Yulsilviana, E., Adytia, P. and Riandika, I.N. (2021) "Pencarian Driver Dry Clean Terdekat dengan Metode Haversine Formula"Sebatik, 25(1).
Submitted: 2021/02/28 Accepted: 2021/04/30 Published: 2021/06/01

Sebatik Vol. 25 No. 1 Juni 2021
ISSN: 1410-3737(p) 2621-069X(e)

DOI:10.46984/sebatik.v25i1.1210

PENCARIAN DRIVER DRY CLEAN TERDEKAT DENGAN METODE *HAVERSINE* FORMULA

Ekawati Yulsilviana[2)], Pitrasacha Adytia[1)], dan **I Nengah Riandika[3)]**

[1]Manajemen Informatika, STMIK Widya Cipta Dharma
[2]Sistem Informasi, STMIK Widya Cipta Dharma
[3]Teknik Informatika, STMIK Widya Cipta Dharma
[1,2,3]Jl. M. Yamin No.25, Samarinda, 75123
E-mail : ekawati@wicida.ac.id[2)], pitra@wicida.ac.id[1)], riandikatkj@gmail.com[3)]

ABSTRAK

Sistem pemesanan *dry cleaning* konvensional tidak bisa menemukan *driver* terdekat dengan pemesan. Penelitian ini bertujuan untuk membangun sistem untuk membantu pencarian *driver* terdekat dengan pemesan agar cepat dalam mengambil pakaian kotor dan sebaliknya mencari driver terdekat dari *dry cleaning* jika pakaian sudah selesai**.** Penelitian ini menerapkan metode *haversine formula* untuk pencarian *driver* terdekat, Google *Maps* sebagai pembangun peta digital, dan dikembangkan berbasis *mobile*. Sistem perancangan pada penelitian ini menggunakan *Unified Modeling Language* (UML) yang terdiri dari *Use Case Diagram*, *Activity Diagram, Class Diagram*, *Sequence Diagram*, dan *Deployment Diagram.* Pengujian sistem dilakukan dengan menggunakan *white box testing* dan *beta testing*.

Kata Kunci: *Metode Haversine Formula, Driver Terdekat, Dry Cleaning, Mobile.*

1. PENDAHULUAN

Sistem Informasi Geografis (SIG) merupakan sebuah sistem komputer yang memiliki kemampuan untuk mengambil, menyimpan, menganalisa, dan menampilkan informasi dengan referensi geografis (Setiawan, 2016). Teknologi sistem informasi geografis yang banyak digunakan saat ini ialah pencarian rute terdekat. Rute terdekat menjadi pilihan bagi setiap pengembang aplikasi seperti Go-jek dan Grab karena berguna untuk mencari *driver* terdekat untuk pelanggannya, hal ini bisa menghemat banyak waktu bagi mereka untuk sampai di tempat tujuan (Arifin, 2018).

Salah satu usaha jasa yang saat ini pertumbuhan bisnisnya sedang berkembang adalah usaha *dry cleaning*, proses jasa pembersihan pakaian dan tekstil dengan menggunakan pelarut selain air. Perubahan gaya hidup dan tuntutan kesibukan yang menjadikan sebagian besar masyarakat lebih memilih menggunakan jasa *dry cleaning* untuk meringankan pekerjaan mereka. Di Samarinda usaha *dry cleaning* ini sendiri sudah tidak jarang kita jumpai di sekitar wilayah rumah kita atau di wilayah tengah perkotaan maupun daerah pelosok.

Pencarian *driver* terdekat ketika pelanggan *dry cleaning order* sangat diperlukan karena pelanggan tidak perlu menunggu lama kedatangan *driver* untuk *pickup* atau *delivery* pakaian. Dengan adanya aplikasi ini juga akan memudahkan para *driver* untuk mencari lokasi pelanggan, dengan menampilkan peta yang dilengkapi rute menuju lokasi.

Adapun proses pencarian *driver* terdekat dilakukan dengan menerapkan metode *Haversine Formula* yang merupakan salah satu metode yang dapat digunakan untuk menghitung jarak antara dua titik, berdasarkan posisi garis lintang (*latitude)* dan posisi garis bujur (*longitude)* sebagai *variabel* inputan (Wicaksana, 2017). *Haversine Formula* ini memiliki tingkat kesalahan sebesar 0,5 % dengan mengambil garis lurus antara objek 1 dengan objek 2 dan mengabaikan tikungan (Setiawan, 2016; Bakhroin, 2020). Karena *Haversine Formula* mengabaikan tikungan maka untuk mengantisipasinya dengan memanfaatkan Google API untuk mendapatkan *direction* atau lintasan, lalu setiap lintasan dihitung menggunakan *Haversine Formula* dan di total semua lintasan sehingga mendapatkan jarak *driver* terdekat dengan tepat dan tidak mengabaikan tikungan.

2. RUANG LINGKUP PENELITIAN

Dalam penelitian ini permasalahan mencakup:

1. Dapat dijalankan pada *smartphone* berbasis sistem operasi android minimal pada versi 4.1 (Jelly Bean).
2. Menggunakan metode *Haversine Formula* untuk pencarian posisi terdekatnya.
3. Memanfaatkan Google API untuk mendapatkan titik-titik lintasan, titik *driver* maupun titik *customer.*

3. BAHAN DAN METODE

Dalam penelitian ini diperlukan suatu konsep dalam merumuskan definisi yang menunjang kegiatan penelitian, baik teori dasar maupun teori umum.

3.1 Layanan Berbasis Lokasi

Location Based Service (LBS) atau layanan berbasis lokasi adalah istilah umum yang digunakan untuk menggambarkan teknologi yang digunakan untuk menemukan lokasi perangkat yang digunakan (Putra, 2017). Teknologi LBS ini terdiri atas perangkat-

perangkat yang digunakan untuk mengumpulkan, menyimpan, menganalisa dan mendistribusikan data dan informasi pada berdasarkan sistem koordinat geografik bumi secara *real-time*. Identifikasi koordinat pengguna memungkinkan LBS untuk menyediakan layanan bagi pengguna perangkat *mobile* (Budiman, 2016).

3.2 *Global Positioning System* (GPS)

GPS (*Global Positioning System*) adalah sistem navigasi yang berbasiskan satelit yang saling berhubungan yang berada di orbitnya. Untuk dapat mengetahui posisi seseorang maka diperlukan alat yang diberi nama GPS *receiver* yang berfungsi untuk menerima sinyal yang dikirim dari satelit GPS. Posisi diubah menjadi titik yang dikenal dengan nama *Way-point* nantinya akan berupa titik-titik koordinat lintang dan bujur dari posisi seseorang atau suatu lokasi kemudian di layar pada peta elektronik (Tafa, 2018). GPS adalah satu-satunya sistem satelit navigasi *global* untuk penentuan lokasi, kecepatan, arah, dan waktu yang telah beroprasi secara penuh di dunia saat ini (Felisitas, 2017).

3.3 Google Maps API

Google Maps API merupakan pengembangan teknologi dari Google yang digunakan untuk menanamkan Google Map di suatu aplikasi yang tidak dibuat oleh Google (Prastyo, 2016), Google Maps API adalah suatu *library* yang berbentuk javascript yang berguna untuk memodifikasi peta yang ada di Google Maps sesuai kebutuhan. Dalam perkembangannya Google Maps API diberikan kemampuan untuk mengambil gambar peta statis. Melakukan geocoding, dan memberikan penuntun arah. Google Maps API bersifat gratis untuk publik (Fajriansyah, 2019).

3.4 *Haversine Formula*

Haversine Formula adalah persamaan penting dalam sistem navigasi, nantinya *Haversine Formula* akan menghasilkan jarak terpendek antara dua titik, misalnya pada bola yang diambil dari garis bujur (*longitude*) dan garis lintang (*latitude*). Formula ini pertama kali ditemukan oleh Jamez Andrew di tahun 1805, dan digunakan pertama kali oleh Josef de Mendoza y Ríos di tahun 1801. Istilah haversine ini sendiri diciptakan pada tahun 1835 oleh Prof. James Inman. Josef de Mendoza y Ríos menggunakan haversine pertama kali dalam penelitiannya tentang "Masalah Utama Astronomi Nautical", Proc. Royal Soc, Dec 22. 1796. *Haversine* digunakan untuk menemukan jarak antar bintang (Hartanto, 2018).

Haversine Formula adalah persamaan yang digunakan dalam navigasi, yang memberikan jarak lingkaran besar antara dua titik pada permukaan bola (bumi) berdasarkan bujur dan lintang (Sulistio, 2019). *Haversine Formula* merupakan metode yang digunakan untuk menghitung jarak antara dua titik, berdasarkan posisi garis lintang (*latitude)* dan posisi garis bujur (*longitude)* sebagai *variabel input*-an (Wicaksana, 2017). Penggunaan rumus ini mengasumsikan pengabaian efek ellipsoidal, cukup akurat untuk sebagian besar perhitungan, juga pengabaian ketinggian bukit dan kedalaman lembah di permukaan bumi. Berikut (1) adalah rumus *Haversine Formula* (Naufal, 2017)

$$d = 2r.\arcsin\left\{\sqrt{\sin^2\left(\frac{lat_2 - lat_1}{2}\right) + \cos(lat_1).\cos(lat_2).\sin^2\left(\frac{long_2 - long_1}{2}\right)}\right\} \quad (1)$$

Keterangan (1)

r	= jari-jari bumi sebesar 6371(km)
lat	= besaran perubahan latitude
long	= besaran perubahan longitude
d	= jarak (km)

3.5 Metode *System Development Life Cycle (SDLC)*

Metode Pengembangan sistem yang digunakan adalah *system development life cycle* (SDLC). *Software development life cycle* (SDLC) merupakan sebuah metodologi atau alur hidup sistem yang digunakan dalam proses pengembangan sistem yang meliputi tahap pengembangan, pemeliharaan serta penggunaan sistem informasi (Dwanoko, 2016). Dan metode *system development life cycle* (SDLC) ini seringkali dinamakan sebagai proses pemecahan masalah, yang langkah-langkahnya adalah:

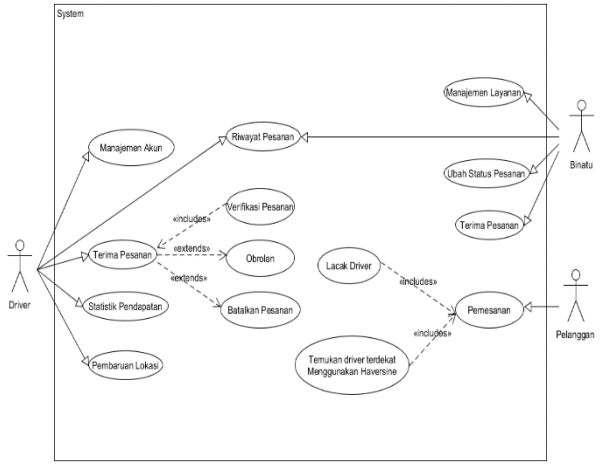
1. Analisis
   Tahapan analisis digunakan oleh sistem untuk membangun keputusan. Apabila sistem saat ini mempunyai masalah atau sudah tidak berfungsi secara baik, dan hasil analisisnya digunakan sebagai dasar untuk memperbaiki sistem.
2. Desain
   Tahapan desain memiliki tujuan untuk mendesain sistem baru yang dapat menyelesaikan masalah-masalah yang dihadapi perusahaan yang diperoleh dari pemilihan alternatif sistem yang terbaik.
3. Implementasi
   Dilakukan untuk kebutuhan *user* untuk mengoperasikan. Pada tahap ini harus dijamin bahwa sistem yang baru dapat berjalan secara optimal.
4. Pengujian
   Melakukan berbagai macam pengujian berkaitan dengan sistem yang dibuat sehingga sesuai dengan hasil yang diinginkan.
5. Pemeliharaan
   Pemeliharaan yang dilakukan analis adalah dengan melakukan perbaikan dan pemeliharaan pada kesalahan atau kegagalan yang timbul dalam penggunaan sistem.

4. PEMBAHASAN

Dalam penelitian ini diperlukan suatu perancangan sistem/aplikasi untuk menunjang kegiatan penelitian dengan baik dan hasil implementasi berupa sebuah aplikasi berbasis android.

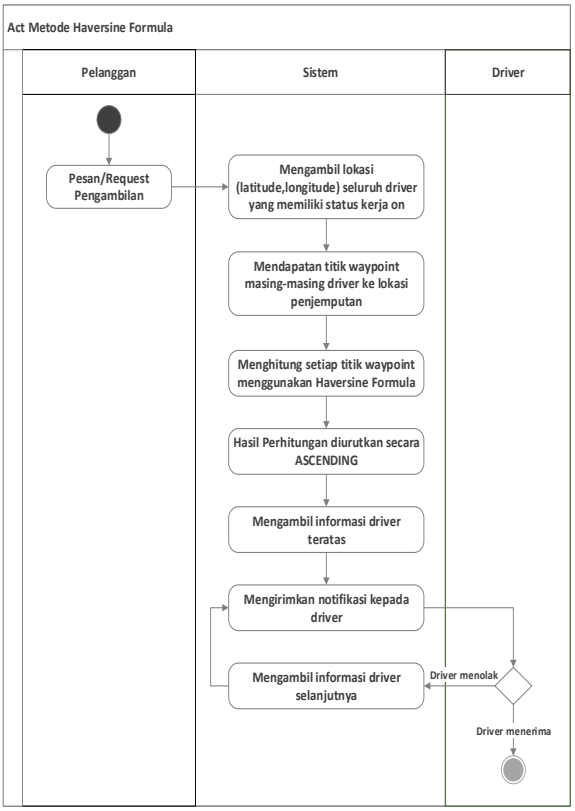
4.1 Analisis Sistem

Berikut pada Gambar 1 merupakan *use case diagram* pada implementasi metode *Haversine Formula* untuk pencarian *driver dry clean* terdekat dengan pelanggan berbasis android.



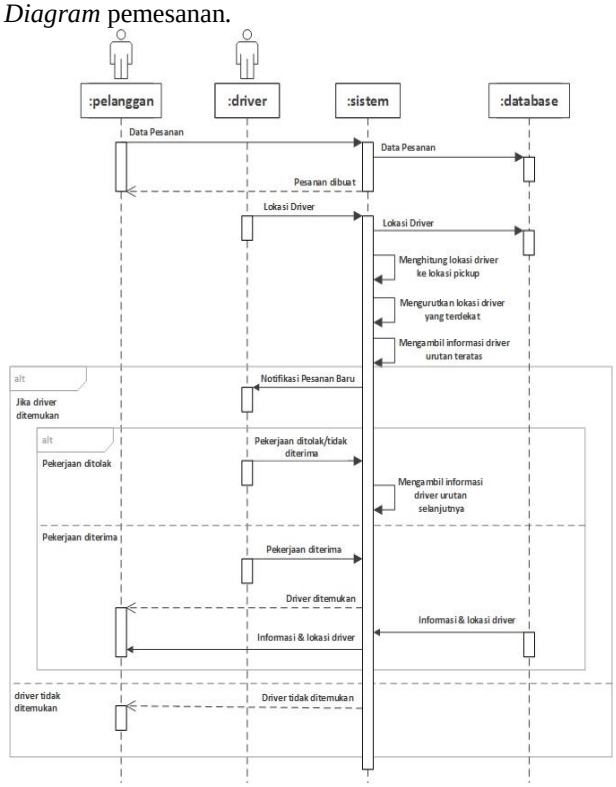
Gambar 1. *Use Case Diagram*

Berikut pada Gambar 2 merupakan gambar *Activity Diagram* penerapan metode *Haversine Formula*.



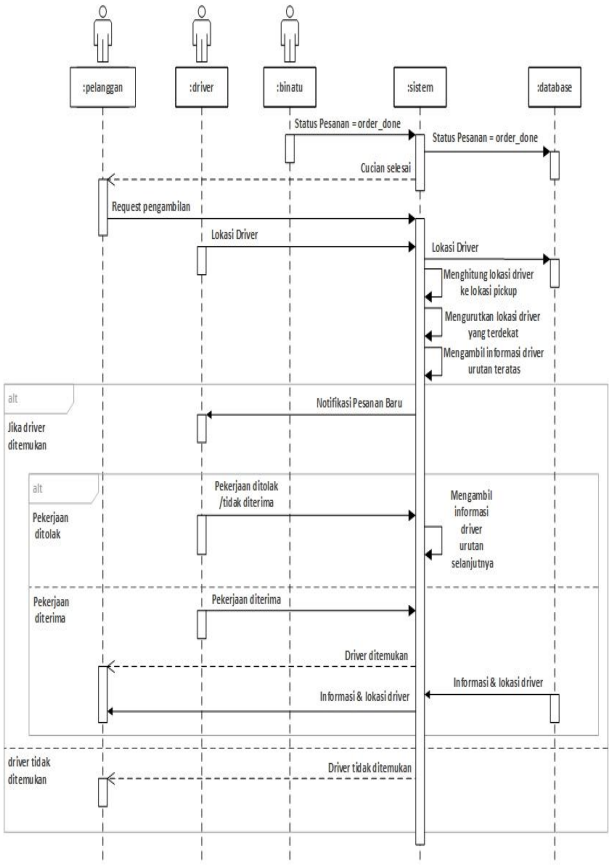
Gambar 2. *Activity Diagram* Metode *Haversine Formula*

Berikut pada Gambar 3 merupakan gambar *Sequence Diagram* pemesanan.



Gambar 3. *Sequence Diagram* Pemesanan

Berikut pada Gambar 4 merupakan gambar *Sequence Diagram* pengambilan.



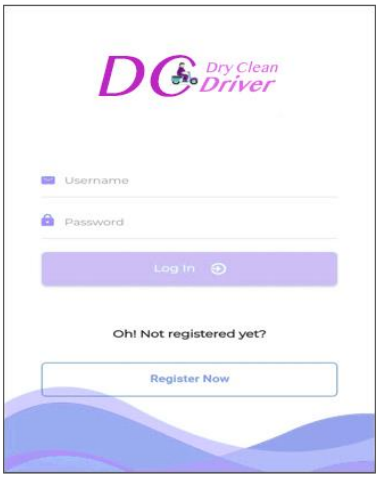
Gambar 4. *Sequence Diagram* Pengambilan

4.2 Implementasi

Hasil implementasi berupa perangkat lunak sebuah aplikasi *driver* dalam menerima *order* baru.

1. Tampilan *Login*.

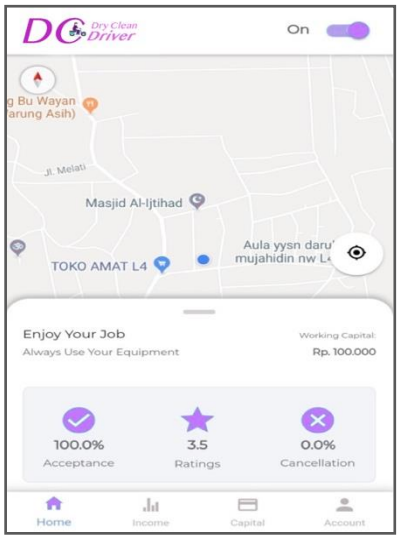
Berikut pada Gambar 5 merupakan tampilan awal dari aplikasi di saat *user* yang belum melakukan *login*, dengan memasukkan *username* dan password maka *user* dapat masuk ke halaman utama dan dapat mulai bekerja sebagai *driver*.



Gambar 5. Tampilan *Login*

2. Tampilan Halaman Utama

Berikut pada Gambar 6 merupakan halaman utama atau *home* setelah *user* melakukan *login*. Terdapat *toogle* pekerjaan di pojok kanan atas yang merupakan kesediaan bagi *driver* apakah siap bekerja atau tidak. Kemudian terdapat sebuah *map* yang memperlihatkan posisi *user* tersebut, persentase penerimaan, persentase pembatalan, *ratings driver*, dan informasi modal kerja yang dimiliki *driver*. Selain itu pada bagian bawah terdapat empat buat *tab* yang mengarahkan *user* menuju halaman yaitu halaman utama, halaman pendapatan *driver*, halaman untuk memilih jumlah modal yang dimiliki *driver*, dan halaman akun *driver*

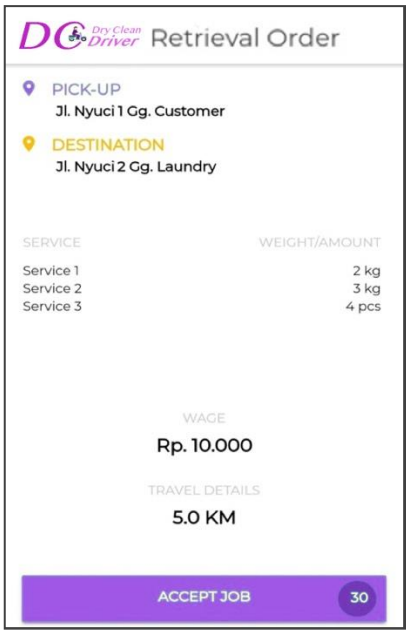


Gambar 6. Halaman Utama

3. Tampilan Pesanan Masuk

Berikut pada Gambar 7 merupakan halaman ketika *driver* mendapatkan *order* baru dimana *driver* memiliki waktu 30 detik untuk menerima *order* tersebut. Jika waktu habis maka *driver* dianggap menolak pesanan tersebut dan akan mempengaruhi persentase penerimaan dan *ratings driver*. Kemudian pada halaman ini juga

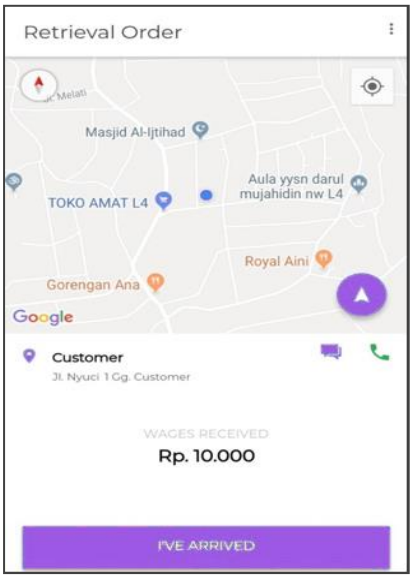
terdapat alamat pelanggan dan alamat binatu, serta terdapat juga informasi pesanan dari pelanggan, upah *driver*, dan jarak perjalanan *driver*.



Gambar 7. Pesanan Masuk

4. Tampilan Penjemputan

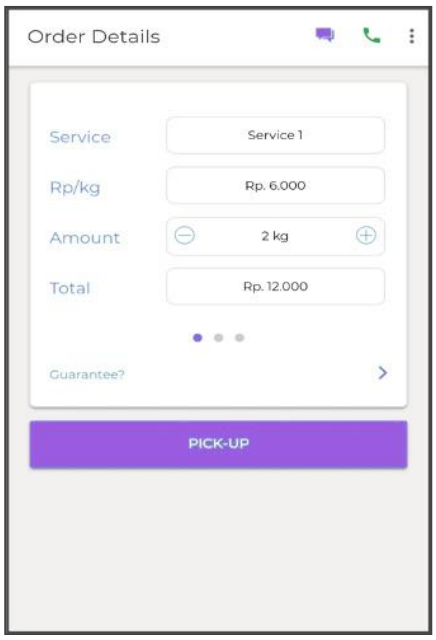
Berikut pada Gambar 8 merupakan yang terdapat informasi penjemputan seperti nama, alamat penjemputan, serta tersedia map, dan fitur lainnya seperti *chat* dan telepon. Selain itu *driver* juga dapat menggunakan *button launch navigate* untuk membuka Google map



Gambar 8. Tampilan Penjemputan

5. Tampilan *Order Details*

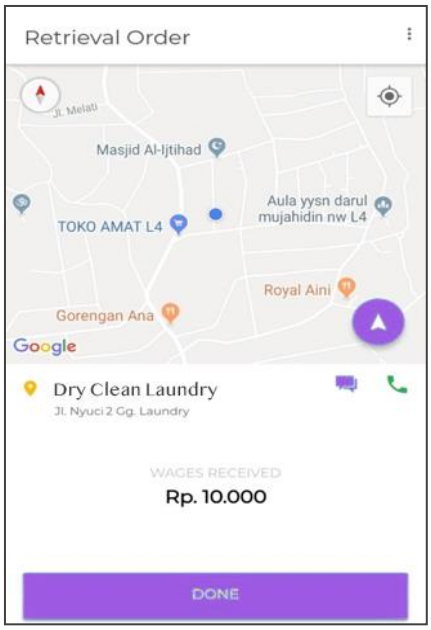
Berikut pada gambar 9 merupakan halaman ketika *driver* sudan tiba di lokasi penjemputan. Dan pada halaman ini terdapat detail pesanan agar *driver* bisa melihat dan memastikan apa pesanan sudah benar atau belum, selain itu *driver* juga dapat mengubah pesanan jika terjadi kesalahan pemesanan pada pelanggan



Gambar 9. *Order Details*

6. Tampilan Tujuan

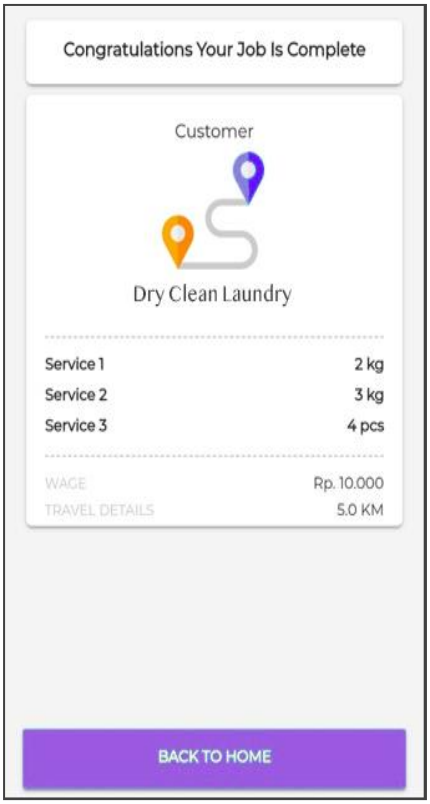
Berikut pada gambar 10 merupakan halaman ketika *driver* sedang menuju ke lokasi tujuan setelah dari lokasi penjemputan. Pada halaman ini terdapat informasi tempat tujuan seperti nama, alamat tujuan, serta tersedia map, dan fitur lainnya seperti *chat* dan telepon. Selain itu *driver* juga dapat menggunakan *button launch navigate* untuk membuka Google map



Gambar 10. Tampilan Tujuan

7. Tampilan Pesanan Selesai

Berikut pada gambar 11 merupakan halaman ketika *driver* sampai ke tempat tujuan dan berhasil menyelesaikan pekerjaan.



Gambar 11. Pesanan Selesai

5. KESIMPULAN

Sistem melakukan perhitungan jarak *driver* terdekat dengan mengambil titik lokasi *driver* yang lagi aktif lalu mendapatkan titik *waypoint* menuju titik pelanggan dari Google *direction* API, dan titik *waypoint* tersebut dihitung menggunakan *haversine formula*.

Sistem melakukan pencarian *driver* terdekat selama 2 menit, setiap *driver* diberi waktu selama 30 detik apabila *driver* pertama tidak menerima dalam waktu 30 detik maka sistem akan mengalihkan ke *driver* selanjutnya, sampai ada *driver* yang menerima dalam waktu 30 detik. Apabila selama 2 menit belum ada *driver* yang menerima maka sistem menghentikan pencarian.

Hasil akhir aplikasi dipublikasikan dalam format .apk yang dapat dijalankan pada *platform* android versi 4.1 sampai dengan versi 7.1.1.

6. SARAN

Diharapkan dapat lebih dikembangkan lagi secara materi maupun tampilan agar lebih menarik dan sempurna, tidak hanya dapat dijalankan di *android* saja namun juga dapat dijalankan di ios maupun windows *phone*. Diharapkan dapat mengembangkan sistem ini dengan metode lainnya dan fitur - fitur tambahan lainnya seperti rekomendasi *driver* ataupun sistem yang dapat menampilkan kinerja *River* yang presensi dengan bintang kinerja.

7. DAFTAR PUSTAKA

Arifin, A. (2018). Implementasi Algoritma Shortest Path Pada Aplikasi Pencarian Rute Distribusi Kurban. In *Seminar Nasional Aplikasi Teknologi Informasi (SNATI)*.

Bakhroin, A. (2020). *Studi komparasi teori KR Muhammad Wardan dan teori James Andrew dalam perhitungan arah kiblat* (Doctoral dissertation, UIN Sunan Ampel Surabaya)

Budiman, E. (2016). Pemanfaatan Teknologi Location Based Service Dalam Pengembangan Aplikasi Profil Kampus Universitas Mulawarman Berbasis Mobile. *ILKOM Jurnal Ilmiah*, *8*(3), 137-144.

Dwanoko, Y. S. (2016). Implementasi Software Development Life Cycle (SDLC) Dalam Penerapan Pembangunan Aplikasi Perangkat Lunak. *Jurnal Teknologi Informasi: Teori, Konsep, dan Implementasi*, *7*(2), 143003.

Fajriansyah, A. (2019). *Sistem Informasi Geografis Pengguna Narkoba Pada Badan Narkotika Nasional Kabupaten Ogan Ilir Berbasis Website* (Doctoral dissertation, POLITEKNIK NEGERI SRIWIJAYA).

Felisitas, E. M. (2017). *Perancangan sistem tracking activity karyawan marketing berbasis android dengan teknologi global positioning system (gps) studi kasus pada PT Astragraphia* (Doctoral dissertation, Universitas Bhayangkara Jakarta Raya).

Hartanto, A. (2018). *Rancang Bangun Aplikasi" Aluminium": Tracer Study untuk Program Studi Teknik Informatika Universitas Muhammadiyah Malang* (Doctoral dissertation, University of Muhammadiyah Malang).

Naufal, M. M. (2017). *Implementasi Haversine formula pada Aplikasi pencarian guru privat terdekat* (Doctoral dissertation, UIN Sunan Gunung Djati Bandung).

Prastyo, Z. R. (2016). *Pemanfaatan Google Maps Api Untuk Mencari Lokasi Spbu Terdekat Di Kota Jepara Dan Kudus Dengan Teknologi Node-Js* (Doctoral dissertation, Universitas Muria Kudus)

Putra, F. S. (2017). *Layanan Berbasis Lokasi Untuk Menemukan Koki Rumahan Yang Terdekat Dan Tersedia* (Doctoral dissertation, Institut Teknologi Sepuluh Nopember).

Setiawan, H. (2016). Implementasi Haversine formula pada lokasi pariwisata berbasis smartphone. *Jurnal Fahma*, *14*(2).

Setiawan, I. (2016). Peran Sistem Informasi Geografis (SIG) dalam Meningkatkan Kemampuan Berpikir Spasial (Spatial Thinking). *Jurnal Geografi Gea*, *15*(1).

Sulistio, J. (2019). *Implementasi Metode Haversine Formula* dalam *Aplikasi untuk Menentukan Lokasi Emergency Service Terdekat di Daerah Istimewa Yogyakarta* (Doctoral dissertation, University of Technology Yogyakarta).

Tafa, I. A. (2018). Analisis Tingkat Akurasi Global Positioning System Smartphone dalam Menentukan Titik Lokasi pada Google Map. *Jurnal Teknik Elektro Universitas Tanjungpura*, *1*(1).

Wicaksana, I. (2017). *Aplikasi Web Berbasis Lokasi untuk Jual Beli Rumah secara Interaktif di Daerah Istimewa Yogyakarta (DIY)* (Skripsi, STMIK Akakom Yogyakarta).