PEMBANGUNAN APLIKASI ANDROID REKOMENDASI TEMPAT RENTAL MOTOR DI KOTA MALANG DENGAN METODE AHP TOPSIS BERBASIS *LOCATION BASED SERVICES*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Jeriko Hosea Julanto
NIM: 145150200111145

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

PEMBANGUNAN APLIKASI ANDROID REKOMENDASI TEMPAT RENTAL MOTOR DI KOTA MALANG DENGAN METODE AHP TOPSIS BERBASIS *LOCATION BASED SERVICES*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Jeriko Hosea Julanto
NIM: 145150200111145

Skripsi ini telah diuji dan dinyatakan lulus pada
18 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing 2
Komang Candra Brata, S.Kom., M.T., M.Sc.	Ratih Kartika Dewi, S.T., M.Kom
NIK: 201607 890711 1 001	NIK: 201503 890520 2 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 18 Juli 2018

Jeriko Hosea Julanto

NIM: 145150200111145

KATA PENGANTAR

Puji syukur penulis panjatkan atas kehadirat Tuhan Yang Maha Esa, karena telah melimpahkan berkat, rahmat, dan anugerah-Nya kepada penulis, sehingga penulis dapat menyelesaikan skripsi yang berjudul "PEMBANGUNAN APLIKASI ANDROID REKOMENDASI TEMPAT RENTAL MOTOR DI KOTA MALANG DENGAN METODE AHP TOPSIS BERBASIS *LOCATION BASED SERVICES*". Berkat bimbingan dan dorongan dari pihak-pihak yang telah membantu penulis dalam menyusun skripsi ini, penulis dapat menyelesaikan skripsi dengan lebih baik. Maka dari itu penulis ingin mengucapkan terima kasih kepada pihak pihak yang telah membantu penulis secara langsung dan tidak langsung. Adapun pihak-pihak yang membantu penulis dalam menyelesaikan skripsi ini antara lain:

1. Bapak Komang Candra Brata, S.Kom., M.T., selaku dosen pembimbing 1, dan Ibu Ratih Kartika Dewi, S.T., M.Kom selaku dosen pembimbing 2 yang telah membimbing, memberi saran serta motivasi kepada penulis selama penyusunan laporan skripsi.
2. Bapak Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya Malang.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Bapak Agus Wahyu Widodo, S.T, M.Cs. selaku Ketua Program Studi Informatika/Ilmu Komputer, Fakultas Ilmu Komputer Universitas Brawijaya Malang.
5. Kedua Orang Tua saya yaitu Bapak Ir. Lukas Martanto dan Ibu Ir. Dorce Lomo, serta seluruh keluarga besar yang selalu memberikan semangat, nasehat, perhatian, kesabaran dalam mendidik penulis, serta memberikan doa demi terselesaikannya skipsi ini.
6. Teman-teman PA Kalasan dan anggota grup BACOTANIJU yang selalu memberikan hiburan dan kebersamaan ketika penulis pulang kampung.
7. Teman-teman PA TS 58 yang selalu menemani dan berbagi canda tawa selama empat tahun di Malang.
8. Teman-teman kontrakan BCT, Zizi, Wildan, Popon, Kevin, Komang yang selalu menemani dan teman lainnya yang sering mampir di kontrakan yang selalu memberikan hiburan tetapi tidak mau cuci piring setelah di pakai.
9. Teman seperjuangan sejak semester satu yang selalu mengunjungi kontrakan, Hapid, Aldo, Rama, Zizi, dan Ardhan.
10. Seluruh pihak yang telah membantu kelancaran penulisan tugas akhir yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa penulisan skripsi ini jauh dari kata sempurna. Maka untuk itu penulis mengharapkan saran, kritik, serta masukan dari semua pihak, demi kesempurnaan dan perbaikan skripsi ini. Akhir kata semoga skripsi ini dapat memberikan manfaat bagi semua pihak.

Malang, 18 Juli 2018

Penulis

jerikojulanto@gmail.com

ABSTRAK

Jeriko Hosea Julanto. 2018. : Pengembangan Aplikasi Android Rekomendasi Tempat Rental Motor di Kota Malang dengan Metode AHP TOPSIS Berbasis *Location Based Services*. Dosen Pembimbing: Komang Candra Brata, S.Kom., M.T., M.Sc. dan Ratih Kartika Dewi, S.T., M.Kom.

Hanya sedikit warga asli Kota Malang yang tahu tempat rental motor di Kota Malang. Padahal seperti yang sudah diketahui, Kota Malang termasuk kota pariwisata yang selalu ramai dikunjungi. Ditambah lagi masing-masing orang memiliki kriteria yang berbeda-beda dalam memilih tempat rental motor. Dengan fasilitas yang seadanya membuat pengguna mengalami kesulitan dalam memilih tempat rental motor di Kota Malang yang sesuai dengan keinginan. Untuk mengatasi masalah tersebut dibutuhkan sistem pendukung keputusan (SPK) untuk mendukung dan mempermudah pengguna dalam memilih lokasi rental motor. Metode SPK tersebut harus memiliki kompleksitas waktu yang rendah. Aplikasi android rekomendasi tempat rental motor di kota malang dengan metode AHP TOPSIS berbasis location based services adalah aplikasi yang memberikan rekomendasi tempat rental motor di kota Malang yang sesuai dengan bobot kepentingan kriteria yang dimasukkan oleh pengguna. Kriteria ini mencakup harga sewa motor 125cc per hari, lokasi tempat rental motor, popularitas dan *rating* tempat rental motor. Hasil dari pengujian penelitian ini mengatakan bahwa nilai usability mencapai nilai 82.82%, dimana nilai ini termasuk dalam kualifikasi baik, kemudian hasil pengujian fungsional bernilai 100%, lalu pengujian kesesuaian juga menyatakan bahwa aplikasi ini memiliki nilai hasil pengujian 90%.

Kata kunci: ahp, topsis, android, *location based services,* rental motor

ABSTRACT

Jeriko Hosea Julanto. 2018. : Development of Application Android Recommendation of Motorcycle Rental in Malang City using AHP TOPSIS Method and Location Based Services. Supervisors: Komang Candra Brata, S.Kom., M.T., M.Sc. and Ratih Kartika Dewi, S.T., M.Kom.

There are very few native townspeople of Malang City, who know the motorcycle rental in Malang City. In fact, as already know, the city of Malang is a tourist city which is always crowded by visitors. Plus each person has different criteria in choosing a motorcycle rental. Scratch facilities make users have difficulty in choosing a motorcycle rental in Malang City. To solve these problems required decision support system (DSS) to support and facilitate the user in choosing a motorcycle rental. The DSS method must have a low time complexity. Application android recommendation of motorcycle rental in malang city by AHP TOPSIS method based on location based services is an application that gives recommendation of motorcycle rental in malang city according to weight of interest criteria entered by user. These criteria include the price of 125cc motorcycle rental per day, the location of the motorcycle rental, the popularity and rating of the motorcycle rental. Tests of this research resulted that the percentage of usability reached the value of 82.82%, where this value is included in good category, then the functional test result is 100%, then the conformity test also stated that this application has 90% test result value.

Keywords: ahp, topsis, android, location based services, motorcycle rental

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Kota Malang adalah salah satu kota yang memiliki kebudayaan yang sangat beragam. Keanekaragaman budaya inilah yang digunakan sebagai asset kekayaan daerah yang memiliki potensi tinggi jika dikembangkan sebaik mungkin. Asset kekayaan dari salah satu kota di jawa timur ini terletak di tempat objek wisata sejarah dan budaya yang selalu ramai oleh pengunjung (Chairi, Mardi Putri, & Fanani, 2018).

Potensi ini bisa berkembang lebih tinggi lagi akibat terdapat banyak perguruan tinggi negeri maupun swasta di Kota Malang. Perguruan tinggi ini menghasilkan mahasiswa yang mampu menjadi promotor-promotor gratis yang mampu mempromosikan Kota Malang serta kekayaan daerahnya di masing-masing daerah asal mereka. Media sosial yang semakin berkembang juga bisa digunakan para mahasiswa untuk memamerkan keindahan tempat wisata yang mereka kunjungi kepada teman-teman mereka di dunia maya (Lestianti, 2016).

Survey yang dilakukan oleh penulis memperolah hasil bahwa banyak dari pengunjung tersebut tidak menggunakan kendaraan pribadi ketika mengunjungi tempat objek wisata tersebut, sehingga diharuskan untuk melakukan rental kendaraan selama berada di Malang. Hasil survey mengatakan bahwa dari 32 responden, ada 68,8 % orang yang dimintai tolong untuk mencarikan tempat rental kendaraan, dan dari 32 responden tersebut hanya 46,9 % saja yang mengetahui tempat rental kendaraan. Permasalahan lain muncul dalam hal pemilihan tempat rental motor. Masing-masing orang memiliki kriteria yang berbeda-beda dalam melakukan pemilihan tempat rental motor. Hasil dari survey yang telah dilakukan kepada 29 responden menyatakan bahwa 62,1% responden memilih harga sebagai kriteria yang paling penting dalam memilih tempat rental, kemudian 17,2% responden memilih jarak tempat rental sebagai kriteria yang paling penting, dan 10,3% responden memilih palayanan tempat rental.

Permasalahan tersebut sebenarnya bisa diselesaikan dengan dengan teknologi internet sebagai wadah untuk mendapatkan informasi yang diinginkan dari suatu tempat. Informasi ini sebenarnya bisa di dapatkan dari pencarian Google atau dari media sosial seperti Facebook, dan Instagram, namun untuk menghitung kriteria jarak harus secara manual untuk menghitung jarak antara posisi pengguna dengan tempat rental motor, sehingga pencarian ini sangatlah menyusahkan. Hal tersebut bisa menimbulkan permasalahan tentang *usability* sehingga masyarakat membutuhkan sebuah media yang bisa digunakan sebagai mencari informasi dari suatu tempat sekaligus memberikan rekomendasi sesuai kriteria yang diminta secara otomatis. Media ini diharapkan bisa memudahkan pengguna untuk memilih tempat rental motor di Kota Malang. Rekomendasi ini bisa didapatkan dari salah satu sistem cerdas menggunakan sistem pendukung keputusan sehingga dapat memberikan informasi dari suatu tempat menjadi keluaran sistem (Firdausy, Agus, & Astuti, 2017).

Penulis juga melakukan survey yang memperoleh hasil bahwa bahwa 65,9% dari 41 responden ternyata lebih banyak menghabiskan waktu pada perangkat smartphone daripada PC Desktop/Laptop dan 92,7% responden memakai smartphone dengan sistem operasi Android.

Berdasarkan penjelasan diatas maka diperlukan suatu aplikasi Android rekomendasi tempat rental motor di Kota Malang yang akan dikembangkan dalam skripsi ini. Dibutuhkan sistem pendukung keputusan untuk mendukung dan mempermudah dalam pemilihan suatu tempat. Banyak metode sistem pendukung keputusan yang sering digunakan, antara lain metode Simple Additive Weighting (SAW), metode Weighted Product (WP), Technique for Order Performance by Similiarity to Ideal Soluction (TOPSIS) dan sebagainya (Nofriansyah, 2014). Namun tidak semua metode sistem pendukung keputusan memiliki performa yang baik jika digunakan dalam merekomendasikan suatu tempat menggunakan aplikasi Android. Dalam pengembangan aplikasi mobile yang memiliki performa baik, hal yang diperlukan dalam suatu metode adalah kompleksitas waktu yang rendah. Metode AHP TOPSIS adalah metode pendukung keputusan yang memiliki performa paling baik dalam kompleksitas waktu (Dewi, Hanggara, & Pinandito, 2018). Maka dari itu metode pendukung keputusan yang akan digunakan dalam penelitian adalah metode AHP TOPSIS.

Uraian-uraian tersebut menjadi dasar dan latar belakang dalam melakukan penelitian pada skripsi yang berjudul "Pembangunan Aplikasi Android Pemilihan Tempat Rental Motor di Kota Malang dengan Metode AHP TOPSIS Berbasis *Location based services*".

1.2 Rumusan masalah

Rumusan masalah pada penelitian ini sebagai berikut:

1. Bagaimana hasil analisis kebutuhan, perancangan, implementasi dan pengujian sistem dari aplikasi Android rekomendasi tempat rental motor di Kota Malang dengan Metode AHP TOPSIS berbasis *location based services*?
2. Bagaimana tingkat kesesuaian keluaran dari aplikasi Android rekomendasi tempat rental motor di Kota Malang dengan Metode AHP TOPSIS berbasis *location based services*?
3. Apakah aplikasi Android rekomendasi tempat rental motor dengan Metode AHP TOPSIS berbasis *Location based services* ini dapat memudahkan pengguna dalam pemilihan tempat rental motor di Kota Malang?
4. Apakah implementasi metode AHP TOPSIS pada aplikasi Android rekomendasi tempat rental motor di Kota Malang berhasil dilakukan?

1.3 Tujuan

Tujuan penelitian ini sebagai berikut:

1. Membuat sebuah aplikasi yang dapat memberikan rekomendasi dan informasi tentang tempat rental motor di Kota Malang pada perangkat bergerak Android menggunakan metode AHP TOPSIS berbasis *location based services*.

2. Mengetahui tingkat kesesuaian keluaran dari aplikasi Android rekomendasi tempat rental motor di Kota Malang dengan Metode AHP TOPSIS berbasis *location based services*.
3. Mengetahui kemudahan pengguna dalam pemilihan tempat rental motor di Kota Malang menggunakan aplikasi Android rekomendasi tempat rental motor dengan Metode AHP TOPSIS berbasis *location based services.*
4. Melakukan implementasi metode AHP TOPSIS pada aplikasi Android rekomendasi tempat rental motor di Kota Malang.

1.4 Manfaat

Manfaat penelitian ini adalah memudahkan orang-orang yag ingin mencari tempat rental motor di malang sesuai dengan kriteria yang diinginkan.

1.5 Batasan masalah

Batasan masalah dari penelitian ini, antara lain:

1. Sistem dikembangkan untuk perangkat *mobile* yang memiliki sistem operasi Android.
2. Daerah yang menjadi objek penelitian adalah Kota Malang.
3. Koneksi internet dianggap selalu terhubung.
4. Perangkat *mobile* memiliki sensor GPS.
5. Kriteria yang digunakan dalam perhitungan aplikasi Android rekomendasi tempat rental motor di Kota Malang adalah harga, jarak, pelayanan yang diwakilkan oleh *rating* Google, dan popularitas.
6. Aplikasi ini tidak memfasilitasi proses pemesanan rental motor.
7. Data yang diambil oleh aplikasi ini bersumber dari database yang disediakan oleh sebuah aplikasi server berbasis web. Namun proses dalam perancangan maupun implemetasi dari aplikasi server tersebut tidak dijelaskan dalam dokumen ini.

1.6 Sistematika pembahasan

Supaya tujuan dari penelitian ini terpenuhi, maka sistematika pembahasan dalam pembentukan penelitian ini sebagai berikut:

Bab 1 Pendahuluan

Bab ini menjelaskan latar belakang permasalahan mengapa dilakukan penelitian tentang rekomendasi tempat rental motor di Kota Malang. Bab ini juga membahas tentang rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika pembahasan.

Bab 2 Landasan Kepustakaan

Bab ini membahas tentang penelitian yang sudah ada, persamaan, dan perbedaan dengan penelitian yang diusulkan. Selain itu, bab ini juga membahas tentang dasar teori yang diperlukan dalam penyusunan penelitian ini. Dasar teori yang dibahas adalah Sistem Pendukung Keputusan (SPK), *AHP TOPSIS,* Android, *Location Based Service,* dan Pengujian Perangkat Lunak.

Bab 3 Metodologi

membahas tentang metode yang digunakan dan langkah-langkah yang diperlukan dalam penyusunan penelitian ini.

Bab 4 Rekayasa Kebutuhan

Bab ini membahas tentang analisis kebutuhan yang akan ada dalam aplikasi dan aktor yang akan berhubungan langsung dengan aplikasi. Pada bab ini juga membahas tentang pengumpulan dan analisis data.

Bab 5 Perancangan dan Implementasi

Membahas tentang perancangan algoritma dan perancangan sistem berdasarkan kebutuhan yang telah dianalisis. Bab ini juga membahas tentang implementasi sistem berdasarkan perancangan yang telah dibuat.

Bab 6 Pengujian

Bab ini menjelaskan proses pengujian sistem dengan membandingankan dengan kebutuhan yang telah dianalisis.

Bab 7 Penutup

Berisi kesimpulan yang didapatkan dari pengujian sistem dan saran yang dapat dimanfaatkan dalam penelitian yang selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini berisikan landasan kepustakaan yang diperlukan sebagai landasan dalam penelitian. Landasan kepustakaan berisi kajian pustaka yang membahas tentang penelitian sebelumnya yang memiliki persamaan dengan penelitian yang diusulkan, dan berisi dasar teori yang membahas tentang teori yang diperlukan dalam menyusun penelitian ini. Dasar teori yang dibahas adalah Sistem Pendukung Keputusan, *AHP, TOPSIS*, *AHP TOPSIS,* Android*, Location Based Service*, dan Pengujian Perangkat Lunak*.*

2.1 Kajian Pustaka

Metode AHP TOPSIS sudah beberapa kali digunakan dalam berbagai penelitian sebagai sistem pendukung keputusan. Kelebihannya adalah memiliki kompleksitas waktu yang rendah sehingga sangat cocok jika diimplementasikan ke dalam aplikasi Android (Dewi, Hanggara, & Pinandito, 2018). Beberapa contoh penelitian yang mengimplementasikan metode ini antara lain, penelitian pertama adalah penelitian oleh Dewi, Hanggara, & Pinandito (2018) yang objek penelitiannya adalah rekomendasi tempat kuliner berbasis mobile. Penelitian ini tidak hanya menggunakan metode AHP TOPSIS namun juga menggunakan AHP dan Fuzzy AHP. Dan hasilnya adalah AHP TOPSIS memiliki kompleksitas waktu yang paling rendah.

Penelitian kedua dilakukan oleh Juliyanti, Irawan, & Mukhlash (2011) menggunakan objek pemilihan guru berprestasi. Penelitian ini menggunakan metode AHP sebagai pemberian bobot kriteria dan metode TOPSIS untuk perankingan untuk menentukan alternatif terpilih. Penelitian ini memiliki hasil bahwa metode ini sangat memadai untuk digunakan dalam proses pemilihan.

Sistem LBS (*Location based services*) banyak digunakan dalam penelitian-penelitian dalam mengetahui lokasi geografis dari pengguna dan lokasi tujuan dari pengguna (Agustina, Risnanto, & Supriadi, 2016). Penelitian yang menggunakan sistem ini antara lain, penelitian ketiga oleh Firdausy, Agus, & Astuti (2017) yang menggunakan Location Based Service dalam mencari lokasi pariwisata dalam di Kota Cimahi.

Penelitian keempat dilakukan oleh Hidayat & Februariyanti (2013). Dalam penelitian ini, peneliti mengembangkan aplikasi Location Based Service dalam pencarian lokasi taxi di Kota Semarang. Penelitan ketiga dan keempat memiliki hasil yang cenderung sama, yaitu tujuan dapat tercapai dengan menerapkan sistem Location Based Service.

Karena referensi-referensi diatas menunjukkan hasil penelitian yang memuaskan, maka dapat diusulkan penelitian pengembangan aplikasi rekomendasi untuk membantu memilih tempat rental motor di malang yang menggunakan metode Weighted Product dalam pengambilan keputusan, dan Location Based Service dalam memberikan informasi berdasarkan lokasi dari pengguna. Ringkasan dari penjelasan diatas, persamaan dan perbedaan dengan penelitian ini dapat dilihat di Tabel 2.1.

Tabel 2.1 Kajian Pustaka

No.	Penulis	Judul	Persamaan	Perbedaan
1	(Dewi, Hanggara, & Pinandito, 2018)	A Comparison Between AHP and Hybrid AHP for Mobile Based Culinary Recommendation System	Menggunakan metode yang sama yaitu AHP TOPSIS	Objek yang diteliti adalah Rekomendasi Tempat kuliner. Metode yang digunakan tidak hanya AHP TOPSIS
2	(Juliyanti, Irawan, & Mukhlash, 2011)	Pemilihan Guru Berprestasi Menggunakan Metode AHP dan TOPSIS	Menggunakan metode yang sama yaitu AHP TOPSIS	Objek yang diteliti adalah Pemilihan Guru Berprestasi.
3	(Firdausy, Agus, & Astuti, 2017)	Aplikasi Android Hybrid Untuk Pemilihan Lokasi Kuliner	Menggunakan Sistem Location Based Service	Objek yang diteliti adalah Pemilihan Lokasi Kuliner
4	(Hidayat & Februariyanti, 2013)	Aplikasi Location Based Service (LBS) Pencarian Lokasi Taxi Pada Android Di Kota Semarang	Menggunakan Sistem Location Based Service	Objek yang diteliti adalah Pencarian Taxi di Kota Semarang

2.2 Sistem Pendukung Keputusan (SPK)

Pengambilan keputusan adalah teknik pengambilan tindakan berdasarkan pengumpulan fakta, dan penentuan secara matang terhadap alternatif yang tersedia. Pengambilan keputusan ini menghadapi permasalahan ketika menghadapi fakta dan alternatif yang terlalu banyak. Dari permasalahan inilah dibutuhkan seperangkat sistem yang mampu mengambil keputusan secara efisien dan efektif yang disebut Sistem Pendukung Keputusan (SPK) (Andayati, 2010).

SPK adalah bagian dari sistem informasi berbasis computer untuk mendukung pengambilan keputusan dengan cara mengolah data dan fakta yang ada untuk dasar dalam menentukan keputusan yang tepat dari alternatif yang ada. Sistem ini dirancang agar interaktif untuk memudahkan integrase antara pengguna dan komponen lain dalam sistem. Dalam mengambil keputusan, sangat penting untuk mengidentifikasi semua faktor yang berpengaruh dan tingkat pengaruhnya (Nurdiyanto & Meilia, 2016).

2.3 Analytic Hierarchy Process (AHP)

Metode AHP adalah metode yang dikembangkan pada tahun 1970 oleh Prof. Thomas Lorie Satty yang berasal dari Wharton Business School. Metode ini digunakan untuk mencari urutan prioritas atau ranking dari berbagai pilihan atau alternative untuk memecahkan permasalahan. Input utama dari metode ini adalah persepsi manusia (Purnomo, Sihwi, & Anggrainingsih, 2013). Latifah dan Siti (2005) menjelaskan bahwa dalam menggunakan metode ini, ada beberapa prinsip dasar yang perlu dipahami, yaitu (Swasono, 2015):

1. *Decomposition*

   Decomposition merupakan pemecahan permasalahan menjadi unsur-unsur terkecil sampai tidak bisa dipecah lagi. Proses ini dilakukan setelah permasalahan berhasil diidentifikasi. Proses ini akan mendapatkan tingkatan-tingkatan permasalahan sehingga proses ini bisa disebut juga sebagai *hierarchy* yang memiliki 2 jenis, yaitu lengkap dan tidak lengkap. *Hierarchy* bisa disebut lengkap jika elemen pada tingkat tertentu memiliki semua elemen pada tingkat yang berikutnya, dan *hierarchy* bisa disebut tidak lengkap jika elemen tidak memiliki seluruh elemen pada tingkat yang berikutnya.
2. *Comparative Judgment*

   Comparative judgement merupakan proses inti dari AHP dimana dalam proses ini suatu elemen akan dibandingkan kepentingan relatifnya dengan elemen lain yang saling berkaitan. Hasil dari proses ini akan disajikan dalam matriks perbandingan berpasangan.
3. *Synthesis of Priority*

   Synthesis of priority adalah proses untuk mendapatkan *local priority* dan *global priority* dimana *lokal priority* didapatkan melalui mencari nilai *eigen vector.* Sedangkan *global priority* didapatkan melalui sintesa *local priority.*
4. Logical Consistency

   Terdapat dua konsistensi yang harus dilakukan jika ingin mendapatkan penilaian yang tepat, konsistensi yang pertama adalah pengelompokkan beberapa objek yang serupa ragamnya dan relevansinya, lalu konsistensi yang kedua adalah hubungan antar objek berdasarkan suatu kriteria.

Junior dan Bogi Farizna (2015) menambahkan bahwa untuk menangani masalah dengan memanfaatkan metode AHP, ada beberapa tahapan yang harus dilalui, yaitu (Swasono, 2015):

1. Mengidentifikasi masalah serta solusinya dan menyusun permasalahan tersebut menjadi sebuah hirearki.
2. Menentukan prioritas elemen

   Prioritas elemen didapatkan dengan membuat matriks perbandingan berpasangan yang berisi kepentingan relatife suatu elemen dengan elemen

lainnya. Skala penilaian perbandingan berpasangan dapat dilihat pada Tabel 2.2 (Purnomo, Sihwi, & Anggrainingsih, 2013).

Tabel 2.2 Skala Penilaian Perbandingan Berpasangan

Intensistas kepentingan	Definisi	Keterangan
1	*Equal Importance*	Suatu elemen sama pentingnya dengan elemen yang lain
3	*Weak importance of one over*	Suatu elemen sedikit lebih penting daripada elemen yang lain
5	*Essential or strong importance*	Suatu elemen lebih penting daripada elemen yang lain
7	*Demonstrated importance*	Suatu elemen jelas lebih mutlak penting daripada elemen yang lain
9	*Extreme importance*	Suatu elemen mutlak lebih penting daripada elemen yang lain
2,4,6,8	*Intermediate values between the two adjacent judgements*	Nilai-nilai di antara dua nilai pertimbangan-pertimbangan yang berdekatan

3. Melakukan normalisasi matriks untuk mendapatkan seluruh prioritas menggunakan Persamaan 2.1.

$$\bar{a}_{jk} = \frac{a_{jk}}{\sum_{l=1}^{m} a_{lk}} \quad (2.1)$$

Dimana :

$\bar{a}_{jk}$ = Nilai hasil matriks perbandingan berpasangan

a_{jk} = Nilai matriks perbandingan berpasangan baris ke-j kolom ke-k

a_{lk} = Nilai matriks perbandingan berpasangan baris ke-l kolom ke-k

4. Menghitung nilai bobot sintesis dengan melakukan penjumlahan antara seluruh hasil matriks pada langkah ketiga pada baris yang sama.
5. Menghitung nilai eigen dengan melakukan pemangkatan dengan seperjumlah total kriteria. Nilai yang dipangkatkan adalah nilai masing-masing cell yang sebelumnya telah dilakukan perkalian dengan matriks perbandingan berpasangan yang mempunyai baris yang sama dengan cell tersebut.
6. Menghitung nilai bobot prioritas dengan melakukan pembagian antara setiap kriteria dengan total nilai eigen.
7. Menghitung nilai kepentingan dan menghitung eigen maksimum (λ maks) dengan melakukan pembagian antara bobot sintesis dengan bobot prioritas di

masing-masing kriteria. Lalu eigen maksimum didapatkan dengan membagi total jumlah kepentingan dengan jumlah total kriteria.

8. Menghitung *Consistency Index (CI)* dengan Persamaan 2.2.

$$CI = \frac{(\lambda\ maks - n)}{n-1} \quad (2.2)$$

Dimana

CI = *Consistency Index*

λ maks = eigen maksimum

n = jumlah elemen

9. Menghitung *Consistency Ratio (CR)* dengan Persamaan 2.3.

$$CR = \frac{CI}{IR} \quad (2.3)$$

Dimana

CR = *Consistency Ratio*

CI = *Consistency Index*

IR = *Index Random Consistency*

10. Melakukan pemeriksaan terhadap konsistensi hierarki dengan ketentuan jika nilai tersebut lebih dari 10% maka harus dilakukan perbaikan dalam penilaian data judgement. Jika *Consistency Ratio (CR)* memiliki nilai kurang dari atau sama dengan 0,1 maka perhitungan yang dilakukan bisa dikatakan benar. Nilai *Index Random Consistency (IR)* bisa dilihat pada Tabel 2.3.

Tabel 2.3 *Index Random Consistency*

Ukuran Matriks	1	2	3	4	5	6	7	8	9	10	11	12
IR	0	0	0,58	0,90	1,12	1,24	1,32	1,41	1,45	1,49	1,51	1,48

2.4 Technique For Order Preference by Similarity to Ideal Solution (TOPSIS)

TOPSIS adalah metode yang mulai dikenal pada tahun 1981 oleh Yoon dan Hwang. Alternatif terpilih berjarak terdekat dengan solusi ideal positif dan berjarak terjauh dari solusi ideal negative (Purnomo, Sihwi, & Anggrainingsih, 2013).

Metode TOPSIS banyak digunakan dalam berbagai penelitian yang menggunakan *Multi Attribute Decision Making (MADM)* karena perhitungannya yang sederhana dan tidak susah untuk dipahami, memiliki keefisienan komputasi yang tinggi, metode ini dianggap praktis untuk mengambil keputusan (Swasono, 2015).

Menurut Satriawan dan Adityaranda (2013) ada beberapa tahapan yang harus dilalui untuk menggunakan metode TOPSIS, yaitu (Swasono, 2015):

1. Menghitung matriks keputusan masing-masing alternatif (m) terhadap kriteria (n) menggunakan Persamaan 2.4. Dimana X_{ij} adalah *rating* kinerja alternatif ke-I terhadap kinerja ke-j. Lalu matriks keputusan tersebut dinormalisasikan menggunakan Persamaan 2.5.

$$X_{ij} = \begin{bmatrix} X_{11} & \cdots & X_{1n} \\ \vdots & \ddots & \vdots \\ X_{m1} & \cdots & X_{mn} \end{bmatrix} \quad (2.4)$$

$$r_{ij} = \frac{X_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}} \quad (2.5)$$

Dimana

r_{ij} = matriks ternormalisasi [i][j]

X_{ij} = matriks keputusan [i][j]

untuk i = 1, 2, 3, …, m

untuk j = 1, 2, 3, …, n

2. Menghitung matriks ternomalisasi yang diberi bobot (V) dengan bobot W = $(w_1, w_2, w_3, \dots, w_n)$ menggunakan Persamaan 2.6.

$$V = \begin{bmatrix} W_1 r_{11} & \cdots & W_n r_{1n} \\ \vdots & \ddots & \vdots \\ W_1 r_{m1} & \cdots & W_n r_{mn} \end{bmatrix} \quad (2.6)$$

Dimana

V = elemen ternormalisasi

r = matriks ternormalisasi

untuk i = 1, 2, 3, …, m

untuk j = 1, 2, 3, …, n

3. Menentukan solusi ideal positif (A^+) menggunakan Persamaan 2.7 dan solusi ideal negative (A^-) menggunakan Persamaan 2.8. Sebelumnya harus menentukan keuntungan (benefit) dan biaya (cost) terlebih dahulu.

$$A^+ = (y_1^+, y_2^+, \dots, y_n^+) \quad (2.7)$$

$$A^- = (y_1^-, y_2^-, \dots, y_n^-) \quad (2.8)$$

Dimana

A^+ = solusi ideal positif

A^- = solusi ideal negatif

y_1^+ adalah max y_{ij} jika $_j$ adalah atribut benefit, dan min y_{ij} jika $_j$ adalah atribut cost

y_1^- adalah min y_{ij} jika $_j$ adalah atribut benefit, dan max y_{ij} jika $_j$ adalah atribut cost

4. Menghitung jarak nilai setiap alternatif dengan matriks solusi ideal positif menggunakan Persamaan 2.9 maupun dengan matriks solusi ideal positif menggunakan Persamaan 2.10.

$$S_i^+ = \sqrt{\sum_{j=1}^{n}(v_{ij} - v_j^+)^2} \quad (2.9)$$

$$S_i^- = \sqrt{\sum_{j=1}^{n}(v_{ij} - v_j^-)^2} \quad (2.10)$$

untuk i = 1, 2, 3, …, m

Dimana

S_i^+ = jarak alternatif Ai dengan sp;iso ideal positif

S_i^- = jarak alternatif Ai dengan sp;iso ideal negatif

v_j^+ = solusi ideal positif [i]

v_j^i = solusi ideal negatif [i]

v_{ij} = matriks normalisasi berbobot [i][j]

5. Menentukan nilai preferensi pada masing-masing alternatif menggunakan Persamaan 2.11.

$$C_i = \frac{S_i^-}{S_i^- + S_i^+} \quad (2.11)$$

Untuk 0 < C_i < 1 dan i = 1, 2, 3, …, m

Dimana

C_i = kedekatan masing-masing alternatif terhadap solusi ideal

S_i^+ = jarak alternatif Ai dengan solusi ideal positif

S_i^- = jarak alternatif Ai dengan solusi ideal negatif

Berdasarkan C_i bisa dilihat alternatif yang terpilih melalui perhitungan TOPSIS, ditandai dengan nilai yang paling besar.

2.5 AHP TOPSIS

Dalam penelitian ini menggunakan 2 metode yang dikombinasikan untuk mendukung keputusan, yang pertama metode AHP dan yang kedua metode TOPSIS. Metode AHP digunakan untuk memberikan bobot pada masing-masing kriteria yang digunakan dalam penelitian ini. Lalu bobot tersebuta akan digunakan sebagai masukan oleh metode TOPSIS untuk menghitung rangking dari alternatif yang disediakan (Abadi, 2016).

2.6 Android

Android adalah sistem operasi berbasis linux untuk perangkat bergerak berlayar sentuh yang berbentuk telepon pintar atau smartphone atau computer tablet (Andikasani, Awaluddin, Suprayogi, & Darmo, 2014).Sama seperti linux untuk PC, Android termasuk siste operasi *open-source* sehingga pengguna bisa mengembangkan dan memodifikasi systemnya sendiri (Hardianto & Handaga, 2015). Pertama kali operasi sistem Android ini dikembangkan oleh perusahaan Android, Inc, namun awal mula terkenalnya Android adalah ketika dia dibeli oleh Google pada tahun 2005. Dan Android-pun diresmikan pada tahun 2007 dan perilisinnanya bebarengan dengan dibentuknya Open Handset Alliance. Dan pada tahun 2008 di bulan Oktober, ponsel Android mulai dijual untuk pertama kali.

Sejak saat itu operasi system yang pertama kali dikembangkan oleh Android Inc ini merajai pasar smartphone dan terus mengembangkan versinya sampai saat ini (Khairani, Soraya, Petrus, & Rachmansyah, 2013). Yang menjadi salah satu keunikan operasi system ini adalah masing-masing versinya bernama jenis makanan dan berdasarkan urutan alphabet, diawali dari Android 1.0 sampai saat ini Android versi 8.0 bernama Oreo.

Google merilis kode untuk Android ini dibawah naungan lisensi Apache, dan alasan mengapa Android termasuk operasi system open-source adalah karena sumber dari kode untuk Android bersifat open dan lisensi perijinannya memungkingkan operasi system ini bisa di modifikasi tanpa melakukan pembayaran dan tanpa batas, dan hasilnya boleh didistribusikan secara bebas oleh developer-developer, maka tidak heran bahwa operasi system ini digemari oleh developer yang menginginkan operasi system yang praktis, dalam artian tidak harus membuat dari awal namun gratis dan bisa dimodifikasi, sehingga yang awalnya Android ini diperuntukkan untuk telepon pintar dan computer tablet, pada saat ini Android juga mulai dikembangkan untuk smart TV, permainan konsol, dan perangkat lainnya.

2.7 *Location Based Services*

Location Based Services adalah layanan informasi yang dapat diakses dari mobile device menggunakan jaringan. Sistem ini dapat dipakai untuk mengetahui lokasi geografis dari pengguna dan lokasi gegografis dari tempat yang ingin dituju (Agustina, Risnanto, & Supriadi, 2016). Contoh sederhananya adalah informasi tentang suatu Hotel terdekat dapat dikirimkan ke piranti bergerak sesuai dengan lokasi geografis piranti tersebut berada (Hidayat & Februariyanti, 2013).

2.7.1 Komponen *Location Based Service*

Dalam *location based service* yang baik harus memenuhi beberapa komponen. Komponen yang diperlukan adalah sebagai berikut (Steiniger, Neun, & Edwardes, 2006):

1. *Mobile Devices* : sebuah alat yang digunakan pengguna untuk meminta informasi yang akan di-*response* dengan informasi yang bisa berbentuk gambar, text atau yang lainnya. Alat ini bisa berbentuk PDA, Mobile Phone, Laptop, atau alat navigasi untuk kendaraan bermotor.
2. *Communicaton Network* : komponen yang mampu mengirim permintaan data dari pengguna menuju penyedia layanan dan mengirimkan kembali data yang diminta kepada pengguna.
3. *Positioning Component* : posisi pengguna yang bisa didapatkan melalui Global Positioning System agar informasi yang diberikan bisa sesuai dengan posisi pengguna.
4. *Service and Applicaton Provider* : penyedia layanan yang akan melayanin permintaan data atau informasi dari pengguna. Komponen ini juga yang akan bertanggung jawab dalam penyedia informasi bagi pengguna menurut posisi pengguna.

5. *Data and Content Provider* : penyedia data atau informasi yang tidak bisa disediakan langsung oleh *Service and Applicaton Provider.* Contohnya adalah data lokasi geografi dari pengguna yang disediakan oleh *mapping agencies.*

2.8 Pengujian Perangkat Lunak

2.8.1 Pengujian Fungsional

Pengujian fungsional adalah bagian dari pengujian *Black Box,* dimana pengujian ini akan menguji dari keluaran sistem apakah sesuai dengan perancangan dan implementasi sistem atau tidak. *Black box testing* juga menguji fungsional dari sistem apakah sudah sesuai dengan perancangan dan implementasi sistem. Pengujian ini tidak menguji kode dari sistem, tetapi hanya menguji keluaran sistem berdasarkan masukan dari pengguna (Pressman, 2001).

2.8.2 Pengujian Kesesuaian

Pengujian kesesuaian adalah pengujian dengan membandingkan 3 tertinggi hasil keluaran dari sistem dengan 3 data pilihan dari pengguna. Tujuan dari pengujian ini adalah untuk mengetahui tingkat kesesuaian hasil sistem dengan hasil pilihan dari user. Pengujian kesesuaian terdiri dari dua pengujian, yaitu pengujian setiap data sampel dan pengujian total. Untuk mendapatkan kesesuaian sampel bisa dihitung menggunakan Persamaan 2.12

$$Kesesuaian\ sampel = \frac{jumlah\ prioritas\ sesuai}{3} x100\% \quad (2.12)$$

Sedangkan untuk mendapatkan kesesuain total bisa dihitung dengan Persamaan 2.13.

$$Kesesuaian\ total = \frac{jumlah\ data\ sesuai}{3} x100\% \quad (2.13)$$

Jumlah data sesuai pada Persamaan 2.13 adalah adalah jumlah sampel data yang memilih persamaan minimal 1 pilihan antara pilihan pengguna dengan keluaran aplikasi (Azis, Cholissidin, & Furqon, 2017).

2.8.3 Pengujian *Usability*

Pengujian ini dilakukan untuk mengetahui tingkat kemudahan pengguna selama memakai aplikasi ini. Pengujian *usability* mencakup 5 komponen yaitu (Nielsen, 2012):

1. *Learnability* : semudah apa pengguna berhasil menguasai aplikasi ketika pertama kali menggunakan aplikasi.
2. *Efficiency* : secepet apa pengguna menyelesaikan tugas ketika pertama kali menguasai aplikasi.
3. *Memorability* : ketiak pengguna kembali menggunakan aplikasi setelah sekian lama tidak menggunakan, semudah apa pengguna dapat menguasainya kembali.
4. *Errors* : seberapa banyak dan parah kesalahan yang pengguna lakukan dan semudah apa pengguna dapat mengatasi kesalahan tersebut.

5. *Satisfaction* : seberapa nyaman pengguna dengan *design* dari aplikasi.

2.8.3.1 Kuisoner USE

Dalam melakukan pengujian *usability*, media kuisoner yang dipakai adalah kuisoner USE yang terdiri dari 3 komponen yaitu yang pertama kegunaan (*usefullness*), yang kedua kepuasan (*satisfaction*) dan yang terakhir adalah kemudahan penggunaan (*ease of use*). Komponen ease of use dibagi lagi menjadi dua faktor yaitu yang pertama adalah kemudahan dalam penggunaan (*ease of use*) dan yang terakhir adalah kemudahan dalam memperlajari aplikasi (*ease of learning*) (Aelani & Falahah, 2012).

2.8.3.2 Skala Likert

Dalam melakukan pengisian untuk kuisoner USE, digunakan skala likert untuk satuan repons responden yang diwakili oleh skala. Respons tersebut mempunyai tingkatan dari sangat positif sampai sangat negatif. Contohnya adalah sebagai berikut (Risnita, 2014):

a. Sangat setujux
b. Setuju
c. Netral
d. Tidak setuju
e. Sangat tidak setuju

Respons diatas akan diberi nilai agar memudahkan analisis kuantitatif seperti yang ditunjukkan dalam Tabel 2.4.

Tabel 2.4 Penilaian Skala Likert

Jawaban	Nilai
Sangat setuju	5
Setuju	4
Netral	3
Tidak setuju	2
Sangat tidak setuju	1

Setelah dilakukan penilaian skala likert, nilai tersebut akan dipakai untuk menghitung indeks persentase. Untuk menghitung indeks persentase, langkah pertama adalah menghitung Total Skor menggunakan Persamaan 2.14. Lalu menghitung nilai *Y* dengan Persamaan 2.15 untuk menghitung Index(%) menggunakan Persamaan 2.16.

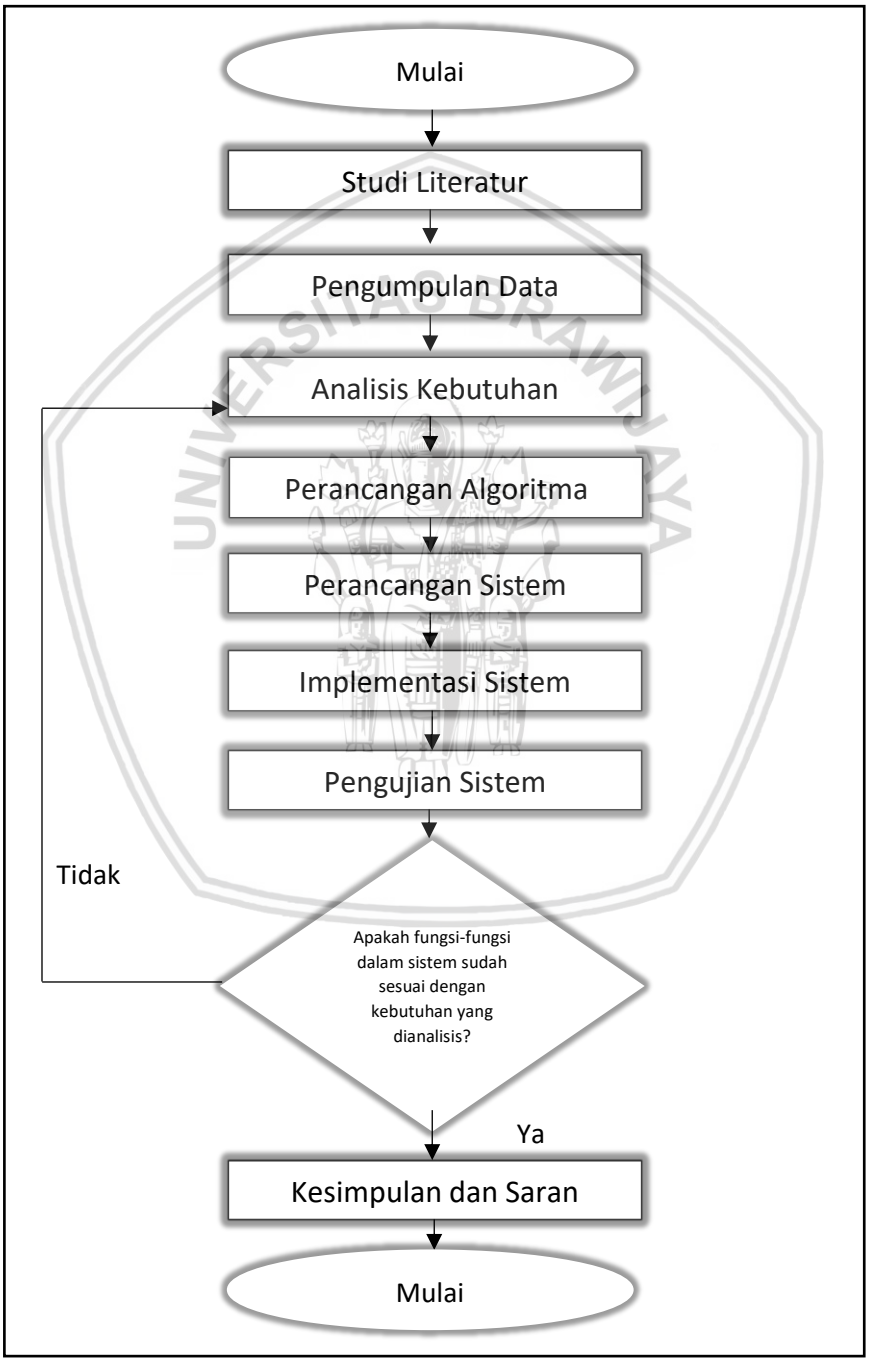
$$TotalSkor=(nilaiSTSx1)+(nilaiTSx2)+(nilaiNx3)+(nilaiSTx4)+(nilaiSSx5) \quad (2.14)$$

$$Y = SkorLikerTertinggi \; x \; JumlahResponden \quad (2.15)$$

$$Index(\%) = (TotalSkor/Y) \; x \; 100\% \quad (2.16)$$

BAB 3 METODOLOGI

Dalam bab ini akan menjelaskan tentang metode penelitian yang akan dilalui selama penelitian ini. Metodologi yang akan digunaakan adalah SDLC *prototyping* dengan dua kali iterasi. Langkah-langkah tersebut meliputi studi literatur, pengumpulan data, analisis kebutuhan, perancangan algoritma, perancangan sistem, implementasi sistem, pengujian sistem, kesimpulan dan saran. Diagram alir dari langkah-langkah penelitian tersebut dapat dilihat pada Gambar 3.1.



Gambar 3.1 Diagram Alir Tahapan Penelitian

3.1 Studi Literatur

Studi literatur adalah daftar referensi yang berhubungan dengan penelitian. Studi literatur ini bermanfaat untuk membantu dalam mempelajari ilmu-ilmu yang terkait. Materi yang digunakan dapat berasal dari berbagai sumber seperti, buku, jurnal, maupun situs web. Materi yang digunakan dalam penelitian ini yaitu:

1. SPK
2. AHP
3. TOPSIS
4. AHP TOPSIS
5. Android
6. Location Based Service
7. Pengujian Perangkat Lunak

3.2 Pengumpulan Data

Tahap pengumpulan data dilakukan untuk mendapatkan data-data yang akan menjadi masukan dalam aplikasi ini. Data yang dibutuhkan adalah data nilai perbandingan kriteria dan data alternatif. Data masukan tersebut yang akan diproses menggunakan metode AHP TOPSIS untuk memperolah hasil sistem rekomendasi.

Data nilai perbandingan kriteria didapatkan dari masukan pengguna yang akan memasukkan nilai perbandingan kriteria satu dengan kriteria yang lain. Data alternatif yang berupa informasi tempat rental didapatkan dari berbagai sumber. Untuk informasi nama, lokasi, *rating*, dan kepopuleran didapatkan melalui mesin pencarian *google places*. Untuk informasi harga didapatkan langsung dari pengelola tempat rental melalui wawancara. Berdasarkan survey yang dilakukan oleh penulis, diperoleh hasil bahwa responden lebih memilih melakukan rental motor berjenis *automatic 125cc*, sehingga informasi harga yang dikumpulkan adalah harga untuk rental motor berjenis *automatic 125cc*. Data alternatif yang akan digunakan untuk data sample berjumlah 10 data yang memiliki kepopuleran paling tinggi dalam wilayah Kota Malang.

3.3 Analisis Kebutuhan

Pada tahap analisis kebutuhan dilakukan identifikasi kebutuhan yang akan ada dalam aplikasi dan aktor yang memiliki peran terhadap aplikasi. Peran tersebut antara lain melakukan masukan data terhadap sistem yang berupa nilai perbandingan kriteria dan menerima output dari sistem yang berupa hasil rekomendasi. Peran ini lah yang akan dirumuskan menjadi kebutuhan fungsional yang akan ada dalam aplikasi ini. Kebutuhan fungsional nantinya akan dijelaskan lebih lanjut dalam *use case diagram* dan *use case scenario.* Aktor yang dapat melakukan peran tersebut juga akan dirumuskan dalam tahap ini. Kebutuhan non fungsional juga dirumuskan untuk memfasilitasi aktor dalam melakukan kebutuhan fungsional dalam aplikasi.

3.4 Perancangan Algoritma

Dalam tahap ini dijelaskan bagaimana cara memproses data masukan nilai perbandingan kriteria dari aktor dan data alternatif menjadi hasil rekomendasi yang akan diterima oleh aktor juga. perhitungan manual menggunakan metode AHP TOPSIS yang digunakan untuk memproses data juga disebutkan disini.

3.5 Perancangan Sistem

Dalam tahap ini dijelaskan cara sistem bisa merealisasikan perancangan algoritma yang digambarkan melalui perancangan UML dan perancangan algoritma dalam bentuk *activity diagram, class diagram, sequence diagram* dan penulisan sampel perancangan kode utama pada masing-masing *class* berbentuk *pseudocode*. Untuk menjelaskan perancangan basis data untuk menyimpan data alternatif dan menyimpan sementara hasil rekomendasi dijelaskan dalam perancangan data yang berisi struktur data yang berbentuk skema json. Dan yang terakhir untuk menggambarkan tampilan aplikasi untuk interaksi *user* untuk memenuhi kebutuhan akan dijelaskan dalam perancangan antarmuka yang berisi tata letak komponen yang akan digunakan dalam sistem berdasarkan kebutuhan sistem dan screenflow. Hasil dari tahap perancangan ini akan digunakan dalam tahap implementasi agar bisa sesuai dengan kebutuhan yang sudah dianalisa.

3.6 Implementasi Sistem

Tahap implementasi sistem dilakukan berdasarkan perancangan sistem yang sudah dilakukan sebelumnya. Tahap implementasi sistem sendiri terdiri dari beberapa proses, yaitu:

1. Implementasi sisi *client* yang akan dirasakan langsung oleh pengguna pada sistem operasi Android. Proses ini mengimplementasikan perancangan antarmuka dan algoritma. Proses ini juga akan mengimplementasikan metode AHP TOPSIS. Implementasi ini dilakukan menggunakan bahasa pemograman *java*.
2. Pembuatan *database* berdasarkan perancangan data berbentuk NoSQL untuk menyimpan data berupa informasi tempat rental motor.

Hasil dari tahap implementasi sistem ini akan diuji dalam tahap pengujian sistem.

3.7 Pengujian Sistem

Tahap pengujian sistem ini berisi pengujian apakah aplikasi berhasil diimplementasi dengan baik sesuai dengan perancangan sistem. Tahap pengujian pertama adalah pengujian fungsioal berupa pengujian *black box* dengan membandingkan hasil setelah fungsional dijalankan melalui aplikasi dengan fungsional yang didefinisikan pada Analisa kebutuhan. Pengujian selanjutnya adalah pengujian algoritma untuk menguji kesesuaian hasil perhitungan perhitungan manual dengan hasil perhitungan aplikasi. Pengujian selanjutnya

adalah pengujian kesesuaian dengan meminta 20 pengguna untuk memilih tempat rental motor yang sesuai dengan bobot kriteria yang mereka inginkan, lalu dibandingkan dengan pilihan tempat rental motor hasil rekomendasi. Pengujian selanjutnya adalah pengujian *usability* dengan memberikan kuisoner kepada 20 pengguna, lalu dihitung skala kepuasannya. Pada tahap ini dapat dilakukan evaluasi untuk penambahan atau perubahan fungsionalitas atau tidak. Jika tidak ada penambahan atau perubahan fungsionalitas, maka bisa langsung menuju tahap pengambil keputusan.

3.8 Pengambilan Kesimpulan

Tahap ini dilakukan setelah tahap perancangan, implementasi, dan pengujian sistem selesai dilakukan. Kesimpulan diambil berdasarkan hasil dari pengujian sistem. Saran juga dituliskan untuk meberikan pertimbangan dalam penelitian selanjutnya yang berhubungan dengan penelitian ini berdasarkan kesalahan yang terjadi selama penelitian berlangsung.

BAB 4 REKAYASA KEBUTUHAN DAN PERANCANGAN

Pada bab ini akan dibahas tentang rekayasa kebutuhan dan perancangan aplikasi yang akan dibangun. Tahap rekayasa kebutuhan adalah tahap yang menggambarkan gambaran umum dan kebutuhan dalam sistem. Selanjutnya adalah tahap perancangan. Perancangan terdiri dari dua tahap. Tahap pertama adalah perancangan algoritma yang melibatkan data bobot kriteria dan data alternatif menggunakan metode AHP TOPSIS. Tahap kedua adalah perancangan sistem yang akan menjelaskan secara lebih detail tentang perancangan aplikasi berdasarkan kebutuhan yang telah diperoleh dari rekayasa kebutuhan.

4.1 Rekayasa Kebutuhan

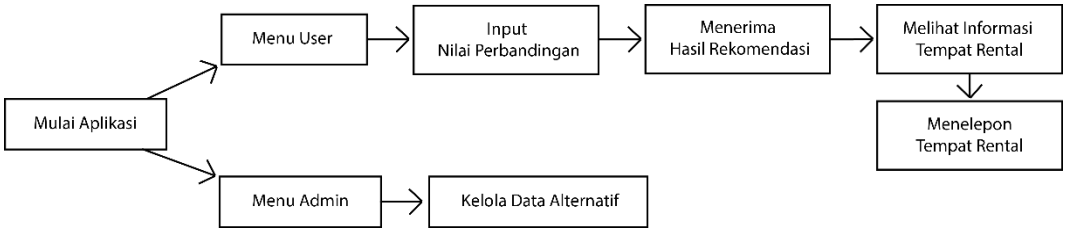
Proses rekayasa kebutuhan ini berisikan tahapan dalam memperolah kebutuhan dalam penelitian yang sedang dilakukan. Kebutuhan-kebutuhan ini nantinya akan digunakan dalam sistem yang lebih dulu akan dirancang dalam proses perancangan sistem. Pada bab ini akan ada beberapa tahapan yaitu gambaran umum dari sistem, identifikasi aktor dari sistem, penentuan kebutuhan fungsional dan non fungsional dari sistem, dan pemodelan kebutuhan yang berisi *use case* dan akan dijelaskan lebih lanjut dalam *use case scenario*.

4.1.1 Gambaran Umum Sistem

Dalam memilih tempat rental, masing-masing orang memiliki ranking untuk kriterianya masing-masing. Ranking inilah yang digunakan untuk prioritas kriteria yang diinginkan dalam memilih tempat rental. Karena hal itu lah dibutuhkan aplikasi sistem rekomendasi tempat rental motor berdasarkan prioritas kriteria yang diinginkan oleh pengguna.

Fitur utama dari aplikasi ini adalah pengguna dapat memberikan nilai perbandingan antar kriteria yang ditampilkan oleh sistem. Kriteria-kriteria yang dapat diberi nilai perbandingan adalah jarak, harga, *rating* dan kepopuleran. Kriteria jarak diperoleh dari jarak antara lokasi *user* deangan lokasi masing-masing tempat rental motor menggunakan *Google Map Directions API*. Berdasarkan nilai perbandingan tersebut akan diperoleh bobot dari masing-masing kriteria menggunakan metode AHP. Dari bobot kriteria tersebut dihasilkan nilai preferensi pada masing-masing data alternatif yang didapatkan dari database yang disediakan web service yang akan diurutkan dari yang paling besar menggunaka metode TOPSIS. Dari urutan tersebut ditampilkan 3 informasi tempat rental motor yang memiliki nilai preferensi paling besar. Selain fitur utama tersebut, ada fitur lain yang ada di aplikasi ini, yaitu *user* dapat melihat informasi dan menelepon tempat rental motor yang dipilih.

Fitur diatas adalah fitur yang bisa dirasakan oleh *user*, sedangkan *admin* sendiri dapat melakukan pengelolaan data alternatif tempat rental. Pengelolaan data ini meliputi tambah, hapus, dan mengubah data. Secara umum gambaran sistem dapat dilihat pada Gambar 4.1.



Gambar 4.1 Gambaran Umum

Karena data yang digunakan diambil dari web service, maka dibutuhkan koneksi internet ketika memakai aplikasi ini.

4.1.2 Identifikasi Aktor

Tahap ini menjelaskan tentang aktor apa saja yang dapat berinteraksi dengan sistem. Dan untuk sistem yang akan dibuat nanti hanya ada dua aktor yaitu aktor *User* dan aktor *Admin*, deskripsi dari kedua aktor ini akan dijelaskan dalam Tabel 4.1.

Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
User	Aktor *User* adalah aktor yang dapat memberi masukan aplikasi berupa nilai perbandingan kriteria untuk mendapatkan hasil rekomendasi dari sistem
Admin	Aktor *Admin* adalah aktor yang dapat melakukan pengelolaan data yang meliputi menambah, mengubah, dan menghapus data alternatif tempat rental.

4.1.3 Aturan Penomoran

Aturan penomoran untuk representasi kebutuhan yang akan digunakan dalam penelitian ini dapat dilihat dalam Tabel 4.2.

Tabel 4.2 *Index Random Consistency*

Kode Penomoran	Keterangan
REKMON	Nama Aplikasi
F	Kode representasi kebutuhan Fungsional
NF	Kode representasi kebutuhan Non- Fungsional
NOMOR	Merupakan angka penomoran kebutuhan

Contoh :

REKMON_F_01 : Representasi kebutuhan fungsional Rekomendasi Tempat Rental Motor dengan nomor urut satu.

4.1.4 Kebutuhan Fungsional

Berikut ini akan dijelaskan kebutuhan fungsional sistem pada aktor *user*. Kebutuhan fungsional dapat dilihat pada Tabel 4.3.

Tabel 4.3 Kebutuhan Fungsional *User*

No.	KODE	NAMA	DESKRIPSI	Aktor
1.	REKMON_F_01	Mendapatkan hasil rekomendasi tempat rental motor	*User* dapat mendapatkan hasil rekomendasi tempat rental motor	*User*
2.	REKMON_F_02	Melihat informasi tempat rental motor	*User* dapat melihat informasi tempat rental motor yang direkomendasikan oleh sistem. Informasi yang ditampilkan meliputi nama, alamat, *rating*, nilai kepopuleran, dan harga	*User*, *Admin*
3.	REKMON_F_03	Menelepon tempat rental motor	*User* dapat menghubungi tempat rental motor dengan cara menelepon tempat rental motor	*User*
4	REKMON_F_04	Menambah data alternatif	*Admin* dapat menambah data alternatif tempat rental motor	*Admin*
5	REKMON_F_05	Mengubah data alternatif	*Admin* dapat mengubah isi data alternatif tempat rental motor	*Admin*
6	REKMON_F_06	Menghapus data alternatif	*Admin* dapat menghapus data alternatif tempat rental motor	*Admin*

4.1.5 Kebutuhan Non Fungsional

Berikut ini akan dijelaskan kebutuhan non fungsional sistem aplikasi rekomendasi tempat rental motor di kota malang. Kebutuhan non fungsional dapat dilihat pada Tabel 4.4.

Tabel 4.4 Kebutuhan Non Fungsional

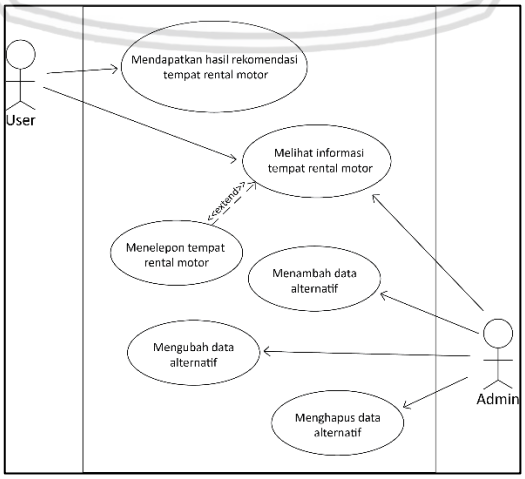
No.	KODE	NAMA	DESKRIPSI
1	REKMON_NF_01	Kesesuaian	Mengetahui tingkat kesesuaian hasil rekomendasi dengan keputusan dari pengguna
2.	REKMON_NF_02	*Usability*	Sistem mampu menyediakan kemudahan kepada pengguna aplikasi dalam memperolah hasil rekomendasi dan dalam menjalankan fitur dalam sistem.

4.1.6 Pemodelan Kebutuhan

Tahap selanjutnya dalam rekayasa kebutuhan adalah pemodelan kebutuhan. Pemodelan kebutuhan dibuat untuk mempermudah memahami kebutuhan-kebutuhan yang ada di sistem. Pemodelan kebutuhan akan digambarkan dalam diagram yaitu *use case diagram*. *Use case diagram* akan menjelaskan apa saja yang dapat dilakukan oleh aktor terhadap sistem. Dan selanjutnya akan dijelaskan lebih rinci dalam *use case scenario*.

4.1.6.1 *Use case* Diagram

Use case diagram dibuat untuk mempermudah memahami apa saja yang dapat dilakukan oleh aktor terhadap sistem dan menjelaskan kebutuhan fungsional apa saja yang ada dalam sistem beserta aktor yang dapat merasakan kebutuhan tersebut. *Use case diagram* dapat dilihat dalam Gambar 4.2.



Gambar 4.2 *Use case* Diagram

4.1.6.2 *Use case Scenario*

Dalam *use case scenario* akan dijelaskan secara lebih rinci masing-masing *use case* yang sudah ditampilkan pada *use case diagram* pada Gambar 4.2.

1. Mendapatkan hasil rekomendasi tempat rental motor (REKMON_F_01)

   Use case scenario dari *use case* mendapatkan hasil rekomendasi tempat rental motor dapat dilihat di Tabel 4.5.

Tabel 4.5 *Use case Scenario* Mendapatkan Hasil Rekomendasi Tempat Rental Motor

Mendapatkan hasil rekomendasi tempat rental motor	
Objective	*User* dapat mendapatkan hasil rekomendasi tempat rental motor
Aktor	*User*
Pre-condition	*User* telah membuka aplikasi Rekomendasi Tempat Rental Motor *User* berada pada Halaman Utama *User*
Main Flow	1. *User* memilih menu "Rekomendasi" pada Halaman Utama *User* 2. Sistem menampilkan form *input* bobot kepentingan masing-masing kriteria 3. *User* mengisi bobot kepentingan kriteria dengan menyeret *thumb* pada *seekbar* 4. *User* menekan tombol "Proses" 5. Akan muncul *form* baru untuk *input* nilai kriteria sebagai filter berdasarkan kriteria yang diberi bobot kepentingan pada poin 3. 6. *User* menekan tombol "Cari Tempat Rental"
Alternative Flow	3. *User* tidak mengisi bobot kepentingan kriteria 4. *User* menekan tombol "Proses"
Post-condition	Sistem menampilkan hasil rekomendasi tempat rental motor di halaman List Hasil Rekomendasi

2. Melihat informasi tempat rental motor (REKMON_F_02)

   Use case scenario dari *use case* melihat informasi tempat rental motor dapat dilihat di Tabel 4.6.

Tabel 4.6 *Use case Scenario* Melihat Informasi Tempat Rental Motor

Melihat informasi tempat rental motor	
Objective	Aktor dapat melihat informasi tempat rental motor
Aktor	*User* dan *Admin*
Pre-condition	Aktor telah membuka aplikasi Rekomendasi Tempat Rental Motor *User* berada pada Halaman Utama *User* *Admin* berada pada Halaman Utama *Admin*

	User telah melakukan fungsionalitas dengan kode REKMON_F_01
Main Flow	Aktor *User* 1. *User* memilih menu "List Tempat Rental" pada Halaman Utama *User* 2. Sistem menampilkan halaman List Tempat Rental 3. Aktor menekan salah satu Data Tempat Rental dari list Aktor *Admin* 1. *User* memilih menu "List Data Alternatif" pada Halaman Utama *Admin* 2. Sistem menampilkan halaman List Data Alternatif 3. Aktor menekan salah satu Data Alternatif dari list
Alternative Flow	Aktor *User* 1. Aktor *User* telah melakukan fungsionalitas dengan kode REKMON_F_01 2. Sistem menampilkan halaman list hasil rekomendasi 3. Aktor menekan salah satu Data Tempat Rental dari list
Post-condition	Sistem menampilkan tampilan detail tempat rental

3. Menelepon tempat rental motor (REKMON_F_03)

   Use case scenario dari *use case* menelepon tempat rental motor dapat dilihat di Tabel 4.7.

Tabel 4.7 *Use case Scenario* Menelepon Tempat Rental Motor

Menelepon tempat rental motor	
Objective	*User* dapat menelepon tempat rental motor
Aktor	*User*
Pre-condition	*User* sudah dalam halaman detail suatu tempat rental
Main Flow	1. *User* memilih tombol lambang bergambar Telepon
Alternative Flow	
Post-condition	Sistem menampilkan aplikasi telepon pada device dengan nomor telepon tempat rental motor

4. Menambah data alternatif (REKMON_F_04)

   Use case scenario dari *use case* menambah data alternatif dapat dilihat di Tabel 4.8.

Tabel 4.8 *Use case Scenario* Menambah Data Alternatif

Menambah data alternatif	
Objective	*Admin* dapat menambah data alternatif
Aktor	*Admin*

Pre-condition	*Admin* telah membuka aplikasi Rekomendasi Tempat Rental Motor *Admin* berada pada Halaman Utama *Admin*
Main Flow	1. *Admin* memilih menu "Add Data Alternatif" pada Halaman Utama *Admin* 2. Sistem menampilkan form Add Data Alternatif 3. *Admin* mengisi form secara lengkap 4. *Admin* menekan tombol "Add Data" 5. Sistem memberikan pemberitahuan bahwa penambahan berhasil
Alternative Flow	3. *Admin* tidak mengisi form secara lengkap 4. *Admin* menekan tombol "Add Data" 5. Sistem memberikan pemberitahuan bahwa data tidak lengkap
Post-condition	Sistem menampilkan halaman List Data Alternatif

5. Mengubah data alternatif (REKMON_F_05)

   Use case scenario dari *use case* mengubah data alternatif dapat dilihat di Tabel 4.9.

Tabel 4.9 *Use case Scenario* Mengubah Data Alternatif

Mengubah data alternatif	
Objective	*Admin* dapat mengubah data alternatif
Aktor	*Admin*
Pre-condition	Aktor telah ada dalam halaman detail suatu tempat rental
Main Flow	1. *Admin* memilih tombol bergambar pensil pada toolbar 2. Sistem menampilkan form *Edit* Data Alternatif dengan form yang sudah terisi dengan data alternatif 3. *Admin* mengisi form secara lengkap 4. *Admin* menekan tombol "Save Data Alternatif" 5. Sistem memberikan pemberitahuan bahwa pengubahan berhasil
Alternative Flow	
Post-condition	Sistem menampilkan halaman Detail Tempat Rental yang baru diubah

6. Menghapus data alternatif (REKMON_F_06)

   Use case scenario dari *use case* menghapus data alternatif dapat dilihat di Tabel 4.10.

Tabel 4.10 *Use case Scenario* Menghapus Data Alternatif

Menghapus data alternatif	
Objective	*Admin* dapat menghapus data alternatif
Aktor	*Admin*
Pre-condition	Aktor telah ada dalam halaman detail suatu tempat rental
Main Flow	1. *Admin* memilih tombol bergambar tempat sampah pada toolbar 2. Sistem memberikan pemberitahuan bahwa penghapusan berhasil
Alternative Flow	
Post-condition	Sistem menampilkan halaman List Data Alternatif

4.2 Perancangan Algoritma

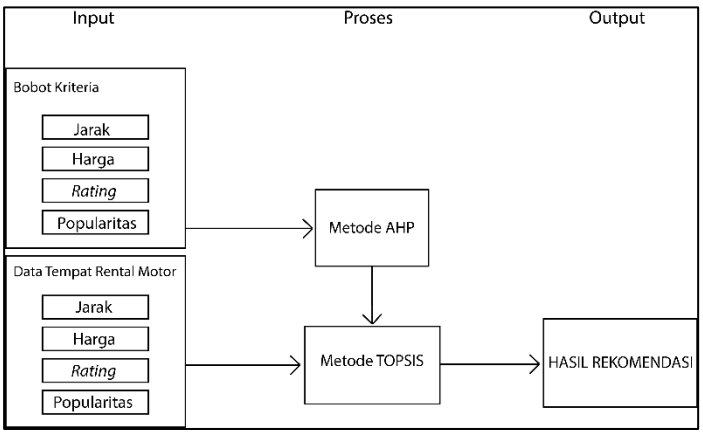
Dalam bagian ini akan dijelaskan perancangan algoritma dari aplikasi ini mulai dari *input* sistem, proses, sampai *output*nya.

Input dari sistem ini adalah bobot kepentingan kriteria dan data alternatif. Bobot kepentingan kriteria yang dipakai dalam perancangan algoritma ini adalah kriteria harga, jarak, popularitas, dan *rating* yang diperoleh dari masukan pengguna. Sedangkan data alternatif diperolah dari data tempat rental yang berada di kota malang berdasarkan kriteria yang dipakai. Untuk perancangan algoritma ini digunakan 10 data alternatif yang memiliki popularitas paling tinggi. Bobot kriteria tersebut akan menjadi masukan metode AHP. Output dari metode AHP dan data alternatif akan menjadi masukan dari metode TOPSIS. *Output* dari metode TOPSIS yang berupa urutan data alternatif yang akan menjadi hasil rekomendasi dari sistem. *Input* sistem bobot kepentingan kriteria yang digunakan untuk contoh kasus perancangan algoritma ini dapat dilihat pada Tabel 4.11.

Tabel 4.11 Bobot Kepentingan Kriteria

Nama Kriteria	Harga	*Rating*	Popularitas	Jarak
Bobot Kepentingan Kriteria	9	1	1	3

Gambaran perancangan algoritma dapat dilihat pada Gambar 4.3



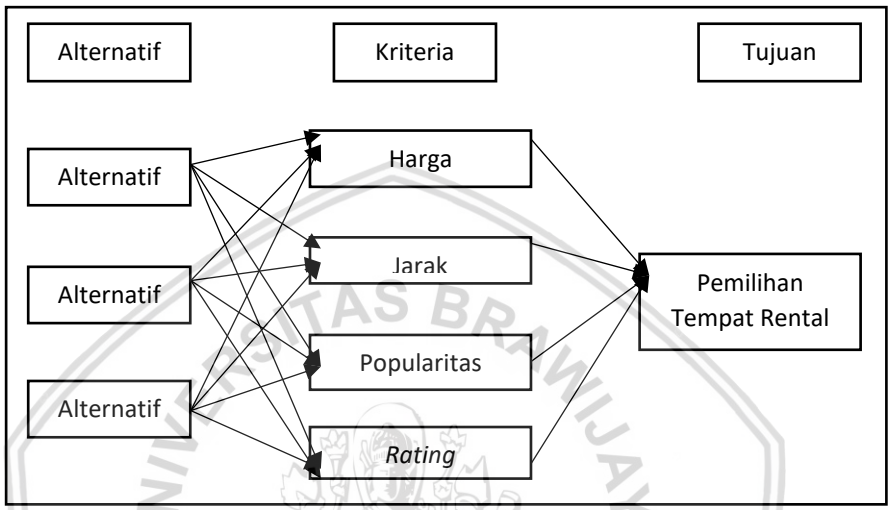
Gambar 4.3 Gambaran Perancangan Algoritma

4.2.1 Proses Metode AHP

Langkah-langkah proses pengerjaan metode AHP menggunakan data kriteria sebagai berikut:

a. Menyusun Hierarki

Penyusunan hierarki bertujuan untuk mengetahui masalah dan pola penyelesaian sesuai kriteria. Gambaran hierarki ditunjukkan pada Gambar 4.4.



Gambar 4.4 Struktur Hierarki

Kriteria disini akan diberi kode untuk mempermudah pembacaan dan penulisan nama kriteria, contohnya adalah kriteria harga yang diberi kode menjadi K1. Selengkapnya dapat dilihat pada Tabel 4.12.

Tabel 4.12 Kriteria Pemilihan

Kriteria	Keterangan
K1	Harga
K2	*Rating*
K3	Popularitas
K4	Jarak

b. Menentukan Prioritas Kriteria

Priroitas kriteria disusun berdasarkan masukan pengguna pada Tabel 4.11 dalam matriks perbandingan kriteria berpasangan yang ditunjukkan pada Tabel 4.13. Matriks perbandingan antar kriteria dihitung dengan cara melakukan pembagian bobot kepentingan antar kriteria.

Tabel 4.13 Matriks Perbandingan Kriteria Berpasangan

Kriteria	K1	K2	K3	K4
K1	1	9	9	3
K2	0.111111	1	1	0.333333
K3	0.111111	1	1	0.333333
K4	0.333333	3	3	1

Tabel 4.13 merupakan tabel matriks perbandingan kriteria berpasangan, sebagai contoh angka 9 pada baris K1 kolom K2 berarti nilai perbandingan K1 dengan K2 adalah 9. Nilai 9 ini adalah hasil pembagian bobot kriteria K1 dengan K2, yaitu 9 dan 1, dan sebaliknya untuk nilai perbandingan K2 dengan K1 adalah 1/9, yaitu 0.111111.

c. Menentukan Normalisasi Nilai Matriks Perbandingan

Nilai matriks perbandingan kriteria dilakukan normalisasi menggunakan Persamaan 2.1 sehingga mendapatkan hasil pada Tabel 4.14.

Tabel 4.14 Matriks Normalisasi Perbandingan Kriteria Berpasangan

Kriteria	K1	K2	K3	K4
K1	0.642857372	0.642857143	0.642857143	0.64285723
K2	0.071428526	0.071428571	0.071428571	0.07142851
K3	0.071428526	0.071428571	0.071428571	0.07142851
K4	0.214285577	0.214285714	0.214285714	0.21428574

Tabel 4.14 adalah hasil dari normalisasi matriks perbandingan pada Tabel 4.13. Sebagai contoh angka 0.642857143 pada baris K1 kolom K2 merupakan hasil dari pembagian antara angka pada baris K1 kolom K2 pada Tabel 4.13 dibagi dengan jumlah seluruh nilai pada kolom K2.

d. Menghitung Bobot Prioritas

Bobot prioritas dihitung dengan membagi sel matrisk perbandingan berpasangan dengan jumlah kolom yang sama lalu dijumlahkan setiap baris. Lalu hasilnya dibagi dengan jumlah kriteria. Hasil lengkapnya dapat dilihat pada Tabel 4.15.

Tabel 4.15 Bobot Prioritas

Kriteria	K1	K2	K3	K4	Jumlah	Prioritas
K1	0.642857372	0.642857143	0.642857143	0.64285723	2.571428893	0.642857223
K2	0.071428526	0.071428571	0.071428571	0.07142851	0.285714179	0.071428545
K3	0.071428526	0.071428571	0.071428571	0.07142851	0.285714179	0.071428545
K4	0.214285577	0.214285714	0.214285714	0.21428574	0.85714275	0.214285687

Tabel 4.15 adalah hasil perhitung bobot prioritas. Dari Tabel 4.15 dapat diketahui bahwa kriteria K1 memiliki prioritas paling tinggi dengan bobot prioritas 0.642857223, disusul dengan kriteria K4 dengan bobot prioritas 0.214285687, lalu yang terakhir kriteria K2 dan K3 dengan bobot prioritas yang sama yaitu 0.071428545.

4.2.2 Proses Metode TOPSIS

Untuk proses metode TOPSIS, masukan data yang dipakai adalah bobot prioritas dari metode AHP, dan matriks keputusan. Matriks keputusan didapati dari pengumpulan data yang sudah dilakukan dan hasil dapat dilihat pada Tabel 4.16.

Tabel 4.16 Matriks keputusan

Kode Alternatif	Nama Alternatif	K1	K2	K3	K4
a1	Arfand Motorent	80000	3.6	112	3.7
a2	Sunday Motorent	80000	2.3	231	4.8
a3	Favian Motorent	80000	5	32	4.7
a4	RPM	80000	7.5	26	4.7
a5	Wuzz Rental Motor	75000	7.6	98	4.7
a6	Putrajaya Tour	80000	7.1	22	4.5
a7	Daffa	80000	7.1	21	4.7
a8	Omah Motorent	70000	9.9	303	4.8
a9	Chibi	80000	6.9	75	4.9
a10	Partnertrip	80000	7.3	25	4.9

Tabel 4.16 merupakan matriks keputusan. Sebagai contoh, nilai 80000 pada baris a1 kolom K1 berarti nilai kriteria K1 dari alternatif a1 bernilai 80000.

Selanjutnya langkah-langkah proses pengerjaan metode TOPSIS menggunakan masukan data tersebut adalah sebagai berikut:

a. Menghitung Matriks Ternormalisasi

Matriks ternormalisasi dihitung menggunakan Persamaan 2.4 dan Persamaan 2.5 dari matriks keputusan. Hasil seperti pada Tabel 4.17.

Tabel 4.17 Matriks keputusan Ternormalisasi

Kode Alternatif	K1	K2	K3	K4
a1	0.101910828	0.079741379	0.125139665	0.055987558
a2	0.101910828	0.103448276	0.258100559	0.035769829
a3	0.101910828	0.101293103	0.03575419	0.077760498
a4	0.101910828	0.101293103	0.029050279	0.116640747
a5	0.095541401	0.101293103	0.109497207	0.118195956
a6	0.101910828	0.096982759	0.024581006	0.110419907
a7	0.101910828	0.101293103	0.023463687	0.110419907
a8	0.089171975	0.103448276	0.338547486	0.153965785
a9	0.101910828	0.105603448	0.027932961	0.107309487
a10	0.101910828	0.105603448	0.027932961	0.113530327

Tabel 4.17 adalah hasil dari normalisasi matriks keputusan pada Tabel 4.16. Sebagai contoh angka 0.079741379 pada baris a1 kolom K2 merupakan hasil dari pembagian antara angka pada baris a1 kolom K2 pada Tabel 4.16 dibagi dengan jumlah seluruh nilai pada kolom K2.

b. Menghitung Matriks Ternormalisasi Terbobot

Selanjutnya adalah menghitung matriks ternormalisasi terbobot dengan cara mengkalikan matriks pada Tabel 4.18 dengan bobot

prioritas pada Tabel 4.15 sesuai dengan Persamaan 2.6. Hasilnya dapat dilihat pada Tabel 4.18.

Tabel 4.18 Matriks keputusan Ternormalisasi Terbobot

Kode Alternatif	K1	K2	K3	K4
a1	0.065514112	0.005696	0.008939	0.011997332
a2	0.065514112	0.007389	0.018436	0.007664962
a3	0.065514112	0.007235	0.002554	0.016662962
a4	0.065514112	0.007235	0.002075	0.024994443
a5	0.06141948	0.007235	0.007821	0.025327702
a6	0.065514112	0.006927	0.001756	0.023661406
a7	0.065514112	0.007235	0.001676	0.023661406
a8	0.057324848	0.007389	0.024182	0.032992664
a9	0.065514112	0.007543	0.001995	0.022994887
a10	0.065514112	0.007543	0.001995	0.024327924

Tabel 4.18 adalah hasil dari perkalian normalisasi matriks keputusan pada Tabel 4.17 dengan bobot prioritas pada Tabel 4.15 pada kriteria yang sama. Sebagai contoh angka 0.005696 pada baris a1 kolom K2 merupakan hasil dari perkalian antara angka pada baris a1 kolom K2 pada Tabel 4.17 dikali dengan bobot prioritas K2 pada Tabel 4.15.

c. Menentukan Solusi Ideal Positif dan Negatif

Solusi ideal positif dihitung menggunakan Persamaan 2.7 sehingga diperoleh hasil pada Tabel 4.19.

Tabel 4.19 Matriks Solusi Ideal Positif

K1	K2	K3	K4
0.057324848	0.007543101	0.024181954	0.007664962

Solusi ideal negatif dihitung menggunakan Persamaan 2.8 sehingga diperoleh hasil pada Tabel 4.20.

Tabel 4.20 Matriks Solusi Ideal Negatif

K1	K2	K3	K4
0.065514112	0.005695811	0.001675977	0.032992664

d. Menghitung Jarak Nilai Setiap Alternatif dengan Matriks Solusi Ideal Positif dan Negatif

Jarak nilai setiap alternatif dengan matriks solusi ideal positif didapatkan dari Persamaan 2.9. Hasilnya dapat dilihat pada Tabel 4.21.

Tabel 4.21 Jarak Nilai Setiap Alternatif dengan Solusi Ideal Positif

Kode Alternatif	Solusi Positif
a1	0.017933419
a2	0.01000533
a3	0.024817269
a4	0.029260661
a5	0.024423484
a6	0.028744781
a7	0.028802153
a8	0.02532817
a9	0.028183723
a10	0.028930432

Jarak nilai setiap alternatif dengan matriks solusi ideal negatif didapatkan dari Persamaan 2.10. Hasilnya dapat dilihat pada Tabel 4.22.

Tabel 4.22 Jarak Nilai Setiap Alternatif dengan Solusi Ideal Negatif

Kode Alternatif	Solusi Negatif
a1	0.022215959
a2	0.030417919
a3	0.016425579
a4	0.008154788
a5	0.01075414
a6	0.009412514
a7	0.009457387
a8	0.024009383
a9	0.010172017
a10	0.008865219

e. Menentukan Nilai Preferensi pada Masing-masing Alternatif

Selanjutnya adalah menghitung nilai preferensi pada masing-masing alternatif dengan Persamaan 2.10. Hasilnya dapat dilihat pada Tabel 4.23.

Tabel 4.23 Jarak Preferensi Masing-masing Alternatif

Kode Alternatif	Preferensi
a1	0.553332587
a2	0.752485747
a3	0.398264906
a4	0.21795242
a5	0.305709669
a6	0.246676649
a7	0.247190288
a8	0.486635065
a9	0.265201957
a10	0.23455659

Dari jarak preferensi tersebut didapatkan urutan alternatif adalah a2, a1, a8, a3, a5, a9, a7, a6, a10, a4.

4.3 Perancangan Sistem

Proses perancangan sistem ini akan dibagi menjadi 4 tahap, yaitu perancangan UML, perancangan basis data, perancangan *pseudocode* dan perancangan antarmuka.

4.3.1 Perancangan UML

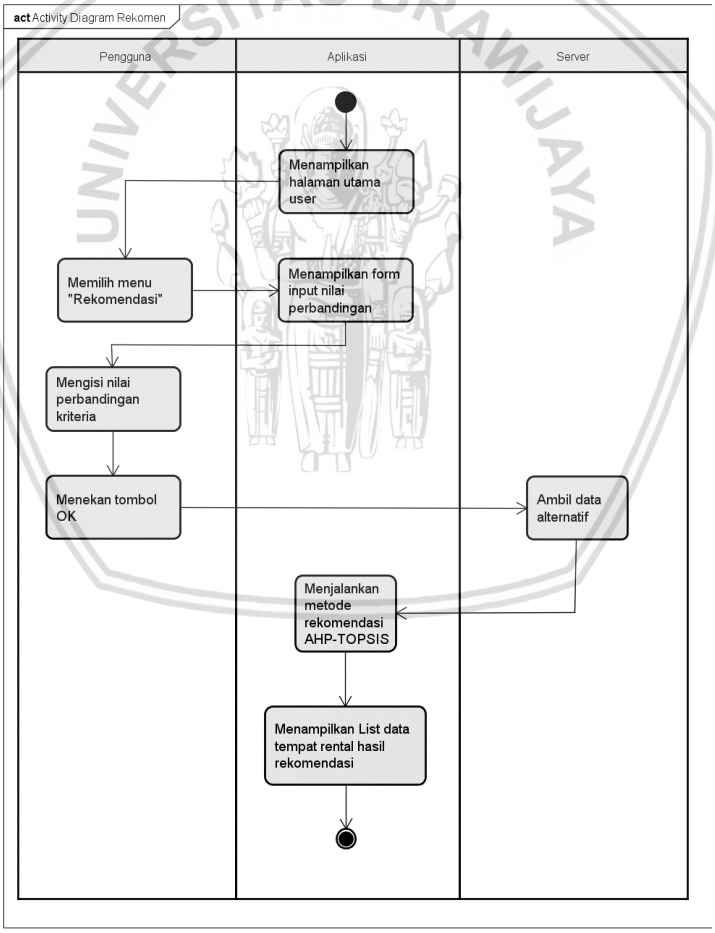
Dalam perancangan UML ini terdiri dari *activity diagram, class diagram, dan sequence diagram*.

4.3.1.1 Activity Diagram

Activity diagram dibuat untuk memodelkan aktivitas pengguna terhadap sistem berdasarkan *use case* yang sudah dibuat sebelumnya.

1. *Activity Diagram* mendapatkan hasil rekomendasi tempat rental motor

   Alur dari mendapatkan hasil rekomendasi tempat rental motor dapat dilihat di Gambar 4.5.



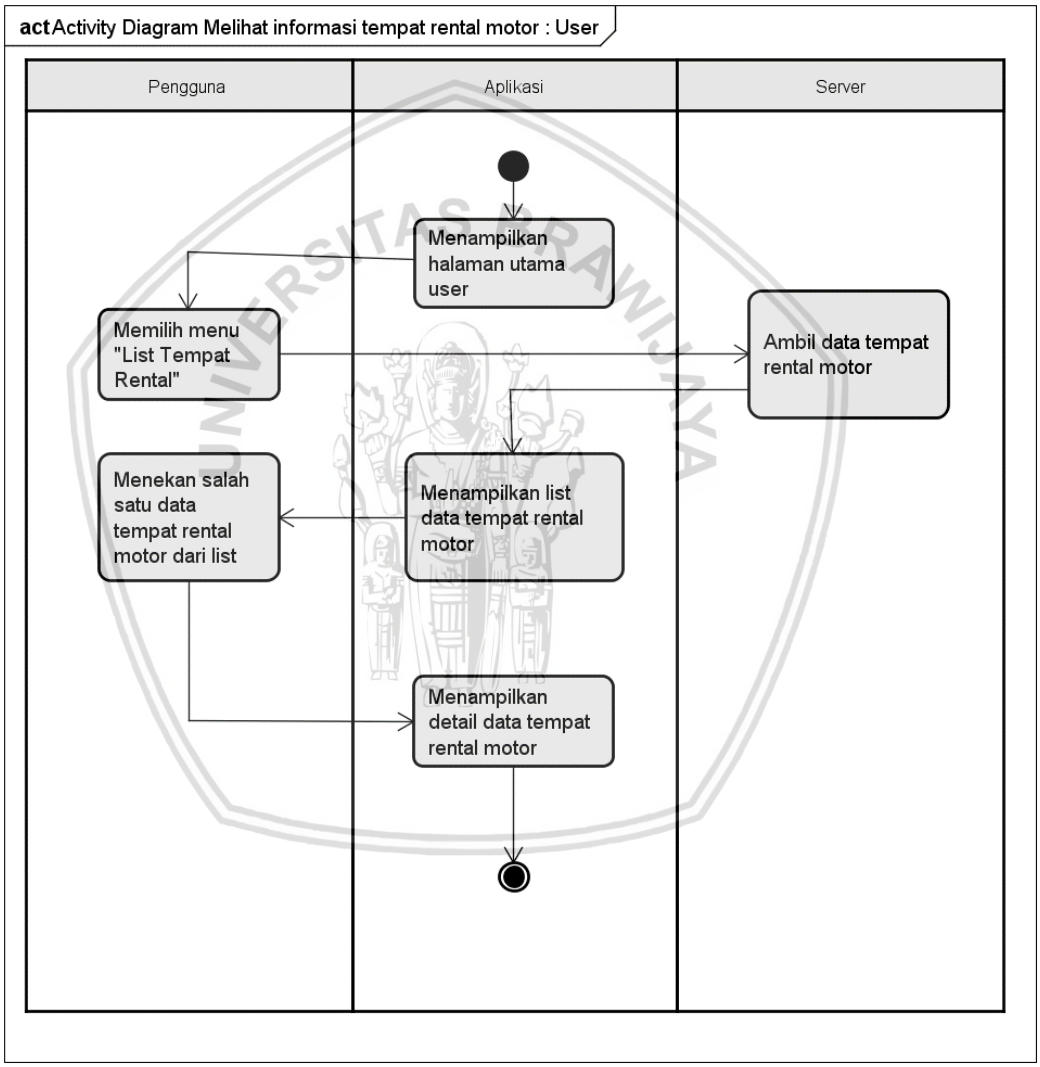
Gambar 4.5 *Activity Diagram* Mendapatkan Hasil Rekomendasi Tempat Rental Motor

Aktor *user* akan memilih menu “Rekomendasi” dari halaman utam *user* yang akan di response dengan form *input* nilai perbandingan, dimana di

halaman ini *user* akan mengisi nilai perbandingan kriteria. Setelah *user* menekan tombol "Cari Tempat Rental", maka aplikasi aka mengambil data alternatif dari server dan akan menggunakan nilai perbandingan dan data alternatif sebagai masukan metode AHP-TOPSIS. Setelah metode selesai maka hasil rekomendasi akan ditampilkan pada halaman hasil rekomendasi

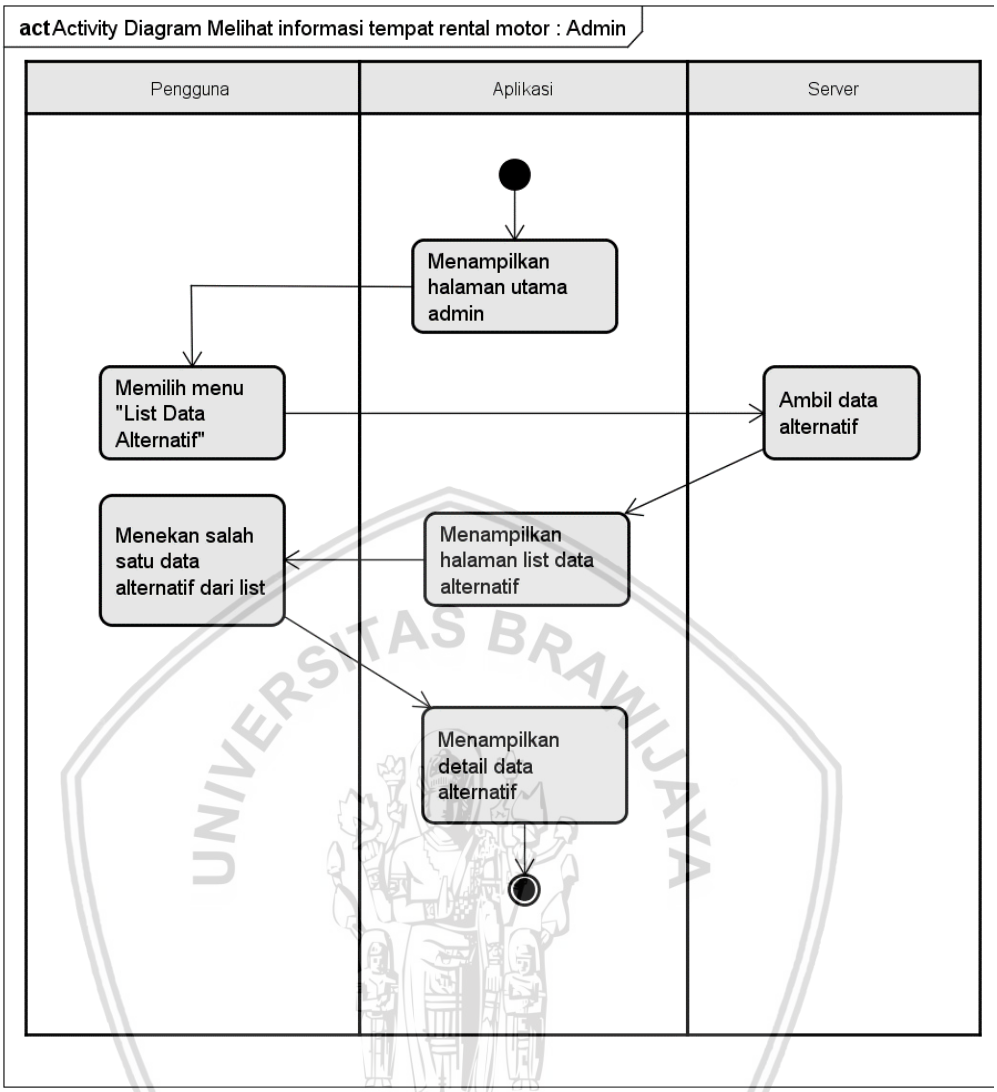
2. *Activity Diagram* melihat informasi tempat rental motor

Fungsionalitas melihat informasi tempat rental motor memiliki dua jenis alur, yang pertama adalah alur dari aktor *User* dan yang kedua adalah alur dari aktor *Admin*. Alur dari aktor *User* dapat dilihat di Gambar 4.6.



Gambar 4.6 *Activity Diagram* Melihat Informasi Tempat Rental Motor *User*

Alur dari aktor *Admin* dapat dilihat pada Gambar 4.7.

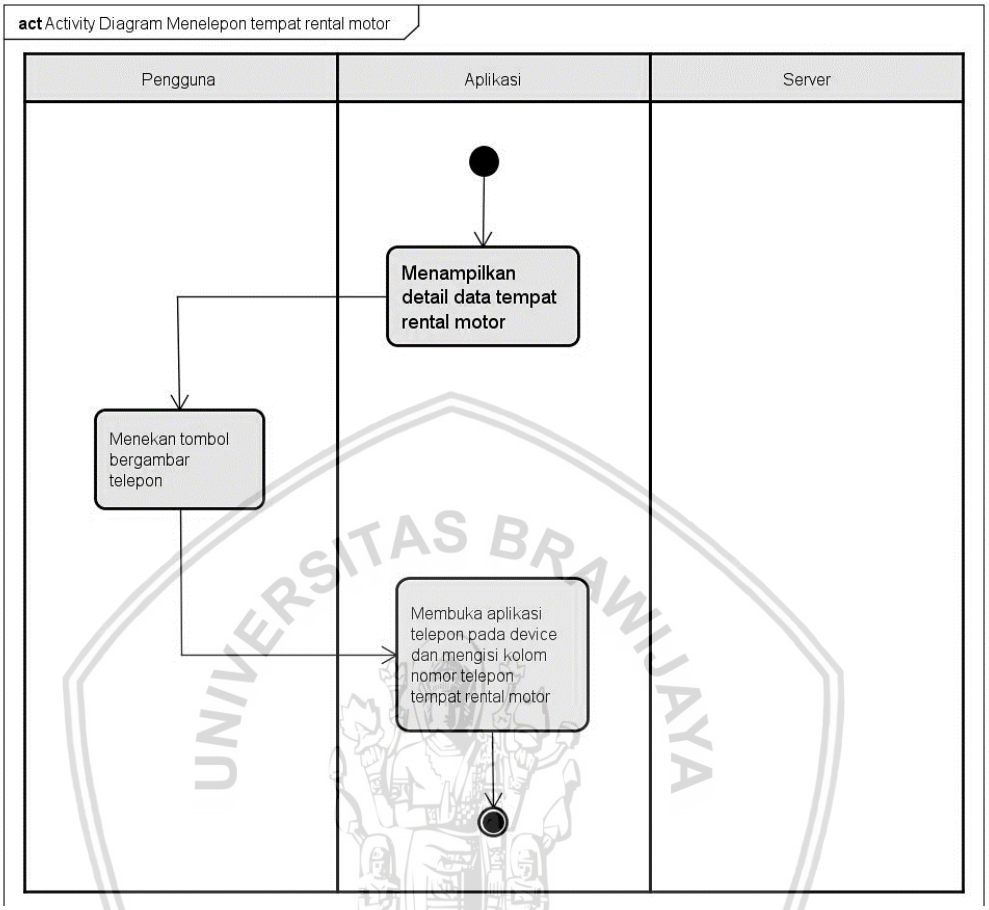


Gambar 4.7 *Activity Diagram* Melihat Informasi Tempat Rental Motor *Admin*

Dari gambar diatas dapat diketahui bahwa perbedaan dari aktor *User* dan *Admin* dalam alur melihat informasi tempat rental motor adalah jenis halaman utama dan penamaan menu yang berbeda. Menu pada aktor *User* adalah List Tempat Rental sedangkan menu pada *Admin* adalah List Data Alternatif. Hal tersebut dikarenakan *User* dianggap kurang mengerti akan istilah Data Alternatif dan *Admin* dianggap paham tentang metode AHP-TOPSIS sehingga familiar dengan istilah Data Alternatif. Setelah memilih menu tersebut alur melihat informasi tempat rental motor pada aktor *User* maupun *Admin* sama, yaitu aplikasi akan mengambil data alternatif atau tempat rental pada database dan akan ditampilkan pada halaman list. Pada halaman list ini aktor akan memilih salah satu data yang akan ditampilkan detail datanya pada halaman detail data tempat rental motor atau detail data alternatif.

3. *Activity Diagram* menelepon tempat rental motor

Alur dari menelepon tempat rental motor dapat dilihat di Gambar 4.8.



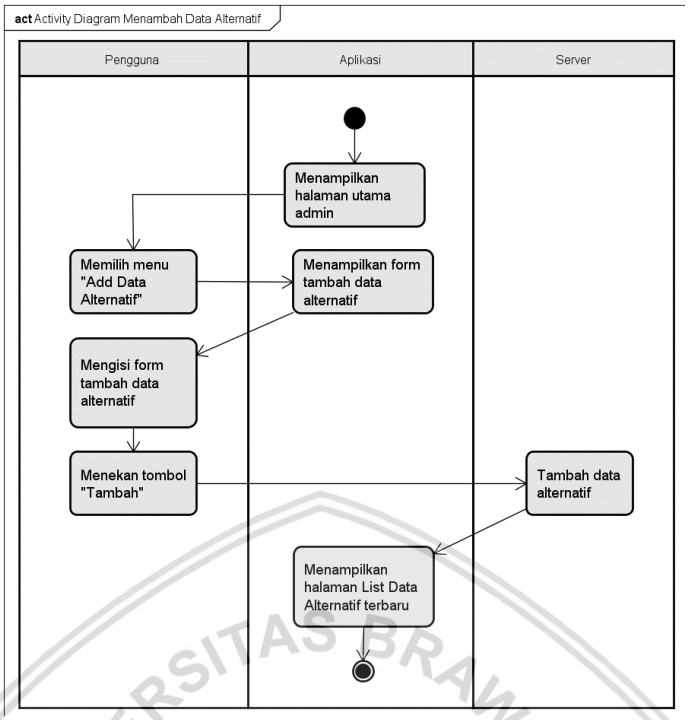
Gambar 4.8 *Activity Diagram* Menelepon Tempat Rental Motor

Berdasarkan Gambar 4.8 *user* dapat melakukan menelepon tempat rental motor dengan cara menekan tombol bergambar telepon pada halaman detail tempat rental motor. Setelah itu *user* akan menerima tampilan aplikasi telepon pada perangkat sedang menelepon nomor telepon tempat rental sesuai dengan databasenya.

4. *Activity Diagram* Menambah Data Alternatif

Alur dari menambah data alternatif adalah yang pertama aktor *admin* memilih menu Add Data Alternatif pada halaman utama *admin* yang akan diresponse dengan halaman form tambah data alternatif. Pada halaman ini *admin* akan mengisi data pada form yang akan ditambahkan pada database server. Setelah itu *admin* akan menerima tampilan halaman list data alternatif dimana salah satu data dari list merupakan data alternatif yang baru ditambahkan.

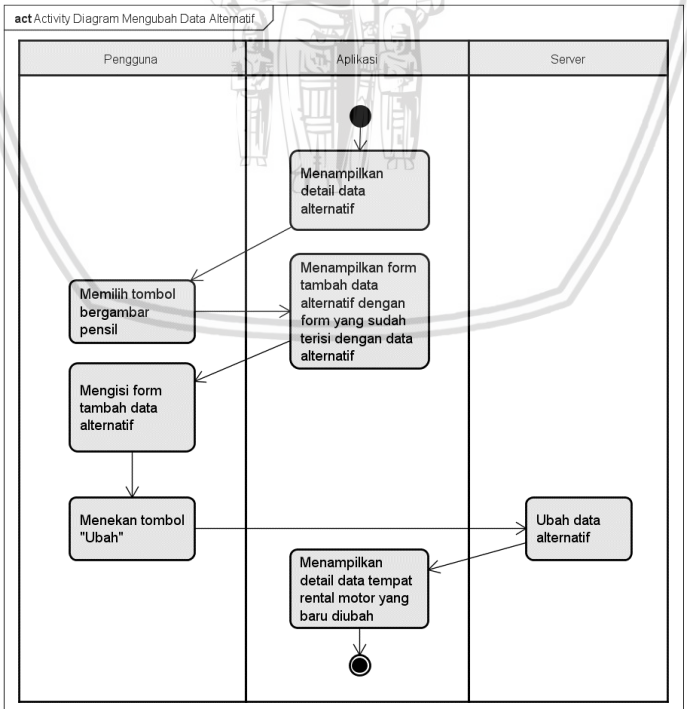
Alur dari menambah data alternatif dapat dilihat di Gambar 4.9.



Gambar 4.9 *Activity* Menambah Data Alternatif

5. *Activity Diagram* Mengubah Data Alternatif

Alur dari mengubah data alternatif dapat dilihat di Gambar 4.10.

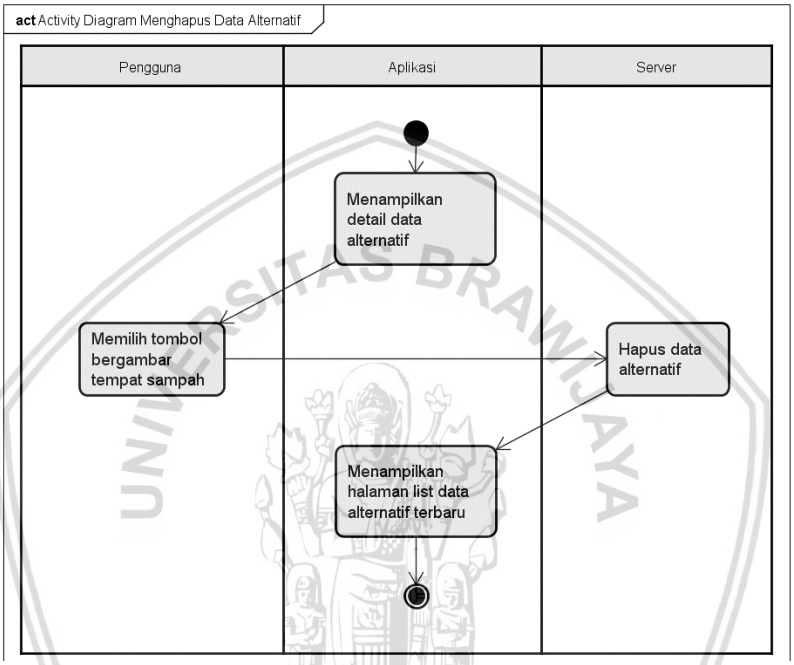


Gambar 4.10 *Activity Diagram* Mengubah Tempat Rental Motor

Aktor *admin* memilih tombol bergambar pensil pada halaman detail data alternatif. Setelah itu aplikasi akan menampilkan halaman form tambah data alternatif namun kolom formnya sudah terisi oleh data alternatif yang ingin dirubah. *Admin* akan mengisi form tersebut dengan data yang baru dimana akan mengubah data pada database server. Setelah itu aplikasi akan menampilkan halaman detail data alternatif yang telah dirubah.

6. *Activity Diagram* Menghapus Data Alternatif

Alur dari menghapus data alternatif dapat dilihat di Gambar 4.11.



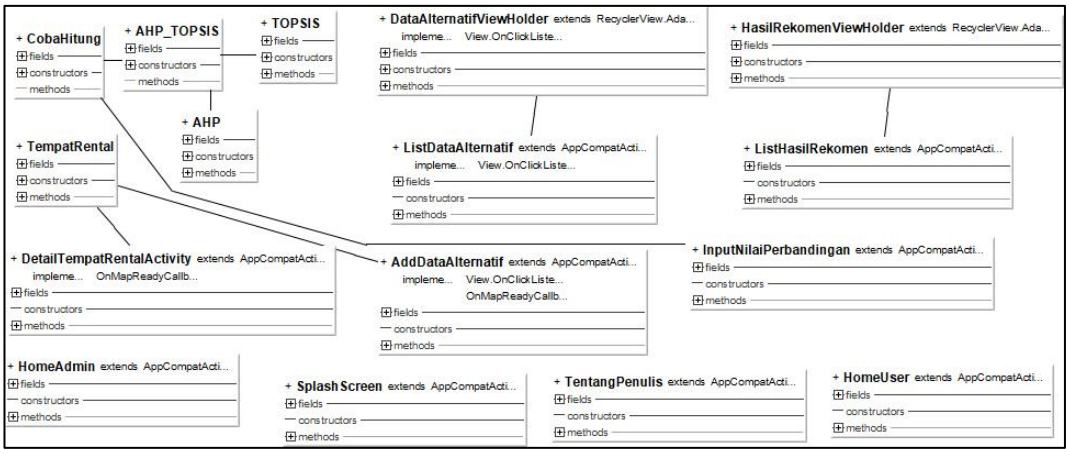
Gambar 4.11 *Activity Diagram* Menghapus Tempat Rental Motor

Aktor *admin* memilih tombol bergambar tempat sampah pada halaman detail data alternatif. Setelah itu data alternatif pada database server akan terhapus lalu aplikasi akan menampilkan halaman list data alternatif yang terbaru setelah dilakukannya penghapusan data.

4.3.1.2 Class Diagram

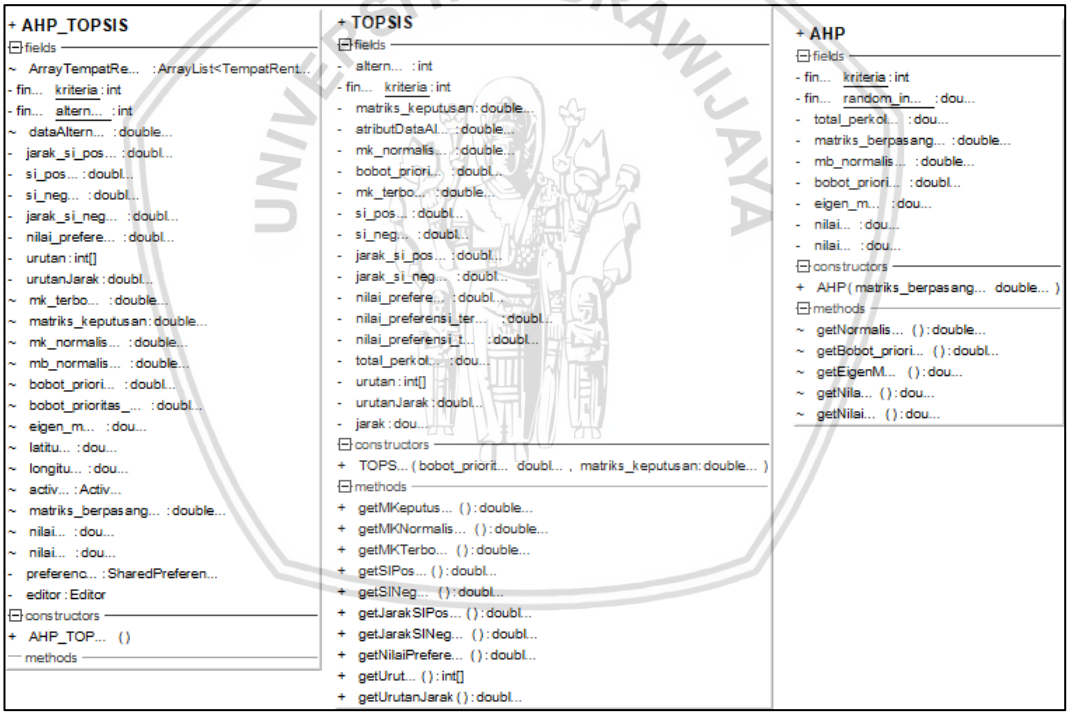
Class diagram ini akan menampilkan class-class apa saja yang akan digunakan dalam pengembangan perangkat lunak ini dan menunjukkan class-class yang saling berhubungan.

Class Diagram juga dilengkapi dengan variable dan method yang digunakan dalam class tersebut. Gambaran lengkap class-class diatas beserta atribut dapat dilihat pada gambar 4.12.



Gambar 4.12 *Class Diagram*

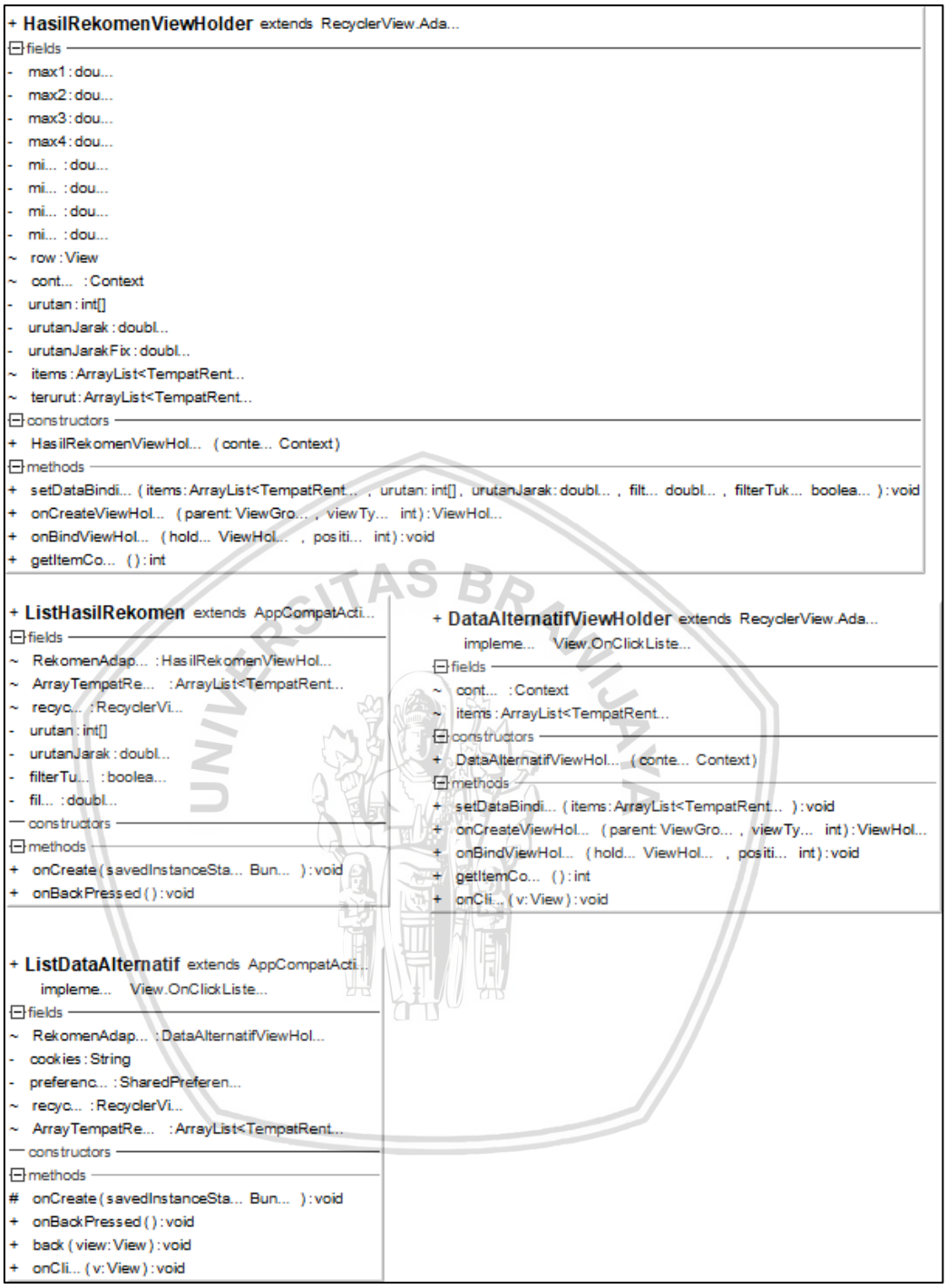
Gambar 4.12 menggambarkan keseluruhan *class* yang digunakan pada aplikasi ini. *Class-class* tersebut dikelompokkan menjadi *package-package* yang didalamnya berisi *class-class* yang memiliki fungsi yang sama. *Package* yang pertama adalah *Package MetodeSPK* yang dapat dilihat pada Gambar 4.13.



Gambar 4.13 *Class Diagram Package Metode SPK*

Package *Metode SPK* berisi *class-class* yang berfungsi menghitung *input* data menggunakan metode sistem pendukung keputusan yaitu AHP dan TOPSIS menjadi *output* yang bisa diproses untuk ditampilkan pada *User*. *Package* ini terdiri dari *class AHP_TOPSIS, AHP,* dan *TOPSIS.*

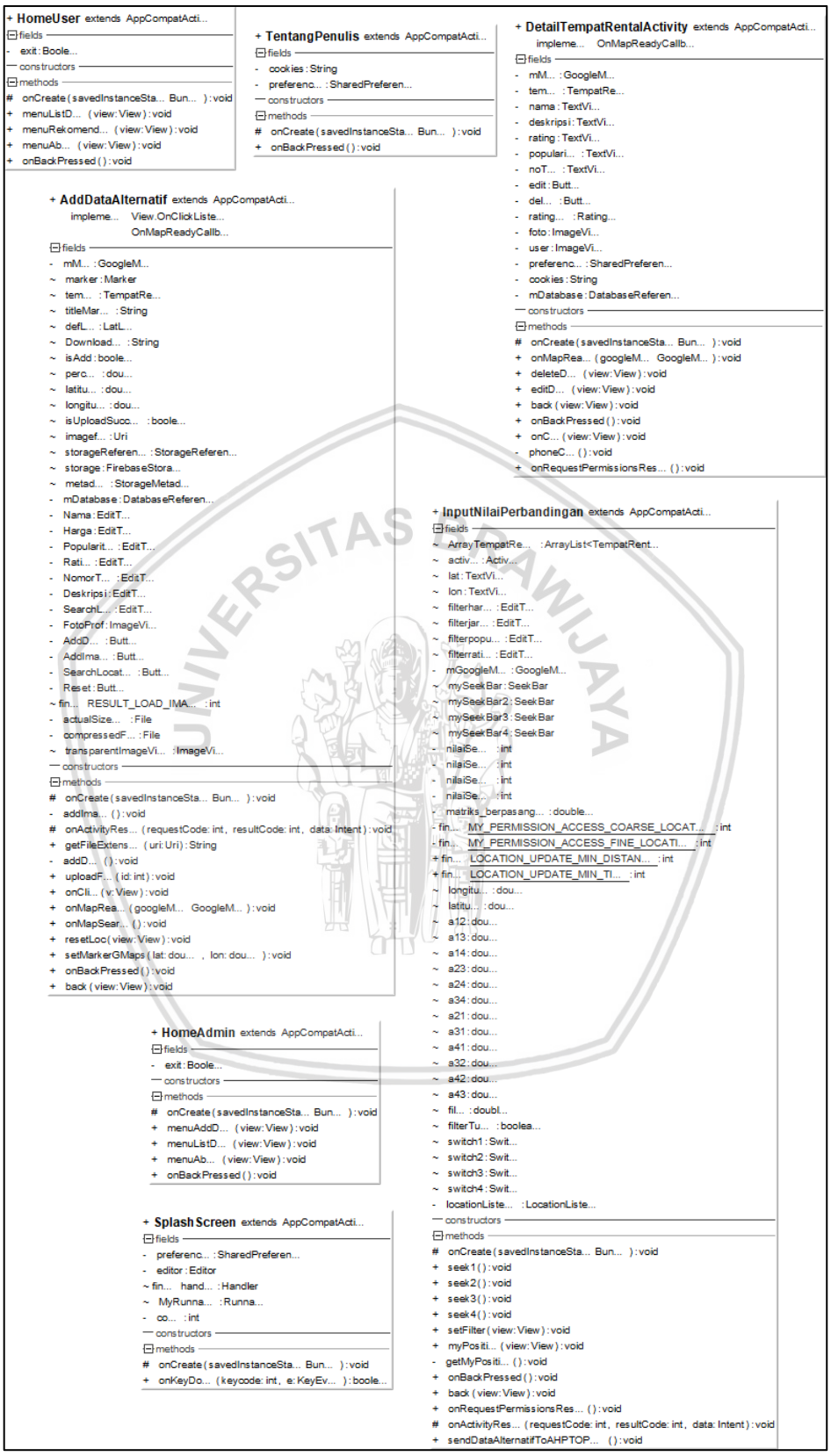
Package selanjutnya adalah *Package Adapters* yang dapat dilihat pada Gambar 4.14.



Gambar 4.14 *Class Diagram Package Adapters*

Package Adapters adalah *package* yang berisi *class-class* untuk menampilkan *list* Hasil Rekomendasi dan *list* Data Alternatif beserta *adapter view holder*-nya. *Package* ini terdiri dari *class ListHasilRekomen* dan *ListDataAlternatif* yang berfungsi sebagai menampilkan *list item* Hasil Rekomendasi dan Data Alternatif. Lalu ada *class HasilRekomenViewHolder* dan *DataAlternatifViewHolder* sebagai *view holder* masing-masing *item* pada *list.*

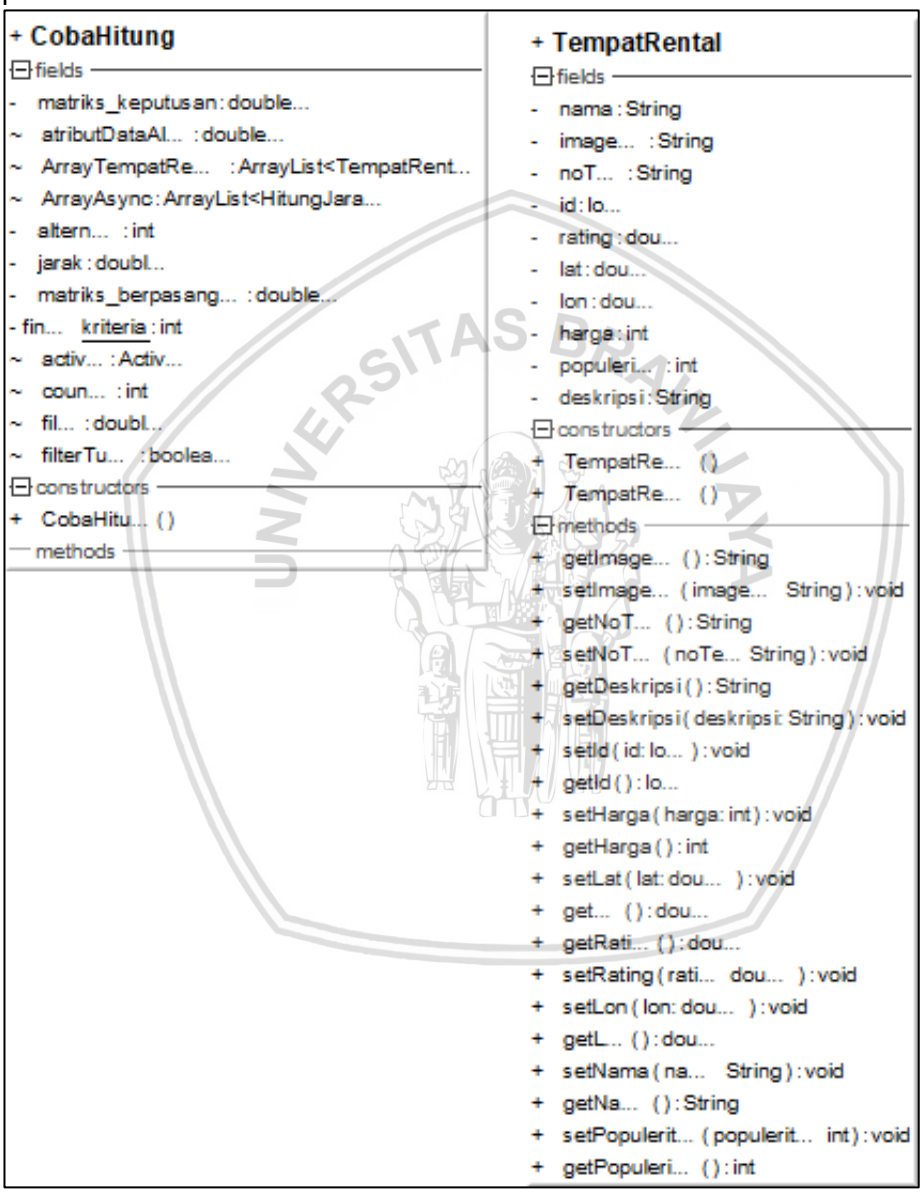
Package selanjutnya adalah *Package Activities* yang dapat dilihat pada Gambar 4.15.



Gambar 4.15 *Class Diagram Package Activities*

Package Activties adalah *package* yang berisi *class-class activity* untuk memberikan tampilan kepada *User*, tampilan ini termasuk tampilan untuk mengisi bobot kriteria, tampilan awal aplikasi, tampilan halaman utama *user* dan *admin*, tampilan detail tempat rental dan tampilan untuk mengisi data alternarif. *Package* ini terdiri dari *class AddDataAlternatif, DetailTempatRentalActivity, HomeAdmin, HomeUser, InputNilaiPerbandingan, SplashScreen, TentangPenulis*.

Package selanjutnya adalah *Package ServiceAndClassObject* yang dapat dilihat pada Gambar 4.16.



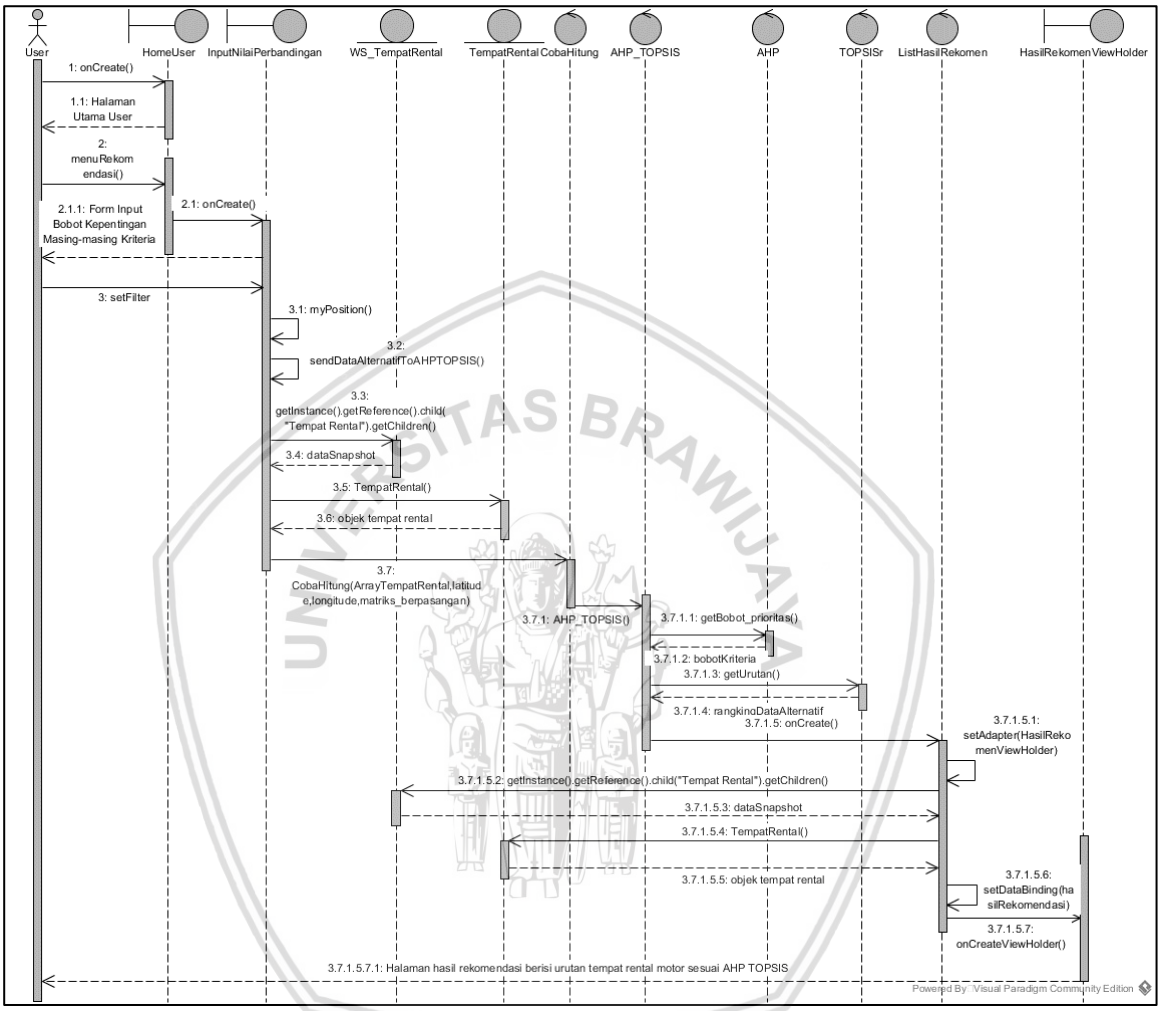
Gambar 4.16 *Class Diagram Package ServiceAndClassObject*

Package ServiceAndClassObject adalah *package* yang berisi *class service CobaHitung* untuk mencari jarak antara lokasi *user* dengan lokasi masing-masing tempat rental dan *class object TempatRental* sebagai enkapsulasi atribut-atribut dari data tempat rental.

4.3.1.3 Sequence Diagram

1. *Sequence Diagram* mendapatkan hasil rekomendasi tempat rental motor

   Diagram ini menggambarkan proses mendapatkan hasil rekomendasi rental motor melalui sistem pendukung keputusan yang hasilnya akan ditampilkan pada ViewHolder. Untuk lebih jelas bisa dilihat pada Gambar 4.17.



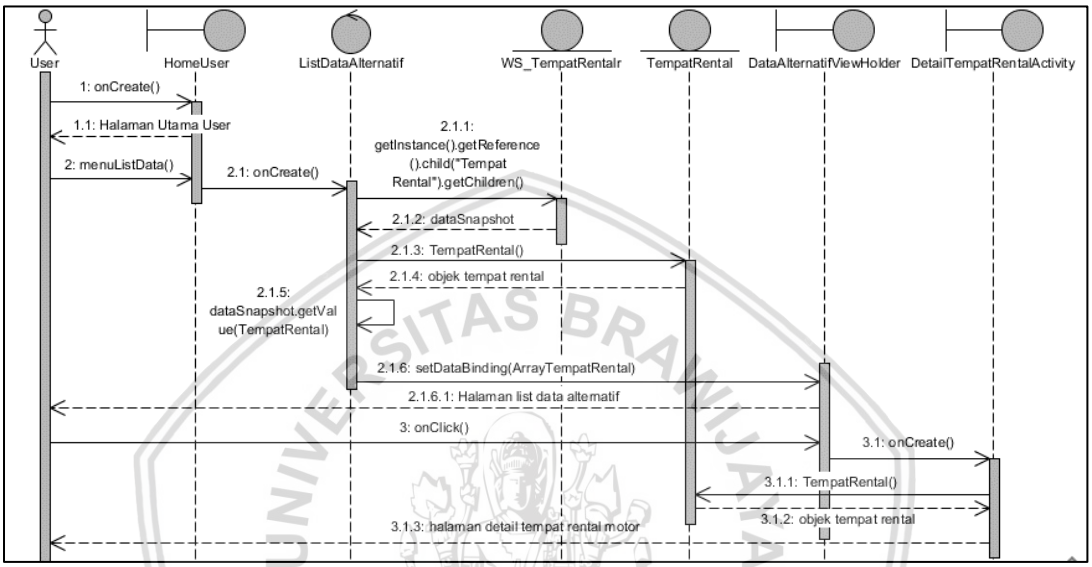
Gambar 4.17 *Sequence Diagram* Mendapatkan Hasil Rekomendasi Tempat Rental Motor

Ketika *user* membuka halaman *HomeUser* maka *user* memanggil method *onClick()*, lalu *user* memilih menu "Rekomendasi" yang akan memanggil *method menuRekomendasi()* yang akan membuka class InputNilaiPerbandingan. Di antarmuka class ini *user* akan mengisi nilai perbandingan yang akan memanggil method myPosition yang akan memanggil method *sendDataAlternatifToAHPTOPSIS()* pada *class ModelDataAlternatif*. Data Alternatif dan Nilai Perbandingan akan diproses pada *class AHP_TOPSIS*, *AHP*, dan *TOPSIS.* Hasilnya akan dikirim ke *class ListHasilRekomen* yang akan menampilkan masing-masing Data Alternatif melalui *class HasilRekomenViewHolder* melalui *method setDataBinding*. Lalu tampilan

halaman hasil rekomendasi yang berisi 3 terbaik tempat rental motor akan diterima oleh *user*.

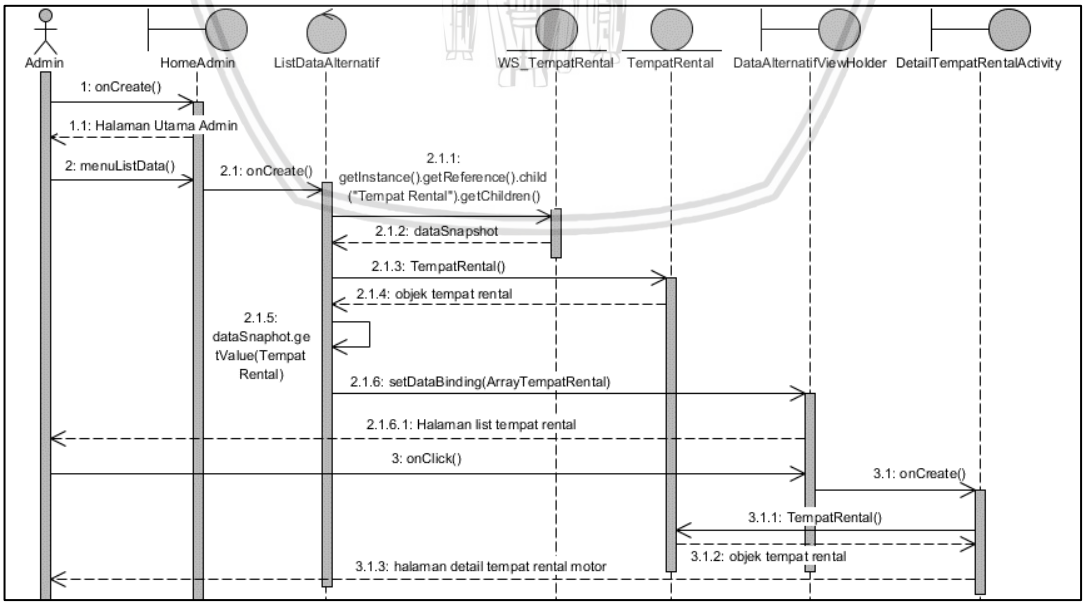
2. *Sequence Diagram* melihat informasi tempat rental motor

Untuk fungsionalitas melihat informasi tempat rental motor memiliki 2 *sequence diagram*, yaitu sequence untuk Aktor *User* dan untuk Aktor *Admin.* Untuk *sequence diagram* melihat informasi tempat rental motor Aktor *User* dapat dilihat pada Gambar 4.18.



Gambar 4.18 *Sequence Diagram* Melihat Informasi Tempat Rental Motor *User*

Untuk *sequence diagram* melihat informasi tempat rental motor Aktor *Admin* dapat dilihat pada Gambar 4.19.

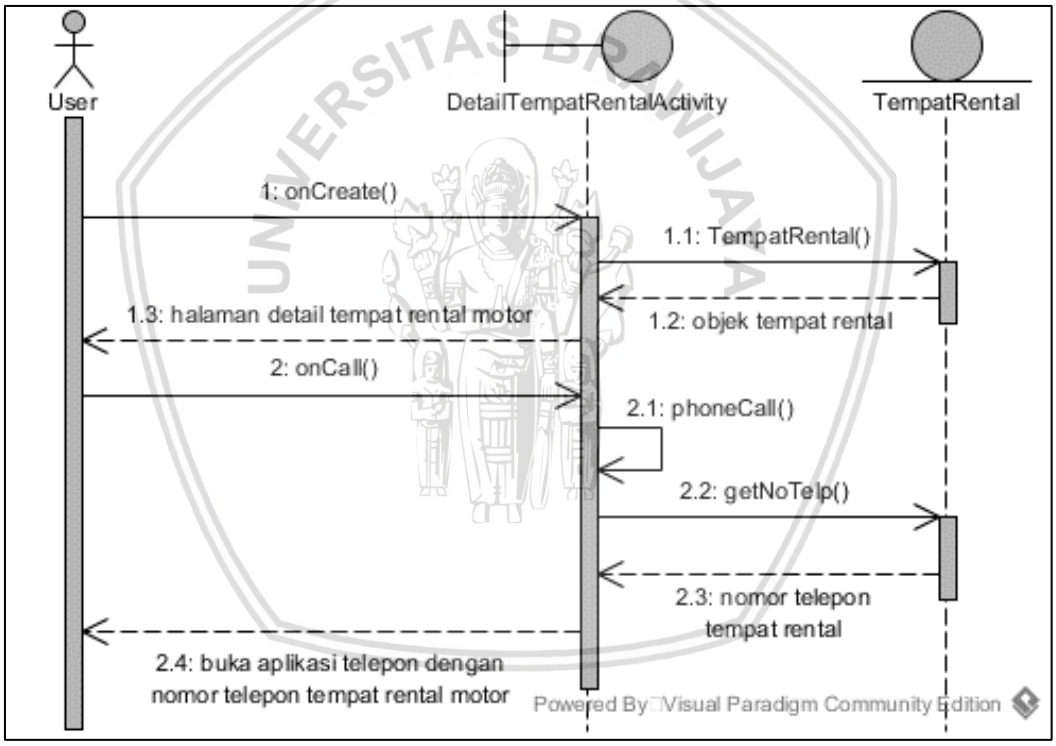


Gambar 4.19 *Sequence Diagram* Melihat Informasi Tempat Rental Motor *Admin*

Kedua sequence diagram diatas hanya memiliki perbedaan pada halaman utama. *User* membuka halaman utama pada *class HomeUser* sedangkan *admin* pada class Home*Admin*. Ketika kedua aktor memilih menu "List Data Alternatif" atau "List Tempat Rental" maka kedua aktor sama-sama memanggil *class ListDataAlternatif* yang akan yang akan menampilkan masing-masing Data Alternatif melalui *class DataAlternatifViewHolder* melalui *method setDataBinding*. Lalu tampilan halaman *list* tempat rental. Ketika aktor memilih salah satu data maka akan memanggil *method onClick()* pada *DataAlternatifViewHolder* yang akan memanggil *class DetailTempatRentalActivity.*

3. *Sequence Diagram* menelepon tempat rental motor

Diagram ini menggambarkan proses menelepon tempat rental motor dengan menekan *image button* bergambarkan telepon pada halaman detail tempat rental motor. Untuk lebih jelas bisa dilihat pada Gambar 4.20.

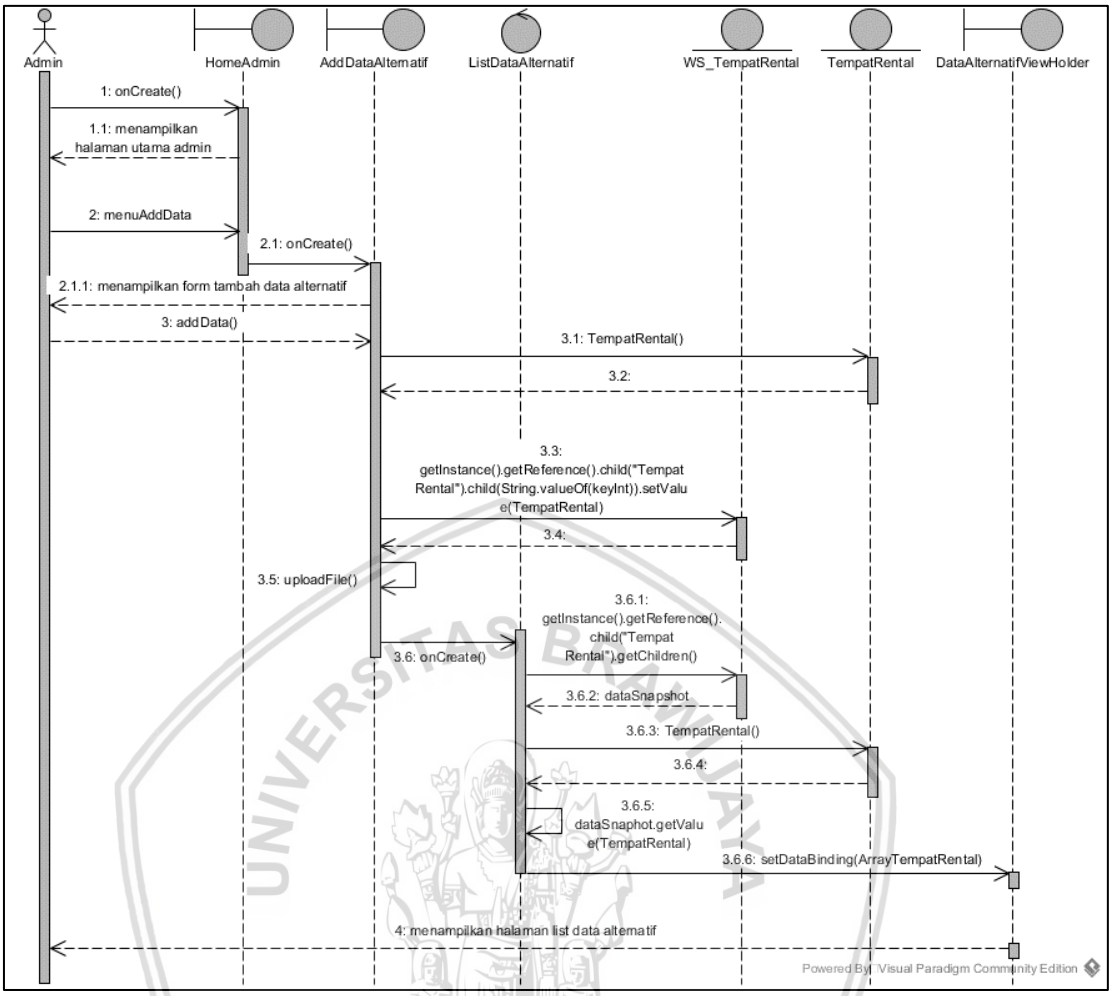


Gambar 4.20 *Sequence Diagram* Menelepon Tempat Rental Motor

Ketika *user* membuka halaman Detail Tempat Rental, *method onCreate()* sudah terpanggil. Lalu *user* menekan tombol bergambar telepon yang akan memanggil *method phoneCall()* yang akan membuka aplikasi telepon dengan berisi nomor telepon tempat rental motor.

4. *Sequence Diagram* menambah data alternatif

Sequence Diagram menambah data alternatif dapat dilihat pada Gambar 4.21.

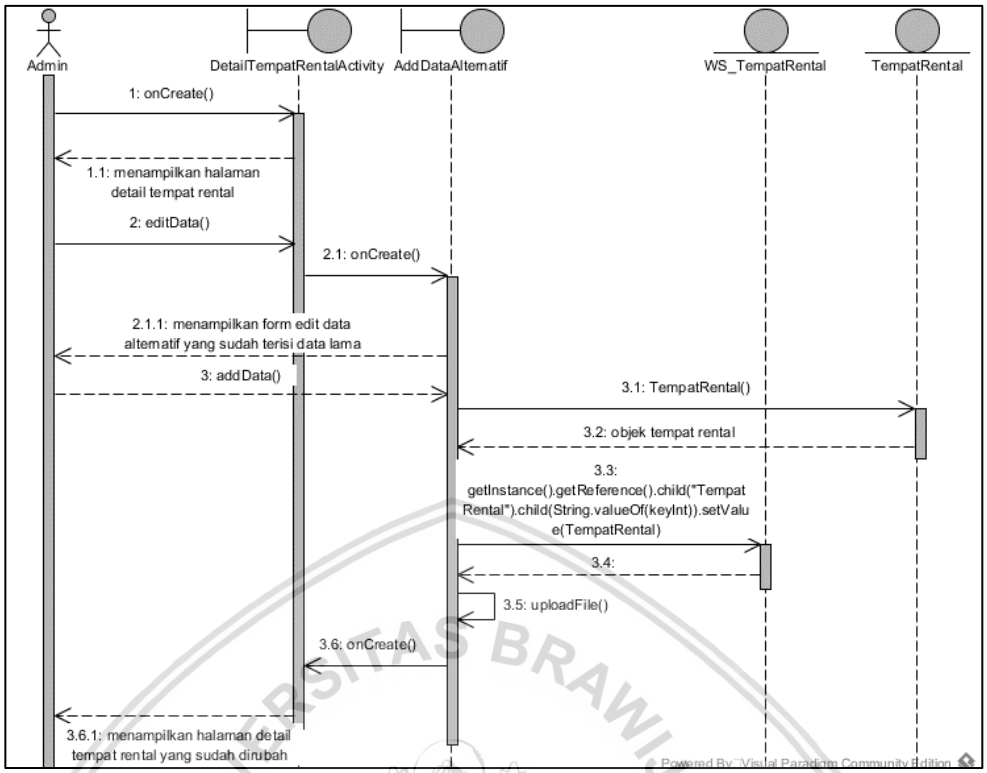


Gambar 4.21 *Sequence Diagram* Menambah data Alternatif

Ketika *admin* membuka halaman utama *admin* pada *class HomeAdmin, method onCreate()* sudah terpanggil, lalu *admin* akan memilih menu "Add Data Alternatif" yang akan memanggil *method menuAddData()* pada *class HomeAdmin* yang akan memanggil *method onCreate()* pada *class AddDataAlternatif* yang akan menampilkan form data alternatif. Setelah *admin* mengisi semua kolom data pada form dan memilih tombol "Add Data" maka *method addData()* pada *class AddDataAlternatif* akan terpanggil dan akan memanggil *method onCreate()* pada *class ListDataAlternatif.* Pada *class ListDataAlternatif* akan memanggil *method setDataBinding()* pada *class DataAlternatifViewHolder* yang akan menampilkan halaman *list* data alternatif.

5. *Sequence Diagram* mengubah data alternatif

   Sequence Diagram mengubah data alternatif dapat dilihat pada Gambar 4.22.

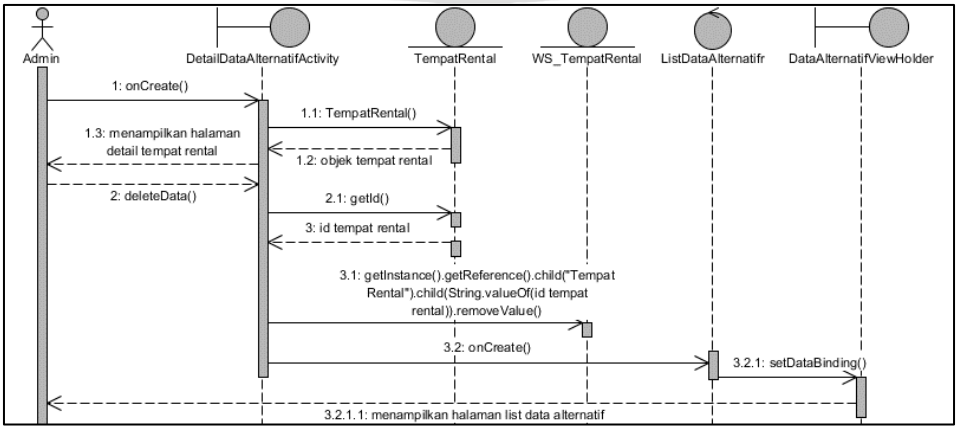


Gambar 4.22 *Sequence Diagram* Mengubah data Alternatif

Ketika *admin* membuka halaman Detail Tempat Rental, *method onCreate()* sudah terpanggil. Ketika *admin* memilih tombol "*Edit*" bergambar pensil maka *method editData()* pada *class DetailTempatRentalActivity* akan memanggil *onCreate()* pada *class AddDataAlternatif* yang akan menampilkan halaman form *edit* data alternatif yang sudah terisi data lama. Ketika *admin* memilih tombol "Save Data Alternatif" maka *method addData* akan terpanggil yang akan menampilkan halaman detail tempat rental yang sudah dirubah dengan memanggil *method onCreate()* pada *class DetailTemoatRentalActivity.*

6. *Sequence Diagram* menghapus data alternatif

Sequence Diagram menghapus data alternatif dapat dilihat pada Gambar 4.23.



Gambar 4.23 *Sequence Diagram* Menghapus data Alternatif

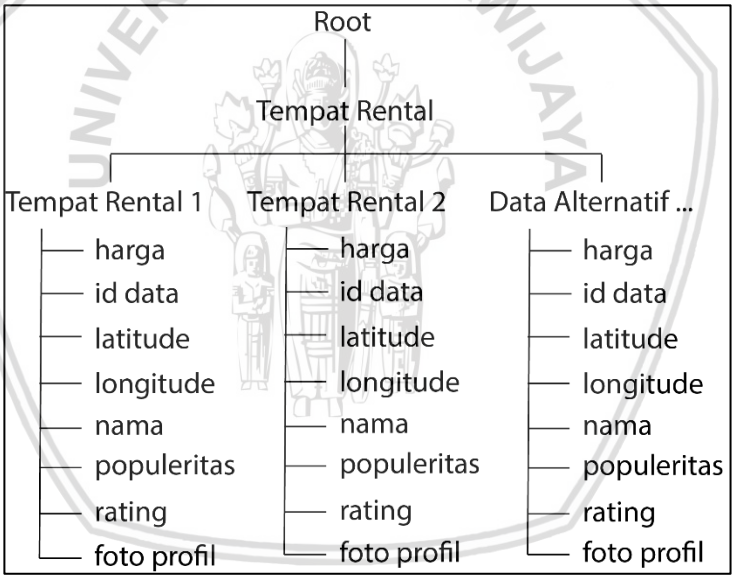
Ketika *admin* membuka halaman detail tempat rental maka *method onCreate()* sudah terpanggil. Ketika *admin* memilih tombol "Hapus" dengan gambar tempat sampah maka akan memanggil *method deleteData()* yang akan memanggil *method onCreate* pada *class ListDataAlternatif.* Pada *class* ini akan memanggil *method setDataBinding()* pada *DataAlternatifViewHolder* yang akan menampilkan halaman *list* data alternatif.

4.3.2 Perancangan Basis Data

Pada perancangan basis data akan dijelaskan tentang perancangan basis data untuk menyimpan data yang diperlukan oleh sistem. Basis data yang digunakan oleh sistem berbentuk NoSQL, maka dari itu basis data tidak berbentuk tabel melainkan berbentuk JSON tree yang terdiri dari node-node.

Untuk pengganti tabel, maka ditambahkan node baru sebagai child pada node Root dengan nama Data Alternatif. Dalam node ini ditambah node baru sebagai child sebagai kumpulan Data alternatif yang memiliki atribut sebagai child berupa harga rental motor, id data, latitude dan longitude lokasi tempat rental motor, nama tempat rental motor, jumlah popularitas, *rating*, dan foto profil.

Gambaran perancangan basis data bisa dilihat pada Gambar 4.24.



Gambar 4.24 Perancangan Basis Data

4.3.3 Perancangan *Pseudocode*

Perancangan ini berisi *pseudocode* dari method dari class-class yang digunakan dalam aplikasi ini. *Pseudocode* yang dituliskan adalah *pseudocode* dari 3 method sample yaitu *method setBobot_prioritas* dari *class AHP*, *method setNilaiPreferensi* dari *class TOPSIS*, dan *method setEigenMaks* dari *class AHP*.

1. *Pseudocode Method setBobot_prioritas*

   Method setBobot_prioritas diawali dengan melakukan *input mb_normalisasi* yang berbentuk variable *array* 2D dan *input* jumlah kriteria. Selanjutnya membuat *looping for* dari 0 sampai jumlah kriteria, didalamnya dibuat juga *looping for* dari 0 sampai jumlah kriteria, di dalam *looping* kedua

ini dilakukan pengisian *array bobot_prioritas* berdasarkan jumlah dengan *array mb_normalisasi index* berdasarkan *variable looping*. Di dalam *looping* pertama dilakukan update isi *bobot_prioritas* dengan membagi masing-masing data dari *bobot_prioritas* dengan jumlah kriteria.

Pseudocode method setBobot_prioritas dapat dilihat pada Tabel 4.24.

Tabel 4.24 *Pseudocode Method setBobot_prioritas*

```
input mb_normalisasi
input jumlah kriteria
FOR i dari 0 sampai jumlah kriteria
      FOR j dari 0 sampai jumlah kriteria
            bobot_prioritas index ke-i= bobot_prioritas
             index ke-i + mb_normalisasi array 2D index
             ke- i,j
      END FOR
      bobot_prioritas index ke-i= bobot_prioritas index
       ke-i / jumlah kriteria
END FOR
```

2. *Pseudocode Method setNilaiPreferensi*

Method setNilaiPreferensi diawali dengan melakukan *input jarak_si_negatif* dan *jarak_si_positif* yang keduanya berbentuk *variable array* 2D dan *input* jumlah data alternatif. Selanjutnya dibuat *looping for* sebanyak jumlah data alternatif yang didalamnya dilakukan proses pengisian *array nilai_preferensi* dengan cara pembagian *jarak_si_negatif* dengan jumlah antar *jarak_si_negatif* dan *jarak_si_positif* yang memiliki *index* yang sama.

Pseudocode method setNilaiPreferensi dilihat pada Tabel 4.25.

Tabel 4.25 *Pseudocode Method setNilaiPreferensi*

```
input jumlah data alternatif
input jarak_si_negatif
input jarak_si_positif
FOR i dari 0 sampai jumlah data alternatif
      nilai_preferensi index ke-i = jarak_si_negatif
       index ke-i / (jarak_si_negatif index ke-i +
       jarak_si_positif index ke-i)
END FOR
```

3. *Pseudocode Method setEigenMaks*

Method ini diawali dengan melakukan *input array* 2D *matriks_berpasangan* dan jumlah kriteria. Selanjutnya dibuat *looping for* dengan *variable i* sejumlah kriteria, setiap *looping for i* dilakukan juga *looping variable j* sejumlah kriteria juga. Setiap *looping* j dilakukan pengisian *variable nilai_lepentingan* dengan menjumlahkannya dengan *matriks_berpasangan index* ke *i,j* sesuai *looping for i* dan *j*. Selain menjalankan *looping j*, setiap *looping i* juga menjalankan pengisian *nilai_kepentingan* dengan mengkalikannya dengan *bobot_prioritas index* ke *i*, lalu nilai kepentingan tersebut ditambahkan dengan *variable eigen_maks*.

Pseudocode method setEigenMaks dilihat pada Tabel 4.26.

Tabel 4.26 *Pseudocode Method setEigenMaks*

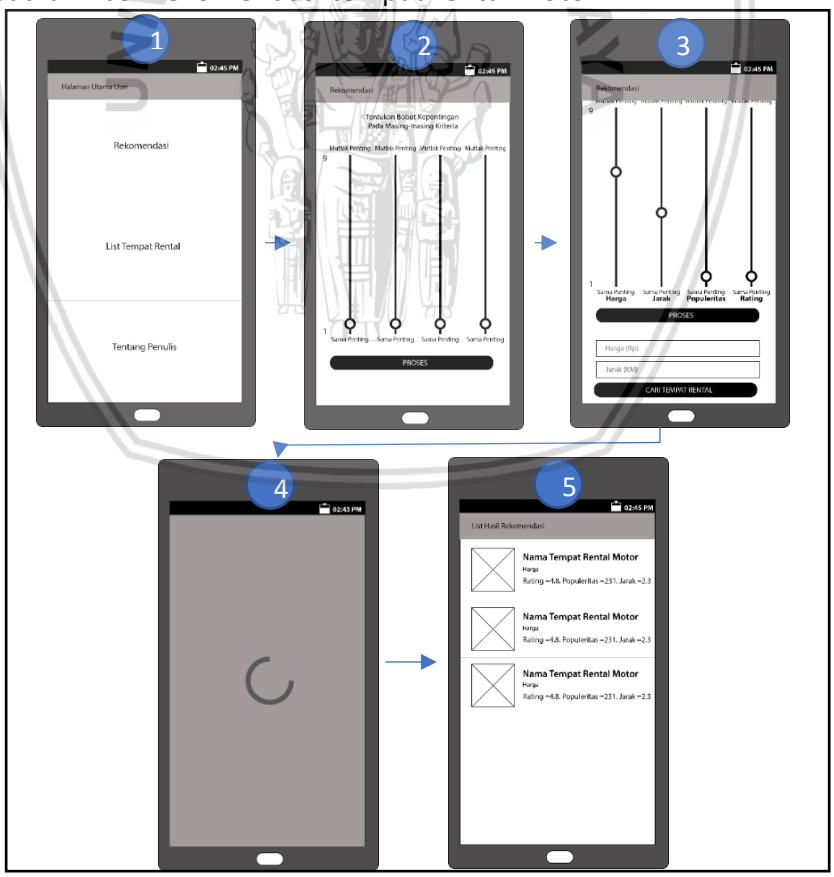
```
input matriks_berpasangan
input jumlah kriteria
FOR i dari 0 sampai jumlah kriteria
      FOR j dari 0 sampai jumlah kriteria
            nilai_kepentingan=nilai_kepentingan +
             matriks_berpasangan array 2D index ke-i,j
      END FOR
      nilai_kepentingan=nilai_kepentingan *
       bobot_prioritas index ke-i
      eigen_maks=eigen_maks + nilai_kepentingan
END FOR
```

4.3.4 Perancangan Antarmuka

Dalam perancangan antarmuka ini menjelaskan bagaimana rancangan antarmuka halaman dari aplikasi dan alur perpindahan antar halaman.

4.3.4.1 Perancangan *Screen Flow* Mendapatkan Hasil Rekomendasi Tempat Rental Motor

Gambar 4.25 adalah urutan halaman ketika menjalankan fungsionalitas Mendapatkan hasil rekomendasi tempat rental motor.

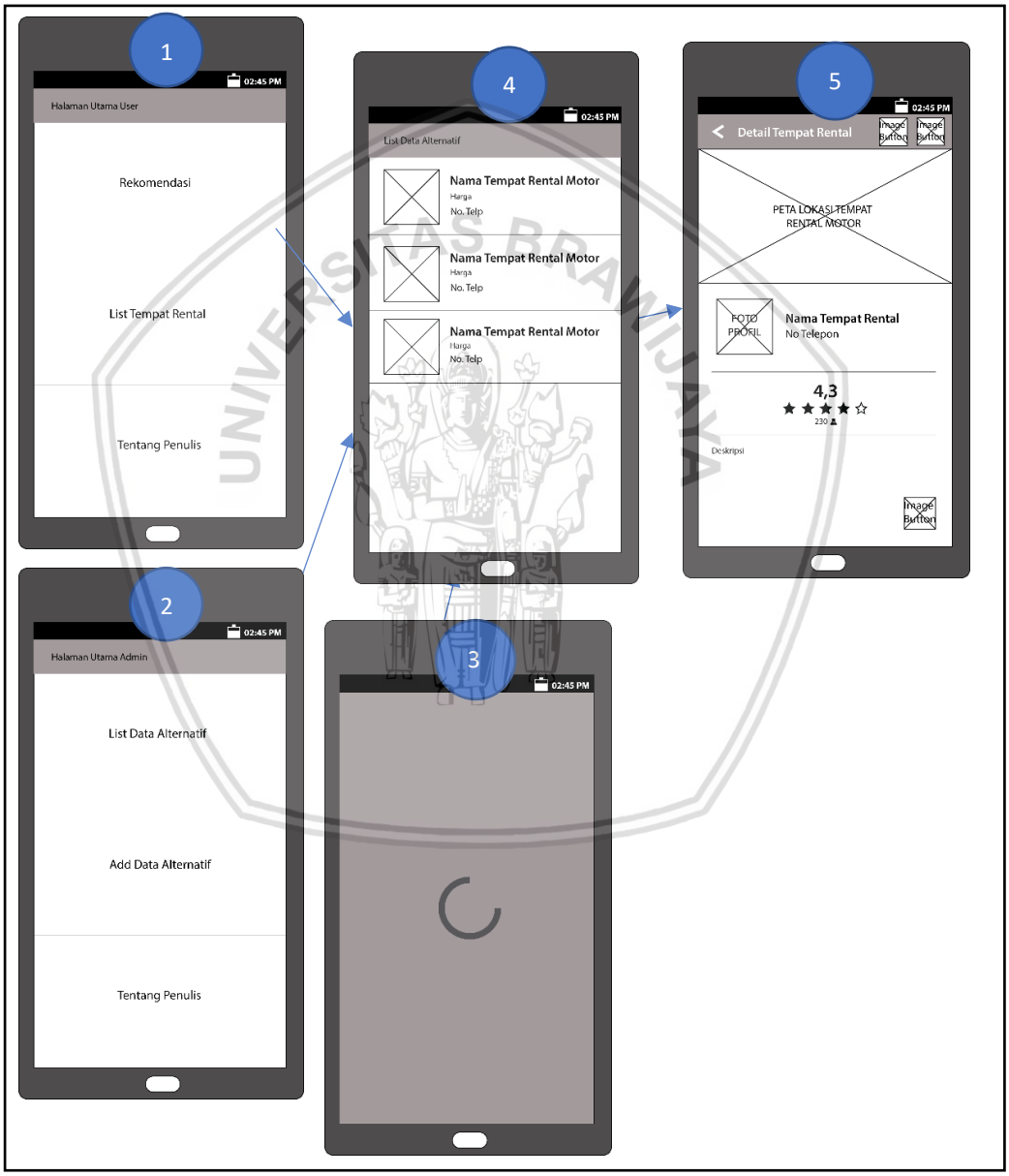


Gambar 4.25 Rancangan *Screen Flow* Mendapatkan Hasil Rekomendasi Tempat Rental Motor

Urutan tampilan Screen Flow mendapatkan hasil rekomendasi tempat rental motor diawali dengan memilih menu “Rekomendasi” pada tampilan 1 sehingga muncul tampilan tampilan 2. Ketika tombol “Proses” diklik maka akan muncul form baru pada tampilan 3 berdasarkan kriteria yang dimasukkan. Lalu muncul tampilan 4 sampai *loading* selesai sehingga muncul halaman tampilan 5.

4.3.4.2 Perancangan *Screen Flow* Melihat Informasi Tempat Rental Motor

Gambar 4.26 adalah urutan halaman ketika menjalankan fungsionalitas Melihat informasi tempat rental motor.



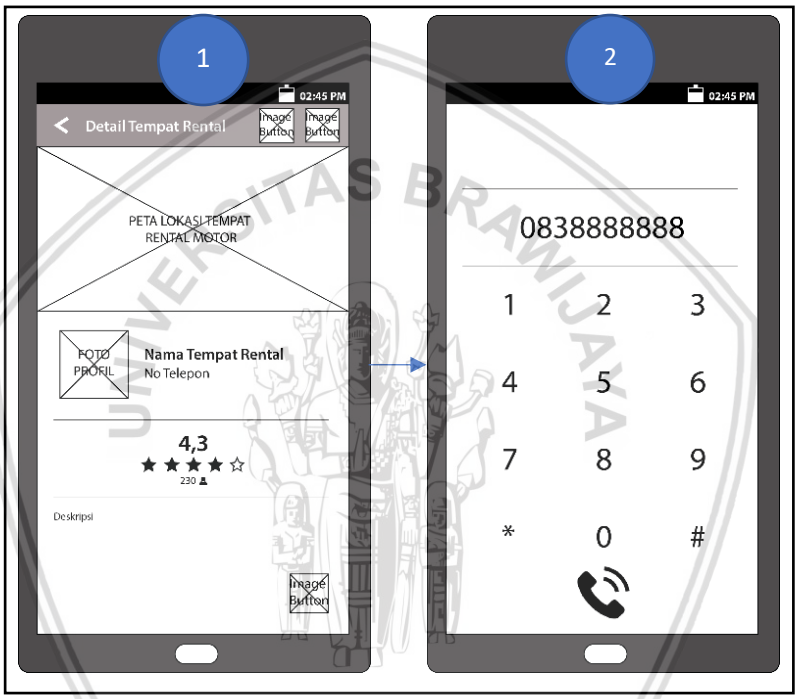
Gambar 4.26 Rancangan *Screen Flow* Melihat Informasi Tempat Rental Motor

Urutan tampilan *Screen Flow* melihat informasi tempat rental motor diawali dengan memilih menu “List Tempat Rental” pada tampilan 1 pada Aktor *User*, dan menu “List Data Alternatif” pada tampilan 2 pada Aktor *Admin*,

atau setelah menyelesaikan fungsionalitas mendapatkan hasil rekomendasi tempat rental motor sehingga muncul tampilan tampilan 4. Pada tampilan 4 Aktor akan memilih salah satu *item* sehingga muncul halaman tampilan 5. Di tampilan 5 ada perbedaan antara Aktor *User* dengan *Admin*. Pada Aktor *User* tidak akan ada *Image Button* disamping judul halaman yang berjumlah 2. Pada Aktor *Admin* akan muncul 2 *Image Button* disamping judul halaman seperti di tampilan 5.

4.3.4.3 Perancangan *Screen Flow* Menelepon Tempat Rental Motor

Gambar 4.27 adalah urutan halaman ketika menjalankan fungsionalitas Menelepon tempat rental motor.

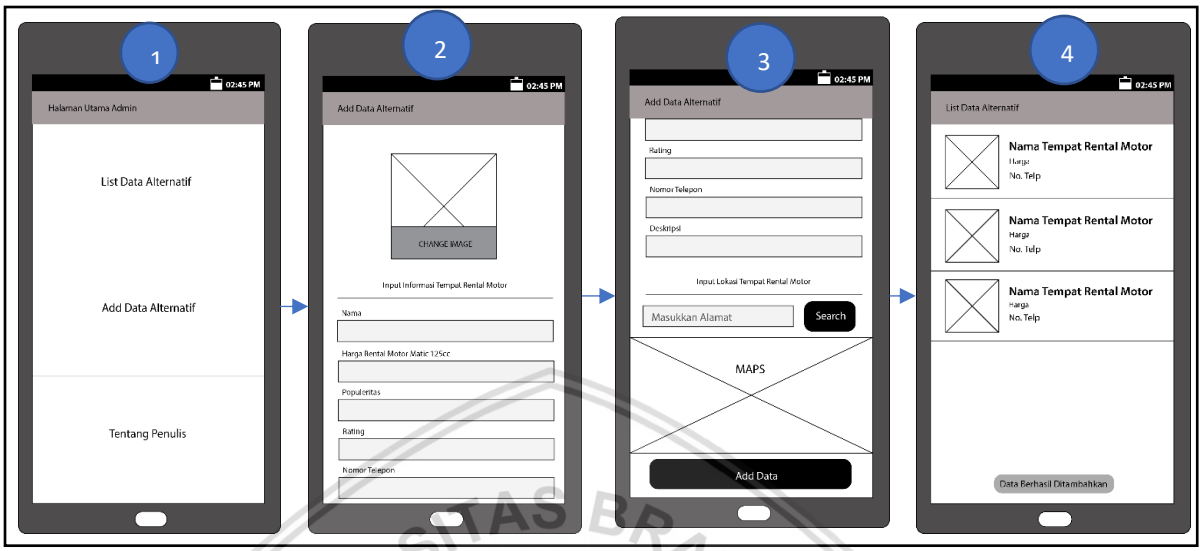


Gambar 4.27 Rancangan *Screen Flow* Menelepon Tempat Rental Motor

Urutan tampilan *Screen Flow* menelepon tempat rental motor diawali dengan meng-klik *Image Button* pada pojok kanan bawah pada tampilan 1, sehingga muncul perkiraan halaman Telepon seperti tampilan 2.

4.3.4.4 Perancangan *Screen Flow* Menambah Data Alternatif

Gambar 4.28 adalah urutan halaman ketika menjalankan fungsionalitas menambah data alternatif.



Gambar 4.28 Rancangan *Screen Flow* Menambah Data Alternatif

Urutan tampilan *Screen Flow* menambah data alternatif diawali dengan memilih menu "Add Data Alternatif" pada tampilan 1, sehingga muncul *form* pada tampilan 2 dan 3. Jika *form* terisi dan klik tombol "Add Data" maka muncul tampilan 4 berisi *List Data Alternatif* beserta notifikasi berhasil.

4.3.4.5 Perancangan *Screen Flow* Mengubah Data Alternatif

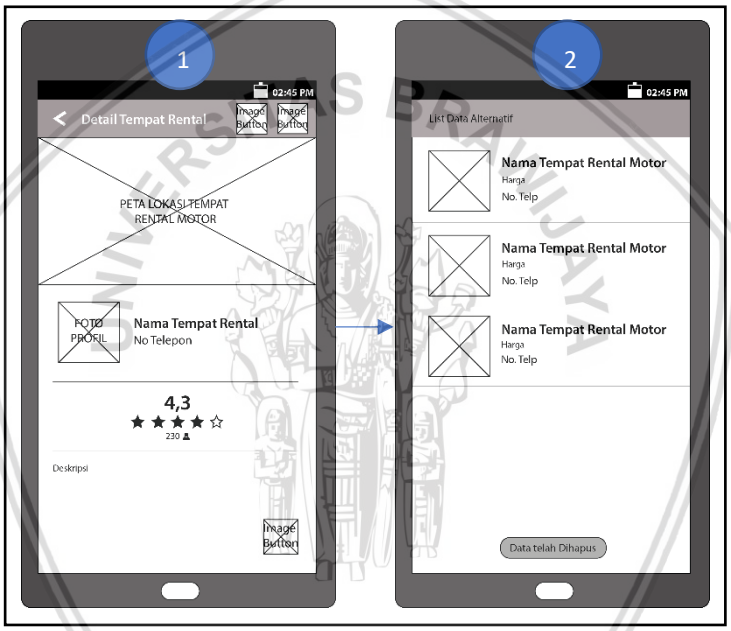
Gambar 4.29 adalah urutan halaman ketika menjalankan fungsionalitas mengubah data alternatif.

Gambar 4.29 Rancangan *Screen Flow* Mengubah Data Alternatif

Urutan tampilan *Screen Flow* mengubah data alternatif diawali dengan memilih *image button* yang berada di pojok kanan atas yang berada pada sisi sebelah kanan pada tampilan 1, sehingga muncul *form* yang sama seperti halaman Add Data Alternatif pada tampilan 2 dan 3 namun memiliki perbedaan pada judul halaman menjadi *Edit* Data Alternatif. Kolom *edit text* pada *form* tersebut pun telah terisi *hint* atribut data alternatif sebelum dilakukan perubahan data. Tombol "Add Data" juga berubah menjadi "Save Data Alternatif". Jika *form* terisi dan klik tombol "Save Data Alternatif" maka muncul tampilan 4 berisi halaman Detail tempat Rental dengan notifikasi berhasil.

4.3.4.6 Perancangan *Screen Flow* Menghapus Data Alternatif

Gambar 4.30 adalah urutan halaman ketika menjalankan fungsionalitas menghapus data alternatif.



Gambar 4.30 Rancangan *Screen Flow* Menghapus Data Alternatif

Urutan tampilan *Screen Flow* menghapus data alternatif diawali dengan memilih *image button* yang berada di pojok kanan atas yang berada pada sisi sebelah kiri pada tampilan 1, sehingga muncul halaman List Data Alternatif yang sudah berkurang *item*-nya dengan notifikasi berhasil pada tampilan 2.

BAB 5 IMPLEMENTASI

Bab ini membahas tentang pembuatan aplikasi Android rekomendasi tempat rental motor di kota malang dengan metode AHP TOPSIS berbasis *location based services* berdasarkan rekayasa kebutuhan dan perancangan yang telah dibuat sebelumnya. Pembahasan implementasi terdiri dari penjelasan tentang spesifikasi lingkungan implementasi, batasan-batasan implementasi, implementasi basis data, implementasi kode program, dan implementasi antarmuka aplikasi.

5.1 Spesifikasi sistem

Implementasi sistem bekerja pada lingkungan perangkat keras dan perangkat lunak. Perangkat keras yang digunakan pada lingkungan *client* adalah perangkat bergerak (*smartphone*).

5.1.1 Spesifikasi perangkat keras

Dalam pengembangan aplikasi Android rekomendasi tempat rental motor di kota malang dengan metode AHP TOPSIS berbasis *location based services* menggunakan komputer dengan spesifikasi yang tertulis pada Tabel 5.1.

Tabel 5.1 Spesifikasi perangkat keras komputer

Nama Komponen	**Spesifikasi**
Model Sistem	Laptop Asus A456UR
Prosesor	Intel® Core™ i5 6200U CPU @2.60Ghz
Memori (HDD)	1 TB
Memori (RAM)	8 GB DDR5
Kartu Grafik	NVIDIA Geforce 930MX

Spesifikasi perangkat keras *smartphone* Android yang digunakan sebagai proses instalasi dan pengujian perangkat dijelaskan Tabel 5.2.

Tabel 5.2 Spesifikasi perangkat keras s*martphone* Android

Nama Komponen	**Spesifikasi**
Model Sistem	Samsung A5 2017
Prosesor	Exynos 7880 Octa Octa-core 1.9 GHz Cortex-A53
Memori Internal	32 GB
Memori (RAM)	3 GB
Layar	5.2 inches, FHD 1080 x 1920 pixels
Kamera	16 Megapiksel
WLAN	802.11 b/g/n

5.1.2 Spesifikasi perangkat lunak

Dalam pengembangan aplikasi Android rekomendasi tempat rental motor di kota malang dengan metode AHP TOPSIS berbasis *location based services* menggunakan komputer dengan spesifikasi perangkat lunak yang tertulis pada Tabel 5.3.

Tabel 5.3 Spesifikasi perangkat lunak komputer

Nama Komponen	**Spesifikasi**
Sistem Operasi	Windows 10 Pro © 2017 64-Bit
Bahasa Pemrograman	Java, XML
Software Development Kit	Java SE Development Kit 8 (64-Bit)
Programming Environtment	Java Runtime Environtment 8
Android SDK	API 16
*Edit*or	Android Studio

Spesifikasi perangkat lunak *smartphone* Android yang digunakan sebagai proses instalasi dan pengujian perangkat dijelaskan Tabel 5.4.

Tabel 5.4 Spesifikasi perangkat lunak *smartphone* Android

No	**Nama Komponen**	**Spesifikasi**
1	Sistem Operasi	Android Versi 7.1.1 (Nougat)

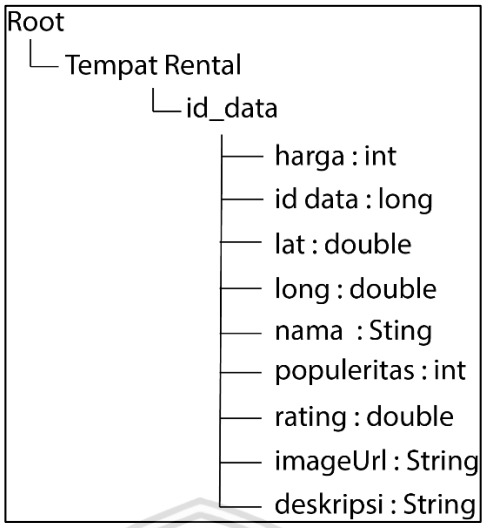
5.2 Batasan implementasi

Beberapa batasan dalam mengimplementasikan sistem adalah sebagai berikut:

1. Aplikasi harus selalu terhubung dengan internet sebagai media pertukaran data.
2. Untuk mendapatkan lokasi dari pengguna aplikasi dibutuhkan sensor GPS ataupun koneksi internet untuk mendapatkan lokasi latitude dan longtitude dari pengguna.
3. Implementasi peta menggunakan *Google Maps Android API*.

5.3 Implementasi basis data

Implementasi basis data perancangan perangkat lunak yang digunakan berbentuk *JSON Tree* pada *web service firebase*. Implementasi basis data ini sesuai dengan perancangan basis data. Implementasi basis data dapat ditunjukkan pada Gambar 5.1.



Gambar 5.1 Implementasi basis data

5.4 Implementasi kode program

Implementasi kode program ini merupakan tahap penulisan kode program berdasarkan fitur utama yang terdapat pada aplikasi rekomendasi tempat rental motor di Kota Malang yang diberi nama REKMON. Fitur tersebut adalah fitur mendapatkan hasil rekomendasi tempat rental motor. Implementasi ini berdasarkan pernacangan *pseudocode* yang telah ditulis sample-nya di bab sebelumnya. Implemetasi kode program ini juga didapat dari hasil analisis pada perancangan *class diagram* dan *sequence diagam*. Kode program menggunakan bahasa pemrograman Java.

Pada fitur tersebut memerlukan *Activity InputNilaiPerbandingan* untuk proses *input* bobot kepentingan masing-masing kriteria, proses untuk mengetahui koordinat posisi *user*, dan proses mengambil data alternatif. Lalu *Class CobaHitung* untuk menghitung jarak antara lokasi *user* dengan lokasi masing-masing data alternatif berupa tempat rental motor. Lalu *Class AHP_TOPSIS, AHP*, dan *TOPSIS* untuk memproses bobot kepentingan kriteria dan data alternatif menjadi hasil rekomendasi yang akan ditampilkan pada *Activity ListHasilRekomen* menggunakan *Class Adapter RecyclerView HasilRekomenViewHolder*.

5.4.1 Kode Program Activity InputNilaiPerbandingan

Potongan kode program Kode 5.1 adalah potongan kode program untuk proses *input* bobot kepentingan masing-masing kriteria, proses untuk mengetahui koordinat posisi *user*, dan proses mengambil data alternatif.

```
1  public class InputNilaiPerbandingan extends AppCompatActivity {
2
3      private LocationListener locationListener = new
4      LocationListener() {
5          public void onLocationChanged(Location location) {
6              longitude = location.getLongitude();
7              latitude = location.getLatitude();
8          }
```

```
9
10          public void onStatusChanged(String provider, int status,
11          Bundle extras){
12          }
13
14          public void onProviderEnabled(String provider){
15          }
16
17          public void onProviderDisabled(String provider){
18          }
19      };
20
21      public void setFilter(View view){
22          if ((nilaiSeek1==1)&&(nilaiSeek2==1)&&(nilaiSeek3==1)
23          &&(nilaiSeek4==1)){
24              myPosition(view);
25          }
26          for (int i = 0; i < 4; i++) {
27              filter[i]=Integer.MIN_VALUE;
28          }
29          switch1.setChecked(false);
30          switch2.setChecked(false);
31          switch3.setChecked(false);
32          switch4.setChecked(false);
33          filterrating.setVisibility(View.GONE);
34          filterpopuler.setVisibility(View.GONE);
35          filterjarak.setVisibility(View.GONE);
36          filterharga.setVisibility(View.GONE);
37          Button proses=(Button)findViewById(R.id.buttonProses);
38          proses.setVisibility(View.GONE);
39          if ((nilaiSeek1>1)||(nilaiSeek2>1)||
40          (nilaiSeek3>1)||(nilaiSeek4>1)){
41              proses.setVisibility(View.VISIBLE);
42              if (nilaiSeek1>1){
43                  filterharga.setVisibility(View.VISIBLE);
44                  switch1.setVisibility(View.VISIBLE);
45              }
46              if (nilaiSeek2>1){
47                  filterjarak.setVisibility(View.VISIBLE);
48                  switch2.setVisibility(View.VISIBLE);
49              }
50              if (nilaiSeek3>1){
51                  filterpopuler.setVisibility(View.VISIBLE);
52                  switch3.setVisibility(View.VISIBLE);
53              }
54              if (nilaiSeek4>1){
55                  filterrating.setVisibility(View.VISIBLE);
56                  switch4.setVisibility(View.VISIBLE);
57              }
58              proses.requestFocus();
59          }
60      }
61
62      public void myPosition(View view){
63          if (filterharga.length()>0){
64              filter[0]=Double.valueOf
65              (filterharga.getText().toString());
66          }
67
```

```
68          if (filterjarak.length()>0){
69              filter[1]=Double.valueOf
70              (filterjarak.getText().toString());
71          }
72          if (filterpopuler.length()>0){
73              filter[2]=Double.valueOf
74              (filterpopuler.getText().toString());
75          }
76          if (filterrating.length()>0){
77              filter[3]=Double.valueOf
78              (filterrating.getText().toString());
79          }
80
81          a12=nilaiSeek1/nilaiSeek4;
82          a13=nilaiSeek1/nilaiSeek3;
83          a14=nilaiSeek1/nilaiSeek2;
84          a23=nilaiSeek4/nilaiSeek3;
85          a24=nilaiSeek4/nilaiSeek2;
86          a34=nilaiSeek3/nilaiSeek2;
87
88          a21=nilaiSeek4/nilaiSeek1;
89          a31=nilaiSeek3/nilaiSeek1;
90          a41=nilaiSeek2/nilaiSeek1;
91          a32=nilaiSeek3/nilaiSeek4;
92          a42=nilaiSeek2/nilaiSeek4;
93          a43=nilaiSeek2/nilaiSeek3;
94
95          matriks_berpasangan =new double[][]
96                          {{1  ,a12,a13,a14},
97                          {a21,1   ,a23,a24},
98                          {a31,a32,1   ,a34},
99                          {a41,a42,a43,1   }};
100         getMyPosition();
101         sendDataAlternatifToAHPTOPSIS();
102     }
103
104     private void getMyPosition(){
105         LocationManager lm =  (LocationManager)this.
106         getSystemService(Context.LOCATION_SERVICE);
107
108         lm.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,
109         LOCATION_UPDATE_MIN_TIME, LOCATION_UPDATE_MIN_DISTANCE,
110         locationListener);
111         Location location =
112         lm.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
113         longitude = location.getLongitude();
114         latitude = location.getLatitude();
115
116         lm.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,
117         2000, 10, locationListener);
118     }
119
120     public void sendDataAlternatifToAHPTOPSIS(){
121         DatabaseReference ref =
122         FirebaseDatabase.getInstance().getReference();
123         Query query=ref.child("Data Alternatif");
124         query.addListenerForSingleValueEvent(new
125         ValueEventListener()
126
```

```
127             {
128                 @Override
129                 public void onDataChange(DataSnapshot dataSnapshot) {
130                     for (DataSnapshot data : dataSnapshot.getChildren())
131                     {
132
133                             if(data.getValue(TempatRental.class).
134                             getId()!=0){
135                             ArrayTempatRental.add(data.getValue
136                             (TempatRental.class));
137                         }
138                     }
139
140                     CobaHitung cobaHitung=new
141                     CobaHitung(activity,ArrayTempatRental,latitude,
142                     longitude,matriks_berpasangan,filter,filterTukar);
143                 }
144
145                 @Override
146                 public void onCancelled(DatabaseError error) {
147                     Log.w("TAG", "Failed to read value.",
148                     error.toException());
149                 }
150             })
151         }
152
153
154         @Override
155         protected void onCreate(Bundle savedInstanceState) {
156             super.onCreate(savedInstanceState);
157             setContentView
158             (R.layout.activity_input_nilai_perbandingan);
159         }
160 }
```

Kode 5.1 Kode program Activity InputNilaiPerbandingan

Penjelasan dari Kode 5.1 adalah sebagai berikut:

1. Pada Class Activity InputNilaiPerbandingan melakukan extends AppCompatActivity.
2. Baris 3-19 berfungsi untuk insialisasi *LocationLisener* yang merupakan *Listener* yang akan menyimpan variable *longitude* dan *latitude* setiap lokasi *user* berubah
3. Baris 21-60 adalah method *setFilter* yang berfungsi untuk menampilkan form filter tambahan berupa *EditText* sesuai dengan *SeekBar* yang diisi oleh *user*, jika tidak ada *SeekBar* yang diisi oleh *user*, maka akan langsung menjalankan *method myPosition.*
4. Baris 62-101 adalah *method myPosition* yang berfungsi untuk mengisi array filter berdasarkan *EditText* yang ditampilkan melalui *method SetFilter*. Lalu mengisi *double array matriks_berpasangan* dari nilai *SeekBar*. Lalu memanggil method *getMyPosition* dan *sendDataAlternatifToAHPTOPSIS.*
5. Baris 103-117 adalah *method getMyPosition* yang berfungsi untuk memanggil *listener LocationListener* untuk mengetahui lokasi *user* saat ini.

6. Baris 119-150 adalah *method SendDataAlternatifToAHPTOPSIS* yang berfungsi mengambil semua Data Alternatif pada *database firebase* dan mengirimnya pada *Class CobaHitung* bersamaan dengan variable *longitude, latitude* dari *listener LocationListener*, lalu *matriks_berpasangan, filter, dan filterTukar* dari *method myPosition*.
7. Baris 154-160 adalah *method onCreate* untuk mengatur *layout* untuk *class InputNilaiPerbandingan* untuk ditampilkan kepada pengguna.

5.4.2 Kode Program Class CobaHitung

Potongan kode program Kode 5.2 adalah potongan kode program untuk proses menghitung jarak *user* dengan lokasi masing-masing tempat rental pada data alternatif.

```
1   public class CobaHitung {
2
3       public CobaHitung(Activity activity, ArrayList
4       ArrayTempatRentalModel,
5       double latitude, double longitude, double [][]
6       matriks_berpasangan,
7       double[] filter,boolean[]filterTukar){
8           this.filter=filter;
9           this.filterTukar=filterTukar;
10          this.activity=activity;
11          this.matriks_berpasangan=matriks_berpasangan;
12          this.ArrayTempatRental=ArrayTempatRentalModel;
13          atributDataAlter=new double[ArrayTempatRental.size()][6];
14          for (int i = 0; i < ArrayTempatRental.size(); i++) {
15              atributDataAlter[i][0]=
16              ArrayTempatRental.get(i).getHarga();
17              atributDataAlter[i][1]=
18              ArrayTempatRental.get(i).getRating();
19              atributDataAlter[i][2]=
20              ArrayTempatRental.get(i).getPopularitas();
21              atributDataAlter[i][3]=
22              ArrayTempatRental.get(i).getLat();
23              atributDataAlter[i][4]=
24              ArrayTempatRental.get(i).getLon();
25              atributDataAlter[i][5]=
26              ArrayTempatRental.get(i).getId();
27          }
28          alternatif=ArrayTempatRental.size();
29          matriks_keputusan=new double[alternatif][kriteria];
30
31          for (int i = 0; i < alternatif; i++) {
32              for (int j = 0; j < kriteria; j++) {
33                  if (j==3){
34                      ArrayAsync.add(i,new HitungJarak());
35                      ArrayAsync.get(i).executeOnExecutor
36                      (AsyncTask.THREAD_POOL_EXECUTOR,
37                      (Double.toString(atributDataAlter[i][3])),
38                      (Double.toString(atributDataAlter[i][4])),
39                      Double.toString(latitude),
40                      Double.toString(longitude),
41                      Integer.toString(i));
42                  }else if(j==4){
```

```
                    matriks_keputusan[i][j]=atributDataAlter[i][5];
                }else{
                    matriks_keputusan[i][j]=atributDataAlter[i][j];
                }
            }
        }
    }

    private class HitungJarak extends AsyncTask<String, Void,
    Double> {
        private Exception exception;
        protected Double doInBackground(String... urls) {
            StringBuilder stringBuilder = new StringBuilder();
            double dist = 0.1;
            int distInt;
            try {
                String url
                ="https://maps.googleapis.com/maps/api/directions
                /json?origin="+urls[2]+","+urls[3]+"&destination=
                "+urls[0]+","+urls[1]+"&key=
                AIzaSyCIRamdXsRwe_47f6biRRD6XTXiqo4qhuo";
                HttpPost httppost = new HttpPost(url);
                HttpClient client = new DefaultHttpClient();
                HttpResponse response;
                stringBuilder = new StringBuilder();
                response = client.execute(httppost);
                HttpEntity entity = response.getEntity();
                InputStream stream = entity.getContent();
                int b;
                while ((b = stream.read()) != -1) {
                    stringBuilder.append((char) b);
                }
            } catch (ClientProtocolException e) {
            } catch (IOException e) {
            }

            JSONObject jsonObject = new JSONObject();
            try {
                jsonObject = new
                JSONObject(stringBuilder.toString());
                JSONArray array =
                jsonObject.getJSONArray("routes");
                JSONObject routes = array.getJSONObject(0);
                JSONArray legs = routes.getJSONArray("legs");
                JSONObject steps = legs.getJSONObject(0);
                JSONObject distance =
                steps.getJSONObject("distance");
                distInt=Integer.parseInt
                (distance.getString("value") );
                if (distInt<1000){
                    dist=distInt/1000;
                    dist=Double.parseDouble(new
                    DecimalFormat("##.##")
                    .format(dist));
                }else{
                    dist =
                    Double.parseDouble(distance.getString("text")
                    .replaceAll("[^\\.0123456789]","") );
                }
```

```
102                 matriks_keputusan[Integer.valueOf(urls[4])][3]=
103                 dist;
104             } catch (JSONException e) {
105                 e.printStackTrace();
106             }
107             return dist;
108         }
109 
110         protected void onPostExecute(Double result) {
111             counter++;
112             if (counter==alternatif){
113                 AHP_TOPSIS ahptopsis=new AHP_TOPSIS(activity,
114                 matriks_berpasangan,
115                 matriks_keputusan,filter,filterTukar);
116             }
117             super.onPostExecute(result);
118         }
119     }
120 }
```

Kode 5.2 Kode program Class CobaHitung

Penjelasan dari Kode 5.2 adalah sebagai berikut:

1. Baris 3-40 merupakan *constructor* dari *class CobaHitung* yang memiliki parameter *activity, ArrayTempatRentalModel, latitude, longitude, matriks_berpasangan, filter,* dan *filterTukar* yang akan dipakai sepanjang *class* ini. Baris 14-48 merupakan pengisian *array atributDataAlter* dari *ArrayTempatRentalModel,* dimana salah duanya merupakan *latitude* dan *longitude* dari masing-masing data alternatif. Pada baris 34-41 dilakukan perhitungan jarak antara *latitude* dan *longitude* dari *user* dengan *latitude* dan *longitude* pada masing-masing data alternatif melalui *AsyncTask HitungJarak*, masing-masing *AsyncTask* menghitung masing-masing jarak data alternatif yang akan dijalankan melalui *THREAD_POOL_EXECUTOR* agar bisa dijalankan secara parallel. Hasilnya akan disimpan pada variabel *matriks_keputusan.*
2. Baris 51-109 merupakan *private class HitungJarak* yang melakukan *extends AsyncTask* dengan *input String* dan *output Double.* Baris 54-108 merupakan *method doInBackground* yang akan dijalankan secara *asynchronous* dimana akan menjalankan url pada baris 60-63. Dimana hasilnya akan dirubah menjadi *String* pada baris 79-89. *String* ini akan diubah menjadi format *JSON* dan akan mengambil data dengan judul *distance* menjadi *variable Double* yang akan dimasukkan ke dalam array *matriks_keputusan* pada baris 90-103. Baris 110-118 adalah *method onPostExecute* adalah *method* yang akan dijalankan jika *method doInBackground* selesai dijalankan, di *method* ini akan dilakukan pengecekan apakah *AsyncTask* ini sudah dijalankan sebanyak jumlah data alternatif, jika sudah maka akan memanggil *class AHP_TOPSIS* sambil mengirimkan data *activity, matriks_berpasangan, matriks_keputusan, filter* dan *filterTukar.*

5.4.3 Kode Program Class AHP_TOPSIS

Potongan kode program Kode 5.3 adalah potongan kode program untuk proses menghitung *matriks_berpasangan* dengan *matriks_keputusan* menjadi hasil rekomendasi menggunakan metode AHP dan TOPSIS.

```
1   public class AHP_TOPSIS {
2       public AHP_TOPSIS(Activity
3       activity,double[][]matriks_berpasangan,
4       double[][]matriks_keputusan,double[]filter,
5       boolean[]filterTukar){
6           AHP ahp=new AHP(matriks_berpasangan);
7           mb_normalisasi =ahp.getNormalisasi();
8           bobot_prioritas=ahp.getBobot_prioritas();
9           eigen_maks=ahp.getEigenMaks();
10          nilai_ci=ahp.getNilaiCI();
11          nilai_cr=ahp.getNilaiCR();
12          TOPSIS topsis= new
13          TOPSIS(bobot_prioritas,matriks_keputusan);
14          matriks_keputusan=topsis.getMKeputusan();
15          mk_normalisasi=topsis.getMKNormalisasi();
16          mk_terbobot=topsis.getMKTerbobot();
17          jarak_si_positif=topsis.getJarakSIPositif();
18          si_positif=topsis.getSIPositif();
19          si_negatif= topsis.getSINegatif();
20          jarak_si_negatif=topsis.getJarakSINegatif();
21          nilai_preferensi=topsis.getNilaiPreferensi();
22          urutan=topsis.getUrutan();
23          urutanJarak=topsis.getUrutanJarak();
24          Intent intent = new Intent(activity,
25          ListHasilRekomen.class);
26          intent.putExtra("urutan",urutan);
27          intent.putExtra("urutan jarak",urutanJarak);
28          intent.putExtra("filter",filter);
29          intent.putExtra("filter tukar",filterTukar);
30          activity.startActivity(intent);
31      }
32  }
```

Kode 5.3 Kode program Class AHP_TOPSIS

Penjelasan dari Kode 5.3 adalah sebagai berikut:

1. Baris 2-5 merupakan penamaan *constructor* dari *class AHP_TOPSIS* yang memiliki parameter *activity, matriks_berpasangan, matriks_keputusan, filter, filterTukar.*
2. Baris 6-11 masih bagian dari *constructor class AHP_TOPSIS* yang berguna untuk menghitung bobot prioritas berdasarkan *matriks_bepasangan* dengan cara memanggil *method* pada *class AHP. Method* yang dipanggil adalah *method getNormalisasi, getBobot_prioritas, getNilaiCI, getNilaiCR.* bobot prioritas disimpan dalam *variabel bobot_prioritas* menggunakan *method getBobot_prioritas* pada baris 8*.*
3. Baris 12-23 juga masih bagian dari *constructor* dari *class AHP_TOPSIS* yang berguna untuk menghitung nilai preferensi dan urutan data alternatif berdasarkan nilai preferensi tersebut berdasarkan *bobot_prioritas* dari *class*

AHP dan *matriks_keputusan.* Setelah itu diapanggil *method* dari *class AHP* untuk menghitung data tersebut mulai dari *method getMKeputusan* sampai *getNilaiPrefernsi* pada baris 14-21. Pada baris 22-23 didapatkan urutan data alternatif yang akan disimpan pada *variabel urutan* untuk *id* data alternatif dan *variabel urutanJarak* untuk jarak data alternatif.

4. Baris 24-30 berguna untuk membuka *activity ListHasilRekomen* sambil mengirim data *urutan, urutanJarak, filter* dan *filterTukar.*

5.4.4 Kode Program Class AHP

Potongan kode program Kode 5.4 adalah potongan kode program untuk proses menghitung *matriks_berpasangan* menjadi *bobot_prioritas* menggunakan metode AHP.

```
1   public class AHP {
2
3       public AHP(double[][] matriks_berpasangan){
4           this.matriks_berpasangan=matriks_berpasangan;
5       }
6
7       double[][] getNormalisasi(){
8           for (int i = 0; i < matriks_berpasangan[0].length; i++) {
9               total_perkolom =0;
10              for (int j = 0; j < matriks_berpasangan.length; j++) {
11                  total_perkolom += matriks_berpasangan[j][i];
12              }
13
14              for (int k = 0; k < matriks_berpasangan.length; k++) {
15                  mb_normalisasi[k][i]= matriks_berpasangan[k][i]/
16                  total_perkolom;
17              }
18          }
19          return mb_normalisasi;
20      }
21
22      double[] getBobot_prioritas(){
23          for (int i = 0; i < bobot_prioritas.length; i++) {
24              for (int j = 0; j < bobot_prioritas.length; j++) {
25                  bobot_prioritas[i]+= mb_normalisasi[i][j];
26              }
27              bobot_prioritas[i]/= kriteria;
28          }
29          return bobot_prioritas;
30      }
31
32      double getEigenMaks(){
33          double nilai_kepentingan=0;
34          for (int i = 0; i < kriteria; i++) {
35              for (int j = 0; j < kriteria; j++) {
36                  nilai_kepentingan+=matriks_berpasangan[j][i];
37              }
38              nilai_kepentingan*=bobot_prioritas[i];
39              eigen_maks+=nilai_kepentingan;
40              nilai_kepentingan=0;
41          }
42          return eigen_maks;
```

```
43      }
44
45      double getNilaiCI(){
46
47          nilai_ci=(eigen_maks-kriteria)/(kriteria-1);
48          return nilai_ci;
49      }
50
51      double getNilaiCR(){
52
53          nilai_cr=nilai_ci/random_index;
54          return nilai_cr;
55      }
56
57  }
```

Kode 5.4 Kode program Class AHP

Penjelasan dari Kode 5.4 adalah sebagai berikut:

5. Baris 3-5 merupakan penamaan *constructor* dari *class AHP* yang memiliki parameter *matriks_berpasangan.*
6. Baris 7-20 merupakan *method getNormalisasi* untuk menghitung *mb_normalisasi* dari *matriks_berpasangan.*
7. Baris 22-30 merupakan *method getBobot_prioritas* dimana dalam *method* ini akan menghitung bobot prioritas pada *variable bobot_prioritas* dari matriks berpasangan ternormalisasi pada *variable mb_normalisasi*
8. Baris 32-43 merupakan *method getEigenMaks* dimana dalam *method* ini akan menghitung eigen maks pada *variable eigen_maks* dari *matriks_berpasangan* dan *bobot_prioritas.*
9. Baris 45-49 merupakan method *getNilaiCI* untuk menghitung nilai CI pada *variable nilai_ci* dari *eigen_maks* dengan jumlah kriteria.
10. Baris 51-55 merupakan *method getNilaiCR* untuk menghitung nilai CR pada *variable nilai_cr* dari *nilai_cr* dengan nilai random index.

5.4.5 Kode Program Class TOPSIS

Potongan kode program Kode 5.5 adalah potongan kode program untuk proses menghitung *bobot_prioritas* dan *matriks_keputusan* menjadi *urutan* dan *urutanJarak* menggunakan metode TOPSIS.

```
1   public class TOPSIS
2       public TOPSIS(double[] bobot_prioritas,double[][]
3       matriks_keputusan){
4           this.bobot_prioritas=bobot_prioritas;
5           this.matriks_keputusan=matriks_keputusan;
6           alternatif=matriks_keputusan.length;    }
7
8       public double[][] getMKNormalisasi(){
9           for (int i = 0; i < kriteria; i++) {
10              total_perkolom =0;
11              for (int j = 0; j < matriks_keputusan.length; j++) {
```

```
12                  total_perkolom += matriks_keputusan[j][i];
13              }
14
15              for (int k = 0; k < matriks_keputusan.length; k++) {
16                  mk_normalisasi[k][i]= matriks_keputusan[k][i]/
17                  total_perkolom;
18              }
19          }
20          return mk_normalisasi;
21      }
22
23      public double[][] getMKTerbobot(){
24
25          for (int i = 0; i < alternatif; i++) {
26              for (int j = 0; j < kriteria; j++) {
27                  mk_terbobot[i][j]=mk_normalisasi[i][j]
28                  *bobot_prioritas[j];
29              }
30          }
31          for (int i = 0; i < alternatif; i++) {
32              for (int j = 0; j < kriteria; j++) {
33                  //solusi ideal positif n negatif
34                  if ((j==1) || (j==2)){//keuntungan
35                      if (mk_terbobot[i][j]>si_positif[j]){
36                          si_positif[j]=mk_terbobot[i][j];
37                      }
38                      if (mk_terbobot[i][j]<si_negatif[j]){
39                          si_negatif[j]=mk_terbobot[i][j];
40                      }
41                  }else if ((j==0) || (j==3)){//biaya
42                      if (mk_terbobot[i][j]>si_negatif[j]){
43                          si_negatif[j]=mk_terbobot[i][j];
44                      }
45                      if (mk_terbobot[i][j]<si_positif[j]){
46                          si_positif[j]=mk_terbobot[i][j];
47                      }
48                  }
49              }
50          }
51          return mk_terbobot;
52      }
53
54      public double[] getSIPositif(){
55          return si_positif;
56      }
57
58      public double[] getSINegatif(){
59          return si_negatif;
60      }
61
62      public double[] getJarakSIPositif(){
63          double temp=0;
64          for (int i = 0; i < alternatif; i++) {
65              for (int j = 0; j < kriteria; j++) {
66                  temp+=Math.pow((mk_terbobot[i][j]-
67                  si_positif[j]),2);
68              }
69              jarak_si_positif[i]=Math.sqrt(temp);
70              temp=0;
```

```
71          }
72          return jarak_si_positif;
73      }
74
75      public double[] getJarakSINegatif(){
76          double temp=0;
77          for (int i = 0; i < alternatif; i++) {
78              for (int j = 0; j < kriteria; j++) {
79                  temp+=Math.pow((mk_terbobot[i][j]-
80                  si_negatif[j]),2);
81              }
82              jarak_si_negatif[i]=Math.sqrt(temp);
83              temp=0;
84          }
85          return jarak_si_negatif;
86      }
87
88      public double[] getNilaiPreferensi(){
89          for (int i = 0; i < alternatif; i++) {
90              nilai_preferensi[i]=(jarak_si_negatif[i]/
91              (jarak_si_negatif[i]+jarak_si_positif[i]));
92          }
93          return nilai_preferensi;
94      }
95
96      public int[] getUrutan(){;
97          nilai_preferensi_terurut=Arrays.copyOf(nilai_preferensi,
98          nilai_preferensi.length);
99          Arrays.sort(nilai_preferensi_terurut);
100
101         for (int i = 0; i < alternatif; i++) {
102             nilai_preferensi_top3[i]=nilai_preferensi_terurut[
103             (alternatif-1)-i];
104         }
105
106         for (int i = 0; i < nilai_preferensi.length; i++) {
107             for (int j = 0; j < alternatif; j++) {
108                 if (nilai_preferensi_top[j]==nilai_preferensi[i]){
109                     urutan[j]=(int)matriks_keputusan[i][4];
110                     urutanJarak[j]=(Double)matriks_keputusan[i][3];
111                 }
112             }
113         }
114
115
116         return urutan;
117     }
118
119     public double[] getUrutanJarak(){
120         return urutanJarak;
121     }
122 }
```

Kode 5.5 Kode program Class TOPSIS

Penjelasan dari Kode 5.5 adalah sebagai berikut:

1. Baris 2-6 merupakan penamaan *constructor* dari *class TOPSIS* yang memiliki parameter *bobot_prioritas* dan *matriks_keputusan.* Disini dilakukan pengisian *variable alternatif* dari Panjang *array* dari *matriks_keputusan.*
2. Baris 8-21 merupakan *method getMKNormalisasi* untuk menghitung matriks keputusan ternormalisasi pada *variable mk_normalisasi* dari *matriks_keputusan.*
3. Baris 23-50 merupakan *method getMKTerbobot.* Pada baris 25-30 untuk menghitung matriks keputusan terbobot pada *variable mk_terbobot.* Baris 31-48 untuk menghitung solusi ideal positif dan solusi ideal negatif pada *variable si_positif* dan *si_negatif.*
4. Baris 54-60 merupakan *method getter getSIPositif* dan *getSINegatif,* untuk mengembalikan *variable si_positif* dan *si_negatif.*
5. Baris 62-73 merupakan *method getJarakSIPositif* untuk menghitung jarak solusi ideal positif pada *variable jarak_si_positif* dari *si_positif.*
6. Baris 88-94 merupakan *method getNilaiPreferensi* untuk menghitung nilai preferensi masing-masing data alternatif pada *variable nilai_preferensi* dari *jarak_si_positif* dan *jarak_si_negatif.*
7. Baris 96-117 adalah *method getUrutan* untuk menghitung urutan data alternatif berdasarkan *nilai_preferensi* masing-masing data alternatif pada *variable urutan*. Pada *method* ini juga berfungsi untuk menghitung urutan jarak berdasarkan *urutan* pada *variable urutanJarak.*
8. Baris 119-121 adalah *method getter getUrutanJarak* untuk memanggil *variable urutanJarak.*

5.4.6 Kode Program Class ListHasilRekomen

Potongan kode program Kode 5.6 adalah potongan kode program untuk menampilkan hasil rekomendasi menggunakan *Class Adapter RecyclerView HasilRekomenViewHolder*.

```
1   public class ListHasilRekomen extends AppCompatActivity {
2
3       @Override
4       public void onCreate(@Nullable Bundle savedInstanceState) {
5           super.onCreate(savedInstanceState);
6           setContentView(R.layout.reycler_hasil_rekomen);
7
8           Intent intent=getIntent();
9           urutan=intent.getIntArrayExtra("urutan");
10          urutanJarak=intent.getDoubleArrayExtra("urutan jarak");
11          filter=intent.getDoubleArrayExtra("filter");
12          filterTukar=intent.getBooleanArrayExtra("filter tukar");
13          recycler=(RecyclerView)findViewById
14          (R.id.recyclerHasilRekomen);
15          recycler.setLayoutManager(new LinearLayoutManager(this));
16          RekomenAdapter =new HasilRekomenViewHolder(this);
17          recycler.setAdapter(RekomenAdapter);
18
```

```
19          DatabaseReference ref =
20          FirebaseDatabase.getInstance().getReference();
21          ref.addValueEventListener(new ValueEventListener() {
22              @Override
23              public void onDataChange(DataSnapshot dataSnapshot) {
24                  ArrayTempatRental.clear();
25                  for (DataSnapshot data : dataSnapshot.child("Data
26                       Alternatif").getChildren()) {
27
28                       ArrayTempatRental.add(data.getValue
29                      (TempatRental.class));
30                  }
31              RekomenAdapter.setDataBinding(ArrayTempatRental,urutan,
32              urutanJarak,filter, filterTukar);
33              }
34
35              @Override
36              public void onCancelled(DatabaseError error) {
37                  // Failed to read value
38                  Log.w("TAG", "Failed to read value.",
39                  error.toException());
40              }
41          });
42      }
43  }
```

Kode 5.6 Kode program Class ListHasilRekomen

Penjelasan dari Kode 5.6 adalah sebagai berikut:

1. Pada Class Activity ListHasilRekomen melakukan extends AppCompatActivity.
2. Baris 4-6 merupakan insialisasi *method onCreate* dan mengatur *layout* untuk tampilan yang bisa dilihat oleh pengguna
3. Baris 8-12 merupakan pengambilan *variable Extra intent* dari *class AHP_TOPSIS* berupa *urutan, urutanJarak, filter,* dan filterTukar.
4. Baris 13-17 berfungsi untuk mengisi *adapter* pada *RecyclerView recycler* dengan *class adapter HasilRekomenViewHolder*
5. Baris 19-41 berfungsi untuk mengambil data alternatif pada *FirebaseDatabase* dan memakai data alternatif tersebut bersama dengan *variable urutan, urutanJarak, filter dan filterTukar* sebagai *data binding* pada *adapter.*

5.4.7 Kode Program Class HasilRekomenViewHolder

Potongan kode program Kode 5.7 adalah potongan kode program untuk menampilkan hasil rekomendasi menggunakan *Class Adapter RecyclerView HasilRekomenViewHolder*.

```
1   public class HasilRekomenViewHolder extends
2   RecyclerView.Adapter<RecyclerView.ViewHolder> {
3
4       public HasilRekomenViewHolder(Context context){
5           this.context=context;
6       }
7
```

```
8       public void setDataBinding(ArrayList<TempatRental> items,int[]
9       urutan,double[] urutanJarak,double[]filter,
10      boolean[]filterTukar){
11          this.items=items;
12          this.urutan=urutan;
13          this.urutanJarak=urutanJarak;
14          this.urutanJarakFix=new double[items.size()];
15          terurut.clear();
16          if(urutan[2]==0){
17          }else{
18              if (filterTukar[0]){
19                  max1=filter[0];
20                  min1=Integer.MIN_VALUE;
21              }
22              if (!filterTukar[0]){
23                  max1=Integer.MAX_VALUE;
24                  min1=filter[0];
25              }
26              if (filterTukar[1]){
27                  max2=filter[1];
28                  min2=Integer.MIN_VALUE;
29              }
30              if (!filterTukar[1]){
31                  max2=Integer.MAX_VALUE;
32                  min2=filter[1];
33              }
34              if (filterTukar[2]){
35                  max3=filter[2];
36                  min3=Integer.MIN_VALUE;
37              }
38              if (!filterTukar[2]) {
39                  max3=Integer.MAX_VALUE;
40                  min3=filter[2];
41              }
42              if (filterTukar[3]){
43                  max4=filter[3];
44                  min4=Integer.MIN_VALUE;
45              }
46              if (!filterTukar[3]){
47                  max4=Integer.MAX_VALUE;
48                  min4=filter[3];
49              }
50              int index=0;
51              for (int i = 0; i < urutan.length; i++) {
52                  for (int j = 0; j < items.size(); j++) {
53                      if (urutan[i] == items.get(j).getId()) {
54                          int harga = items.get(j).getHarga();
55                          int populer =
56                          items.get(j).getPopularitas();
57                          double rating = items.get(j).getRating();
58                          double jarak = urutanJarak[i];
59                          if (((min1 <= harga) && (harga <= max1)) &&
60                          ((min2 <= jarak) && (jarak <= max2)) &&
61                          ((min3 <= populer) && (populer <= max3)) &&
62                          ((min4 <= rating) && (rating <= max4))) {
63                              urutanJarakFix[index]=urutanJarak[i];
64                              terurut.add(index,items.get(j));
65                              index++;
66                          }
```

```
                    }
                }
            }
        }
        notifyDataSetChanged();
    }

    @Override
    public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup
    parent, int
        viewType) {
        LayoutInflater inflater=LayoutInflater.from(context);
        row=inflater.inflate(R.layout.custome_row,parent,false);
        Item item=new Item(row);
        return item;
    }

    @Override
    public void onBindViewHolder(RecyclerView.ViewHolder holder,
    final int position) {

        ((Item)holder).name.setText(terurut.get
        (position).getNama());
        ((Item)holder).telp.setText("Rating
         ="+terurut.get(position).getRating()+".Popularitas
         ="+terurut.get(position).getPopularitas()+".Jarak
         ="+urutanJarakFix[position]);
        ((Item)holder).harga.setText("Rp.
         "+terurut.get(position).getHarga());
        Picasso.with(context)
                .load(terurut.get(position).getImageUrl())
                .fit()
                .centerCrop()
                .into(((Item)holder).imgView);
        ((Item)holder).linearMain.setOnClickListener(new
         View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Intent intent = new Intent(context,
                    DetailTempatRentalActivity.class);
                    intent.putExtra("Tempat",
                    Parcels.wrap(terurut.get(position)));
                    context.startActivity(intent);
                }
            });

    }

    public class Item extends RecyclerView.ViewHolder{
        TextView name, telp, harga;
        LinearLayout linearMain;
        ImageView imgView;

        public Item(View itemView){
            super(itemView);
            name=(TextView)itemView.findViewById(R.id.itemName);
            telp=(TextView)itemView.findViewById(R.id.itemTelp);
            harga=(TextView)itemView.findViewById(R.id.itemHarga);
            imgView =
```

```
126                 (ImageView)itemView.findViewById(R.id.imgView);
127                 linearMain=(LinearLayout)
128                 itemView.findViewById(R.id.linearMain);
129             }
130         }
131 }
132
133
134
135
136
```

Kode 5.7 Kode program Class HasilRekomenViewHolder

Penjelasan dari Kode 5.7 adalah sebagai berikut:

1. Pada Class HasilRekomenViewHolder melakukan extends RecyclerView.Adapter<RecyclerView.ViewHolder>.
2. Baris 4-5 merupakan *constructor* dari Class HasilRekomenViewHolder dengan parameter *context.*
3. Baris 8-72 merupakan *method setDataBinding* untuk melakukan proses pembuatan *variable terurut* dan *urutanJarakFix* yang akan ditampilkan pada *RecyclerView*.
4. Baris 74-82 merupakan *method onCreateViewHolder* untuk memilih *layout resource* pada masing-masing *item* pada *RecyclerView* dimana masing-masing komponennya akan diinisialisasi pada *Class Item* pada baris 115-130.
5. Baris 84-113 merupakan *method onBindViewHolder* untuk mengisi komponen pada *layout item* dengan berisi *attribute* dari *variable terurut* dan *urutanJarakFix* berdasarkan *index*-nya dan posisi dari *item* tersebut.

5.5 Implementasi antarmuka

Implementasi antarmuka ini merupakan tahap pembuatan antarmuka yang diimplementasikan berdasarkan perancangan antarmuka yang telah dilakukan.

Implementasi antarmuka akan disusun berdasarkan susunan *screen flow* pada perancangan.

5.5.1 Implementasi *Screen Flow* Mendapatkan Hasil Rekomendasi Tempat Rental Motor

Screen *Flow* implementasi ini dapat dilihat pada Gambar 5.2 dimana *screen flow* ini mempunyai penjelas yang sama dengan perancangannya.

Gambar 5.2 Implementasi *Screen Flow* Mendapatkan Hasil Rekomendasi Tempat Rental Motor

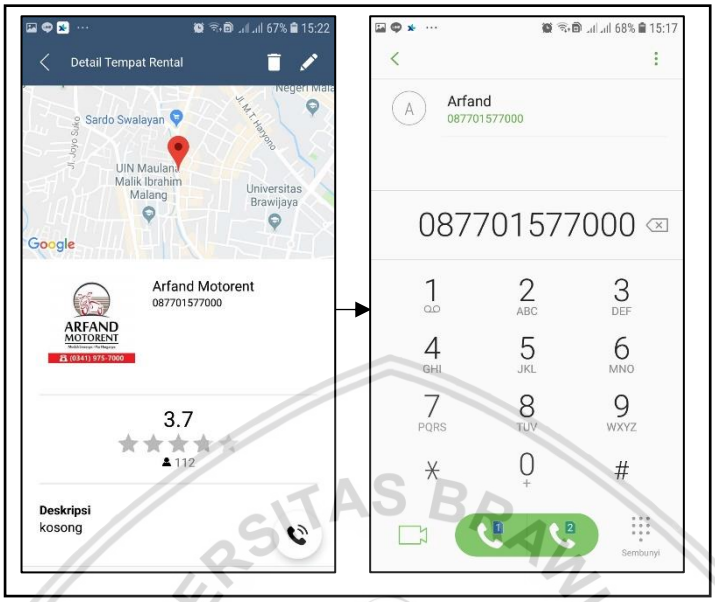
5.5.2 Implementasi *Screen Flow* Melihat Informasi Tempat Rental Motor

Screen *Flow* implementasi ini dapat dilihat pada Gambar 5.3 dimana *screen flow* ini mempunyai penjelas yang sama dengan perancangannya.

Gambar 5.3 Implementasi *Screen Flow* Melihat Informasi Tempat Rental Motor

5.5.3 Implementasi *Screen Flow* Menelepon Tempat Rental Motor

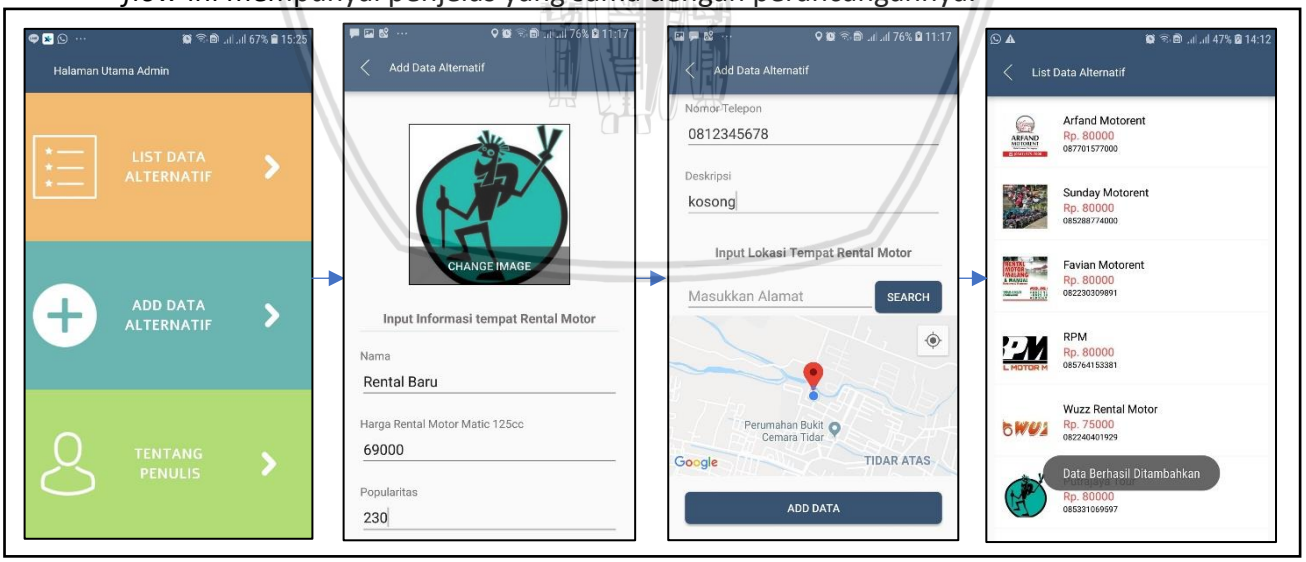
Screen *Flow* implementasi ini dapat dilihat pada Gambar 5.4 dimana *screen flow* ini mempunyai penjelas yang sama dengan perancangannya.



Gambar 5.4 Implementasi *Screen Flow* Menelepon Tempat Rental Motor

5.5.4 Implementasi *Screen Flow* Menambah Data Alternatif

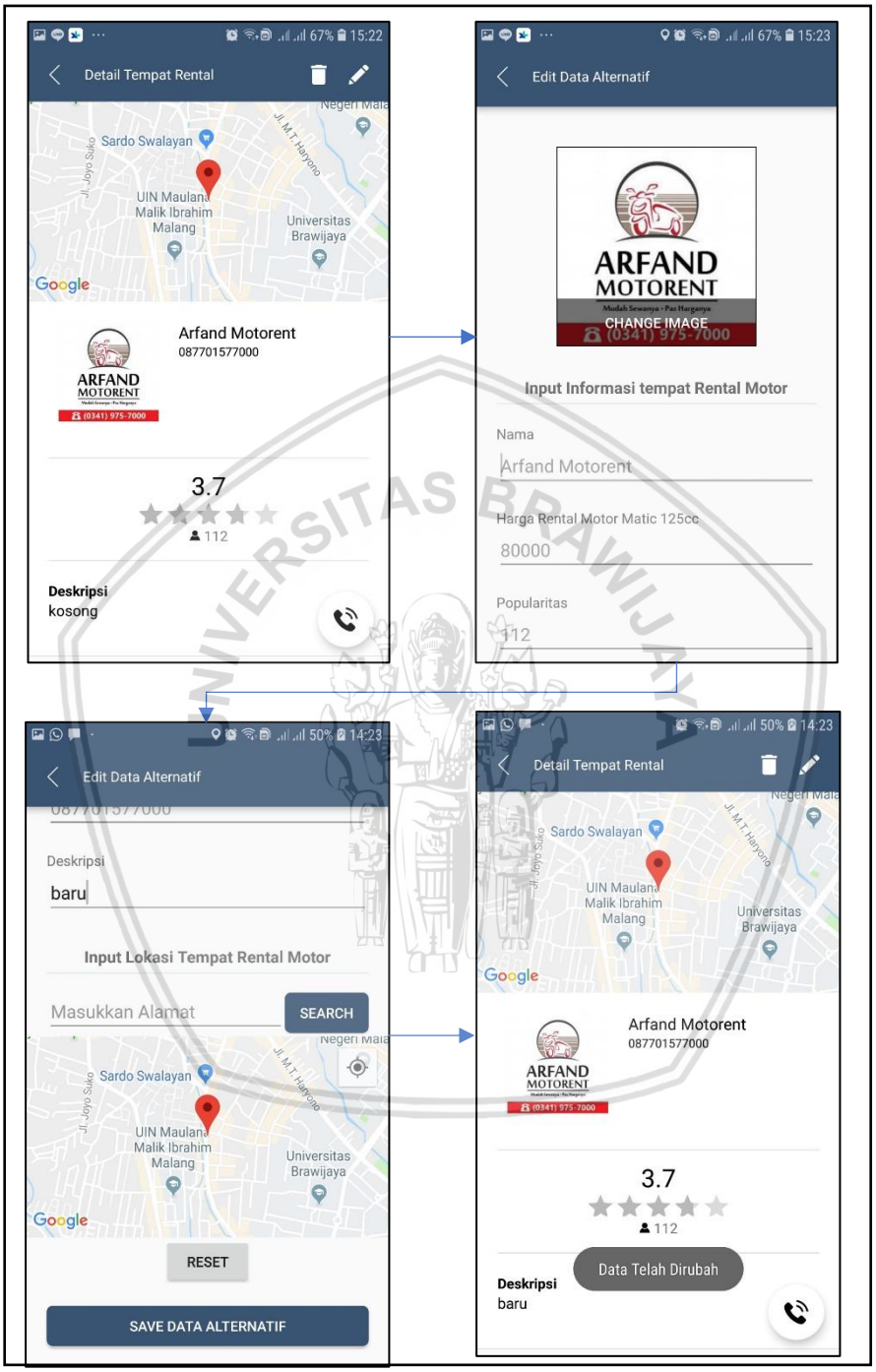
Screen *Flow* implementasi ini dapat dilihat pada Gambar 5.5 dimana *screen flow* ini mempunyai penjelas yang sama dengan perancangannya.



Gambar 5.5 Implementasi *Screen Flow* Menelepon Tempat Rental Motor

5.5.5 Implementasi *Screen Flow* Mengubah Data Alternatif

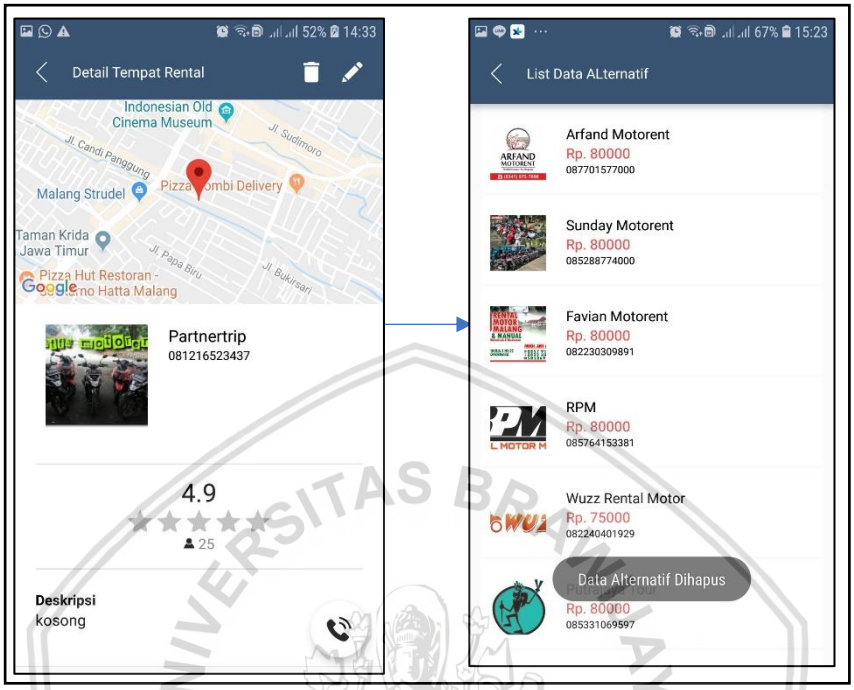
Screen *Flow* implementasi ini dapat dilihat pada Gambar 5.6 dimana *screen flow* ini mempunyai penjelas yang sama dengan perancangannya.



Gambar 5.6 Implementasi *Screen Flow* Mengubah Data Alternatif

5.5.6 Implementasi *Screen Flow* Menghapus Data Alternatif

Screen *Flow* implementasi ini dapat dilihat pada Gambar 5.7 dimana *screen flow* ini mempunyai penjelas yang sama dengan perancangannya.



Gambar 5.7 Implementasi *Screen Flow* Menghapus Data Alternatif

BAB 6 PENGUJIAN

Bab pengujian ini akan dilakukan pengujian terhadap aplikasi rekomendasi tempat rental motor di kota malang dengan metode ahp topsis berbasis *location based services*. Hasil dari pengujian tersebut akan dianalisa. Pengujian ini akan dilakukan dalam tiga metode, yaitu pengujian fungsional, pengujian kesesuaian, dan pengujian *usability.*

6.1 Pengujian

Pengujian dilakukan untuk mencari kesalahan yang mungkin dapat muncul dalam penggunaan aplikasi sebelum aplikasi dapat dinikmati oleh pengguna. Pengujian juga dilakukan untuk mencari tahu apakah aplikasi dapat bekerja sesuai dengan rekayasa kebutuhan dan perancangan. Pengujian yang dilakukan berupa pengujian fungsional, pengujian kesesuaian, pengujian *usability*, dan pengujian algoritma.

6.1.1 Pengujian fungsional

Pengujian fungsional ini menggunakan pengujian *blackbox* untuk menguji keluaran sistem apakah sudah sesuai dengan rekayasa kebutuhan dan perancangan sistem berdasarkan masukan dari pengguna. Masing-masing kasus uji yang dibuat memiliki kode kasus uji, nama kasus uji, kode fungsional, tujuan pengujian, prosedur pengujian dan hasil yang diharapkan. Kasus uji fungsional dapat dilihat pada Tabel 6.1 hingga Tabel 6.8.

Tabel 6.1 Kasus uji fungsional mendapatkan hasil rekomendasi tempat rental motor

KODE KASUS UJI	**PF_REKMON_F_01**
NAMA KASUS UJI	Mendapatkan hasil rekomendasi tempat rental motor
KODE FUNGSIONAL	REKMON_F_01
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk mendapatkan hasil rekomendasi tempat rental
PROSEDUR PENGUJIAN	1. Mengisi bobot kepentingan kriteria 2. Menekan tombol “Proses” 3. Mengisi form filter jika muncul dan menekan tombol “Cari Tempat Rental”
HASIL YANG DIHARAPKAN	Sistem dapat menampilkan hasil rekomendasi tempat rental motor di halaman List Hasil Rekomendasi

Tabel 6.2 Kasus uji fungsional melihat informasi tempat rental motor aktor *user*

KODE KASUS UJI	**PF_REKMON_F_02**
NAMA KASUS UJI	Melihat informasi tempat rental motor aktor *user*
KODE FUNGSIONAL	REKMON_F_02
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk melihat informasi dari tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi
PROSEDUR PENGUJIAN	1. Memilih menu "List Tempat Rental" pada Halaman Utama *User* 2. Memilih salah satu item dari list
HASIL YANG DIHARAPKAN	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor

Tabel 6.3 Kasus uji fungsional melihat informasi tempat rental motor aktor *admin*

KODE KASUS UJI	**PF_REKMON_F_03**
NAMA KASUS UJI	Melihat informasi tempat rental motor aktor *admin*
KODE FUNGSIONAL	REKMON_F_02
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk melihat informasi dari tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi
PROSEDUR PENGUJIAN	1. Memilih menu "List Data Alternatif" pada Halaman Utama *Admin* 2. Memilih salah satu item dari list
HASIL YANG DIHARAPKAN	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor

Tabel 6.4 Kasus uji fungsional melihat informasi tempat rental motor aktor *user* alternative

KODE KASUS UJI	**PF_REKMON_F_04**
NAMA KASUS UJI	Melihat informasi tempat rental motor aktor *user* alternative
KODE FUNGSIONAL	REKMON_F_02
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk melihat informasi dari tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi
PROSEDUR PENGUJIAN	1. Selesai melakukan fungsionalitas REKMON_F_01 2. Memilih salah satu item dari list
HASIL YANG DIHARAPKAN	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor

Tabel 6.5 Kasus uji fungsional menelepon tempat rental motor

KODE KASUS UJI	**PF_REKMON_F_05**
NAMA KASUS UJI	Menelepon tempat rental motor
KODE FUNGSIONAL	REKMON_F_03
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk menelepon tempat rental motor dengan nomor telepon masing-masing tempat rental motor
PROSEDUR PENGUJIAN	1. Menekan tombol yang memiliki gambar telepon pada halaman Detail Tempat Rental
HASIL YANG DIHARAPKAN	Sistem dapat menampilkan tampilan aplikasi telepon bawaan *smartphone* dengan nomor yang sudah terisi oleh nomor telepon tempat rental motor yang terpilih

Tabel 6.6 Kasus uji fungsional menambah data alternatif

KODE KASUS UJI	**PF_REKMON_F_06**
NAMA KASUS UJI	Menambah data alternatif
KODE FUNGSIONAL	REKMON_F_04
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk menambah data alternatif
PROSEDUR PENGUJIAN	1. Memilih menu "Add Data Alternatif" pada Halaman Utama *Admin* Memilih salah satu item dari list 2. Mengisi form secara lengkap 3. Menekan tombol "Add Data"
HASIL YANG DIHARAPKAN	Sistem dapat menyimpan data alteratif baru pada basis data dan menampilkan halaman List Data Alternatif dengan data alternatif baru berada pada list tersebut

Tabel 6.7 Kasus uji fungsional mengubah data alternatif

KODE KASUS UJI	**PF_REKMON_F_07**
NAMA KASUS UJI	Mengubah data alternatif
KODE FUNGSIONAL	REKMON_F_05
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk mengubah data alternatif
PROSEDUR PENGUJIAN	1. Menekan tombol yang memiliki gambar pensil pada halaman Detail Tempat Rental 2. Mengisi form yang ingin dirubah datanya 3. Menekan tombol "Save Data Alternatif"
HASIL YANG DIHARAPKAN	Sistem dapat menyimpan perubahan data alteratif pada basis data dan menampilkan halaman Detail Tempat Rental yang berisi data terbaru

Tabel 6.8 Kasus uji fungsional menghapus data alternatif

KODE KASUS UJI	**PF_REKMON_F_08**
NAMA KASUS UJI	Menghapus data alternatif
KODE FUNGSIONAL	REKMON_F_06
TUJUAN PENGUJIAN	Untuk membuktikan bahwa sistem memiliki fungsional yang dapat digunakan untuk menghapus data alternatif
PROSEDUR PENGUJIAN	1. Menekan tombol yang memiliki gambar tempat sampah pada halaman Detail Tempat Rental
HASIL YANG DIHARAPKAN	Sistem dapat menghapus data alteratif yang dipilih pada basis data dan menampilkan halaman List Data Alternatif yang sudah berkurang data alternatif

Setelah dilakukan pendefinisian kasus uji, selanjutnya diakukan penentuan hasil pengujian berupa status pengujian berdasarkan hasil yang diharapkan dibandingkan dengan hasil yang diperoleh. Selengkapnya dapat dilihat pada Tabel 6.9.

Tabel 6.9 Hasil pengujian fungsional

Kode Kasus Uji	Nama Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Diperoleh	Status
PF_REKMON_F_01	Mendapatkan hasil rekomendasi tempat rental motor	Sistem dapat menampilkan hasil rekomendasi tempat rental motor di halaman List Hasil Rekomendasi	Sistem dapat menampilkan hasil rekomendasi tempat rental motor di halaman List Hasil Rekomendasi	Valid
PF_REKMON_F_02	Melihat informasi tempat rental motor aktor *user*	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor	Valid

		telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor	telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor	
PF_REKMON_F_03	Melihat informasi tempat rental motor aktor *admin*	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor	Valid
PF_REKMON_F_04	Melihat informasi tempat rental motor aktor *user* alternative	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor	Sistem dapat menampilkan tampilan detail tempat rental motor yang terdiri dari nama, deskripsi, nomor telepon, *rating*, popularitas, foto profil dan lokasi tempat rental motor	Valid
PF_REKMON_F_05	Menelepon tempat rental motor	Sistem dapat menampilkan tampilan aplikasi telepon bawaan *smartphone* dengan nomor	Sistem dapat menampilkan tampilan aplikasi telepon bawaan *smartphone* dengan nomor	Valid

		yang sudah terisi oleh nomor telepon tempat rental motor yang terpilih	yang sudah terisi oleh nomor telepon tempat rental motor yang terpilih	
PF_REKMON_F_06	Menambah data alternatif	Sistem dapat menyimpan data alteratif baru pada basis data dan menampilkan halaman List Data Alternatif dengan data alternatif baru berada pada list tersebut	Sistem dapat menyimpan data alteratif baru pada basis data dan menampilkan halaman List Data Alternatif dengan data alternatif baru berada pada list tersebut	Valid
PF_REKMON_F_07	Mengubah data alternatif	Sistem dapat menyimpan perubahan data alteratif pada basis data dan menampilkan halaman Detail Tempat Rental yang berisi data terbaru	Sistem dapat menyimpan perubahan data alteratif pada basis data dan menampilkan halaman Detail Tempat Rental yang berisi data terbaru	Valid
PF_REKMON_F_08	Menghapus data alternatif	Sistem dapat menghapus data alteratif yang dipilih pada basis data dan menampilkan halaman List Data Alternatif yang sudah berkurang data alternatif	Sistem dapat menghapus data alteratif yang dipilih pada basis data dan menampilkan halaman List Data Alternatif yang sudah berkurang data alternatif	Valid

6.1.2 Pengujian kesesuaian

Pengujian kesesuaian ini dilakukan untuk mengetahui bagaimana tingkat kesesuaian sistem rekomendasi aplikasi ini dengan cara membandingkan keluaran aplikasi dengan pilihan pengguna. Pengujian dilakukan dengan cara meminta pengguna memilih tempat rental yang sesuai dengan bobot kriteria yang diinginkan. Tempat rental motor yang digunakan untuk pengujian yang akan dipilih oleh responden adalah 10 tempat rental motor yang sama seperti tertulis pada data alternatif pada matriks keputusan pada Tabel 4.17. Setelah itu dilakukan perbandingan dengan pilihan tempat rental motor hasil rekomendasi. Data yang digunakan berjumlah 20 data dari 20 pengguna. Pengguna ini adalah mahasiswa dan pelajar SMA yang berdomisili di Kota Malang. Terdapat dua pengujian kesesuaian, yaitu pengujian setiap data sampel dan pengujian total. Hasil dari kedua pengujian tersebut dapat dilihat pada Tabel 6.10.

Tabel 6.10 Hasil rekapitulasi kuisoner pengujian *usability*

Pengguna	Pilihan Pengguna			Keluaran Aplikasi			Nilai Pengujian sampel	Nilai Pengujian Total
	Pilihan 1	Pilihan 2	Pilihan 3	Rangking 1	Rangking 2	Rangking 3		
Pengguna-01	a2	a8	a5	a2	a1	a8	2	Sesuai
Pengguna-02	a1	a2	a3	a2	a1	a8	2	Sesuai
Pengguna-03	a1	a2	a3	a2	a1	a3	3	Sesuai
Pengguna-04	a2	a9	a5	a2	a1	a3	1	Sesuai
Pengguna-05	a8	a9	a5	a9	a2	a1	1	Sesuai
Pengguna-06	a5	a10	a4	a8	a2	a1	0	Tidak
Pengguna-07	a1	a2	a3	a8	a2	a1	2	Sesuai
Pengguna-08	a2	a5	a8	a8	a2	a1	2	Sesuai
Pengguna-09	a2	a9	a10	a8	a2	a1	1	Sesuai
Pengguna-10	a3	a5	a8	a8	a2	a1	1	Sesuai
Pengguna-11	a3	a6	a8	a2	a8	a1	1	Sesuai
Pengguna-12	a1	a5	a8	a1	a2	a3	1	Sesuai
Pengguna-13	a2	a5	a8	a8	a2	a1	2	Sesuai
Pengguna-14	a1	a2	a8	a2	a8	a1	3	Sesuai
Pengguna-15	a1	a5	a9	a8	a2	a1	1	Sesuai
Pengguna-16	a1	a2	a8	a8	a2	a1	3	Sesuai
Pengguna-17	a1	a3	a7	a2	a8	a1	1	Sesuai
Pengguna-18	a1	a2	a3	a8	a2	a1	2	Sesuai
Pengguna-19	a7	a9	a10	a5	a1	a2	0	Tidak

Pengguna-20	a5	a6	a8	a8	a2	a1	1	Sesuai

Tabel 6.10 adalah hasil pengujian akurasi dengan nilai hasil pengujian setiap data sampel pada kolom nilai pengujian kolom dan nilai hasil pengujian total didapatkan dari pengolahan nilai pada kolom hasil pengujian total dengan Persamaan 2.13 sehingga hasilnya sebagai berikut:

Akurasi Total =$\frac{18}{20} x100\% = 90\%$

6.1.3 Pengujian *usability*

Pengujian ini dilakukan untuk menghitung tingkat kepuasan dan kemudahan pengguna ketika menggunakan aplikasi yang dikembangkan. Pengujian ini dilakukan dengan cara menyebarkan kuisoner kepada 20 pengguna setelah mereka menggunakan aplikasi ini khususnya menggunakan fitur mendapatkan hasil rekomendasi tempat rental motor. Pengujian ini melibatkan 20 pengguna karena indicator *usability* yang baik adalah dengan menggunakan 20 responden (Sauro, 2013).

Target kuisoner adalah 20 pengguna personal yang familiar dengan *smartphone* khususnya yang memiliki sistem operasi Android yang akan mencoba aplikasi sebagai aktor *user.* Pengguna ini berlatar belakang mahasiswa dan pelajar SMA yang berdomisili di Kota Malang. Hasil pengisian kuisioner oleh responden ditunjukan pada Tabel 6.11.

Tabel 6.11 Hasil rekapitulasi kuisoner pengujian *usability*

NO	PERNYATAAN	JAWABAN					TOTAL
		STS	TS	N	S	SS	
		1	2	3	4	5	
Usefullness							
1	Aplikasi ini membantu memberikan rekomendasi tempat rental motor di kota malang berdasarkan bobot kriteria yang diinginkan.			3	10	7	20
2	Aplikasi ini menghemat waktu saya saat mencari tempat rental motor di kota malang.			1	8	11	20
3	Aplikasi ini sangat berguna bagi saya.			5	11	4	20
4	Aplikasi ini mempermudah saya mencari tahu informasi tempat rental motor di Indonesia meliputi lokasi, nomor telepon, *rating*, popularitas dan harga.				10	10	20
5	Aplikasi ini sesuai dengan apa yang saya harapkan.		2	3	11	4	20

Easy To Learn							
6	Butuh waktu yang cepat untuk mempelajari aplikasi ini.		1	4	6	9	20
7	Penggunaan aplikasi sangat mudah teringat.			1	12	7	20
8	Saya dapat menguasai penggunaan aplikasi ini dengan cepat.			1	12	7	20
Easy To Use							
9	Aplikasi ini mudah digunakan.			6	6	8	20
10	Penggunaan aplikasi ini sangat sederhana.			5	12	3	20
11	Semua kalangan masyarakat dapat menggunakan aplikasi ini.	1		4	13	2	20
12	Aplikasi memiliki langkah-langkah yang sederhana dalam mencapai tujuan dari aplikasi ini.			1	16	3	20
13	Tidak perlu usaha yang banyak dalam penggunaan aplikasi ini.			2	8	10	20
14	Penggunaan aplikasi ini mudah walaupun tanpa instruksi tertulis.			7	8	5	20
15	Inkonsistensi (ketidaksesuaian) tidak ditemukan selama penggunaan aplikasi ini.		2	6	10	2	20
16	Semua kalangan masyarakat akan menyukai aplikasi ini.			9	5	6	20
17	Kesalahan dalam penggunaan aplikasi ini sangat mudah untuk diatasi.			4	13	3	20
18	Penggunaan aplikasi ini sangat praktis ketika saya membutuhkannya.				11	9	20
Satisfaction							
19	Saya puas menggunakan aplikasi ini.		1	1	10	8	20
20	Saya akan merekomendasikan orang lain untuk menggunakan aplikasi ini.			3	13	4	20
21	Saya menggunakan aplikasi ini dengan perasaan yang menyenangkan.			3	13	4	20
22	Aplikasi ini bekerja sesuai dengan apa yang saya inginkan.			2	7	11	20
23	Aplikasi ini sangat bagus.			6	9	5	20

24	Saya ingin memiliki aplikasi ini di *smartphone saya.*			7	6	7	20
25	Aplikasi ini sangat nyaman untuk digunakan.		1	2	12	5	20

Dari hasil pengujian *usability* diatas, dicari indeks persentase untuk mehetahui nilai *usability* dari aplikasi yang dikembangkan, indeks persentase dihitung setiap pertanyaan menggunakan Persamaan 2.14, Persamaan 2.15, dan Persamaan 2.16. Contohnya nilai 84 pada baris 1 kolom indeks merupakan hasil perhitungan dari Persamaan 2.14, Persamaan 2.15, dan Persamaan 2.16. Indeks persentase setiap pertanyaan tersebut akan dihitung rata-ratanya setiap aspek penilain. Contohnya adalah nilai 84 baris rata-rata *usefulness* kolom ideks merupakan hasil dari rata-rata nilai baris 1 sampai 5 kolom indeks. Hasil perhitungan dapat dilihat pada Tabel 6.12.

Tabel 6.12 Hasil perhitungan pengujian *usability*

NO	PERNYATAAN	JAWABAN					TOTAL	INDEKS
		STS	TS	N	S	SS		
		1	2	3	4	5		
Usefullness								
1	Aplikasi ini membantu memberikan rekomendasi tempat rental motor di kota malang berdasarkan bobot kriteria yang diinginkan.			3	10	7	20	84
2	Aplikasi ini menghemat waktu saya saat mencari tempat rental motor di kota malang.			1	8	11	20	90
3	Aplikasi ini sangat berguna bagi saya.			5	11	4	20	79
4	Aplikasi ini mempermudah saya mencari tahu informasi tempat rental motor di Indonesia meliputi lokasi, nomor telepon, *rating*, popularitas dan harga.				10	10	20	90
5	Aplikasi ini sesuai dengan apa yang saya harapkan.		2	3	11	4	20	77
Rata-rata *Usefullness*								84
Easy To Learn								
6	Butuh waktu yang cepat untuk mempelajari aplikasi ini.		1	4	6	9	20	83
7	Penggunaan aplikasi sangat mudah teringat.			1	12	7	20	86

8	Saya dapat menguasai penggunaan aplikasi ini dengan cepat.			1	12	7	20	86
Rata-rata *Easy To Learn*								85
Easy To Use								
9	Aplikasi ini mudah digunakan.			6	6	8	20	82
10	Penggunaan aplikasi ini sangat sederhana.			5	12	3	20	78
11	Semua kalangan masyarakat dapat menggunakan aplikasi ini.	1		4	13	2	20	75
12	Aplikasi memiliki langkah-langkah yang sederhana dalam mencapai tujuan dari aplikasi ini.			1	16	3	20	82
13	Tidak perlu usaha yang banyak dalam penggunaan aplikasi ini.			2	8	10	20	88
14	Penggunaan aplikasi ini mudah walaupun tanpa instruksi tertulis.			7	8	5	20	78
15	Inkonsistensi (ketidaksesuaian) tidak ditemukan selama penggunaan aplikasi ini.		2	6	10	2	20	72
16	Semua kalangan masyarakat akan menyukai aplikasi ini.			9	5	6	20	77
17	Kesalahan dalam penggunaan aplikasi ini sangat mudah untuk diatasi.			4	13	3	20	79
18	Penggunaan aplikasi ini sangat praktis ketika saya membutuhkannya.				11	9	20	89
Rata-rata *Easy To Use*								80
Satisfaction								
19	Saya puas menggunakan aplikasi ini.		1	1	10	8	20	85
20	Saya akan merekomendasikan orang lain untuk menggunakan aplikasi ini.			3	13	4	20	81
21	Saya menggunakan aplikasi ini dengan perasaan yang menyenangkan.			3	13	4	20	81
22	Aplikasi ini bekerja sesuai dengan apa yang saya inginkan.			2	7	11	20	89

23	Aplikasi ini sangat bagus.			6	9	5	20	79
24	Saya ingin memiliki aplikasi ini di *smartphone saya.*			7	6	7	20	80
25	Aplikasi ini sangat nyaman untuk digunakan.		1	2	12	5	20	81
Rata-rata *Satisfaction*								82.285

Dari seluruh indeks persentase akan dihitung rata-ratanya untuk menghitung rata-rata indeks persentase seluruh kriteria. Hasil perhitungannya dapat dilihat pada Tabel 6.13.

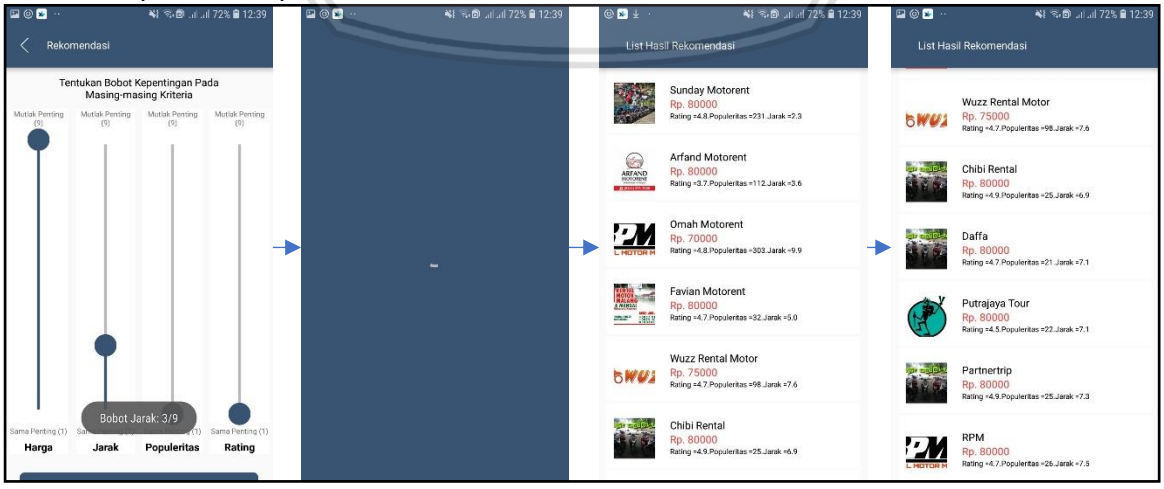
Tabel 6.13 Hasil perhitungan pengujian *usability*

Aspek Penilaian	Rata-rata Persentase (%)
Usefulness	84
Ease to learn	85
Ease to use	80
Satisfaction	82.285
Rata-rata	**82.82 %**

Hasil yang diperoleh untuk mencari rata-rata indeks persentase seluruh kriteria adalah 82.82 %.

6.1.4 Pengujian algoritma

Pengujian algoritma ini dilakukan untuk mengetahui apakah perancangan algoritma berhasil dilakukan implementasi atau tidak. Pengujian dilakukan dengan cara membandingkan hasil dari perancangan algoritma dengan keluaran aplikasi. Keluaran aplikasi dengan masukan yang sama seperti contoh kasus pada Bagian 4.2 dapat dilihat pada Gambar 6.1.



Gambar 6.1 Tampilan aplikasi pengujian algoritma

Dari Gambar 6.1 dapat diketahui bahwa masukan aplikasi berupa masukan bobot kriteria menyatakan bahwa kriteria harga lebih penting dari kriteria jarak, popularitas, dan *rating* dengan nilai bobot kepentingan 9. Disusul dengan kriteria jarak yang lebih penting dari kriteria popularitas dan *rating* dengan bobot kepentingan 3. Sedangkan kriteria popularitas dan *rating* memiliki tingkat kepentingan yang sama dengan bobot kepentingan 1. Berdasarkan masukan tersebut diperoleh keluaran aplikasi berupa urutan tempat rental motor, yaitu Sunday Motorent, Arfand Motorent, Omah Motorent, Faviant Motorent, Wuzz Rental Motor, Chibi, Daffa, Putrajaya Tour, Partnertrip, RPM.

6.2 Analisis

Analisis dilakukan pada setiap pengujian yang sudah dilakukan, yaitu pengujian fungisonal, pengujian kesesuaian, dan pengujian *usability,* analisis dilakukan untuk mempermudah penarikan kesimpulan.

6.2.1 Analisa hasil pengujian fungsional

Hasil dari pengujian menyimpulkan bahwa implementasi pengembangan pembangunan aplikasi Android rekomendasi tempat rental motor di kota malang dengan metode AHP TOPSIS berbasis *location based services* telah memenuhi rekayasa kebutuhan dan perancangan karena semua kebutuhan fungsional yang diuji terbukti valid sehingga tingkat validitas bernilai 100%.

6.2.2 Analisa hasil pengujian kesesuaian

Hasil dari pengujian kesesuaian didapati dari nilai hasil pengujian total yang bernilai 90 %. Nilai ini termasuk cukup tinggi sebagai hasil dari pengujian keluaran sistem dengan pilihan pengguna. Hal ini berarti aplikasi ini memiliki tingkat kesesuaian yang tinggi.

6.2.3 Analisa hasil pengujian *usability*

Hasil dari pengujian *usability* memiliki nilai 82.82 %. Berdasarkan interpretasi skor di Tabel 6.14, hasil dari pengujian ini berarti pengguna sangat setuju dengan pertanyaan-pertanyaan yang diberikan. Aplikasi ini dapat dikatakan memenuhi indeks dari pengujian *usability* karena hasil pengujian melebihi 60% (Kharisma, 2018).

Tabel 6.14 Interpretasi skor Likert

Skor Likert	Interpretasi skor dengan interval = 20	Pilihan
1	0% - 19,99%	Sangat tidak setuju
2	20% - 39,99%	Tidak setuju
3	40% - 59,99%	Netral
4	60% - 79,99%	Setuju
5	80% - 100%	Sangat setuju

Keterangan:
Interval = 20 dari pembagian jumlah skor Likert dengan nilai 100.

Hasil dari pengujian *usability* ini membuktikan bahwa aplikasi ini dapat memudahkan pengguna untuk memilih tempat rental motor di Kota Malang.

6.2.4 Analisa hasil pengujian algoritma

Hasil dari pengujian algoritma adalah Sunday Motorent, Arfand Motorent, Omah Motorent, Faviant Motorent, Wuzz Rental Motor, Chibi, Daffa, Putrajaya Tour, Partnertrip, RPM. Sedangkan hasil dari perancangan algoritma berdasarkan Tabel 4.24 adalah a2, a1, a8, a3, a5, a9, a7, a6, a10, a4. Berdasarkan Tabel 4.17 hasil dari perancangan algoritma ini sesuai dengan hasil pengujian algoritma yaitu Sunday Motorent, Arfand Motorent, Omah Motorent, Faviant Motorent, Wuzz Rental Motor, Chibi, Daffa, Putrajaya Tour, Partnertrip, RPM. Hal ini berarti perancangan algoritma berhasil melalui implementasi dengan baik.

BAB 7 PENUTUP

Bab ini berisi kesimpulan dan saran. Kesimpulan berisi jawaban akan rumusan masalah. Saran ditujukan kepada pembaca yang ingin melanjutkan penelitian ini dengan penambahan-penambahan atau kemajuan-kemajuan.

7.1 Kesimpulan

Dari hasil rekayasa kebutuhan, perancngan, implementasi dan, pengujian yang telah dilakukan peneliti sebelumnya, maka dapat ditarik kesimpulan sebagai berikut:

1. Rekayasa kebutuhan yang telah dilakukan sebelumnya memiliki hasil dalam bentuk gambaran umum sistem, identifikasi aktor, aturan penomoran, kebutuhan fungsional yang terdiri dari 6 kebutuhan, dan kebutuhan non fungsional yang terdiri dari 2 kebutuhan. Kebutuhan fungsional tersebut dimodelkan ke dalam *use case diagram* dan *use case scenario.*
2. Perancangan yang dilakukan terdiri dari dua tahap, yaitu yang pertama adalah perancangan algoritma dan yang kedua perancangan sistem. Perancangan algoritma berisi langkah manual perhitungan menggunakan metode AHP dan TOPSIS. Percangan sistem terdiri dari perancangan UML, perancangan basis data, perancangan *pseudocode,* dan perancangan antarmuka.
3. Implementasi yang telah dilakukan memperolah hasil yang berbentuk implementasi basis data, implementasi kode program, dan implementasi antarmuka berdasarkan perancangan yang sudah disusun.
4. Tingkat kesesuaian yang diperoleh dari pengujian dapat dibilang baik dikarenakan memiliki hasil yang bisa dikatakan tinggi yaitu 90 %.
5. Tingkat usability aplikasi ini juga tergolong baik karena menyentuh nilai yang cukup tinggi, yaitu 82.82 %.
6. Perancangan algoritma berhasil melalui implementasi dengan baik karena hasil pengujian dengan perancangan algoritma sesuai.

7.2 Saran

Saran yang dapat diberikan untuk pengembangan sistem selanjutnya adalah sebagai berikut:

1. Perbaikan antarmuka untuk memberi kenyaman pengguna.
2. Penambahan fitur untuk melakukan pemesanan pada tempat rental.
3. Penambahan kriteria lain untuk pemilihan tempat rental motor.
4. Mendapatkan hasil rekomendasi tempat rental motor menggunakan metode SPK yang berbeda.

DAFTAR PUSTAKA

Abadi, F. (2016). Penentuan Penerima Bantuan Dana untuk Sekolah Menengah Di Kab. Banjar Menggunakan Metode AHP-TOPSIS Dengan Pendekatan Fuzzy. *Journal Speed – Sentra Penelitian Engineering dan Edukasi – Volume 8 No 1*, 44-50.

Aelani, K., & Falahah. (2012). Pengukuran Usability Sistem Menggunakan USE Questionnaire (Studi Kasus Aplikasi Perwalian Online STIK "AMIKBANDUNG"). 1-6.

Agustina, N., Risnanto, S., & Supriadi, I. (2016). Pengembangan Aplikasi Location Based Service Untuk Informasi Dan Pencarian Lokasi Pariwisata Di Kota Cimahi Berbasis Android. *Jurnal Ilmiah Teknologi Informasi Terapan*, 53-39.

Andayati, D. (2010). Sistem Pendukung Keputusan Pra-Seleksi Penerimaan Siswa Baru (Psb) On-Line Yogyakarta. *Jurnal Teknlogi, Volume 3 Nomor 2*, 145-153.

Andikasani, M. R., Awaluddin, M., Suprayogi, A., & Darmo, B. Y. (2014). *Aplikasipersebaran Objek Wisata Di Kota Semarang Berbasis Mobile Gismemanfaatkan Smartphone Android.* Semarang: Universitas Diponegoro.

Azis, G. M., Cholissidin, I., & Furqon, M. T. (2017). Sistem Pendukung Keputusan untuk Rekomendasi Wirausaha. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer* , 1204-1214.

Chairi, A., Mardi Putri, R. R., & Fanani, L. (2018). Rekomendasi Tempat Wisata Kota Malang Menggunakan Metode Profile Matching Dan Saran Rute Menggunakan Floyd Warshall Berbasis Android . *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer* , 2060-2069.

Dewi, R. K., Hanggara, B. T., & Pinandito, A. (2018). A Comparison Between AHP and Hybrid AHP for Mobile Based Culinary Recommendation System. *International Journal of Interactive Mobile Technologies*, 133-140.

Firdausy, V. N., Agus, F., & Astuti, I. F. (2017). Aplikasi Android Hybrid Untuk Pemilihan Lokasi Kuliner. *Jurnal Informatika Mulawarman* , 30-37.

Hardianto, F., & Handaga, B. (2015). *Aplikasi Grupchat Di Android Menggunakan Websocket.* Surakarta: Universitas Muhammadiyah .

Hidayat, B. R., & Februariyanti, H. (2013). Aplikasi Location Based Service (Lbs) Pencarian Lokasi Taxi Pada Android Di Kota Semarang. *DINAMIKA INFORMATIKA*, 16-25.

Juliyanti, Irawan, M. I., & Mukhlash, I. (2011). Pemilihan Guru Berprestasi Menggunakan Metode Ahp Dan Topsis . *Prosiding Seminar Nasional Penelitian, Pendidikan dan Penerapan MIPA*, 63-68.

Khairani, L., Soraya, R. E., Petrus, J., & Rachmansyah. (2013). *Rancang Bangun Aplikasi Pemantauan Posisi Anggota Keluarga Berbasis Android.* Palembang: STMIK GI MDP.

Kharisma, R. (2018). *Pengembangan Aplikasi Mobile Untuk Mencari Dan Memberikan Pertolongan Terhadap Masalah Pada Kendaraan Berdasarkan Lokasi Terdekat .* Malang: UNIVERSITAS BRAWIJAYA.

Lestianti, L. (2016, November 12). *Kemajuan Pariwisata Kabupaten Malang Berkembang Pesat!* Retrieved from Lestelita: http://www.lestelita.com/2016/11/kemajuan-pariwisata-kabupaten-malang.html

Nielsen, J. (2012, Januari 4). *Usability 101: Introduction to Usability*. Retrieved from Nielsen Norman Group: https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Nofriansyah, D. (2014). *Konsep Data Mining Vs Sistem Pendukung Keputusan.* Yogyakarta: Deepublish.

Nurdiyanto, H., & Meilia, H. (2016). Sistem Pendukung Keputusan Penentuan Prioritas Pengembangan Industri Kecil Dan Menengah Di Lampung Tengah Menggunakan Analitical Hierarchy Process (AHP). *Seminar Nasional Teknologi Informasi dan Multimedia 2016*, 37-42.

Pressman, R. S. (2001). *Software Engineering A Practitioner's Approach.* Thomas Casson.

Purnomo, E. N., Sihwi, S. W., & Anggrainingsih, R. (2013). Analisis Perbandingan Menggunakan Metode AHP, TOPSIS, dan AHP-TOPSIS dalam Studi Kasus Sistem Pendukung Keputusan Penerimaan Siswa Program Akselerasi. *JURNAL ITSMART* , 16-23.

Risnita. (2014). Pengembangan Skala Model Likert. 1-14.

Sauro, J. (2013, Juni 18). *10 Things To Know About The System Usability Scale (SUS)*. Retrieved From Measuring U: https://measuringu.com/10-things-sus/

Steiniger, S., Neun, M., & Edwardes, A. (2006). Foundations of Location Based Services. In *CartouCHe* (pp. 1-28). Zürich: University of Zurich, ETH Zurich.

Swasono, R. U. (2015). *Sistem Pemilihan Guru Berprestasi Menggunakan Metode AHP-TOPSIS (Studi Kasus: Dinas Pendidikan Kabupaten Bojonegoro).* Malang: FILKOM UB.