Jurnal TIKomSiN, Vol. 8, No. 1, 2020
DOI: https://doi.org/10.30646/tikomsin.v8i1.483
ISSN Cetak : 2338-4018
ISSN Online : 2620-7532

Implementasi Algoritma Dijkstra Dan Metode Haversine Pada Penentuan Jalur Terpendek Pendakian Gunung Merapi Jalur Selo Berbasis Android

Roddy Yoto Sumaryo[1)]; Paulus Harsadi [2)]; Didik Nugroho[3)]
[1) 2) 3)] Program Studi Teknik Informatika, STMIK Sinar Nusantara
[1)] roddyyotos@gmail.com; [2)] paulusharsadi@sinus.ac.id; [3)] didikhoho@sinus.ac.id

ABSTRACT

Mount Climbing or mountaineering is one of high-risk sports having certain techniques. There are many factors emerging the lost of mountaineer, such as, getting lost or having out of the track because of bad weather and trying for a new track without well-preparations. Meanwhile, the information of mountaineering route usually can be got from brochure or map at mount base camp in general. Furthermore, some cases state that the mountaineers also must find the closest path to the peak or the closest base camp because of bad weather or condition. The lack of mountain cartographic application and the importance of the closest track become the background of the Dijkstra algorithm implementation based on Android for deciding the shortest track to Mount Merapi via Selo. The methods of this research are Dijkstra algorithm for deciding the shortest rate and Harversine method for giving the value of passed node. The purpose of this research is to determine the shortest path to Mount Merapi via Selo. The use of Google map and GPS applications are used for mapping and finding the location coordinate. The results of research are this application can be used easily using smartphone and the feasibility test shows "Good" value with 77% average.

Keyword : Shortest Path Problem, Haversine formula, Dijkstra Algorithm

I. PENDAHULUAN

Pendakian gunung merupakan salah satu olahraga ekstrim yang melibatkan aktifitas pemanjatan pada lereng dan tebing, sehingga termasuk olahraga yang berbahaya.

Di indonesia dengan banyaknya gunung membuat banyak orang suka terhadap olahraga ini meskipun memiliki pengetahuan dan pengalaman yang sedikit. Pengetahuan dan pengalaman tentang pendakian merupakan hal utama yang harus dimiliki oleh para pendaki, tetapi sedikitnya media informasi rute menuju puncak yang aman membuat angka pendaki tersesat dan hilang semakin tinggi meski para pendaki memiliki pengetahuan dan pengalaman yang cukup.

Pada tahun 2018, BASARNAS menginformasikan terjadi peningkatan pendaki tersesat dan hilang sebanyak 16 kasus jika dibandingkan tahun sebelumnya yang hanya terjadi 3 kasus. Sedang di tahun 2019 ada 15 kasus yang ditemukan dalam keadaan meninggal.

Informasi rute pendakian pada umumnya di dapat melalui brosur atau informasi di basecamp pendakian atau GPS. tetapi informasi ini masih kurang untuk kasus-kasus tertentu semisal pendaki ingin segera sampai kepuncak dikarenakan waktu dan cuaca atau tersesat dijalur tertentu. Sehingga tujuan penelitian ini untuk menentukan jalur terpendek pendakian gunung merapi jalur selo.

Berdasarkan hal diatas maka dalam penelitian ini akan menggunakan Algoritma Dijkstra dengan penentuan nilai titik menggunakan Metode Haversine yang di implementasikan ke andorid yang bisa di akses offline.

II. TINJAUAN PUSTAKA

2.1. Penelitian Terkait

Penelitian tentang jalur di area gunung atau pegunungan sudah banyak dilakukan antara lain penyelamatan di area gunung dan hutan menggunakan UAV [1] dimana kalkulasi algoritma yang diusulkan mampu untuk melakukan pencarian rute terbaik menghindari bahaya di udara semisal asap dari kebakaran atau tebing tebing tinggi dalam penerbangan rendah, kemudian dipenelitian yang lain yaitu tentang skenario pencarian rute di pegunungan menggunakan Web 3D [2] dimana dari hasil penelitian ini kita bisa memilih rute perjalanan dengan tampilan virtual seperti daerah sebenarnya.

Di penelitian yang lain pencarian rute terbaik dan aman di gunakan untuk perancangan rute pembuatan jalan di pegunungan [3].

ISSN Cetak : 2338-4018
ISSN Online : 2620-7532

2.2. Metode Haversine

Metode Haversine digunakan untuk menghitung jarak geografis antara titik simpul di permukaan bumi menggunakan garis lintang (longitude) dan garis bujur (lattitude).

Sebagai variabel input Haversine formula adalah persamaan penting pada navigasi, memberikan jarak lingkaran besar antara dua titik pada permukaan bola (bumi) berdasarkan bujur dan lintang.

Haversine memiliki banyak kelebihan dibanding perhitungan jarak geodetic lain yaitu mudah dalam perhitungan, akurat, memiliki tingkat eror rendah dalam kecepatan menganalisa [4]. Penggunaan algoritma ini sudah banyak dilakukan antara lain untuk pencarian rute sekolah [5] , masjid [6], bahkan service center [7].

Dengan mengasumsikan bahwa bumi berbentuk bulat sempurna dengan jari-jari R 6.371 km, dan lokasi dari 2 titik di koordinat bola (lintang dan bujur) masing-masing adalah lon1, lat1, dan lon2, lat2, maka rumus Haversine dapat ditulis dengan persamaan sebagai berikut:

$$x = (lon2 - lon1)cos((lat1 + lat2)/2);$$
$$y = (lat2 - lat1);$$
$$d = \sqrt{(x^2 + y^2)R} \quad \text{..... (1)}$$

Keterangan :
x : Longitude (Lintang)
y : Lattitude (Bujur)
d : Jarak
R : Radius Bumi =6.371 km
1^0 : 0.0174532925 radian.

2.3. Algoritma Dijkstra

Algoritma Dijkstra merupakan salah satu bentuk algoritma greedy. Algoritma ini termasuk, algoritma pencarian graf yang digunakan untuk menyelesaikan masalah lintasan terpendek dengan satu sumber pada sebuah graf yang tidak sebuah pohon lintasan terpendek [8]. Algoritma ini sering digunakan pada routing. Algoritma Dijkstra menggunakan adjacent list untuk merepresentasikan sebuah jaringan.

Secara garis besar algortima Dijkstra membagi semua node menjadi dua, kemudian dimasukkan ke dalam tabel yang berbeda, yaitu tabel permanen dan tabel temporal. Tabel permanen berisi node awal dan node-node yang telah melalui proses pemeriksaan dan labelnya telah diubah dari temporal menjadi permanen. Tabel temporal berisi node-node yang berhubungan dengan node pada tabel permanen.

Algoritma ini sudah banyak pengembangan menyesuaikan dengan tema masing-masing penelitian [9], [10], [11], [12].

Beberapa penelitian yang sudah dilakukan semisal di evakuasi krumunan [13], atau penerapan pada robot untuk menentukan rute [14].

III. METODE PENELITIAN

3.1. Sumber data

Sumber data primer (utama) dalam pengerjaan skripsi ini didapatkan dari studi pustaka, untuk lebih detailnya dapat dijelaskan dibawah ini :

a) Observasi

Pengumpulan data yang tepat sasaran untuk mempelajari suatu sistem yang dimaksud. Melakukan pengamatan langsung terhadap suatu kegiatan yang sedang dilakukan. Peneliti melaksanakan observasi untuk mengumpulkan data-data dengan cara melakukan pendakian langsung di Gunung Merapi.

b) Studi Pustaka

Mengumpulkan data, informasi serta pengetahuan dengan cara mencari dari buku-buku tentang teori bersangkutan dalam pembuatan aplikasi dan melakukan survey ke jalur pendakian.

3.2. Langkah Penelitian

Langkah penelitian yang ada pada penelitian ini dengan menggunakan tahap analisa data, tahap desain (*system design*), tahap implementasi sistem, tahap pengujian sistem.

3.2.1 Tahap Analisa Data

Dalam tahap ini dilakukan analisis terhadap data yang didapat dari observasi dan GPS

3.2.2 Tahap Analisa Kebutuhan Sistem

Analisa kebutuhan sistem baik perangkat keras (*hardware*) maupun perangkat lunak (*software*) yang digunakan dalam pembuatan aplikasi penentuan jalur terpendek pendakian gunung merapi jalur selo.

3.2.3 Desain Sistem

Pada tahap desain sistem menggunakan visual dari GIS dengan tampilan sistem dan lain – lain yang telah disesuaikan dengan analisis kebutuhan pada tahap awal untuk

ISSN Cetak : 2338-4018
ISSN Online : 2620-7532

menyelesaikan permasalahan tersebut.

3.2.4 Implementasi Sistem

Implementasi sistem dimulai dari perancangan kemudian dilanjutkan dengan pembuatan program dengan GIS dan tool android.

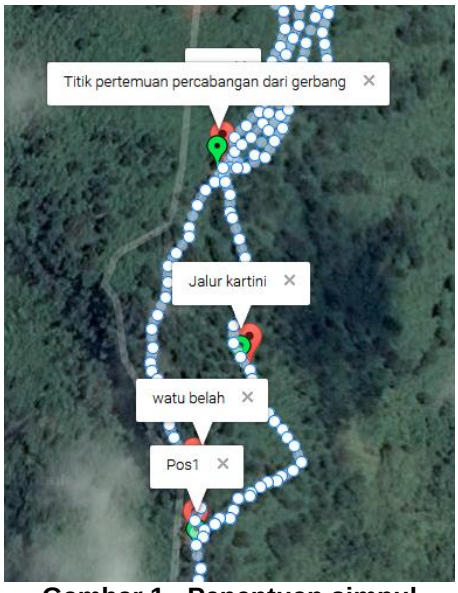
3.2.5 Pengujian Sistem

Pengujian perangkat lunak dengan menggunakan uji fungsionalitas sistem, pengujian dilakukan dengan hanya menjalankan atau mengeksekusi unit atau modul, kemudian diamati apakah hasil dari unit itu sesuai dengan proses bisnis yang diinginkan. Sistem dinyatakan berjalan lancar, jika sistem yang dijalankan sesuai dengan yang diharapkan. Pengujian lain yaitu dengan uji kelayakan.

IV. HASIL DAN PEMBAHASAN

4.2.1 Penentuan Titik Simpul

Peneliti menentukan titik simpul dengan menggunakan GPS dan menggunakan aplikasi web simple map untuk mengetahui koordinat pos atau jalur kemudian ditandai dengan nama pos dan jalur yang sudah peneliti dapatkan dari observasi. Gambar 1. menunjukkan hasil penentuan simpul yang telah dilakukan.



Gambar 1. Penentuan simpul

4.2.2 Representasi Graph

Dalam pemrograman, agar data yang ada dalam graph dapat diolah, maka graph harus dinyatakan dalam suatu struktur data yang dapat mewakili graph tersebut. Dalam hal ini graph perlu direpresentasikan kedalam bentuk array dan dimensi yang sering disebut matrix atau direpresentasikan dalam bentuk linked list seperti Gambar 2. Bentuk mana yang dipilih biasanya tergantung kepada efisiensi dan kemudahan dalam membuat program. Berikut ini bentuk representasi graph.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
A	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
F	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
K	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	

Gambar 2. Adjenceny Matrik

4.2.3 Perhitungan Bobot Haversine

Perhitungan ini dilakukan untuk menentukan bobot nilai antar titik simpul. Gambar 3. merupakan matriks hasil perhitungan keseluruhan bobot haversine.

	1 A	2 B	3 C	4 D	5 E	6 F	7 G	8 H	9 I	10 J	11 K	12 L	13 M	14 N	15 O	16 P	17 Q	18 R	19 S	20 T
1 A	0	20	14	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~
2 B	20	0	~	5	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~
3 C	14	~	0	5	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~
4 D	~	5	5	0	3	4	6	6	~	~	~	~	~	~	~	~	~	~	~	~
5 E	~	~	~	3	0	~	~	~	7	~	~	~	~	~	~	~	~	~	~	~
6 F	~	~	~	4	~	0	~	~	2	~	~	~	~	~	~	~	~	~	~	~
7 G	~	~	~	6	~	~	0	~	4	~	~	~	~	~	~	~	~	~	~	~
8 H	~	~	~	6	~	~	~	0	3	~	~	~	~	~	~	~	~	~	~	~
9 I	~	~	~	~	7	2	4	3	0	4	4	~	~	~	~	~	~	~	~	~
10 J	~	~	~	~	~	~	~	~	4	0	~	8	~	~	~	~	~	~	~	~
11 K	~	~	~	~	~	~	~	~	4	~	0	10	~	~	~	~	~	~	~	~
12 L	~	~	~	~	~	~	~	~	~	8	10	0	15	~	~	~	~	~	~	~
13 M	~	~	~	~	~	~	~	~	~	~	~	15	0	6	8	~	~	~	~	~
14 N	~	~	~	~	~	~	~	~	~	~	~	~	6	0	~	10	~	~	~	~
15 O	~	~	~	~	~	~	~	~	~	~	~	~	8	~	0	10	~	~	~	~
16 P	~	~	~	~	~	~	~	~	~	~	~	~	~	10	10	0	5	~	~	~
17 Q	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	5	0	10	15	~
18 R	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	10	0	~	15
19 S	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	15	~	0	5
20 T	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	15	5	0

Gambar 3. Matrix bobot haversine

Nilai yang ada dalam tiap elemen matrik, menyatakan bobot busur yang menghubungkan dua buah simpul yang bersangkutan. Untuk dua buah simpul yang tidak berhubungan langsung oleh sebuah busur, maka dianggap dihubungkan oleh sebuah busur yang nilai bobotnya tidak terhingga. Dalam pemograman, karena keperluan algoritma, maka dari total bobot seluruh busur yang ada atau yang mungkin ada.

ISSN Cetak : 2338-4018
ISSN Online : 2620-7532

Contoh pada gambar 5. dua simpul A dan D tidak berhubungan langsung melalui sebuah busur, maka untuk elemen matrik yang bersangkutan diisi dengan nilai'~' karena nilai '~' dalam kasus ini cukup mewakili nilai tidak terhingga.

Berikut contoh perhitungan jarak antara titik A ke C berdasarkan rumus no.1:

$Lokasi\ A$
$lon1 = -0.889902,120.900901$

$lat1 \quad = -0.889902 \times 0.0174532925$
$radian = -0.0155317199$

$lon1 = 120.900901 \times 0.0174532925$
$radian = 2.11011878867$

$Lokasi\ C$
$lon2 = -0.9765,120.8089$

$lat2 = -0.9765 \times 0.0174532925$
$radian = -0.0170431401$

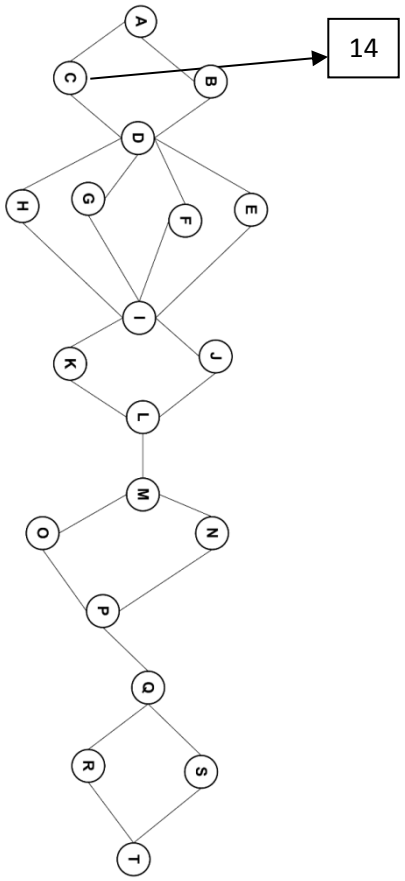
$lon2 = 120.8089 \times 0.0174532925$
$radian = 2.1085130683$

Menentukan nilai X dan Y

$x = (lon2 - lon1) \times cos((lat1 + lat2)/2)$
$x = (2.1085130683 - 2.11011878867) \times$
$cos((-0.00155317199 \pm 0.0170431401)/2)$
$x = -0.0016055074$
$y = (lat2 - lat1)$
$y = (-0.0170431401 - 0.0155317199)$
$y = 0.0015114202$

$d = \sqrt{(x^2 + y^2)} \times R$
$d = AR$
dengan $A = \sqrt{(x^2 + y^2)}$
$R = 6371$
$A =$
$\sqrt{(-0{,}0016055074)^2 + (0{,}0015114202)^2}$ x 6371

Hasil perhitungan Jarak node A – C = 14 sesuai Gambar 4.



Gambar 4. Graph Bobot A-C

Formula Haversine akan menghitung latitude dan longtitude pada tiap tiap node yang ada didalam database. Sehingga seluruh node dari A ke T akan diketahui hasil perhitunganya.

4.2.4 Perhitungan Algoritma Dijkstra

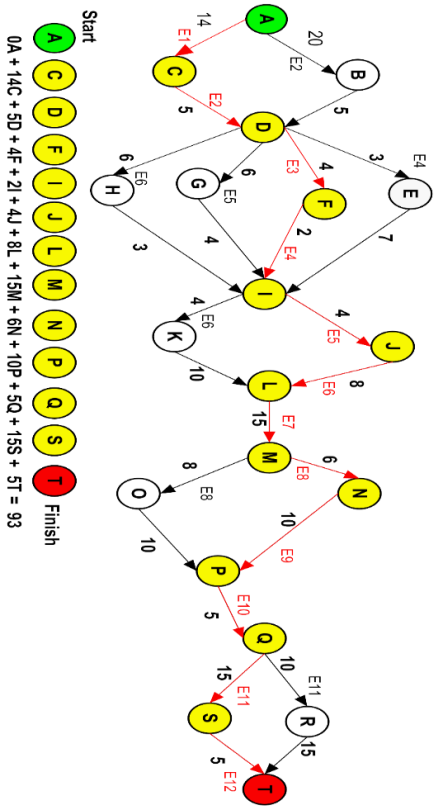
Algoritma ini bertujuan untuk menemukan jalur terpendek berdasarkan bobot terkecil dari satu titik ke titik lainya.

Langkah perhitungan:

1. Tentukan titik mana yang akan menjadi node awal (A) dan tujuan akhir (T).
2. Memilih ruas garis yang jumlah bobotnya terkecil.
3. Pilih lintasan dengan memilih E2 yang menghasilkan bobot terkecil yang tidak membentuk siklus.
4. Ulangi langkah no 3 sampai diperoleh tujuan.

Kesimpulan bobot terkecil dari perhitungan diatas adalah 93, sehingga node yang harus dilewati dari A hingga T adalah A-C-D-F-I-J-L-M-N-P-Q-S-T sesuai dengan Gambar 5.

ISSN Cetak : 2338-4018
ISSN Online : 2620-7532



Gambar 5. Graph lintasan terpendek

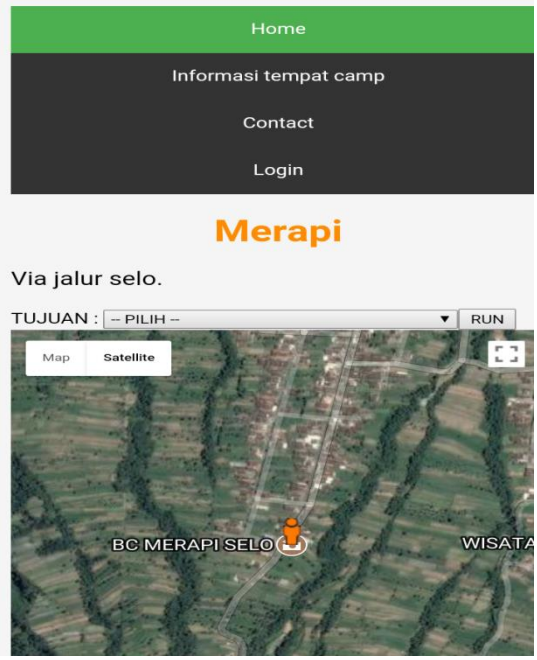
Berikut keterangan nama node :

a. Basecamp NewSelo
b. Jalur 2 Menuju gerbang
c. Jalur 1 Menuju gerbang
d. Gerbang
e. Percabangan 4 menuju pos1
f. Percabangan 3 menuju pos1
g. Percabangan 2 menuju pos1
h. Percabangan 1 menuju pos1
i. Titik pertemuan percabangan menuju pos 1
j. Jalur kartini
k. Watu belah
l. Pos 1
m. Percabangan menuju lumutan via jalur evakuasi dan pos2
n. Lumutan via pos 2
o. Jalur evakuasi
p. Jalur menuju pos pemancar
q. Pos pemancar
r. Puncak kiri kawah mati
s. Pasar bubrah jalur 1
t. Puncak

4.2.5 Implementasi

Perancangan program aplikasi mengacu pada desain-desain sistem sebelumnya. Pada tahap perancangan input dan output menggunakan software dreamweaver sedangkan untuk implementasi pembuatan database menggunakan MySQL. Dan untuk sistem operasinya menggunakan Windows 7 Ultimate. Dan diimplementasikan di smartphone android dengan sistem operasi *marshmellow.*

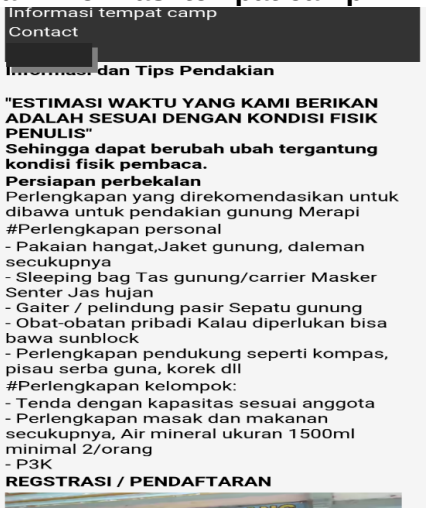
Tampilan home



Gambar 6. Halaman home

Halaman home seperti Gambar 6. berisi tampilan map untuk menentukan tampilan awal dan untuk menentukan posisi awal dan tujuan.

Tampilan informasi tempat camp



Gambar 7. Halaman informasi tempat camp

Gambar 7 menjelaskan tentang informasi dan tips pendakian gunung Merapi.

4.2.6 Pengujian sistem

ISSN Cetak : 2338-4018
ISSN Online : 2620-7532

a. Pengujian fungsionalitas

Pengujian fungsionalitas yang digunakan adalah untuk menguji aplikasi setiap form inputan yang akan diolah dalam sistem. Tabel 1 merupakan hasil pengujian fungsionalitas sistem.

Tabel 1. Uji fungsionalitas

No	Jenis Uji	Komponen Sistem yang diuji	Skenario Uji	Hasil yang diharapkan	Hasil Aplikasi	Status Uji	Hasil Pengujian
1	Uji Normal	Form home,run rute,informasi,contact	• Masukan node awal dan tujuan,run	• Cek validasi Tampil halaman menu utama	• Tampil halaman	Normal	Diterima
	Uji Salah	Form halaman run	• Masukan node awal dan tujuan	• Tolak validasi Muncul pemberitahuan	• Muncul pesan "lokasi anada sudah dekat"	Normal	Diterima

b. Pengujian kelayakan

Dalam pengujian kelayakan ini penulis membagikan angket kepada 20 responden dengan rubrik seperti Tabel 2.

Tabel 2. Kategori Penilaian

Interval	Kategori
75 % - 100 %	Baik
60 % - 74 %	Cukup
40% - 59 %	Kurang Baik
Kurang dari 24 %	Tidak Baik

Tabel 3. Hasil penilaan semua variabel

No	Kriteria Penilaian	Frekuensi Jawaban				Presentase	Kategori
		SB	B	KB	TB		
1	Tampilan						
	Komposisi warna	3	13	4	0	73	Cukup
	Kejelasan teks yang ada	6	12	2	0	80	Baik
	Variasi tampilan / gambar	2	15	4	0	76	Baik
2	Kemudahan penggunaan						
	Kemudahan mengoperasikan aplikasi	6	12	2	0	80	Baik
	Kemudahan penginputan data	3	16	1	0	77	Baik
3	Kinerja sistem						
	Tujuan sistem	7	12	1	0	82	Baik
	Hasil yang diberikan sudah sesuai dengan kebutuhan dan tujuan	4	14	2	0	77	Baik
	Presentase rata-rata	22%	67%	11 %	0%		
	Jumlah					545	
	Rata-rata					77,857	Baik

Dari hasil perhitungan Tabel 3. terlihat bahwa penilaian terhadap semua variable memiliki nilai rata-rata 77,8 %. Berdasarkan kategori penilaian, nilai rata-rata 77,8 % berada dalam interval 76 % - 100 %. Jadi dapat disimpulkan bahwa penilaian pada semua variable termasuk dalam kategori "BAIK"

c. Pengujian Akurasi

Berdasarkan perhitungan algoritma Dijkstra dengan metode haversine yang dilakukan bobot terkecil dari perhitungan algoritma adalah 93, sehingga node yang harus dilewati dari A hingga T adalah A-C-D-F-I-J-L-M-N-P-Q-S-T sesuai dengan Jalur sesungguhnya yang diperhitungkan menggunakan google map.

V. PENUTUP

5.1. Kesimpulan

Pada akhir laporan skripsi ini saya telah uraikan dan membahas permasalahan yang terjadi pada "Implementasi Algoritma Dijkstra Pada Penentuan Jalur Terpendek Pendakian Gunung Merapi Jalur Selo Berbasis Android", dimana sesuai dengan perumusan masalah dan pembatasan masalah yang saya sampaikan sebelumnya maka dapat menarik kesimpulan sebagai berikut :

Dalam penelitian yang telah dilakukan penulis telah berhasil merancang dan membangun sebuah Implementasi Algoritma Dijkstra Pada Penentuan Jalur Terpendek Pendakian Gunung Merapi Jalur Selo Berbasis Android.

Dengan aplikasi Aplikasi Implementasi Algoritma Dijkstra Pada Penentuan Jalur Terpendek Pendakian Gunung Merapi Jalur Selo Berbasis Android ini dapat membantu pendaki untuk memilih rute terpendek diantara beberapa cabang jalur didalam rute.pendakian jalur selo. Hal ini dibuktikan dari hasil perbandingan antara perhitungan manual dan perhitungan sistem yang memliki nilai yang sama selain itu dibuktikan dari uji kelayakan sistem yang memperoleh nilai kategori "Baik" atau dapat disimpulkan aplikasi tersebut layak dan membantu pendaki pemula.

5.2. Saran

Dalam pembuatan aplikasi ini peneliti berhasil mengimplementasikan algoritma Dijkstra untuk menentukan rute terpendek dipegunungan. Diharapkan untuk penelitian selanjutnya dapat mengembangkan aplikasi pemetaan sekaligus pemandu pendakian dan bisa dijalankan secara online maupun offline dan bisa ditambah banyak rute gunung lain di seluruh dunia. Oleh karena itu peneliti

ISSN Cetak : 2338-4018
ISSN Online : 2620-7532

mengharapkan kepada pembaca untuk melakukan penyempurnaan, sehingga akan dihasilkan sebuah aplikasi pemetaan jalur pendakian yang lebih sempurna.

DAFTAR PUSTAKA

[1] L. Liu, F. Yang, and Z. Wang, "Path planning of UAV in mountain area's forest rescuing," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 490, no. 4, 2019.

[2] F. Yan and J. Jia, "m 2 ACO : Multi-agent Path Planning Algorithm for Web3D Mountain Scenario," no. Aita, pp. 196–199, 2016.

[3] K. Samani and S. Hosseiny, "Planning road network in mountain forests using GIS and Analytic Hierarchical Process (AHP)," *Casp. J. Env. Sci.*, vol. 8, no. 2, pp. 151–162, 2010.

[4] O. G. Esenbuga, "Comparison of Principal Geodetic Distance Calculation Methods for Automated Province Assignment in Turkey," *16th Int. Multidiscip. Sci. GeoConference SGEM2016, Informatics, Geoinformatics Remote Sens.*, vol. 2, no. June 2016, 2016.

[5] P. Dauni, M. D. Firdaus, R. Asfariani, M. I. N. Saputra, A. A. Hidayat, and W. B. Zulfikar, "Implementation of Haversine formula for school location tracking," *J. Phys. Conf. Ser.*, vol. 1402, no. 7, 2019.

[6] I. Setyorini and D. Ramayanti, "Finding Nearest Mosque Using Haversine Formula on Android Platform," *J. Online Inform.*, vol. 4, no. 1, p. 57, 2019.

[7] D. Putra, B. DaniawanDaniawan, S. Witno, and A. Wijaya, "The Analysis and Design Marketplace Information Systems Web-Based of Electronic Repair Service Providers with Haversine Method," *bit-Tech*, vol. 2, no. 1, pp. 53–62, 2019.

[8] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische mathematik*, vol. 1, no. 1. pp. 269–271, 1959.

[9] Y. Huang, Q. Yi, and M. Shi, "An Improved Dijkstra Shortest Path Algorithm Yizhen," *Proc. 2nd Int. Conf. Comput. Sci. Electron. Eng. (ICCSEE 2013)*, vol. 2, no. 2, pp. 226–229, 2013.

[10] C. Yin and H. Wang, "Developed Dijkstra shortest path search algorithm and simulation," *2010 Int. Conf. Comput. Des. Appl. ICCDA 2010*, vol. 1, no. Iccda, pp. 116–119, 2010.

[11] Y. Deng, Y. Chen, Y. Zhang, and S. Mahadevan, "Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment," *Appl. Soft Comput. J.*, vol. 12, no. 3, pp. 1231–1237, 2012.

[12] S. Broumi, A. Bakal, M. Talea, F. Smarandache, and L. Vladareanu, "Applying Dijkstra algorithm for solving neutrosophic shortest path problem," *Int. Conf. Adv. Mechatron. Syst. ICAMechS*, vol. 0, pp. 412–416, 2016.

[13] Y. Z. Chen, S. F. Shen, T. Chen, and R. Yang, "Path optimization study for vehicles evacuation based on Dijkstra algorithm," *Procedia Eng.*, vol. 71, pp. 159–165, 2014.

[14] H. Wang, Y. Yu, and Q. Yuan, "Application of Dijkstra algorithm in robot path-planning," *2011 2nd Int. Conf. Mech. Autom. Control Eng. MACE 2011 - Proc.*, no. 2010011004, pp. 1067–1069, 2011.