PENERAPAN ALGORITMA DIJKSTRA UNTUK OPTIMASI RUTE TERPENDEK DARI FAKULTAS KEDOKTERAN UNIMED KE EMPAT GERBANG KAMPUS MENGGUNAKAN PYTHON

Rizky Wahyudi, Muhammad Alfin, John Bush Henrydunan, Putri Harliana
Ilmu Komputer, Universitas Negeri Medan
Jalan W. Iskandar Psr V Medan Esatate Kab. Deli Serdang, Indonesia
rizkyw23@mhs.unimed.ac.id

ABSTRAK

Dengan semakin banyaknya kendaraan dan kompleksnya infrastruktur jalan di sekitar kampus, maka diperlukan sistem navigasi yang efisien. Hal tersebut akan menjadi langkah besar untuk meminimalkan waktu tempuh dan mengatasi kemacetan. Dalam penelitian ini, rute dari Fakultas Kedokteran menuju salah satu dari empat gerbang kampus akan ditetapkan sebagai rute optimal, dengan menggunakan data jarak Google Maps dan mengimplementasikan algoritma dengan Python. Metodologi penelitian ini melibatkan pengamatan aktualitas dan telaah pustaka untuk memperoleh informasi yang dapat diandalkan dan mendukung landasan teori. Terlihat bahwa algoritma Dijkstra cukup efektif dalam mengidentifikasi jalur terpendek, yang membantu dalam pengembangan sistem navigasi kampus dan memperluas pengetahuan dalam algoritma perencanaan rute. Penelitian ini mencakup pseudocode, kode, dan output program yang merepresentasikan rute yang tersedia beserta jaraknya. Hasil simulasi menunjukkan bahwa rute terpendek dari Fakultas Kedokteran ke Gerbang N melalui jalur R-M-O-N yang memiliki jarak 251 meter, hal ini menunjukkan efektivitas algoritma Dijkstra dalam mengidentifikasi rute optimal. Kesimpulannya menegaskan kembali keandalan dan efektivitas algoritma untuk navigasi kampus dan menegaskan kembali bahwa akurasi lebih lanjut dapat ditambahkan dengan menganalisis faktor lalu lintas dan memperluas sistem pemetaan.

Kata kunci : *Navigasi Kampus, Algoritma Dijkstra, Python, Rute Terpendek*

1. PENDAHULUAN

Sejalan dengan rata-rata peningkatan jumlah kendaraan dan kompleksitas infrastruktur jalan di sekitar kampus, dapat dikatakan bahwa sistem navigasi perlu diterapkan untuk meminimalisir waktu perjalanan yang diambil sehingga kemacetan juga seharusnya dapat diredakan. Ada banyak pendekatan yang bisa menyelesaikan masalah tersebut, salah satunya yaitu dengan menggunakan Algoritma Dijkstra. Algoritma Djikstra adalah algoritma greedy yang umumnya dipakai untuk menemukan jarak terpendek, dengan input graf berarah berbobot serta dengan titik awal dari sekumpulan garis. Algoritma ini sangat efektif dalam menemukan jarak terpendek dari titik tertentu ke titik lain, sehingga banyak digunakan dalam bidang pemetaan dan perencanaan rute [1]. Oleh karena itu, Algoritma Dijkstra memainkan peran yang berarti di lingkungan kampus. Implementasi yang efektif dari metode pencarian jalur ini dapat membantu mahasiswa dan staf Fakultas Kedokteran menemukan rute tercepat dari fakultas ke masing-masing empat gerbang kampus.

Adapun tujuan penelitian ini yaitu untuk mengetahui dan menganalisis bagaimana cara menemukan jalur terpendek dari Fakultas Kedokteran UNIMED menuju salah satu dari keempat gerbang UNIMED. Penentuan jalur tersebut menggunakan bahasa pemrograman Python untuk menerapkan Algoritma Dijkstra guna menemukan jalur terpendek dari fakultas ke masing-masing empat gerbang. Penulis ingin memperoleh jalur terbaik yang optimal dengan mempertimbangkan jarak antara titik tertentu yang akan dilalui. Rute tersebut kemudian direkomendasikan kepada mahasiswa dan staf berdasarkan pengalaman yang diambil dari hasil pemulusan berbagai simulasi.

Penulis berpendapat bahwa penelitian ini diperlukan karena kontribusi yang dihasilkannya dapat digunakan oleh pengembang sistem navigasi kampus. Selain itu, penelitian ini akan memperluas ilmu di bidang penerapan algoritma pencarian jalur. Itu dikarenakan hasil penelitiannya dapat diaplikasikan oleh lingkungan kampus lainnya. Manfaat selanjutnya dari penelitian ini adalah pemanfaatan Algoritma Dijkstra pada sudut pandang pemetaan dan perencanaan rute yang diimplementasikan dengan menggunakan program bahasa python.

.

2. TINJAUAN PUSTAKA

2.1. Navigasi

Proses mengawasi dan mengontrol pergerakan seseorang atau alat transportasi, seperti motor, mobil, kapal, ataupun pesawat terbang dari satu tempat ke tempat yang lainnya disebut dengan navigasi. Teknologi navigasi Ini sangat berkembang pesat. Mulai dari alat bantu navigasi ketika manusia memanfaatkan alam sekitarnya untuk menentukan lokasinya, sampai berkembangnya IPTEK sehingga manusia menciptakan peta yang memvisualisasikan keadaan dunia pada saat itu [2].

2.2. Graf

Teori graf merupakan pembahasan yang sudah lama, tetapi masih mempunyai banyak implementasi dalam kehidupan sehari-hari kita. Salah satu implementasi visual dari graf adalah peta, yang mana dapat dipakai untuk menunjukkan banyak hal yang sebenarnya di dunia nyata, seperti menetapkan jalur terpendek dari satu tempat ke tempat lainnya, memvisualisasikan dua daerah yang bertetanggaan dengan warna yang berbeda pada peta, menyusun tata letak jalur kendaraan, menata jaringan komunikasi atau internet, dan lain sebagainya. Secara matematis, graf digambarkan sebagai pasangan himpunan (V, E), yang ditulis dengan notasi G = (V, E). Di sini, V adalah himpunan simpul (vertex atau node) yang tidak kosong, dan E adalah himpunan sisi (edge) yang menghubungkan dua simpul [3].

Pada awalnya, mengatakan bahwa beberapa teori graf tampaknya tidak langsung relevan bagi seorang insinyur mungkin terdengar kejam, tetapi, seperti yang telah dibuktikan, teori ini memiliki aplikasi yang sangat nyata dan masih diperlukan dalam beragam situasi. Misalnya, terapkan konsep graf di ilmu komputer, jaringan, analisis sosial, maupun transportasi. Dengan demikian, ini menekankan urgensi pemahaman konsep graf dan penggunaannya saat memecahkan masalah sehari-hari kompleks.

2.3. Graf Berarah

Graf berarah *D* terdiri dari pasangan dua himpunan V(*D*), yaitu himpunan berhingga tak kosong di mana elemen-elemennya dinamakan titik *D*, dan himpunan berhingga (boleh kosong) di mana elemen-elemennya disebut busur *D*, sehingga setiap busur adalah pasangan berurutan dua titik *D*. Dalam kasus di mana v_1 dan v_2 merupakan dua titik *D* dan τ= (v_1, v_2) merupakan busur *D*, maka busur keluar dari titik v_1 disimbolkan dengan od v_1, sedangkan busur masuk dari titik v_2 disimbolkan dengan id v_2 dan busur τ = (v_1,v_2) biasa ditulis τ = v_1v_2 atau (1,2) [4].

2.4. Algoritma

Algoritma adalah kumpulan langkah sitematis dan logis yang digunakan untuk menyelesaikan suatu masalah. Kumpulan langkah tersebut dapat digunakan dalam berbagai situasi, seperti saat memecahkan masalah sehari-hari [5].

Salah satu permasalahan yang bisa didapati dalam kehidupan sehari-hari adalah mencari jalan terbaik untuk perjalanan menuju kampus. Dalam kasus ini, pencarian rute terbaik menuju fakultas kedokteran UNIMED adalah masalah yang dapat diselesaikan dengan bantuan algoritma. Algoritma Djikstra adalah salah satu algoritma yang bisa dipakai untuk menyelesaikan masalah ini. Algoritma ini menghitung dari titik awal ke titik terdekat, lalu ke titik kedua, dan begitu seterusnya.

2.5. Algoritma Dijkstra

Ilmuwan komputer Belanda Edsger Dijkstra adalah penemu Algoritma Dijkstra. Algoritma Djikstra dipakai untuk mendapatkan lintasan terpendek pada sebuah graf berarah. Ini termasuk dalam pembahasan teori graf pada mata kuliah matematika diskrit serta berkaitan dengan graf berbobot dan lintasan terpendek. Algoritma Djikstra menggunakan stategi greedy, di mana sisi dengan bobot terkecil dipilih pada setiap langkah untuk menghubungkan simpul yang sudah dipilih dengan simpul yang belum dipilih [6]. Meskipun demikian, algoritma Dijkstra ini juga berlaku untuk sebuah graf tak berarah [7].

Dengan asumsi bobotnya tidak negatif, algoritma Djikstra adalah salah satu teknik yang bisa digunakan secara menyeluruh terhadap titik-titiknya. Algoritma Djikstra mampu menyelesaikan masalah jalur terpendek. Ini mencakup jalur terpendek antara dua titik, jalur terpendek antara semua pasangan titik, dan jalur terpendek antara dua titik melalui beberapa titik tertentu [8].

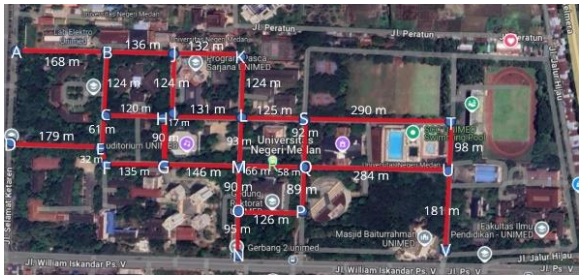
3. METODE PENELITIAN

Penelitian ini menggunakan dua pendekatan utama, yaitu pengamatan langsung dan studi pustaka, yang bertujuan untuk memperoleh data yang akurat serta mendukung analisis yang komprehensif terkait penerapan Algoritma Dijkstra dalam menemukan rute terpendek dari Fakultas Kedokteran UNIMED ke empat gerbang kampus. Pengamatan langsung dilakukan dengan memanfaatkan data jarak yang diperoleh dari Google Maps, sementara studi pustaka digunakan untuk memperkuat landasan teori melalui referensi dari berbagai sumber ilmiah yang relevan. Alur penelitian ini meliputi tahap pengumpulan data, implementasi algoritma, simulasi menggunakan Python, serta analisis hasil untuk menentukan rute optimal yang diteliti.

3.1. Pengumpulan Data

Data perhitungan operasional diambil dari Google Maps yang dinyatakan dalam meter. Aplikasi Google Maps adalah layanan peta digital gratis yang disediakan oleh Google dan bersifat open-source. Pengembang dapat mengimplementasi-kan library yang disediakan oleh Google untuk mengembangkan Google Maps sesuai keinginan mereka. Selain itu, Google Maps API memungkinkan integrasi layanan ke dalam aplikasi berbasis smartphone dan web [9].

Penulis memilih 27 rute yang berbeda dengan 21 titik atau node untuk analisis lebih lanjut. Dengan menggunakan bantuan Google Maps memudahkan kami dalam menghitung jarak dari tiap-tiap titik yang sudah kami tentukan sebelumnya seperti pada Gambar 1.



Gambar 1. Rute Kendaraan yang melalui FK UNIMED.

Berdasarkan Gambar 1 Terlihat bahwa titik A merupakan Gerbang Rusunawa, titik D merupakan Gerbang 1, Titik N merupakan Gerbang 2, titik V merupakan Gerbang 4, dan titik R merupakan Fakultas Kedokteran UNIMED. Setelah dilakukan pengukuran menggunakan Google Maps, diperoleh data rute antar titik sebagai berikut.

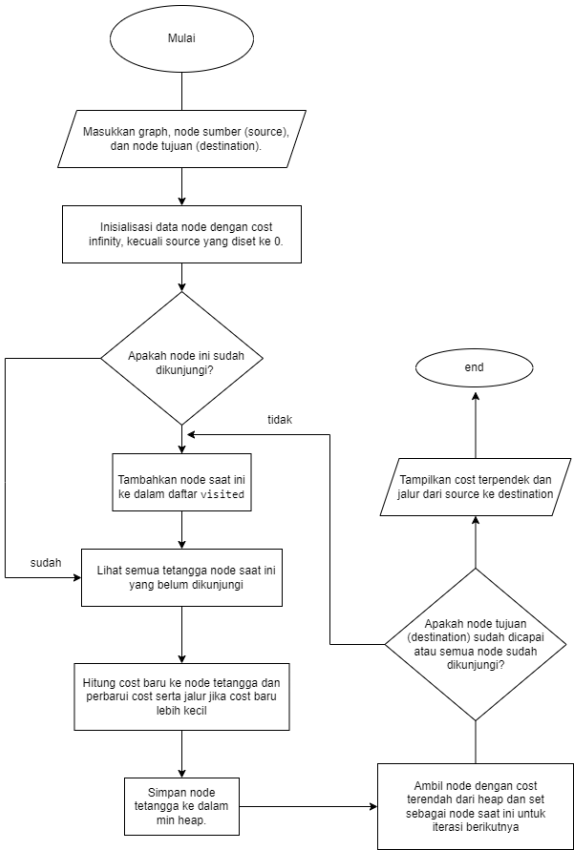
Tabel 1. Rute Menuju FK Unimed

Rute	Jarak (m)
A-B	168
B-J	136
B-C	124
C-E	61
C-H	119
D-E	179
E-F	32
F-G	135
G-H	90
G-M	146
H-I	17
I-J	124
I-L	131
J-K	132
K-L	124
L-M	93
L-S	125
M-O	90
N-O	95
O-P	126
P-Q	89
Q-R	58
Q-S	92
Q-U	284
R-M	66
S-T	290
T-U	98
U-V	181

3.2. Flowchart

Suatu jenis diagram, juga dikenal sebagai diagram alir, menampilkan algoritma atau langkah-langkah instruksi yang runtut dalam sistem disebut Flowchart. Gambaran flowchart dapat membantu menyelesaikan masalah yang mungkin muncul saat membangun sistem. Simbol pada dasarnya digunakan untuk menggambarkan flowchart. Setiap simbol menunjukkan langkah-langkah tertentu. Namun, garis penghubung digunakan untuk menghubungkan satu proses ke proses berikutnya [10].

Gambar 2 merupakan flowchart dari implementasi algoritma Djikstra yang akan menggunakan Python.



Gambar 2. Flowchart Program Algoritma Dijkstra

3.3. Pseudocode

Suatu kode atau simbol yang terlihat seperti kode palsu atau kode yang menafsirkan cara menyelesaikan suatu masalah disebut Pseudocode. Pseudocode biasanya digunakan oleh pengembang untuk menulis algoritma dan memakai simbol yang mirip atau identik dengan kode program dalam suatu bahasa pemrograman tertentu. Pseudocode juga memakai kata-kata untuk menjelaskan alur logika suatu masalah, sehingga algoritma yang ditunjukkan dapat diterapkan pada setiap bahasa pemrograman [11]. Adapun pseudocode dari program algoritma dijkstra yang akan diimplementasikan seperti pada Gambar 3.

```
  Fungsi dijkstra(graf, sumber, tujuan) {
      inf ← nilai maksimum sistem
(sys.maxsize)
      node_data ← Dictionary {
          'cost' ← inf
          'pred' ← []
      }
      node_data[sumber]['cost'] ← 0
      visited ← []
      temp ← sumber

      Untuk i dari 0 hingga 11 {
          Jika temp tidak ada di visited {
```

```
            Tambahkan temp ke visited
            min_heap ← []
            Untuk j di graf[temp] {
                Jika j tidak ada di
visited {
                    cost ←
node_data[temp]['cost'] + graf[temp][j]
                    Jika cost <
node_data[j]['cost'] {

node_data[j]['cost'] ← cost

node_data[j]['pred'] ←
node_data[temp]['pred'] + [temp]
                    }
                    Tambahkan (cost, j)
ke min_heap
                }
            }
        }
        heapify(min_heap)
        temp ← min_heap[0][1]
    }
    Cetak "Jarak Terpendek: " +
node_data[tujuan]['cost']
    Cetak "Jalur Terpendek: " +
node_data[tujuan]['pred'] + [tujuan]
}
```

Gambar 3. Pseudocode Program Algoritma Dijkstra

4. HASIL DAN PEMBAHASAN

4.1. Phyton

Python yakni salah satu bahasa pemrograman tingkat tinggi yang dirilis pada tahun 1991. Python dikembangkan oleh Guido van Rossum pada tahun 1989 dan merupakan pengembangan dari bahasa pemrograman ABC [12].

Untuk memastikan bahwa hasil penelitian penulis lakukan benar atau tidak ada kesalahan, bahasa Python digunakan pada algoritma Djikstra. Dalam implementasi ini, penulis menggunakan versi terbaru dari Python, yaitu Python 3.12.6. Dalam versi ini, banyak fitur dan perbaikan telah dilakukan jika dibandingkan dengan versi sebelumnya. Selain itu, versi ini menyediakan beberapa pustaka dan modul lainnya yang sangat berguna untuk menunjang berbagai jenis pengembangan, termasuk penelitian dan pembelajaran, seperti yang dilakukan oleh penulis dalam merupakan hal ini. Penulis menggunakan Python dengan harapan dapat mencapai hasil yang valid dan tepat dari penelitian ini. Selain itu, penggunaan bahasa pemrograman ini juga menjadi cerminan dari penulis yang berkomitmen untuk menggunakan teknologi dan sarana canggih dalam meraih tujuan.

4.2. Implementasi Perhitungan

Implementasi perhitungan rute terpendek dengan algoritma Dijkstra dilakukan dengan menggunakan aplikasi code editor Visual Studio Code. Program yang akan diimplementasikan menggunakan bahasa Python seperti pada Gambar 4.

```
  import sys
  from heapq import heappush, heappop

  def dijkstra(graph, src, dest):
      inf = sys.maxsize

      node_data  =  {node:  {'cost':  inf,
'pred': []} for node in graph}
      node_data[src]['cost'] = 0
      visited = set()
      min_heap = [(0, src)]

      while min_heap:
          current_cost,    current_node    =
heappop(min_heap)
          if current_node in visited:
              continue
          visited.add(current_node)

          if current_node == dest:
              print("Jarak  Terpendek:  "  +
str(node_data[dest]['cost']))
              print("Jalur  Terpendek:  "  +
str(node_data[dest]['pred'] + [dest]))
              return

          for    neighbor,    weight    in
graph[current_node].items():
              if neighbor not in visited:
                  new_cost = current_cost +
weight
                  if       new_cost       <
node_data[neighbor]['cost']:

node_data[neighbor]['cost'] = new_cost

node_data[neighbor]['pred']               =
node_data[current_node]['pred']           +
[current_node]
                      heappush(min_heap,
(new_cost, neighbor))

      print("Tidak  ada  jalur  dari  {}  ke
{}".format(src, dest))

  if __name__ == "__main__":
      graph = {
          'A': {'B': 168},
          'B':  {'A':  168,  'C':  124,  'J':
136},
          'C': {'B': 124, 'E': 61, 'H': 119},
          'D': {'E': 179},
          'E': {'C': 61, 'D': 179, 'F': 32},
          'F': {'E': 32, 'G': 135},
          'G': {'F': 135, 'H': 90, 'M': 146},
          'H': {'C': 119, 'G': 90, 'I': 17},
          'I': {'H': 17, 'J': 124, 'L': 131},
          'J':  {'B':  136,  'I':  124,  'K':
132},
          'K': {'J': 132, 'L': 124},
          'L': {'I': 131, 'K': 124, 'M': 93,
'S': 125},
```

```
          'M': {'G': 146, 'L': 93, 'O': 90,
'R': 66},
          'N': {'O': 95},
          'O': {'M': 90, 'N': 95, 'P': 126},
          'P': {'O': 126, 'Q': 89},
          'Q': {'P': 89, 'R': 58, 'S': 92,
'U': 284},
          'R': {'M': 66, 'Q': 58},
          'S': {'L': 125, 'Q': 92, 'T': 290},
          'T': {'S': 290, 'U': 98},
          'U': {'Q': 284, 'T': 98, 'V': 181},
          'V': {'U': 181},
      }
      source = 'R'
      destination = 'N'
        dijkstra(graph, source, destination)
```

Gambar 4. Kode Program Algoritma Dijkstra

4.3. Hasil

Gambar 5. Output Program Algoritma Dijkstra

Gambar 5 diatas merupakan output dari program yang sebelumnya telah dijalankan menggunakan code editor Visual Studio Code. Terlihat terminal menampilkan 4 output yang berbeda. Pada semua input *source* dimulai dengan titik awal R yakni FK UNIMED. Kemudian input *destination* adalah keempat gerbang yang ada di UNIMED yakni titik A, D, N, dan V. Pada output pertama terlihat jika titik awal dimulai dari R menuju titik akhir A diperoleh jarak terpendek sepanjang 713m dengan jalur atau rute terpendek yang dilalui adalah R-M-G-H-C-B-A. Lalu pada output kedua terlihat jika titik awal dimulai dari R menuju titik akhir D diperoleh jarak terpendek sepanjang 558m dengan jalur atau rute terpendek yang dilalui adalah R-M-G-F-E-D. Selanjutnya pada output ketiga terlihat jika titik awal dimulai dari R menuju titik akhir N diperoleh jarak terpendek sepanjang 251m dengan jalur atau rute terpendek yang dilalui adalah R-M-O-N. Kemudian pada output pertama terlihat jika titik awal dimulai dari R menuju titik akhir V diperoleh jarak terpendek sepanjang 523m dengan jalur atau rute terpendek yang dilalui adalah R-Q-U-V. Berdasarkan keempat output yang dihasilkan, terlihat jika titik awal dimulai dari R menuju titik akhir A merupakan rute terpendek diantara ketiga rute lainnya dengan jarak terpendek sepanjang 251m dengan rute R-M-O-N. Artinya, rute terpendek untuk menuju Fakultas Kedokteran UNIMED adalah dengan masuk melalui titik N (Gerbang 2).

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian yang dilakukan, penerapan Algoritma Dijkstra dengan menggunakan Python berhasil menemukan jalur terpendek dari Fakultas Kedokteran UNIMED menuju salah satu dari empat gerbang kampus UNIMED. Jalur terbaik dan terpendek dari hasil simulasi itu berhasil didapatkan untuk masing-masing gerbang, dimana rute R-M-O-N adalah jalur terpendek dari R (Fakultas Kedokteran UNIMED) menuju N (Gerbang 2). Oleh karena itu, Algoritma Dijkstra dapat dijadikan solusi yang handal dan efisien dalam permasalahan pencarian jalur terpendek di lingkungan kampus. Mengingat keberhasilan yang telah diraih, disarankan agar penelitian dilanjutkan dengan memperluas variabel yang sudah ada. Ini terutama berlaku untuk variabel yang berkaitan dengan perbandingan waktu untuk rute yang berbeda.

Dengan mempertimbangkan faktor-faktor lalu lintas yang mungkin berdampak, penambahan variabel ini akan sangat membantu meningkatkan akurasi yang dihasilkan oleh algoritma. Aplikasi yang telah dibuat juga dapat ditingkatkan melalui penambahan titik lokasi tambahan ke peta kampus. Metode ini memungkinkan penggabungan sistem navigasi kampus yang lebih efektif, yang memungkinkan pengguna menemukan cara tercepat dan paling efisien untuk mencapai tujuan mereka.

DAFTAR PUSTAKA

[1] I. Arthalia Wulandari and P. Sukmasetyan, “Implementasi Algoritma Dijkstra untuk Menentukan Rute Terpendek Menuju Pelayanan Kesehatan,” *J. Ilm. Sist. Inf.*, vol. 1, no. 1, pp. 30–37, 2022, doi: 10.24127/jisi.v1i1.1953.

[2] R. Y. F. Hutapea, T. D. Pramesthy, R. P. Situmorang, and A. A. Rosalia, “Identifikasi Peralatan Navigasi Dan Keselamatan Yang Digunakan Di Km Dioskuri 8,” *J. Kemaritiman Indones. J. Marit.*, vol. 3, no. 1, pp. 1–10, 2022, doi: 10.17509/ijom.v3i1.46728.

[3] M. Muharrom, “Implementasi Algoritma Dijkstra Dalam Penentuan Jalur Terpendek Studi Kasus Jarak Tempat Kuliah Terdekat,” *Indones. J. Bus. Intell.*, vol. 3, no. 1, p. 25, 2020, doi: 10.21927/ijubi.v3i1.1229.

[4] A. A. Kusumastuti and I. K. Budayasa, “Segitiga Pelangi pada Pewarnaan-Sisi Graf,” *MATHunesa J. Ilm. Mat.*, vol. 8, no. 1, pp. 35–44, Apr. 2020, doi: 10.26740/mathunesa.v8n1.p35-44.

[5] G. Stevans, A. Eleazar Rangin, J. Timothy Souk, J. Pratama, and V. Handrianus Pranatawijaya, “Implementasi Algoritma Dijkstra Dalam Menentukan Rute Wisata Di Kota Palangka Raya,” *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 3, pp. 3518–3523, 2024, doi: 10.36040/jati.v8i3.9701.

[6] Y. Suharya and J. Anwar Sani, “Analisis Kinerja Implementasi Algoritma Dijkstra Untuk Mencari Rute Terdekat Dari Baleedah Ke Perpustakaan

Kawaluyaan Dengan Menggunakan Python," vol. 09, pp. 65–69, 2022.

[7] 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, doi: 10.26905/jtmi.v6i1.3837.

[8] A. M. Inayah, N. Cintya Resti, and I. Kediri, "Analisa Perbandingan Algoritma Floyd-Warshall Dan Algoritma Dijkstrauntuk Penentuan Rute Terdekat," *J. Ilm. Mat. Realis. (JI-MR*, vol. 4, no. 2, pp. 146–155, 2023.

[9] I. W. W. Karsana and G. S. Mahendra, "Sistem Informasi Geografis Pemetaan Lokasi Puskesmas Menggunakan Google Maps Api Di Kabupaten Badung," *J. Komput. dan Inform.*, vol. 9, no. 2, pp. 160–167, 2021, doi: 10.35508/jicon.v9i2.5214.

[10] R. Rosaly and A. Prasetyo, "Flowchart Beserta Fungsi dan Simbol-Simbol," *J. Chem. Inf. Model.*, vol. 2, no. 3, pp. 5–7, 2020.

[11] Halimatussya'diyah Purba and Yahfizham Yahfizham, "Konsep Dasar Pemahaman Algoritma Pemrograman," *J. Arjuna Publ. Ilmu Pendidikan, Bhs. dan Mat.*, vol. 1, no. 6, pp. 290–301, 2023, doi: 10.61132/arjuna.v1i6.356.

[12] R. Suherman[1] and N. R. Kurnianda, "Optimization of VSAT IP Installation Routes using the Dijkstra Algorithm," *Ijctjournal.Org*, vol. 8, no. 1, pp. 153–157, 2021, [Online]. Available: http://www.ijctjournal.org/volume8/issue1/ijct-v8i1p16.pdf