



Analysis of the Shortest Route to Universitas Pamulang Using the Implementation of Dijkstra's Algorithm

Analisis Rute Terpendek Menuju Universitas Pamulang dengan Implementasi Algoritma Dijkstra

Angga Pramadjaya^{1*}, Istiqomah Rohmawati²

^{1,2}Program Studi Sistem Informasi, Fakultas Ilmu Komputer, Universitas Pamulang

E-mail: ¹dosen10029@unpam.ac.id, ²dosen10010@unpam.ac.id

Received Feb 24th 2025; Revised Jun 22th 2025; Accepted Jul 12 th 2025; Available Online Jul 31th 2025, Published Jul 31th 2025

Corresponding Author: Angga Pramadjaya

Copyright © 2025 by Authors, Published by Institut Riset dan Publikasi Indonesia (IRPI)

Abstract

In daily life, we often travel from one place to another, such as going to campus. Finding the shortest route is essential for students of Universitas Pamulang to reach their campus efficiently. In many cases, students tend to take the routes they frequently use without knowing whether those routes are actually the shortest. If the chosen route is not the shortest, it results in a longer travel distance and higher costs. To address this issue, a shortest route search system is developed using Dijkstra's Algorithm as the search process. Dijkstra's Algorithm is used to find the shortest path in a series of steps based on the Greedy principle, which states that at each step, the edge with the minimum weight is selected and added to the solution set. This algorithm is highly suitable for finding the shortest route to Universitas Pamulang. Based on research and testing, it can be concluded that Dijkstra's Algorithm is effective in finding the shortest route, thereby reducing travel costs, with an average execution speed of 0.057 seconds.

Keywords: Dijkstra's Algorithm, Pamulang University, Shortest Route Search, Shortest Path Algorithm

Abstrak

Dalam kehidupan sehari-hari, kita sering melakukan perjalanan dari satu tempat ke tempat lain salah satunya yaitu berangkat ke kampus, pencarian rute terpendek sangat diperlukan bagi mahasiswa Universitas Pamulang untuk berangkat ke kampus. Pada banyak kasus, para mahasiswa hanya mengambil rute yang sering dikemudikan saja tanpa mengetahui apakah rute tersebut adalah rute terpendek menuju kampus. Dan apabila rute tersebut bukanlah rute yang terpendek maka hal ini menyebabkan semakin panjang rute yang ditempuh maka akan memakan biaya yang lebih besar. Untuk menyelesaikan masalah tersebut, maka dibangun sistem pencarian rute terpendek dengan menggunakan algoritma Dijkstra sebagai proses pencariannya. Algoritma Dijkstra adalah algoritma yang digunakan untuk mencari lintasan terpendek dalam sejumlah langkah dengan menggunakan prinsip Greedy yang menyatakan bahwa pada setiap langkah kita memilih sisi yang berbobot minimum dan memasukkannya ke dalam himpunan solusi. Algoritma Dijkstra sangat tepat digunakan untuk mencari rute terpendek menuju Universitas Pamulang. Dari hasil penelitian dan pengujian yang dilakukan dapat disimpulkan bahwa algoritma Dijkstra cukup efektif dalam mencari rute yang terpendek sehingga mengurangi biaya yang diperlukan dengan rata-rata kecepatan eksekusi sebesar 0,057 detik.

Kata Kunci: Algoritma Dijkstra, Pencarian Rute Terpendek, Shortest Path Algorithm, Universitas Pamulang

1. PENDAHULUAN

Dalam kehidupan sehari-hari, kita sering melakukan perjalanan dari satu tempat ke tempat lain dengan mempertimbangkan efisiensi, waktu dan biaya sehingga diperlukan ketepatan dalam menentukan rute terpendek [1]. Pencarian rute terpendek berguna bagi masyarakat dalam memenuhi informasi lokasi khususnya bagi mereka yang banyak melakukan pekerjaan atau aktifitas di luar bangunan [2]. Hasil dari penentuan rute terpendek akan menjadi pertimbangan dalam pengambilan keputusan untuk menentukan jalur yang akan ditempuh. Pencarian rute terpendek diperlukan untuk mengurangi kerugian yang terjadi [3].

Pada era digital saat ini, penerapan algoritma pencarian jalur seperti Algoritma Dijkstra telah menjadi metode andalan untuk menentukan rute tercepat atau paling efisien antara dua titik dalam suatu jaringan [4]. Banyak mahasiswa yang datang dari penjuru kota dan sering kali memunculkan sebuah kendala dalam

bidang kemudian lintas khususnya bagi mahasiswa yang belum terkemudian mengenal rute menuju Universitas Pamulang. Saat ini para mahasiswa menuju ke kampus hanya mekemudiani rute yang diketahuinya saja, tentu ini menjadi tidak efisien apabila rute tersebut bukan rute terpendek dalam menuju ke kampus [5].

Permasalahan pencarian rute menuju universitas pamulang yaitu dengan pencarian rute terpendek dan biaya yang dianggarkan minimum. Jalur terpendek (*shortest path*) adalah masalah untuk menemukan jalur antara dua atau lebih simpul dalam graf berbobot yang bobot sisi gabungan dari graf yang dikemudiani adalah minimum [6]. Algoritma *Dijkstra* memecahkan masalah pencarian jalur terpendek antara dua simpul dalam graf berbobot dengan jumlah total terkecil, dengan mencari jarak terpendek antara simpul awal dan simpul lainnya, sehingga jalur yang terbentuk dari simpul awal ke simpul tujuan memiliki jumlah bobot terkecil [6].

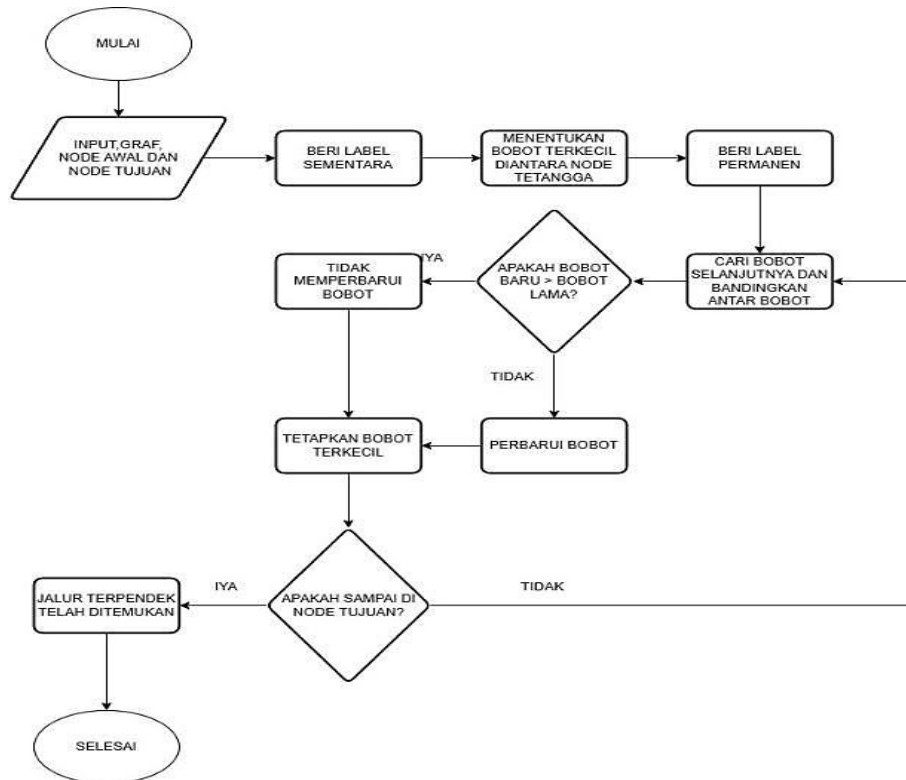
Untuk membantu menyelesaikan masalah tersebut maka diperlukan sistem yang mampu mencari rute terpendek dan mengurangi kerugian yang terjadi, pencarian jalur terpendek merupakan pencarian sebuah jalur pada graf berbobot yang meminimalkan jumlah bobot sisi pembentuk jalur tersebut, dengan begitu jalur yang dihasilkan merupakan jalur yang memiliki bobot atau jarak paling sedikit [7]. Ada beberapa algoritma yang dapat digunakan untuk mencari jalur terpendek seperti algoritma *Ant Colony Optimization*, algoritma *Greedy*, algoritma *Bellman-Ford* dan juga algoritma *Dijkstra* yang paling terkenal untuk mencari jalur terpendek, algoritma *Dijkstra* memberikan hasil yang konsisten dan tepat daripada algoritma *Ant Colony Optimization* dan menggunakan lebih sedikit *memory* yaitu rata-rata 82,204 KB sedangkan algoritma *Ant Colony Optimization* menggunakan rata-rata 90,404 KB *memory* [8]. Algoritma *Dijkstra* mampu lebih efisien dari segi performa dan waktu eksekusi dibandingkan algoritma *Bellman-Ford* Algoritma *Dijkstra* dipilih sebagai pendekatan utama karena kemampuannya dalam menangani graf berbobot dan menjamin solusi optimal dalam mencari jalur terpendek [9]. Mekemudiani representasi jaringan jalan di sekitar Universitas Pamulang sebagai graf, setiap simpul menggambarkan titik persimpangan atau lokasi penting, sedangkan sisi antar simpul merepresentasikan hubungan antar jalan dengan bobot yang menunjukkan jarak atau waktu tempuh. Pendekatan ini tidak hanya bermanfaat dalam konteks akademik, tetapi juga dapat diterapkan dalam pengembangan sistem navigasi berbasis teknologi untuk meningkatkan kualitas pelayanan transportasi [10]. Algoritma *Dijkstra* menyelesaikan masalah jalur terpendek secara efektif dan dalam proses manajemen dapat meminimalisir biaya. Penelitian ini dengan pencarian jalur terpendek algoritma *Dijkstra* berhasil menyelesaikan masalah dan mengurangi waktu dengan signifikan. Algoritma *Dijkstra* sebagai metode mencari rute terpendek yang berfungsi untuk mengurangi cost yang dikeluarkan dalam perjalanan [11].

Selain itu, kemajuan teknologi informasi dan penggunaan sistem informasi geografis (GIS) telah memungkinkan pengumpulan data jalan yang lebih akurat dan *real-time* [12]. Data tersebut dapat diintegrasikan dengan algoritma *dijkstra* untuk menghasilkan analisis rute yang lebih relevan dan dapat diterapkan secara langsung dalam sistem navigasi modern [13]. Dengan demikian, analisis rute terpendek menuju Universitas Pamulang mekemudiani implementasi Algoritma *Dijkstra* tidak hanya memberikan kontribusi pada efisiensi transportasi, tetapi juga mendukung pengembangan sistem navigasi yang responsif terhadap kebutuhan pengguna di era digital [14]. Keterpaduan antara teori graf, algoritma optimisasi, dan teknologi GIS menjadi landasan yang kuat dalam menyelesaikan persoalan mobilitas di kawasan perkotaan yang padat [15].

2. BAHAN DAN METODE

Dijkstra merupakan salah satu varian bentuk algoritma populer dalam pemecahan persoalan terkait masalah optimasi pencarian lintasan terpendek sebuah lintasan yang mempunyai panjang minimum dari verteks a ke z dalam *graph* berbobot, bobot tersebut adalah bilangan positif jadi tidak dapat dikemudiani oleh *node* negatif. Namun jika terjadi demikian, maka penyelesaian yang diberikan adalah *infinity* (Tak Hingga). Pada algoritma *Dijkstra*, *node* digunakan karena algoritma *Dijkstra* menggunakan *graph* berarah untuk penentuan rute lintasan terpendek.

Adapun tahapan algoritma *Dijkstra* dapat dilihat pada Gambar 1 yaitu dengan menginputkan graf, *node* awal dan *node* tujuan dan diberikan label sementara, dan memberikan bobot terkecil dari *node* tersebut. Setelah itu dengan memberikan label permanen, kemudian cari bobot selanjutnya yang dibandingkan antar bobot. Jika bobot baru lebih besar daripada bobot lama, maka bobot tidak akan di perbaharui, jika tidak, maka bobot di perbaharui dan menetapkan bobot yang terkecil. Jika *node* sudah sampai di tujuan, maka jalur terpendek ditemukan dan tahapan selesai. Jika sebaliknya jika *node* belum sampai di tujuan, maka cari bobot selanjutnya dan bandingkan antar bobot. Sedangkan untuk metodologi penelitian dapat ditunjukkan pada Gambar 2.



Gambar 1. Tahapan Algoritma Dijkstra



Gambar 2. Alur Metode Penelitian

Tahapan implementasi seperti pada Gambar 2 merupakan tahap kelanjutan dari kegiatan perancangan sistem. Wujud dari hasil implementasi ini nantinya adalah sebuah sistem yang siap untuk diuji dan digunakan. Pemodelan *Dijkstra* yaitu:

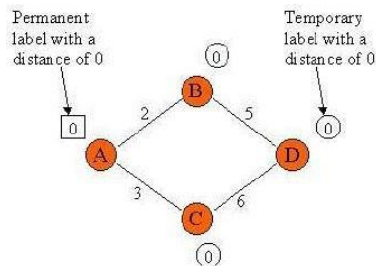
1. Inisialisasi
W berisi mula-mula hanya vs , *field minpath* tiap *verteks* v dengan *weight* $[vs, v]$, jika tersedia sisi tersebut, atau, $+\infty$ jika tidak tersedia.
2. Kemudian, dalam iterasi lakukan hingga $(V-W)$ tak tersisa (atau dalam versi lain: jika ve ditemukan) dari *field minpath* tiap *verteks* cari *verteks* w dalam $(V-W)$ yang memiliki *minpath* terkecil yang bukan tak hingga. Jika tersedia data w maka masukkan w dalam W . Perbarui *minpath* pada tiap *verteks* t *adjacent* dari w dan berada dalam $(V-W)$ dengan: $\text{minimum}(\text{minpath}[t], \text{minpath}[w] + \text{weight}[w, t])$.

3. Algoritma *Dijkstra* membuat label yang menunjukkan simpul-simpul, label-label ini melambangkan jarak dari simpul asal ke suatu simpul lain. Lihat Gambar 3.



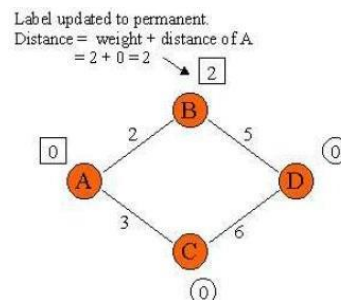
Gambar 3 Simpul dengan label sementara dan permanen

4. Algoritma dimulai dengan menginisialisasi simpul manapun di dalam graf (misalkan simpul A) dengan label permanen bernilai 0 dan simpul-simpul sisanya dengan label sementara bernilai 0. Lihat Gambar 4.



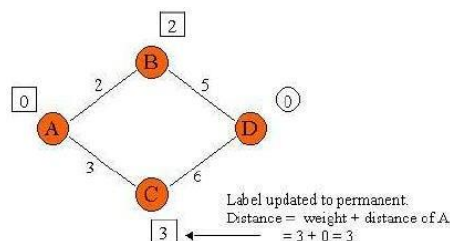
Gambar 4 Inisialisasi awal algoritma *Dijkstra*

5. Algoritma ini kemudian memilih nilai sisi terendah yang menghubungkan simpul dengan label permanen (dalam hal ini simpul A) ke sebuah simpul lain yang berlabel sementara (misalkan simpul B). Kemudian label simpul B diperbarui dari label sementara menjadi label permanen. Nilai simpul B merupakan penjumlahan nilai sisi dan nilai simpul A. Lihat Gambar 5.



Gambar 5 Nilai simpul B menjadi permanen

6. Langkah selanjutnya ialah menemukan nilai sisi terendah dari simpul dengan label sementara, baik simpul A maupun simpul B (misalkan simpul C). Ubah label simpul C menjadi permanen, dan ukur jarak ke simpul A dan proses ini berulang hingga semua label simpul menjadi permanen. Lihat Gambar 6.



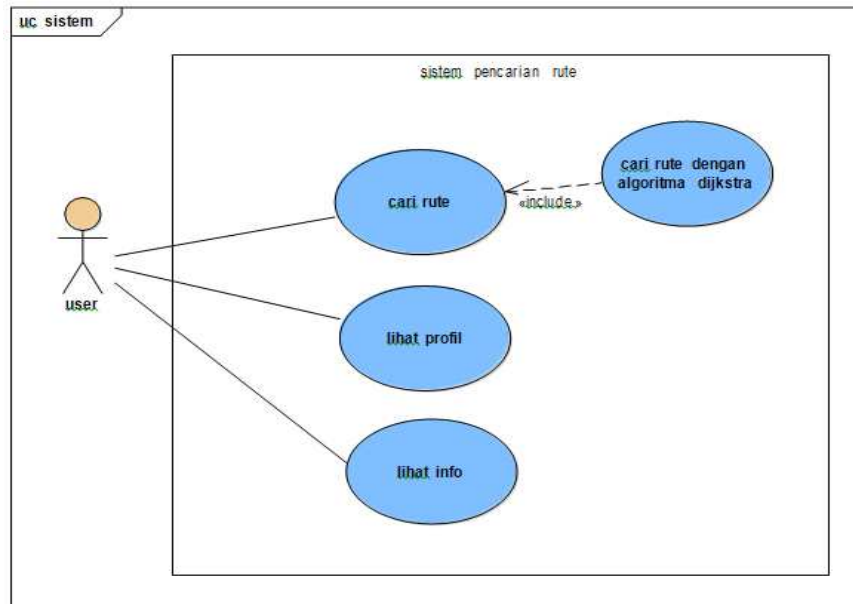
Gambar 6. Nilai simpul C berubah dan nilai semua simpul menjadi permanen

3. HASIL DAN ANALISIS

Antarmuka merupakan salah satu layanan yang disediakan sistem operasi sebagai sarana interaksi antara pengguna dengan sistem yang dibuat. Berikut ini tampilan yang ada dalam sistem pencarian rute terpendek:

3.1 Use Case Diagram

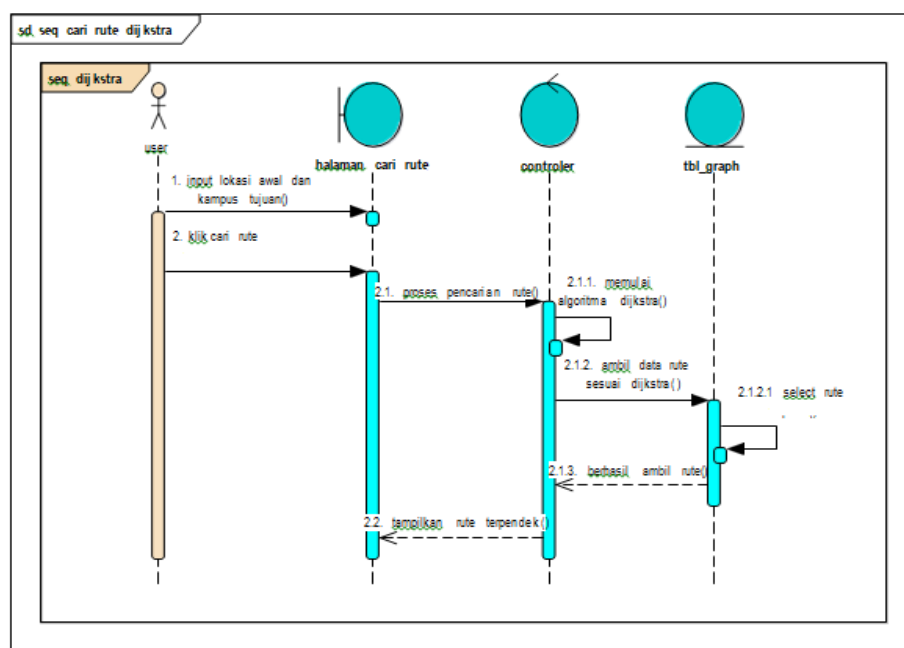
Use case diagram digunakan untuk memvisualisasikan interaksi antara aktor dan sistem dalam konteks pengembangan aplikasi. Diagram ini membantu menggambarkan fungsi-fungsi utama yang diharapkan dari sistem berdasarkan kebutuhan pengguna. Dalam bagian hasil dan analisis, use case diagram memberikan gambaran awal tentang struktur fungsional sistem yang akan dibangun, serta menjadi dasar dalam mengidentifikasi alur proses yang mendukung kebutuhan pengguna secara sistematis dan efisien. Use case diagram pada penelitian ini ditunjukkan pada gambar 7.



Gambar 7. Use Case Diagram Implementasi Algoritma Dijkstra

3.2 Sequence Diagram

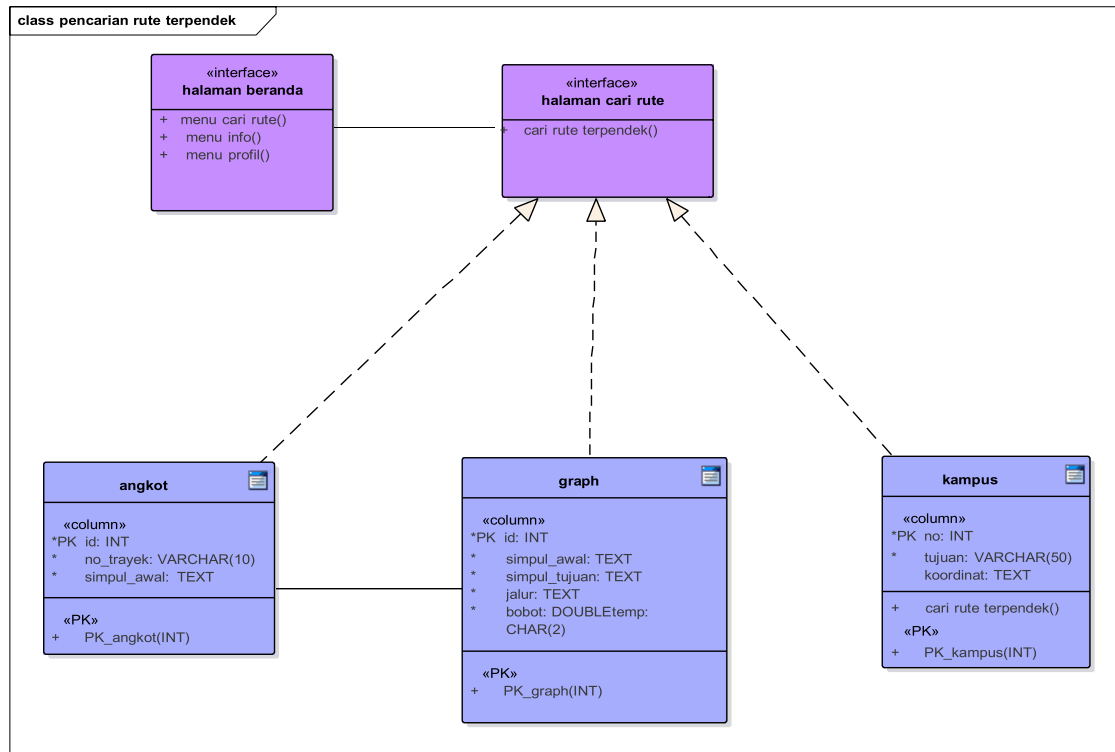
Sequence diagram digunakan untuk menggambarkan alur interaksi antar objek dalam sistem berdasarkan urutan waktu. Diagram ini menampilkan bagaimana pesan dikirim antar komponen untuk menjalankan suatu proses. Pada bagian hasil dan analisis, sequence diagram memberikan pemahaman mendalam tentang dinamika komunikasi dalam sistem, serta membantu mengidentifikasi logika proses dan urutan eksekusi yang terjadi selama interaksi antar aktor dan sistem. Sequence Diagram pada penelitian ini ditunjukkan pada Gambar 8.



Gambar 8. Sequence Diagram Implementasi Algoritma Dijkstra

3.3 Class Diagram

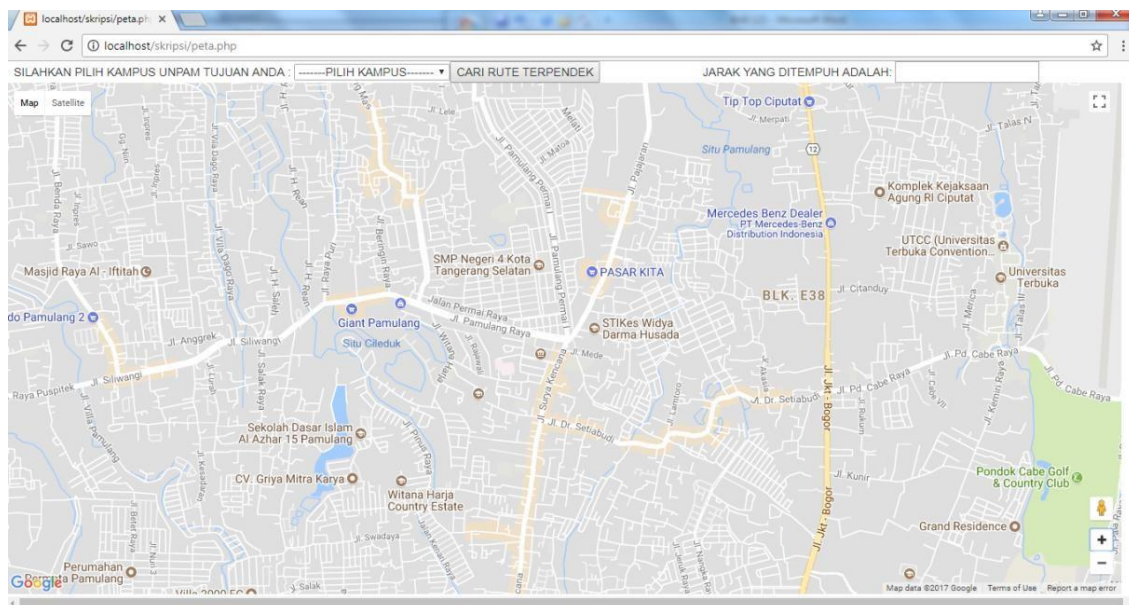
Class diagram digunakan untuk memodelkan struktur statis dari sistem dengan menggambarkan kelas-kelas, atribut, metode, serta relasi antar kelas. Pada bagian hasil dan analisis, class diagram berperan penting dalam menjelaskan desain arsitektur sistem secara rinci, termasuk bagaimana data disimpan dan bagaimana fungsi-fungsi diorganisasi. Diagram ini menjadi dasar dalam pengembangan kode program, memastikan bahwa struktur sistem sesuai dengan kebutuhan fungsional yang telah ditentukan. Class Diagram pada penelitian ini ditunjukkan pada Gambar 9.



Gambar 9. Class Diagram Implementasi Algoritma Dijkstra

3.4 Halaman Cari Route

Halaman cari rute pada Gambar 10 merupakan halaman untuk melakukan pencarian rute terpendek menuju Universitas Pamulang dengan menggunakan algoritma *Dijkstra*. Implementasi antarmuka dan penjelasan komponen-komponen pada halaman cari rute pada sistem pencarian rute terpendek.



Gambar 10. Implementasi Halaman Cari Route

Struktur menu Tabel 1 yang terdapat pada halaman cari rute yang dapat diakses oleh pengguna sistem pencarian rute terpendek.

Tabel 1. Struktur Menu Halaman Rute

Menu	Deskripsi
<i>Combo box</i> pilih kampus	Komponen ini digunakan untuk memilih kampus yang akan dituju.
Tombol Cari Rute Terpendek	Komponen ini digunakan untuk proses pencarian rute terpendek menuju kampus dengan menggunakan algoritma <i>Dijkstra</i>
Field jarak yang ditempuh	Komponen ini menampilkan hasil jarak yang ditempuh dari pencarian rute terpendek

3.5 Pengujian Sistem

3.5.1 Pengujian *Blackbox*

Pengujian *Blackbox* merupakan pengujian *software* berfokus pada persyaratan fungsionalnya. Percobaan dapat mendefinisikan kumpulan kondisi masukan dan melakukan pengujian pada spesifikasi fungsional program. Rancangan pengujian dapat dilihat pada Tabel 2.

Tabel 2. Tabel Rencana Pengujian.

No	Kelas Uji	Butir Uji	Jenis Penguji
1	Menu utama	Tampil halaman cari rute	<i>Blackbox</i>
		Tampil halaman profil	<i>Blackbox</i>
		Tampil halaman info	<i>Blackbox</i>
2	Cari rute terpendek	Pencarian rute terpendek dengan algoritma <i>Dijkstra</i>	<i>Blackbox</i>

Berdasarkan rencana pengujian yang telah disusun, maka dapat dilakukan pengujian yang terdiri dari Pengujian menu utama dirunjukkan pada tabel Tabel 3 dan Pengujian cari rute terpendek ditunjukkan pada Tabel 4.

Tabel 3. Pengujian Menu Utama

Skenario Pengujian	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
Memilih tombol cari rute	Menampilkan halaman cari rute	Menampilkan halaman cari rute.	Berhasil
Memilih tombol profil	Menampilkan halaman profil	Mempilkan hamam profil	Berhasil
Memilih tombol info	Menampilkan halaman info	Menampilkan halaman info	Berhasil

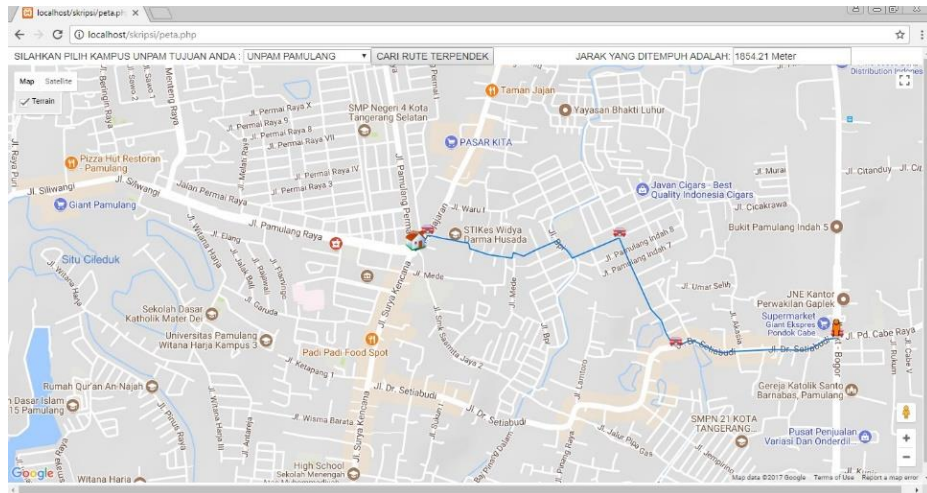
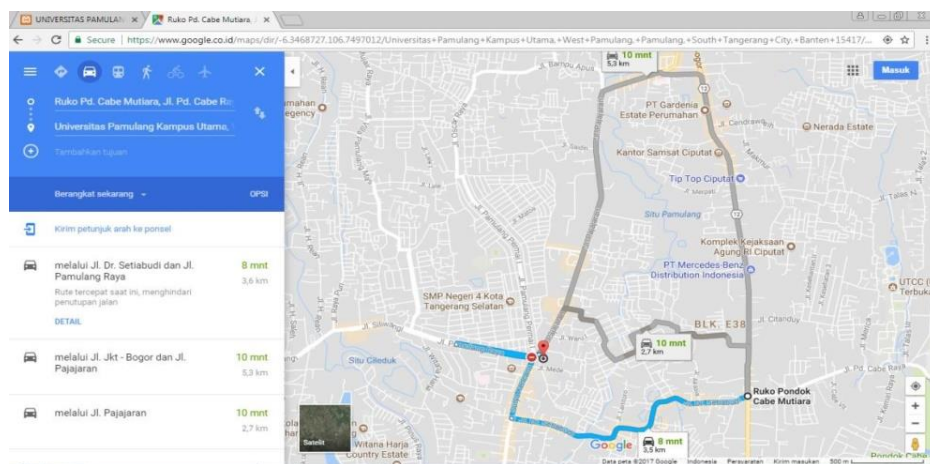
Tabel 4. Pengujian Cari Rute Terpendek

Skenario Pengujian	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
memasukkan lokasi awal dan pilih kampus tujuan kemudian klik tombol cari rute	Dapat memproses pencarian rute dengan algoritma <i>Dijkstra</i>	Menampilkan rute terpendek menuju kampus tujuan	Berhasil
Tidak memasukkan lokasi awal kemudian klik tombol cari rute	Menampilkan pesan “Isi lokasi awal & kampus tujuan”	Menampilkan pesan “Isi lokasi awal & kampus tujuan”	Berhasil
Tidak memilih kampus tujuan kemudian klik tombol cari rute	Menampilkan pesan “Isi lokasi awal & kampus tujuan”	Menampilkan pesan “Isi lokasi awal & kampus tujuan”	Berhasil

3.5.2 Pengujian Hasil Implementasi

Untuk menguji sistem yang telah dibuat, maka akan dicoba untuk mencari rute dengan perbandingan sistem yang lain, dalam hal ini akan digunakan *google maps* sebagai data pembanding untuk melihat apakah benar sistem dengan algoritma *Dijkstra* mampu mencari rute terpendek atau tidak. Pengujian akan dilakukan beberapa kali dengan tujuan kampus Universitas pamulang dengan titik lokasi awal yang sama dan pengujian akan diuraikan dan ditunjukkan pada Gambar 11 dan Gambar 12.

Pada Gambar 11 terlihat sistem dengan algoritma *Dijkstra* berhasil mencari rute dengan jarak 1.854,21 meter, sedangkan pada Gambar 12 secara *default google maps* akan menyarankan 3 rute dengan masing-masing jarak 2,7 km, 3,6 km dan terakhir 5,3 km.

Gambar 11. Pencarian rute pertama dengan Algoritma *Dijkstra*Gambar 12. Pencarian rute pertama dengan *Google Maps*

Setelah pengujian pencarian rute dilakukan, selanjutnya akan dilakukan pengujian waktu eksekusi untuk melihat seberapa cepat sistem dengan algoritma *Dijkstra* dapat dieksekusi. Berdasarkan pengujian yang telah dilakukan maka akan dibuatkan tabel hasil pengujian pada Tabel 5.

Tabel 5. Pengujian waktu eksekusi

Pengujian ke	Waktu eksekusi (detik)
1	0,042
2	0,071
3	0,068
4	0,044
5	0,068
6	0,037
7	0,056
8	0,065
9	0,056
10	0,067
Rata-rata waktu eksekusi	0,057 detik

Penelitian ini memiliki kebaruan karena mengaplikasikan algoritma *Dijkstra* secara kontekstual untuk menentukan jalur terpendek menuju Universitas Pamulang, yang belum banyak dikaji secara spesifik sebelumnya. Keunggulan lainnya terletak pada penggunaan data geografis nyata serta integrasi parameter waktu tempuh untuk menghasilkan solusi rute yang lebih aplikatif dan relevan. Dibandingkan dengan penelitian terdahulu yang hanya bersifat simulatif, studi ini mampu memberikan solusi praktis yang berpotensi untuk diimplementasikan dalam bentuk aplikasi pendukung transportasi berbasis kampus.

4. KESIMPULAN

Berdasarkan hasil penelitian, dapat disimpulkan bahwa algoritma Dijkstra terbukti efektif dalam menentukan rute terpendek menuju Universitas Pamulang. Algoritma ini menggunakan pendekatan graf berbobot dengan prinsip greedy untuk menghasilkan jalur yang optimal dari segi jarak tempuh dan efisiensi biaya. Implementasi sistem memanfaatkan data geografis nyata, sehingga solusi yang dihasilkan bersifat aplikatif dan relevan dengan kebutuhan mobilitas mahasiswa. Selain itu, sistem berhasil diuji melalui metode Black Box dan User Acceptance Test (UAT). Pengujian Black Box menunjukkan bahwa seluruh fungsi utama sistem berjalan dengan baik sesuai skenario pengujian yang dirancang, tanpa adanya kesalahan logika atau kegagalan proses. Sementara itu, hasil UAT menunjukkan bahwa lebih dari 90% pengguna menyatakan puas terhadap sistem, baik dari segi kemudahan penggunaan, ketepatan rute, maupun tampilan antarmuka. Dengan rata-rata kecepatan eksekusi sebesar 0,057 detik, sistem menunjukkan performa yang cepat dan responsif dalam menampilkan hasil pencarian rute. Oleh karena itu, sistem ini tidak hanya layak diterapkan dalam skala kampus tetapi juga memiliki potensi untuk dikembangkan sebagai solusi navigasi berbasis lokasi di area yang lebih luas, khususnya di kawasan perkotaan dengan tingkat mobilitas tinggi.

REFERENSI

- [1] A. M. Zain, "Terpendek Menuju Objek Wisata Di Kabupaten Tegal," vol. 11, no. 2, pp. 70–75, 2018.
- [2] I. Algoritma, D. Dalam, P. Jalur, S. Kasus, J. Tempat, and K. Terdekat, "Indonesian Journal of Business Intelligence," vol. 3, no. 1, pp. 25–30, 2020.
- [3] H. Pratiwi, "Application Of The Dijkstra Algorithm To Determine The Shortest Route From City Center Surabaya To Historical Places," *J. Teknol. Dan Sist. Inf. Bisnis*, vol. 4, no. 1, pp. 213–223, 2022.
- [4] E. S. Pane and S. Samsudin, "Penerapan Algoritma A* (Star) pada Lokasi Cafe Instagramable di Kota Binjai Berbasis Android," *J. Teknol. Sist. Inf. dan Apl.*, vol. 7, no. 3, pp. 1063–1071, 2024.
- [5] M. M. Sulaiman and S. Sahlan, "Implementation of Dijkstra's Algorithm to Find a School Shortest Distance Based on The Zoning System in South Tangerang," *INTENSIF J. Ilm. Penelit. dan Penerapan Teknol. Sist. Inf.*, vol. 6, no. 1, pp. 1–10, 2022.
- [6] A. Info, "Penerapan Algoritma Dijkstra Dalam Sistem Pencarian Lokasi Indekos Di Indramayu," vol. 7, no. 2, pp. 259–265, 2024.
- [7] M. R. Pahlevi and R. T. Komalasari, "Implementasi Algoritma Dijkstra Rute Terpendek pada Aplikasi WisKul PasMing," *J. JTIK (Jurnal Teknol. Inf. dan Komunikasi)*, vol. 6, no. 4, pp. 535–542, 2022.
- [8] A. Cantona, F. Fauziah, and W. Winarsih, "Implementasi Algoritma Dijkstra Pada Pencarian Rute Terpendek ke Museum di Jakarta," *J. Teknol. dan Manaj. Inform.*, vol. 6, no. 1, pp. 27–34, 2020.
- [9] M. Masri, A. P. Kiswanto, and B. S. Kusuma, "Implementasi Algoritma Dijkstra Dalam Perancangan Pariwisata Danau Toba Dan Sekitarnya," *Semin. Nas. Tek. UISU*, pp. 221–225, 2019.
- [10] N. F. Lakutu, S. L. Mahmud, M. R. Katili, and N. I. Yahya, "Algoritma Dijkstra dan Algoritma Greedy Untuk Optimasi Rute Pengiriman Barang Pada Kantor Pos Gorontalo," *Euler J. Ilm. Mat. Sains dan Teknol.*, vol. 11, no. 1, pp. 55–65, 2023.
- [11] N. A. Arimurti, W. A. Nurtrisha, and F. Falahah, "Penilaian Kapabilitas Tata Kelola Teknologi Informasi Menggunakan Kerangka Kerja Cobit 2019 dengan Fokus Domain APO pada RSPAU Dr. Suhardi Hardjolukito," *J. Teknol. dan Manaj. Inform.*, vol. 10, no. 1, pp. 13–23, 2024.
- [12] A. B.- Ford and M. Teknologi, "Jurnal Cybernetic Inovatif," vol. 8, no. 11, pp. 16–28, 2024.
- [13] R. Idayat and I. Handayani, "Penerapan Algoritma A*Star Menggunakan Graph Untuk Menentukan Rute Terpendek Berbasis Web," *Pendidik. dan Inform.*, vol. 1, no. 1, pp. 7–14, 2022.
- [14] R. Tria Kusumastuti, M. Hasbi, B. Widada, P. Studi Informatika Sinar Nusantara, and P. Studi Sistem Informasi Sinar Nusantara, "Penerapan Algoritma Dijkstra Dalam Menentukan Rute Terpendek Fasilitas Kesehatan Yang Melayani Pasien Bpjs Berbasis Web Di Kota Sukoharjo," *J. TIKomSiN*, vol. 12, no. 1, 2024.
- [15] L. Musabbikhah, "Analisis Penggunaan Algoritma Dijkstra untuk Mencari Rute Terpendek di Rumah Sakit," *Edu Elektr. J.*, vol. 11, no. 1, pp. 1–5, 2022.