.7 Grafik Penilaian Pernyataan 1

Dari data pada Gambar 5.9 dapat dilakukan perhitungan sebagai berikut:

- Sangat setuju = 4 × 11 = 44
- Setuju = 3 × 9 = 27
- Tidak Setuju = 2 × 1 = 2
- Sangat Tidak Setuju = 1 × 0 = 0

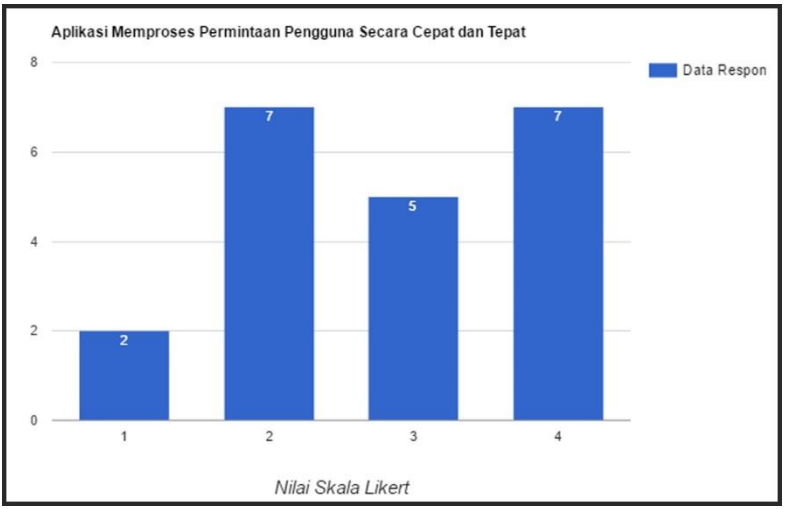
Total Skor = 44+27+2+0 = 72

$$Hasil\ Interpretasi = \frac{72}{4 \times 21} \times 100\% = 85{,}71\%$$

Dari hasil di atas, maka responden dapat dinyatakan sangat setuju bahwa aplikasi memudahkan pengguna dalam mendapatkan informasi mengenai angkutan Sarbagita.

5.3.2 Hasil Analisis Pernyataan 2

Pernyataan "Aplikasi memproses permintaan pengguna secara cepat dan tepat" mendapatkan respon seperti yang dapat dilihat pada Gambar 5.8.



Gambar 5.8 Grafik Penilaian Pernyataan 2

Dari data pada Gambar 5.10 dapat dilakukan perhitungan sebagai berikut:

- Sangat setuju = 4 × 7 = 28
- Setuju = 3 × 5 = 15
- Tidak Setuju = 2 × 7 = 14
- Sangat Tidak Setuju = 1 × 2 = 2

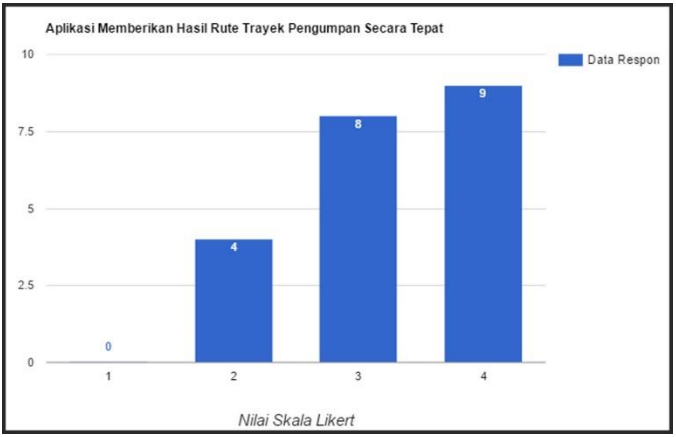
Total Skor = 44+21+2+0 = 59

$$Hasil\ Interpretasi = \frac{59}{4 \times 21} \times 100\% = 70{,}24\%$$

Dari hasil di atas, maka responden dapat dinyatakan setuju bahwa aplikasi memproses permintaan pengguna secara cepat dan tepat.

5.3.3 Hasil Analisis Pernyataan 3

Pernyataan "Aplikasi memberikan hasil rute trayek pengumpan secara tepat" mendapatkan respon seperti yang dapat dilihat pada Gambar 5.9.



Gambar 5.9 Grafik Penilaian Pernyataan 3

Dari data pada Gambar 5.11 dapat dilakukan perhitungan sebagai berikut:

- Sangat setuju = 4 × 9 = 36
- Setuju = 3 × 8 = 24
- Tidak Setuju = 2 × 4 = 8
- Sangat Tidak Setuju = 1 × 0 = 0

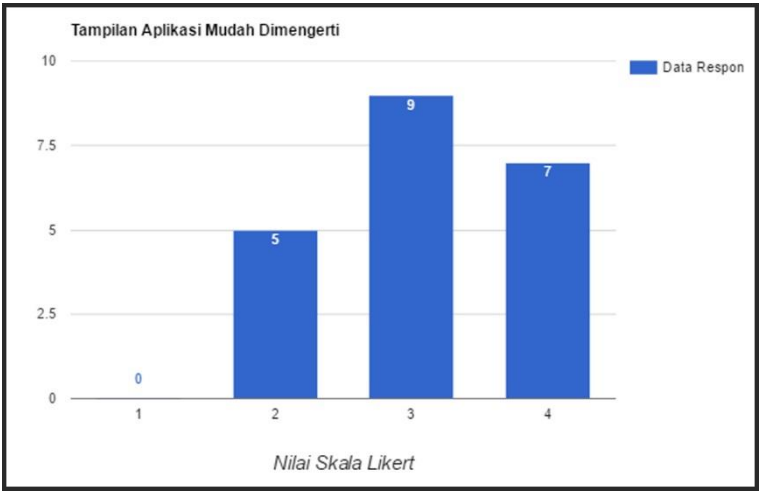
Total Skor = 44+21+8+0 = 68

$$Hasil\ Interpretasi = \frac{68}{4 \times 21} \times 100\% = 80{,}95\%$$

Dari hasil di atas, maka responden sangat setuju bahwa aplikasi Aplikasi memberikan hasil rute trayek pengumpan secara tepat.

5.3.4 Hasil Analisis Pernyataan 4

Pernyataan "Tampilan aplikasi mudah dimengerti" mendapatkan respon seperti yang dapat dilihat pada Gambar 5.10.



Gambar 5.10 Grafik Penilaian Pernyataan 4

Dari data pada Gambar 5.12 dapat dilakukan perhitungan sebagai berikut:

- Sangat setuju = 4 × 7 = 28
- Setuju = 3 × 9 = 27
- Tidak Setuju = 2 × 5 = 10
- Sangat Tidak Setuju = 1 × 0 = 0

Total Skor = 28+27+10+0 = 65

$$Hasil\ Interpretasi = \frac{65}{4 \times 21} \times 100\% = 77{,}38\%$$

Dari hasil di atas, maka responden sangat setuju bahwa Tampilan aplikasi mudah dimengerti.

5.3.5 Rangkuman Analisis

Penilaian uji coba pengguna mendapatkan persentase nilai yang dapat dilihat pada Tabel 5.9. Pernyataan "Aplikasi memproses permintaan pengguna secara cepat dan tepat" mendapatkan nilai persentase terendah dengan nilai 70,24% dimana pada fungsi memproses permintaan pengguna harus dioptimalkan lagi. Sedangkan pernyataan lain mendapat nilai persentase yang cukup memuaskan.

Tabel 5.9 Tabel Nilai Uji Coba Pengguna

Pernyataan	Isi Pernyataan	Nilai
1	Aplikasi memudahkan pengguna dalam mendapatkan informasi mengenai angkutan Sarbagita.	85,71%
2	Aplikasi memproses permintaan pengguna secara cepat dan tepat.	70,24%

3	Aplikasi memberikan hasil rute trayek pengumpan secara tepat.	80,95%
4	Tampilan aplikasi mudah dimengerti.	77,38%

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan dibahas mengenai kesimpulan yang dapat diambil dari perancangan sistem hingga hasil pengujian. Selain itu juga akan dibahas mengenai hasil yang sudah dicapai dan belum dicapai. Pada bab ini juga akan menjawab pertanyaan yang dikemukakan pada Bab 1. Pada penutup ini juga terdapat saran-saran untuk pengembangan selanjutnya.

6.1 Kesimpulan

Dalam proses pengerjaan Tugas Akhir yang melalui tahap perancangan, implementasi, serta uji coba, didapatkan kesimpulan sebagai berikut:

1. Desain peta digital untuk rute angkutan Trans Sarbagita berhasil diimplementasikan pada aplikasi dengan menggunakan Google Maps sebagai layanan utama dalam visualisasi peta dalam perangkat.
2. Aplikasi berhasil menampilkan rute angkutan Trans Sarbagita yang akan dilalui sesuai dengan lokasi keberangkatan dan tujuan yang ditentukan pengguna yang ditampilkan pada Google Maps dengan menampilkan *polyline* yang didapat dari titik-titik koordinat yang sudah diproses dengan algoritma Dijkstra.
3. Aplikasi berhasil mengkalkulasikan rute terpendek yang dapat dilalui pengguna dengan menggunakan algoritma Dijkstra dengan menggunakan titik temu angkutan sebagai simpul dan jarak antar simpul sebagai beban pada algoritma Dijkstra.
4. Aplikasi memiliki fungsi serta tampilan yang cukup memuaskan pengguna melihat nilai pada data uji coba pengguna cukup tinggi.

6.2 Saran

Adapun saran-saran yang diberikan untuk pengembangan sistem ini selanjutnya adalah:

1. Mengoptimalkan fitur, seperti fitur untuk memproses permintaan pengguna yang didapat kurang memuaskan berdasarkan penilaian uji coba pengguna.
2. Menambahkan fitur-fitur untuk menggabungkan pencarian rute trayek dan juga bus.
3. Melakukan kerjasama lebih lanjut dengan Dinas Perhubungan Informasi dan Komunikasi Provinsi Bali dalam pengembangan dan untuk memberikan pengguna informasi yang lebih mendetail.

DAFTAR PUSTAKA

[1] Badan Pusat Statistik Provinsi Bali, Bali Dalam Angka 2014, Denpasar: BPS Provinsi Bali, 2014.

[2] "Angkutan Umum Trans Sarbagita," Dinas Perhubungan Informasi dan Komunikasi Provinsi Bali, 18 Juni 2012. [Online]. Available: http://www.dishubinkom.baliprov.go.id/id/ANGKUTAN-UMUM-Trans-SARBAGITA. [Diakses 10 April 2016].

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest dan C. Stein, "Introduction to Algorithms, Second Edition," dalam *Section 24.3: Dijkstra's algorithm*, MIT Press and McGraw-Hill, 2001, pp. 595-601.

[4] "Web Service-Geocoding API," Google, 16 Desember 2016. [Online]. Available: https://developers.google.com/maps/documentation/geocoding/intro. [Diakses 18 Desember 2016].

[5] "Google Maps APIs | Google Developers," Google, [Online]. Available: https://developers.google.com/maps/. [Diakses 10 April 2016].

[6] "Android Developer," Google, [Online]. Available: https://developer.android.com/. [Diakses 3 Januari 2017].

[7] "Meet Android Studio," Google, [Online]. Available: https://developer.android.com/studio/intro/index.html. [Diakses 16 Desember 2016].

[8] J. Gosling, B. Joy, G. Steele, G. Bracha dan A. Buckley, The Java® Language Specification (Java SE 8 ed.), 2015.

[9] S. S. Sriparsa, JavaScript and JSON Essentials, Birmingham: Packt, 2013.

[10] "What is MySQL," Oracle Corporation, [Online]. Available: http://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html. [Diakses 18 Desember 2016].

[11] Web Services Architecture Working Group, “World Wide Web Consortium (W3C),” 2004. [Online]. Available: https://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/. [Diakses 15 April 2016].

[12] M. Gudgin, M. Hadley, N. Mendelsohn, J.-J. Moreau, H. F. Nielsen, A. Karmarkar dan Y. Lafon, “SOAP Version 1.2,” W3C, 27 April 2007. [Online]. Available: https://www.w3.org/TR/soap12/. [Diakses 6 April 2016].

[13] R. T. Fielding, “Representational State Transfer,” UCI, [Online]. Available: http://www.ics.uci.edu/~fielding/talks/webarch_9805/. [Diakses 6 April 2016].

[14] “February 2009 Web Server Survey,” Netcraft, 18 Februari 2009. [Online]. Available: https://news.netcraft.com/archives/2009/02/18/february_2009_web_server_survey.html. [Diakses 16 Desember 2016].

[15] “OS/Linux Distributions using Apache,” Security Space, 1 Agustus 2009. [Online]. Available: https://secure1.securityspace.com/s_survey/data/man.200907/apacheos.html. [Diakses 16 Desember 2016].

[16] PHP Documentation Group, “PHP: Hypertext Preporocessor,” The PHP Group, [Online]. Available: http://www.php.net/manual/en/intro-whatis.php. [Diakses 15 April 2016].

[17] W. M. Trochim, “Likert Scaling,” Research Method Knowledge Base, 2006. [Online]. Available: http://www.socialresearchmethods.net/kb/scallik.php. [Diakses 3 Januari 2017].

[18] A. Beaulieu, Learning SQL (2nd ed.), Sebastapol, CA, USA: O'Reilly, 2009.

[19] A. Imam, SQL Server 2000, Yogyakarta: Graha Ilmu, 2005.

[20] I. D. K. A. Pradnyana, *Permasalahan Transportasi di Kota Denpasar*, Denpasar, Bali, 2015.

[21] I. D. K. A. Pradnyana, *Program Layanan Angkutan Pengumpan Transarbagita (Feeder) dalam Upaya Pengembangan Angkutan Massal di Kota Denpasar*, Denpasar, Bali, 2015.

[Halaman ini sengaja dikosongkan]

LAMPIRAN

Bagian ini merupakan lampiran sebagai dokumen pelengkap dari buku Tugas Akhir, pada bagian ini diberikan informasi mengenai kode sumber dari sistem yang dibuat.

A. Kode Sumber

```
1    <?php
2    class Dijkstra
3    {
4       function jalurTerpendek($arg_graph, $simpulAwal,
     $simpulTujuan){
5
6               if($simpulAwal == $simpulTujuan){
7                       return
     json_encode(['status'=>'error','error'=>'lokasi_anda_
     sudah_dekat','teks'=>'Lokasi Anda Sudah
     Dekat','content'=>'']);
8               }
9
10              if(!array_key_exists($simpulAwal,
     $arg_graph)    ||    !array_key_exists($simpulTujuan,
     $arg_graph)){
11                      return
     print_r(json_encode(['status'=>'error','error'=>'simp
     ul_input_tidak_ditemukan','teks'=>"could not find the
     input     :     $simpulAwal     or     $simpulTujuan",
     'content'=>'']));
12              }
13              $graph                  = $arg_graph;
14           $simpul_awal      = $simpulAwal;
15           $simpul_maju      = $simpulAwal;
16           $simpul_tujuan    = $simpulTujuan;
17              $jml_simpul     = count($arg_graph);
18
19              $simpulYangDikerjakan = array();
20
21              $simpulYangSudahDikerjakan_bawah = array();
22
23              $nilaiSimpulYgDitandai                = 0;
24              $nilaiSimpulFixYgDitandai     = 0;
25
26
27              for($perulangan  =  0;  $perulangan  <  1;
     $perulangan++)
28              {
29                      $perbandinganSemuaBobot = array();
```

```
30
31                      if(!in_array($simpul_maju,
    $simpulYangDikerjakan)){
32
       array_push($simpulYangDikerjakan, $simpul_maju);
33                      }
34
35                      for($perulanganSimpul      =      0;
    $perulanganSimpul    <    count($simpulYangDikerjakan);
    $perulanganSimpul++)
36                      {
37                              $jumlah_baris               =
    count($graph[  $simpulYangDikerjakan[$perulanganSimpul
    ] ]);
38
39                              $bobot_baris = array();
40
41                              $baris_belum_dikerjakan = 0;
42
43                              for($start_baris     =     0;
    $start_baris < $jumlah_baris; $start_baris++)
44                              {
45                                      $ruas_dan_bobot     =
    $graph[  $simpulYangDikerjakan[$perulanganSimpul]  ][$s
    tart_baris];//pasti berurutan
46                                      $explode = explode('-
    >', $ruas_dan_bobot);
47
48                                      if(count($explode) ==
    2)
49                                      {


50
       $baris_belum_dikerjakan += 1;
51
52
       if(!empty($simpulYangSudahDikerjakan_bawah))
53                                                      {
54
       if(in_array($simpulYangDikerjakan[$perulanganSimpu
    l], $simpulYangSudahDikerjakan_bawah)){
55
    $nilaiSimpulYgDitandai = 0;
56
       }else{
57
    $nilaiSimpulYgDitandai = $nilaiSimpulFixYgDitandai;
58                                                              }
59                                              }
```

```
60
      array_push($bobot_baris,
    ($explode[1]+$nilaiSimpulYgDitandai)); // (bobot+0) or
    (bobot+232)
61
62
      $graph[ $simpulYangDikerjakan[$perulanganSimpul] ]
    [$start_baris] = $explode[0] . "->" . $explode[1] .
    $nilaiSimpulYgDitandai;
63                                    }
64                              }
65
66                              if($baris_belum_dikerjakan >
    0)
67                              {
68                                    for($index_bobot = 0;
    $index_bobot < count($bobot_baris); $index_bobot++){
69
    if($bobot_baris[$index_bobot] <= $bobot_baris[0]){
70
    $bobot_baris[0] = $bobot_baris[$index_bobot];
71                                       }
72                                    }
73
74
      array_push($perbandinganSemuaBobot,
    $bobot_baris[0]);
75                              }
76                              else{
77
78                              }
79
80
      if(!in_array($simpulYangDikerjakan[$perulanganSimp
    ul], $simpulYangSudahDikerjakan_bawah)){
81
      array_push(       $simpulYangSudahDikerjakan_bawah,
    $simpulYangDikerjakan[$perulanganSimpul] );
82                              }
83                    }
84
85                    for($min_indexAntarBobotYgDitandai =
    0;          $min_indexAntarBobotYgDitandai          <
    count($perbandinganSemuaBobot);
    $min_indexAntarBobotYgDitandai++)
86                    {
87
      if($perbandinganSemuaBobot[$min_indexAntarBobotYgD
    itandai] <= $perbandinganSemuaBobot[0]){
```

```
88
       $perbandinganSemuaBobot[0]                              =
    $perbandinganSemuaBobot[$min_indexAntarBobotYgDitanda
    i];
89                              }
90                      }
91
92                      $indexAwalAsli
                = 0;
93                      $baris_belum_dikerjakan1        = 0;
94                      $dapat_indexAsliBobot           = 0;
95                      $simpul_lama
       = 0;
96
97                      foreach($simpulYangDikerjakan     as
    $idx => $v)
98                      {
99                              $length_baris                =
    $graph[$simpulYangDikerjakan[$idx]];
100
101                             for($baris1  =  0;  $baris1  <
    $length_baris; $baris1++)
102                             {
103
       if( isset($graph[  $simpulYangDikerjakan[$indexAwal
    Asli] ][$baris1]) )
104                                     {
105
       $bobot_baris_dan_ruas1                                  =
    $graph[  $simpulYangDikerjakan[$indexAwalAsli]  ][$bari
    s1];
106                                             $explode1    =
    array();
107                                             $explode1    =
    explode('->', $bobot_baris_dan_ruas1);
108
       if(count($explode1) == 2)
109                                             {
110
       if($perbandinganSemuaBobot[0] == $explode1[1])
111                                                     {
112
       $dapat_indexAsliBobot = $baris1;
113
       $simpul_lama                                            =
    $simpulYangDikerjakan[$indexAwalAsli];
114
       $simpul_maju = $explode1[0];
```

```
115
        $baris_belum_dikerjakan1 += 1;
116                                                     }
117                                                 }
118                                             }
119                                             else{
120                                                     break;
121                                             }
122                                     }
123
124                                     $indexAwalAsli++;
125                             }
126
127                             if($baris_belum_dikerjakan1 > 0){
128
        $graph[$simpul_lama][$dapat_indexAsliBobot]       =
    $graph[$simpul_lama][$dapat_indexAsliBobot] . "->y";
129
130                                     for($min_kolom      =      0;
    $min_kolom < $jml_simpul; $min_kolom++)
131                                     {
132                                             $length_baris1      =
    count($graph[$min_kolom]);
133
134                                             for($min_baris  =  0;
    $min_baris < $length_baris1; $min_baris++)
135                                             {
136
        if(isset($graph[$min_kolom][$min_baris]))
137                                                 {
138
        $ruasYgAkanDihapus                                  =
    $graph[$min_kolom][$min_baris];
139
        $explode3 = explode('->', $ruasYgAkanDihapus);
140
        if(count($explode3) == 2){
141
        if($explode3[0] == $simpul_maju){
142
                $graph[$min_kolom][$min_baris]              =
    $graph[$min_kolom][$min_baris]+"->t";
143
        }
144                                                     }
145                                                 }
146                                             }
147                                     }
148                             }
```

```
149
150
        if(!isset($perbandinganSemuaBobot[0]))
151                             return
     json_encode(['status'=>'error',
     'error'=>'alur_graph_anda_salah', 'teks'=>'Alur graph
     Anda Salah', 'content'=>'']);
152
153                     $nilaiSimpulFixYgDitandai          =
     $perbandinganSemuaBobot[0];
154
155                     if($simpul_maju != $simpul_tujuan){
156                       --$perulangan;
157                     }
158                     else{
159                             break;
160                     }
161             }
162
163             $gabungSimpulPilihan = array();
164             for($h = 0; $h < $jml_simpul; $h++)
165             {
166                     $length_baris2 = count($graph[$h]);
167
168                     for($n  =  0;  $n  <  $length_baris2;
     $n++)
169                     {
170                             if(isset($graph[$h][$n]))
171                             {
172                                     $str_graph          =
     $graph[$h][$n];
173
        if(    substr($str_graph,    (strlen($str_graph)-1),
     strlen($str_graph)) == "y" ){
174                                                 $explode4    =
     explode('->', $graph[$h][$n]);
175                                                 $simpulGabung
     = $h . "-" . $explode4[0];
176
177
        array_push($gabungSimpulPilihan, $simpulGabung);
178                                     }
179                             }
180                     }
181             }
182
183             $simpulFix_finish = array();
184
```

```
185             array_push($simpulFix_finish,
    $simpul_tujuan);
186
187             $simpul_explode = $simpul_tujuan;
188             for($v = 0; $v < 1; $v++)
189             {
190                     for($w      =      0;      $w      <
    count($gabungSimpulPilihan); $w++)
191                     {
192                             $explode_simpul             =
    $gabungSimpulPilihan[$w];
193                             $explode5   =   explode('-',
    $explode_simpul);
194                             if($simpul_explode         ==
    $explode5[1])
195                             {
196
       array_push($simpulFix_finish, $explode5[0]);
197                                     $simpul_explode     =
    $explode5[0];
198                             }
199                             if($simpul_explode         ==
    $simpul_awal){
200                                     break;
201                             }
202                     }
203
204                     if($simpul_awal != $simpul_explode){
205                             --$v;
206                     }else{
207                             break;
208                     }
209             }
210             $simpulFix_finish_reverse                   =
    array_reverse($simpulFix_finish);
211
212             $jalur_terpendek = "";
213             for($x        =        0;        $x        <
    count($simpulFix_finish_reverse); $x++)
214             {
215                     if($x                              ==
    (count($simpulFix_finish_reverse)-1))
216                     {
217                             $jalur_terpendek           .=
    $simpulFix_finish_reverse[$x];
218                     }else{
219                             $jalur_terpendek           .=
    $simpulFix_finish_reverse[$x] . "->";
220                     }
```

```
221             }
222
223             $json['status']        = 'success';
224             $json['success']       =
    'generate_jalur_terpendek';
225             $json['teks']          =   'Jalur   berhasil
    dibuat';
226             $json['content']       = $jalur_terpendek;
227
228             return json_encode($json);
229
230     }
231 }
232 ?>
```

Kode Sumber A.1 Implementasi Dijsktra

```
1   <?php
2   include "Koneksi.php";
3   include "DistanceTo.php";


4   include "Get_koordinat_awal_akhir.php";
5   include "GraphToArray.php";
6   include "Tambah_simpul.php";
7   include "Dijkstra.php";
8   include "Angkot.php";


9   class Main extends GraphToArray
10  {
11  public $koneksi;
12  public $graph;
13  public $id_buang;


14  public $maxRow0;
15  public $maxRow1;


16  public $old_simpul_awal;
17  public $old_simpul_akhir;
18  public $simpul_awal;
19  public $simpul_akhir;


20  public function __construct()
21  {
22  $koneksi = new Koneksi();
```

```
23  $this->koneksi = $koneksi->connect();


24  $k = $koneksi->connect();
25  mysqli_query($k, "DELETE FROM graph where temp =
    'Y'");


26  $this->maxRowDB();


27  $graph                    = new GraphToArray();
28  $graphArray       = $graph->graphArray();
29  $this->graph      = $graphArray;


30  $this->id_buang = 0;


31  }


32  public function core($lat0, $lng0, $lat1, $lng1)
33  {
34  if($lat0<-8&&$lat0>-9&&$lat1<-8&&$lat1>-
    9&&$lng0<115.8&&$lng0>114.4&&$lng1<115.8&&$lng1>114.4
    ){
35  $this->getSimpulAwalAkhirJalur($lat0, $lng0, 'awal',
    /*tambahan-->*/ $this->id_buang);
36  $this->getSimpulAwalAkhirJalur($lat1, $lng1, 'akhir',
    /*tambahan-->*/ $this->id_buang);


37  $dijkstra = new Dijkstra();


38  $json             = $dijkstra->jalurTerpendek($this-
    >graph, $this->simpul_awal, $this->simpul_akhir);
39  $decode    = json_decode($json, true);
40  $status           = $decode['status'];
41  $content  = $decode['content'];


42  if($status == 'error'){
43  $jsonPolyline =
    json_encode(['jalur_shortest_path'=>[],
    'error'=>$decode]);
44  return $jsonPolyline;
```

```
45  }
46  else{
47  $jsonPolyline = $this->drawRoute($content);
48  return $jsonPolyline;
49  }
50  }
51  else{


52  $jsonPolyline =
    ['jalur_shortest_path'=>[],'error'=>['status'=>'error
    ','error'=>'anda_berada_di_luar_bali']];
53  return json_encode($jsonPolyline);
54  }
55  }


56  public function getSimpulAwalAkhirJalur($lat, $lng,
    $kerjain, $id_buang)
57  {
58  $get              = new Get_koordinat_awal_akhir();
59  $jsonPosisi = $get->Get_simpul($lat, $lng,
    $id_buang);


60  $j = json_decode($jsonPosisi, true);
61  $status                   = $j['status'];
62  $this->id_buang           = $j['row_id'];
63  $node_simpul_awal0        = $j['node_simpul_awal0'];
64  $node_simpul_awal1        = $j['node_simpul_awal1'];
65  $index_coordinate         =
    $j['index_coordinate_json'];


66  if( $status == 'tidak_tambah_simpul' )
67  {
68  ($index_coordinate == 0) ? $fix_simpul_awal =
    $node_simpul_awal0 :$fix_simpul_awal =
    $node_simpul_awal1;


69  if($kerjain == "awal"){


70  $this->old_simpul_awal    = $node_simpul_awal0 . "-" .
    $node_simpul_awal1;
71  $this->simpul_awal               = $fix_simpul_awal;
```

```
72   }

73   else{

74   $this->old_simpul_akhir = $node_simpul_awal0 . "-" .
     $node_simpul_awal1;
75   $this->simpul_akhir      = $fix_simpul_awal;
76   }
77   }
78   else if( $status == 'tambah_simpul_double' )
79   {
80   if($kerjain == "awal"){

81   $tb = new Tambah_simpul();
82   $jadi_json = $tb->dobelSimpul($node_simpul_awal0,
     $node_simpul_awal1, $index_coordinate, $this->graph);

83   $d = json_decode($jadi_json, true);

84   $this->old_simpul_awal   = $d['simpul_lama'];
85   $this->simpul_awal               = $d['simpul_baru'];
86   $this->graph                     =
     json_decode($d['graph'], true);

87   }else{

88   $tb = new Tambah_simpul();
89   $jadi_json = $tb->dobelSimpul($node_simpul_awal0,
     $node_simpul_awal1, $index_coordinate, $this->graph);

90   $d = json_decode($jadi_json, true);

91   $this->old_simpul_akhir = $d['simpul_lama'];
92   $this->simpul_akhir      = $d['simpul_baru'];
93   $this->graph                     =
     json_decode($d['graph'], true);

94   }
```

```
95   }
96   else if( $status == 'tambah_simpul_single' )
97   {
98   if($kerjain == "awal"){


99   $tb = new Tambah_simpul();
100  $jadi_json = $tb->singleSimpul($node_simpul_awal0,
     $node_simpul_awal1, $index_coordinate, $this->graph);


101  $d = json_decode($jadi_json, true);


102  $this->old_simpul_awal   = $d['simpul_lama'];
103  $this->simpul_awal               = $d['simpul_baru'];
104  $this->graph                     =
     json_decode($d['graph'], true);


105  }else{


106  $tb = new Tambah_simpul();
107  $jadi_json = $tb->singleSimpul($node_simpul_awal0,
     $node_simpul_awal1, $index_coordinate, $this->graph);


108  $d = json_decode($jadi_json, true);


109  $this->old_simpul_akhir = $d['simpul_lama'];
110  $this->simpul_akhir      = $d['simpul_baru'];
111  $this->graph                     =
     json_decode($d['graph'], true);
112  }
113  }
114  }


115  public function drawRoute($shortest_path)
116  {
117  $exp_shortest_path = explode("->", $shortest_path);
118  $start = 0;
119  $jarak = 0;
120  $semua_latlng = array();
```

```
121  for($i = 0; $i < (count($exp_shortest_path)-1);
     $i++){

122  $select = "SELECT * FROM graph where simpul_awal =" .
     $exp_shortest_path[$start] . " and simpul_tujuan =" .
     $exp_shortest_path[(++$start)];
123  $query  = mysqli_query($this->koneksi, $select);

124  $fetch    = mysqli_fetch_array($query, MYSQLI_ASSOC);

125  $json = json_decode($fetch['jalur'], true);
126  $koordinat = $json['coordinates'];
127  $jarak = $jarak+$fetch['bobot'];
128  for($w = 0; $w < count($koordinat); $w++){

129  $latlngs = $koordinat[$w];
130  $lats = $latlngs[0];
131  $lngs = $latlngs[1];

132  $lat_lng['lat'] = $lats;
133  $lat_lng['lng'] = $lngs;

134  array_push($semua_latlng, $lat_lng);
135  }
136  }

137  $a = new Angkot();
138  $angkot_array = $a-
     >angkot_shortest_path($exp_shortest_path, $this-
     >old_simpul_awal, $this->old_simpul_akhir,$this-
     >maxRow0, $this->maxRow1);

139  $return_json = ['jalur_shortest_path'=>$semua_latlng,
     'jarak'=>$jarak, 'angkot'=>$angkot_array];
140  return $return_json;
141  }

142  public function maxRowDB(){
```

```
143  $select = "SELECT max(CONVERT(simpul_awal, SIGNED
     INTEGER)) as max_sa, max(CONVERT(simpul_tujuan,
     SIGNED INTEGER)) as max_st FROM graph";
144  $query  = mysqli_query($this->koneksi, $select);
145  $fetch  = mysqli_fetch_array($query, MYSQLI_ASSOC);


146  $max_simpul_db                  = 0;
147  $max_simpulAwal_db              = $fetch['max_sa'];
148  $max_simpulTujuan_db     = $fetch['max_st'];


149  if( $max_simpulAwal_db >= $max_simpulTujuan_db ){
150  $max_simpul_db = $max_simpulAwal_db;
151  }else{
152  $max_simpul_db = $max_simpulTujuan_db;
153  }


154  // return
155  $this->maxRow0 = ($max_simpul_db+1);
156  $this->maxRow1 = ($max_simpul_db+2);
157  }


158  public function geocode($address){


159  $address = urlencode($address);


160  $url =
     "https://maps.google.com/maps/api/geocode/json?addres
     s={$address}&key=AIzaSyAjE6SameEz5tLRzJZA83zc9Y87-
     6RixqM";


161  $resp_json = file_get_contents($url);


162  $resp = json_decode($resp_json, true);


163  if($resp['status']=='OK'){


164  $lati =
     $resp['results'][0]['geometry']['location']['lat'];
```

```
165  $longi =
     $resp['results'][0]['geometry']['location']['lng'];


166  if($lati && $longi){


167  $data_arr = array();


168  array_push(
169  $data_arr,
170  $lati,
171  $longi
172  );


173  return $data_arr;


174  }
175  else{
176  return false;
177  }


178  }
179  else{
180  return false;
181  }
182  }
183  }
184  if(isset($_GET['koord_user'],
     $_GET['koord_destination'])){


185  $koord_user              = $_GET['koord_user'];
186  $koord_destination       =
     $_GET['koord_destination'];


187  $a = new Main();
188  $arrayorigin = $a->geocode($koord_user);
189  $arraydestination= $a->geocode($koord_destination);
190  $shortest_path = $a->core($arrayorigin[0],
     $arrayorigin[1], $arraydestination[0],
     $arraydestination[1]);
```

```
191  $shortest_path['dataawal'] =
     ['origin'=>$arrayorigin,'destination'=>$arraydestinat
     ion];
192  echo json_encode($shortest_path);


193  }


194  ?>
```

Kode Sumber A.2 Implementasi Layanan Pencarian Rute Terpendek

```
1.   <?php
2.   include "Koneksi.php";


3.   class Rute {
4.   public $koneksi;
5.   public $graph;




6.   public function __construct()
7.   {
8.   $koneksi = new Koneksi();
9.   $this->koneksi = $koneksi->connect();
10.  }


11.  public function findroute($trayek)
12.  {
13.  $select = "SELECT `simpul` FROM `angkutan_umum` WHERE
     `id` = ".$trayek;
14.  $query  = mysqli_query($this->koneksi, $select);


15.  $fetch    = mysqli_fetch_array($query, MYSQLI_ASSOC);
16.  return($fetch['simpul']);
17.  }


18.  public function drawRoute($shortest_path, $trayek)
19.  {
```

```
20. $shortest_path = ltrim($shortest_path,",");
21. $shortest_path = rtrim($shortest_path,",");
22. $exp_shortest_path = explode(",", $shortest_path);
23. $node=array();
24. $semua_latlng = array();
25. $koordinattengah= array();


26. for($i = 0; $i < (count($exp_shortest_path)); $i++){


27. $node=explode("-",$exp_shortest_path[$i]);
28. if ( ! isset($node[1])) {
29. $node[1] = null;
30. }


31. $select = "SELECT `jalur` FROM `graph` WHERE
    `simpul_awal` =" . $node[0] . " AND `simpul_tujuan`
    =" . $node[1];
32. $query  = mysqli_query($this->koneksi, $select);
33. if (!$query) {
34. printf("Error: %s\n", mysqli_error($this->koneksi));
35. exit();
36. }
37. $fetch    = mysqli_fetch_array($query, MYSQLI_ASSOC);
38. $json = json_decode($fetch['jalur'], true);
39. $koordinat = $json['coordinates'];


40. for($w = 0; $w < count($koordinat); $w++){


41. $latlngs = $koordinat[$w];
42. $lats = $latlngs[0];
43. $lngs = $latlngs[1];


44. $lat_lng['lat'] = $lats;
45. $lat_lng['lng'] = $lngs;


46. array_push($semua_latlng, $lat_lng);
47. }
48. $koordinattemp=
    $koordinat[floor(count($koordinat)/2)];
49. $latt = $koordinattemp[0];
50. $lngt = $koordinattemp[1];
```

```
51.  $latlngt['lat'] = $latt;
52.  $latlngt['lng'] = $lngt;
53.  array_push ($koordinattengah, $latlngt);
54.  }
55.  $return_json = ['jalur'=>$semua_latlng,
     'koordinat'=>$koordinattengah,
     'no_trayek'=>$trayek+1];
56.  return json_encode($return_json);
57.  }
58.  }


59.  if(isset($_GET['no_trayek'])){


60.  $no_trayek        = $_GET['no_trayek'];


61.  $a = new Rute();


62.  $path = $a->findroute($no_trayek);
63.  $showpath = $a->drawRoute($path, $no_trayek);
64.  echo $showpath;
65.  }
66.  ?>
```

Kode Sumber A.3 Implementasi Layanan Rute Trayek Pengumpan

```
1.   <?php
2.   include "Koneksi.php";


3.   class Halte {
4.   public $koneksi;
5.   public $graph;


6.   public function __construct()
7.   {
8.   $koneksi = new Koneksi();
9.   $this->koneksi = $koneksi->connect();
10.  }


11.  public function showhalte()
12.  {
```

```
13. $select = "SELECT * FROM `halte_bus`";
14. $query  = mysqli_query($this->koneksi, $select);


15. while($row = $query->fetch_assoc())
16. {
17. $fetch[] = $row;
18. }


19. $return_array = [];
20. for($j=0;$j<count($fetch);$j++)
21. {
22. $halte = explode("-",$fetch[$j]['next_halte']);
23. $haltearr = [];
24. for ($i=1; $i<count($halte)-1; $i++)
25. {
26. $selectx = "SELECT `nama_halte` FROM `halte_bus`
    WHERE id = ". $halte[$i];
27. $queryx  = mysqli_query($this->koneksi, $selectx);
28. $fetchx   = mysqli_fetch_array($queryx,
    MYSQLI_ASSOC);
29. array_push($haltearr, $fetchx['nama_halte']);
30. }
31. $haltnext = implode (",",$haltearr);


32. $temp = $fetch[$j]['id'];
33. $selectx = "SELECT `nama_halte` FROM `halte_bus`
    WHERE `next_halte` LIKE '%-" . $temp ."-%'";
34. $queryx  = mysqli_query($this->koneksi, $selectx);
35. while($row = $queryx->fetch_assoc())
36. {
37. $fetchx[] = $row;
38. }


39. $haltearr = [];
40. for ($i=0; $i<count($fetchx)-1; $i++)
41. {]
42. array_push($haltearr, $fetchx[$i]['nama_halte']);
43. }
44. $haltprev = implode (",",$haltearr);


45. $gabung_array     =
    ['koordinat'=>$fetch[$j]['koordinat'],
```

```
    'id'=>$fetch[$j]['id'],
    'nama'=>$fetch[$j]['nama_halte'] ,
    'haltenext'=>$haltnext, 'halteprev'=>$haltprev,
    'koridor'=>$fetch[$j]['koridor']];
46. array_push($return_array, $gabung_array);
47. }
48. return json_encode($return_array);
49. }
50. }


51. {
52. $a = new Halte();
53. $showhalte = $a->showhalte();
54. echo $showhalte;
55. }
56. ?>
```

Kode Sumber A.4 Implementasi Layanan Lokasi Halte Bus

```
1.  <?php
2.  //# Call DB in GraphToArray.php
3.  include "Koneksi.php";


4.  class Ruteb {
5.  public $koneksi;
6.  public $graph;


7.  public function __construct()
8.  {
9.  $koneksi = new Koneksi();
10. $this->koneksi = $koneksi->connect();
11. }


12. public function drawRoute()
13. {
14. $select = "SELECT * FROM `rute_bus`";
15. $query  = mysqli_query($this->koneksi, $select);
16. $returnarray = array();


17. while($row = $query->fetch_assoc())
18. {
19. $fetch[] = $row;
```

```
20.  }

21.  for($i = 0; $i < (count($fetch)); $i++){
22.  $semua_latlng = array();

23.  $json = json_decode($fetch[$i]['jalur'], true);
24.  $koordinat = $json['coordinates'];

25.  for($w = 0; $w < count($koordinat); $w++){

26.  $latlngs = $koordinat[$w];
27.  $lngs = $latlngs[0];
28.  $lats = $latlngs[1];

29.  $lat_lng['lat'] = $lats;
30.  $lat_lng['lng'] = $lngs;

31.  array_push($semua_latlng, $lat_lng);
32.  }
33.  $nama = "jalur"+$i;
34.  $gabungarray = (object) array($nama=>$semua_latlng);
35.  array_push($returnarray, $gabungarray);
36.  }
37.  return json_encode($returnarray);
38.  }
39.  }

40.  {
41.  $a = new Ruteb();
42.  $showpath = $a->drawRoute();
43.  echo $showpath;
44.  }
45.  ?>
```

Kode Sumber A.5 Implementasi Layanan Rute Bus Trans Sarbagita

B. Kuisioner

Form Kuisioner

Nama :
Umur :
Pekerjaan :

*centang salah satu

Pernyataan	Jawaban			
	Sangat Tidak Setuju	Tidak Setuju	Setuju	Sangat Setuju
Aplikasi memudahkan pengguna dalam mendapatkan informasi mengenai angkutan Sarbagita.				
Aplikasi memproses permintaan pengguna secara cepat dan tepat.				
Aplikasi memberikan hasil rute trayek pengumpan secara tepat.				
Tampilan aplikasi mudah dimengerti.				

Gambar B.1 Form Kuisioner

Tabel B.1 Hasil Kuisioner Pengguna

Nama	**Pekerjaan**	Umur	Pernyataan			
			1	2	3	4
Agus Adi Wirawan	Pegawai	22	3	4	4	4
Dwika Setiawan	Mahasiswa	22	3	2	3	3

Daniel Henry	Mahasiswa	22	2	4	2	4
Adhi Purwanto	Mahasiswa	24	2	4	1	3
Norberta Yekti Setya Nastiti	Pegawai	23	3	3	3	3
Ni Nyoman Trisna Juliandari	Pegawai	25	4	4	4	4
Putu Adhi Purwanto	Swasta	22	2	3	3	4
Putu Agus Antara Adiputra	Mahasiswa	22	3	3	2	4
Made Diksi Narendra	Pengangguran	22	3	4	2	3
I Putu Adi Wiranata	Mahasiswa	22	4	3	3	3
Angga santosa	Mahasiswa	23	3	3	2	2
gusti ayu indah	Mahasiswa	21	4	4	3	3
Ni Wayan Sutraeni R	Swasta	22	4	4	4	4
Putu Dwipa Krisna Devi	Mahasiswa	22	3	3	2	4
Indra K Raj Suweda	Mahasiswa	23	2	3	2	2
Anin	Pegawai	22	4	4	4	4
Ngurah Desta	Pegawai	23	4	4	4	3
harindra pasimpangan	Mahasiswa	23	3	4	4	4
Gekmas	Swasta	23	4	4	4	3
Nyoman Ade Satwika	Mahasiswa	22	3	3	2	2
Aldi Pradana	Mahasiswa	21	2	3	1	2

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

I Made Aditya Pradnyadipa Mustika, lahir di Kota Denpasar, Bali, 28 September 1994. Lahir sebagai anak kedua dari tiga bersaudara dari pasangan I Made Adhy Mustika dan Luh Putu Sri Armini. Menempuh pendidikan di SDP Negeri Tulangampiang, SMP Negeri 1 Denpasar, SMA Negeri 4 Denpasar, dan terakhir melanjutkan tingkat Strata 1 di jurusan Teknik Informatika – Institut Teknologi Sepuluh Nopember Surabaya.

Selama Masa Perkuliahan, penulis aktif dalam kegiatan organisasi kampus sebagai anggota Himpunan Mahasiswa Teknik Computer-Informatika, staf departemen pengembangan sumber daya mahasiswa dan anggota Tim Pembina Kerohanian Hindu, staf departemen komunikasi dan informasi pada tahun kedua, serta menjadi kepala departemen komunikasi dan informasi Tim Pembina Kerohanian Hindu pada tahun ketiga. Penulis aktif menjadi panitia dalam berbagai kegiatan di tingkat jurusan maupun fakultas.

Ketertarikan penulis dibidang informatika berada pada bidang sistem teknologi informasi, dan manajemen informasi. Selain ketertarikan dalam bidang informatika, penulis juga memiliki ketertarikan dalam bidang seni, khusunya seni grafis dan seni peran yang sering dituangkan sebagai hobi dan pekerjaan sampingan.

Penulis terbuka dengan ajakan diskusi, baik mengenai Tugas Akhir ini maupun hal lain dan bisa dihubungi melalui surat elektronik di alamat adtmustika@gmail.com.