OPTIMASI RUTE SALES COVERAGE MENGGUNAKAN ALGORITMA CHEAPEST INSERTION HEURISTIC DAN LAYANAN GOOGLE MAPS API

Erwin Yulianto
Universitas Langlangbuana
rwinyulianto@yahoo.com

Awan Setiawan
Universitas Langlangbuana
awans2425@gmail.com

ABSTRACT

Traveling Salesman Problem (TSP) is one of the distribution issues which is long discussed in the optimization review, where the review problem is how a salesman visits the entire travel route within the coverage area and returns to the starting point of departure with the rule that no destination should be visited more than once. PT. Panjunan is a company engaged in the products distribution of various principles. Currently the managing process of customer location data at the company is still not maximal due to the absence of calculations in every daily visit routes so it impact on the effectiveness of time. In addition, the absence of visualization of real customer data on the field leads to the error distribution and marketing of goods which is uneven, not on target and impact on the decline in sales value. One of the techniques used to accelerate the search solutions of TSP problems is to use an insertion algorithm, or also called the Cheapest Insertion Heuristic (CIH) algorithm.

The discussion of this research is to study and apply CIH algorithm in determining the optimization of daily sales traffic route with the aim of this research is to make daily trip route optimization program by applying Cheapest Insertion Heuristics algorithm and provide real customer data visualization on original map in the field as analysis material for operational team and management by utilizing Google Maps API service. System development method used in this research using waterfall model with implementation tools is laravel framework and mySQL as DBMS.

Keywords : *Route Optimization, Traveling Salesman Problem (TSP), Cheapest Insertion Heuristic (CIH) Algorithm, Google Maps API.*

ABSTRAK

Traveling Salesman Problem (TSP) adalah salah satu masalah distribusi yang cukup lama dibahas dalam kajian optimasi, dimana masalah yang kaji adalah bagaimana seorang *salesman* mengunjungi seluruh rute perjalanan di dalam *coverage area* dan kembali ke tempat awal keberangkatan dengan aturan bahwa tidak boleh ada tujuan yang dikunjungi lebih dari satu kali. PT. Panjunan adalah perusahaan yang bergerak di bidang distribusi produk dari berbagai macam *principles.* Saat ini proses pengelolaan data lokasi pelanggan pada perusahaan masih belum maksimal dikarenakan tidak adanya perhitungan dalam setiap rute kunjungan harian sehingga berdampak pada efektifitas waktu. Selain itu, tidak adanya visualisasi data pelanggan secara *real* di lapangan sehingga berujung pada kesalahan pendistribusian dan pemasaran barang yang tidak merata, tidak tepat sasaran serta berdampak pada menurunnya nilai penjualan. Salah satu teknik yang digunakan untuk mempercepat pencarian solusi masalah TSP adalah menggunakan algoritma penyisipan, atau disebut juga dengan algoritma *Cheapest Insertion Heuristic* (CIH).

Pembahasan penelitian ini adalah mengkaji dan menerapkan algoritma CIH dalam penentuan optimasi rute kunjungan harian *sales* dengan tujuan penelitian yaitu membuat

program optimasi rute kunjungan harian dengan menerapkan algoritma *Cheapest Insertion Heuristics* dan memberikan visualisasi data pelanggan secara *real* pada peta asli di lapangan sebagai bahan analisis tim operasional dan manajemen dengan memanfaatkan layanan Google Maps API. Metode pengembangan sistem yang digunakan pada penelitian menggunakan model *waterfall* dengan perangkat implementasi yaitu *framework* Laravel dan mySQL sebagai DBMS.

Kata Kunci : Optimasi Rute, *Traveling Salesman Problem (TSP),* Algoritma *Cheapest Insertion Heuristic (CIH), Google Maps API.*

1. PENDAHULUAN

1.1 Latar Belakang Masalah

Penentuan pencarian rute optimum merupakan aktivitas yang sangat penting yang digunakan dalam berbagai permasalahan seperti pengiriman paket, pengiriman surat kabar atau pos, penentuan rute pendistribusian barang dari satu gudang ke gudang yang lainnya, begitu juga untuk proses kunjungan harian *salesman* ke masing-masing pelanggan. Permasalahan ini disebut dengan *Travelling Salesman Problem (TSP)* yaitu proses pencarian rute optimum suatu perjalanan dalam mencari jarak terpendek untuk mencapai tempat tujuan tertentu. TSP merupakan salah satu masalah dalam pencarian rute terpendek yang dapat dilalui sales untuk mengunjungi beberapa pelanggan tanpa mendatangi pelanggan yang sama lebih dari satu kali.

PT. Panjunan merupakan perusahaan yang bergerak di bidang jasa distribusi produk dari berbagai macam *principles*. Proses distribusi merupakan roda penggerak yang sangat penting bagi kelangsungan perusahaan dimana efektifitas dan tepat sasaran merupakan salah satu tujuan perusahaan. Dengan banyaknya pelanggan yang sudah tergabung dalam perusahaan, proses pencarian rute optimum ini menjadi salah satu masalah utama yang dialami perusahaan terutama untuk *salesman* yang setiap hari harus mengunjungi beberapa pelanggan sesuai dengan data yang ada. Tidak sedikit proses distribusi dan pemasaran yang tidak tepat sasaran sehingga manajemen dinilai kurang efektif dalam setiap proses pemerataan dan pemasaran barang kepada setiap daerah. Pada saat ini proses kunjungan harian sales kepada pelanggan hanya berdasarkan data alamat yang disimpan pada *file* komputer. Meskipun sudah menggunakan komputer, laporan yang menjadi data acuan analisis tim operasional terlihat kurang memenuhi kebutuhan perusahaan, tidak adanya visualisasi data untuk melihat kondisi *real* di lapangan serta tidak adanya perhitungan optimasi rute yang optimum untuk setiap kunjungan salesman menjadi salah satu permasalahan yang sampai saat ini masih dalam tahap perbaikan.

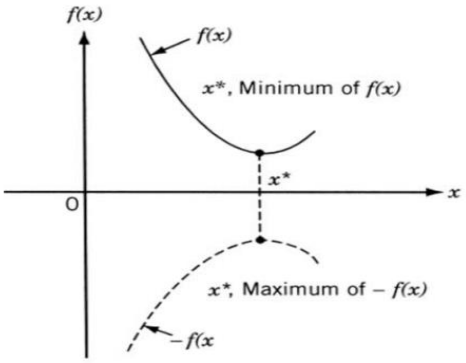
Seiring dengan kemajuan dalam bidang teknologi komputer dan informasi saat ini, GIS (*Geografis Information System*) merupakan teknologi yang terus berkembang, baik untuk aplikasi *desktop* maupun aplikasi berbasis web (*online).* Berdasarkan latar belakang diatas, penulis akan membangun sebuah sistem optimasi rute kunjungan dengan menggunakan metode *Cheapest Insertion Heuristics (CIH)* dan pemanfaatan layanan Google Maps API. Hasil analisa yang dihasilkan dari sistem diharapkan dapat mempermudah perusahaan dalam melakukan proses visualisasi data pelanggan sesuai dengan lokasi masing-masing alamat pelanggan yang sudah didaftarkan pada peta, jumlah pelanggan berdasarkan masing-masing *supplier* serta adanya perhitungan optimasi rute untuk setiap kunjungan harian *sales,* sehingga dapat memudahkan tim operasional perusahaan untuk mengelola dan mengambil keputusan berdasarkan hasil analisa yang ada, dengan efisiensi waktu kunjungan harian *sales* yang sudah diperhitungkan.

1.2 Tinjauan Pustaka

1.2.1 Optimasi

Optimasi (*Optimization*) adalah aktivitas untuk mendapatkan hasil terbaik di bawah keadaan yang diberikan. Tujuan akhir dari semua aktivitas tersebut adalah meminimumkan usaha (*effort*) atau memaksimumkan manfaat (*benefit*) yang diinginkan. Menurut Kamus Besar Bahasa Indonesia (1994), optimalisasi adalah berasal dari kata dasar optimal yang berarti terbaik, tertinggi, paling menguntungkan, menjadikan paling baik, menjadikan paling tinggi, pengoptimalan proses, cara, perbuatan mengoptimalkan (menjadikan paling baik, paling tinggi, dan sebagainya) sehingga optimalisasi adalah suatu tindakan, proses, atau metodologi untuk membuat sesuatu (sebagai sebuah desain, sistem, atau keputusan) menjadi lebih/sepenuhnya sempurna, fungsional, atau lebih efektif.

Mengacu kepada pendapat Rao (2009), optimalisasi dapat didefinisikan sebagai proses untuk mendapatkan keadaan yang memberikan nilai maksimum atau minimum dari suatu fungsi. Hal ini dapat dilihat dari gambar 1, bahwa jika titik x* berkaitan dengan nilai minimum fungsi f(x), titik yang sama juga berkaitan dengan nilai maksimum dari negatif fungsi tersebut –f(x).



Gambar 1 Analogi Optimasi Pada Sebuah Fungsi

Usaha yang diperlukan atau manfaat yang diinginkan dapat dinyatakan sebagai fungsi dari variabel keputusan. Oleh karena itu, optimasi dapat didefinisikan sebagai proses untuk menemukan kondisi yang memberikan nilai minimum atau maksimum dari sebuah fungsi. Optimasi dapat diartikan sebagai aktivitas untuk mendapatkan nilai minimum suatu fungsi karena untuk mendapatkan nilai maksimum suatu fungsi dapat dilakukan dengan mencari minimum dari negatif fungsi yang sama.

1.2.2 *Traveling Salesman Problem*

Permasalahan matematika tentang *Traveling Salesman Problem* (TSP) dikemukakan pada tahun 1800 oleh matematikawan Irlandia bernama William Rowan Hamilton dan matematikawan Inggris bernama Thomas Penyngton. Beberapa definisi *travelling salesman problem* menurut beberapa referensi dapat dilihat sebagai berikut:

1. Suyanto (2017) mengatakan bahwa *Travelling Salesman Problem* dapat difefinisikan sebagai pencarian urutan semua lokasi (misalnya kota) yang harus di kunjungi, mulai dari suatu kota tertentu dan kembali ke kota tersebut, dengan meminimalkan total biaya. Setiap kota harus di kunjungi satu dan hanya satu kali (tidak boleh kurang atau lebih).
2. Apriliani (2011) mengatakan bahwa *TSP* dinyatakan sebagai permasalahan dalam mencari jarak minimal sebuah perjalanan yang berangkat dari sebuah titik awal dan kembali ke titik awal serta mengunjungi setiap titik hanya sekali.

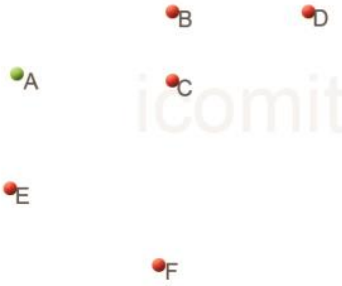
TSP direpresentasikan dengan menggunakan sebuah graf lengkap dan berbobot G = (V, E) dengan V adalah himpunan titik yang merepresentasikan kota-kota yang akan dikunjungi dan E adalah himpunan sisi yang merepresentasikan jalan antar titik serta sisi e = (vi, vj) memiliki bobot c(e) = cij yaitu bobot sisi antara titik vi dan vj yang menujukan jarak antara titik i dan titik j. Diberikan juga C = (cij) sebagai matriks bobot yang bersesuaian dengan E

sehingga permasalahan *TSP* adalah bagaimana menemukan sikel Hamilton dalam graf G dengan bobot minimum.

Traveling Salesman Problem (TSP) adalah salah satu masalah distribusi yang cukup lama dibahas dalam kajian optimasi. Masalahnya adalah bagaimana seorang *salesman* mengunjungi seluruh kota di suatu daerah dan kembali ke kota awal keberangkatan dengan aturan bahwa tidak boleh ada kota yang dikunjungi lebih dari satu kali. Berikut adalah aturan-aturan yang mengidentifikasikan bahwa permasalahan tersebut adalah TSP:

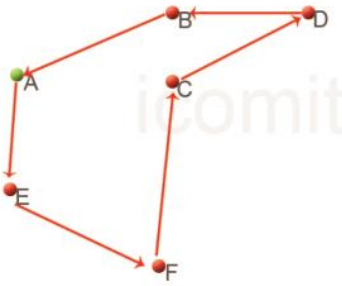
1. Perjalanan dimulai dan diakhiri di kota yang sama sebagai kota asal sales.
2. Seluruh kota harus dikunjungi tanpa satupun kota yang terlewatkan.
3. Salesman tidak boleh kembali ke kota asal sebelum seluruh kota terkunjungi.
4. Tujuan penyelesaian permasalahan ini adalah mencari nilai optimum dengan meminimumkan jarak total rute yang dikunjungi dengan mengatur urutan kota.

Contoh dari permasalahan TSP dapat dilihat sebagai berikut. Seorang *salesman* akan mengawali perjalanannya di kota asal (Kota A) untuk mengunjungi seluruh kota yakni kota A-F sebagaimana gambar 2 di bawah ini.



Gambar 2 Rute *Salesman* Antar Kota (Dalam Titik)

Dari studi kasus tersebut didapatkan salah satu kemungkinan jalur yang paling optimum dengan jalur urutan kota A→E→F→C→D→B→A sebagaimana gambar 3 di bawah. Tentunya hasil tersebut dengan mempertimbangkan jarak dari masing-masing kota hingga menghasilkan kombinasi urutan kota dengan jarak yang optimum.



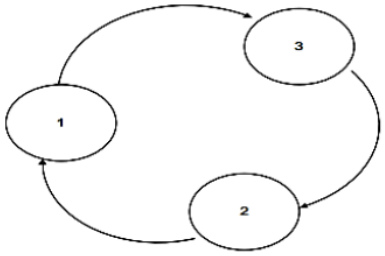
Gambar 3 Rute Perjalanan *Salesman* Antar Kota

1.2.3 Algoritma *Cheapest Insertion Heuristic*

Salah satu teknik yang digunakan untuk mempercepat pencarian solusi masalah *Traveling Salesman Problem* (TSP) adalah dengan teknik *heuristic*. *Heuristic* sering kali memecahkan masalah dengan cukup baik dan lebih cepat dari pencarian solusi secara lengkap karena teknik *heuristic* bertujuan untuk mengurangi jumlah pencarian solusi. Salah satu algoritma dalam *heuristic* yang cukup efektif digunakan untuk menyelesaikan masalah TSP adalah algoritma penyisipan. Dalam algoritma penyisipan terdapat beberapa metode, salah satunya adalah metode *Cheapest Insertion Heuristic (CIH)* yang akan dibahas di dalam penelitian kali ini.

Metode *cheapest insertion heuristic* membangun suatu *tour* dari sikel-sikel kecil dengan bobot minimal dan secara berturut-turut ditambah dengan titik baru. Pemilihan titik baru tersebut dilakukan bersamaan dengan pemilihan sisi sehingga didapatkan nilai penyisipan minimum. Selanjutnya titik baru tersebut disisipkan di antara dua titik yang membentuk sisi yang telah terpilih. Algoritma CIH dikombinasikan dengan basis data yang digunakan sebagai penyimpanan data proses sehingga pengambilan informasi jarak minimal dari beberapa alternatif yang ada dapat dilakukan dengan mudah yaitu dengan menggunakan *query* (Lutfi, 2008). Konsep CIH sendiri memiliki algoritma sebagai berikut : (Wiyanti, 2013)

1. Penelusuran, dimulai dari sebuah kota pertama yang dihubungkan dengan kota terakhir.
2. Pembuatan hubungan *sub tour*, sebuah hubungan subtour dibuat antara 2 kota tersebut. Yang dimaksud dengan *sub tour* adalah perjalanan dari kota pertama dan berakhir di kota pertama, misal (1,3) → (3,2) → (2,1) yang ditunjukan pada gambar 4 di bawah ini.



Gambar 4 *Sub Tour*

3. Mengganti arah hubungan.
   Salah satu arah hubungan (*arc*) dari dua kota dengan kombinasi dua *arc*, yaitu arc(i,j) dengan *arc*(i,k) dan *arc*(k,j), dengan k merupakan kota sisipan dengan tambahan jarak terkecil, yang diperoleh dari:
   C*ik* + C*kj* – C*ij*
   dimana:
   C*ik* adalah jarak dari lokasi *i ke lokasi k*
   C*kj* adalah jarak dari lokasi *k ke lokasi j*
   C*ij* adalah jarak dari lokasi *i ke lokasi j*
4. Ulangi langkah ke 3 sampai seluruh kota masuk ke dalam *sub tour.*

1.2.4 *Google Maps API*

Google Maps merupakan layanan gratis yang disediakan oleh *Google*. *Google Maps* adalah suatu peta dunia yang dapat digunakan untuk melihat suatu daerah dengan menggunakan *browser*. *Google Maps API* adalah suatu *library* yang berbentuk dari *JavaScript*. Cara membuat *Google Maps* untuk ditampilkan pada suatu *web* atau *blog* sangat mudah dan hanya dengan membutuhkan pengetahuan mengenai HTML, *Java Script*, serta koneksi Internet yang sangat stabil. Dengan menggunakan *Google Maps* API, dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga yang difokuskan hanya pada data-data yang akan ditampilkan. Dengan kata lain, kita hanya membuat suatu data sedangkan peta yang akan ditampilkan adalah milik *Google* sehingga tidak dipusingkan dengan membuat peta suatu lokasi. Dalam pembutan program *Google Map API* menggunakan urutan sebagai berikut : (Putra, 2012)

1. Memasukkan Maps API *JavaScript* ke dalam HTML kita.
2. Membuat *element div* dengan nama *map_canvas* untuk menampilkan peta.
3. Membuat beberapa objek literal untuk menyimpan properti-properti pada peta.
4. Menuliskan fungsi *JavaScript* untuk membuat objek peta.
5. Meng-inisialisasi peta dalam *tag body* HTML dengan *event onload*.

Pada *Google Maps API* terdapat 4 jenis pilihan model peta yang disediakan oleh Google, diantaranya adalah:

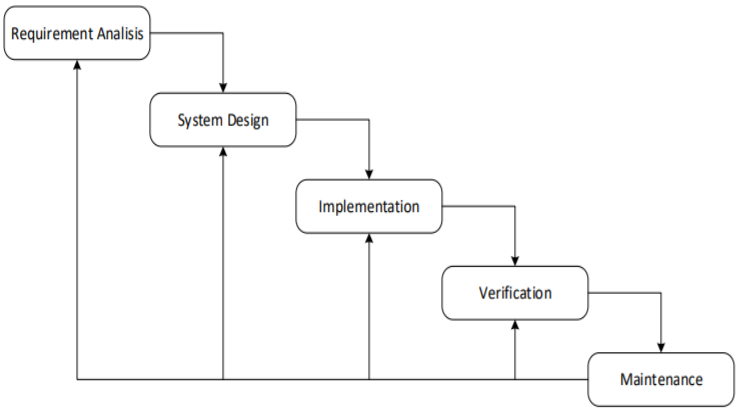
1. *ROADMAP*, untuk menampilkan peta biasa 2 dimensi. Penelitian kali ini menggunakan model peta *Roadmap.*
2. *SATELLITE*, untuk menampilkan foto satelit.
3. *TERRAIN*, untuk menunjukkan relief fisik permukaan bumi dan menunjukkan seberapa tingginya suatu lokasi, contohnya akan menunjukkan gunung dan sungai.
4. *HYBRID*, akan menunjukkan foto satelit yang di atasnya tergambar pula apa yang tampil pada *ROADMAP* (jalan dan nama kota).

Dalam penerapan Google Maps API, terdapat beberapa *syntax* penulisan program yang perlu diketahui. Secara umum, berikut beberapa sintax yang sering digunakan dalam penerapan Google Maps API:

1. google.maps.LatLng, merupakan sintaks yang digunakan untuk menunjuk pada lokasi di peta. LatLng memiliki banyak kegunaan pada *Google Maps API,* contohnya google.maps. *Marker* menggunakan LatLng dalam *constructor* dan meletakkan *marker* penanda pada lokasi geografis di peta. Variabel yang digunakan yaitu:
   var my Latlng = new google. maps.LatLng(my Latitude,my Longitude;
2. google.maps.Map, merupakan *class Java Script* yang merepresentasikan sebuah peta. Objek pada *class* didefinisikan dalam sebuah peta di halaman *website*. Secara sederhana dapat dikatakan bahwa Map() membuat sebuah objek peta dan memasukkannya ke dalam map Div. Variabel yang digunakan yaitu:
   var map = new google.maps.Map (document.getElementBy Id ("map _canvas"), my Options);
3. google.maps.Marker, membuat sebuah *marker* atau penanda pada pilihan tertentu. Bila sebuah peta spesifik, penanda diletakkan pada peta pada saat *construction*. Perhatikan bahwa penanda harus diatur agar *pin point* dapat ditampilkan. Variabel yang digunakan yaitu:
   var marker = new goo gle.map s.M arker(op ts?:MarkerOptions);
4. google.maps.event.addListener, setiap objek *Google Maps API* membuat sejumlah nama *event*. Program yang ingin menggunakan *event* tertentu akan memanggil *event listener* untuk *event* tersebut dan menjalankan kode ketika *event* diterima dengan mendaftarkannya melalui *addListener().* Secara sederhana, addListener() memanggil nama–nama suatu objek dan menjalankannya ketika suatu kejadian terjadi. Variabel yang digunakan yaitu:
   var handle=google.maps.event.addListener(Object,UserEvent, function(event)
   {//Do something});
5. getZoom() dan setZoom(), getZoom() mengembalikan sebuah nilai yang mengidentifikasikan nilai dari *zoom level* yang sekarang. setZoom (zomLevel:number) mengatur tingkat *zoom* peta. Variabel yang digunakan yaitu:
   var zoomLevel= map.getZoom(; map.setZoom(12)

2. METODOLOGI

Metode penelitian yang digunakan oleh penulis adalah metode deskriptif dimana proses pengumpulan data dan kebutuhan pengguna memakai teknik wawancara, studi observasi dan studi pustaka. Adapun metode pengembangan sistem yang digunakan yaitu memakai tahapan siklus hidup perangkat lunak/*software development life cycle (SDLC)* dengan model *waterfall.* Model air terjun/*waterfall*, terkadang disebut siklus hidup klasik, menunjukkan pendekatan, sistematis sekuensial untuk pengembangan perangkat lunak. Adapun tahapan-tahapan aktivitas dari model *waterfall* dapat dilihat pada gambar 5 di bawah ini.



Gambar 5 Model *Waterfall*

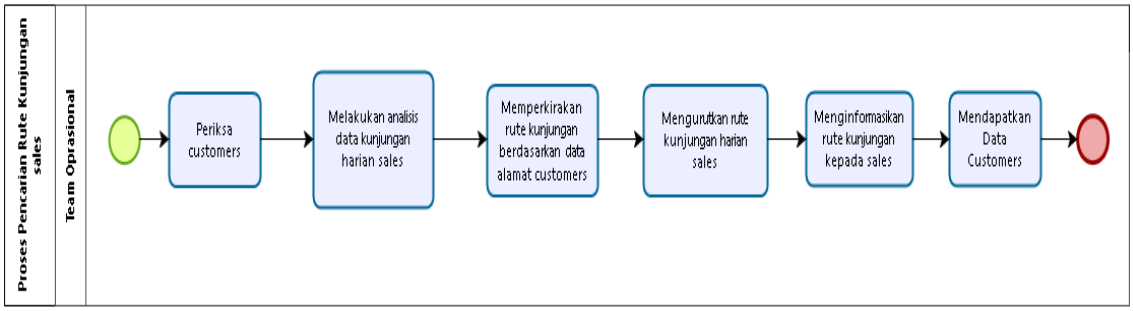
Tahapan dari Model *Waterfall* merefleksikan pokok-pokok dari aktivitas pengembangan perangkat lunak sebagai berikut:

1. *Requirement Analysis*, pada tahap ini penulis melakukan pengembangan sistem komunikasi yang bertujuan untuk memahami perangkat lunak yang diharapkan oleh pengguna dan batasan perangkat lunak tersebut. Informasi ini diperoleh melalui wawancara, diskusi atau survei langsung.
2. *System Design*, spesifikasi kebutuhan dari tahap sebelumnya akan disusun dalam fase ini terkait desain sistem yang akan dikembangkan. Hasil pemetaan pada fase sebelumnya akan digunakan sebagai landasan dalam tahapan perencanaan desain sistem, basis data, antar muka, pemodelan, dan konstruksi *site map*. Desain Sistem akan membantu dalam menentukan perangkat keras (*hardware*) dan mendefinisikan arsitektur sistem secara keseluruhan.
3. *Implementation*, pada tahap ini, langkah pertama adalah mengembangkan program kecil yang disebut unit/modul, yang terintegrasi dengan unit-unit yang lain dalam tahap selanjutnya.
4. *Verification*, setiap unit/modul yang telah dikembangkan akan diuji fungsionalitasnya yang disebut sebagai *unit testing*. Selanjutnya integrasi antar unit akan melalui tahapan *integration testing*, *system testing*, sampai kepada *user acceptable testing*.
5. *Maintenance*, tahap akhir dalam model *waterfall*. Perangkat lunak akan dilakukan pemeliharaan termasuk memperbaiki temuan kesalahan fungsional yang tidak ditemukan pada langkah pengujian sebelumnya. Pada fase ini juga dilakukan peningkatan/*upgrade* sistem sesuai kebutuhan berdasarkan *change for request*.

3. ANALISA DAN PERANCANGAN SISTEM

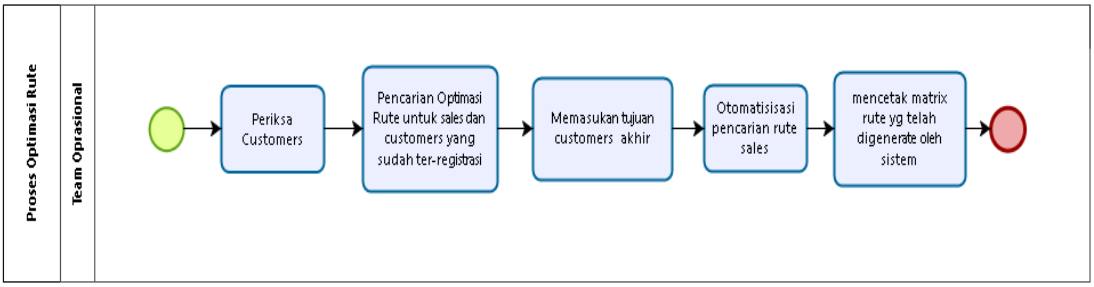
3.1 Proses Bisnis

Berdasarkan tahapan *Requirement Analysis*, proses bisnis pencarian rute kunjungan *sales* yang berjalan sekarang dapat dilihat pada gambar 6 berikut.



Gambar 6 Proses Bisnis Pencarian Rute Kunjungan *Sales*

Selanjutnya gambar 7 di bawah merupakan proses bisnis optimasi rute yang diharapkan oleh PT. Panjunan.



Gambar 7 Proses Bisnis Optimasi Rute

3.2 Formula Perhitungan Algoritma *Cheapest Insertion Heuristic* (CIH)

Sebagai contoh diberikan 5 titik lokasi pelanggan, antara lain PT. Panjunan, Borma TKI SM, PT. Indomart Gatot Subroto, Yogya Riau SM, dan Borma Cibaduyut SM. Pemetaan titik lokasi pelanggan diatas dapat dilihat pada gambar 8 di bawah ini.

Gambar 8 Titik Lokasi Pelanggan

Tahapan-tahapan dalam menentukan rute optimal menggunakan algoritma Algoritma *Cheapest Insertion Heuristic* (CIH) dapat dijelaskan sebagai berikut:

1. Penelusuran, dimulai dari sebuah kota pertama yang dihubungkan dengan kota terakhir, sebagaimana tabel 1 berikut.

Tabel 1 Penelusuran Perhitungan Jarak

NO	TITIK ASAL	TITIK TUJUAN	JARAK ANTAR TITIK (METER)	WAKTU TEMPUH (DETIK)
1	1	2	9732	1046
2	1	3	8582	1729
3	1	4	5806	1098
4	1	5	5417	1047
5	2	1	5904	1157
6	2	3	23929	2463
7	2	4	11710	2255
8	2	5	6834	1518
9	3	1	8439	1643
10	3	2	21195	2143
11	3	4	5662	1150
12	3	5	7231	1494
13	4	1	7346	1383
14	4	2	21251	1835
15	4	3	4437	930
16	4	5	6897	1403
17	5	1	4999	953
18	5	2	5997	1309
19	5	3	7263	1597
20	5	4	7824	1650

2. Pembuatan hubungan *sub tour*, sebuah hubungan *sub tour* dibuat antara 2 kota tersebut. Pada tahapan ini dibuat tabel proses untuk perhitungan selanjutnya dengan tambahan *subtour* pada setiap perhitungan. *Subtour* adalah perjalanan dari titik pertama dan titik akhir pada rute pertama, sehingga membentuk sebuah sikel, yang dapat dilihat pada Tabel 2 Hasil Ke-1 untuk membuat sebuah *subtour* awal yakni (1,5).

Tabel 2 Hasil Ke-1

NO	TITIK ASAL	TITIK TUJUAN	NOMOR RUTE
1	1	5	1

3. Tahap selanjutnya, berdasarkan tabel 2 di implementasikan pada tabel proses dengan melakukan pengulangan untuk menyisipkan titik pelanggan yang belum di hitung, kemudian lakukan pengulangan untuk mengganti titik asal dan titik tujuan. Proses pencarian jarak pada tabel proses dengan menggunakan rumus algoritma *Cheapest Insertion Heuristic* (Cik + Ckj – Cij) dan perhitungan dapat di lihat pada Tabel 3 proses ke-1 di bawah ini.

Tabel 3 Proses Ke-1

NO	cij	cik	ckj	RUMUS (Cik + Ckj - Cij)
	ASAL	SISIP	TUJUAN	JARAK (METER)
1	(1,5)	(1,2)	(2,5)	11149
2	(1,5)	(1,3)	(3,5)	10396
3	**(1,5)**	**(1,4)**	**(4,5)**	**7286**

4. Mengganti arah hubungan; hasil dari perhitungan Tabel 3 Proses Ke-1 menghasilkan total jarak terkecil yakni 7.286 Meter, sehingga *sub tour* lama terganti dengan *sub tour* yang baru, yaitu (1)-(5) diganti menjadi (1)-(4)-(5). Perhitungan selanjutnya adalah memasukan hasil yang ada pada tabel proses, ke tabel hasil dengan menginputkan *sub tour* baru dan memberikan nomor rute, yang dapat dilihat pada tabel 4 Hasil ke-2 di bawah ini:

Tabel 4 Hasil Ke-2

NO	TITIK ASAL	TITIK TUJUAN	NOMOR RUTE
1	1	4	1
2	4	5	2
3	5	1	3

5. Ulangi langkah ke 4 sampai seluruh pelanggan masuk ke dalam *sub tour*, sehingga dapat diperoleh hasil optimasi rute seperti Tabel 5 dan Gambar 9 berikut.

Tabel 5 Hasil Optimasi Rute

Nomor Rute	Rute Asal	Rute Tujuan	Jarak (Meter)	Waktu (Detik)
1	PT. Panjunan	Yogya Riau SM	5.806	1.098
2	Yogya Riau SM	PT. Indomart Gatot Subroto	4.437	930
3	PT. Indomart Gatot Subroto	Borma Cibaduyut SM	7.231	1.494
4	Borma Cibaduyut SM	Borma TKI SM	5.997	1.309
5	Borma TKI SM	PT. Panjunan	5.904	1.157
Total Jarak			**29.375**	
Total Waktu Tempuh			**5.988**	

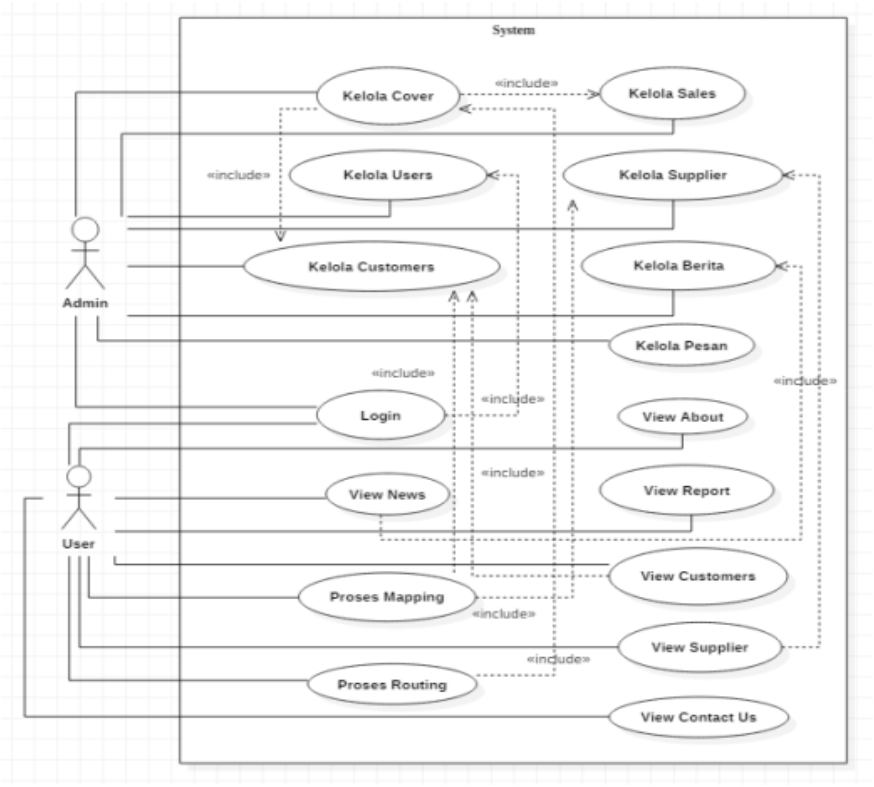
Gambar 9 Hasil Optimasi Rute

3.3 Diagram UML

Pemodelan perancangan perangkat lunak manajemen aset menggunakan *Unified Modelling Language* (*UML*) untuk membantu dalam pendekatan berorientasi objek terhadap perangkat lunak manajemen aset yang akan dibangun.

3.4 *Use Case Diagram*

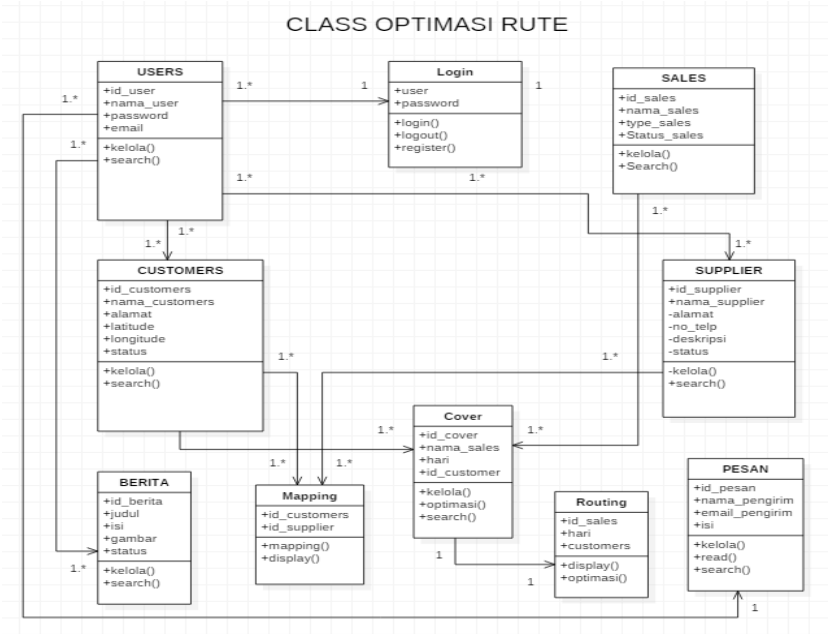
Perancangan *use case diagram* menggambarkan fungsionalitas yang diharapkan dan merupakan hubungan antara aktor dan sistem dari aplikasi yang dikembangkan dapat dilihat pada gambar 10 berikut ini.



Gambar 10 ***Use Case Diagram***

3.5 *Class Diagram*

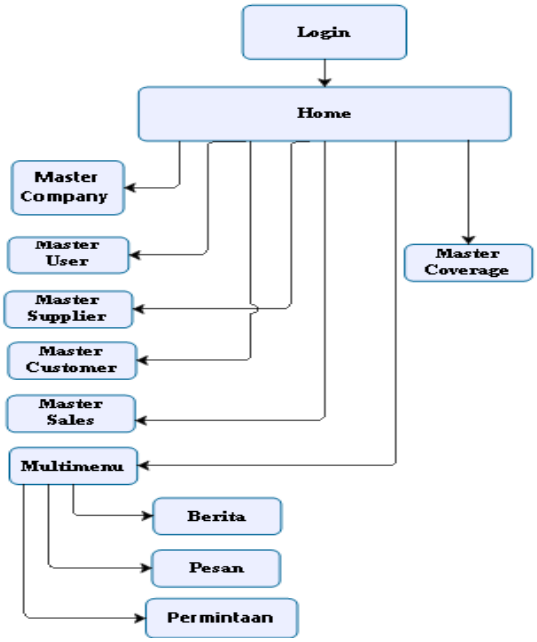
Class diagram ini digunakan untuk menggambarkan kumpulan dari *class* dan hubungannya. *Class* menggambarkan *view*, *model* dan *controller (MVC)* dari suatu sistem, sekaligus layanan untuk memanipulasi keadaan *metode* atau fungsi sehingga *class* memiliki tiga komponen pokok, yaitu nama, atribut, dan metode. *Class diagram* dari aplikasi Optimasi Rute *Sales Coverage* Menggunakan Algoritma *Cheapest Insertion Heuristic* Dan Layanan *Google Maps API* yang akan dikembangkan dapat dilihat pada gambar 11 berikut:



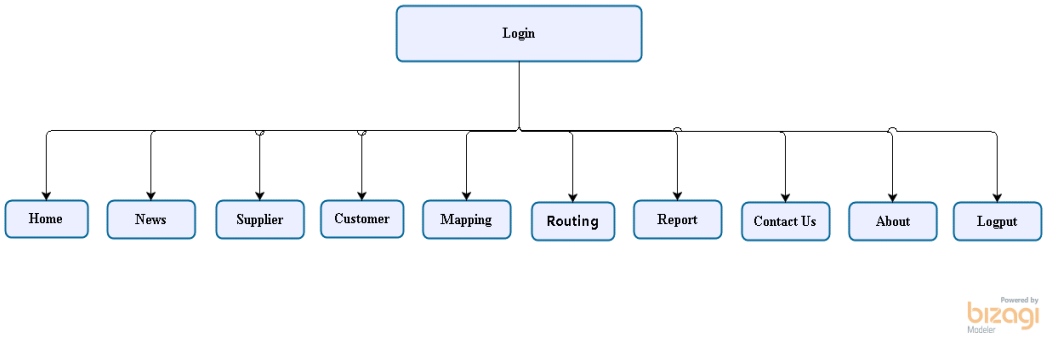
Gambar 11 ***Class Diagram***

3.6 *Site Map*

Desain *sit emap* dipakai sebagai rancangan awal dalam tahap pembangunan aplikasi untuk acuan dalam tahapan *implementasi* antar muka aplikasi. Desain *sitemap* untuk hirarki menu dapat dilihat pada Gambar 12 dan Gambar 13 berikut.



Gambar 12 *Desain Site Map Admin*



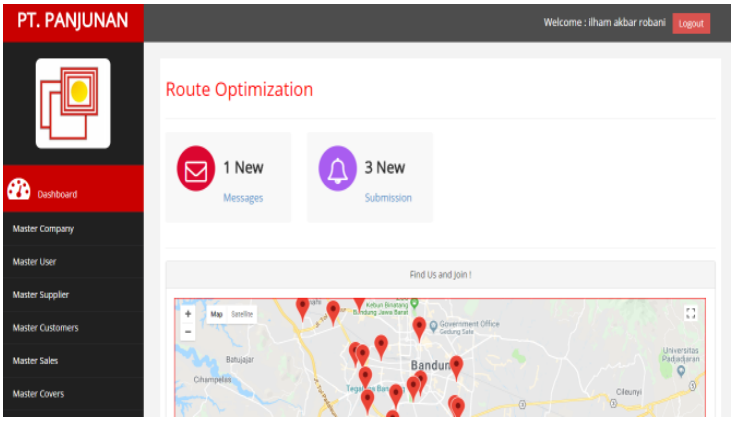
Gambar 13 *Desain Site Map User*

3.7 Implementasi & Pengujian

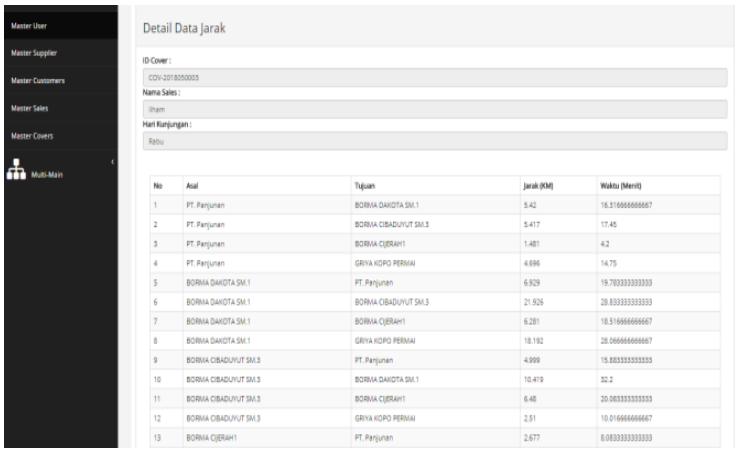
Sistem Informasi Manajemen Pengelolaan Aset Perkantoran yang dibangun berfungsi sebagai *tools* tata kelola aset tetap perusahaan, berikut beberpa fungsi yang dapat dilakukan oleh perangkat lunak ini:

1. Melakukan depresiasi aset menggunakan metode *straight line*.
2. Aplikasi berbasis alur proses bisnis (*work flow)* dan *business control (maker, checker, approver*) sehingga aset terkontrol dengan baik.
3. Memiliki fungsi perawatan aset tetap yang bisa menambah nilai asset setelah depresiasi.
4. Setiap proses pengelolaan manajemen asset tetap terekam di menu riwayat.
5. Memiliki tata kelola dari aset yang dimiliki.

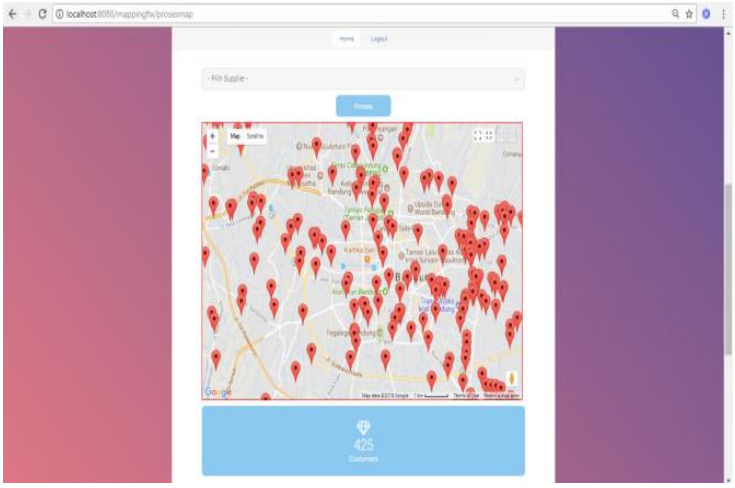
Tampilan antar muka dari fitur-fitur utama aplikasi dapat dilihat pada pada Gambar 14 sampai dengan Gambar 19 berikut.



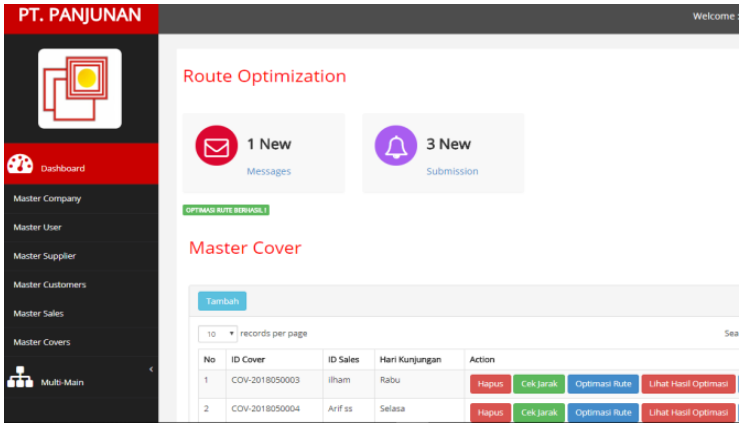
Gambar 14 Tampilan Menu Utama *Admin*



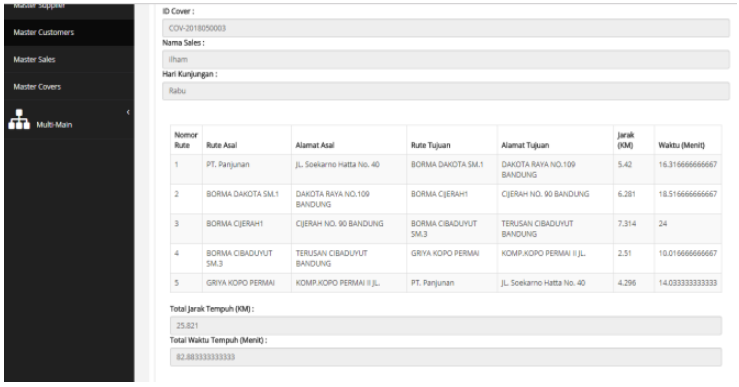
Gambar 15 Tampilan *Form* Cek Jarak *Coverage Area*



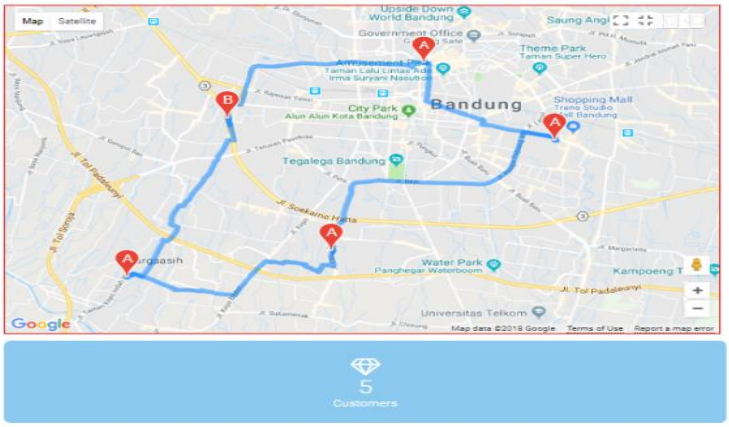
Gambar 16 Tampilan *Form Mapping*



Gambar 17 Tampilan *Form* Optimasi Rute



Gambar 18 Tampilan *Form* Hasil Optimasi



Gambar 19 Tampilan *Form* Hasil Optimasi (*Google Map*)

4. PENUTUP

4.1 Kesimpulan

Berdasarkan analisis yang dilakukan pada penelitian terhadap implementasi aplikasi Optimasi Rute *Sales Coverage* Menggunakan Algoritma *Cheapest Insertion Heuristic* dan Layanan *Google Maps API* dapat ditarik kesimpulan sebagai berikut:

1. Perhitungan Algoritma *Cheapest Insertion Heuristic* (CIH) pada setiap rute kunjungan harian sales, menjadikan proses kunjungan salesman kepada pelanggan menjadi lebih

efektif, karena setiap rute yang dihasilkan merupakan rute optimal yang sudah diperhitungkan sebelumnya.

2. Solusi mampu melakukan pemetaan data, baik secara keseluruhan data pelanggan maupun berdasarkan masing-masing *supplier* dengan penyimpanan data berkapasitas besar yang dapat di akses kapan dan dimana saja, ditambah dengan fungsi visualisasi pemetaan yang berguna untuk mempersingkat waktu pada saat pemasaran barang ke pelanggan oleh *salesman*, dengan adanya penandaan lokasi pelanggan *(customers marker)* yang sudah ter-*mapping* sebelumnya, sehingga dapat mengurangi kesalahan pendistribusian barang.

4.2 Saran

Beberapa saran dan masukan berikut diharapkan dapat memberikan perbaikan dalam penelitian selanjutnya, yaitu:

1. Mengkonsultasikan masalah-masalah baru yang muncul, seiring dengan dengan berkembangnya teknologi Sistem Informasi Geografis (SIG) dan *Google Maps API*, agar aplikasi yang telah di buat dapat mengikuti perkembangan sistem yang baru.
2. Menambakan nomor urutan (*marker)* untuk setiap rute kunjungan harian *sales* pada menu *routing* di halaman *frontend* user, agar proses visualisasi optimasi rute *sales* lebih mudah dipahami.
3. Menambahkan form *import* data pelanggan atau modul otomatisisasi *insert* data dengan format **.txt*, **.csv*, dan *.xlsx sehingga proses integrasi antara sistem dengan aplikasi lainnya dapat dilakukan lebih akurat.

DAFTAR PUSTAKA

Apriliani, Fitri dkk. (2011). Pengaruh *Relationship Marketing* Terhadap Kepuasan Dan Loyalitas Nasabah. Administrasi Bisnis. Vol.17, No. 1.

KBBI (Kamus Besar Bahasa Indonesia). 1994. Jakarta: Balai Pustaka

Lutfi, Ahmad. (2008). Penyelesaian *Traveling Salesman Problem* dengan Menggunakan Metode *Cheapest Insertion Heuristic*. Malang: Universitas Negeri Malang.

Putra, Candra Adi. (2012). Pengantar *Google Maps* API. Yogyakarta: STMIK AKAKOM.

Rao, Singiresu S. (2009). *Engineering Optimization: Theory and Practice. 3rd Edition.* New Jersey: John Wiley and Sons.

Suyanto. (2017). *Swarm Intelligence*: Komputasi Modern untuk Optimasi dan *Big Data Mining*, Bandung : Informatika.

Wiyanti, Dian Tri. (2013). Algoritma Optimasi Untuk Penyelesaian *Traveling Salesman Problem*, Jurnal Transformatika Vol. 11, No. 1, p1-6. Semarang : Universitas Semarang.