PENERAPAN ALGORITMA FLOYD-WARSHALL PADA APLIKASI PENCARIAN SPBU DENGAN RUTE TERPENDEK

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Pada
Jurusan Teknik Informatika

Oleh :

ROSLINA MURSALIN
NIM : 10751000047

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU
PEKANBARU

2013

IMPLEMENTATION OF FLOYD-WARSHALL ALGORITHM ON APPLICATION WITH SEARCH OF SPBU SHORTEST ROUTE

ROSLINA MURSALIN

10751000047

Information Engineering Department

Faculty of Sciences and Technology

State Islamic University of Sultan Syarif Kasim Riau

ABSTRACT

Floyd-Warshall Algorithm is an algorithm that can use to searching for determine the shortest route. These algorithms research troubleshooting by viewing the final solution to be obtained as an inter-related decisions. Research of SPBU is very important means of public existence at the present time actually is an object of this study. SPBU are available to finding the shortest route of Pekanbaru development and the extent of the path of travel. With the Floyd-Warshall Algorithm and technology development at the present time, these can be applied to smartphones with Android based operating systems and can be a SPBU search applications by applying the Floyd-Warshall algorithm to determine the shortest route. The aim of SPBU search application is to facilitate users to the SPBU intended to shortest route. In this study, a database used only database main roads, arterial and SPBU location. In conducting the search process, the time required is relatively longer, because the database retrieval and submission of its response in the form of client-server, but it can provide the optimum results.

Keywords: A*ndroid, client, Floyd-Warshall algorithm, server, shortest route, SPBU.*

PENERAPAN ALGORITMA FLOYD-WARSHALL PADA APLIKASI PENCARIAN SPBU DENGAN RUTE TERPENDEK

ROSLINA MURSALIN

10751000047

Jurusan Teknik Informatika

Fakultas Sains dan Teknologi

Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Algoritma *Floyd-Warshall* merupakan suatu algoritma yang melakukan pencarian untuk menentukan rute terpendek. Algoritma ini melakukan pemecahan masalah dengan memandang solusi akhir yang akan diperoleh sebagai suatu keputusan yang saling terkait. Objek pada penelitian ini adalah sarana umum yang sangat penting keberadaannya pada masa sekarang, yaitu SPBU. Berkembangnya kota Pekanbaru dan luasnya jalur perjalanan, menjadi suatu kendala untuk menemukan rute terpendek dari beberapa lokasi SPBU yang tersedia. Dengan algoritma *Floyd-Warshall* dan berkembangnya teknologi pada masa sekarang, kedua hal tersebut dapat diterapkan pada perangkat pintar (*smartphone*) dengan sistem operasi berbasis Android dan dapat menjadi suatu aplikasi pencarian SPBU dengan menerapkan algoritma *Floyd-Warshall* untuk menentukan rute terpendek. Aplikasi pencarian SPBU ini bertujuan untuk memudahkan pengguna menuju SPBU yang dituju dengan rute terpendek. Pada penelitian ini, *database* yang digunakan hanya *database* jalan utama, arteri dan lokasi SPBU saja. Dalam melakukan proses pencarian, waktu yang dibutuhkan memang relatif lebih lama, karena pengambilan *database* dan penyampaian responnya berupa *client-server*, namun dapat memberikan hasil rute yang optimum.

Kata kunci: *Algoritma Floyd-Warshall, android, client, rute terpendek, server, SPBU.*

KATA PENGANTAR

Alhamdulillaahi Robbil'alamin, penulis ucapkan syukur yang setinggi-tinggi ke-hadirat Allah SWT, karena atas segala limpahan rahmat dan karunianya yang diberikan sehingga penulis dapat menyelesaikan penelitian sekaligus penulisan laporan tugas akhir ini. *Allahumma sholli'ala Muhammad wa'ala ali sayyidina Muhammad*, yang tidak lupa penulis haturkan juga untuk junjungan alam, kekasih Allah, Rasul Allah, dan tauladan kita yakni Nabi Muhammad SAW.

Laporan tugas akhir ini merupakan salah satu prasyarat untuk memenuhi persyaratan akademis dalam rangka meraih gelar kesarjanaan di Jurusan Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sultan Syarif Kasim Riau (UIN SUSKA Riau). Selama menyelesaikan tugas akhir ini, penulis telah banyak mendapatkan bantuan, bimbingan, dan petunjuk dari banyak pihak baik secara langsung maupun tidak langsung. Untuk itu dalam kesempatan ini penulis ingin mengucapkan rasa terima kasih yang sebesar-besarnya kepada:

1. Prof. Dr. H. M. Nazir, selaku Rektor Universitas Islam Negeri Sultan Syarif Kasim Riau.
2. Dra. Yenita Morena, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau.
3. Elin Haerani, M.Kom, selaku Ketua Jurusan Teknik Informatika dan dosen pembimbing, terima kasih juga untuk ilmu-ilmunya, saran-sarannya, perbaikan-perbaikannya, dan masukan yang Ibu berikan untuk penyempurnaan laporan ini.
4. Muhammad Affandes, ST, selaku Koordinator Tugas Akhir.
5. Teddie, ST, selaku pembimbing akademis.

6. Febi Yanto, M.Kom, selaku dosen penguji 1 yang banyak membantu dan memberi masukan kepada penulis dalam penyempurnaan Laporan Tugas Akhir ini, untuk ilmu-ilmunya saya ucapkan terima kasih.
7. M. Safrizal, ST, M.Cs, selaku penguji 2 yang juga telah memberi masukan kepada penulis dalam penyempurnaan Laporan Tugas Akhir ini, untuk ilmu-ilmunya saya ucapkan terima kasih.
8. Sahabat penulis, Yonni, Rizky, Fadli, Erma, Dimas, Batri, Mayang, Freddy, terima kasih atas saran dan bantuannya serta semangat yang diberikan selama ini.
9. Teman-teman TIF D angkatan 2007, serta pihak-pihak lain yang tidak dapat penulis sebutkan satu persatu, terima kasih banyak penulis ucapkan atas segala bentuk bantuan yang telah ikhlas diberikan kepada penulis.

Akhirnya, penulis menyadari dalam penulisan laporan ini masih terdapat kekurangan. Oleh karena itu, saran dan kritik sangat penulis harapkan untuk kemajuan penulis secara pribadi. Terima kasih.

Pekanbaru, 31 Juli 2013

Penulis

DAFTAR ISI

BAB I

PENDAHULUAN

1.1. Latar Belakang

Kota Pekanbaru merupakan salah satu daerah atau wilayah yang luas dan sedang berkembang serta memiliki sangat banyak jalur perjalanan. Menurut Badan Pusat Statistik (BPS) yang disampaikan Walikota Pekanbaru, Firdaus yang dikutip dari LKBN Antara, "Penduduk kota Bertuah kini telah mencapai lebih dari satu juta jiwa."(sumber: www.haluanriaupress.com, 25 Agustus 2012). Mereka adalah penduduk yang beraktifitas di Pekanbaru, tetapi bermukim di pinggiran kota yang masuk wilayah Kampar, Siak dan Pelalawan. Hal ini menjadi bukti bahwa Pekanbaru sudah menjadi "kutub" perekonomian bagi masyarakat luar daerah. Selain banyaknya pendatang yang ingin bekerja ataupun menetap, kota Pekanbaru juga diramaikan oleh kunjungan wisatawan. Kota Pekanbaru juga menjadi salah satu faktor meningkatnya arus kendaraan bermotor sehingga jalan menjadi lebih ramai. Ketidaktahuan lokasi jalan yang ada di Pekanbaru bagi para pendatang juga menjadi suatu kendala sehingga membuat pengendara membutuhkan media informasi untuk membantu mencari lokasi layanan atau prasarana umum bagi pengendara kendaraan bermotor.

Layanan atau prasarana umum berperan penting bagi pengendara kendaraan bermotor. Salah satu prasarana umum yang disediakan adalah SPBU (Stasiun Pengisian Bahan Bakar Umum). Sekarang ini banyak SPBU yang juga menyediakan layanan tambahan. Misalnya, musholla, pompa angin, toilet, swalayan kecil dan ATM. Pengendara kendaraan bermotor yang sedang bepergian ke tempat yang tidak dikenalnya akan kesulitan dalam mencari lokasi SPBU terdekat.

Luasnya kota Pekanbaru serta banyaknya jalan raya seringkali menyulitkan pengendara untuk mencari rute optimum, baik dari segi jarak maupun biaya yang dikeluarkan untuk bepergian dari satu tempat ke tempat yang lain. Untuk dapat

memilih rute yang optimum, maka orang harus mengetahui jarak antar tempat tujuan dan juga keadaan jalan dari rute yang akan dilalui. Kemudian dipilihlah jalur terpendek dari tempat awal ke tempat tujuan. Tetapi hal ini seringkali tidak membantu karena banyaknya jalan yang ada sehingga menyebabkan banyaknya pilihan jalur yang dapat ditempuh. Kita sebagai konsumen tentu saja harus selektif dalam memilih jalur yang pendek dan efisien, tidak berliku-liku, dapat menghemat waktu, dan menghemat biaya. Untuk itu, dibutuhkan sebuah algoritma yang mampu menangani masalah pencarian rute terpendek dan efisien.

Terdapat banyak algoritma yang dapat menyelesaikan masalah pencarian rute terpendek, antara lain algoritma *Dijkstra*, *Bellman-Ford*, *A-Star* dan *Floyd-Warshall*. Akan tetapi merujuk pada jurnal atau penelitian yang dilakukan oleh Raden Aprian Diaz Novandi dimana, kesimpulan yang dihasilkan adalah: "Algoritma Floyd-Warshall yang menerapkan pemrograman dinamis lebih menjamin keberhasilan penemuan solusi optimum untuk kasus penentuan lintasan terpendek (*all-pairs shortest path*)". (Novandi, 2007)

Penelitian tentang pencarian lokasi dan jarak SPBU sudah pernah dilakukan oleh mahasiswa jurusan Teknik Informatika Universitas Gunadarma, yang merancang suatu program aplikasi GPS dan GIS untuk mencari lokasi dan jarak SPBU di Tangerang Selatan dengan peta dan *Augmented Reality Camera-View* pada perangkat bergerak berbasis *Android*, program ini tidak menggunakan metode atau algoritma apapun. Sistem tersebut hanya menyajikan informasi SPBU yang berbasis lokasi, maksudnya sistem hanya menampilkan lokasi SPBU saja, tidak menampilkan rute mana yang harus dilewati untuk mencapai lokasi SPBU. (Sutrisno, 2011).

Di antara banyaknya algoritma yang menggunakan informasi sebagai pijakannya, algoritma *Floyd-Warshall* merupakan suatu metode untuk mengambil sebuah keputusan dimana pemecahan masalah dilakukan dengan bertahap sehingga akan terbentuk solusi-solusi yang berasal dari tahap sebelumnya dan ada kemungkinan solusi tersebut lebih dari satu. Algoritma ini juga bekerja secara singkat dan efisien dibandingkan dengan algoritma *Greedy* dan *Dijkstra*. Jika dibandingkan dengan algoritma A*, konsep penyelesaian masalahnya hampir

mirip dengan konsep penyelesaian dari algoritma *Dijkstra*, sedangkan pada algoritma *Bellman-Ford*, proses penyelesaiannya tergolong lebih lama daripada algoritma *Dijkstra*.

Penelitian tugas akhir ini difokuskan untuk meneliti penerapan algoritma *Floyd-Warshall* yang diimplementasikan pada aplikasi pencarian SPBU dengan rute terpendek. Hasil penelitian ini akan memberikan rute terpendek dari lokasi SPBU dengan penerapan algoritma *Floyd-Warshall* yang lebih singkat dan efisien setelah dibandingkan dengan hasil yang didapat dari kuesioner untuk lebih mengoptimalkan hasil akhir rute terpendek yang akan dilalui.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas, yang menjadi pokok permasalahan dalam hal ini adalah "Bagaimana menerapkan algoritma *Floyd-Warshall* pada aplikasi pencarian SPBU dengan rute terpendek".

1.3 Batasan Masalah

Untuk mendapatkan hasil yang optimal dan mengetahui cakupan penulisan Tugas Akhir ini, maka akan diberikan batasan-batasan masalah dalam penulisan Tugas Akhir ini. Batasan Tugas Akhir ini adalah sebagai berikut:

1. Pencarian SPBU dengan rute terpendek disimulasikan menggunakan *software* JDK (*Java Development Kit*) dan Eclipse IDE (*Integrated Development Environment*).
2. Lingkungan dan beberapa lokasi SPBU hanya mencakup di kota Pekanbaru yang disediakan oleh PT. Pertamina.
3. Peta yang ditampilkan dalam bentuk model data vektor, yaitu format titik, garis, dan poligon.
4. Proses perhitungan rute terpendek hanya berdasarkan pada jarak yang ditempuh dari tiap titik koordinat jalan.

1.4 Tujuan Penelitian

Tujuan yang ingin dicapai penulis dari penelitian Tugas Akhir ini adalah untuk menerapkan algoritma *Floyd-Warshall* pada aplikasi pencarian SPBU dengan rute terpendek.

1.5 Sistematika Penulisan

Berikut merupakan rencana sistematika penulisan laporan Tugas Akhir yang akan dibuat:

Bab I Pendahuluan

Bab ini berisi penjelasan mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat dan sistematika penulisan dari Tugas Akhir yang dibuat.

Bab II Landasan Teori

Bab ini membahas tentang teori umum dan khusus yang berhubungan dengan Tugas Akhir ini, antara lain tentang *Problem* dan Algoritma, Optimalisasi, Metode *Graph,* Algoritma *Floyd-Warshall*, SPBU (Stasiun Pengisian Bahan Bakar), Perangkat Lunak yang Digunakan, GIS (*Geographical Information System*) dan GPS (*Global Positioning System*).

Bab III Metodologi Penelitian

Bab ini membahas langkah-langkah yang dilaksanakan dalam proses penelitian, yaitu pengamatan pendahuluan dan pengumpulan data, tahapan identifikasi masalah, perumusan masalah, analisa dan implementasi algoritma.

Bab IV Analisa dan Perancangan

Bab ini berisi pembahasan mengenai kebutuhan aplikasi yang terdiri dari; gambaran umum aplikasi, deskripsi kebutuhan, fungsi aplikasi, model

aplikasi, deskripsi pengguna, perancangan aplikasi serta rancangan struktur menu aplikasi dan perancangan antarmuka pengguna aplikasi.

Bab V Implementasi dan Pengujian

Bab ini berisi penjelasan yang terdiri dari lingkungan implementasi, hasil implementasi, pengujian sistem dengan *Blackbox* serta kesimpulan pengujian.

Bab VI Penutup

Bab ini berisi kesimpulan dan saran yang dihasilkan dari penelitian Tugas Akhir mengenai Penerapan Algoritma *Floyd-Warshall* pada Aplikasi Pencarian SPBU dengan Rute Terpendek.

BAB II

LANDASAN TEORI

2.1 *Problem* dan Algoritma

2.1.1 Definisi *Problem*

Problem (permasalahan) adalah sebuah kendala atau hambatan dimana membuat suatu tujuan sulit untuk dicapai, hal ini berhubungan dengan situasi, kondisi, atau persoalan-persoalan yang belum terselesaikan. Sebuah *problem* timbul dikarenakan seorang individu menjadi sadar akan perbedaan yang berarti antara apakah yang sebenarnya dan apakah yang merupakan hasrat atau keinginan. Setiap *problem* membutuhkan jawaban atau solusi. Suatu *problem* dikatakan telah selesai apabila tujuan telah dicapai atau sudah tidak ada lagi masalah yang terjadi.

2.1.2 Definisi Algoritma

Algorithm (Algoritma) berasal dari kata *Algoris* dan *Ritmis*, kata-kata ini berasal dari nama seorang ahli Matematika, *Abu Ja'far Mohammed ibnu Musa al-Khwarizmi*, yang merupakan bagian dari *Royal Court, Baghdad*, dan hidup antara tahun 750 sampai 850. (Munir, 2006)

Algorithm adalah sebuah prosedur yang terstruktur dan dituliskan secara sistematis untuk menyelesaikan sebuah tugas dimana, memberikan keadaan awal (*initial state*), dan akan *terminate* di akhir (*end state*) dengan bantuan komputer. (Cormen, p.5)

Teori *Algorithm* menurut *Horowitz*:

1. *Input*, nol atau sejumlah kuantitas yang disuplai secara eksternal.
2. *Output*, paling sedikit dihasilkan suatu kuantitas.
3. *Definiteness*, setiap instruksi jelas atau tidak ambigu.

4. *Finiteness*, jika suatu instruksi algoritma akan ditelusuri, dan dalam semua kasus, algoritma berakhir dalam beberapa langkah terbatas.
5. *Effectiveness*, setiap instruksi harus harus bersifat mendasar sehingga mudah diterapkan, secara prinsip dapat dikerjakan oleh seseorang walaupun dengan menggunakan pensil dan kertas.

Gambar 2.1 berikut merupakan gambaran mengenai hubungan yang saling terkait antara masalah, algoritma dan solusi.

Gambar 2.1. Hubungan Masalah, Algoritma dan Solusi

2.2 Optimalisasi

Optimalisasi adalah suatu proses utuk mencapai hasil yang ideal atau optimal (nilai efektif yang dicapai). Menurut Hadley (1975) optimalisasi fungsi numerik yang dibuat minimal atau maksimal dari sejumlah variabel atau fungsi dengan variabel atau fungsi tersebut yang memenuhi batasan-batasan tertentu dari suatu masalah umum.

Menurut Bronson (1997) optimalisasi merupakan suatu cara untuk menentukan suatu kuantitas maksimal atau minimal yang secara spesifik disebut objektif, dan tergantung pada suatu bilangan terhingga atau variabel *input*, variabel tersebut dapat berdiri sendiri atau saling berkaitan satu sama lain melalui satu atau beberapa kendala.

Salah satu persoalan optimasi yang sering ditemui dalam kehidupan sehari-hari adalah pencarian lintasan terpendek (optimum). Persoalan ini bisa dimodelkan ke dalam suatu graf berbobot dengan nilai pada masing-masing sisi yang merepresentasikan persoalan yang akan dipecahkan.

2.3 Metode *Graph*

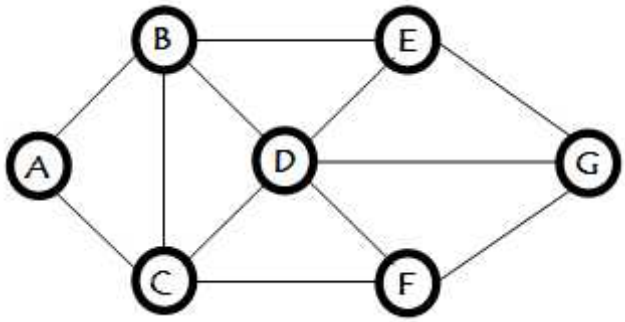
Graf adalah suatu kumpulan simpul (*nodes*) yang dihubungkan satu sama lain melalui sisi atau busur (*edges*) (Zakaria, 2006). Secara informal, suatu graf adalah himpunan benda-benda yang disebut *verteks* atau *node* yang terhubung

oleh *edge-edge*. Biasanya graf digambarkan sebagai kumpulan titik-titik (melambangkan *verteks*) yang dihubungkan oleh garis-garis (melambangkan *edge-edge*).

2.3.1 Macam-macam *Graph*

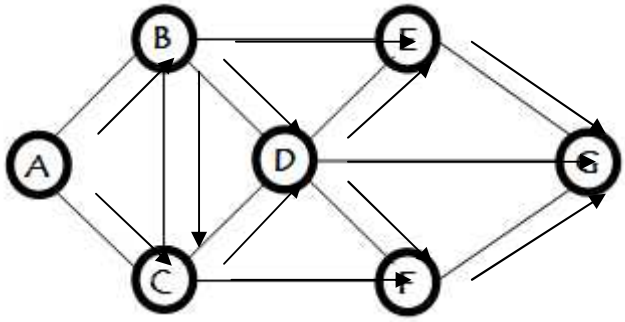
Menurut arah dan bobotnya, graf dibagi menjadi empat bagian, yaitu:

1. Graf tidak berarah dan tidak berbobot: tiap busur tidak mempunyai anak panah dan tidak berbobot. Dapat dilihat pada gambar 2.2 berikut.



Gambar 2.2. Graf tidak berarah dan tidak berbobot

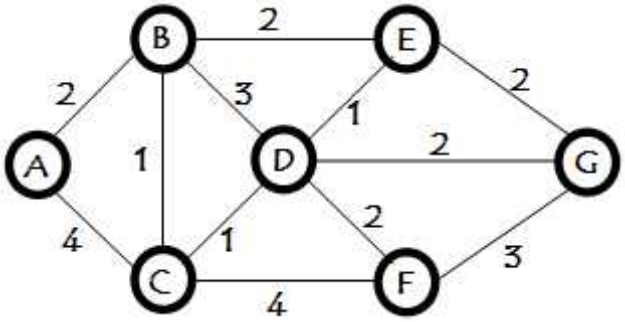
2. Graf berarah dan tidak berbobot: tiap busur mempunyai arah atau anak panah tetapi tidak memiliki bobot. Dapat dilihat pada gambar 2.3 berikut.



Gambar 2.3. Graf berarah dan tidak berbobot

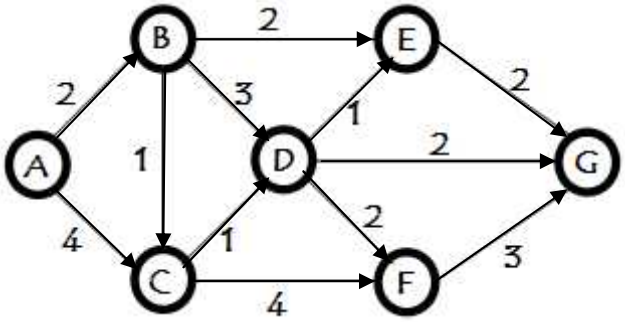
3. Graf tidak berarah dan berbobot: tiap busur tidak mempunyai anak panah tetapi mempunyai bobot. Gambar 2.4 menunjukkan graf tidak

berarah dan berbobot. Graf terdiri dari tujuh titik yaitu titik A, B, C, D, E, F, G. Titik A tidak menunjukkan arah ke titik B atau C, tetapi bobot antara titik A dan titik B telah diketahui. Begitu juga dengan titik yang lain. Dapat dilihat pada gambar 2.4 berikut.



Gambar 2.4. Graf tidak berarah dan berbobot

4. Graf berarah dan berbobot: tiap busur mempunyai anak panah dan bobot. Gambar 2.5 menunjukkan graf berarah dan berbobot yang terdiri dari tujuh titik yaitu titik A,B, C, D, E, F, G. Titik A menunjukkan arah ke titik B dan titik C, titik B menunjukkan arah ke titik D dan titik C, dan seterusnya. Bobot antar titik A dan titik B pun telah diketahui. Dapat dilihat pada gambar 2.5 berikut.



Gambar 2.5. Graf berarah dan berbobot

2.4 Algoritma Pencarian Rute Terpendek

Permasalahan pencarian rute terpendek ini telah dapat dipecahkan dengan berbagai algoritma. Beberapa algoritma populer yang memecahkan persoalan pencarian rute terpendek tersebut adalah algoritma *Dijkstra*, *Bellman-Ford*, *A-Star*

dan *Floyd-Warshall*. Berikut akan dijelaskan secara singkat mengenai algoritma tersebut.

a. Algoritma Dijkstra

Algoritma Dijkstra merupakan salah satu varian dari algoritma *Greedy*, yaitu salah satu bentuk algoritma populer dalam pemecahan persoalan yang terkait dengan masalah optimasi. Sifatnya sederhana dan lempang (*straightforward*). Sesuai dengan artinya yang secara harafiah berarti tamak atau rakus (namun tidak dalam konteks negatif), algoritma *Greedy* ini hanya memikirkan solusi terbaik yang akan diambil pada setiap langkah tanpa memikirkan konsekuensi ke depan.

Intinya algoritma *Greedy* ini berupaya membuat pilihan nilai optimum lokal pada setiap langkah dan berharap agar nilai optimum lokal ini mengarah kepada nilai optimum global.

Penggunaan strategi *Greedy* pada algoritma Dijkstra adalah pada setiap langkah, ambil sisi berbobot minimum yang menghubungkan sebuah simpul yang sudah terpilih dengan sebuah simpul lain yang belum terpilih. Lintasan dari simpul asal ke simpul yang baru haruslah merupakan lintasan yang terpendek di antara semua lintasannya ke simpul-simpul yang belum terpilih. (Novandi, 2007)

b. Algoritma *Bellman-Ford*

Algoritma *Bellman-Ford* menghitung jarak terpendek (dari satu sumber) pada sebuah digraf berbobot. Maksudnya dari satu sumber ialah bahwa ia menghitung semua jarak terpendek yang berawal dari satu titik node. Algoritma *Dijkstra* dapat lebih cepat mencari hal yang sama dengan syarat tidak ada sisi (edge) yang berbobot negatif. Maka Algoritma *Bellman-Ford* hanya digunakan jika ada sisi berbobot negatif.

Algoritma *Dijkstra* akan membuat rute yang lebih banyak dan mengambil waktu yang lebih banyak dalam mencapai rute tersingkat dalam membuat *spanning tree* karena ia tidak mendefinisikan nilai minus sebagai sebuah rute dan harus melakukan *trace-back* berulang-ulang.

Sebaliknya, *Bellman-ford* menggunakan nilai minus dalam mencapai rute tersingkat sehingga lebih sedikit melakukan *trace-back*. Tetapi algoritma *Bellman-Ford* tidak lebih baik dari algorima *Floyd-Warshall*. Dalam kasus lain, karena algoritma *Dijkstra* sudah melakukan *trace-back* dan telah membandingkan dengan nilai rute akhir dengan nilai-nilai sebelumnya maka algoritma ini sudah bisa memastikan bahwa nilai akhir merupakan rute terpendek yang diambil. Berbeda dengan *Bellman-Ford*, nilai terakhir dari rute memang benar rute yang terpendek, tapi masalahnya adalah algoritma ini tidak mengetahui mana simpul yang terakhir, sehingga ia harus memunculkan simpul yang tidak perlu untuk membentuk sebuah sirkuit dan melakukan *trace-back* kembali.

c. Algoritma *A-Star* (A*)

Algoritma A* menerapkan teknik heuristik dalam membantu penyelesaian persoalan. Heuristik adalah penilai yang memberi harga pada tiap simpul yang memandu A* mendapatkan solusi yang diinginkan. Algoritma A* bisa dikatakan mirip dengan algoritma Dijkstra, namun pada algoritma Dijkstra, nilai fungsi heuristiknya selalu 0 (nol) sehingga tidak ada fungsi yang mempermudah pencarian solusinya. (http://ragunk.wordpress.com/2010/01/05/algoritma-a-a-star)

d. Algoritma *Floyd-Warshall*

Algoritma *Floyd-Warshall* adalah salah satu varian dari pemrograman dinamis, yaitu suatu metode yang melakukan pemecahan masalah dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait. Artinya solusi-solusi tersebut dibentuk dari solusi yang berasal dari tahap sebelumnya dan ada kemungkinan solusi yang didapat lebih dari satu. (http://fahmiramadhan.wordpress.com/page/4/).

Algoritma *Floyd-Warshall* juga membandingkan semua kemungkinan lintasan pada graf untuk setiap sisi dari semua simpul. Algoritma *Floyd-Warshall* menerapkan pemrograman dinamis sehingga lebih menjamin keberhasilan penemuan solusi optimum untuk kasus penemuan lintasan terpendek (*single pair shortest path*).

Dari beberapa kelemahan yang terdapat pada algoritma sebelumnya, dapat diambil kesimpulan bahwa Algoritma *Floyd-Warshall* yang menerapkan pemrograman dinamis lebih menjamin keberhasilan penemuan solusi optimum untuk kasus penentuan lintasan terpendek.

2.4.1 Karakteristik Program Dinamis

Beberapa karakteristik yang dimiliki oleh program dinamis antara lain:

a. Persoalan dibagi atas beberapa tahap, yang setiap tahapnya hanya akan diambil satu keputusan.
b. Masing-masing tahap terdiri atas sejumlah status yang saling berhubungan dengan status tersebut. Status yang dimaksud di sini adalah berbagai kemungkinan masukan yang ada pada tahap tersebut.
c. Ketika masuk ke suatu tahap, hasil keputusan akan ditransformasi.
d. Biaya (beban) pada suatu tahap akan meningkat secara teratur seiring bertambahnya jumlah tahapan.
e. Biaya yang ada pada suatu tahap tergantung dari biaya tahapan yang telah berjalan dan biaya pada tahap itu sendiri.
f. Keputusan yang terbaik pada suatu tahap bersifat independen terhadap keputusan pada tahap sebelumnya.
g. Terdapat hubungan rekursif yang menyatakan bahwa keputusan terbaik dalam setiap status pada tahap k akan memberikan keputusan terbaik untuk setiap status pada tahap k+1.
h. Prinsip optimalisasi berlaku pada persoalan yang dimaksud.

Dalam proses penyelesaian menggunakan program dinamis, pendekatan yang dilakukan bisa jadi ada dua macam, yaitu pendekatan maju (*forward*) dan pendekatan mundur (*backward*), dan perlu untuk diketahui pula bahwa solusi yang dihasilkan dari kedua pendekatan tersebut adalah sama. Solusi dari program dinamis bisa jadi lebih dari satu macam. (Cormen, 2003).

2.4.2 Analisis Algoritma *Floyd-Warshall*

Algoritma *Floyd-Warshall* membandingkan semua kemungkinan lintasan pada graf untuk setiap sisi dari semua simpul. Menariknya, algoritma ini mampu mengerjakan proses perbandingkan ini sebanyak V3 kali (bandingkan dengan kemungkinan jumlah sisi sebanyak V2 (kuadrat jumlah simpul) pada graf, dan setiap kombinasi sisi diujikan). Hal tersebut bisa terjadi karena adanya perkiraan pengambilkan keputusan (pemilihan jalur terpendek) pada setiap tahap antara dua simpul, hingga perkiraan tersebut diketahui sebagai nilai optimal. Misalkan terdapat suatu graf G dengan simpul-simpul V yang masing-masing bernomor 1 s.d. N (sebanyak N buah). Misalkan pula terdapat suatu fungsi shortestPath(i, j, k) yang mengembalikan kemungkinan jalur terpendek dari i ke j dengan hanya memanfaatkan simpul 1 s.d. k sebagai titik perantara. Tujuan akhir penggunaan fungsi ini adalah untuk mencari jalur terpendek dari setiap simpul i ke simpul j dengan perantara simpul 1 s.d. k+1.

Ada dua kemungkinan yang terjadi:

1. Jalur terpendek yang sebenarnya hanya berasal dari simpul-simpul yang berada antara 1 hingga k.
2. Ada sebagian jalur yang berasal dari simpul-simpul i s.d. k+1, dan juga dari k+1 hingga j.

Perlu diketahui bahwa jalur terpendek dari i ke j yang hanya melewati simpul 1 s.d. k telah didefinisikan pada fungsi shortestPath(i, j, k) dan telah jelas bahwa jika ada solusi dari i s.d. k+1 hingga j, maka panjang dari solusi tadi adalah jumlah (konkatenasi) dari jalur terpendek dari i s.d. k+1 (yang melewati simpul-simpul 1 s.d. k), dan jalur terpendek dari k+1 s.d. j (juga menggunakan simpul-simpul dari 1 s.d. k).

Maka dari itu, rumus untuk fungsi shortestPath(i, j, k) bisa ditulis sebagai suatu notasi rekursif sebagai berikut:

```
Basis-0
      shortestPath(i, j, 0) = edgeCost(i, j);
Rekurens
      shortestPath(i, j, k) =
min(shortestPath(i, j, k-1),
shortestPath(i, k, k-1) +
```

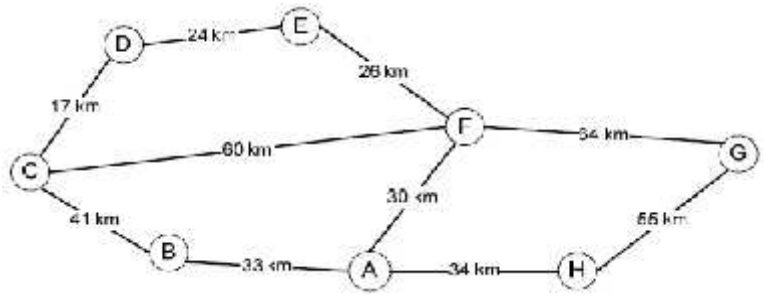
```
shortestPath(k, j, k-1));
```

Rumus ini adalah inti dari algoritma *Floyd-Warshall*. Algoritma ini bekerja dengan menghitung shortestPath(i,j,1) untuk semua pasangan (i,j), kemudian hasil tersebut akan digunakan untuk menghitung shortestPath(i,j,2) untuk semua pasangan (i,j), dan seterusnya. Proses ini akan terus berlangsung hingga k = n dan kita telah menemukan jalur terpendek untuk semua pasangan (i,j) menggunakan simpul-simpul perantara.

2.4.2.1 Contoh Kasus Pencarian Rute Terpendek

Contoh kasus pencarian rute terpendek menggunakan algoritma *Dijkstra* dan algoritma *Floyd-Warshall*:

Misalkan terdapat suatu graf berbobot yang merepresentasikan kondisi keterhubungan antarkota di suatu daerah, dengan ilustrasi pada gambar 2.6 sebagai berikut:



Gambar 2.6. Representasi keterhubungan antarkota dalam graf berbobot

Misalkan seseorang akan melakukan perjalanan dari kota A ke kota C. Orang tersebut mencoba untuk menerapkan algoritma Dijkstra dan algoritma *Floyd-Warshall* untuk mencari jalur terpendek dari kota A ke kota C.

Berikut ini adalah penelusuran jalur apabila orang tersebut menggunakan algoritma *Dijkstra* (prinsip *greedy*):

Tahap 1:

Dari kota A, orang tersebut akan memilih kota F dengan bobot minimum dari kota A (30 km).

Tahap 2:

Dari kota F, orang tersebut kemudian memilih kota E yang memiliki bobot minimum dari kota F (26 km).

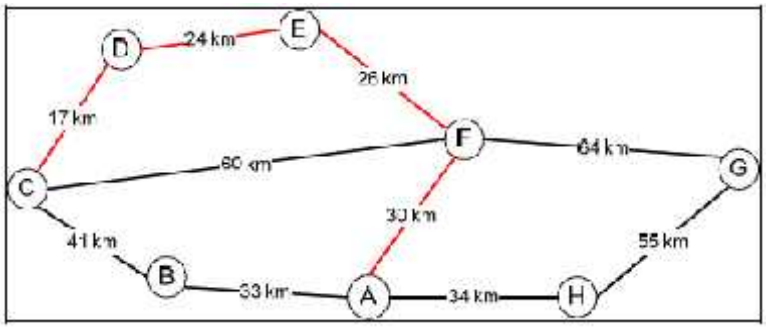
Tahap 3:

Dari kota E, orang tersebut akan melanjutkan perjalanan ke kota D (satu-satunya simpul yang terhubung).

Tahap 4:

Dari kota D, orang tersebut lalu melanjutkan perjalanan dan sampai ke kota C.

Dalam representasi graf, warna merah pada sisi graf menunjuk ke jalur terpendek menurut algoritma *Dijkstra*. Total jarak yang ditempuh oleh orang tersebut adalah = **97** km dengan jalur (A – F – E – D – C). Dapat dilihat pada gambar 2.7 berikut.



Gambar 2.7. Representasi keterhubungan antarkota setelah menerapkan algoritma *Dijkstra*

Sekarang, orang tersebut mencoba menerapkan algoritma *Floyd-Warshall* dengan pendekatan pemrograman dinamis maju (*forward*).

Basis => f_1 (s) = cx_1s

Rekurens

f_k (s) = min x_k {$cx_k s$ + f_k-1(x_k)}, k = 2, 3, 4

Tahap 1:

f_1 (s) = cx_1s

s	Solusi Optimum

	f_1 (s)	x_1
B	33	A
F	30	A
H	34	A

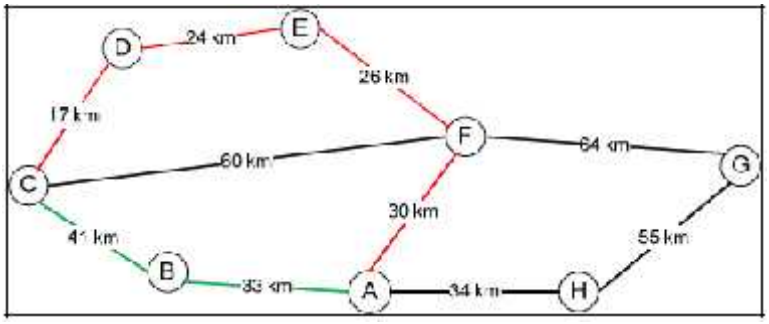
Tahap 2:

f_2 (s) = min s_2 {$cx_2s + f_1(x_2)$}

x_2 / s	f_2 (x_2, s) = $cx_2s + f_1(x_2)$			Solusi Optimum	
	B	F	H	F_2 (s)	X_2
C	**74**	90	-	**74**	B
E	-	56	-	26	F
G	-	94	89	89	H

Dari hasil pencarian jalur terpendek dari A ke C menggunakan algoritma *Floyd-Warshall* (pemrograman dinamis), ditemukan bahwa jarak terpendek dari A ke C adalah **74** km dengan jalur (A – B – C).

Berikut ini representasi graf setelah menggunakan kedua algoritma tadi. Dalam representasi graf, warna merah pada sisi graf menunjuk ke jalur terpendek menurut algoritma *Dijkstra*, sementara warna hijau menurut algoritma *Floyd-Warshall*. Dapat dilihat pada gambar 2.8 berikut.



Gambar 2.8. Representasi keterhubungan antarkota setelah menerapkan algoritma *Dijkstra* (sisi berwarna merah), dan algoritma *Floyd-Warshall* (sisi berwarna hijau)

Terdapat perbedaan yang cukup signifikan untuk perbedaan penerapan kedua algoritma tadi (selisih 23 km), ini berarti algoritma *Dijkstra* gagal memberi solusi optimum untuk kasus di atas.

2.5 SPBU (Stasiun Pengisian Bahan Bakar untuk Umum)

SPBU (Stasiun Pengisian Bahan Bakar untuk Umum) merupakan prasarana umum yang disediakan oleh PT. Pertamina untuk masyarakat luas guna memenuhi kebutuhan bahan bakar. Pada umumnya SPBU menjual bahan bakar sejenis premium, solar, pertamax, dan pertamax plus. (spbu.pertamina.com)

Pada kasus pencarian SPBU ini, SPBU yang akan dicari nantinya hanya SPBU yang dikelola oleh PT. Pertamina. Hingga pertengahan Oktober 2005, perusahaan pemerintah, Pertamina, merupakan satu-satunya perusahaan yang mendirikan SPBU di Indonesia. Pada Oktober 2005, Shell menjadi perusahaan swasta pertama yang membuka SPBU-nya di Indonesia. Sekarang ini banyak SPBU yang juga menyediakan layanan tambahan. Misalnya, musholla, pompa angin, toilet dan lain sebagainya. SPBU modern, biasanya dilengkapi pula dengan swalayan kecil dan ATM. Tak heran apabila SPBU kini juga menjadi tempat istirahat. SPBU yang dikelola oleh PT. Pertamina yang terdapat di kota Pekanbaru terdiri dari beberapa SPBU, yang dapat dilihat pada Lampiran B. Berikut ini adalah salah satu SPBU Pertamina, yang dapat dilihat pada gambar 2.9.

Gambar 2.9. SPBU Pertamina (Pasti Pas)

(sumber: pastipas. pertamina.com/lokasi.asp?pastipas=oke)

2.6 Perangkat Lunak yang Digunakan

Aplikasi pencarian SPBU ini akan disimulasikan menggunakan beberapa bantuan perangkat lunak (*software*). Perangkat lunak yang digunakan pembuatan aplikasi ini adalah perangkat lunak yang sering digunakan dalam pengembangan aplikasi android. Beberapa perangkat lunak yang digunakan adalah:

2.6.1 JDK (*Java Development Kit)*

JDK merupakan dasar dari Android SDK. JDK diperlukan untuk men-*develop* aplikasi Android. Pada JDK sudah termasuk di dalamnya JRE (*Java Runtime Environment*) yang berfungsi untuk menjalankan program yang dibuat. Untuk sumber JDK dapat dilihat dan diunduh langsung di situs http://www.oracle.com/technetwork/java/javase/download.

2.6.2 Android SDK (*Software Development Kit)*

Android SDK adalah *tools* API (*Application Programming Interface*) yang diperlukan untuk memulai pengembangan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang dirilis oleh Google. Untuk sumber SDK Android dapat dilihat dan diunduh langsung ke situs resmi Android di http://developer.Android.com.

2.6.3 Eclipse IDE (*Integrated Development Environment)*

Eclipse IDE ini sama seperti editor teks biasa seperti *Notepad* atau *Wordpad*. Eclipse IDE merupakan *tool* untuk menulis kode program Android, juga sebagai *tool* yang menyatukan antara Java, Android SDK, dan Android ADT. Untuk sumber Eclipse IDE dapat dilihat dan diunduh langsung di situs http://www.eclipse.org/downloads/.

2.7 Peta Interaktif

Penggunaan peta yang akan dibangun nantinya merupakan penggunaan peta interaktif dengan mengaplikasikannya pada perangkat pintar (s*martphone*). Peta interaktif adalah penyajian peta dengan media web yang mudah digunakan untuk memperoleh informasi spasial. Informasi spasial itu sendiri adalah informasi mengenai hasil pengolahan data-data mengenai unit spasial yang format datanya dapat berupa vektor (poligon, titik, garis). Data spasial adalah data yang memiliki referensi kebumian (*georeference*), dimana berbagai data atribut terletak dalam berbagai unit spasial.

2.8 GPS & A-GPS

GPS (*Global Positioning System*) adalah sistem satelit navigasi dan penentuan posisi yang dimiliki dan dikelola oleh Amerika Serikat. Sistem ini didesain untuk memberikan posisi dan kecepatan tiga-dimensi serta informasi mengenai waktu, secara berkelanjutan di seluruh dunia tanpa

bergantung waktu dan cuaca, bagi banyak orang secara simultan. GPS dapat memberikan informasi posisi dengan ketelitian bervariasi dari beberapa millimeter (orde nol) sampai dengan puluhan meter. Beberapa kemampuan GPS antara lain dapat memberikan informasi tentang posisi, kecepatan, dan waktu secara cepat, akurat, murah, dimana saja di bumi ini tanpa tergantung cuaca. http://gaulwahyu.wordpress.com /2008/10/16/pengertian-gps/.

Assisted-Global Positioning System (A-GPS) merupakan penyempurnaan dari GPS sebagai satelit penentu posisi di belahan bumi. Metode A-GPS merupakan metode yang berbasis pada waktu. Pada metode ini, akan dilakukan pengukuran waktu tiba dari sebuah sinyal yang dikirimkan dari satelit GPS. Hal ini berarti pada perangkat yang digunakan harus memiliki fasilitas untuk mengakses GPS. A-GPS seperti halnya GPS, juga menggunakan satelit yang memancarkan sinyal radio ke penerima yang terpasang pada permukaan atas bumi. Penerima GPS dihubungkan dengan antena yang menerima sinyal radio untuk mengkalkulasi posisi penerima GPS.

A-GPS menawarkan solusi terakurat dari metode-metode yang telah ada sebelumnya. Lebih lanjut, A-GPS merupakan layanan yang menggabungkan sistem GPS dan layanan GSM (Global System for Mobile Communications). Layanan ini juga berguna untuk dapat menjembatani kekurangan dan kelebihan GPS dan LBS (*Location Based Service)*. A-GPS menjadikan proses akses informasi menggunakan satelit menjadi lebih mudah dan cepat, tidak terbatas ketika berada di dalam dan di luar ruangan ataupun gedung. (El-Rabbany, 2002).

Perbedaan A-GPS dan GPS secara teknis adalah : A-GPS mendapat bantuan data posisi dari operator selular utk mengkalkulasi posisi sedangkan GPS hanya mengandalkan sinyal satelit gps. Dari segi non teknis : A-GPS menawarkan penentuan posisi yang lebih cepat dan lebih akurat berkat bantuan server data operator.

Namun, di samping kelebihan yang dimiliki oleh A-GPS, terdapat juga kekurangannya, yaitu A-GPS yang melibatkan koneksi data, sehingga menyebabkan pemakaian A-GPS akan mengurangi pulsa sebagai biaya koneksi data via GPRS atau 3G.

2.9 Jalan

Jalan adalah prasarana transportasi darat yang meliputi segala bagian jalan, termasuk bangunan pelengkap dan perlengkapannya yang diperuntukkan bagi lalu lintas, yang berada pada permukaan tanah, di atas permukaan tanah, di bawah permukaan tanah dan/atau air, serta di atas permukaan air, kecuali jalan kereta api, jalan lori, dan jalan kabel. (KBBI. 1988)

2.9.1 Pengelompokkan Jalan

Jalan sesuai dengan peruntukannya terdiri atas jalan umum dan jalan khusus. (KBBI. 1988)

a. Jalan Umum : dikelompokkan menurut sistem, fungsi, status dan kelas.
    1. Sistem jaringan jalan : merupakan satu kesatuan jaringan jalan yang terdiri dari sistem jaringan jalan primer dan sistem jaringan jalan sekunder yang terjalin dalam hubungan hierarki.
    2. Fungsi : dikelompokkan ke dalam jalan arteri, jalan kolektor, jalan lokal, dan jalan lingkungan.
        a. Jalan arteri : jalan umum yang berfungsi melayani angkutan utama dengan ciri perjalanan jarak jauh, kecepatan rata-rata tinggi, dan jumlah jalan masuk dibatasi secara berdaya guna.
        b. Jalan kolektor : jalan umum yang berfungsi melayani angkutan pengumpul atau pembagi dengan ciri perjalanan jarak sedang, kecepatan rata-rata sedang, dan jumlah jalan masuk dibatasi.
        c. Jalan lokal : jalan umum yang berfungsi melayani angkutan setempat dengan ciri perjalanan jarak dekat, kecepatan rata-rata rendah, dan jumlah jalan masuk tidak dibatasi.
        d. Jalan lingkungan : jalan umum yang berfungsi melayani angkutan lingkungan dengan ciri perjalanan jarak dekat, dan kecepatan rata-rata rendah.
    3. Status : dikelompokkan ke dalam jalan nasional, jalan provinsi, jalan kabupaten, jalan kota, dan jalan desa.

4. Kelas : berdasarkan spesifikasi penyediaan prasarana jalan maka dikelompokkan atas :
   a. Bebas hambatan,
   b. Jalan raya,
   c. Jalan sedang, dan
   d. Jalan kecil.

   Sedangkan menurut berat kendaraan yang Iewat, jalan raya terdiri atas:
   a. Jalan Kelas I
   b. Jalan Kelas IIA
   c. Jalan Kelas IIB
   d. Jalan Kelas IIC
   e. Jalan Kelas III.

2.10 Analisa dan Perancangan Berorientasi Objek

Teknologi objek menganalogikan sistem aplikasi seperti kehidupan nyata yang didominasi oleh objek. Didalam membangun sistem berorientasi objek akan menjadi lebih baik apabila langkah awalnya didahului dengan proses analisis dan perancangan yang berorientasi objek. Tujuannya adalah untuk mempermudah *programmer* di dalam mendesain program dalam bentuk objek-objek dan hubungan antar objek tersebut untuk kemudian dimodelkan dalam sistem nyata (Suhendar, 2002). Perusahaan *software*, Rational *Software*, telah membentuk konsorsium dengan berbagai organisasi untuk meresmikan pemakaian *Unified Modelling Language* (UML) sebagai bahasa standar dalam *Object Oriented Analysis Design* (O*OAD).*

2.10.1 *Unified Modelling Language* (UML)

Unified Modelling Language (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem (Dharwiyanti dan Wahono, 2006).

Untuk merancang sebuah model, UML memiliki beberapa diagram lain: *usecase diagram, class diagram, statechart diagram, activity diagram, sequence diagram, collaboration diagram, component diagram, deployment diagram.*

2.10.1.1 *Usecase Diagram*

Usecase diagram merupakan sebuah gambaran fungsionalitas sebuah sistem. Sebuah *usecase* merepresentasikan interaksi antara aktor dengan sistem. *Usecase* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, *create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu (Dharwiyanti: 2006).

Dalam sebuah sistem *usecase diagram* akan sangat membantu dalam hal menyusun *requirment*, mengkomunikasikan rancangan dengan klien dan merancang *test case* untuk semua fitur yang ada pada sistem.

2.10.1.2 *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek danmerupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi) (Dharwiyanti: 2006).

Class diagram menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi,

dan lain-lain. *Class* memiliki tiga area pokok yaitu, nama, *stereotype* ,atribut dan metoda.

2.10.1.3 *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang memicu aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan (Dharwiyanti: 2006).

2.10.1.4 *Statechart Diagram*

(Suhendar dkk, 2002) menyebutkan bahwa *statechart diagram* digunakan untuk memodelkan perilaku dinamis satu kelas atau objek. *Statechart diagram* memperlihatkan urutan keadaan sesaat (*state*) yang dilalui sebuah objek, kejadian yang menyebabkan sebuah transisi dari satu *state* atau aktivitas kepada yang lainnya, dan aksi yang menyebabkan perubahan satu *state* atau aktivitas. *Statechart diagram* khususnya digunakan untuk memodelkan tahap-tahap diskrit dari sebuah siklus hidup objek, sedangkan *activity diagram* lebih cocok digunakan untuk memodelkan urutan aktivitas dalam suatu proses.

2.10.1.5 *Deployment Diagram*

Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, *server* atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi *server*, dan hal-hal lain yang bersifat fisikal Sebuah *node* adalah *server, workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya

TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini (Dharwiyanti: 2006).

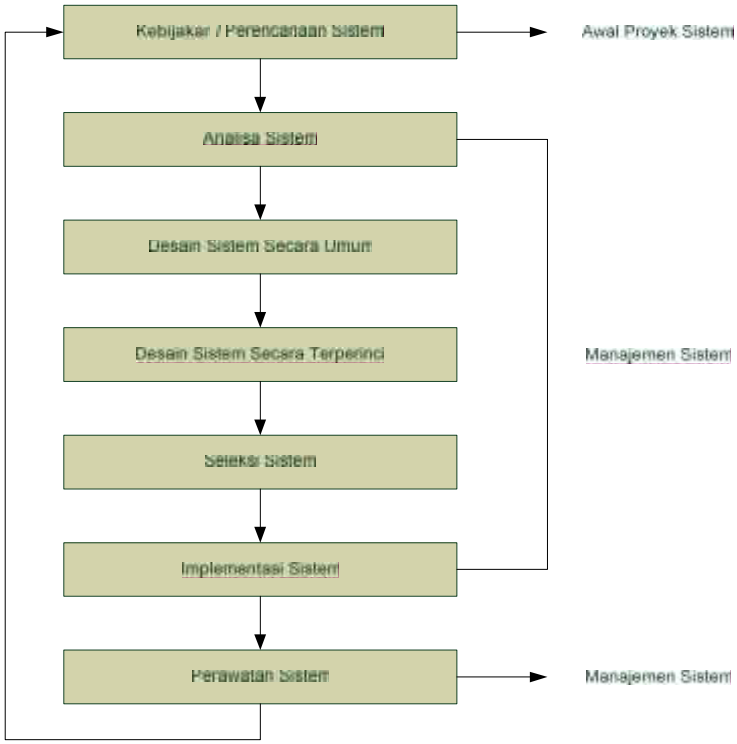
2.10.1.6 *Activity Diagram*

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang,bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi (Dharwiyanti: 2006).

Activity diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

2.11 Siklus Hidup Pengembangan Sistem

Siklus hidup pengembangan sistem *(System Development Life Cycle)* adalah proses evolusioner yang diikuti dalam menerapkan sistem atau subsistem informasi berbasis komputer. SDLC terdiri dari serangkaian tugas yang erat mengikuti langkah-langkah pendekatan sistem. Karena tugas-tugas tersebut mengikuti suatu pola yang teratur dan dilakukan secara *top-down*, SDLC sering disebut sebagai pendekatan air terjun *(waterfall approach)* bagi pengembangan dan penggunaan sistem. Tahapan siklus hidup pengembangan sistem adalah kebijakan dan perencanaan, analisa sistem, desain, seleksi, implementasi dan pemeliharaan (Kadir, 2003). Pada gambar 2.11 berikut akan dijelaskan mengenai siklus hidup pengembangan sistem.



Gambar 2.10. Siklus Hidup Pengembangan Sistem

a. Tahap Kebijakan dan Perencanaan Sistem

Sebelum suatu sistem informasi dikembangkan, umumnya terlebih dahulu dimulai dengan adanya suatu kebijakan dan perencanaan untuk mengembangkan sistem itu. Tanpa adanya perencanaan sistem yang baik, pengembangan sistem tidak akan berjalan sesuai dengan yang diharapkan. Kebijakan sistem merupakan landasan dan dukungan dari manajemen puncak untuk membuat perencanaan sistem. Perencanaan sistem merupakan pedoman untuk melakukan pengembangan sistem.

b. Tahap Analisa Sistem (*Analysis*)

Tahap ini dilakukan setelah tahap perencanaan sistem dan sebelum tahap desain sistem. Tahap ini merupakan tahap yang kritis dan sangat penting, karena kesalahan dalam tahap ini akan menyebabkan kesalahan di tahap selanjutnya. Di

dalam tahap ini terdapat langkah-langkah dasar yang harus dilakukan oleh analis sistem sebagai berikut:

1. *Identify*, yaitu mengidentifikasi masalah.
2. *Understand*, yaitu memahami kerja dari sistem yang ada.
3. *Report*, yaitu membuat laporan hasil analisa.

c. Tahap Desain Sistem (*Design*)

Tahap ini dilakukan setelah tahap analisa selesai dilakukan. Setelah tahap analis dilakukan, maka analis sistem telah mendapatkan gambaran dengan jelas apa yang harus dilakukan. Untuk itu seorang analis sistem harus memikirkan bagaimana membentuk sistem tersebut.

d. Tahap Seleksi Sistem (*Selection*)

Hasil desain sistem ini belum dapat diimplementasikan. Untuk itu komponen-komponen secara fisik perlu dimiliki. Komponen fisik sistem ini adalah komponen teknologi yang dapat berupa perangkat keras dan perangkat lunak, karena banyaknya alternatif teknologi yang tersedia dan banyaknya penyedia teknologi. Maka perlu diadakan suatu penyeleksian.

e. Tahap Implementasi Sistem (*Implementation*)

Pada tahap ini suatu sistem siap untuk dioperasikan. Tahap ini terdiri dari langkah-langkah sebagai berikut:

1. Menerapkan rencana implementasi.
2. Melakukan kegiatan implementasi.
3. Tindak lanjut implementasi.

f. Tahap Pemeliharaan Sistem (*Maintenance*)

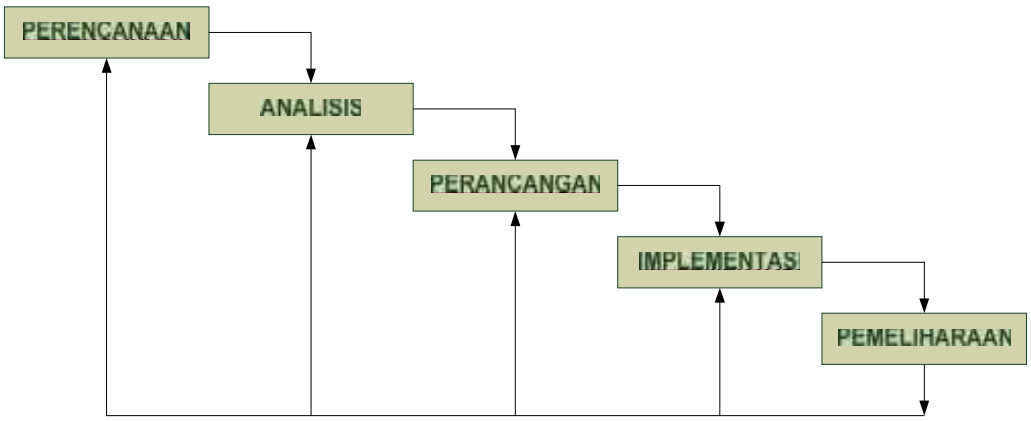
Tahap ini merupakan tahap akhir dalam sebuah pengembangan sistem yang lebih menekankan pada pemeliharaan sistem.

2.11.1 Pengembangan Sistem

Perancangan perangkat lunak implementasi aplikasi SPBU Pekanbaru, dibangun dengan menggunakan model sistem yang dikembangkan dalam menganalisa perangkat lunak menggunakan metode konvensional dengan

memanfaatkan model atau paradigma siklus hidup klasik atau lebih sering disebut *Waterfall Model*.

Model ini bersifat linear karena prosesnya mengalir secara sekuensial mulai dari awal hingga akhir. Model ini mensyaratkan penyelesaian suatu tahap secara tuntas sebelum dilanjutkan pada tahap berikutnya. Hasil-hasilnya harus didokumentasikan dengan baik. Gambar 2.12 berikut adalah gambaran umum dari kerangka kerja model *waterfall*.



Gambar 2.11. Kerangka Kerja Model *Waterfall*

Keterangan:

1. Perencanaan

   Menyangkut studi kebutuhan pengguna, studi kelayakan baik secara teknis maupun secara teknologi serta penjadwalan pengembangan perangkat lunak. Dapat juga dikatakan sebagai definisi kebutuhan sistem.

2. Analisa

   Tahap dimana kita berusaha mengenali seluruh permasalahan yang muncul pada pengguna (*user*), mengenali komponen-komponen sistem, objek-objek, hubungan antar objek, dan sebagainya. Merupakan analisa keadaan internal dan eksternal.

3. Perancangan

   Merupakan tahap pencarian solusi dari permasalahan yang didapat dari tahap analisa.

4. Implementasi

Tahap pengimplementasian rancangan sistem ke situasi nyata. Pada tahap ini dimulai dengan proses pemilihan perangkat keras, penyusunan perangkat lunak aplikasi (*coding*), dan pengujian (*testing*) apakah sistem sudah sesuai dengan kebutuhan. Jika belum, dilakukan proses iteratif, yaitu kembali ke tahap-tahap sebelumnya.

5. Pemeliharaan

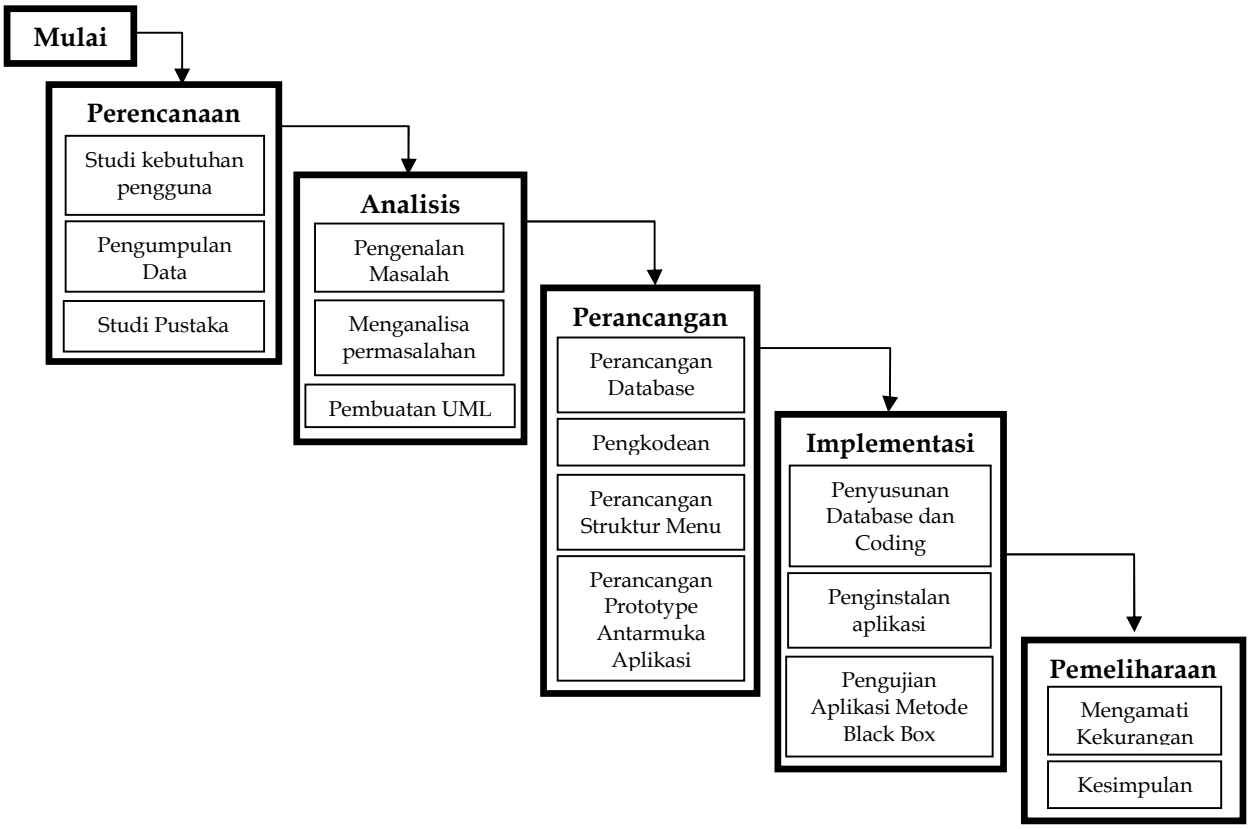
Mulai melakukan pengoperasian sistem dan melakukan perbaikan-perbaikan kecil jika diperlukan. Jika masa penggunaan sistem habis, maka akan kembali ke tahap pertama, yaitu perencanaan.

BAB III

METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Tahapan penelitian yang akan dilaksanakan pada Tugas Akhir ini menggunakan metode *Waterfall Model*. Seperti yang telah dijelaskan pada bab landasan teori bahwa *Waterfall Model*, merupakan suatu metode yang digunakan untuk proses pembangunan sebuah perangkat lunak karenaprosesnya mengalir secara sekuensial mulai dari awal hingga akhir. Gambar 3.1 di bawah ini menjelaskan tahapan penelitian terhadap aplikasi yang akan dibangun berdasarkan kepada metode *Waterfall Model*.



Gambar 3.1. Tahapan penelitian dengan metode *Waterfall Model*.

3.2. Tahapan *Waterfall Model*

Berikut ini akan diuraikan tahapan-tahapan penerapan algoritma *Floyd-Warshall* yang diimplementasikan pada aplikasi pencarian SPBU dengan rute terpendek menggunakan metode *Waterfall Model*.

3.2.1. Tahap Perencanaan

Tahap perencanaan merupakan suatu tahapan yang menyangkut studi kebutuhan pengguna, studi kelayakan baik secara teknis maupun secara teknologi serta penjadwalan pengembangan aplikasi yang dibangun. Pada penelitian ini, tahapan perencanaan akan dijelaskan sebagai berikut:

a. Pada bab 1 yang berisi tentang layak atau tidaknya penelitian ini dilanjutkan untuk penerapan algoritma *Floyd-Warshall* yang diimplementasikan pada aplikasi pencarian SPBU dengan rute terpendek, yaitu mencakup latar belakang permasalahan, pokok permasalahan, tujuan dan batasan permasalahan.
b. Pada bab 2 berisi mengenai studi kebutuhan pengguna berupa pengumpulan data-data yang dibutuhkan serta pemilihan *software* yang akan digunakan dalam pembuatan aplikasi.
c. *Study literature* yang berhubungan dengan pembangunan aplikasi, mencakup penelusuran teori-teori yang berhubungan dengan permasalahan dan teknik pembangunan pada penerapan algoritma *Floyd-Warshall* yang diimplementasikan pada aplikasi pencarian SPBU dengan rute terpendek.

3.2.2. Tahap Analisis

Tahap analisis merupakan tahapan untuk mengenali seluruh permasalahan yang muncul pada pengguna serta mengenali komponen-komponen yang dibutuhkan oleh aplikasi yang dibangun, menggambarkan hubungan antar objek dan sebagainya. Merupakan analisa keadaan internal dan eksternal. Pada tahapan analisis ini akan dilakukan tugas-tugas sebagai berikut:

a. Pengenalan masalah, yaitu memahami permasalahan yang terjadi pada penerapan algoritma *Floyd-Warshall* yang diimplementasikan pada aplikasi pencarian SPBU dengan rute terpendek yang akan dibangun.
b. Menganalisa permasalahan yang terjadi pada penerapan algoritma *Floyd-Warshall* yang diimplementasikan pada aplikasi pencarian SPBU dengan rute terpendek yang akan dibangun.
c. Penyempurnaan perancangan aplikasi dengan *Unified Modeling Language* (UML).

3.2.3. Tahap Perancangan

Tahap perancangan merupakan tahapan pencarian solusi dari permasalahan yang dianalisa pada tahap analisa. Pada tahap perancangan ini akan dilakukan tugas-tugas sebagai berikut:

a. Membangun rancangan *database* untuk aplikasi pencarian SPBU dengan rute terpendek.
b. Membangun penerapan algoritma *Floyd-Warshall* yang diimplementasikan pada aplikasi pencarian SPBU dengan rute terpendek yang berpedoman pada model *use-case* menggunakan bahasa pemrograman Java.
c. Merancang dan membangun struktur menu untuk aplikasi pencarian SPBU dengan rute terpendek.
d. Merancang dan membangun *prototype* antarmuka aplikasi pencarian SPBU dengan rute terpendek.

3.2.4. Tahap Implementasi

Tahap pengimplementasian rancangan aplikasi ke situasi nyata. Pada tahap ini dimulai dengan proses pemilihan perangkat keras, penyusunan perangkat lunak aplikasi (*coding*), dan pengujian (*testing*) apakah aplikasi yang dibangun sudah sesuai dengan kebutuhan pengguna. Jika belum, maka akan dilakukan proses iteratif, yaitu kembali ke tahap-tahap sebelumnya. Pada tahapan ini akan dilakukan tugas-tugas sebagai berikut:

a. Menyusun perangkat lunak (*software*) berupa *database* dan *coding* yang telah dirancang dan dibuat sebelumnya pada tahap perancangan.
b. Memilih perangkat keras (*hardware*) yang akan digunakan dalam mengembangkan aplikasi yang dibangun.
c. Menguji aplikasi pencarian SPBU dengan rute terpendek yang mengimplementasikan algoritma *Floyd-Warshall,* apakah aplikasi yang dibangun telah sesuai dengan kebutuhan atau belum.

3.2.5. Tahap Pemeliharaan

Tahap pemeliharaan merupakan tahapan yang berisi mengenai proses peninjauan tentang aplikasi yang telah diuji. Pada tahapan ini terdapat kesimpulan dari aplikasi pencarian SPBU dengan rute terpendek yang mengimplementasikan algoritma *Floyd-Warshall*.

BAB IV

ANALISA DAN PERANCANGAN

Bab ini merupakan bagian dari tahapan analisis dan tahapan perancangan, dimana akan dilakukan analisa sejalan dengan pembuatan deskripsi arsitektur yang dibutuhkan pada aplikasi dan pembuatan UML *(Unified Modelling Language)* sebagai bagian dari tahapan analisis, dan dilakukan perancangan *database*, perancangan struktur menu aplikasi, dan perancangan *prototype* antarmuka aplikasi yang akan dibangun sebagai bagian dari tahapan perancangan.

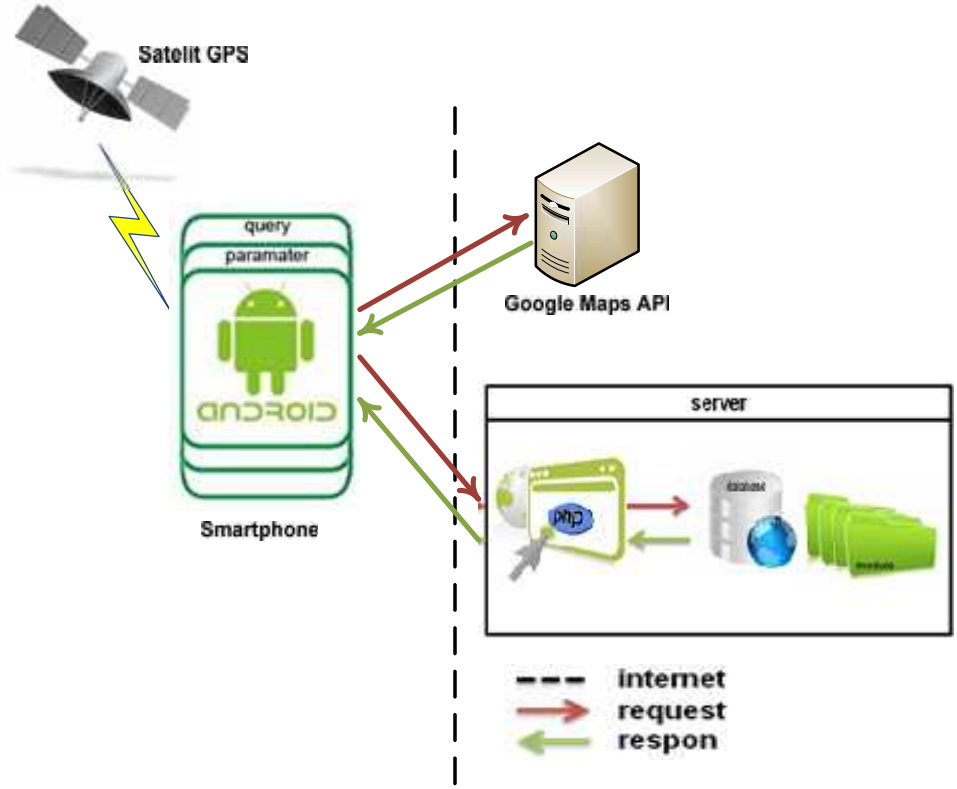
4.1 Tahapan Analisis

4.1.1 Gambaran Umum Sistem

Aplikasi pencarian SPBU dengan rute terpendek yang dibahas dalam penelitian ini adalah aplikasi yang berbasis *client-server*, dan akan dijalankan pada perangkat bergerak-pintar (*smartphone*) dengan sistem operasi Android. Aplikasi ini menerapkan algoritma *Floyd-Warshall* dalam pencarian SPBU dengan rute terpendek, dan *database Google Map* sebagai *server*-nya, menggunakan bahasa pemrograman PHP sebagai *connector* dalam pengiriman *request* dan penerimaan respon terhadap *server* yang menggunakan Apache dan *database* MySQL.

Aplikasi pencarian SPBU dengan rute terpendek ini hanya mampu mengakses data berupa peta dalam bentuk vektor, yaitu dengan format titik, garis dan poligon. Aplikasi ini hanya memberikan akses sebagai pembaca *(reader-user)* tanpa bisa memanipulasi *database*. Aplikasi *client* yang dibangun merupakan aplikasi yang mampu me-*request* akses untuk menampilkan lokasi SPBU yang ada di Pekanbaru dari *server* yang meliputi *database* melalui bahasa pemrograman berbasis *web*, kemudian *server* akan memberikan respon kepada *client* melalui jalur pengiriman *request* sebelumnya.

Gambaran umum aplikasi pencarian SPBU dengan rute terpendek dengan menerapkan Algoritma *Floyd-Warshall* ini bertujuan memberikan gambaran mengenai struktur menu dan konsep dasar aplikasi. Untuk lebih jelasnya deskripsi arsitektur sistem ini dapat dilihat pada gambar 4.1. di bawah ini.



Gambar 4.1. Arsitektur Sistem

Dari gambar 4.1 diatas dapat dilihat proses kerja aplikasi yang akan dibuat, ada tiga bagian penting yang saling terhubung dalam kerja sistemnya, diantaranya:

1. Perangkat Android merupakan perangkat tempat berjalannya aplikasi pencarian SPBU dengan rute terpendek yang menerapkan Algoritma *Floyd-Warshall*. Dari perangkat inilah pengguna berinteraksi dengan sistem melalui PHP, dengan memanfaatkan jaringan internet *mobile* untuk mengakses *server*.

2. *Server*, terdiri dari dua bagian, yaitu:
    a. *Connector* (PHP), berfungsi sebagai jembatan penghubung antara sistem yang berjalan pada perangkat Android (*client*) dan *database*. Peran *connector* sangat penting, karena sisi *client* tidak bisa langsung menyentuh *database* tanpa perantara. *Connector* ini yang bertugas mengirimkan *request* dan respon antara *client* dan *server*.
    b. *Database*, merupakan bagian yang berfungsi sebagai *database* dari aplikasi pencarian SPBU dengan rute terpendek yang menerapkan Algoritma *Floyd-Warshall*. *Database* ini yang bertanggung jawab memberikan respon sesuai *request* dari *client*. *Database* yang digunakan adalah MySQL.

Pengguna langsung bisa mendapatkan hak akses penuh jika aplikasi ini telah dipasangkan pada perangkat Android. Aplikasi ini juga tidak membatasi hak akses.

4.1.2 Algoritma *Floyd-Warshall*

4.1.2.1 Analisis Cara Kerja Algoritma *Floyd-Warshall*

Algoritma Floyd-Warshall bertujuan mencari panjang lintasan (rute) terpendek dari node asal ke node tujuan (node akhir) dalam sebuah graf dengan cara membandingkan semua kemungkinan lintasan (rute) pada graf untuk setiap sisi dari semua simpul. Menariknya, algoritma ini mampu mengerjakan proses perbandingkan ini sebanyak V3 kali (bandingkan dengan kemungkinan jumlah sisi sebanyak V2 (kuadrat jumlah simpul) pada graf, dan setiap kombinasi sisi diujikan). Hal tersebut bisa terjadi karena adanya perkiraan pengambilan keputusan (pemilihan rute terpendek) pada setiap tahap antara dua simpul, hingga perkiraan tersebut diketahui sebagai nilai optimal.

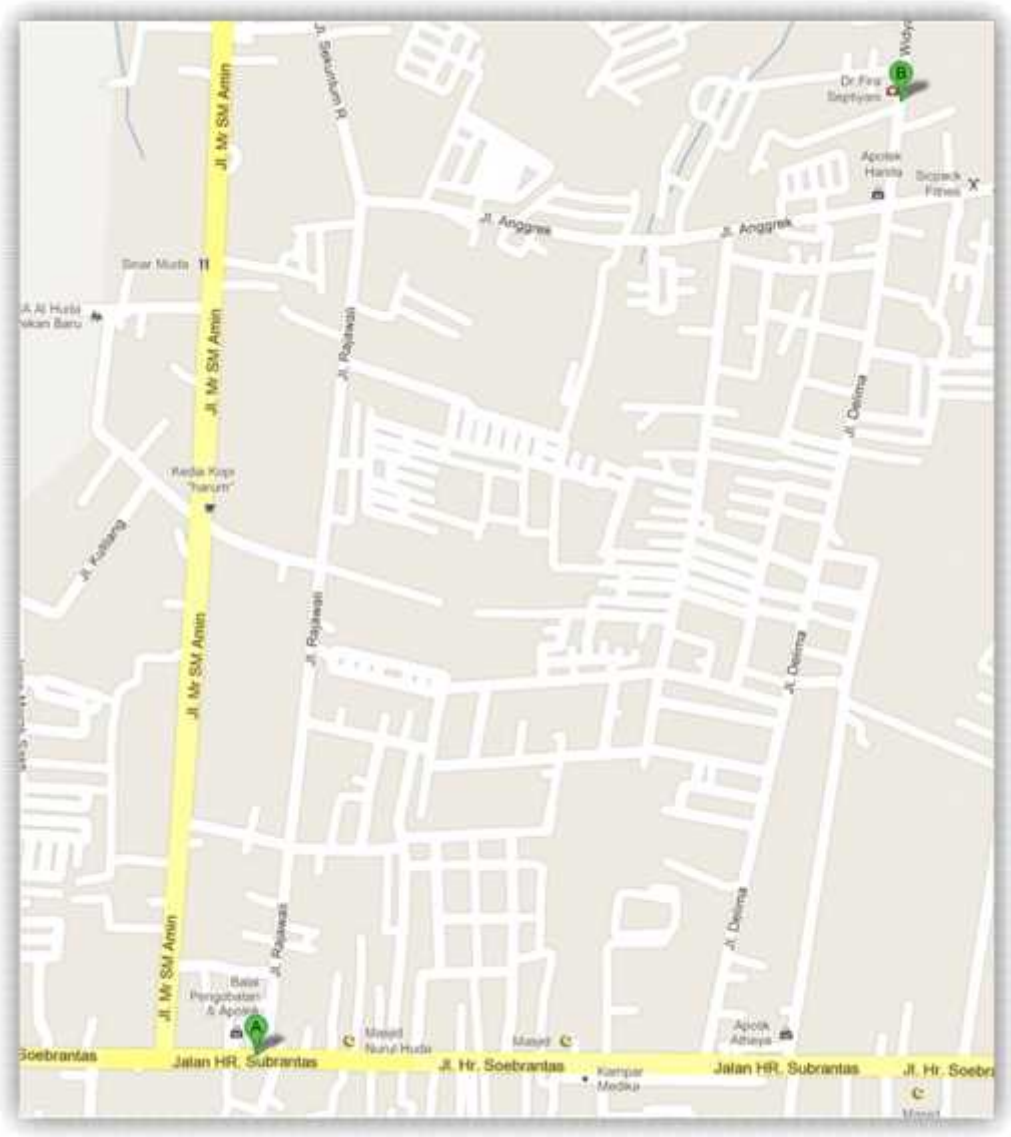
Tahapan dalam menentukan rute terpendek pada algoritma *floyd warshall* yaitu :

1. Persoalan dibagi atas beberapa tahap dan sebaiknya membuat *flowchart* untuk lebih mempermudah pencarian.

2. Ketika masuk ke suatu tahap, hasil pada tahap tersebut akan menjadi simpul baru untuk tahap selanjutnya.
3. Tentukan 1 titik sebagai titik awal agar pencarian algoritma dapat dilakukan.
4. Cari node yang bertetangga langsung dengan titik simpul (titik awal).
5. Bandingkan rute tiap tahap yang bobotnya sudah dijumlahkan dengan bobot-bobot pada tahap sebelumnya, jika sudah maka cari rute dengan bobot yang terkecil sampai proses pencarian berakhir.
6. Bobot yang dimiliki oleh suatu tahap akan dijumlahkan dengan bobot yang ada pada tahap-tahap sebelumnya seiring dengan bertambahnya jumlah tahapan.
7. Pencarian berhenti apabila node tujuan (node akhir) telah ditemukan.
8. Setelah proses selesai, lihat ada berapa rute yang diperoleh untuk dapat sampai ke suatu tujuan tertentu dan pilih rute yang paling kecil untuk menjadi rute terpendek dari Algoritma *Floyd-Warshall.*

4.1.2.2 Perhitungan Manual Algoritma *Floyd-Warshall*

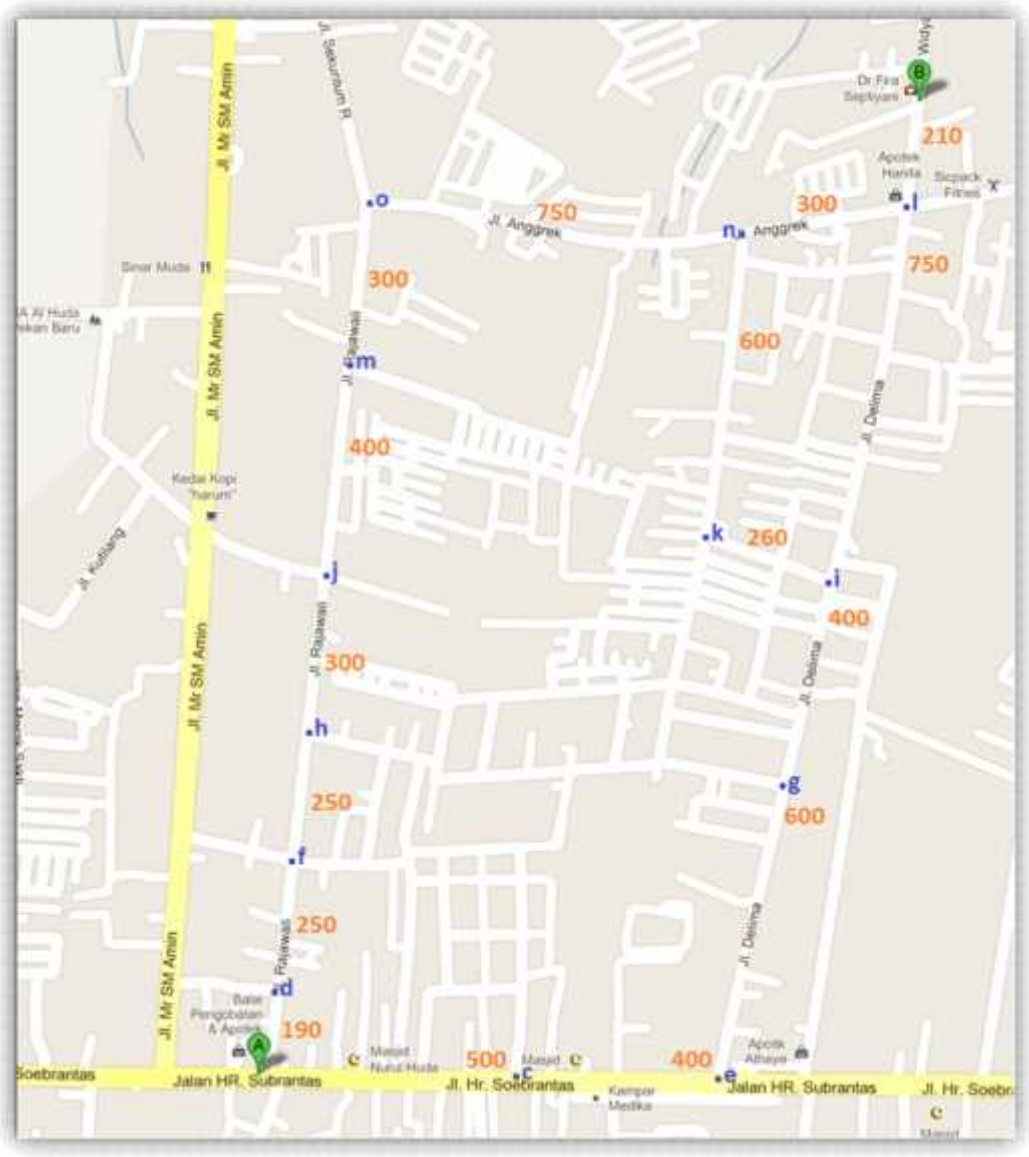
Untuk rincian jelasnya dari cara kerja algoritma *Floyd-Warshall*, berikut ini akan diberikan contoh kasus yang menampilkan perhitungan manual dari algoritma *Floyd-Warshall*. Dicontohkan terdapat suatu graf berbobot yang menampilkan kondisi keterhubungan antar jalan di suatu daerah dalam kasus ini, dimisalkan seseorang akan melakukan perjalanan dari arah UIN Suska Riau yang berada pada jalan HR. Subrantas (Poin A) menuju SPBU 14.2826.123 yang berada pada jalan Widya Graha II (Poin B). Seperti yang dapat dilihat pada peta di gambar 4.2 berikut.



Gambar 4.2. Contoh graf tidak berarah dan tidak berbobot

Berdasarkan pada gambar 4.2 tersebut, jalan-jalan yang berada pada peta yang ditampilkan belum memiliki bobot atau nilai untuk masing-masing jarak tempuh dari poin A ke poin B. Untuk dapat menghitung jarak optimal yang akan ditempuh dari poin A ke poin B menggunakan algoritma *Floyd-Warshall* maka harus diketahui jarak dari masing-masing jalan yang terhubung dari poin A ke poin B.

Pada gambar 4.3 berikut akan ditampilkan jarak dari masing-masing jalan yang terhubung dari poin A menuju poin B.



Gambar 4.3. Graf tidak berarah dan berbobot

4.1.2.3 *Flowchart* Algoritma *Floyd-Warshall*

Langkah awal adalah mengelompokkan proses pencarian setiap tahap dan mencari node yang terhubung langsung menuju titik simpul yang sedang ditinjau, prosesnya adalah sebagai berikut:

Tahap 1 : Pada tahap ini titik simpul yang sedang ditinjau adalah A. Kemudian titik A tersebut memiliki 2 kandidat solusi yaitu titik c dan d.

Tahap 2 : Setelah tahap 1 selesai ditinjau, maka sekarang proses yang dilakukan ada pada tahap 2, dimana titik kandidat solusi yang ada pada tahap 1 yaitu titik c dan d untuk selanjutnya dijadikan titik simpul pada tahap 2. Titik c dan d ini memiliki kandidat solusi yaitu titik e dan f.

Tahap 3 : Setelah tahap 2 selesai ditinjau, maka sekarang proses yang akan dilakukan ada pada tahap 3, dimana titik kandidat solusi yang ada pada tahap 2 yaitu titik e dan f untuk selanjutnya dijadikan titik simpul pada tahap 3. Titik e dan f ini memiliki kandidat solusi yaitu titik g dan h.

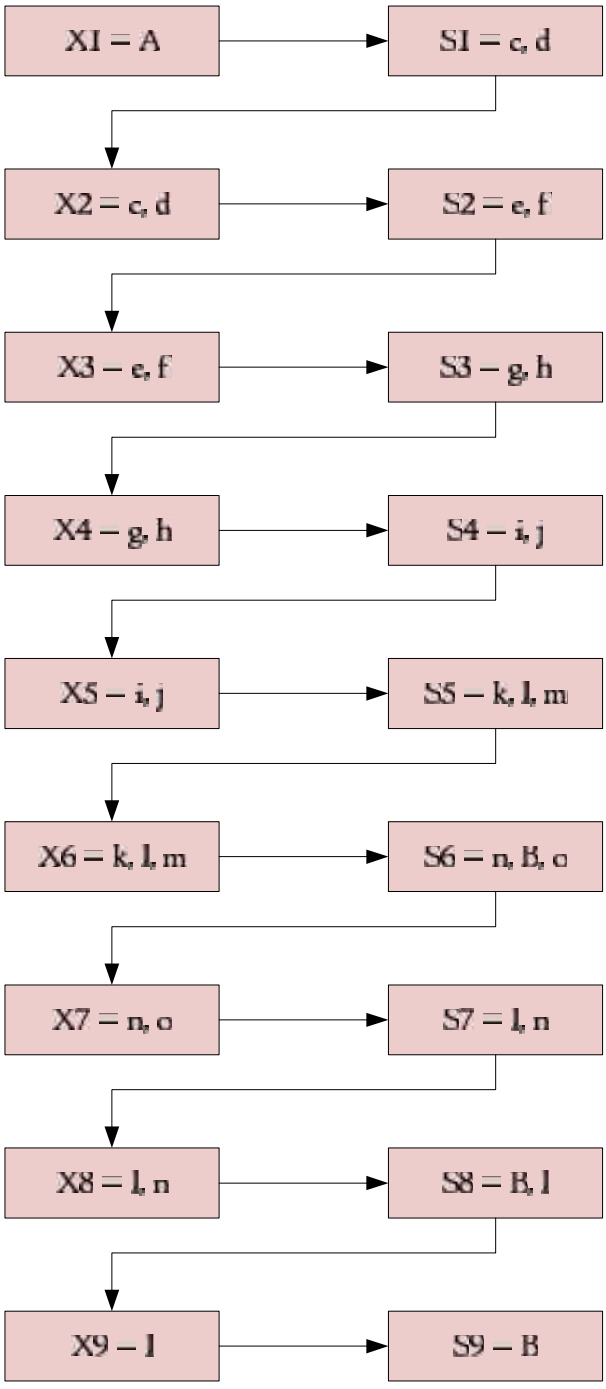
Tahap 4 : Setelah tahap 3 selesai ditinjau, maka sekarang proses yang akan dilakukan ada pada tahap 4, dimana titik kandidat solusi yang ada pada tahap 3 yaitu titik g dan h untuk selanjutnya dijadikan titik simpul pada tahap 4. Titik g dan h ini memiliki kandidat solusi yaitu titik i dan j.

Tahap 5 : Setelah tahap 4 selesai ditinjau, maka sekarang proses yang akan dilakukan ada pada tahap 5, dimana titik kandidat solusi yang ada pada tahap 4 yaitu titik i dan j untuk selanjutnya dijadikan titik simpul pada tahap 5. Titik i dan j ini memiliki kandidat solusi yaitu titik k, l dan m.

Tahap 6 : Setelah tahap 5 selesai ditinjau, maka sekarang proses yang akan dilakukan ada pada tahap 6, dimana titik kandidat solusi yang ada pada tahap 5 yaitu titik k, l dan m untuk selanjutnya dijadikan titik simpul pada tahap 6. Titik k, l dan m ini memiliki kandidat solusi yaitu titik n, B dan o.

Tahap 7 : Pada tahap 7 sampai dengan tahap terakhir akan dijelaskan melalui *flowchart*, proses yang dilakukan pada tahap 7 sama dengan proses yang dilakukan pada tahapan-tahapan sebelumnya.

Penggambaran *flowchart* berikut merupakan deskripsi alur proses dari pencarian menggunakan algoritma *Floyd-Warshall* berdasarkan kasus yang ada.



Gambar 4.4. *Flowchart* kasus menggunakan Algoritma *Floyd-Warshall*

Deskripsi *flowchart* dari gambar 4.4:

Xi : Titik Simpul

Si : Kandidat Solusi

4.1.2.4 Analisa Algoritma *Floyd-Warshall*

Diketahui:

f : nilai jarak antar titik per-tahap (dalam meter)

k : tahap ke-(1, 2, 3, 4, ..., n)

x : titik simpul / titik asal per-tahap

s : titik simpul yang sedang ditinjau untuk menjadi kandidat solusi per-tahap

Tahap 1: $f_1(s) = cx_1s$

Titik Tujuan	Titik Asal	
s1	Solusi Optimum	
	f1 (s)	*x1*
c	500	A
d	190	A

Keterangan :

Pada tahapan pertama ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **A** menuju titik-titik yang saling berhubungan ke titik **B**. Pada tahapan pertama titik **A** terhubung dengan titik **c** dan **d**, yang masing-masing titik memiliki nilai jarak **500 m** dan **190 m**. Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 1 pada gambar 4.5.

Gambar 4.5. Tahap 1-Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 2: $f_2(s) = min\ x_2 \{cx_2s + f_{2-1}(x_2)\}$

Titik Tujuan	Titik Asal			
x2	*f2 (x2, s) = cx2s + f1 (x2)*		Solusi Optimum	
s2	*c*	*d*	*f2 (s)*	*x2*
e	900	-	900	c
f	-	440	440	d

Keterangan :

Pada tahapan kedua ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **c** dan **d**, yang sebelumnya menjadi titik kandidat solusi pada tahap 1. Dan mempunyai kandidat solusi yaitu titik **e** dan **f**, yang masing-masing titik memiliki nilai jarak sebagai berikut:

1. Jarak yang didapat pada perhitungan tahap 1, yaitu: **A → c = 500 m**
2. Jarak yang didapat pada perhitungan tahap 2, yaitu: **c → e = 400 m**
3. Total jarak yang didapat sampai perhitungan tahap 2, yaitu:

   (A → c) + (c → e) = 500 m + 400 m = 900 m

Sedangkan perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Jarak yang didapat pada perhitungan tahap 1, yaitu: **A → d = 190 m**
2. Jarak yang didapat pada perhitungan tahap 2, yaitu: **d → f = 250 m**
3. Total jarak yang didapat sampai perhitungan tahap 2, yaitu:

   (A → d) + (d → f) = 190 m + 250 m = 440 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 2 pada gambar 4.6.

Gambar 4.6. Tahap 2 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 3: $f_3(s) = min\ x_3 \{cx_3s + f_{3-1}(x_3)\}$

Titik Tujuan	Titik Asal			
x3	*f3 (x3, s) = cx3s + f2 (x3)*		Solusi Optimum	
s3	*e*	*f*	*f3 (s)*	*x3*
g	1500	-	1500	e
h	-	690	690	f

Keterangan :

Pada tahapan ketiga ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **e** dan **f**, yang sebelumnya menjadi titik kandidat solusi pada tahap 2. Dan mempunyai kandidat solusi yaitu titik **g** dan **h**, yang masing-masing titik memiliki nilai jarak sebagai berikut:

1. Total jarak yang didapat sampai perhitungan tahap 2, yaitu:
   (A → c) + (c → e) = 500 m + 400 m = 900 m
2. Jarak yang didapat pada perhitungan tahap 3, yaitu: **e → g = 600 m**
3. Total jarak yang didapat sampai perhitungan tahap 3, yaitu:
   (A → e) + (e → g) = 900 m + 600 m = 1500 m

Sedangkan perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Total jarak yang didapat sampai perhitungan tahap 2, yaitu:
   (A → d) + (d → f) = 190 m + 250 m = 440 m
2. Jarak yang didapat pada perhitungan tahap 3, yaitu: **f → h = 250 m**
3. Total jarak yang didapat sampai perhitungan tahap 3, yaitu:
   (A → f) + (f → h) = 440 m + 250 m = 690 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 3 pada gambar 4.7.

Gambar 4.7. Tahap 3 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 4: $f_4(s) = min\ x_4 \{cx_4s + f_{4-1}(x_4)\}$

Titik Tujuan	**Titik Asal**			
x4	***f4 (x4, s) = cx4s + f3 (x4)***		**Solusi Optimum**	
s4	***g***	***h***	***f4 (s)***	***x4***
i	1900	-	1900	g
j	-	990	990	h

Keterangan :

Pada tahapan keempat ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **g** dan **h**, yang sebelumnya menjadi titik kandidat solusi pada tahap 3. Dan mempunyai kandidat solusi yaitu titik **i** dan **j**, yang masing-masing titik memiliki nilai jarak sebagai berikut:

1. Total jarak yang didapat sampai perhitungan tahap 3, yaitu:

   (A → g) = 1500 m

2. Jarak yang didapat pada perhitungan tahap 4, yaitu: **g → i = 400 m**
3. Total jarak yang didapat sampai perhitungan tahap 4, yaitu:

   (A → g) + (g → i) =1500 m + 400 m = 1900 m

Sedangkan perhitungan untuk jarak dengan titik yang lainnya adalah:

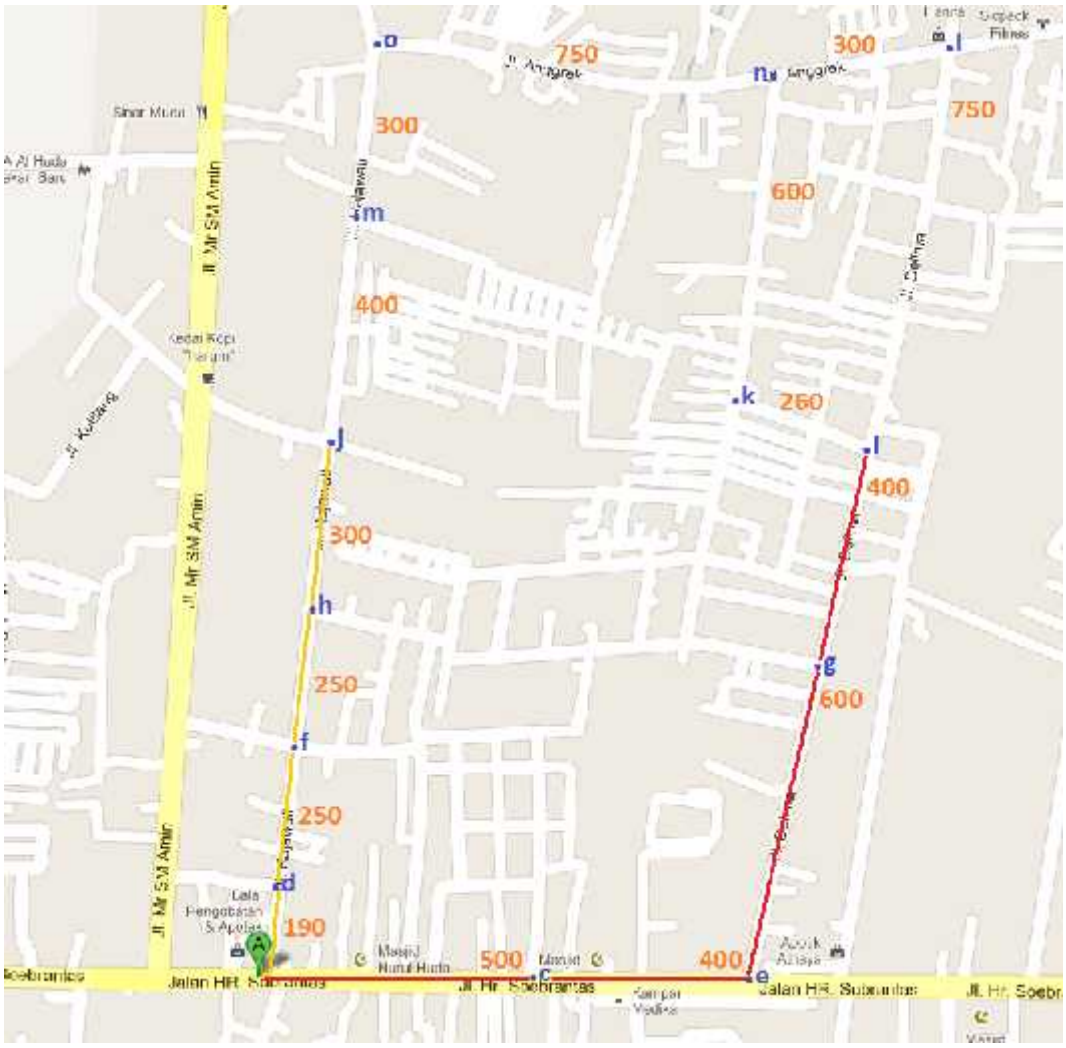
1. Total jarak yang didapat sampai perhitungan tahap 3, yaitu:

(A → h) = 690 m

2. Jarak yang didapat pada perhitungan tahap 4, yaitu: **h → j = 300 m**
3. Total jarak yang didapat sampai perhitungan tahap 4, yaitu:

   (A → h) + (h → j) = 690 m + 300 m = 990 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 4 pada gambar 4.8.



Gambar 4.8. Tahap 4 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 5: $f_5\ (s) = min\ x_5\ \{cx_5s + f_{5-1}\ (x_5)\}$

Titik Tujuan	Titik Asal			
x5	*f5 (x5, s) = cx5s + f4 (x5)*		Solusi Optimum	
s5	*i*	*j*	*f5 (s)*	*x5*
k	2160	-	2160	i
l	2650	-	2650	i
m	-	1390	1390	j

Keterangan :

Pada tahapan kelima ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **i** dan **j**, yang sebelumnya menjadi titik kandidat solusi pada tahap 4. Dan mempunyai kandidat solusi yaitu titik **k**, **l** dan **m**, yang masing-masing titik memiliki nilai jarak sebagai berikut:

1. Total jarak yang didapat sampai perhitungan tahap 4, yaitu:
   (A → i) = 1900 m
2. Jarak yang didapat pada perhitungan tahap 5, yaitu: **i → k = 260 m**
3. Total jarak yang didapat sampai perhitungan tahap 5, yaitu:
   (A → i) + (i → k) =1900 m + 260 m = 2160 m

Perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Total jarak yang didapat sampai perhitungan tahap 4, yaitu:
   (A → i) = 1900 m
2. Jarak yang didapat pada perhitungan tahap 5, yaitu: **i → l = 750 m**
3. Total jarak yang didapat sampai perhitungan tahap 5, yaitu:
   (A → i) + (i → l) = 1900 m + 750 m = 2650 m

Sedangkan perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Total jarak yang didapat sampai perhitungan tahap 4, yaitu:
   (A → j) = 990 m
2. Jarak yang didapat pada perhitungan tahap 5, yaitu: **j → m = 400 m**
3. Total jarak yang didapat sampai perhitungan tahap 5, yaitu:
   (A → j) + (j → m) = 990 m + 400 m = 1390 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 5 pada gambar 4.9.

Gambar 4.9. Tahap 5 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 6: $f_6(s) = min\ x_6 \{cx_6s + f_{6-1}(x_6)\}$

Titik Tujuan	**Titik Asal**				
x6	***f6 (x6, s) = cx6s + f5 (x6)***			**Solusi Optimum**	
s6	***k***	***l***	***m***	***f6 (s)***	***x6***
n	2760	2950	-	2760	k
B	-	2860	-	2650	l
o	-	-	1690	1690	m

Keterangan :

Pada tahapan keenam ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **k**, **l** dan **m**, yang sebelumnya menjadi titik kandidat solusi pada tahap 5. Dan mempunyai kandidat solusi yaitu titik **n**, **B** dan **o**, yang masing-masing titik memiliki nilai jarak sebagai berikut:

1. Total jarak yang didapat sampai perhitungan tahap 5, yaitu:
   (A → k) = 2160 m
2. Jarak yang didapat pada perhitungan tahap 6, yaitu: **k → n = 600 m**
3. Total jarak yang didapat sampai perhitungan tahap 6, yaitu:
   (A → k) + (k → n) = 2160 m + 600 m = 2760 m

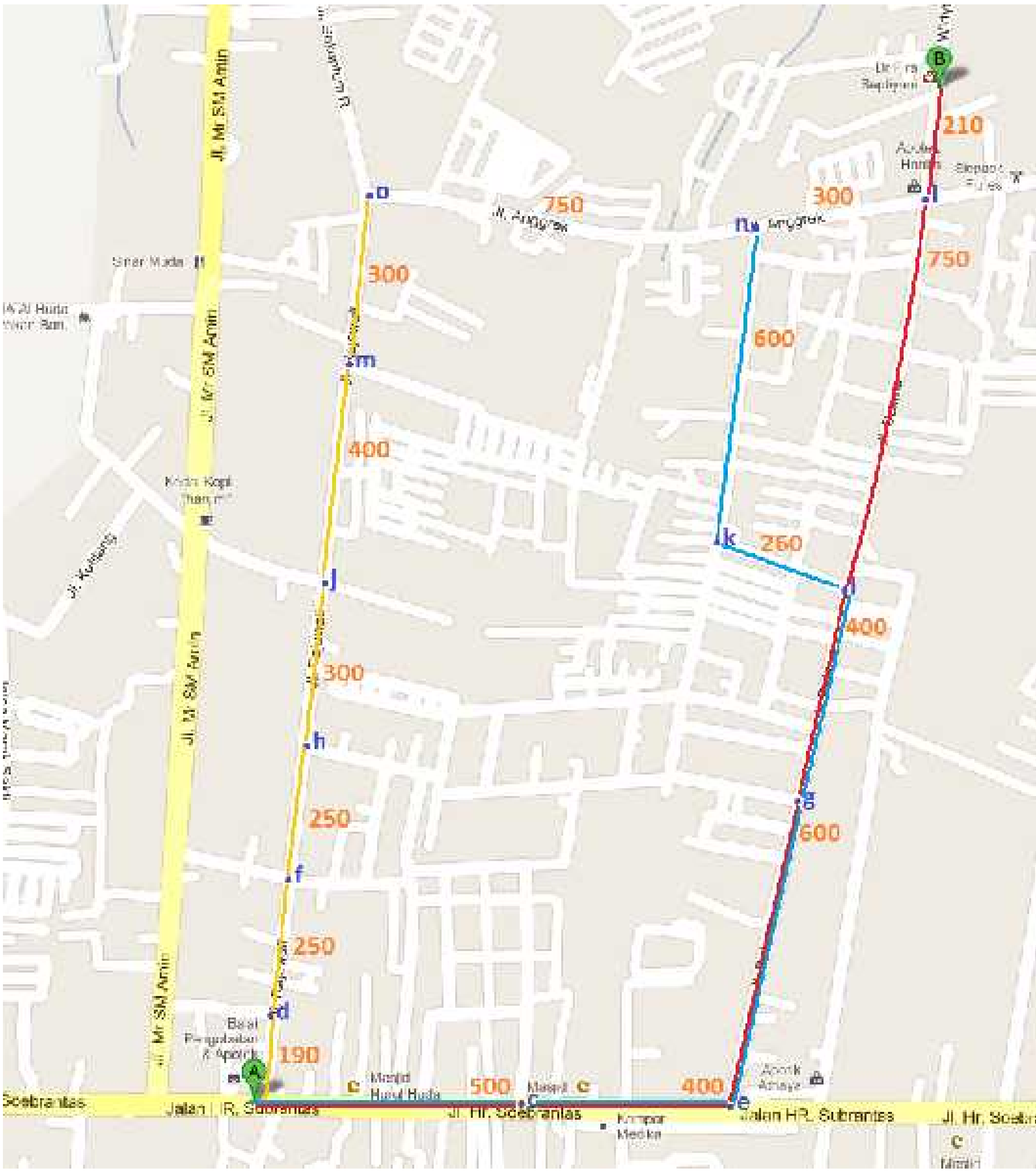
Perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Total jarak yang didapat sampai perhitungan tahap 5, yaitu:
   (A → l) = 2650 m
2. Jarak yang didapat pada perhitungan tahap 6, yaitu: **l → B = 210 m**
3. Total jarak yang didapat sampai perhitungan tahap 6, yaitu:
   (A → l) + (l → B) = 2650 m + 210 m = 2860 m

Sedangkan perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Total jarak yang didapat sampai perhitungan tahap 5, yaitu:
   (A → m) = 1390 m
2. Jarak yang didapat pada perhitungan tahap 6, yaitu: **m → o = 300 m**
3. Total jarak yang didapat sampai perhitungan tahap 6, yaitu:
   (A → m) + (m → o) = 1390 m + 300 m = 1690 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 6 pada gambar 4.10.



Gambar 4.10. Tahap 6 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 7: $f_7 (s) = min\ x_7 \{cx_7s + f_{7-1} (x_7)\}$

Titik Tujuan	**Titik Asal**			
x7	***f7 (x7, s) = cx7s + f6 (x7)***		**Solusi Optimum**	
s7	***n***	***o***	***f7 (s)***	***x7***
l	3060	-	3060	n
n	2760	2440	2440	o

Keterangan :

Pada tahapan ketujuh ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **n** dan **o**, yang sebelumnya menjadi titik kandidat solusi pada tahap 6. Titik **B** tidak memiliki kandidat solusi karena titik tersebut adalah titik tujuan dan pencarian titik **B** perhitungannya telah berhenti di titik tujuan. Titik **n** dan **o** mempunyai kandidat solusi yaitu titik **l**, dan **n**, yang masing-masing titik memiliki nilai jarak sebagai berikut:

1. Total jarak yang didapat sampai perhitungan tahap 6, yaitu:
   (A → n) = 2760 m
2. Jarak yang didapat pada perhitungan tahap 7, yaitu: **n → l = 300 m**
3. Total jarak yang didapat sampai perhitungan tahap 7, yaitu:
   (A → n) + (n → l) = 2760 m + 300 m = 3060 m

Sedangkan perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Total jarak yang didapat sampai perhitungan tahap 6, yaitu:
   (A → o) = 1690 m
2. Jarak yang didapat pada perhitungan tahap 7, yaitu: **o → n = 750 m**
3. Total jarak yang didapat sampai perhitungan tahap 7, yaitu:
   (A → o) + (o → n) = 1690 m + 750 m = 2440 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 7 pada gambar 4.11.

Gambar 4.11. Tahap 7 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 8: $f_8(s) = min\ x_8 \{cx_8s + f_{8-1}(x_8)\}$

Titik Tujuan	**Titik Asal**			
x8	***f8 (x8, s) = cx8s + f7 (x8)***		**Solusi Optimum**	
s8	***l***	***n***	***f8 (s)***	***x8***
B	3270	-	3060	l
l	3060	2740	2740	n

Keterangan :

Pada tahapan kedelapan ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **l** dan **n**, yang sebelumnya menjadi titik kandidat solusi pada tahap 7. Dan mempunyai kandidat solusi yaitu titik **B** dan **l**, yang masing-masing titik memiliki nilai jarak sebagai berikut:

1. Total jarak yang didapat sampai perhitungan tahap 7, yaitu:
   (A → l) = 3060 m
2. Jarak yang didapat pada perhitungan tahap 8, yaitu: **l → B = 210 m**
3. Total jarak yang didapat sampai perhitungan tahap 8, yaitu:
   (A → l) + (l → B) = 3060 m + 210 m = 3270 m

Sedangkan perhitungan untuk jarak dengan titik yang lainnya adalah:

1. Total jarak yang didapat sampai perhitungan tahap 7, yaitu:
   (A → n) = 2440 m
2. Jarak yang didapat pada perhitungan tahap 8, yaitu: **n → l = 300 m**
3. Total jarak yang didapat sampai perhitungan tahap 8, yaitu:
   (A → n) + (n → l) = 2440 m + 300 m = 2740 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 8 pada gambar 4.12.

Gambar 4.12. Tahap 8 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Tahap 9: $f_9(s) = min\ x_9 \{cx_9s + f_{9-1}(x_9)\}$

Titik Tujuan	**Titik Asal**		
x9	***f9 (x9, s) = cx9s + f8 (x9)***	**Solusi Optimum**	
s9	***l***	***f9 (s)***	***x9***
B	2950	2950	l

Keterangan :

Pada tahapan kesembilan ini, algoritma *Floyd-Warshall* melakukan proses perhitungan dari titik asal **n**, yang sebelumnya menjadi titik kandidat solusi pada

tahap 8. Dan mempunyai kandidat solusi yaitu titik **B**, yang memiliki nilai jarak sebagai berikut:

1. Total jarak yang didapat sampai perhitungan tahap 8, yaitu:
   (A → l) = 2740 m
2. Jarak yang didapat pada perhitungan tahap 9, yaitu: **l → B = 210 m**
3. Total jarak yang didapat sampai perhitungan tahap 9, yaitu:
   (A → l) + (l → B) = 2740 m + 210 m = 2950 m

Berikut ini akan ditampilkan peta perhitungan rute terpendek sesuai dengan tahap 9 pada gambar 4.13.

Gambar 4.13. Tahap 9 - Pencarian Rute Terpendek Algoritma *Floyd-Warshall*

Dari hasil analisa yang telah dicari, maka didapatkan 3 rute menuju SPBU 14.2826.123 yang berada pada jalan Widya Graha II (Poin B). Berikut ini akan diurutkan kembali 3 rute yang dapat menuju SPBU 14.2826.123 (poin B), beserta dengan jarak dari masing-masing rute yang telah ditelusuri untuk kemudian dapat dibandingkan mana rute terpendek yang didapat menggunakan pencarian algoritma *Floyd-Warshall*.

Rute 1 : **A, c, e, g, i, k, n, l, B = 3270 m**

Rute 2 : **A, c, e, g, i, l, B = 2860 m**

Rute 3 : **A, d, f, h, j, m, o, n, l, B = 2950 m**

Dari ketiga rute tersebut, hasil dari tiap rute kemudian dibandingkan lalu didapatkan rute terpendek pada rute 2 dengan total jarak **2860 m**, jadi untuk menuju SPBU 14.2826.123 (poin B), rute yang didapatkan dengan menggunakan algoritma *Floyd-Warshall* adalah **A, c, e, g, i, l, B.** Berikut ini akan ditampilkan peta rute dari hasil perhitungan rute terpendek yang didapatkan dengan menggunakan algoritma *Floyd-Warshall*, dapat dilihat pada gambar 4.14.

Gambar 4.14. Hasil pencarian rute terpendek algoritma *Floyd-Warshall*

4.1.3 Deskripsi Kebutuhan Sistem

Untuk membangun sebuah sistem yang efisien, kebutuhan sistem merupakan hal yang harus diperhatikan. Mengetahui kebutuhan sistem akan membantu dalam pembangunan sistem.

4.1.3.1 Sistem yang akan dibangun

Untuk kebutuhan sistem yang akan dibangun terdapat dua bagian yaitu dari kebutuhan sistem pada perangkat Android dan *connector* menggunakan bahasa pemrograman PHP sebagai penghubung antara sistem di perangkat Android dengan *dabatase server*.

a. Sistem Pada Perangkat Android

Kebutuhan sistem pada perangkat Android adalah:

1. Bahasa pemrograman yang digunakan adalah *Java*.
2. Saat aplikasi dibuka, halaman yang tersedia adalah halaman *Home*, yang nantinya akan memiliki fitur di antaranya menu berita, lokasi spbu, pengingat, kontak dan *about*. Untuk kembali ke halaman *Home* setelah melakukan penelusuran dari beberapa fitur menu, pengguna dapat melakukan Klik menu *Home* jika ingin melihat fitur menu yang lainnya.
3. Aplikasi ini nantinya lebih ditujukan untuk menampilkan lokasi SPBU yang ada di kota Pekanbaru.
4. Untuk membaca isi berita, pengguna melakukan pemilihan berita yang akan dibaca sesuai daftar berita yang tersedia, kemudian aplikasi Android akan menampilkan berita tersebut. Berita yang ditampilkan hanya berupa teks.
5. Pengguna dapat melihat Pengingat pada menu Pengingat, dengan melakukan Klik menu Pengingat.
6. Pengguna dapat melihat Kontak pada menu Kontak, dengan melakukan Klik menu Kontak.
7. Untuk mengetahui informasi mengenai aplikasi pencarian SPBU dengan rute terpendek dengan menerapkan Algoritma *Floyd-Warshall*, pengguna dapat melihat menu *about*.

b. *Connector*

Connector ini berfungsi sebagai perantara atau penghubung antara sistem yang berjalan di perangkat Android dengan *database*. *Connector* dibangun menggunakan bahasa pemrograman PHP. Pada aplikasi pencarian SPBU Pekanbaru ini *connector* yang digunakan adalah sebagai berikut, yang dapat dilihat pada gambar 4.15:

```
<?php
$host = "localhost";
$user = "root";
$pwd = "";
$databasename = "db_spbu";
$connection = mysql_connect($host, $user, $pwd) or die("Kesalahan Koneksi ...
!!");
mysql_select_db($databasename, $connection) or die("Databasenya Error");
?>
```

Gambar 4.15. *Connector* aplikasi SPBU Pekanbaru

4.1.3.2 Performansi Aplikasi

Aplikasi pencarian SPBU dengan rute terpendek ini merupakan aplikasi yang beroperasi di lingkungan perangkat pintar bersistem operasi Android. Terdapat beberapa keterbatasan yang ditemui pada perangkat ini, sehingga perlu diperhatikan untuk menjadi acuan dalam pengembangan aplikasi ini, yaitu di antaranya:

1. Sumber daya yang terbatas, hingga saat ini perangkat Android yang banyak beredar memiliki kapasitas memori terbatas. Adapun yang tertinggi saat ini adalah 512Mb.
2. Sumber daya baterai yang secara efektif hanya mampu bertahan selama kurang lebih 200 jam dalam keadaan *standby*.
3. Tampilan antar muka dengan pengguna sangat berpengaruh terhadap waktu tunggu aplikasi hingga aplikasi benar-benar siap digunakan, semakin banyak komponen yang digunakan akan semakin lama pula waktu tunggu yang dibutuhkan.

Dari keterbatasan-keterbatasan pada perangkat Android, maka diusulkan beberapa alternatif untuk meningkatkan performa aplikasi terhadap keterbatasan yang ada, di antaranya:

1. Merancang aplikasi yang menggunakan memori seefektif mungkin, sehingga tidak mengganggu siklus operasi Android dan aplikasi lain.
2. Merancang aplikasi dengan pemanfaatan sumber daya seefisien mungkin namun tidak mengurangi fungsi dan performa aplikasi.
3. Merancang aplikasi dengan antarmuka yang sederhana namun tetap menarik dan ramah bagi pengguna.
4. Merancang aplikasi yang memiliki fitur menu yang hanya berkaitan dengan tujuan utama dibuatnya aplikasi pencarian SPBU dengan rute terpendek.

4.1.4 Fungsi Sistem

Secara umum fungsi sistem ada dua bagian yaitu sistem yang akan dibangun dari sisi perangkat Android dan media penghubung.

4.1.4.1 Fungsi Sistem dari Sisi Perangkat Android

Sistem yang akan dibangun dari sisi perangkat Android memiliki fungsi-fungsi sebagai berikut:

1. Menampilkan berita dalam bentuk teks.
2. Menampilkan lokasi SPBU berdasarkan kategori, lokasi pengguna (*realtime*) yang di tampilkan secara bersamaan dengan lokasi-lokasi SPBU yang ada.
3. Menampilkan pengingat.
4. Menampilkan kontak.
5. Menampilkan *about*.

4.1.4.2 Fungsi Media Penghubung

Media penghubung yang akan dibangun memiliki fungsi untuk mengelola koordinat lokasi SPBU dan koordinat jalan yang telah disimpan pada *database*.

4.1.5 Deskripsi Pengguna

Pengguna dari sistem ini adalah pengguna atau masyarakat yang akan diberikan akses penuh terhadap semua fitur dan fungsi yang ada pada aplikasi ini. Untuk lebih jelasnya dapat dilihat pada tabel 4.1. berikut ini.

Tabel 4.1. Tabel deskripsi pengguna.

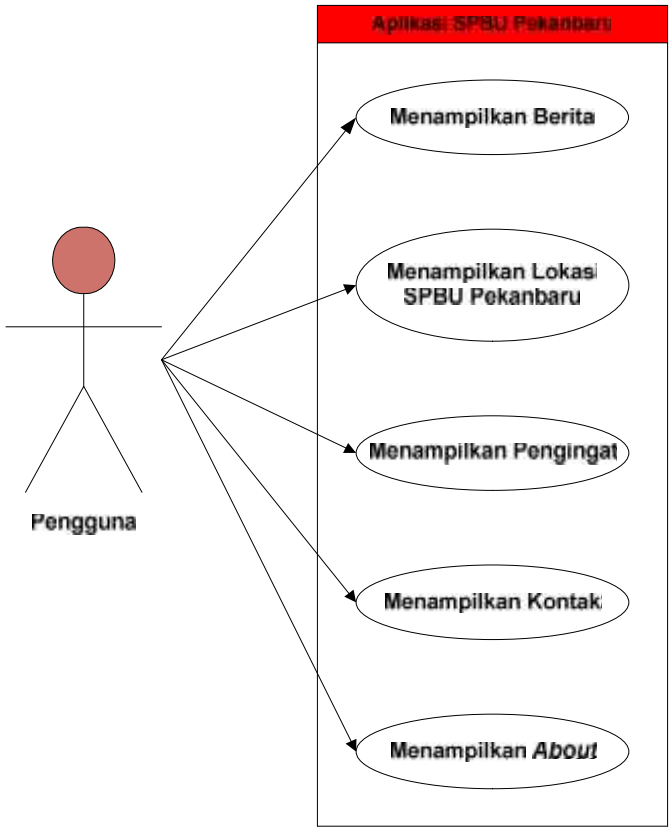
No.	Kategori Pengguna	Hak Akses	Keterangan
1	Pengguna (Perangkat Android)	a. Melihat info lokasi terkini b. Melihat menu berita c. Melakukan pencarian lokasi SPBU dengan rute terpendek d. Melihat menu pengingat e. Melihat menu kontak	Hak akses penuh

4.1.6 *Unified Modeling Language* (UML)

Setelah dilakukan beberapa tahapan dalam analisa sistem, maka dapat dilakukan beberapa analisa terhadap aplikasi yang akan dibangun untuk dirancang selanjutnya sesuai dengan analisa yang telah dilakukan. Perancangan-perancangan yang akan dijelaskan dalam analisa laporan ini meliputi perancangan model dalam bentuk UML *(Unified Modeling Language)* yang terdiri dari *Usecase Diagram*, *Class Diagram*, *Activity Diagram*, dan *Sequence Diagram*. Selain itu juga ada perancangan *interface* sistem yang terdiri dari perancangan *prototype* dan struktur menu yang terdapat pada tahapan perancangan berikutnya.

4.1.6.1 *Usecase Diagram*

Usecase diagram merupakan suatu aktivitas yang menggambarkan urutan interaksi antar satu atau lebih aktor dan sistem. *Usecase* yang akan dirancang yaitu *usecase diagram* untuk pengaksesan melalui perangkat Android. Gambar 4.16 berikut ini menjelaskan aliran *usecase diagram* pengaksesan aplikasi melalui perangkat Android.



Gambar 4.16. Aliran *Usecase diagram* (pengaksesan melalui perangkat Android)

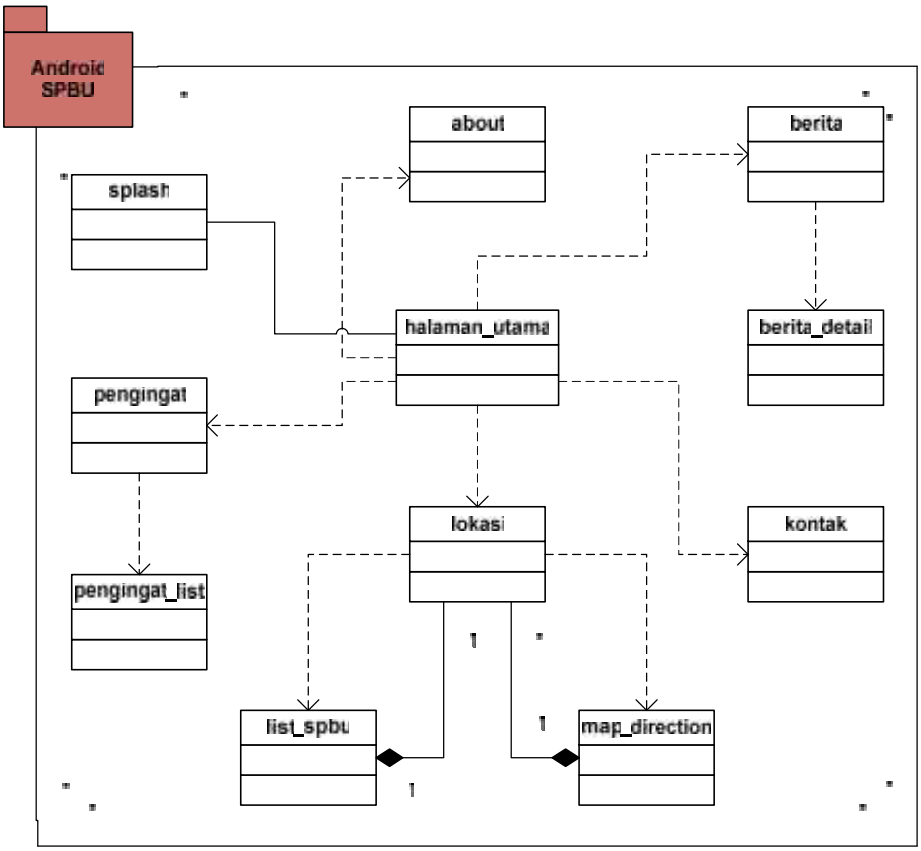
Dari gambar 4.16 dapat dilihat bahwa sistem ini terdiri dari 1 aktor dan 5 *case*. Untuk lebih jelasnya, spesifikasi dari *usecase diagram* (pengaksesan melalui perangkat Android) dapat dilihat pada tabel 4.2 berikut ini.

Tabel 4.2. Spesifikasi *usecase diagram*

No.	**Aktor**	**Nama *Usecase***	**Deskripsi**
1.	Pengguna	Menampilkan berita	Proses menampilkan berita
		Menampilkan lokasi SPBU Pekanbaru	Proses menampilkan lokasi SPBU Pekanbaru berdasarkan *database*
		Menampilkan pengingat	Proses menampilkan pengingat
		Menampilkan kontak	Proses menampilkan kontak Pertamina
		Menampilkan *about*	Proses untuk menampilkan keterangan mengenai aplikasi

4.1.6.2 *Class Diagram*

Class Diagram menggambarkan struktur dan deskripsi *class*, *package*, dan objek yang saling terhubung. *Class diagram* yang dijelaskan pada analisa ini adalah *class diagram* sistem yang terpasang pada perangkat Android. Gambar 4.17 berikut ini menjelaskan *class diagram* sistem yang terpasang pada perangkat Android.



Gambar 4.17. *Class Diagram* sistem pada aplikasi Android SPBU

Keterangan mengenai *class diagram* pada gambar di atas dapat dilihat pada tabel 4.3. berikut:

Tabel 4.3. *Class Diagram* Android SPBU

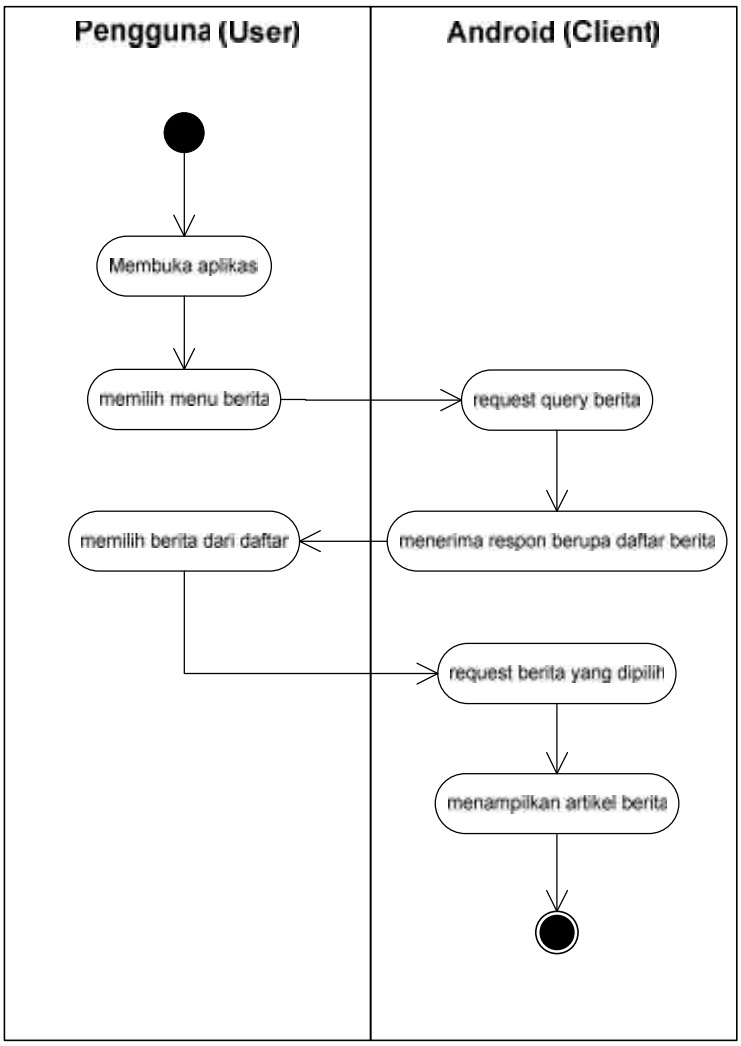
No.	Nama *Class*	Keterangan
1	splash	Merupakan kelas yang menangani tampilan awal aplikasi
2	halaman_utama	Merupakan kelas yang menangani tampilan awal aplikasi yang berisi menu
3	about	Merupakan kelas yang menangani tampilan menu *about*
4	berita	Merupakan kelas yang menangani tampilan menu berita
5	berita_detail	Merupakan kelas yang menangani tampilan berita lebih detil
6	lokasi	Merupakan kelas yang menangani tampilan menu lokasi
7	list_spbu	Merupakan kelas yang menangani tampilan daftar spbu
8	map_direction	Merupakan kelas yang menangani tampilan skema peta
9	pengingat	Merupakan kelas yang menangani tampilan menu pengingat
10	pengingat_list	Merupakan kelas yang menangani tampilan isi pengingat
11	kontak	Merupakan kelas yang menangani tampilan menu kontak

Mengenai deskripsi atribut dan *method* dari masing-masing *class* dapat dilihat pada lampiran C.

4.1.6.3 *Activity Diagram*

Activity diagram merupakan alur kerja pada setiap *usecase*. *Activity diagram* pada analisa ini mencakup *activity diagram* setiap *usecase*. Untuk memudahkan dalam perancangan *activity diagram* maka *activity diagram* dalam aplikasi pencarian SPBU dengan rute terpendek ini akan dipecah menjadi beberapa bagian.

Gambar 4.18. berikut ini menjelaskan *activity* untuk menampilkan berita pada aplikasi Android. Untuk *activity diagram* lainnya dapat dilihat pada lampiran C.

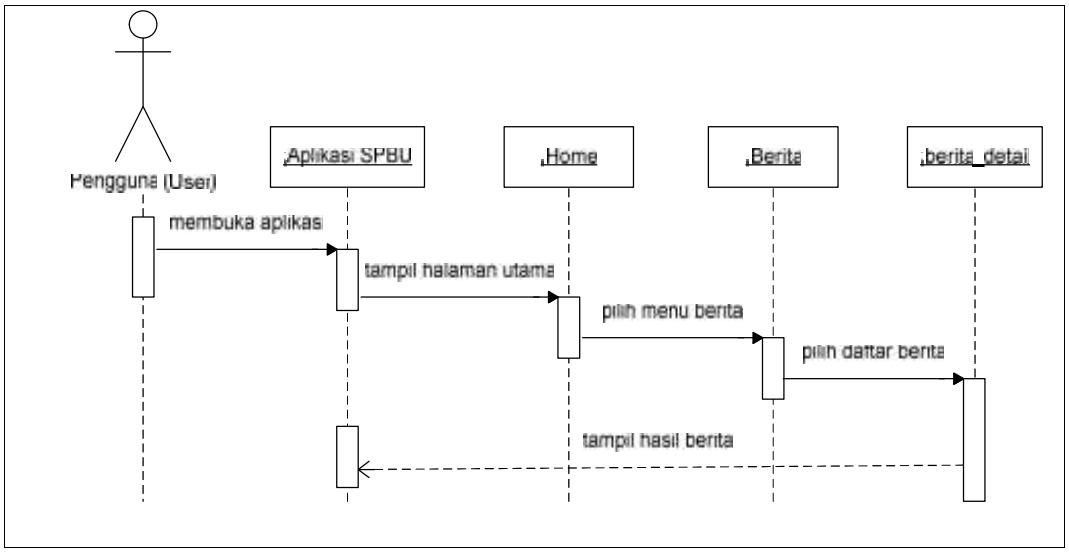


Gambar 4.18. *Activity diagram* menampilkan berita pada perangkat Android

4.1.6.4 *Sequence Diagram*

Sequence diagram adalah representasi dari interaksi-interaksi objek yang berjalan pada sistem. Dengan menggunakan *sequence diagram* kita dapat melihat bagaimana objek-objek bekerja. *Sequence diagram* dapat menampilkan bagaimana sistem merespon setiap kejadian atau permintaan dari user, dapat mempertahankan integritas internal, bagaimana data dipindah ke *user interface* dan bagaimana objek-objek diciptakan dan dimanipulasi.

Gambar 4.19 berikut ini akan menjelaskan mengenai *sequence diagram* untuk menampilkan berita. Penjelasan mengenai *sequence diagram* yang lainnya pada aplikasi android dapat dilihat pada lampiran C.



Gambar 4.19. *Sequence diagram* untuk menampilkan berita

Pada gambar 4.19. tersebut, pengguna memulai untuk membuka aplikasi SPBU dan akan ditampilkan halaman utama yang berisi beberapa menu dari aplikasi SPBU. Proses selanjutnya adalah memilih menu berita, kemudian akan diberikan daftar berita yang tersedia untuk kemudian dipilih dan akan ditampilkan berita sesuai pilihan berita yang dipilih pengguna.

4.2 Tahapan Perancangan

Tahap perancangan merupakan tahapan pencarian solusi dari permasalahan yang dianalisa pada tahap analisa. Perancangan yang akan dibangun meliputi perancangan database, pengkodean yang menerapkan algoritma *Floyd-Warshall* berdasarkan pada analisa UML aplikasi, perancangan struktur menu aplikasi dan perancangan *prototype* antarmuka aplikasi pencarian SPBU.

4.2.1 Perancangan *Database* Aplikasi SPBU

Aplikasi pencarian SPBU merupakan suatu aplikasi yang berbasis *client-server*, serta memerlukan sebuah *database* yang dapat menjalankan fungsi dari aplikasi pencarian SPBU secara optimal. Berikut ini akan diberikan gambaran umum dari *database* yang terdapat pada aplikasi pencarian SPBU.

a. Tabel Berita

Tabel berita memuat *database* yang berisi informasi mengenai berita tentang SPBU ataupun peristiwa-peristiwa yang berkaitan dengan Pertamina dan kawasan beritanya mencakup kota Pekanbaru. Dapat dilihat pada tabel 4.4 berikut.

Tabel 4.4. Deskripsi Tabel Berita

No.	Nama Atribut	*Type*	*Null*	*Size*	*Primary Key*
1	berita_id	*int*	□	5	*yes*
2	subject	*varchar*	□	200	-
3	berita	*text*	□	-	-
4	tgl_berita	*date*	□	-	-
5	galat	*int*	□	5	-

b. Tabel Jenis Kendaraan

Tabel jenis kendaraan memuat *database* yang berisi informasi mengenai jenis kendaraan yang dimiliki, dan akan digunakan kembali sebagai informasi pada menu pengingat pada aplikasi pencarian SPBU. Terdapat pada tabel 4.5 berikut.

Tabel 4.5. Deskripsi Tabel Jenis Kendaraan

No.	Nama Atribut	*Type*	*Null*	*Size*	*Primary Key*
1	jeniskendaraan_id	*int*	□	5	*yes*
2	jenis_kendaraan	*varchar*	□	10	-

c. Tabel Merek

Tabel merek memuat *database* yang berisi informasi mengenai merek dari jenis kendaraan yang dimiliki, dan akan digunakan kembali sebagai informasi pada menu pengingat yang ada pada aplikasi pencarian SPBU.

Tabel 4.6. Deskripsi Tabel Merek

No.	Nama Atribut	*Type*	*Null*	*Size*	*Primary Key*
1	merek_id	*int*	□	5	*yes*
2	merek	*varchar*	□	20	-

d. Tabel Kendaraan

Tabel kendaraan memuat *database* yang berisi informasi mengenai tipe dari merek kendaraan sesuai dengan jenis kendaraan yang dimiliki, dan akan digunakan kembali sebagai informasi pada menu pengingat yang ada pada aplikasi pencarian SPBU. Dapat dilihat pada tabel 4.7 berikut.

Tabel 4.7. Deskripsi Tabel Kendaraan

No.	**Nama Atribut**	***Type***	***Null***	***Size***	***Primary Key***
1	kendaraan_id	*int*	□	5	*yes*
2	nama_kendaraan	*varchar*	□	20	-
3	jenis_bensin	*varchar*	□	20	-
4	rasio_kompresi	*varchar*	□	20	-
5	merek_id	*int*	□	5	-
6	jeniskendaraan_id	*int*	□	5	-

e. Tabel Koordinat

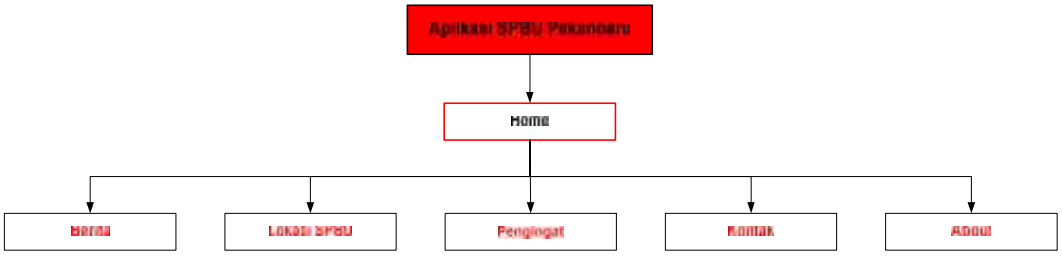
Tabel koordinat memuat informasi mengenai koordinat jalan yang akan dijadikan arah untuk menuju lokasi SPBU, dan merupakan komponen penting bagi aplikasi pencarian SPBU. Dapat dilihat pada tabel 4.8 berikut.

Tabel 4.8. Deskripsi Tabel Koordinat

No.	**Nama Atribut**	***Type***	***Null***	***Size***	***Primary Key***
1	id_koordinat	*int*	□	100	*yes*
2	nama_jalan	*varchar*	□	1000	-
3	titik_awal	*int*	□	100	-
4	titik_akhir	*int*	□	100	-
5	x1	*varchar*	□	1000	-
6	y1	*varchar*	□	1000	-
7	x2	*varchar*	□	1000	-
8	y2	*varchar*	□	1000	-
9	panjang	*varchar*	□	1000	-
10	keterangan	*varchar*	□	1000	-
11	kategori	*varchar*	□	1000	-

4.2.2 Perancangan Struktur Menu Sistem

Rancangan struktur menu merupakan tahapan untuk merancang bagaimana struktur menu yang akan dibangun. Berikut struktur menu dari sistem yang akan dibangun dapat dilihat pada gambar 4.20. berikut ini.



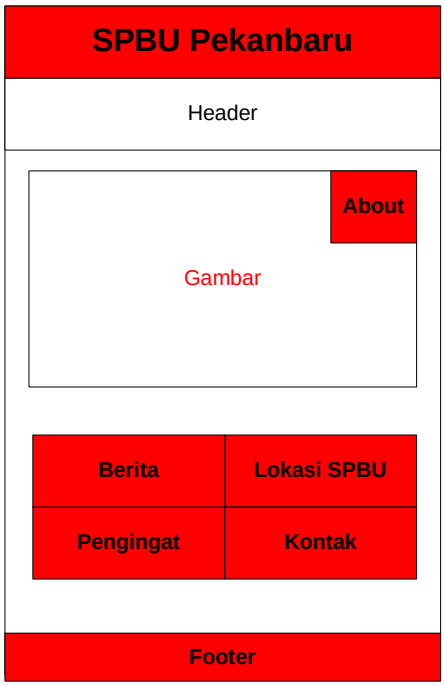
Gambar 4.20. Rancangan Struktur Menu

4.2.3 Perancangan Antarmuka (*Interface*) Pengguna Sistem

Rancangan *interface* pengguna sistem berfungsi sebagai landasan awal dalam merancang tampilan *interface* sistem. Pada analisa dan perancangan ini, *interface* untuk pengguna sistem pada perangkat Android antara lain, *interface Home*, *interface* berita, *interface* lokasi SPBU, *interface* pengingat, *interface* kontak, dan *interface about*. Berikut adalah tampilan *interface* aplikasi SPBU Pekanbaru yang berjalan pada sistem operasi android. Untuk perancangan antarmuka yang lebih lengkapnya dapat dilihat pada Lampiran D.

4.2.3.1 Perancangan Antarmuka (*Interface*) *Home* di Android

Gambar 4.21. berikut ini menjelaskan perancangan *interface Home* aplikasi SPBU Pekanbaru pada perangkat Android.



Gambar 4.21. Perancangan *interface Home* pada perangkat Android

Deskripsi gambar 4.21. tentang perancangan *interface Home* pada perangkat Android dapat dilihat pada tabel 4.9 berikut ini.

Tabel 4.9. Deskripsi *interface home* pada perangkat Android

No.	Nama *Item*	Deskripsi
1	SPBU Pekanbaru	Merupakan sebuah *widget* berupa *TextView*
2	*Header*	Merupakan sebuah *widget* berupa *TextView*
3	Gambar	Merupakan sebuah *widget* berupa *ImageView*
4	Berita	Merupakan sebuah *widget* berupa *Button*
5	Lokasi SPBU	Merupakan sebuah *widget* berupa *Button*
6	Pengingat	Merupakan sebuah *widget* berupa *Button*
7	Kontak	Merupakan sebuah *widget* berupa *Button*
8	*About*	Merupakan sebuah *widget* berupa *ImageButton*
9	*Footer*	Merupakan sebuah *widget* berupa *TextView*

BAB V

IMPLEMENTASI DAN PENGUJIAN

Bab ini merupakan bagian dari tahapan implementasi dan tahapan pemeliharaan, dimana telah dilakukan pengkodean aplikasi, dan akan dilakukan implementasi aplikasi dan pengujian fungsi-fungsi aplikasi dengan metode *Blackbox* sebagai bagian dari tahapan implementasi dan akan dilakukan pengujian aplikasi terhadap pengguna, kemudian akan dilakukan pengamatan dari hasil pengujian tersebut untuk mengetahui kekurangan aplikasi dan kemudian dilakukan pengambilan kesimpulan sebagai bagian dari tahapan pemeliharaan.

5.1 Tahapan Implementasi

5.1.1 Implementasi

Tahapan implementasi merupakan tahapan dimana aplikasi yang telah dirancang, dianalisa, dan dibangun, kemudian diuji kelayakannya untuk selanjutnya dijalankan sebagaimana mestinya sesuai dengan fungsinya. Berikut ini akan dijelaskan tentang implementasi dari analisis dan perancangan yang telah dilakukan terhadap aplikasi pencarian SPBU berbasis *client-server* pada sistem operasi Android menggunakan metode *Floyd-Warshall*.

5.1.1.1 Lingkungan Pengembangan

Komponen-komponen yang dibutuhkan untuk mengembangkan aplikasi ini antara lain berupa komponen perangkat keras dan perangkat lunak.

1. Perangkat keras

   Processor : *Intel(R) Core(TM) i3-2328M CPU @ 2.20 GHz*

   Memori (RAM) : 2 GB

2. Perangkat Lunak

   Sistem Operasi : *Windows 7 Ultimate 32-bit Operating System*

Bahasa Pemrograman : Java dan PHP

Tools Pengembangan : Eclipse Galileo 3.5, Notepad ++

: *Java Development Kit* 6u24 (JDK 6u24)

: Android *SDK, ADT* 8.0

: Android *Virtual Device* 2.2 (Froyo)

: *Google API's 8*

Browser : *Google Chrome 11.0.672.2, Firefox 6.0.2*

Server : XAMPP (Apache 2, MySQL, PhpMyAdmin)

Pemodelan UML : Microsoft Visio

5.1.1.2 Lingkungan Implementasi

Untuk lingkungan implementasi aplikasi ini dilakukan dengan menggunakan perangkat keras dan perangkat lunak di antaranya:

1. Perangkat keras : *Smartphone* Android Samsung Galaxy Ace Plus
2. Perangkat lunak : Sistem operasi Android 2.3.6 (Gingerbread)

5.1.1.3 Langkah-langkah Implementasi

Pada bagian implementasi ini akan dijelaskan bagaimana langkah-langkah yang penulis lakukan dalam implementasi aplikasi yang telah dibangun. Adapun langkah-langkah yang dilakukan adalah:

a. Instalasi Penghubung

Seperti yang telah dijelaskan pada bab analisa dan perancangan, *connector* penghubung mempunyai peran penting dalam aplikasi SPBU Pekanbaru dengan menerapkan algoritma *Floyd-Warshall* berbasis *client-server* pada sistem operasi Android ini. *Connector* ini berperan sebagai penghubung antara aplikasi di Android dengan *database server*.

File connector ini dibangun menggunakan bahasa pemrograman PHP, fungsi dari *file connector* ini adalah untuk meneruskan *query* dan merespon perintah dari *client* ataupun *server*. Untuk lebih jelasnya mengenai isi dari *file connector* tersebut dapat dilihat pada gambar 5.1 berikut.

```php
<?php
$host = "localhost";
$user = "root";
$pwd = "";
$databasename = "db_spbu";
$connection = mysql_connect($host, $user, $pwd) or die("Kesalahan Koneksi ...
!!");
mysql_select_db($databasename, $connection) or die("Databasenya Error");
?>
```

Gambar 5.1. Isi *File Connector* koneksi.php

b. Instalasi Aplikasi SPBU Pekanbaru

Tahap ini merupakan tahap memasang aplikasi SPBU Pekanbaru yang telah dibangun berdasarkan analisa dan perancangan. Aplikasi tersebut dipasang pada perangkat Android. Perangkat Android yang digunakan yaitu *smartphone Samsung Galaxy Ace Plus.* Untuk melakukan instalasi aplikasi, cukup klik aplikasi SPBU yang telah di-*pakcage* ke dalam format *.apk (SPBU.apk) pada perangkat Android dan selesai. Kemudian aplikasi siap untuk dijalankan.

c. Implementasi Unjuk Kerja Pada Perangkat Android

Implementasi kali ini menggunakan perangkat dengan sistem operasi Android Versi *2.3.6* (*Gingerbread*). Hasil implementasi menu lainnya aplikasi SPBU Pekanbaru pada perangkat Android dapat dilihat pada penjelasan berikut.

Gambar 5.2. Menampilkan halaman awal (*Samsung Galaxy* Ace Plus)

1. Menampilkan Halaman Menu Awal

Gambar 5.3 berikut ini menampilkan halaman menu awal ketika membuka aplikasi Pencarian SPBU Pekanbaru pada perangkat Android.

Gambar 5.3. Menampilkan halaman menu awal (Samsung Galaxy Ace Plus)

2. Menampilkan Halaman Menu *About*

Gambar 5.4 berikut ini menampilkan halaman menu *about* pada aplikasi Pencarian SPBU Pekanbaru pada perangkat Android.

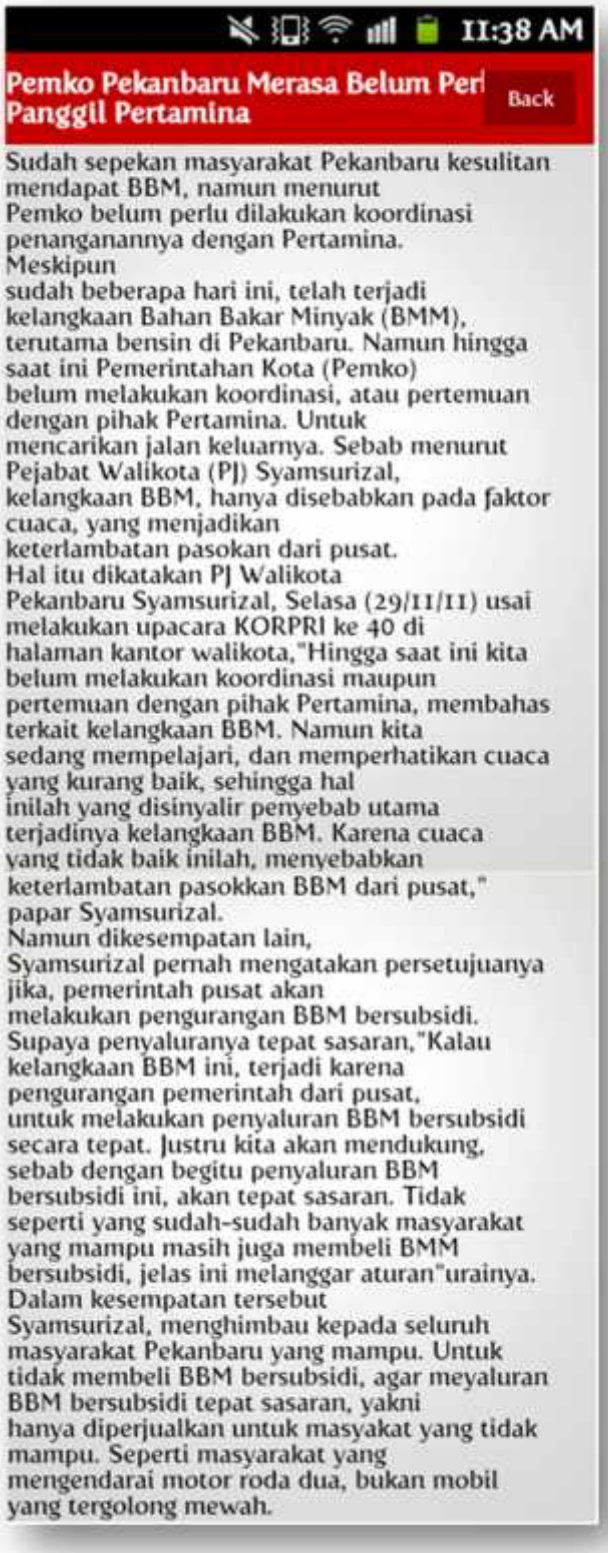
Gambar 5.4. Menampilkan halaman menu *about* (Samsung Galaxy Ace Plus)

3. Menampilkan Halaman Menu Berita

Gambar 5.5 berikut ini menampilkan halaman menu berita pada aplikasi Pencarian SPBU Pekanbaru pada perangkat Android.

Gambar 5.5. Menampilkan halaman menu berita (Samsung Galaxy Ace Plus)

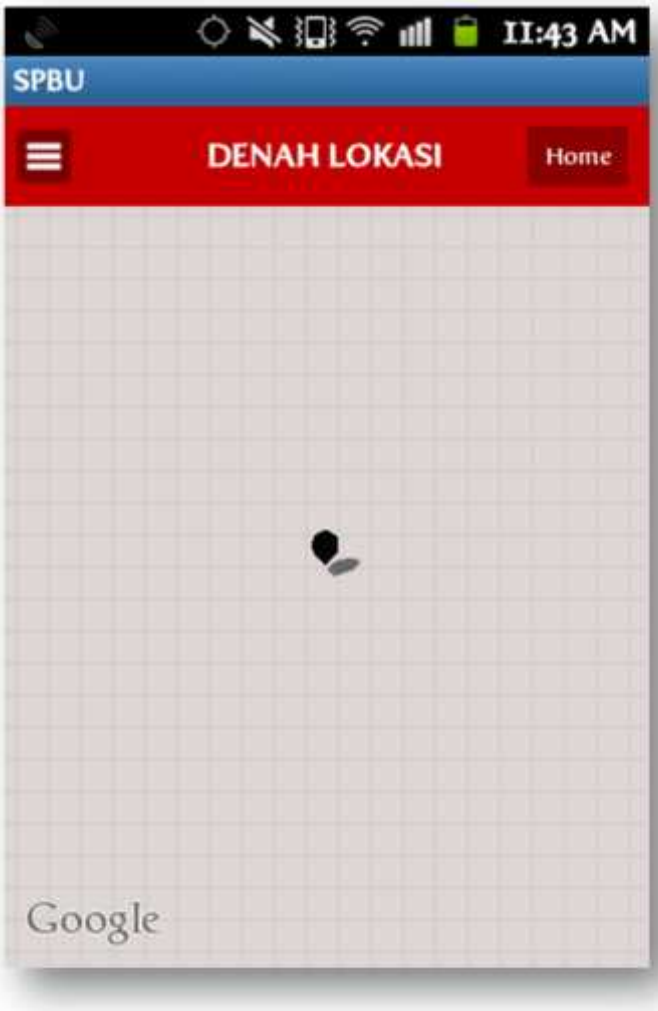
Ketika pengguna memilih urutan berita kedua dari tampilan judul berita di atas, maka aplikasi akan menampilkan detil berita yang dipilih, dapat dilihat pada gambar 5.6. berikut:



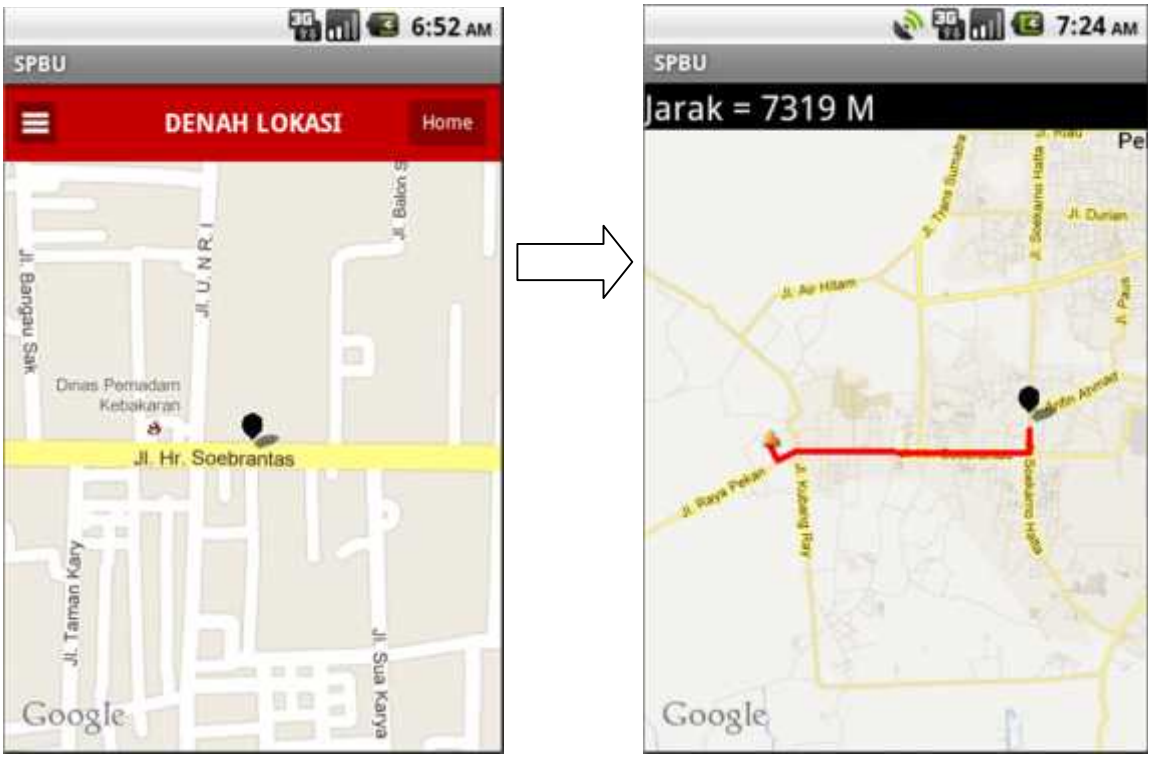
Gambar 5.6. Menampilkan detil dari berita (Samsung Galaxy Ace Plus)

4. Menampilkan Halaman Menu Lokasi SPBU

Gambar 5.7 berikut ini menampilkan halaman menu lokasi SPBU pada aplikasi Pencarian SPBU Pekanbaru pada perangkat Android.



Gambar 5.7. Menampilkan halaman awal menu lokasi SPBU (Samsung Galaxy Ace Plus)



Gambar 5.8. Menampilkan halaman menu lokasi SPBU sebelum (kiri) dan sesudah (kanan) mencari rute terpendek (Samsung Galaxy Ace Plus)

Selain menu lokasi SPBU, aplikasi pencarian SPBU Pekanbaru juga memberikan daftar dari beberapa SPBU yang ada di Pekanbaru. Gambar berikut ini akan menampilkan daftar beberapa SPBU di Pekanbaru.

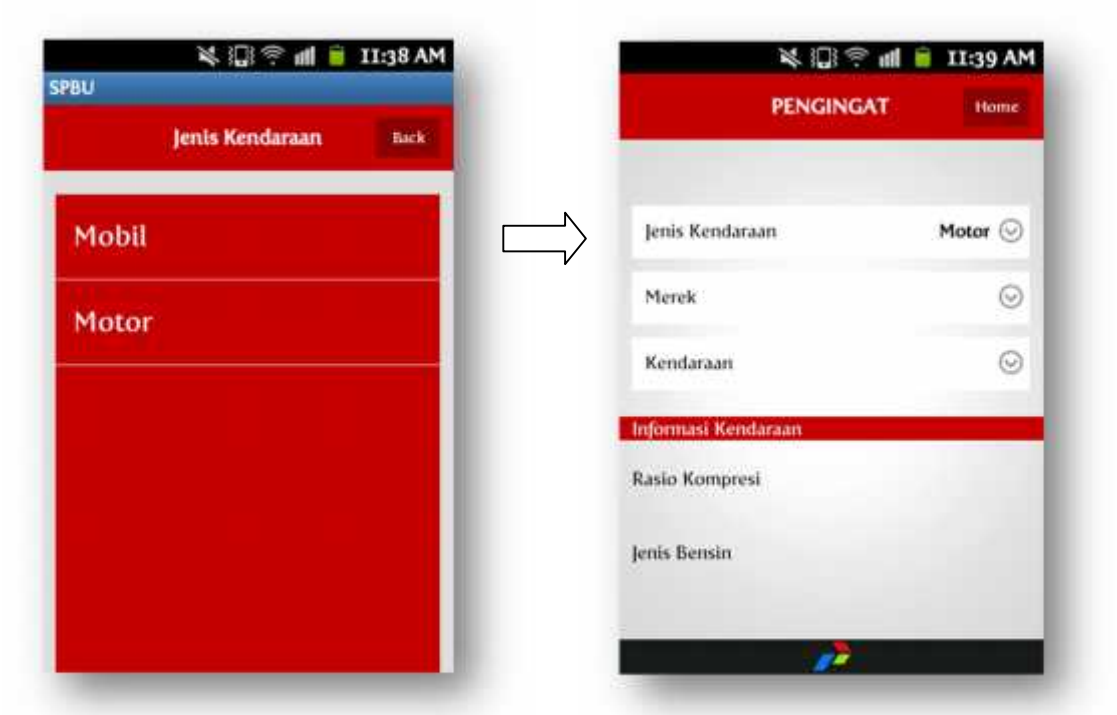
Gambar 5.9. Menampilkan daftar SPBU Pekanbaru (Samsung Galaxy Ace Plus)

5. Menampilkan Halaman Menu Pengingat

Gambar 5.10. berikut ini menampilkan halaman menu pengingat pada aplikasi Pencarian SPBU Pekanbaru pada perangkat Android.

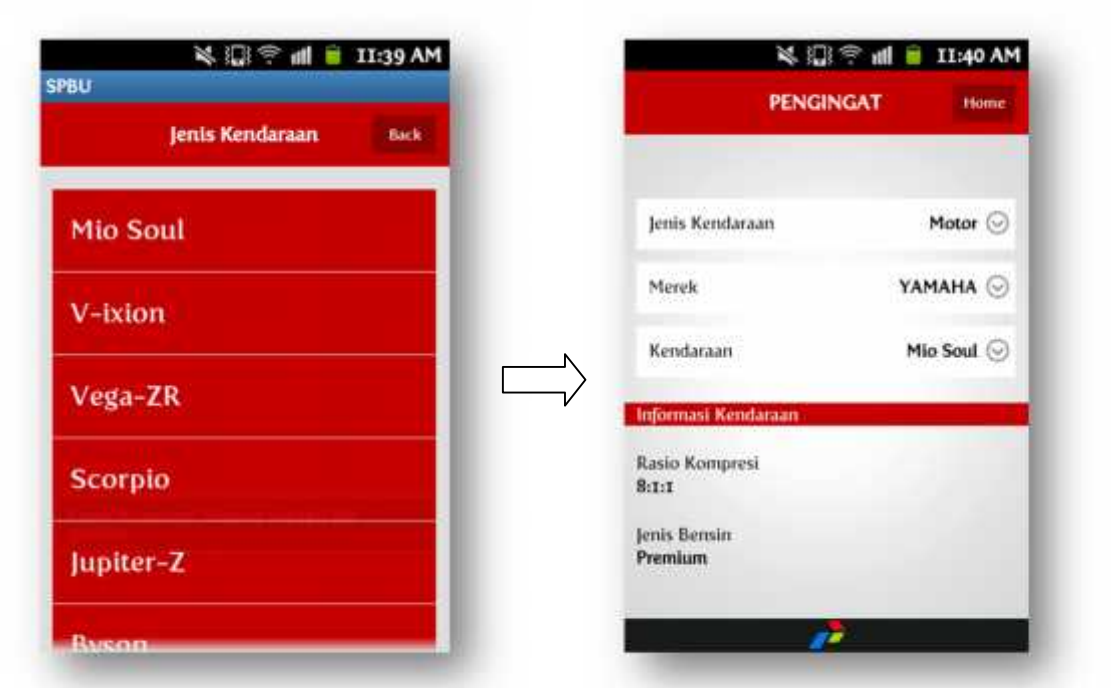
Gambar 5.10. Menampilkan halaman menu pengingat (*Samsung Galaxy Ace Plus*)

Di halaman menu pengingat ini, untuk mengisi form yang bertuliskan langkah 1, 2 dan 3, maka pengguna harus mengisikan terlebih dahulu jenis kendaraan yang dimiliki pengguna, agar aplikasi nantinya dapat memunculkan informasi mengenai data kendaraan yang telah dimasukkan pengguna ke aplikasi. Langkah pengisiannya dapat dimulai terlebih dahulu ketika menyentuh tombol Langkah 1, kemudian berurut sampai pada langkah terakhir, yaitu Langkah 3.



Gambar 5.11. Menampilkan hasil pengisian dari Langkah 1 (Samsung Galaxy Ace Plus)

Gambar 5.12. Menampilkan hasil pengisian dari Langkah 2 (Samsung Galaxy Ace Plus)



Gambar 5.13. Menampilkan hasil pengisian dari Langkah 3 (Samsung Galaxy Ace Plus)

6. Menampilkan Halaman Menu Kontak

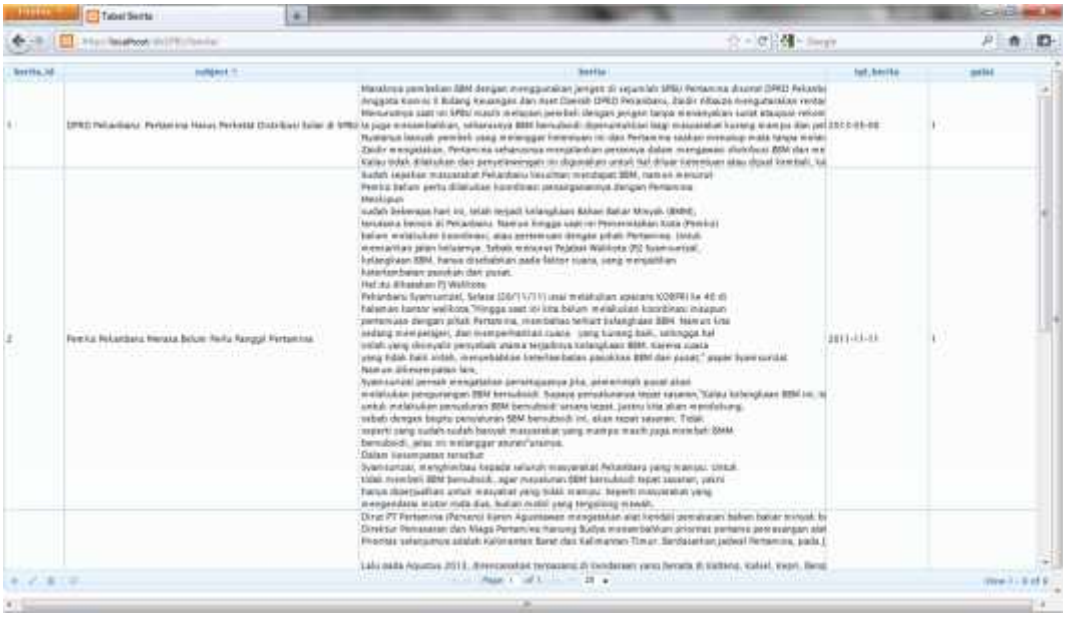
Gambar 5.14 berikut ini menampilkan halaman menu kontak pada aplikasi Pencarian SPBU Pekanbaru pada perangkat Android.

Gambar 5.14. Menampilkan halaman menu kontak (Samsung Galaxy Ace Plus)

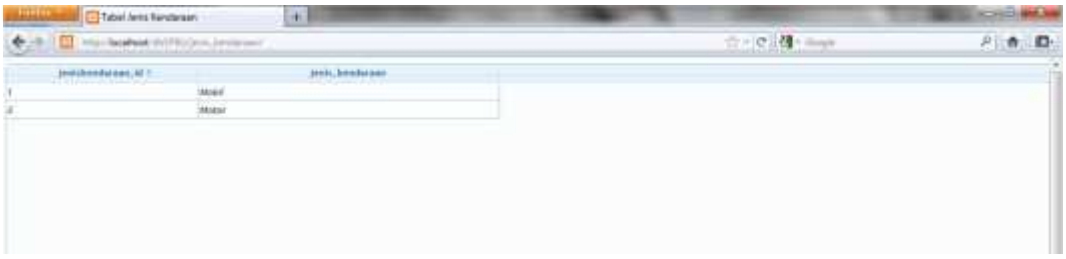
Sistem yang dibangun berjalan dengan baik di perangkat Android, hal ini dilihat dari keberhasilan aplikasi dalam menampilkan menu awal dari *server* dan menampilkannya di perangkat Android.

d. Hasil Implementasi *Interface Database* Aplikasi

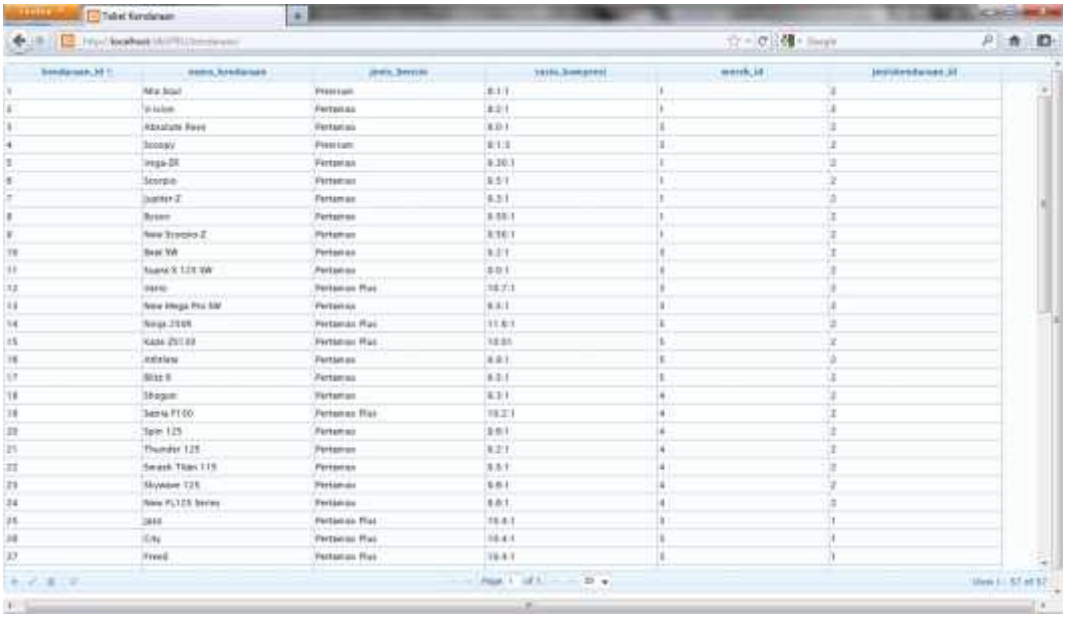
Pada tahap implementasi ini, seluruh *connector* php telah berada di *server* dan siap untuk diakses melalui alamat *url*-nya masing-masing. Berikut gambar dari *interface database* aplikasi SPBU, yang terdiri dari 5 tabel, yaitu : berita, jenis kendaraan, kendaraan, koordinat SPBU dan merek.



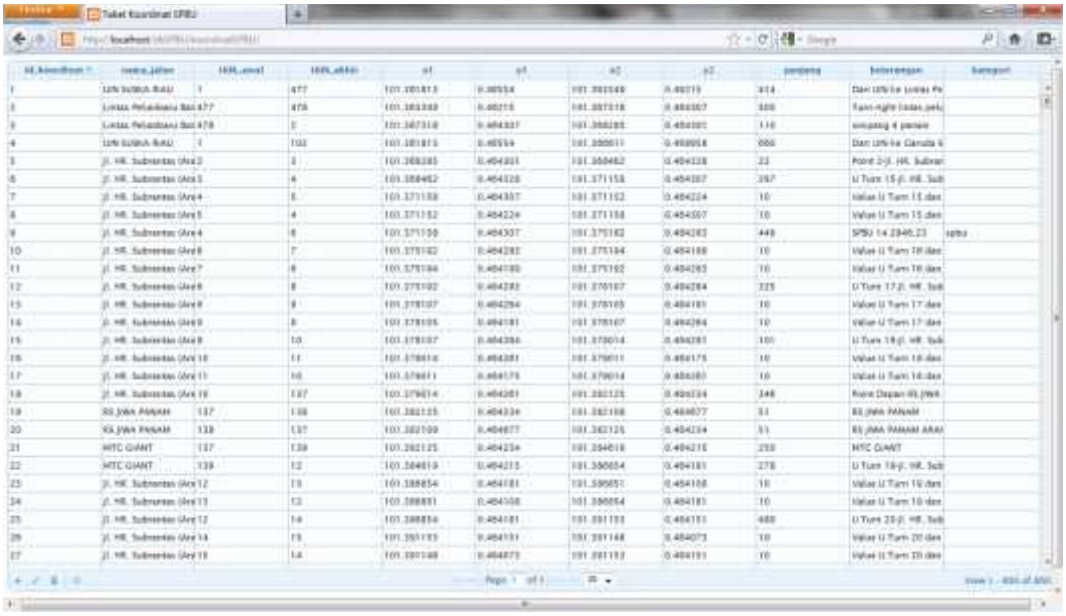
Gambar 5.15. *Interface* tabel berita



Gambar 5.16. *Interface* tabel jenis kendaraan



Gambar 5.17. *Interface* tabel kendaraan



Gambar 5.18. *Interface* tabel koordinat SPBU

Gambar 5.19. *Interface* tabel merek

5.1.2 Pengujian Aplikasi SPBU Pekanbaru

Tahapan pengujian dilakukan untuk mengetahui apakah aplikasi yang dibangun telah sesuai dengan yang diharapkan. Tujuan utama dari pengujian aplikasi adalah untuk memastikan bahwa elemen-elemen atau komponen-komponen dari aplikasi telah berfungsi sesuai dengan yang diharapkan. Salah satu metode pengujian jenis ini dikenal dengan pengujian *blackbox.*

5.1.2.1. Pengujian *Blackbox* Aplikasi SPBU Pekanbaru

Pada tahap pengujian sistem ini, perangkat keras yang digunakan yaitu *smartphone Samsung Galaxy Ace Plus*. Sedangkan material pengujian untuk aplikasi ini menggunakan data koordinat yang telah dimasukan oleh Admin *Database* ke *database* pada *server*. Pengujian yang akan dilakukan adalah pengujian akses pada aplikasi dan pengujian fungsionalitas aplikasi menggunakan metode *blackbox*. Hasil dari pengujian ini dapat dilihat pada tabel 5.1.

Tabel 5.1. Hasil Pengujian *Blackbox*

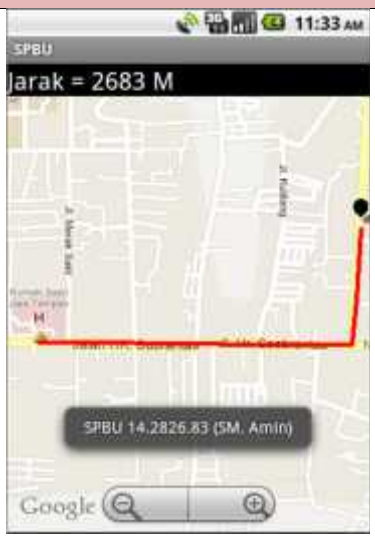
No.	Komponen Pengujian	Hasil yang Diharapkan	Hasil Pengujian Aplikasi	Keterangan
1	Menu Halaman Awal	Pengguna memilih aplikasi SPBU pada *smartphone* , aplikasi menampilkan halaman awal berupa menu-menu utama aplikasi	Halaman awal aplikasi muncul disertai beberapa menu utama dari aplikasi SPBU	Benar
2	Menu Berita	Pengguna memilih menu berita, kemudian muncul beberapa judul berita, untuk dipilih dan akan tampil detil dari judul berita yang telah dipilih	Menu berita berisi judul-judul berita berupa daftar judul, menampilkan detil berita dari judul berita yang dipilih	Benar
3	Menu Lokasi SPBU	Pengguna memilih menu lokasi SPBU, kemudian peta koordinat lokasi pengguna akan muncul pada layar aplikasi	Menu lokasi SPBU tampil dengan memunculkan peta koordinat lokasi pengguna	Benar
4	Memilih 1 SPBU dari daftar SPBU	Pengguna memilih salah satu SPBU dari daftar SPBU yang tersedia pada aplikasi, peta memunculkan koordinat lokasi SPBU yang telah dipilih serta rute terpendek yang akan dilalui	SPBU yang dipilih muncul pada peta bersamaan dengan rute terpendek yang diperoleh dari hasil pencarian menggunakan algoritma *Floyd-Warshall*	Benar
5	Menu Pengingat	Pengguna memilih menu pengingat, aplikasi akan menampilkan halaman yang berisi form untuk mengisi data yang dibutuhkan, aplikasi akan menyimpan data yang dimasukkan pada aplikasi kemudian akan menampilkan kembali informasi mengenai kendaraan yang dimiliki	Menu pengingat menampilkan halaman yang berupa form yang akan diisi datanya oleh pengguna, kemudian menampilkan kembali informasi yang telah diisi oleh pengguna	Benar
6	Menu Kontak	Pengguna memilih menu kontak, kemudian aplikasi akan menampilkan halaman yang berisi info mengenai kontak yang dimiliki oleh Pertamina	Menu kontak menampilkan info mengenai kontak yang dimiliki oleh Pertamina	Benar
7	Menu *About*	Pengguna memilih menu *about* , kemudian aplikasi menampilkan info mengenai aplikasi SPBU Pekanbaru yang sedang digunakan.	Menu *about* menampilkan info mengenai aplikasi SPBU Pekanbaru yang sedang digunakan.	Benar

5.1.2.2. Pengujian Akses Aplikasi SPBU Pekanbaru

Pengujian yang dilakukan bertujuan untuk mengetahui proses hasil dari aplikasi, yaitu memperlihatkan aplikasi SPBU Pekanbaru yang telah dibangun dapat diakses melalui berbagai versi Android. Hasil dari pengujian dapat dilihat apabila halaman awal telah tampil, dan semua menu serta fitur dapat digunakan.

Pada tabel 5.2 berikut ini menjelaskan hasil pengujian akses ke aplikasi SPBU Pekanbaru untuk menemukan jarak terpendek menuju SPBU yang ingin dicari.

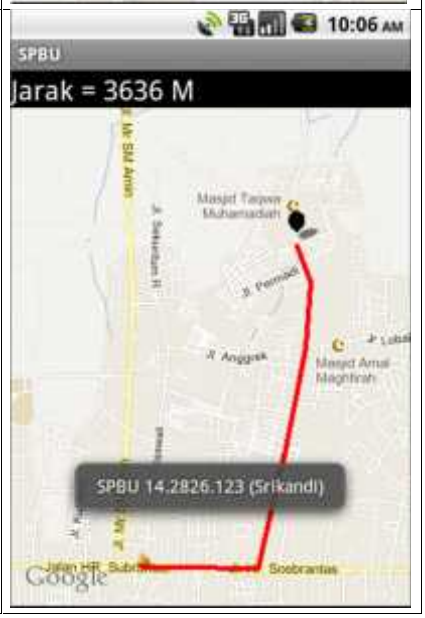
Tabel 5.2. Hasil pengujian akses ke aplikasi SPBU Pekanbaru

Tanggal Jam	Lokasi Awal	Lokasi Tujuan	Provider	Jarak Tempuh (Meter)	Tampilan Aplikasi
19 Juli 2013 11.19 - 11.33 WIB 14 menit	Jl. H.R. Soebrantas 0.464234, 101.382125	SPBU 14.2826.83 Jl. SM. Amin	XL	2683	
20 Juli 2013 10.04 - 10.19 WIB 15 menit	Jl. Jend. Sudirman 0.500129, 101.452986	SPBU 14.2826.21 Jl. Jend. Sudirman	XL	1887	

Tabel 5.2. Hasil pengujian akses ke aplikasi SPBU Pekanbaru (Lanjutan)

Tanggal Jam	Lokasi Awal	Lokasi Tujuan	Provider	Jarak Tempuh (Meter)	Tampilan Aplikasi
20 Juli 2013 09.06 - 09.21 WIB 15 menit	Jl. HR. Soebrantas 0.464006, 101.417819	SPBU 14.2826.35 Jl. Arifin Ahmad	XL	2711	
20 Juli 2013 08.26 - 08.41 WIB 15 menit	Jl. Delima 0.479174, 101.406718	SPBU 14.2826.36 Jl. Soekarno Hatta	XL	2800	

Tabel 5.2. Hasil pengujian akses ke aplikasi SPBU Pekanbaru (Lanjutan)

Tanggal Jam	Lokasi Awal	Lokasi Tujuan	Provider	Jarak Tempuh (Meter)	Tampilan Aplikasi
19 Juli 2013 10.39 – 10.52 WIB 13 menit	Jl. HR. Soebrantas 0.464028, 101.412508	SPBU 14.2826.117 Jl. HR, Soebrantas	XL	1010	
19 Juli 2013 09.48 - 10.03 WIB 15 menit	Jl. HR. Soebrantas 0.464178, 101.395595	SPBU 14.2826.123 Jl. Srikandi	XL	3636	

Sedangkan pada tabel 5.3 berikut adalah hasil pengujian akses aplikasi SPBU berdasarkan pada faktor keadaan cuaca ketika mengakses aplikasi serta penentuan lama waktu yang dibutuhkan berdasarkan jauh dan dekatnya jarak antara pengguna aplikasi dengan SPBU yang dituju.

Tabel 5.3. Hasil pengujian akses ke aplikasi SPBU Pekanbaru berdasarkan faktor cuaca dan jauh dekatnya jarak yang akan ditempuh.

Tanggal Jam	Lokasi Awal	Lokasi Tujuan	Provider	Keterangan	Tampilan Aplikasi
31 Juli 2013 20.41 – 20.55 WIB 14 menit	Jl. Soekarno Hatta 0.480829, 101.418614	SPBU 14.2826.35 Jl. Arifin Ahmad	XL	Berdasarkan jarak 2230m. Cuaca cerah.	
19 Juli 2013 21.06 - 21.25 WIB 19 menit	Jl. Arifin Ahmad 0.476076, 101.427455	SPBU 14.2826.35 Jl. Arifin Ahmad	XL	Berdasarkan jarak 1019m. Cuaca hujan.	

Pada tabel 5.4 berikut ini menjelaskan hasil pengujian akses ke aplikasi SPBU Pekanbaru dengan berbagai perangkat dan sistem operasi android yang telah diujicobakan.

Tabel 5.4. Pengujian Aplikasi Pada Beberapa Perangkat dan Sistem Operasi

Tanggal Jam	Versi Android	Provider	Pengujian	Hasil
15 Mei 2013 10.30 WIB	Frozen Yoghurt (2.2)	XL	Menampilkan menu berita	Berhasil
			Menampilkan menu lokasi SPBU	Berhasil
			Menampilkan menu pengingat	Berhasil
			Menampilkan menu kontak	Berhasil
			Menampilkan menu *about*	Berhasil
20 Mei 2013 08.25 WIB	Gingerbread (2.3.6)	XL	Menampilkan menu berita	Berhasil
			Menampilkan menu lokasi SPBU	Berhasil
			Menampilkan menu pengingat	Berhasil
			Menampilkan menu kontak	Berhasil
			Menampilkan menu *about*	Berhasil
25 Juli 2013 08.47 WIB	Gingerbread (2.3.6)	Smartfren	Menampilkan menu berita	Berhasil
			Menampilkan menu lokasi SPBU	Berhasil
			Menampilkan menu pengingat	Berhasil
			Menampilkan menu kontak	Berhasil
			Menampilkan menu *about*	Berhasil
24 Juli 2013 11.13 WIB	Ice Cream Sandwich (4.0)	Telkomsel	Menampilkan menu berita	Berhasil
			Menampilkan menu lokasi SPBU	Berhasil
			Menampilkan menu pengingat	Berhasil
			Menampilkan menu kontak	Berhasil
			Menampilkan menu *about*	Berhasil
24 Juli 2013 12.00 WIB	Ice Cream Sandwich (4.0)	3	Menampilkan menu berita	Berhasil
			Menampilkan menu lokasi SPBU	Berhasil
			Menampilkan menu pengingat	Berhasil
			Menampilkan menu kontak	Berhasil
			Menampilkan menu *about*	Berhasil
24 Juli 2013 14.00 WIB	Ice Cream Sandwich (4.0)	IM3	Menampilkan menu berita	Berhasil
			Menampilkan menu lokasi SPBU	Berhasil
			Menampilkan menu pengingat	Berhasil
			Menampilkan menu kontak	Berhasil
			Menampilkan menu *about*	Berhasil

5.2 Tahapan Pemeliharaan

Pada tahapan ini, akan ditinjau kembali kekurangan yang terdapat pada aplikasi yang telah dibangun serta akan diberikan kesimpulan dan saran dari pengujian yang telah dilakukan pada tahap implementasi.

5.2.1 Kesimpulan Pengujian

Setelah dilakukan beberapa pengujian terhadap aplikasi yang telah dibangun, maka dapat ditarik kesimpulan dari hasil pengujian tersebut. Berikut kesimpulannya:

1. Aplikasi SPBU Pekanbaru yang dibangun untuk perangkat Android, dapat melakukan koneksi ke *server* dan dapat menampilkan konten sesuai dengan analisa dan perancangan.
2. Aplikasi yang dijalankan di beberapa versi Android yang berbeda, dan berbeda provider dapat berjalan dengan lancar.
3. Daftar lokasi SPBU dan rute terpendek yang ditampilkan di perangkat Android telah sesuai dengan *database* yang berada pada *server*.
4. Semakin banyak data jalan dan data yang memuat lokasi SPBU, maka proses pencarian menggunakan algoritma *Floyd-Warshall* akan semakin lama, karena karakteristik dari pemrograman dinamis yang dimiliki oleh algoritma *Floyd-Warshall*.
5. Aplikasi yang dijalankan di perangkat android ini memiliki beberapa kendala, yaitu:
    a. Kondisi cuaca: kondisi cuaca yang cerah lebih baik dan cepat ketika mengaktifkan GPS untuk mengunci atau menemukan posisi, dan sebaliknya jika kondisi cuacanya dalam keadaan mendung maka GPS akan mengalami kesulitan untuk mengunci atau menemukan posisi.
    b. *Obstackle*/ hambatan, seperti: berada didalam gedung atau ruangan tertutup.
    c. Kualitas sinyal dari masing-masing provider jaringan seluler.
6. Aplikasi ini sangat berpengaruh pada koordinat pengguna. Apabila koordinat pengguna berhasil didapat, maka aplikasi bisa berjalan sebagaimana mestinya. Apabila tidak, maka aplikasi tidak dapat membantu pengguna untuk menentukan rute terpendek.

7. Waktu yang dibutuhkan dalam mengakses aplikasi jika dalam kondisi yang baik, sinyal yang tersedia cukup, cuaca cerah dan tidak mendung serta tidak berada di dalam ruangan maka lama proses pencarian rute terpendek menggunakan algoritma *Floyd-Warshall* lebih kurang membutuhkan waktu 10 menit.

BAB VI

PENUTUP

6.1 Kesimpulan

Setelah menyelesaikan serangkaian tahapan dalam merancang dan membangun aplikasi pencarian SPBU Pekanbaru berbasis *client-server* dengan menggunakan perhitungan algoritma *Floyd-Warshall* pada sistem operasi Android yang dimulai dari pengumpulan data tentang lokasi SPBU yang ada di Pekanbaru dan teknologi Android hingga pada tahapan pengujian, maka dapat diambil beberapa kesimpulan di antaranya adalah sebagai berikut:

1. Aplikasi yang dibangun sudah berjalan pada perangkat Android dan bisa mengakses konten yang berada pada *database* di *server* melalui mesin penghubung, serta berhasil menunjukkan arah rute terpendek menuju lokasi SPBU yang diinginkan.
2. Berbagai *software* (berbagai versi OS android) yang berbeda, dapat menjalankan semua fitur aplikasi dengan baik.
3. Aplikasi SPBU Pekanbaru sudah bisa menampilkan beberapa daftar SPBU yang berada di kota Pekanbaru, serta menemukan rute terpendek dengan menerapkan algoritma *Floyd-Warshall.*
4. Waktu yang diperlukan selama proses pencarian rute terpendek dengan menggunakan algoritma *Floyd-Warshall* berlangsung lebih kurang 10 menit jika keadaan ketika mengakses aplikasi dalam kondisi dengan sinyal provider yang kuat dan terjangkau dengan GPS dan sarana internet yang memadai.
5. Respon untuk menampilkan peta koordinat lokasi tergantung kepada kekuatan sinyal dari provider yang digunakan oleh masing-masing pengguna aplikasi.

6.2 Saran

Beberapa hal yang disarankan dalam pengembangan aplikasi SPBU Pekanbaru dengan menerapkan algoritma *Floyd-Warshall* berbasis *client server* pada sistem operasi Android ini adalah sebagai berikut:

1. Pada pengembangan aplikasi SPBU Pekanbaru selanjutnya diharapkan sudah mampu menampilkan lokasi SPBU yang telah mencakup keseluruhan SPBU yang ada di kota Pekanbaru.
2. Pada pengembangan aplikasi SPBU Pekanbaru selanjutnya diharapkan sudah mampu menampilkan data jalan yang mencakup jenis jalan (jalan utama dan jalan kecil) serta mampu menyesuaikan dengan kondisi lalu lintas jalan yang berada di kota Pekanbaru.
3. Pada pengembangan aplikasi SPBU Pekanbaru selanjutnya diharapkan tidak hanya mampu menampilkan berita yang berupa teks saja, tetapi juga telah mampu menampilkan berita berupa gambar maupun video.
4. Pada pengembangan aplikasi SPBU Pekanbaru selanjutnya diharapkan memiliki lebih banyak fitur lain pada bagian menu aplikasi.
5. Pada pengembangan aplikasi SPBU Pekanbaru selanjutnya diharapkan dapat dioperasikan tidak hanya di *smartphone* android saja, melainkan juga bisa dioperasikan pada *platform* yang lain.

DAFTAR PUSTAKA

Darmawan, Arief. 2006. *Sekilas Tentang Sistem Informasi Geografis (Geographic Information System)*. [Online] available ilmukomputer.com/arifdarmawan-gis, 24 Juni 2006.

Darwiyanti, Sri dan Romi Satria Wahono. *Pengenalan Unified Modeling Language (UML).* [Online] Available http://ilmukomputer.org/2006/ 08/05/pengantar-uml/ 14 Januari 2012 .

Depdikbud. 1988. *Kamus Besar Bahasa Indonesia*. Jakarta: Balai Pustaka.

El-Rabbany, Ahmed. 2002. *Introduction to GPS: the Global Positioning System*. Ernst K & Artech House : mobile communications series. Page 42-64.

Kadir, Abdul. 2003. *Pengenalan Sistem Informasi.* Yogyakarta: Penerbit Andi.

Mufidah, Nur Meita Indah. 2005. *Pengantar GIS (Geographical Information System)*. [Online] available ilmukomputer.com/nurmeita-gis, 11 Juni 2006.

Munir, Rinaldi. 2006. *Algoritma Pemrograman Dalam Bahasa Pascal dan C*. Bandung: Informatika Bandung.

Novandi, Raden Aprian Diaz. 2007. *Perbandingan Algoritma Dijkstra dan Algoritma Floyd-Warshall dalam Penentuan Lintasan Terpendek (Single Pair Shortest Path)*. Bandung: Institut Teknologi Bandung.

pastipas. pertamina.com/lokasi.asp?pastipas=oke, 10 Oktober 2012.

Pertamina. *SPBU*. [Online] available spbu.pertamina.com/spbu.aspx, 25 Juli 2012.

Purnama, Debora. 2010. *Perancangan Program Aplikasi Jarak dan Biaya Optimum Pada Kota-kota di Sumatera dengan Metode Graf dan Floyd-Warshall*. Teknik Informatika, Binus University. Jakarta.

Ramadhan, Fahmi. 2009. *Algoritma Bellman-Ford dan Floyd-Warshall*. [Online] available http://fahmiramadhan.wordpress.com/page/4/, 25 Juli 2012.

Sutrisno, Eko Prasetyo Adi. 2011. *Program Aplikasi GPS dan GIS Untuk Mencari Lokasi dan Jarak SPBU di Tangerang Selatan dengan Peta dan Augmented Reality Camera-ViewPada Perangkat Bergerak Berbasis Android*. Teknik Informatika, Teknik Industri, Universitas Gunadarma.

Tim EMS. 2012. *Panduan Cepat Pemrograman Android*. Jakarta: PT. Elex Media Komputindo.

Wahyu. 2008. *Pengertian GPS.* [Online] available http://gaulwahyu.wordpress. com/2008/10/16/pengertian-gps/, 25 Juli 2012.

Wati, Erma. 2012. *Rancang Bangun Aplikasi Reader Koran Online Berbasis Client-Server Pada Sistem Operasi Android.* Laporan Tugas Akhir Sarjana, Jurusan Teknik Informatika, Universitas Islam Negeri Sultan Syarif Kasim Riau.

Winarno, Edy, dkk. 2011. *Membuat Sendiri Aplikasi Android Untuk Pemula.* Jakarta: PT. Elex Media Komputindo.