PERANCANGAN APLIKASI PENCARIAN RUTE PERJALANAN ANGKUTAN KOTA (ANGKOT) DI KOTA BANDUNG BERBASIS WEB MENGGUNAKAN ALGORITMA A*

Manuel Kristo[1]
Yusup Jauhari Shandi[2]

[1,2]Sekolah Tinggi Manajemen Informatika dan Komputer LIKMI
Jl. Ir. H. Juanda Bandung 40132

[1]manuelkristo12@gmail.com
[2]ujshandi@gmail.com

ABSTRAK

Transportasi dapat membuat perjalanan menjadi mudah, membuat bepergian dari satu tempat ke tempat lainnya menjadi semakin mudah dan cepat. Transportasi umum, salah satunya angkutan kota (Angkot) muncul sebagai solusi bagi masyarakat yang tidak memiliki kendaraan untuk bepergian serta membantu mengurangi beban kemacetan di perkotaan. Untuk menentukan rute yang terpendek dapat menggunakan algoritma A* (A star). Algoritma A* dapat membuat pencarian rute yang menjadi lebih optimal. Penelitian ini merancang aplikasi pencarian rute perjalanan angkutan kota (objek penelitian di kota Bandung) berbasis web dengan menggunakan Algoritma A*.

*Kata kunci : rute perjalanan, angkot, algoritma A**

1. PENDAHULUAN

Transportasi dibagi menjadi dua yaitu angkutan umum dan angkutan pribadi. Angkutan umum muncul sebagai solusi bagi masyarakat yang tidak memiliki kendaraan untuk beraktivitas. Salah satu moda transportasi di Kota Bandung yaitu Angkutan Kota (angkot). Angkot dapat menaikan atau menurunkan penumpang di mana saja tidak seperti Bus yang mempunyai halte khusus sebagai tempat perhentian. Angkot merupakan transportasi umum dalam kota dengan rute yang sudah ditentukan. Angkot mempunyai berbagai trayek atau rute dalam mencapai tujuan dan mempunyai tempat pemberhentian akhir yang sudah ditentukan. Dengan demikian rute bisa digambarkan menyerupai sebuah graph. Dalam penggunaan Angkot, sebagaian masyarakat masih ada yang mengalami kesulitan dan masih belum memahami dalam mencari rute atau tempat yang akan dituju.

Menentukan rute yang terpendek dapat menggunakan algoritma A star. Algoritma A* (A Star) dikenal sebagai salah satu algoritma yang paling sering digunakan untuk pencarian jalur (*path finding*) dan penerusan grafis (*graph traversal*), yaitu proses ploting jalur yang paling efisien antar titik, yang disebut dengan nodes. Dengan menggunakan algoritma A* dapat membuat proses pencarian rute terpendek menjadi lebih cepat dibandingkan dengan menggunakan Algoritma *Dijkstra*. Sehingga pencarian rute yang menjadi lebih optimal.

2. TINJAUAN PUSTAKA

2.1. Angkutan Kota

Dalam jurnal ditulis oleh Yahya Peranginangin, Angkutan Kota (Angkot) adalah transportasi umum yang sering digunakan di Kota bandung. Jenis kendaraan yang digunakan oleh angkot yaitu menggunakan minibus dengan kapasistas 12-15 penumpang [7].

2.2. Graph

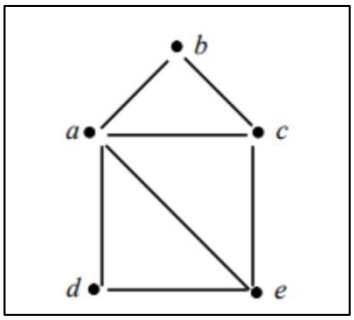
Ida Bagus menjelaskan dalam jurnalnya bahwa *Graph* merupakan mata pelajaran dasar matematika diskrit dan ilmu komputer. Merupakan objek diskrit dan hubungan di antara mereka. Representasi visual dari *graph* adalah dengan menggunakan *node*, bulatan atau titik untuk merepresentasikan objek, dan hubungan antar objek diwakili oleh garis atau sisi [1]. Dalam jurnal yang ditulis oleh Ida Bagus, menjelaskan bahwa *graph* adalah pasangan himpunan *vertex* atau simpul dan *edges* atau sisi, yang dimana setiap sisi berhubungan dengan satu atau 2 buah simpul [1].

Berdasarkan jurnal yang ditulis oleh Mira Kusmira, graph yang memiliki sisi (node) dan simpul (edge) pada suatu graph, graph dapat digolongkan menjadi dua jenis yaitu :

a. *Simple Graph* (Graf Sederhana)
   Graph Sederhana adalah graph yang tidak mengandung simpul atau sisi ganda.
b. *Unsimple Graph* (Graf tak Sederhana)
   Graph tak sederhana merupakan graph yang mengandung simpul atau sisi ganda. Ada dua macam graph tak sederhana, yaitu : multigraph (graf ganda) dan pseudograph (graf semu) [5].

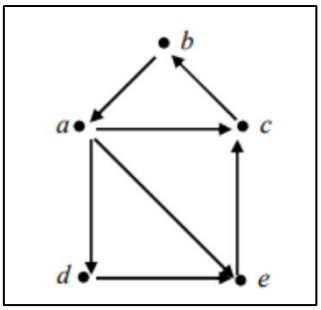
Dalam *graph*, berdasarkan orientasi arah pada sisi, maka graph dibagi menjadi dua macam yaitu [10] :

a. *Undirected graph* (graf tidak berarah)
   Graf tak berarah yaitu sisi yang tidak memiliki orientasi arah. Urutan dari pasangan simpul pada suatu graph tidak akan diperhatikan. Ilustrasi pada Gambar 1.



Gambar 1. Graf Tidak Berarah

b. *Directed graph* (graf berarah)
   Graf berarah adalah graph yang setiap sisi diberikan orientasi arah. Sisi sisinya yang berarah ini biasa disebut busur. Ilustrasi pada Gambar 2.



Gambar 2. Graf Berarah

2.3. Algoritma A*

Dalam Buku "Object Oriented Programming and Data Structures" yang ditulis oleh E. Balagurusamy, menjelaskan bahwa algoritma dapat didefinisikan sebagai sebuah prosedur langkah demi langkah dalam memberikan sebuah solusi dalam suatu masalah [1].

Sedangkan menurut Kani dalam buku "Algoritma dan Pemrograman" menjelaskan bahwa, algoritma adalah sebuah urutan operasi yang disusun secara logis dan sistematis untuk menyelesaikan suatu masalah buat menghasilkan suatu hasil atau output tertentu [4].

Selain algoritma *Dijkstra* dan algoritma *Floyd-Warshall*, terdapat Algoritma A* untuk mencari jalur tercepat. Algoritma A* digunakan sebagai salah satu algoritma pencarian terbaik (*best first search*) dalam mencari jalur terpendek dengan melakukan perhitungan pada jalur *node* awal menuju *node* akhir.

Dalam jurnal M. Azan Cahyadi menjelaskan bahwa Algoritma Dijkstra sebagai versi ringkas dari algoritma A* [2]. Algoritma ini meminimalkan total biaya jalur (*least-cost path*), dan saat kondisi yang sesuai dan tepat akan memberikan jalur yang terbaik dalam waktu yang optimal. Algoritma A* (A Star) dikenal sebagai salah satu algoritma yang paling sering digunakan untuk pencarian jalur (*path finding*) dan penerusan grafis (*graph traversal*), yaitu proses ploting jalur yang paling efisien antar titik, yang disebut dengan *nodes* [9].

Algoritma A* memiliki *OPEN list* dan *CLOSED list*. *OPEN list* adalah tempat untuk menyimpan data simpul yang mungkin diakses dari *starting point* maupun simpul yang sedang dijalankan. *OPEN list* juga berpeluang untuk terpilih sebagai simpul terbaik (*best node*). Sedangkan *CLOSED list* adalah menyimpan data simpul yang sudah pernah diakses dan sudah pernah terpilih sebagai simpul terbaik (*best node*) [3].

Untuk Menghitung biaya yang diperhitungkan didapat dari biaya yang sebenarnya ditambah dengan biaya perkiraan. Dalam notasi matematika dituliskan berupa Rumus 1.

$$f(n) = g(n) + h(n) \ldots\ldots (1)$$

Keterangan :
f(n) = Biaya evaluasi
g(n) = jarak koordinat ke titik n
h(n) = nilai heuristik antar koordinat

Dari Rumus 1, g(n) adalah jarak koordinat antara titik awal dan titik tujuan, dimana nilai g(n) di peroleh dari jarak pada peta dikalikan dengan skala peta. Jika ditulis berupa Rumus 2.

$$g(n) = \text{jarak pada peta x skala peta [6]} \ldots\ldots (2)$$

Dalam metode pencarian, *heuristik* diartikan sebagai fungsi yang memberikan suatu nilai berupa biaya perkiraan dari suatu solusi. Pencarian *heuristik* merupakan proses pencarian secara selektif dan proses pencarian yang memiliki kemungkinan paling besar,

akan tetapi kemungkinan mengorbankan kelengkapan (*completeness*). Pencarian *heuristik* akan diterapkan dalam fungsi *heuristik* [8].

Dalam jurnal yang ditulis oleh Kiki Setiawan menjelaskan bahwa. *Euclidean distance* adalah perhitungan dari 2 titik jarak yang saling berhubungan [8]. *Euclidean* diperoleh berdasarkan jarak langsung yang bebas hambatan untuk mendapatkan nilai dari panjang garis diagonal. Dalam *Euclidean*, akan berkaitan dengan *Teorema Phytagoras* dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Tetapi bisa disederhanakan jika diterapkan pada dimensi yang lebih tinggi [8]. Maka formulasi *Euclidean* dapat dituliskan berupa Rumus 3.

$$d = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \text{ ….. (3)}$$

Sehingga untuk memperoleh nilai h(n), dapat di tunjukan dengan Rumus 4 [6].

$$h(n) = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \text{ ….. (4)}$$

Keterangan :
h(n) : nilai heuristik untuk node/titik n
X_n : nilai koordinat x dari node/titik n
Y_n : nilai koordinat y dari node/titik n
X_{goal} : nilai koordinat x dari node/titik tujuan
Y_{goal} : nilai koordinat y dari node/titik tujuan.

3. ANALISIS DAN PERANCANGAN

Aplikasi pencarian rute dengan menggunakan algortima A* sebagai metode pencariannya dan disajikan dalam bentuk aplikasi berbasis web. Aplikasi ini menggunakan rute jalur angkot sebagai trayek. Dalam mencapai tempat tujuan, rute akan memilih jalur mana yang akan di gunakan untuk mencapai tempat tujuan. Hasil dari rute pencariannya akan ditampilkan pada maps dari *Google Maps Platform* dengan tipe peta yaitu *Roadmap*.
Sebelum melakukan pencarian rute, admin harus menginput data trayek angkot yang akan digunakan oleh user. Dalam menginput data trayek, admin harus login agar data yang dimasukan *valid*. Untuk menginput data trayek, berisi nama trayek angkot, node awal (keberangkatan), dan node akhir (tujuan). Admin juga dapat menghapus data trayek angkot jika tidak dipakai atau adanya kesalahan input. Admin pun dapat melihat data trayek yang ada di aplikasi.

Pada aplikasi ini, user dapat melihat data trayek angkot dan bisa melakukan pencarian rute yang akan dituju. Untuk menentukan rute yang akan diambil dan akan dilalui, digunakan metode algoritma A*. Metode tersebut dapat mencari jalan agar sampai ke tempat tujuan, mencari jalan yang terpendek dan mencari waktu terpendek.

Pada penelitian ini Algoritma A* akan diimplementasikan dengan alur sebagai berikut :

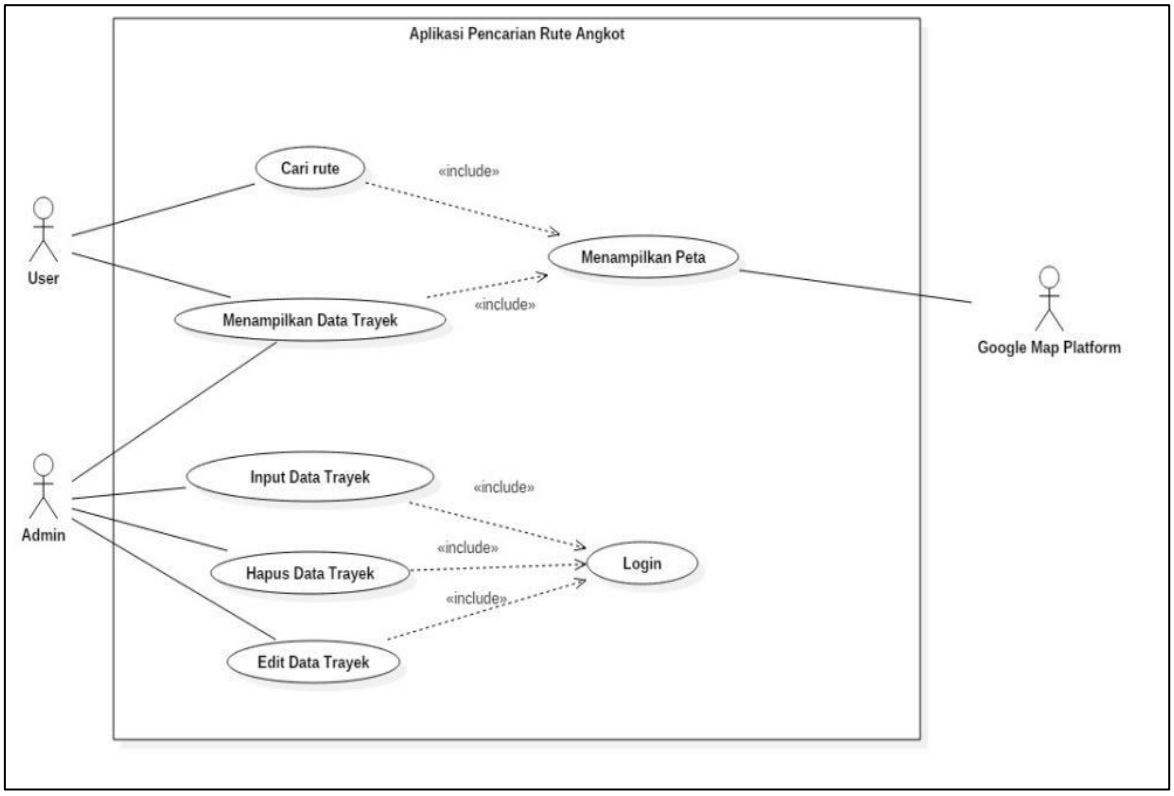
a. Untuk mencari rute tujuan, dimulai dengan menentukan titik awal dan titik tujuan.
b. Dari kedua titik langkah point (a) dapat menetapkan node awal dan node akhir dari trayek tersebut. Node awal akan berfungsi sebagai *current node* karena hanya terdapat 1 simpul.
c. Semua *node* yang terhubung dengan *current node* akan masuk kedalam *OPEN list. Current node* sendiri pada *OPEN list* akan masuk kedalam *CLOSE list*.
d. Dari *OPEN list* akan dilakukan perhitungan Algoritma A*. Hasil dari perhitungan tersebut akan dicari dengan f(n) dan hasil akan disimpan. *Node* dengan f(n) terkecil akan pindah ke dalam *CLOSED list*. Tahap diatas akan terus diulang sampai rute yang ditentukan sudah menemukan *node* akhir. Untuk penyimpanan rute, akan diurut dimulai

dari belakang yaitu urutan mulai dari *node* akhir ke *node* tetangga dalam *close list*. Rute terpendek dengan harga murah akan didapatkan dari hasil perhitungan agloritma A*.

Untuk lebih jelas, dapat simulasikan seorang penumpang akan menuju ke STMIK LIKMI dan berangkat dari Cimindi, maka di web kita menulis tujuan awal yaitu Cimindi dan tujuan akhir yaitu STMIK LIKMI. Ketika textbox sudah terisi maka kita mengklik tombol search. Web akan melakukan proses untuk mencari jalur angkot yang akan dipakai menggunakan algoritma A* dan hasil nya akan di munculkan pada peta. Hasil rute dari gunung batu ke STMIK LIKMI akan memunculkan garis berwarna pada jalan dan memunculkan icon angkot beserta nama trayek angkot. Untuk trayek angkot yang digunakan ada 3 yaitu, Angkot jurusan St. Hall - Cimahi, Angkot Jurusan Ciroyom - Antapani, dan Angkot jurusan Dago - St. Hal

3.1. Use Case Diagram

Dalam UML, seluruh model perancangan sistem berawal dari *use case diagram*. Fitur utama kelengkapan aplikasi digambarkan dalam diagram ini.

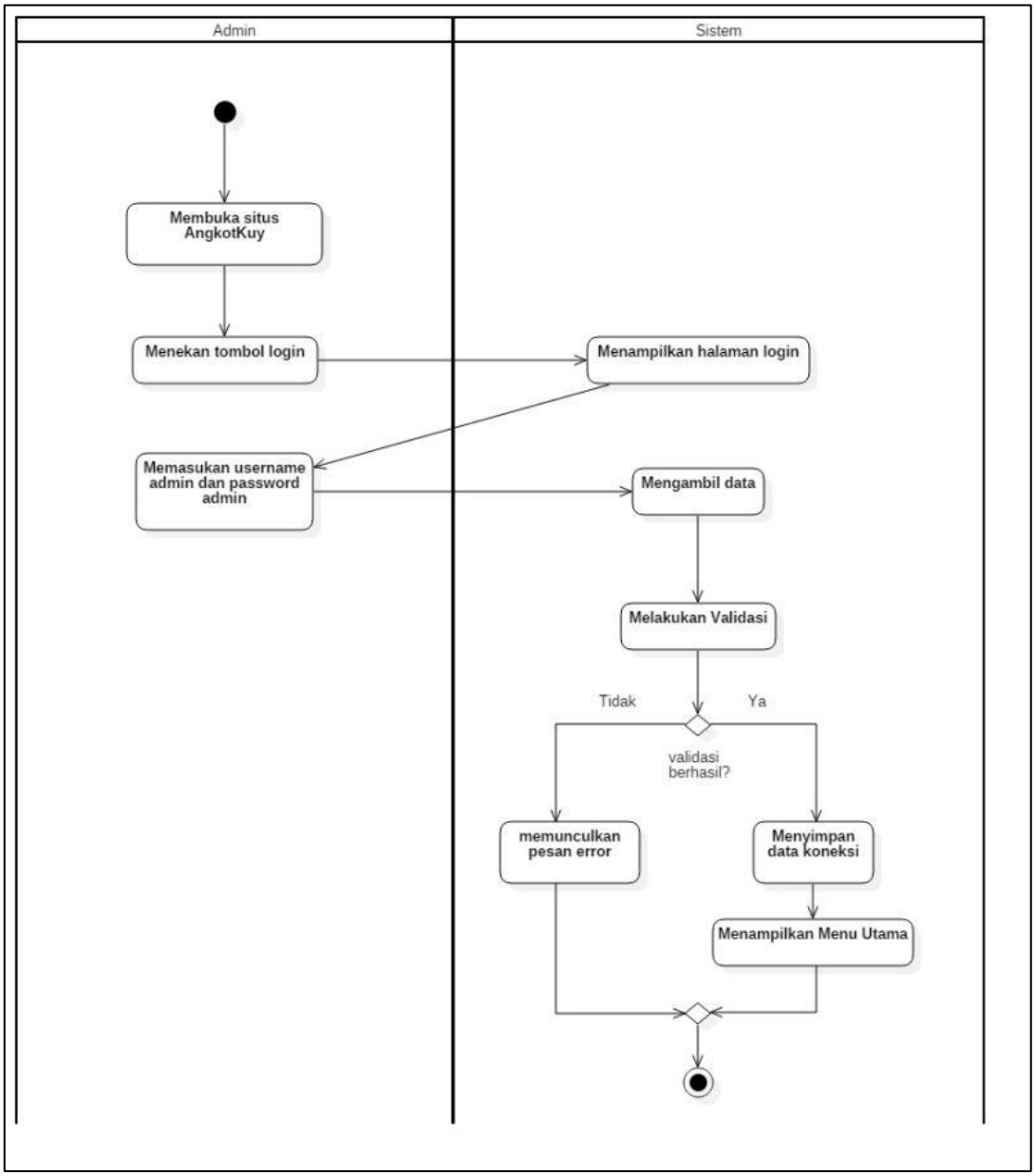


Gambar 3. Use Case Diagram

Aktor terdiri dari :
1. User : pengguna aplikasi (umum)
2. Admin : pengguna aplikasi (*backend*, mengelola data-data pendukung aplikasi)
3. *Google Map Platform* : layanan penyedia peta grafis

3.2. Activity Diagram Login

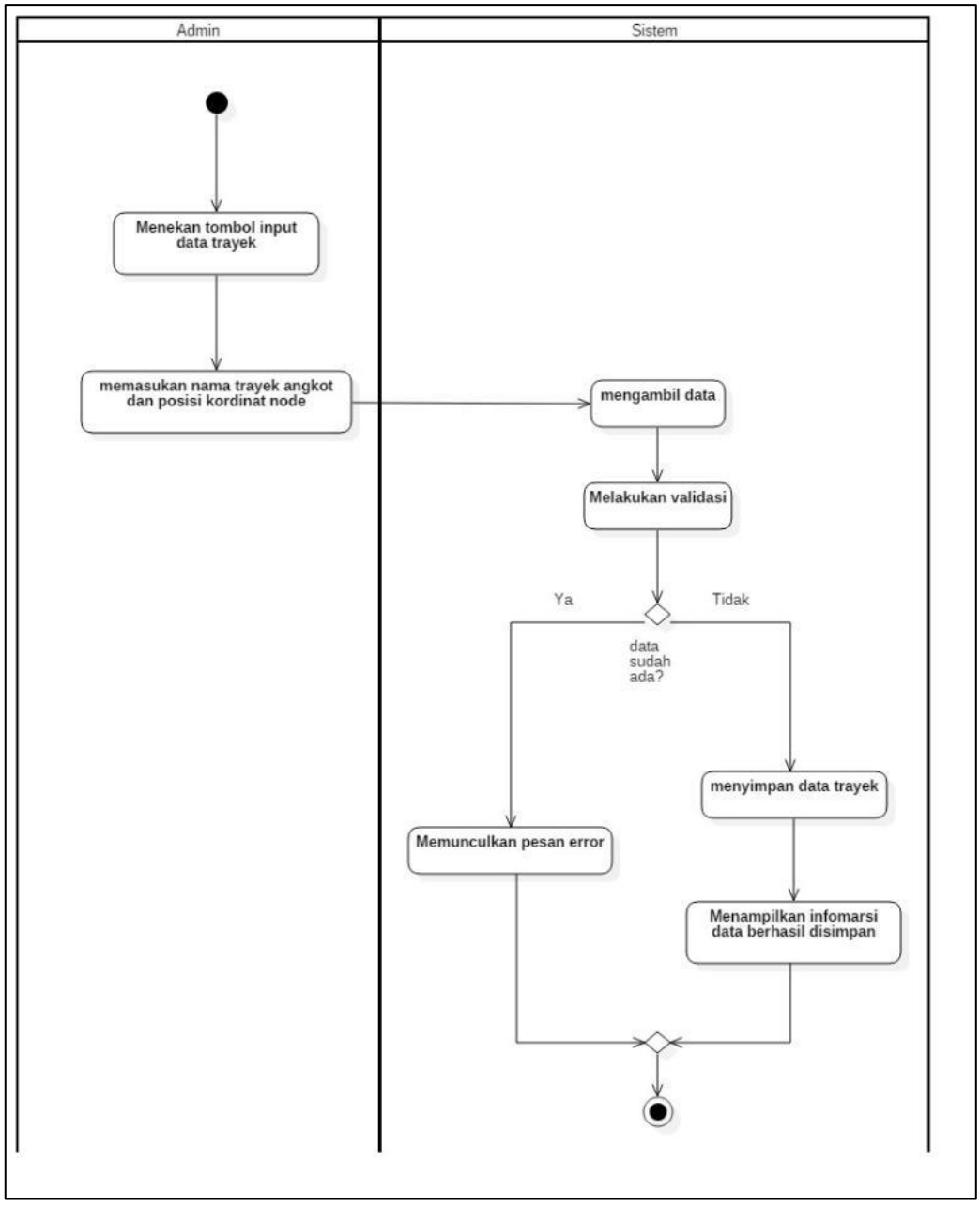
Pada Gambar 4 ditampilkan *activity diagram* yang menjelaskan alur proses login untuk melakukan otentikasi terhadap user admin.



Gambar 4. Activity Diagram Proses Login

3.3. Activity Diagram Input Data Trayek

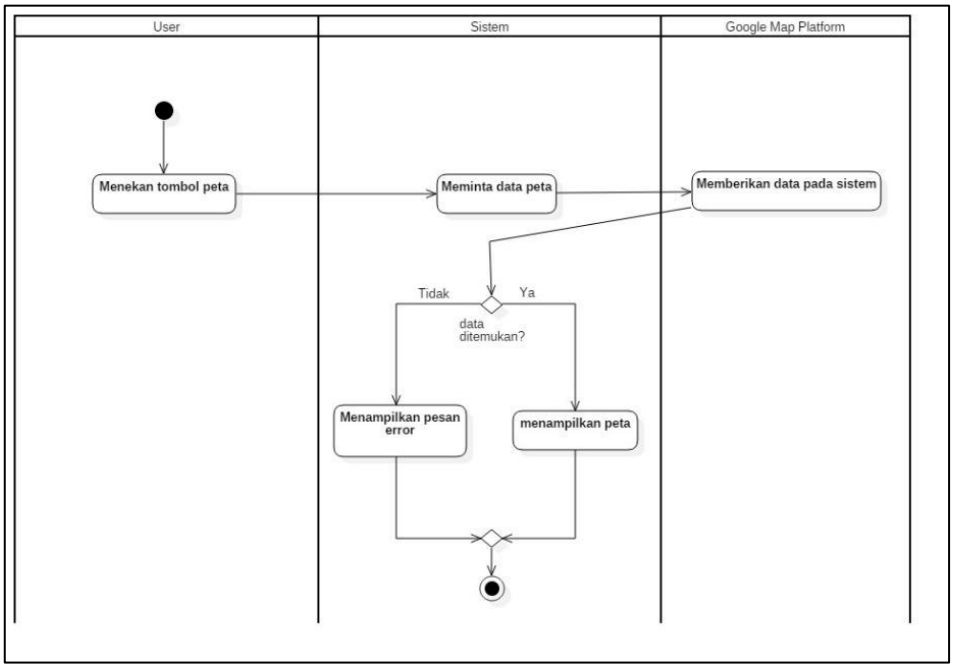
Pada Gambar 5 ditampilkan *activity diagram* yang menjelaskan alur proses untuk memasukkan data trayek angkutan kota (angkot).



Gambar 5. Activity Diagram Proses Input Data Trayek

3.4. Activity Diagram Menampilkan Peta

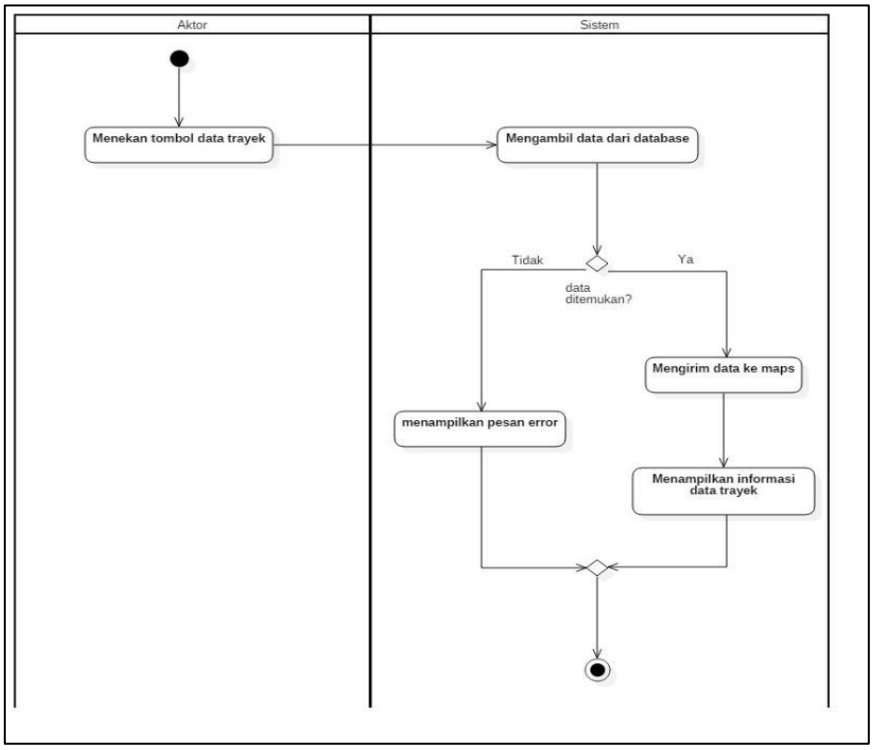
Gambar 6 merupakan *acitivity diagram* yang menjelaskan alur proses untuk menampilkan peta.



Gambar 6. Activity Diagram Proses Menampilkan Peta

3.5. Activity Menampilkan Data Trayek

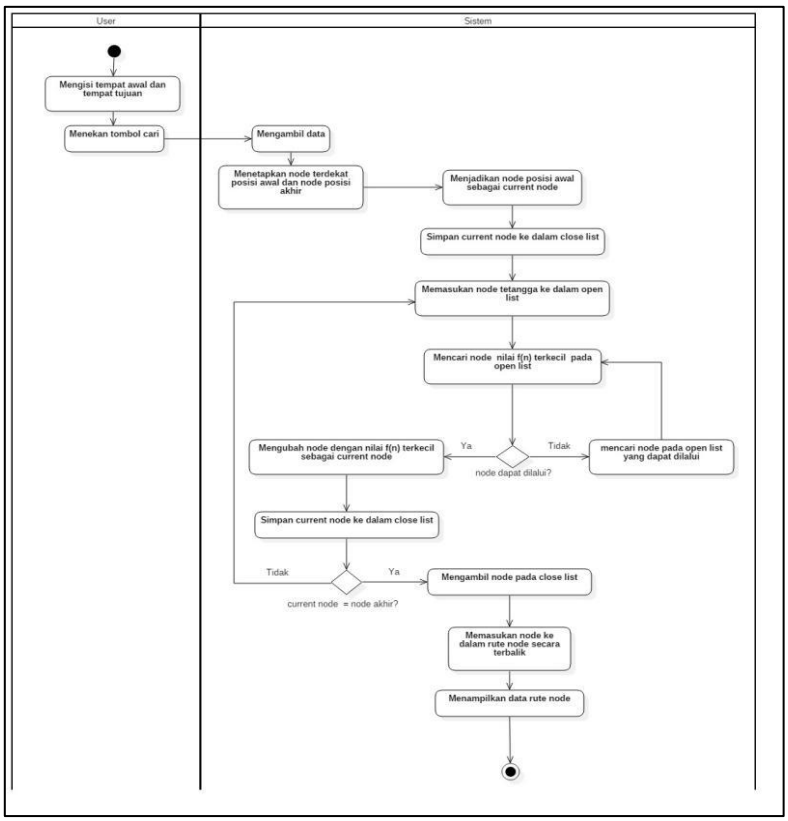
Gambar 7 merupakan *acitivity diagram* yang menjelaskan alur proses untuk menampilkan data trayek.



Gambar 7. Activity Diagram Proses Menampilkan Data Trayek

3.6. Activity Diagram Cari Rute

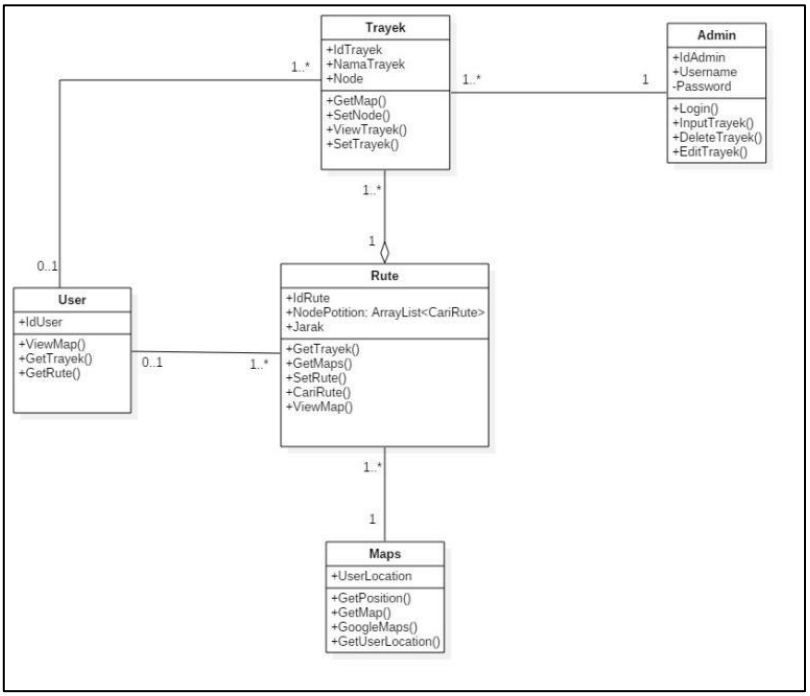
Gambar 8 merupakan *acitivity diagram* yang menjelaskan alur proses mencari rute.



Gambar 8. Activity Diagram Proses Mencari Rute

3.7. Class Diagram

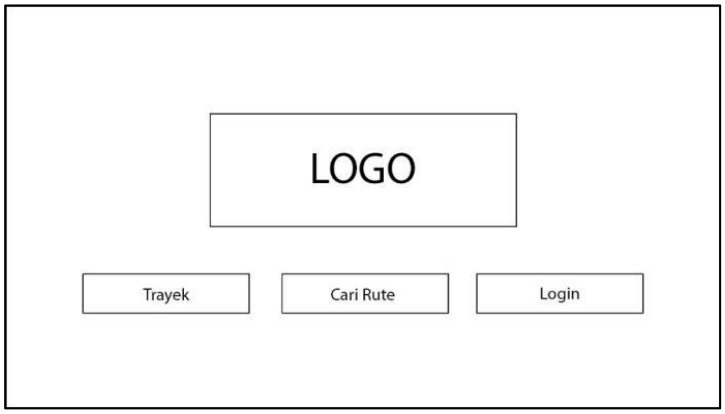
Untuk membuat aplikasi pada penelitian ini dibutuhkan empat buah class yang digambarkan dengan *class diagram* pada Gambar 9.



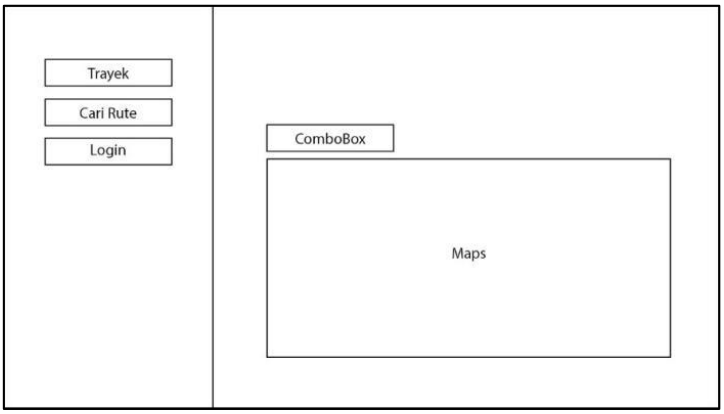
Gambar 9. Class Diagram

4. Rancangan Antarmuka

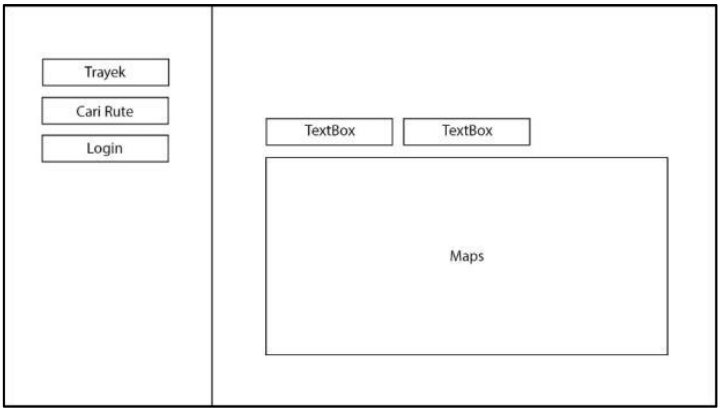
Pada Gambar 10, 11, dan 12 ditampilkan rancangan antarmuka aplikasi pencarian rute perjalanan angkot berupa tampilan untuk halaman utama dan menu aplikasi, halaman mengenai trayek angkot, serta halaman untuk pencarian rute angkot.



Gambar 10. Rancangan Antarmuka Halaman Utama



Gambar 11. Rancangan Antarmuka Trayek Angkot



Gambar 12. Rancangan Antarmuka Cari Rute

5. KESIMPULAN

Pada penelitian mengenai perancangan aplikasi pencarian rute perjalanan angkutan kota dengan menggunakan Algoritma A* dapat disimpulkan :

1. Algoritma A* dapat digunakan sebagai algoritma untuk mencari rute terbaik atau jalur tercepat.
2. Pada aplikasi ini metode dari Algoritma A* digunakan untuk mencari jalan agar sampai ke tempat tujuan, mencari jalan yang terpendek dan mencari waktu terpendek dari rute-rute jalur angkot yang sudah diinput.
3. Untuk pengembangan lebih lanjut disarankan menggunakan algoritma lain selain algoritma A* untuk menentukan jalur (rute) yang terbaik. Dan juga dapat membandingkan algoritma lain tersebut apakah lebih baik dalam penentuan jalur tercepat.

DAFTAR PUSTAKA

[1] Balagurusamy, E. (2014) Object Oriented Programming and Data Structures. McGraw Hill Education (India) Private Limited. Available at: http://gen.lib.rus.ec/book/index.php?md5=7DC3E9B174E9E8EBF9351E17571F8EB3

[2] Cahyadi, M. A., P, M. A. B. and Widhiarso, W. (2012) 'Perbandingan Algoritma a * , Dijkstra Dan Floyd Warshall Untuk Menentukan Jalur Terpendek Pada Permainan " Bacteria Defense "', Perbandingan Algoritma a * , Dijkstra Dan Floyd Warshall Untuk Menentukan Jalur Terpendek Pada Permainan " Bacteria Defense ", (x), p. 10.

[3] Gede Wahyu Antara Dalem, I. B. (2018). Penerapan Algoritma A* (Star) Menggunakan Graph Untuk Menghitung Jarak Terpendek. Jurnal RESISTOR (Rekayasa Sistem Komputer), 1(1), 41–47. https://doi.org/10.31598/jurnalresistor.v1i1.253

[4] Kani (2020) 'Modul Pengantar Algoritma dan Pemrograman', Algoritma dan Bahasa Pemrograman, 1, pp. 1–36. Available at: https://pustaka.ut.ac.id/lib/wpcontent/uploads/pdfmk/MSIM4203-M1.p

[5] Kusmira, M. and Taufiqurrochman (2017) 'Pemanfaatan Aplikasi Graf Pada Pembuatan Jalur Angkot 05 Tasikmalaya', Seminar Nasional Sains dan Teknilogi, (November), pp.1–6

[6] Nuryoso, Y. H., Pradjoko, & Lelah. (2020). Penerapan Algoritma A * pada Pencarian Rute Terpendek pada. 8(1), 21–35.

[7] Peranginangin, Y., Andi, A. and Sisilia, K. (2018) 'Route Recommendation Using Community Detection Algorithm (Case: Kota Bandung)', in 2018 6th International Conference on Information and Communication Technology (ICoICT). IEEE, pp. 113–118. doi: 10.1109/ICoICT.2018.8528740

[8] Setiawan, K., Supriyadin, S., Santoso, I., & Buana, R. (2018). Menghitung Rute Terpendek Menggunakan Algoritma A* Dengan Fungsi Euclidean Distance. Seminar Nasional Teknologi Informasi Dan Komunikasi, 2018(ISSN: 2089-9815), 70–79. https://fti.uajy.ac.id/sentika/publikasi/makalah/2018/9.pdf

[9] Syukriah, Y., Falahah, F., & Solihin, H. (2016). Penerapan Algoritma a* (star) untuk Mencari Rute Tercepat dengan Hambatan. Seminar Nasional Telekomunikasi Dan informatika (SELISIK), 1, 219–224.

[10] Wirdasari, D. (2011) 'Teori graph dan implementasinya dalam ilmu komputer', Teori graph dan implementasinya dalam ilmu komputer, 10(1), pp. 23–34.