Implementasi Algoritma Dijkstra Untuk Mencari Rute Tujuan Wisata Terpendek Dalam Aplikasi Berbasis Web

Wahyu Tisno Atmojo dan Susiana
Institut Sains dan Teknologi Pradita
Jl. Palmerah Barat No.46-48, Jakarta Barat
wahyu.tisno@pradita.ac.id, susiana.budi@gmail.com

ABSTRAK

Perjalanan yang efisien dalam segi jarak, waktu maupun biaya merupakan kebutuhan setiap pelancong. Guna membantu pelancong untuk mendapat rencana perjalanan yang matang, dibutuhkan pemanfataan teknologi agar para pelancong mudah untuk memperoleh rute perlajanan dengan jarak terpendek. Jarak yang lebih kecil dapat mengoptimalkan penggunaan biaya dan mungkin waktu perjalanan. Salah satunya adalah pembuatan aplikasi pencarian pencarian rute tujuan wisata terpendek berbasis website dengan tampilan mobile friendly. Algoritma dijkstra merupakan salah satu algoritma untuk menentukan rute terpendek dari lokasi objek wisata satu menuju objek wisata lainnya. Perhitungan algoritma dijkstra diawalai dengan menetukan node (titik) yang akan dibentuk menjadi sebuah graf. Titik merepresentasikan tujuan wisata pilihan pelancong. Pengujian beta digunakan untuk menguji fungsionalitas fitur aplikasi dan hasil perhitungan rute tujuan wisata terpendek. Hasil pengujian dari segi fungsionalitas fitur aplikasi dan perhitungan rute tujuan wisata terpendek pada pengujian beta fungsi fitur-fitur pada aplikasi bekerja dengan baik. Kecepatan proses perhitungan rute dibandingkan dengan manual mendapat nilai 80% dengan kesimpulan sangat cepat. Sedangkan uji aplikasi dari segi keakuratan rekomendasi rute mendapat hasil 76,7% sehingga dapat dikategorikan akurat sehingga aplikasi baik untuk digunakan.

Kata Kunci : *Perjalanan, Rute Terpendek, Algoritma Dijkstra, Pengujian Beta*

PENDAHULUAN

Berbagai layanan mengenai informasi geografis, lokasi, jalur tempuh, jarak tempuh bahkan hingga tingkat kepadatan jalur tertentu semakin mudah diperoleh seiring dengan perkembangan teknologi. Namun dalam implementasinya terhadap kebutuhan *traveller* (pelancong) masih memiliki beberapa permasalahan, seperti sulitnya menentukan jarak tempuh untuk mencapai lokasi tujuan wisata yang bisa ditempuh. Selain itu permasalah utamanya pelancong harus menentukan jalur yang akan ditempuh secara manual, dimana dirasa masih ada kemungkinan hasil yang ada kurang efisien, efisien dalam hal ini ialah jalur yang ditempuh selama perjalanan memiliki total rute terpendek sehingga dapat membantu mengoptimalkan waktu dan biaya perjalanan. Hasil penentuan rute terpendek akan menjadi pertimbangan dalam pengambilan keputusan untuk menunjukkan jalur yang ditempuh. Dengan menerapkan konsep *Travelling Salesman Problem* (TSP) dengan algoritma dijkstra dalam sebuah aplikasi yang dapat membantu pelancong menetukan jalur perjalanan yang lebih efisien dengan memanfaatkan *location based service google maps* yang mencakup area geografis yang lebih luas. Pengguna dapat memilih banyak destinasi sekaligus mendapatkan rute perjalanan efisien.

Untuk mempermudah pelancong dalam memperoleh urutan rute tujuan wisata serta untuk membantu pelancong memperoleh rencana perjalanan yang efisien baik jarak, waktu maupun biaya maka diperlukan sebuah aplikasi berbasis website yang dapat digunakan oleh pelancong untuk mencari rute terpendek.

Shortest path problem adalah permasalahan dalam menentukan rute diantara 2 (dua) atau lebih *node* dan simpul berdasarkan bobot yang paling kecil. Dari hasil penerapannya dapat digunakan sebagai petujuk rute yang harus ditempuh untuk mendapakan jarak terpendek atau dapat menyelesaikan permasalahan yang digambarkan dengan sebuah graf, robotik, transportasi dan lain-lain. Pencarian rute terpendek (*shortest path problem*) dalam sebuah graf adalah upaya optimasi untuk mendapatkan rute dengan bobot terkecil. "*A graph G = (V, E) is composed by a set of*

vertices V, also called nodes, and a set of edges E, also called arcs" [2]

TSP merupakan sekumpulan node dan bobot yang ditemukan pada setiap pasangan node, digunakan untuk menemukan rute dengan bobot yang paling kecil dan kembali ke titik awal dalam upaya meminimalkan biaya atau jarak perjalanan tanpa harus mengujungi node lebih dari dari satu kali. "*Conceptually, it's very simple: the travelling salesman must visit every city in his territory exactly once and then return home covering the sorthest distance*" [1]

Algoritma secara sederhana merupakan sebuah rumus untuk mendapatkan sebuah solusi sebagai pemecahan masalah. Definisi algoritma menurut seorang pakar dapat diartikan sebagai "Algoritma adalah langkah-langkah yang sistematis, logis, dan lengkap untuk penyelesaian suatu masalah" [4]

Algoritma Dijkstra ditemukan oleh seorang ilmuwan Edsger Dijkstra, adalah salah satu algoritma yang sering digunakan dalam pemecahan persoalan yang berkaitan dengan masalah optimisasi dan bersifat sederhana. Algoritma ini menyelesaikan masalah dalam menentukan sebuah lintasan dari verteks a ke verteks z dalam graf berbobot, bobot tersebut adalah bilangan positif jadi tidak dapat dilalui oleh *node* negatif, namun jika terjadi demikian, maka penyelesaian yang diberikan adalah infiniti.

"*Dijkstra' s algorithm is often known as single source shortest path algorithm. Dijkstra algorithm used the method of increasing node by node to get a shortest path tree which makes the starting point as its root*" Patel dan Chitra Baggar dalam Sahputra, [3]

METODE PENELITIAN

Metode Pengumpulan Data

Metode pengumpulan data yang digunakan ada 2 (dua), diantaranya:

a. Studi Literatur
   Dengan mempelajari buku-buku referensi dan jurnal yang berkaitan dengan permasalahan penelitian yang diangkat serta mencari solusi yang terbaik. Topik bahasan utama yang dibutuhkan adalah pencarian rute terpendek dengan algoritma dijkstra.
b. Observasi
   Suatu metode yang digunakan untuk mencari dan mengumpulkan data yang langsung dari sumbernya dengan cara pengamatan langsung ke tempat objek riset.

Metode Pengembangan Sistem

Pengembangan Sistem menggunakan analisa berorientasi objek dengan metode *waterfall*. Proses pengembangan sistem dilakukan secara berurutan, di mana proses pengerjaan terus mengalir ke bawah (seperti air terjun) melewati fase-fase perencanaan, pemodelan, implementasi (konstruksi), dan pengujian.

HASIL DAN PEMBAHASAN

Analisa Sistem Berjalan

Untuk mengetahui bisnis proses sistem yang sedang berjalan, maka diperlukan analisis sistem dan prosedur yang sedang berjalan. Di dalam analisis sistem terdapat beberapa langkah dasar yaitu sebagai berikut:

a. Mengidentifikasi masalah yang sedang terjadi .
b. Memahami kerja dari sistem yang ada.
c. Analisis sistem yang berjalan.
d. Membuat laporan hasil analisis.

1. Prosedur sistem berjalan

Secara garis besar, proses perhitungan rute perjalanan yang dilakukan pelancong digambarkan seperti berikut:

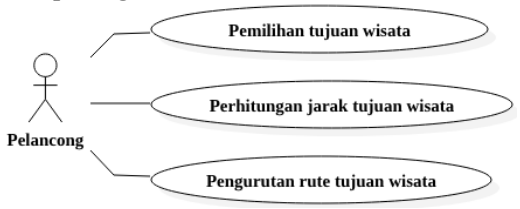
a. Pelancong menentukan tujuan-tujuan wisata yang akan dikunjungi dan mencatatnya dalam bentuk daftar.
b. Pelancong melakukan pencarian jarak tempuh masing-masing lokasi tujuan wisata dari lokasi pelancong berada.
c. Pelancong memilih lokasi tujuan wisata terdekat dari lokasi pelancong berada.
d. Kembali melakukan tahap b dan c hingga tujuan wisata terpenuhi.

1.1. Penggambaran UML Sistem Berjalan

Analisis yang dilakukan dimodelkan dengan menggunakan *Unified Model Language* (UML). Tahap-tahap dalam pemodelan analisis tersebut antara lain *use case diagram* dan *activity diagram*.

a. *Use Case Diagram*

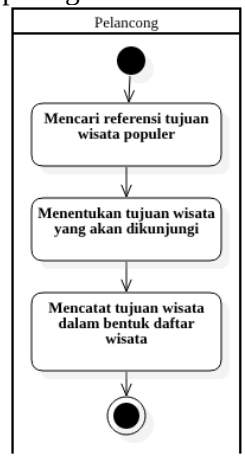
Use case diagram pelancong dalam menentukan rute tujuan wisata suatu perjalanan secara manual dapat dilihat pada gambar 1 berikut ini:



Gambar 1. *Use case system berjalan*

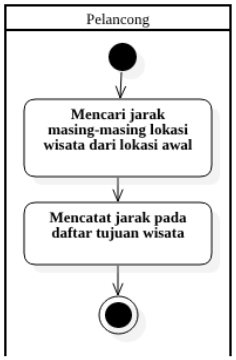
b. *Activity Diagram*

Activity diagram berfungsi memodelkan alur kerja (*workflow*) sebuah proses bisnis dan urutan aktifitas pada suatu proses, dibuat untuk menggambarkan aktifitas aktor. Diagram aktifitas pada *use case* Pemilihan Tujuan Wisata dapat dilihat pada gambar 2 dibawah ini:



Gambar 2. *Activity Diagram pemilihan tujuan wisata*

Diagram aktifitas pada *use case* Perhitungan Jarak Tujuan Wisata dapat dilihat pada gambar 3 dibawah ini:



Gambar 3. *Activity Diagram perhitungan jarak tujuan wisata*.

Diagram aktifitas pada *use case* Pengurutan Rute Tujuan Wisata seperti terlihat dalam gambar 4.

Gambar 4. *Activity Diagram Pengurutan Rute tujuan wisata.*

1.2. Masalah yang ada pada sistem berjalan

Dalam sistem berjalan masih terdapat beberapa permasalahan seperti berikut:

a. Pelancong kesulitan menda-patkan rute perjalanan terpendek dengan melakukan perhitungan manual.
b. Pelancong belum bisa mendapatkan rute perjalanan yang efisien untuk mengunjungi tujuan-tujuan wisata yang diinginkan.

Solusi untuk permasalahan diatas ialah dengan menyediakan fitur perhitungan rute perjalanan otomatis dan sistem rekomendasi rute perjalanan terpendek yang optimal.

2. Implementasi dan Pembahasan

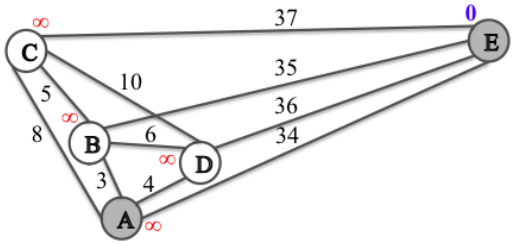
Dalam menentukan rute terpendek yang dapat mencakup semua titik tijuan wisata tanpa ada perulangan titik yang sama menggunakan algoritma dijkstra dapat dilakukan dengan cara menentukan titik mana yang akan menjadi titik awal (beri bobot 0) dan titik akhir, lalu beri bobot jarak

pada titik awal ke titik lainnya satu per satu, dijkstra akan melakukan pengembangan pencarian dari satu titik ke titik lain dan ke titik selanjutnya tahap demi tahap. Berikut ini merupakan langkah-langkah untuk menghitung rute terprndek menggunakan algoritma dijkstra:

A. Beri nilai bobot (jarak) untuk setiap titik ke titik lainnya, lalu beri nilai 0 pada titik awal (titik keberangkatan) dan nilai tak terhingga terhadap titik lain (belum terisi).
B. Dari titik keberangkatan, pertimbangkan titik tetangga yang belum terjamah dan hitung jaraknya dari titik keberangkatan.
C. Saat kita selesai mempertimbangkan setiap jarak terhadap titik tetangga, tandai titik yang telah terjamah sebagai "titik terpilih". Titik terpilih tidak akan pernah dicek kembali, jarak yang disimpan adalah jarak terakhir dan paling minimal bobotnya.
D. Beri nilai "titik belum terpilih" dengan jarak terkecil (dari titik keberangkatan) sebagai "titik keberangkatan" selanjutnya, kembali ke langkah b hingga semua titik terpilih.

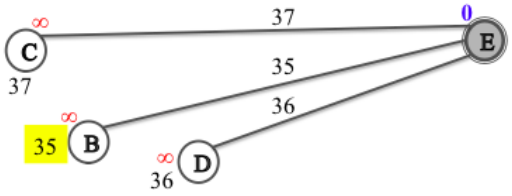
Dengan menggunakan tahapan langkah diatas akan dicari rute tujuan wisata terpendek dengan titik E sebagai titik awal dan A sebagai titik Akhir adalah sebagai berikut:

a. Gambar 5 menunjukan proses pertama yaitu memberi tanda titik E sebagai titik awal (keberangkatan) dan titik A sebagai titik akhir. Beri nilai setiap jalur yang terhubung.



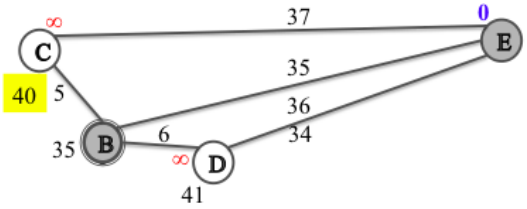
Gambar 5. *Titik E sebagai titik awal dan Titik E sebagai titik Akhir.*

b. Dijkstra melakukan perhitungan terhadap titik tetangga yang terhubung langsung dengan titik keberangkatan (titik E), dan hasil yang didapat adalah titik B karena bobot nilai titik B paling kecil dibandingkan nilai pada titik lain, nilai=35(0+35). Untuk lebih jelas dapat dilihat pada gambar 6.



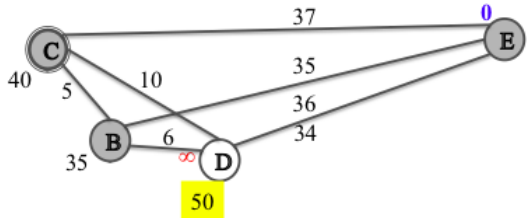
Gambar 6. *Perhitungan terhadap titik tetangga yang terhubung langsung dengan titik keberangkatan.*

c. Titik B diubah menjadi titik keberangkatan dan ditandai sebagai titik yang telah terpilih. Dijkstra melakukan perhitungan kembali terhadap titik-titik tetangga yang belum terpilih dengan titik B sebagai titik keberangkatan. Dari semua titik tetangga belum terjamah yang tehubung langsung dengan titik terpilih, titik selanjutnya yang ditandai menjadi titik terpilih adalah titik C karena nilai bobot yang terkecil, nilai 40 (35+5). Gambar 7 menunjukan proses tersebut.



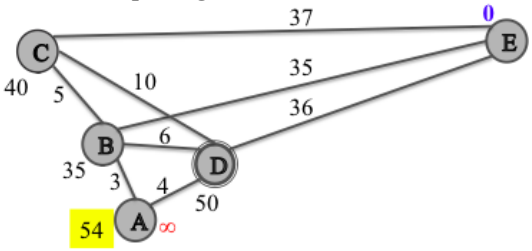
Gambar 7. *Perhitungan kembali terhadap titik-titik tetangga yang belum terpilih dengan titik B.*

d. Titik C diubah menjadi titik keberangkatan dan ditandai sebagai titik yang telah terpilih. Dijkstra melakukan perhitungan kembali terhadap titik-titik tetangga yang belum terpilih dengan titik C sebagai titik keberangkatan.Titik selanjutnya yang ditandai menjadi titik terpilih adalah titik D karena nilai bobot yang terkecil, nilai 50 (40+10), untuk lebih jelasnya dapat dilihat pada gambar 8.



Gambar 8. *Perhitungan kembali terhadap titik-titik tetangga yang belum terpilih dengan titik C*

e. Titik D menjadi titik terpilih, dijkstra melakukan perhitungan kembali, dan menemukan bahwa titik E (titik tujuan akhir) telah tercapai lewat titik D. Jalur lintasan terpendeknya adalah E-B-C-D-A, dan nilai bobot yang didapat adalah 54 (50+4). Bila titik tujuan akhir telah tercapai maka perhitungan dijkstra dinyatakan telah selesai. Proses ini dapat dilihat pada gambar 9.



Gambar 9. *Titik D menjadi titik terpilih.*

Dari langkah-langkah yang ada dapat dibentuk *Pseudocode* dari algoritma dijkstra sebagai berikut:

Procedure dijkstra (masukan m:matriks, a:simpul awal) {mencari lintasan terpendek dari simpul awal a ke semua simpul lainnya. Masukkan: matriks (m) dari graf berbobot G dan simpul awal a. Keluaran: lintasan terpendek dari a ke semua simpul lainnya.}

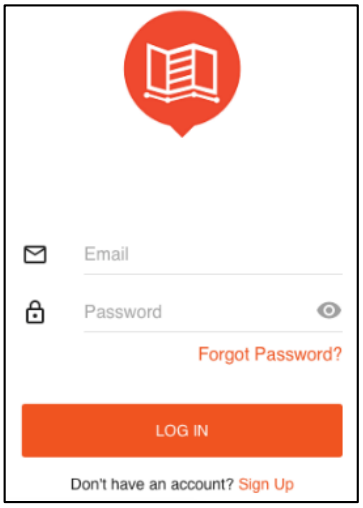
Deklarasi

```
function dijkstra(graph, source);
    for each vertex v in graph:
        dist[v]:=infinity;
        previous[v]:=undefined;
    end for
    dist[source]:=0;
    q:=then set of all graf in graph;
    while q is not empty:
        u:= vertex in q with smallest distance in dist[];
        remove u from q;
        if dist[u] = infinity:
            break;
        end if
        for each neighbor v of u:
            alt:=dist[u] + dist_between(u,v);
            if alt<dist[v]:
                dist[v]:=alt;
                previous[v]:=u;
                decrease-key v in q;
            end if
        end for
    end while
    return dist;
end function
```

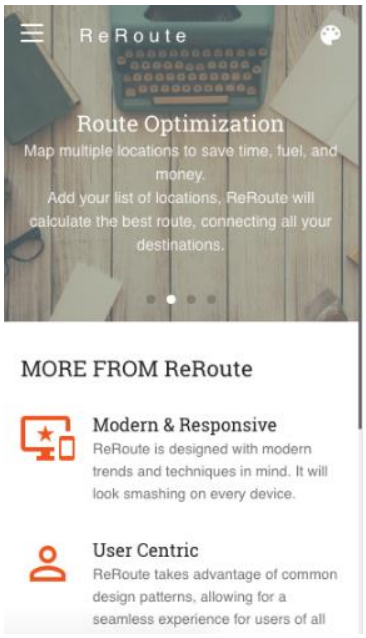
Penerapan algoritma dijkstra kedalam sebuah program aplikasi pencarian rute tujuan wisata terpendek dapat membantu pelancong menentukan rute wisata. Berikut ini merupakan hasil implementasi algoritma dijkstra dengan konsep TSP kedalam sebuah program aplikasi menggunakan bahasa pemrograman PHP *CodeIgniter* dan *JavaScript* serta basis data MySQL sebagai penyimpanan data. Aplikasi ini dapat digunakan melalui *smartphone* maupun komputer secara online untuk dapat mengakses informasi geografis secara aktual menggunakan Google API.

ReRoute merupakan aplikasi yang dibangun untuk mengimplementasikan algoritma dijkstra guna mencari rute tujuan wisata terpendek. Selain itu aplikasi Reroute juga dilengkapi dengan fitur rekomendasi tujuan wisata yang menyediakan informasi tempat wisata popular disekitar pengguna dan kota pilihan. Adapun langkah-langkah penggunaan aplikasi tersebut adalah:

a. Masuk aplikasi untuk membuat rencana perjalanan serta mencari tempat wisata popular dikota pilihan. Masuk aplikasi dengan memasukan username berupa email dan password seperti terlihat pada gambar 10. Setelah berhasil login, maka akan muncul form pencarian tempat wisata popular seperti terlihat pada gambar 11.



Gambar 10. *Form Login*

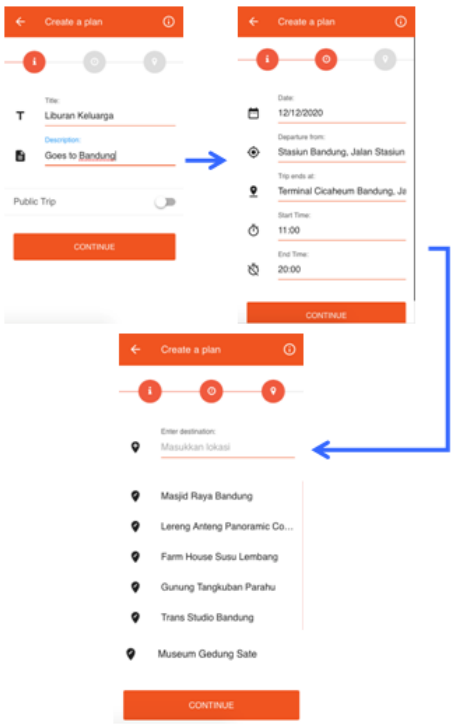


Gambar 11. *Form pencarian tempat wisata popular.*

b. Pilih menu My Trip kemudian tekan tombol tambah (+) untuk membuat rencana perjalanan baru. Proses ini dapat dilihat pada gambar 12.

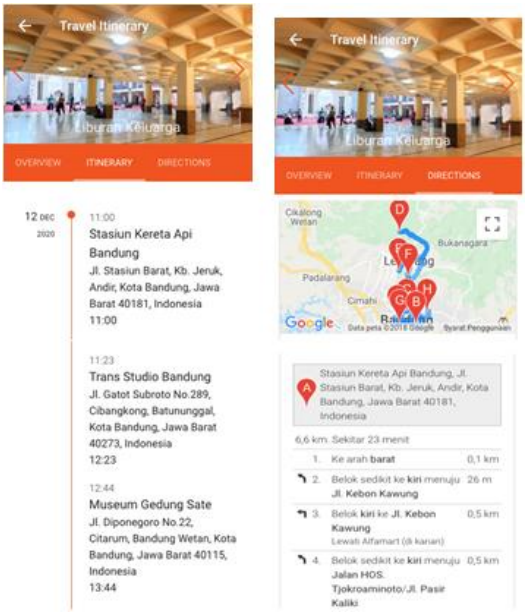
Gambar 12. *Form Penambahan rencana perjalanan baru.*

c. Isikan judul dan deskripsi perjalanan serta tanggal, lokasi awal dan akhir perjalanan. Kemudian pilih tujuan wisata mana saja yang akan dikunjungi. Tekan Continue untuk menghitung dan memperoleh rekomendasi rute perjalanan. Prosedur ini dapat dilihat dari gambar 13.



Gambar 13. *Form untuk menghitung rekomendasi perjalanan.*

d. Hasil dari perhitungan berupa total jarak rute wisata terpendek dan *itenerary* yang dapat dilihat pada gambar 14 berikut:



Gambar 14. *Form hasil perhitungan*

PENUTUP

Algoritma dijkstra dapat diterapkan pada aplikasi pencarian rute perjalanan terpendek sebagai penghitung jarak dari lokasi awal tujuan wisata sampai dengan lokasi akhir tujuan wisata tanpa mengulangi tujuan-tujuan wisata lainnya. Dijkstra akan menghitung setiap jarak terpendek dari setiap titik atau objek wisata dari lokasi awal hingga lokasi akhir objek wisata. Dijkstra menyimpan setiap nilai terkecil pada setiap jarak dari tiap titik sampai memperoleh total jarak terpendek dengan keakuratan sebesar 76,7%.

Disarankan untuk penelitian selanjutnya dapat mengembangkan sistem yang dapat mengoptimalkan algoritma dijkstra dengan sempurna, dimana mengacu pada kekurangan algoritma dijkstra dimana algoritma ini hanya sebatas antar objek.

DAFTAR PUSTAKA

[1] Michalewicz, Z. dan Fogel, D. B, *How to Solve It: Modern Heuristic*, New York: Springer, 2000.

[2] O.A, Hector, dkk, *The Shortest-Path Problem: Analysis and Comparison of Methods*, Spain: Morgan & Claypool, 2015.

[3] S.M. Firman Aji, Riri Nada Devita, dan Sherly Allsa Siregar, *Implementation of Traveling Salesman Problem (TSP) based on Dijkstra's Algorithm in Logistics System*, Malang: Universitas Negri Malang, 2016.

[4] Sahyar, Algoritma dan Pemrograman Menggunakan Matlab (Matrix Laboratorium), Jakarta: Kencana, 2016.

[5] R. Halda, Implementasi Algoritma Dijkstra Untuk Menentukan Jalur Terpendek Rumah Sakit Di Kota Palembang. Palembang: Universitas Bina Darma, 2016.

[6] AD, Hidayat, Bintang Yuda dan Dian Septiana. Analisis Pemecahan Maslah Rute Terpendek Antara Kota Jakarta Dengan Kota Bandung, 2017.

[7] S. Nandiroh, Haryanto dan Hafidh Munawir, Implementasi Algoritma Dijkstra Sebagai Solusi Efektif Pembuatan Sistem Bantuan Bencana *Real Time*. Surakarta: Universitas Muhammadiyah Surakarta, 2014