PENGEMBANGAN APLIKASI ANDROID REKOMENDASI LAPANGAN TENIS DI MALANG RAYA DENGAN *GROUP DECISION SUPPORT SYSTEM* DAN *LOCATION BASED SERVICE*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Erastus Mauliate
NIM: 155150200111224

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2021

PENGESAHAN

PENGEMBANGAN APLIKASI ANDROID REKOMENDASI LAPANGAN TENIS DI MALANG RAYA DENGAN *GROUP DECISION SUPPORT SYSTEM* DAN *LOCATION BASED SERVICE*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh:
Erastus Mauliate
NIM: 155150200111224

Skripsi ini telah diuji dan dinyatakan lulus pada
7 Mei 2021
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Ratih Kartika Dewi, S.T., M.Kom.	Komang Candra Brata, S.Kom., M.T., M.Sc.
NIK: 201503 89020 2 001	NIK: 201607 890711 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Achmad Basuki, S.T., M.MG., Ph.D.
NIP: 19741118 200312 1 002

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 secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan 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, 20 April 2021

Erastus Mauliate
NIM: 155150200111224

PRAKATA

Puji syukur kehadirat Tuhan Yang Maha Esa, karena telah memberikan berkat dan kasih karunia yang melimpah kepada penulis, sehingga skripsi yang berjudul "Pengembangan Aplikasi Android Rekomendasi Lapangan Tenis di Malang Raya dengan *Group Decision Support System* dan *Location Based Service*" dapat terselesaikan. Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada :

1. Ibu Ratih Kartika Dewi, S.T., M.Kom selaku dosen pembimbing 1 yang telah meluangkan waktu untuk membimbing, memberikan saran dan motivasi dalam proses penyelesaian skripsi.
2. Bapak Komang Candra Brata, S.Kom., M.T., M.Sc. selaku dosen pembimbing 2 yang telah memberikan saran dan masukan dalam proses penyelesaian skripsi.
3. Bapak Achmad Basuki, S.T., M.MG., Ph.D selaku Ketua Jurusan Teknik Informatika.
4. Bapak Adhitya Bhawiyuga, S.Kom., M.Sc. selaku Ketua Program Studi Teknik Informatika.
5. Mama, Abang, dan Adik yang telah memberikan dukungan moral dan finansial serta selalu mendoakan untuk menyelesaikan skripsi.
6. Teman-teman mahasiswa jurusan Teknik informatika Angkatan 2015 yang telah memberikan dukungan dalam proses penyelesaian skripsi.
7. Teman-teman PMK Daniel yang selalu ada dalam memberikan dukungan secara rohani dan mental.
8. Teman-teman Anonyminus yang selalu memberikan bantuan dan dukungan yang penuh semangat.
9. Pihak-pihak lain yang terlibat secara langsung ataupun tidak langsung dalam menyelesaikan skripsi ini.

Penulis menyadari bahwa dalam penyusunan laporan ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 20 April 2021

Penulis

emauliate17@gmail.com

ABSTRAK

Erastus Mauliate, Pengembangan Aplikasi Android Rekomendasi Lapangan Tenis di Malang Raya dengan GDSS dan LBS

Pembimbing: Ratih Kartika Dewi, S.T., M.Kom dan Komang Candra Brata, S.Km., M.T., M.Sc.

Olahraga tenis lapangan merupakan salah satu dari lima olahraga terpopuler di Indonesia. Tenis lapangan dapat dimainkan oleh dua orang pemain atau antara dua pasangan yang terdiri dari dua pemain. Tenis lapangan juga cukup digemari oleh para pendatang yang ada di kota Malang. Para pendatang tersebut memiliki permasalahan terkait pencarian lapangan tenis terbaik yang bisa digunakan lebih dari satu orang. Oleh karena itu, penelitian ini bertujuan untuk mengembangkan aplikasi rekomendasi lapangan tenis di kota Malang berbasis Android dengan menggunakan *Group Decision Support System* (GDSS). Metode ini diharapkan mampu membantu pengguna dalam menentukan rekomendasi lapangan tenis berdasarkan kriteria yang ada. Pada penelitian sebelumnya, Heru Budiyanto telah melakukan hal yang serupa namun lebih menekankan kepada pendekatan rekomendasi yang ditujukan kepada setiap individu saja. Pada penelitian ini, aplikasi difokuskan kepada perhitungan rekomendasi secara berkelompok dengan menggunakan metode *Technique for Order Preference by Similarity to Ideal Solution* (TOPSIS) dan BORDA dengan empat kriteria yang ada seperti harga, jarak, rating, dan jumlah lapangan. Aplikasi ini juga menerapkan metode *Location Based Service* (LBS) untuk memudahkan pengguna menuju lokasi lapangan yang ditandai dengan fitur penunjuk arah. Data dari setiap lapangan akan tersimpan pada Google Firebase. Aplikasi ini telah diuji berdasarkan tiga metode, yaitu pengujian *black-box*, pengujian validasi algoritma, dan pengujian *usability*. Dari ketiga pengujian tersebut, didapatkan hasil bahwa aplikasi dapat berjalan sesuai dengan fungsionalitasnya dan juga sesuai dengan perhitungan algoritma secara manual. Aplikasi ini mendapatkan skor sebesar 73,5 berdasarkan pengujian *usability*, yang berarti aplikasi cukup dapat diterima oleh pengguna.

Kata kunci: Rekomendasi, Tenis Lapangan, GDSS, LBS, Android, TOPSIS-BORDA.

ABSTRACK

Erastus Mauliate, *Development of Recommendations Android Applications for Tennis Courts in Malang Raya with GDSS and LBS*

Supervisors: Ratih Kartika Dewi, S.T., M.Kom and Komang Candra Brata, S.Kom., M.T., M.Sc.

Tennis is the one of the five most popular sports in Indonesia. Tennis is played by two players or two pairs of two players. Tennis is also quite popular with newcomers in Malang. They have a problem to finding the best tennis court that can be used by more than one person. Therefore, this study aims to develop an Android-based tennis court recommendation application in Malang using the Group Decision Support System (GDSS). This method is expected to be able to assist users in determining recommendations for tennis courts based on existing criteria. Heru Budiyanto has working the same thing but more focused on the recommendation approach aimed at each individual only in previous research. In this research, the application enables a group-based recommendations using the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) and BORDA methods with four criteria such as price, distance, rating, and number of fields. This application also implements the Location Based Service (LBS) method to aid users to reach the selected tennis court. Data from each field will be stored on Google Firebase. This application has been tested based on three methods, such as black-box testing, algorithm validation testing, and usability testing. From the three tests, the results show that the application can run according to its functionality and also according to the algorithmic calculation manually. The score of usability testing is 73,5, showing that the application is acceptable for the user.

Keywords: *Recommendations*, *Tennis Court*, *GDSS, LBS, Android, TOPSIS-BORDA.*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Olahraga tenis lapangan merupakan salah satu dari lima olahraga terpopuler di Indonesia. Data ini didapatkan berdasarkan survei Media Nielsen Sport yang telah dilakukan pada bulan Mei tahun 2020 (Insights, 2020). Tenis adalah permainan yang membutuhkan kecepatan, ketepatan yang terkontrol, daya tahan, antisipasi, tekad dan kreativitas (Lardner, 2003). Permainan tenis lapangan mulai dipopulerkan di Indonesia pada tahun 1920-an melalui sekolah-sekolah menengah yang ada di Jakarta dan Surabaya, seperti Stovia, Rechrsschool, dan NIAS. Kepopuleran olahraga ini dapat dilihat dari keikutsertaan tiga wakil daerah Indonesia dalam kompetisi De Alegemeene Nederland-sche Turf Tennis Bond (ANILTB) yang diadakan di Malang pada akhir tahun 1934, olahraga tenis di Indonesia menjadi semakin berkembang pesat. Kemudian pada bulan Desember 1935, Persatuan Lawn Tennis Indonesia (PELTI) didirikan untuk mengembangkan dan memajukan permainan olahraga tenis di dalam negeri dan seluruh tanah air.

Di masa kini, permainan tenis lapangan masih tetap digemari oleh banyak lapisan masyarakat di berbagai kota khususnya di kota Malang, tempat dimana kompetisi tenis diadakan pertama kali di Indonesia. Di kota Malang terdapat banyak pendatang dari berbagai wilayah untuk sekadar berwisata ataupun menetap. Para pendatang yang menggemari olahraga tenis sendiri memiliki permasalahan terkait pencarian lapangan tenis yang bisa digunakan di kota Malang. Kurangnya informasi yang cukup bagi pendatang dapat mengakibatkan meredupnya olahraga tenis di masa mendatang. Dalam hal ini, tentu saja para pendatang tersebut memerlukan aplikasi sistem pendukung keputusan yang tepat untuk bisa menunjukkan lokasi terdekat lapangan tenis yang ada di kota Malang.

Sistem pendukung keputusan dapat berjalan sebagai alat yang bisa digunakan untuk menyampaikan masalah dan untuk memecahkan masalah dalam kondisi semi terstruktur dan tidak terstruktur (Turban, 2001). *Group Decision Support system* (GDSS) merupakan salah satu metode yang ada dalam sistem pendukung keputusan dan digunakan pada penelitian ini. GDSS merupakan sebuah sistem yang dapat melakukan pemecahan masalah dengan pikiran komputer untuk mendukung sekelompok orang dalam menyelesaikan tugas bersama dan memberikan tampilan layar bersama.

Sistem pendukung keputusan juga dapat digunakan dalam pencarian lokasi dengan penerapan *Location Based Service* (LBS). LBS dapat mendeteksi lokasi perangkat yang digunakan oleh pengguna sekaligus menampilkan kalkulasi jarak antara pengguna dengan tempat yang dituju (teknojurnal.com, 2019). Layanan tersebut juga menggunakan teknologi layanan *global positioning* atau teknologi yang biasa disebut dengan GPS.

Penulis menggunakan metode TOPSIS dalam kasus ini karena dari beberapa kasus penelitian sebelumnya, TOPSIS memiliki beberapa keunggulan khusus dalam mengolah data yang memiliki banyak kriteria dan memiliki perhitungan yang sederhana sehingga proses logika yang dilakukan menjadi ringan dan cepat.

TOPSIS mendukung sistem pendukung keputusan dengan akurat dan memiliki kompleksitas waktu yang rendah. Selain itu TOPSIS juga memiliki nilai solusi positif yang dekat dan nilai solusi negatif yang jauh (Organ, A., 2013). Kompleksitas waktu yang rendah dan memiliki performa yang baik karena sifatnya yang ringan akan sangat cocok untuk dikembangkan dan diimplementasikan dalam perangkat berbasis mobile (Julanto, H.J., 2018).

Pada penelitian ini menggunakan kriteria seperti jarak antara pengguna dengan tempat lapangan tenis, rata-rata harga sewa lapangan tenis per jamnya, *rating* lapangan, dan jumlah lapangan yang tersedia. Setiap kriteria berfungsi sebagai bobot dalam perhitungan menggunakan metode TOPSIS. Hasil dari perhitungan kemudian akan diperingkatkan menggunakan BORDA agar pengguna dapat melihat rekomendasi terbaik dari atas ke bawah yang muncul pada layar pengguna.

BORDA digunakan dalam penelitian ini karena perhitungannya yang akurat jika dihubungkan dengan TOPSIS. Metode BORDA digunakan untuk melakukan penggabungan penilaian para pengambil keputusan. Aplikasi ini dapat menghasilkan urutan lapangan tenis terbaik, sehingga dapat membantu para pengambil keputusan untuk mendapatkan keputusan terbaik.

Melihat permasalahan tersebut, peneliti dapat memberikan solusi berupa penggunaan metode GDSS dan LBS untuk mengembangkan aplikasi rekomendasi lapangan tenis berbasis android di Malang Raya. Metode perhitungan yang digunakan oleh penulis adalah metode TOPSIS-BORDA. Harapannya dapat membantu pendatang baru ataupun penggemar olahraga tenis saat berada di wilayah Malang Raya dalam menentukan lapangan tenis terdekat dari tempat tinggal.

1.2 Rumusan Masalah

Dari uraian masalah pada latar belakang di sub-bab sebelumnya, maka rumusan masalah yang didapat adalah sebagai berikut:

1. Bagaimana hasil perancangan sistem rekomendasi lapangan tenis di kota Malang berbasis android?
2. Bagaimana hasil implementasi sistem rekomendasi lapangan tenis dengan GDSS dan LBS?
3. Bagaimana hasil pengujian dari sistem rekomendasi lapangan tenis di kota malang setelah diimplementasikan?

1.3 Tujuan

Tujuan dari penelitian ini dapat diringkas sebagai berikut:

1. Menghasilkan rancangan aplikasi rekomendasi lapangan tenis di Malang Raya berbasis LBS.
2. Menghasilkan implementasi penentuan lokasi lapangan tenis di kota Malang secara akurat menggunakan metode GDSS dan LBS.

3. Mengetahui hasil pengujian aplikasi rekomendasi lapangan tenis di Malang Raya dapat berjalan dengan baik.

1.4 Manfaat

Hasil penelitian diharapkan dapat membantu pengguna aplikasi perangkat bergerak khususnya masyarakat yang menyukai olahraga tenis dalam memilih ketersediaan lapangan tenis di wilayah Malang Raya.

1.5 Batasan Masalah

Supaya topik permasalahan dapat difokuskan dan tidak melebar, maka dari itu perlu dibuat Batasan masalah. Batasan-batasan dari penelitian yang telah ditentukan adalah sebagai berikut:

1. Penelitian ini hanya menganalisa data sekunder, yaitu persebaran lapangan tenis di kota Malang.
2. Penelitian ini menggunakan Bahasa pemrograman java sebagai implementasi kode program dan Firebase sebagai tempat penyimpanan data.
3. Aplikasi dijalankan pada platform Android dengan minimum API 18 sampai dengan API 28.
4. Aplikasi membutuhkan internet untuk keluaran dan penyimpanan lokasi.
5. Informasi lokasi lapangan tenis memerlukan akurasi GPS sesuai perangkat bergerak yang dimiliki pengguna.

1.6 Sistematika Pembahasan

Pembahasan dari penelitian memiliki sistematika yang harus diikuti. Kerangka acuan berpikir yang disusun secara sistematis tersebut terdiri atas tujuh bab pembahasan, yaitu:

BAB I PENDAHULUAN

Pada bab ini menjelaskan berbagai hal yang akan melatarbelakangi pengembangan dan implementasi aplikasi, menjelaskan rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan menjelaskan sistematika pembahasan penelitian.

BAB II LANDASAN PUSTAKA

Bab ini berisikan tentang kajian pustaka dan landasan teori yang ada dalam penelitian. Kajian pustaka akan dijadikan sebagai acuan penelitian komparatif dan penelitian sebelumnya yang dilakukan oleh peneliti lain, dan landasan teori akan memaparkan teori pendukung yang digunakan dalam aplikasi.

BAB III METODE PENELITIAN

Bab ini membahas tentang tahapan secara terstruktur untuk mengerjakan dan menyelesaikan penelitian tentang Pengembangan Aplikasi Android Rekomendasi Lapangan Tenis di Malang Raya dengan GDSS dan LBS.

BAB IV ANALISIS DAN PERANCANGAN

Bab ini menguraikan setiap kebutuhan fungsional maupun nonfungsional pada penelitian. Bab ini juga menjelaskan tentang perancangan dalam aplikasi sehingga menyelesaikan permasalahan yang dibahas dan telah siap untuk ke tahap selanjutnya.

BAB V IMPLEMENTASI

Bab ini membahas cara melakukan implementasi dari perancangan yang telah dibuat sebelumnya. Bab ini menjelaskan tentang batasan spesifikasi sistem, batasan implementasi, implementasi basis data, implementasi kode program, dan implementasi antarmuka dari aplikasi.

BAB VI PENGUJIAN

Bab ini berisikan proses pengujian fungsional berupa pengujian *black-box*, pengujian non-fungsional berupa pengujian *usability*, dan pengujian validasi algoritme. Pengujian dilakukan untuk melihat kelayakan aplikasi oleh pengguna.

BAB VII PENUTUP

Bab ini berisikan tentang kesimpulan yang dapat diambil dari hasil yang pengembangan aplikasi yang telah dilakukan dan saran yang dapat diberikan untuk penelitian ke depannya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Sistem Rekomendasi

Sistem rekomendasi merupakan aplikasi yang dapat memberikan saran kepada pengguna dalam mengambil keputusan yang tepat (Ungkawa, et al., 2013). Sistem rekomendasi memanfaatkan opini seseorang terhadap ketertarikan pada suatu barang dalam kategori tertentu untuk membantu dalam memilah barang yang disukai oleh pengguna (Ungkawa, et al., 2013). Secara sederhananya, rekomendasi seringkali ditampilkan dalam bentuk peringkat dari suatu barang. Dalam melakukan pemeringkatan ini, sistem rekomendasi mencoba memprediksi produk atau layanan apa saja yang paling cocok kepada pengguna berdasarkan preferensi dan kendala pengguna (F. Ricci, et al., 2011).

Saat ini, konsep sistem rekomendasi sudah banyak ditemui di berbagai bidang bisnis untuk proses pengambilan keputusan yang diserahkan secara bebas oleh konsumen (Sharda, 2010). Sistem rekomendasi lapangan tenis menggunakan konsep ini untuk menolong masyarakat penggemar olahraga tenis di kota Malang berdasarkan wilayah terdekat dari tempat tinggalnya.

Beberapa metode atau teknik digunakan dalam sistem rekomendasi. Setiap metode berlaku untuk pertanyaan yang menghasilkan informasi yang sesuai. Belka (2004) menunjukkan bahwa metode atau pendekatan yang dipilih dalam sistem rekomendasi bergantung pada masalah yang akan dipecahkan. Teknik rekomendasi yang berbeda digunakan untuk aplikasi yang berbeda, dasar tujuan, dan tujuan aplikasi.

2.2 Sistem Pendukung Keputusan

Sistem adalah sekumpulan objek dengan fungsi tertentu yang dirancang untuk mencapai tujuan. Sistem Pendukung Keputusan (SPK) atau *Decision Support System* (DSS) merupakan sebuah sistem yang ditujukan untuk manajemen pengambilan keputusan (Subakti, 2002).

Menurut Subakti (2002), Sistem Pendukung Keputusan (SPK) memiliki kemampuan untuk menyelesaikan masalah yang tidak terstruktur. SPK hanya mampu membantu untuk mengambil keputusan tanpa mengesampingkan kemampuan seorang pengambil keputusan dalam mengambil keputusan. SPK bertujuan untuk memberikan informasi, panduan, memberikan prakiraan dan memandu pengguna informasi untuk mengambil keputusan yang lebih baik. SPK tidak hanya dipengaruhi oleh masukkan, proses, dan keluaran, tetapi ada juga umpan balik dalam SPK, yaitu aliran informasi dari komponen keluaran ke pengambilan keputusan, di mana keluaran atau kinerja sistem harus diperhatikan.

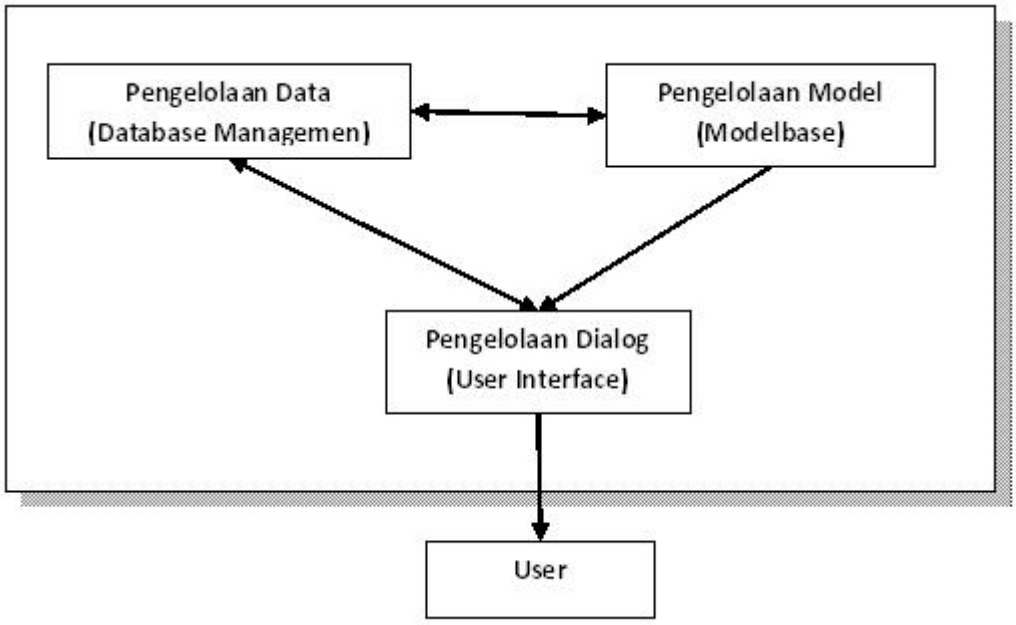
Sprague dan Waton (1993) mendefinisikan Sistem Pendukung Keputusan (SPK) sebagai sistem dengan lima ciri utama, yaitu:

1. Sistem yang berbasis komputer.

2. Sistem yang digunakan untuk membantu para pengambil keputusan.
3. Sebagai solusi dalam memecahkan masalah yang rumit dan mustahil jika dilakukan dengan kalkulasi manual.
4. Simulasi yang interaktif.
5. Komponen utama adalah data dan model.

2.2.2 Komponen Sistem Pendukung Keputusan

Menurut Turban (2001), Sistem Pendukung Keputusan (SPK) memiliki komponen penting di dalamnya, di antaranya adalah Pengelolaan Data, Pengelolaan Model, dan Pengelolaan Dialog. Setiap komponen saling terhubung dan memiliki perannya masing-masing. Ketiga komponen tersebut ditunjukkan pada Gambar 2.1 di bawah ini.



Gambar 2.1 Komponen Besar Sistem Pendukung Keputusan

Sumber: Turban (2001)

Pengelolaan Data. Merupakan subsistem data yang diatur dalam basis data. Sebagai sistem pendukung keputusan, data dapat berasal dari luar atau dalam lingkungan. Untuk keperluan SPK, diperlukan data yang terkait dengan masalah yang akan diselesaikan dengan simulasi.

Pengelolaan Model. Merupakan model (misalnya, model matematika) yang mengungkapkan suatu masalah dalam format kuantitatif sebagai dasar untuk simulasi atau pengambilan keputusan, termasuk tujuan (objektif) masalah, komponen terkait, batasan yang ada, dan hal-hal terkait lainnya. Pengelolaan Model memungkinkan pengambil keputusan untuk melakukan analisis komprehensif dengan mengembangkan dan membandingkan solusi alternatif.

Pengelolaan Dialog. Bentuknya seperti subsistem dialog, ini adalah kombinasi dari dua komponen sebelumnya (yaitu Pengelolaan Data dan Pengelolaan Model), yang dipadukan menjadi komponen baru yaitu Pengelolaan Dialog yang dapat dimengerti oleh komputer. Pengelolaan Dialog berfungsi sebagai keluaran sistem bagi pengguna, dan masukan dari pengguna ke dalam SPK.

2.3 Tenis Lapangan

Tenis adalah olahraga yang biasanya dimainkan antara dua pemain atau antara dua pasangan yang terdiri dari dua pemain. Setiap pemain memukul bola karet dengan raket dan dibatasi oleh net. Tenis adalah permainan yang membutuhkan kecepatan, ketepatan yang terkontrol, daya tahan, antisipasi, tekad dan kreativitas (Lardner, 2003).

Permainan tenis pada dasarnya dilakukan dengan cara memukul bola hingga melewati net lalu jatuh ke daerah lawan, mengincar jarak terjauh dari lawan agar bola sulit dikembalikan sehingga mendapatkan angka untuk kemenangan. Di dalam lapangan, setiap pemain akan dibatasi net di tengah lapangan. Permainan tenis lapangan mewajibkan pemain untuk menggunakan raket dan bola sebagai alat dari permainan ini. Pergerakan dalam tenis lapangan diharuskan untuk memiliki respon gerakan yang cepat dalam menangkis serangan lawan karena faktor lapangan yang cukup luas. Tujuan utama dari tenis adalah untuk memukul bola ke kotak lawan sehingga lawan tidak bisa menyetnuhnya sama sekali, atau membuat bola lawan menyimpang dari garis lurus, atau membuat bola lawan membentur net (Lardner, 2003).

2.4 *Group Decision Support System* (GDSS)

Group Decision Support System (GDSS) adalah sistem interaktif berbasis komputer yang dapat membantu sekelompok pembuat keputusan untuk memecahkan masalah yang tidak terstruktur. Sistem ini terdiri dari komponen SPK secara keseluruhan, ditambah dengan perangkat lunak khusus untuk membantu kelompok penentu keputusan (Istudor, 2010).

GDSS mendukung penggunaan lebih dari satu orang untuk melakukan tugas bersama. Tujuan dari tugas ini adalah untuk meningkatkan efisiensi dan efektivitas dalam pertemuan untuk pengambilan keputusan dengan cara mempercepat proses pengambilan keputusan atau meningkatkan kualitas keputusan akhir. Tujuan tersebut dapat dicapai dengan memberikan dukungan untuk pertukaran ide, pendapat, dan preferensi dalam kelompok.

GDSS terdiri dari perangkat lunak, perangkat keras, komponen bahasa, dan proses. Proses ini mendukung sekelompok orang yang berpartisipasi dalam pertemuan yang terkait dengan pengambilan keputusan. Menurut Turban (2001), karakteristik penting yang dimiliki oleh GDSS adalah sebagai berikut:

1. Merupakan sistem informasi yang dirancang khusus secara berkelompok.

2. Proses pengambilan keputusan berkelompok meningkatkan hasil yang didapat hingga semaksimal mungkin.
3. Mudah dipelajari dan digunakan.
4. GDSS memiliki mekanisme yang sesuai untuk meminimalkan perkembangan perilaku negatif kelompok, seperti konflik yang merusak, komunikasi yang buruk, atau pemikiran yang mengganggu.

2.5 *Location Based Service* (LBS)

Location Based Service (LBS) adalah sistem yang menyebarkan suatu informasi yang dapat diakses menggunakan perangkat bergerak menggunakan internet. Cara mengaksesnya dengan menggunakan *Global Positioning System* (GPS) yang ada pada perangkat bergerak tersebut (Yuliana, 2013).

Location Based Service (LBS) diimplementasikan di platform android. Metode *Location Based Service* (LBS) merupakan salah satu metode layanan berbasis lokasi untuk mengetahui rute perjalanan yang memberikan hasil dengan tingkat keakuratan yang cukup tinggi dan dapat memberikan informasi lokasi pengguna (Nazruddin, 2011). Langkah awal yang harus dilakukan adalah mengaktifkan fungsi pencarian posisi pengguna yang dapat diperoleh dengan adanya fitur GPS. Setelah itu, perangkat seluler akan mengirimkan informasi yang berisi tujuan untuk menemukan lokasi melalui jaringan komunikasi dan mengirimkan lokasi tersebut ke *gateway*. Kemudian, layanan menganalisis pesan dan memutuskan informasi tambahan pencarian dan posisi pengguna dan selanjutnya layanan akan menemukan bahwa informasi mengenai jalan, jarak, dan memeriksa apakah tujuan dapat dicapai dan ditampilkan kepada pengguna dalam bentuk peta. Menurut Yuliana (2013), penerapan LBS memiliki dua unsur utama, yaitu:

1. Location Manager (API Maps)

   Application Programming Interface (API) menyediakan alat / sumber daya untuk LBS, menyediakan alat untuk menampilkan, memproses peta, dan fungsi lainnya (seperti tampilan satelit, jalan, dan keduanya).
2. Location Providers (API Location)

   Pengguna dapat menentukan posisinya dengan mendeteksi perpindahan, melacak pergerakan / perpindahan dan kedekatan dengan lokasi tertentu.

2.6 TOPSIS

Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) merupakan salah satu metode yang digunakan untuk mengambil keputusan berdasarkan kriteria yang telah ditentukan. Solusi ideal positif digunakan untuk menentukan jarak alternatif terdekat, dan solusi ideal negatif digunakan untuk menentukan jarak alternatif terjauh (Kurniasih, 2013).

Berikut ini merupakan langkah-langkah perhitungan dengan metode TOPSIS (Bhutia & Phipon, 2012) :

1. Membangun *normalized decision matrix*

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}} \quad (2.1)$$

Keterangan: x_{ij} → Rating Alternatif ke-I terhadap j

R_{ij} → hasil matriks ternormalisasi

Dengan i=1,2,3,…m; dan j=1,2,3 …n

2. Membangun *weighted normalized decision matrix*

Solusi ideal positif A^+ dan solusi ideal negatif A^- dapat ditentukan berdasarkan rating bobot ternormalisasi sebagai berikut:

$$v_{ij} = w_j * r_{ij} \quad (2.2)$$

Dengan i=1,2,3,…m; dan j=1,2,3 …n

3. Menentukan matriks solusi ideal positif dan matriks solusi ideal negatif

Solusi ideal positif A^+ dihitung berdasarkan:

$$A^+ = \{(\max v_{ij} \mid j \epsilon J), (\min v_{ij} \mid j \epsilon J'), i = 1,2,3, \dots, m$$
$$= \{V_1^+, V_2^+, V_3^+, \dots, V_n^+\} \quad (2.3)$$

solusi ideal negatif A^- dihitung berdasarkan:

$$A^- = \{(\min v_{ij} \mid j \epsilon J), (\max v_{ij} \mid j \epsilon J'), i = 1,2,3, \dots, m$$
$$= \{V_1^-, V_2^-, V_3^-, \dots, V_n^-\} \quad (2.4)$$

4. Menggunakan matriks solusi ideal positif dan matriks ideal negatif untuk menentukan jarak antara nilai setiap alternatif.

Jarak antara alternatif Ai dengan solusi ideal positif didefinisikan sebagai:

$$D_i^+ = \sqrt{\sum_{i=1}^{n} \left(v_{ij} - v_j^+\right)^2} \quad (2.5)$$

Jarak antara alternatif Ai dengan solusi ideal negatif didefinisikan sebagai:

$$D_i^- = \sqrt{\sum_{i=1}^{n} \left(v_{ij} - v_j^-\right)^2} \quad (2.6)$$

5. Menentukan nilai preferensi untuk setiap alternatif

$$C_i^+ = \frac{s_i^-}{s_i^- + s_i^+} \quad (2.7)$$

6. Mengurutkan nilai alternatif C+ mulai dari nilai terbesar hingga nilai terkecil. Nilai terbesar pada urutan tersebut merupakan solusi terbaik.

Nilai terbesar dalam urutan tersebut adalah solusi terbaik. Nilai prioritas masing-masing alternatif merupakan hasil akhir dari perhitungan metode TOPSIS. Semakin tinggi nilainya maka alternatif tersebut merupakan alternatif yang diinginkan.

2.7 BORDA

Metode Borda pertama kali dikemukakan oleh Jean-Charles de Borda pada abad 18. Borda merupakan metode yang digunakan untuk mencari pangkat tertinggi dengan pembobotan. Metode ini mendukung pengambilan keputusan berkelompok dengan perkalian nilai referensi dan dipertahankan dengan bobot peringkat (Saharuddin, 2012). Metode ini tidak mempertimbangkan pendapat subjektif para pengambil keputusan yang memiliki pengaruh besar dalam pengambilan keputusan tim. Penggunaan sistem pemungutan suara untuk menghitung kompleksitas pemilihan dilakukan dengan menerapkan metode Barda (Russel, 2007). Menurut penelitian Saputra (2017), metode Borda perlu melalui beberapa tahapan yaitu:

1. Nilai ranking dari alternatif tertinggi (teratas) diberi titik m, dimana m adalah jumlah dari semua alternatif.
2. Titik m digunakan sebagai pengali untuk nilai bobot yang relevan.

2.8 Android

Menurut Nazrudin Safaat H (2011), Android merupakan sistem operasi untuk perangkat bergerak berbasis Linux yang meliputi sistem operasi, middleware, dan aplikasi. Android juga memiliki sifat *opensource*, yang berarti dapat dikembangkan lebih lanjut sesuai dengan kebutuhan pengguna. Android memberi kebebasan berekspresi bagi pengembang untuk membuat aplikasi ciptaan mereka sendiri.

Saat pertama kali dirilis, Android memang diincar untuk digunakan pada perangkat kamera digital. Namun pada akhirnya, perusahaan Android memiliki sudut pandang lain, bahwa pasar penjualan kamera digital tidak akan terlalu besar. Oleh karena itu, pendiri dari Android memutuskan untuk memilih Android, dan kemudian menggunakannya untuk perangkat seluler (yaitu ponsel pintar), karena ternyata ponsel lebih diminati pada masa sekarang.

Android menjadi benar-benar bersifat terbuka terhadap aplikasi lainnya. Misalnya, aplikasi diizinkan untuk menggunakan fungsi utama ponsel, contohnya untuk mengirimkan pesan teks dan membuat panggilan dengan menggunakan kamera. Tentunya sifat terbuka ini menciptakan kemungkinan yang besar supaya pengembang membuat aplikasi yang lebih baik di masa mendatang (Hermawan & Stephanus, 2011). Namun, sistem operasi Android akan selalu mengutamakan aplikasi inti yang dibangun dengan sendirinya dengan menghiraukan potensi besar dari aplikasi pihak ketiga. Maka dari itu, wadah dan distribusi untuk penggunaan aplikasi pihak ketiga dalam mendapatkan data ponsel asli dan komunikasi antar proses dibatasi.

2.8.1 Android SDK

Android *Software Development Kit* (SDK) merupakan sebuah *Application Programming Interface* (API) yang digunakan dalam pengembangan aplikasi Android (Safaat, 2011). Safaat juga menjelaskan jika Android SDK adalah *library* yang saat ini sering digunakan dalam pengembangan aplikasi Android. Android SDK berisikan berbagai alat, seperti dokumentasi, perpustakaan perangkat lunak, debugger, contoh kode, tutorial dan emulator. Salah satu bahasa pemrograman paling sering digunakan dalam pengembangan aplikasi Android adalah Java SE Development Kit.

2.9 Firebase Realtime Database

Pada saat ini terdapat banyak basis data yang bukan merupakan tempat untuk menyimpan data di penyimpanan lokal pengguna, salah satunya adalah Firebase. Dalam beberapa tahun terakhir, Firebase mulai dikenal dan digunakan secara luas oleh para pengembang aplikasi termasuk Indonesia. Perkembangan teknologi niscaya akan membuat penggunaan basis data Firebase semakin mudah.

Firebase merupakan layanan yang disediakan oleh Google, yang memberikan kemudahan bagi para pengembang aplikasi untuk mengembangkan aplikasinya. Firebase merupakan solusi yang disediakan oleh Google agar dapat mempercepat kerja para pengembang aplikasi di bidang *back-end developer* (Firebase, 2020).

Firebase didirikan oleh Andrew Lee dan James Tamplin pada tahun 2011 (Wikipedia, 2020). Produk yang diluncurkan pertama kali adalah Firebase Realtime Database. Database real-time ini digunakan untuk menyimpan data dan menyinkronkannya ke banyak pengguna. Kemudian, Firebase semakin dikembangkan sebagai layanan bagi para pengembang aplikasi Android maupun iOS. Pada Oktober 2014, perusahaan diakuisisi oleh Google. Saat ini, selain Realtime Database, Firebase juga memiliki 5 fungsi, seperti Firebase Analytics, Firebase Cloud Messaging and Notifications, Firebase Authentication, Firebase Cloud Firestore, dan Firebase Hosting (Firebase, 2020). Dalam studi ini, peneliti hanya akan menggunakan fitur Firebase Realtime Database.

Firebase dulu menyediakan layanan uji coba kepada penggunanya, tetapi sekarang setiap pengguna dapat menggunakan dan menggunakan layanan Firebase secara gratis. Firebase memiliki batasan tertentu. Batasan tersebut terpisah dalam dua opsi, yaitu Spark, dan Blaze.

Firebase memiliki beberapa keunggulan dibanding tempat penyimpanan data yang lainnya, diantaranya adalah:

1. Firebase berjalan dengan cepat dan juga responsif.
2. Firebase bersifat No SQL karena menggunakan JSON.
3. Firebase bersifat gratis dengan waktu yang tidak terbatas.
4. Firebase dapat digunakan untuk pengembangan Android, iOS, Java, Objective-C, Swift, Node.Js, Unity, dan JavaScript.

5. Firebase memiliki dokumentasi yang lengkap jika pengguna mengalami berbagai kendala.

2.10 Bahasa Pemrograman Java

Java adalah bahasa pemrograman yang disusun oleh James Gosling, yang diprogram pada tahun 1991 dengan bantuan rekannya dari sebuah perusahaan perangkat lunak bernama Sun Microsystems (Suyanto, 2015). Awal mula bahasa pemrograman ini muncul, menggunakan nama "*Oak*" karena sebatang pohon oak yang berdiri di luar kantor Gosling, namun pada tahun 1995 diganti menjadi "Java" yang berasal dari kopi jawa, kopi yang disukai oleh Gosling. Gosling merancang java dengan sintaks gaya bahasa C/C++ yang akan dikenal oleh pemrogram sistem dan aplikasi.

Menurut situs resmi dari bahasa pemrograman java, saat ini sudah tercatat sekitar 12 juta pengembang aplikasi di dunia menggunakan bahasa pemrograman java. Adapun beberapa karakteristik Java menurut Rickyanto (2003) adalah:

1. Sederhana: Java memberikan banyak kemudahan dan keringkasan jika dibandingkan dengan sintaks dari bahasa C++.
2. Berorientasi Objek: Dalam pemrograman java, semua variabel adalah objek, terkecuali tipe primitif.
3. Dapat didistribusikan dengan mudah.
4. Portabel: Java mampu dijalankan di berbagai platform tanpa harus ada perubahan kode.
5. Multi *threading*: Java memiliki banyak kemampuan untuk menangani dan menjalankan banyak *thread* sekaligus.
6. Dinamis: Java merupakan teknologi yang terus berkembang dan hal ini tampak nyata sekali dengan *library* yang terus ditingkatkan kemampuan dan kelengkapannya.

2.11 Pengujian Perangkat Lunak

Pada pengembangan aplikasi rekomendasi lapangan tenis ini, peneliti menggunakan tiga pengujian, yakni pengujian black box, pengujian validasi algoritma dan pengujian *usability*.

2.11.1 Pengujian *Black-Box*

Pengujian *black-box* merupakan metode pengujian untuk membuktikan metodologi produk diuji sesuai dengan spesifikasi perangkat lunak atau kebutuhan yang sudah ditentukan oleh bisnis analis, sistem analis atau pengguna (Limaye, 2009). Keuntungan pengujian menggunakan metode *black-box* (Limaye, 2009):

1. Pengujian *black box* merupakan satu-satunya metode untuk membuktikan apakah perangkat lunak melakukan apa yang seharusnya dilakukan dan tidak melakukan hal yang lain yang dapat

menyebabkan masalah bagi pengguna.
2. Cara untuk membuktikan bahwa sebuah perangkat lunak tersebut berjalan dan benar-benar bekerja.
3. Beberapa pengujian dapat selesai hanya dengan pengujian *black box*, seperti kinerja sistem.

Kelemahan pengujian menggunakan metode *black-box* (Limaye, 2009):

1. Beberapa kesalahan logika dalam *coding* dapat terlewat.
2. Memungkinkan pengujian *redundant* karena kebutuhan dapat menjalankan kode yang sama terus-menerus.

Teknik yang digunakan dalam pengujian *black box* adalah dengan menentukan *test case* dan *expected result* dari masing-masing kebutuhan. *Test case* merupakan tindakan untuk menentukan apakah fitur dari sebuah sistem berjalan dengan benar, sedangkan *expected result* merupakan perkiraan hasil jika *test case* berhasil dijalankan. *Test case* yang berhasil akan berstatus *valid,* sedangkan *test case* yang tidak berhasil akan berstatus *not valid*, yang artinya ditemukan kesalahan saat program dieksekusi.

2.11.2 Pengujian Validasi Algoritme

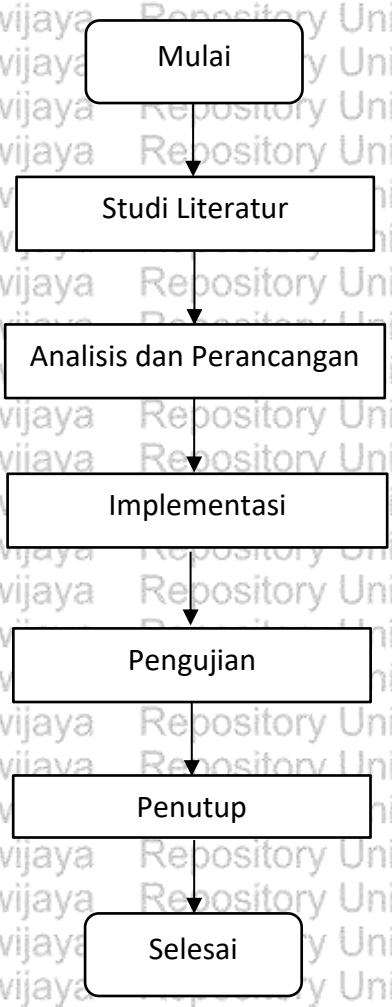
Pengujian validasi algoritme pada penelitian ini bertujuan untuk mengetahui tingkat kecocokan antara hasil perhitungan sistem dengan hasil perhitungan manual yang dilakukan oleh peneliti (Paypas dkk., 2019). Pengujian manual dilakukan menggunakan penghitungan di Microsoft Excel. Masukan ke algoritme tidak menjamin bahwa masukan lain akan memberikan hasil yang benar. Oleh karena itu, dalam kasus ini, diperlukan sebuah proses pengujian validasi algoritma untuk menjamin algoritma memenuhi spesifikasi kebutuhan yang telah ditentukan sebelumnya.

2.11.3 Pengujian Usability

Pengujian usability pada penelitian ini menggunakan *System Usability* Scale (SUS). SUS adalah salah satu cara survei untuk kebutuhan *usability* (penggunaan) dari sebuah produk atau jasa. Biasanya SUS terdiri dari hanya sepuluh pernyataan, sehingga memudahkan pengisi survei lebih paham dan untuk para pengguna untuk memberikan skor. Aspek *usability* merupakan kunci keberhasilan aplikasi dan syarat penerimaan pengguna terhadap aplikasi *mobile* (Nielsen, 1993).

BAB 3 METODOLOGI

Pada bab ini menjelaskan mengenai metode yang diterapkan dalam pengembangan aplikasi pada penelitian ini serta membahas mengenai beberapa tahapan dalam alur metodologi mulai dari studi literatur, analisis dan perancangan, implementasi, pengujian dan penutup seperti pada Gambar 3.1.



Gambar 3.1 *Flowchart* Metodologi Penelitian

3.1 Studi Literatur

Pada tahap ini menjelaskan mengenai literatur-literatur yang digunakan dalam mendukung proses analisis kebutuhan, perancangan, implementasi, dan pengujian dari aplikasi. Sumber pencarian literatur didapatkan dari beberapa pustaka sumber tertulis, baik berupa buku-buku, e-book, arsip, paper, artikel, jurnal, dan dokumen-dokumen yang relevan dengan permasalahan yang dikaji serta berbagai tutorial pemrograman yang terdapat di internet. Tujuannya adalah untuk memperkuat proses penyelesaian masalah yang ada dalam penelitian serta sebagai dasar teori dalam penulisan penelitian ini. Dasar teori yang digunakan adalah sebagai berikut:

1. Sistem Rekomendasi,
2. Sistem Pendukung Keputusan,
3. Tenis Lapangan,
4. *Group Decision Support System* (GDSS),
5. *Location Based Service* (LBS),
6. *Technique for Order of Preference by Similarity to Ideal Solution* (TOPSIS),
7. Borda,
8. Android,
9. Firebase,
10. Bahasa pemrograman Java,
11. Pengujian perangkat lunak.

3.2 Analisis dan Perancangan

Tahap ini menjelaskan proses penggalian analisis kebutuhan serta tahap perancangan. Langkah awal dalam pengembangan aplikasi yang paling utama adalah tahapan analisis kebutuhan. Ada dua poin utama dalam proses analisis kebutuhan yaitu analisis kebutuhan fungsional dan analisis kebutuhan non-fungsional. Tujuan dari analisis kebutuhan fungsional adalah untuk mendapatkan informasi yang berkaitan dengan kebutuhan, yang berisi tentang proses-proses yang akan dijalankan pada aplikasi ini. Tujuan dari analisis kebutuhan non fungsional adalah untuk mendapatkan data yang berkaitan dengan kebutuhan, termasuk atribut perilaku yang dimiliki oleh aplikasi. Hasil analisis kebutuhan fungsional kemudian digambarkan dalam pemodelan *use case* diagram.

Selanjutnya dilakukan tahap perancangan aplikasi. Perancangan aplikasi dilakukan berdasarkan data dari analisis kebutuhan di tahap sebelumnya. Pada tahap ini, peneliti mengumpulkan sumber data yang dibutuhkan dalam pengembangan aplikasi dan menyimpannya ke dalam Firebase Realtime Database. Kemudian setiap sumber data tersebut dimasukkan ke dalam perancangan aplikasi. Tahap perancangan bertujuan untuk menghasilkan pemodelan yang nanti akan mempermudah proses pengerjaan dalam tahap implementasi. Pada tahap ini pemodelan akan dibuat dalam bentuk *use case scenario, sequence diagram,* ERD, *class diagram,* perancangan algoritme, dan desain antarmuka aplikasi.

3.3 Implementasi

Pada tahap implementasi, proses pembuatan aplikasi dibuat berdasarkan hasil analisis dan perancangan sebelumnya. Tahap implementasi membutuhkan spesifikasi perangkat keras dan spesifikasi perangkat lunak dalam pengembangannya. Pada tahap implementasi digunakan beberapa bahasa pemrograman untuk mendukung proses pengembangan aplikasi android dalam penelitian ini.

Tahap ini juga membutuhkan beberapa *library* pihak ketiga untuk menunjang kebutuhan dari aplikasi. Sumber data pada penelitian ini didapatkan dari Firebase pada Langkah sebelumnya. Aplikasi dapat dijalankan hanya jika pengguna telah mengaktifkan internet pada ponselnya.

3.4 Pengujian

Tahap pengujian dilakukan untuk mengetahui jika aplikasi yang telah dikembangkan mampu berjalan secara maksimal dan tidak terdapat kesalahan pada penanganan kode implementasi. Pada penelitian ini terdapat 3 bentuk pengujian yaitu pengujian *black-box*, pengujian manual algoritma dan pengujian *usability* menggunakan SUS. Pengujian ini sebagai acuan dalam proses perbaikan aplikasi untuk menyempurnakan aplikasi yang telah dibuat.

3.5 Penutup

Tahap penutup merupakan tahap terakhir dari penelitian ini. Penutup berisikan tentang kesimpulan serta saran untuk pengembangan aplikasi di kemudian hari. Kesimpulan diambil dari hasil analisis keseluruhan dan hasil pengujian aplikasi. Pengambilan kesimpulan bertujuan untuk melaporkan hasil akhir yang didapat dalam pengembangan aplikasi dan juga sebagai acuan guna perbaikan untuk di kemudian hari.

BAB 4 ANALISIS DAN PERANCANGAN

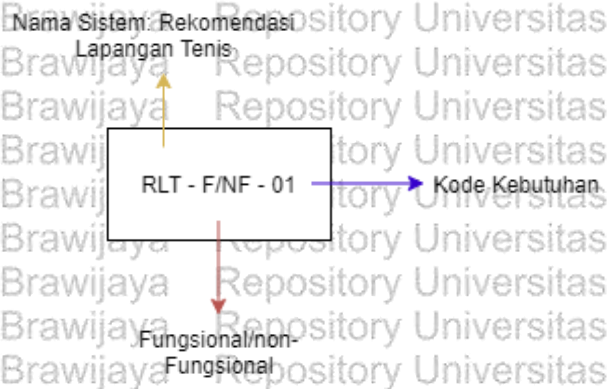
Pada bab ini menjelaskan tentang bagaimana tahap-tahap melakukan analisis dan perancangan pada aplikasi rekomendasi lapangan tenis di Malang. Tujuannya adalah untuk menggali dan mendapatkan setiap kebutuhan yang dibutuhkan dalam proses pengembangan aplikasi serta membuat perancangan dari setiap kebutuhan yang telah dijabarkan.

4.1 Data Penelitian

Pada tahap ini peneliti mengumpulkan data yang digunakan dalam penelitian. Data lapangan tenis didapatkan berdasarkan hasil wawancara yang telah dilakukan oleh Heru Budiyanto (2019) pada penelitiannya yang berjudul "Implementasi Topsis Pada Sistem Rekomendasi Pemilihan Lapangan Tenis di Malang Berbasis Lokasi". Data yang diambil dan digunakan dalam penelitian meliputi nama lapangan tenis, jarak, *rating,* harga, sifat lapangan, dan alamat lapangan. Data yang digunakan dapat dilihat pada Lampiran A Data Lapangan Tenis (Budiyanto, 2019).

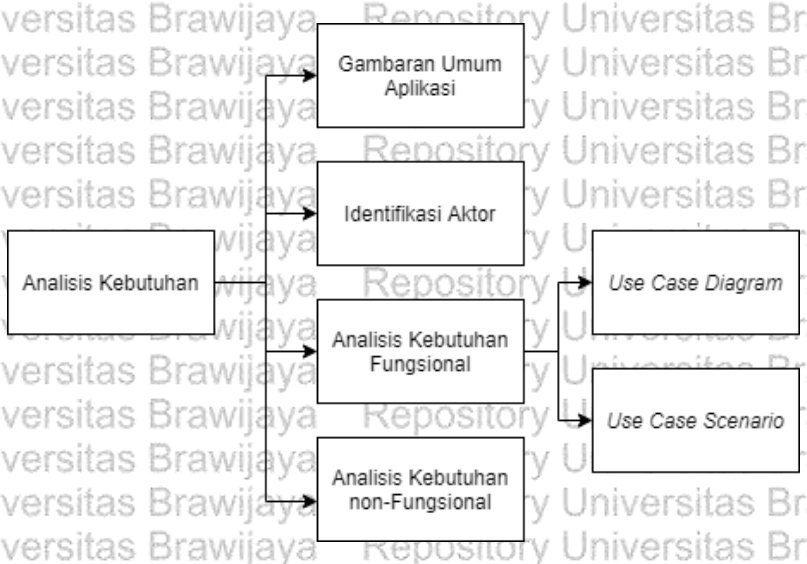
4.2 Analisis Kebutuhan

Pada tahap ini peneliti mengidentifikasi kebutuhan dalam pengembangan aplikasi rekomendasi lapangan tenis yang merupakan pengembangan dari kebutuhan pada penelitian sebelumnya (Budiyanto, 2019). Pada tahap analisis kebutuhan terdapat kode kebutuhan untuk mempermudah penjabaran kebutuhan. Penjelasan dari kode tersebut dapat dilihat pada Gambar 4.1 di bawah ini.



Gambar 4.1 Penjelasan Kode Kebutuhan

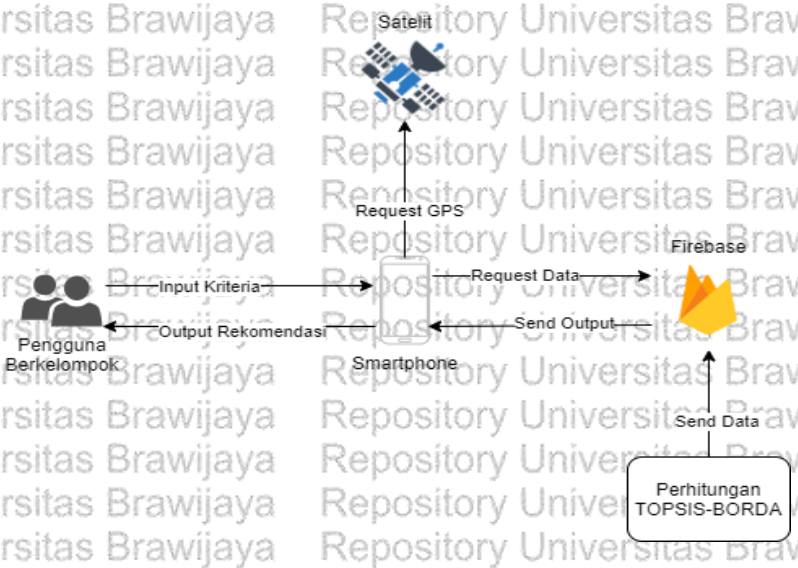
Kebutuhan fungsional merupakan pendeskripsian mengenai sistem dan fitur-fitur pada aplikasi yang dikembangkan. Pada Tabel 4.2 menjelaskan setiap kebutuhan fungsional yang terdapat pada aplikasi ini.



Gambar 4.2 Analisis Kebutuhan

4.2.1 Gambaran Umum Aplikasi

Gambaran umum aplikasi merupakan penjelasan singkat mengenai aplikasi yang dikembangkan. Tujuannya untuk mendeskripsikan bagaimana alur aplikasi dapat bekerja. Gambaran umum aplikasi diharapkan dapat membantu pembaca dalam memahami penelitian tentang pengembangan aplikasi ini.



Gambar 4.3 Gambaran Umum Aplikasi

Alur aplikasi secara garis besar dapat dilihat pada Gambar 4.3 di atas. Pengguna berkelompok dapat mengakses aplikasi dimanapun berada jika terdapat jaringan internet di wilayah tersebut. Sebelum menjalankan aplikasi, pengguna diminta untuk mengaktifkan GPS agar sistem mengetahui lokasi dari pengguna.

Pertama kali aplikasi dijalankan akan menampilkan 2 pilihan tombol yang tersedia yaitu tombol untuk melihat daftar lapangan tenis yang ada di Malang dan 1 lagi adalah tombol untuk membuat rekomendasi GDSS. Dalam rekomendasi GDSS nantinya pengguna akan diminta untuk memasukkan bobot yang tersedia seperti jarak, rating, harga, dan juga jumlah lapangan. Data dari pengguna akan dikirimkan ke firebase untuk dilakukan perhitungan berdasarkan algotirme TOPSIS-BORDA. Kemudian setelah dilakukan perhitungan, firebase akan mengembalikan hasil perhitungan ke pengguna berupa daftar rekomendasi lapangan tenis.

4.2.2 Identifikasi Aktor

Identifikasi aktor dilakukan untuk mengetahui siapa saja aktor yang terlibat dalam proses pengembangan aplikasi rekomendasi lapangan tenis. Identifikasi aktor dijelaskan dalam Tabel 4.1.

Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
Pengguna	Pengguna dapat menentukan jumlah anggota kelompok, jarak, harga, dan rating. Pengguna mendapatkan hasil rekomendasi lapangan tenis dengan tampilan daftar lokasi dari lapangan tenis yang sesuai masukan pengguna

4.2.3 Analisis Kebutuhan Fungsional

Pada tahap ini peneliti mengidentifikasi kebutuhan fungsional dalam pengembangan aplikasi rekomendasi lapangan tenis. Kebutuhan fungsional merupakan pendeskripsian mengenai sistem dan fitur-fitur pada aplikasi yang dikembangkan. Pada Tabel 4.2 menjelaskan setiap kebutuhan fungsional yang terdapat pada aplikasi ini.

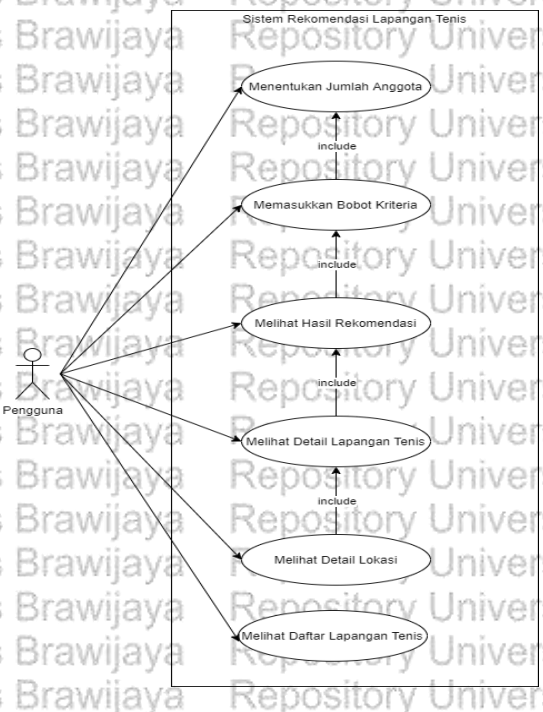
Tabel 4.2 Analisis Kebutuhan Fungsional

Kode	Kebutuhan Fungsional	Nama *Use Case*
RLT-F-01	Sistem dapat menentukan jumlah anggota	Menentukan jumlah anggota
RLT-F-02	Sistem dapat memasukkan bobot kriteria	Memasukkan bobot kriteria
RLT-F-03	Sistem dapat menampilkan daftar dari rekomendasi lapangan tenis	Melihat hasil rekomendasi
RLT-F-04	Sistem dapat menampilkan detail dari rekomendasi lapangan tenis	Melihat detail lapangan tenis
RLT-F-05	Sistem dapat menampilkan lokasi dari setiap lapangan tenis beserta	Melihat detail lokasi

	rute dari lokasi pengguna menuju lokasi lapangan tenis	
RLT-F-06	Sistem dapat menampilkan daftar lapangan tenis yang ada di Malang	Melihat daftar lapangan tenis

4.1.3.1 *Use Case Diagram*

Use Case Diagram merupakan pemodelan kebutuhan untuk mendeskripsikan sebuah interaksi antara aktor dengan sistem yang akan dibuat. *Use Case Diagram* hanya memberikan gambaran singkat dan tidak menjelaskan lebih detail tentang hubungan antara aktor dengan sistem. Pada penelitian ini, use case diagram dari kebutuhan fungsional dijabarkan seperti pada Gambar 4.4.



Gambar 4.4 Use Case Diagram Aplikasi

4.1.3.2 *Use Case Scenario*

Use Case Scenario berfungsi untuk menjabarkan setiap kebutuhan fungsional yang telah disebutkan dalam *Use Case Diagram* secara lebih detail. *Use Case Scenario* berisikan kode kebutuhan, aktor yang berhubungan dengan *usecase* tersebut, tujuan *usecase*, kondisi awal dan akhir dari sebuah *usecase*, alur utama, dan juga skenario alternatif apabila sistem tidak berjalan dengan normal atau sedang mengalami gangguan. Pada penelitian ini terdapat 6 buah *Use Case Scenario* yang telah dijelaskan pada Tabel 4.3 sampai dengan Tabel 4.8.

Tabel 4.3 Skenario Menentukan Jumlah Anggota

Nama Use Case	Menentukan jumlah anggota
Kode Kebutuhan	RLT-F-01
Tujuan	Menentukan jumlah anggota yang akan memasukkan bobot kriteria
Aktor	Pengguna
Pre-Condition	Pengguna telah membuka aplikasi rekomendasi lapangan tenis
Main Flow	1. Pengguna menekan tombol “Create GDSS” 2. Sistem menampilkan dialog masukkan jumlah pengguna 3. Pengguna menentukan jumlah anggota grup dengan menekan tombol kurang untuk mengurangi jumlah anggota dan menekan tombol tambah untuk menambah jumlah anggota 4. Pengguna menekan tombol “Konfirmasi”
Alternative Flow	Sistem menampilkan pesan bahwa jumlah pengguna kurang dari 1
Post Condition	Sistem menyimpan data jumlah anggota yang telah ditentukan dan menampilkan pesan bahwa data telah disimpan

Tabel 4.4 Skenario Memasukkan Bobot Kriteria

Nama Use Case	Memasukkan bobot kriteria
Kode Kebutuhan	RLT-F-02
Tujuan	Memasukkan bobot kriteria untuk menentukan hasil rekomendasi dengan akurat
Aktor	Pengguna
Pre-Condition	Pengguna telah menentukan jumlah anggota
Main Flow	1. Pengguna berada pada halaman Tentukan bobot kriteria 2. Pengguna mengisikan setiap bobot kriteria yang ada berulang kali berdasarkan jumlah anggota grup yang telah ditentukan sebelumnya 3. Pengguna menekan tombol “Konfirmasi”

Alternative Flow	Sistem menampilkan pesan peringatan jika ada bobot kriteria yang belum diisi
Post Condition	Sistem menyimpan setiap data bobot kriteria dari masukkan pengguna

Tabel 4.5 Skenario Melihat Hasil Rekomendasi

Nama Use Case	Melihat hasil rekomendasi
Kode Kebutuhan	RLT-F-03
Tujuan	Menampilkan daftar lapangan tenis dari hasil rekomendasi
Aktor	Pengguna
Pre-Condition	Pengguna telah menentukan bobot kriteria dengan benar
Main Flow	1. Sistem menampilkan pesan agar pengguna mengaktifkan GPS 2. Pengguna berada pada halaman daftar lapangan
Alternative Flow	Sistem menampilkan pesan peringatan jika terjadi gangguan pada koneksi internet
Post Condition	Sistem menampilkan daftar lapangan tenis sesuai hasil rekomendasi

Tabel 4.6 Skenario Melihat Detail Lapangan Tenis

Nama Use Case	Melihat detail lapangan tenis
Kode Kebutuhan	RLT-F-04
Tujuan	Menampilkan informasi dari hasil rekomendasi lapangan tenis secara lebih detail
Aktor	Pengguna
Pre-Condition	Pengguna telah melihat hasil rekomendasi lapangan tenis yang ditampilkan
Main Flow	1. Pengguna menekan salah satu *item* dari daftar rekomendasi 2. Sistem berpindah ke halaman detail lapangan
Alternative Flow	Sistem menampilkan pesan gangguan jika koneksi internet tidak stabil

Post Condition	Sistem menampilkan informasi detail terkait lapangan tenis yang telah dipilih oleh pengguna

Tabel 4.7 Skenario Melihat Detail Lokasi

Nama Use Case	Melihat detail lokasi
Kode Kebutuhan	RLT-F-05
Tujuan	Mengetahui lokasi lapangan tenis serta rute menuju lapangan tenis dari lokasi pengguna
Aktor	Pengguna
Pre-Condition	Pengguna berada pada halaman detail lapangan
Main Flow	1. Pengguna menekan tombol “Lihat Rute” 2. Sistem mengalihkan ke aplikasi Google Maps untuk melihat lokasi beserta rutenya
Alternative Flow	Sistem menampilkan pesan gangguan jika koneksi internet tidak stabil
Post Condition	Sistem menampilkan detail lokasi yang dipilih pengguna beserta rute menuju lapangan tenis dari lokasi pengguna

Tabel 4.8 Skenario Melihat Daftar Lapangan Tenis

Nama Use Case	Melihat daftar lapangan tenis
Kode Kebutuhan	RLT-F-06
Tujuan	Menampilkan keseluruhan daftar lapangan tenis
Aktor	Pengguna
Pre-Condition	Pengguna telah membuka aplikasi rekomendasi lapangan tenis
Main Flow	1. Pengguna berada pada halaman awal aplikasi 2. Pengguna menekan tombol “List Lapangan” 3. Sistem berpindah ke halaman daftar lapangan tenis di Malang
Alternative Flow	-
Post Condition	Sistem menampilkan keseluruhan daftar lapangan tenis yang ada di Malang

4.2.4 Analisis Kebutuhan non-Fungsional

Pengujian black box merupakan metode pengujian untuk membuktikan metodologi produk diuji sesuai dengan spesifikasi perangkat lunak atau kebutuhan yang sudah ditentukan oleh bisnis analis, sistem analis atau pengguna (Limaye, 2009).

Tabel 4.9 Kebutuhan Non-Fungsional

Kode	Kebutuhan Non-Fungsional	Nama Kebutuhan
RLT-NF-01	Menentukan kecocokan perhitungan yang dilakukan manual oleh peneliti dengan algoritma dari aplikasi	Validasi algoritma
RLT-NF-02	Mengevaluasi pengalaman pengguna dalam menggunakan aplikasi	*Usability*

4.3 Perancangan

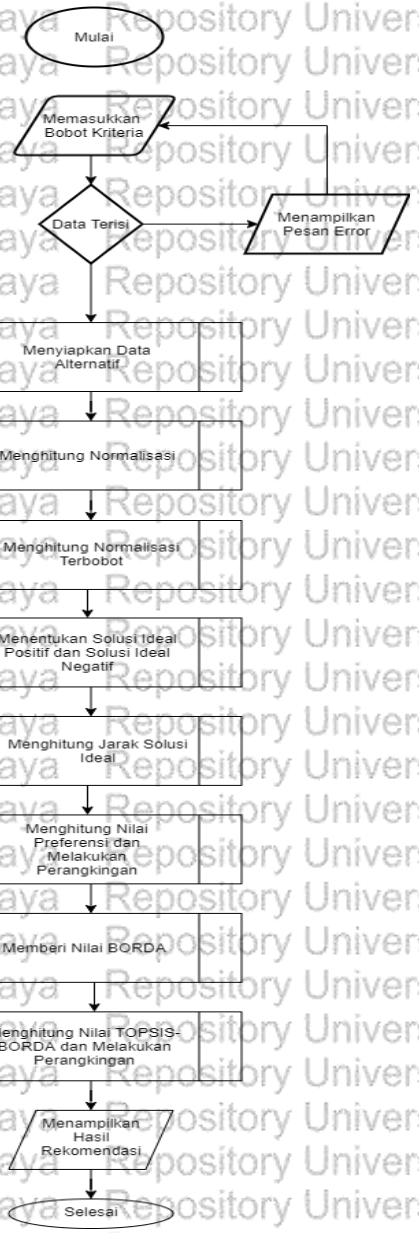
Setelah semua analisis kebutuhan telah dikumpulkan, maka tahap selanjutnya adalah melakukan sebuah perancangan. Pada tahap perancangan menjelaskan tahap awal dari proses pengembangan aplikasi rekomendasi lapangan tenis. Langkah-langkah pada tahap perancangan akan dijelaskan pada sub bab berikutnya.

4.3.1 Perancangan Algoritme

Perancangan algoritme ditujukan untuk menggambarkan algoritme perhitungan yang digunakan pada rekomendasi lapangan tenis di Malang. Langkah yang dilakukan pada bagian ini adalah pembuatan skema penggunaan algoritme TOPSIS-Borda dan pemaparan hitung secara manual.

4.3.1.1 Perancangan TOPSIS-Borda

Perancangan ini berfungsi sebagai bobot data yang akan dikalkulasi melalui masukkan yang diberikan oleh pengguna. Proses perhitungan metode TOPSIS-Borda memiliki beberapa tahapan. Peneliti telah merancang sebuah *flow chart* untuk perhitungan tersebut. *Flow chart* dapat dilihat pada Gambar 4.5.



Gambar 4.5 Flowchart Perhitungan TOPSIS-BORDA

4.3.1.2 Perhitungan Manual

Tahap ini menjelaskan setiap tahapan untuk melakukan perhitungan manual dengan penggabungan TOPSIS dan Borda. Setiap persamaan yang dibutuhkan telah dijelaskan dalam bab 2, sehingga dalam tahap ini peneliti tidak akan menulis ulang setiap persamaan yang dibutuhkan dan hanya melakukan perhitungan saja. Langkah perhitungan manual diawali menggunakan metode TOPSIS. Hasil dari setiap tahap dapat dilihat pada pembahasan berikut.

A. Menyiapkan data alternatif yang digunakan

Langkah pertama adalah menyiapkan data alternatif dan bobot kriteria. Setiap bobot kriteria yang ada dibuat berdasarkan data penelitian

sebelumnya. Terdapat 2 bobot yang ditampilkan pada Tabel 4.10 dan Tabel 4.11 yang dibuat sebagai contoh. Kemudian pada Tabel 4.12 diberikan 5 sampel data alternatif dengan kriteria jarak, harga, rating, dan jumlah lapangan.

Tabel 4.10 Kriteria Bobot Pertama

Kriteria	Bobot
Harga	1
Jarak	1
Rating	1
Jumlah Lapangan	1

Tabel 4.11 Kriteria Bobot Kedua

Kriteria	Bobot
Harga	2
Jarak	3
Rating	3
Jumlah Lapangan	2

Tabel 4.12 Sampel Data Lapangan Tenis

No	Nama Lapangan Tenis	Harga per Jam	Jarak	Rating	Jumlah Lapangan
1	Lapangan Tenis PELTI	15000	2.5495	5	4
2	Lapangan Tenis Istana Dieng Club House	75000	3.1252	4.3	2
3	Lapangan Tenis Cakrawala UM	20000	2.0374	5	2
4	Lapangan Tenis Wijaya Kusuma	20000	3.3477	4.8	2
5	Lapangan Tenis Araya	30000	7.0259	4.3	2

B. Menghitung normalisasi

Pada tahap ini, peneliti melakukan kalkulasi dari setiap bobot kriteria yang dipangkatkan, lalu hasilnya akan dibagi melalui akar kuadrat. Berikut contoh penghitungan nilai pembagi pada persamaan pada kolom harga:

$$\sqrt{15000^2 + 75000^2 + 20000^2 + 20000^2 + 30000^2}$$
$$= 86890{,}736$$

Sehingga hasil dari keseluruhan perhitungan awal dapat ditunjukkan pada Tabel 4.13.

Tabel 4.13 Hasil Penjumlahan Awal Tiap Kriteria

No	Nama Lapangan Tenis	Harga per Jam	Jarak	Rating	Jumlah Lapangan
1	Lapangan Tenis PELTI	15000	2.5495	5	4
2	Lapangan Tenis Istana Dieng Club House	75000	3.1252	4.3	2
3	Lapangan Tenis Cakrawala UM	20000	2.0374	5	2
4	Lapangan Tenis Wijaya Kusuma	20000	3.3477	4.8	2
5	Lapangan Tenis Araya	30000	7.0259	4.3	2
		86890.736	8.9993	10.4890	5.6568

Setelah itu, hasil kalkulasi digunakan sebagai nilai pembagi. Contoh normalisasi menggunakan nilai pada kolom harga baris pertama:

$$\frac{15000}{86890{,}736} = 0.1726$$

Maka didapat keseluruhan hasil normalisasi dari setiap perhitungan bobot kriteria yang dijelaskan pada Tabel 4.14.

Tabel 4.14 Hasil Normalisasi Alternatif

No	Nama Lapangan Tenis	Harga per Jam	Jarak	Rating	Jumlah Lapangan
1	Lapangan Tenis PELTI	0.1726	0.2832	0.4767	0.7071
2	Lapangan Tenis Istana Dieng Club House	0.8631	0.3473	0.4099	0.3535
3	Lapangan Tenis Cakrawala UM	0.2301	0.2263	0.4766	0.3535
4	Lapangan Tenis Wijaya Kusuma	0.2301	0.3720	0.4577	0.3535
5	Lapangan Tenis Araya	0.3452	0.7807	0.4099	0.3535

C. Menghitung normalisasi terbobot

Pada tahap ini peneliti melakukan perkalian nilai bobot kriteria dengan hasil normalisasi matriks yang telah dihitung sebelumnya. Pembobotan dilakukan dengan kriteria 2 pengguna karena penelitian ini berfokus untuk menentukan rekomendasi berdasarkan masukkan pengguna berkelompok. Contoh perhitungan normalisasi terbobot pengguna pertama menggunakan kolom harga baris pertama:

$$0.1726 * 1 = 0.1726$$

Dan perhitungan normalisasi terbobot pengguna kedua menggunakan kolom harga baris pertama:

$$0.1726 * 2 = 0.3452$$

Karena ada perbedaan bobot, maka terdapat perbedaan hasil antara pengguna pertama dengan pengguna kedua. Hasil normalisasi terbobot dari perhitungan diatas dapat dilihat pada Tabel 4.15 dan Tabel 4.16.

Tabel 4.15 Hasil Normalisasi Terbobot Pengguna Pertama

No	Nama Lapangan Tenis	Harga per Jam	Jarak	Rating	Jumlah Lapangan
1	Lapangan Tenis PELTI	0.1726	0.2832	0.4767	0.7071
2	Lapangan Tenis Istana Dieng Club House	0.8631	0.3473	0.4099	0.3535
3	Lapangan Tenis Cakrawala UM	0.2301	0.2263	0.4766	0.3535
4	Lapangan Tenis Wijaya Kusuma	0.2301	0.3720	0.4577	0.3535

5	Lapangan Tenis Araya	0.3452	0.7807	0.4099	0.3535

Tabel 4.16 Hasil Normalisasi Terbobot Pengguna Kedua

No	Nama Lapangan Tenis	Harga per Jam	Jarak	Rating	Jumlah Lapangan
1	Lapangan Tenis PELTI	0.3452	0.8496	1.4301	1.4142
2	Lapangan Tenis Istana Dieng Club House	1.7262	1.0419	1.2297	0.7071
3	Lapangan Tenis Cakrawala UM	0.4602	0.6789	1.4298	0.7071
4	Lapangan Tenis Wijaya Kusuma	0.4602	1.116	1.3731	0.7071
5	Lapangan Tenis Araya	0.6904	2.3421	1.2297	0.7071

D. Menentukan solusi ideal positif dan ideal negatif

Tahap berikutnya adalah menentukan nilai solusi ideal positif yang dilambangkan dengan A+ dan solusi ideal negatif yang dilambangkan dengan A-. Nilai-nilai tersebut didapat dari perhitungan yang sudah dilakukan sebelumnya. Solusi ideal positif adalah jika nilai kriteria semakin tinggi maka akan semakin menguntungkan sedangkan solusi ideal negatif adalah jika semakin nilai kriteria rendah maka akan semakin menguntungkan.

Pada Tabel 4.17 dijelaskan solusi ideal positif dan negatif dari setiap bobot yang ada untuk pengguna pertama, sedangkan pada Tabel 4.18 untuk penjelasan dari pengguna kedua.

Tabel 4.17 Solusi Ideal Positif dan Negatif Pengguna Pertama

Solusi	Harga per Jam	Jarak	Rating	Jumlah Lapangan
A+	0.8631	0.7807	0.4767	0.7071
A-	0.1726	0.2263	0.4099	0.3535

Tabel 4.18 Solusi Ideal Positif dan Negatif Pengguna Kedua

Solusi	Harga per Jam	Jarak	Rating	Jumlah Lapangan
A+	1.7262	2.3421	1.4301	1.4142
A-	0.3452	0.6789	1.2297	0.7071

E. Menghitung jarak solusi ideal

Pada tahap ini dilakukan pencarian nilai jarak solusi ideal untuk mencari preferensi di tahap berikutnya. Di bawah ini merupakan contoh perhitungan jarak solusi ideal menggunakan Lapangan Tenis Istana Dieng Club House pada pengguna pertama:

$$D^{+} = \sqrt{\begin{matrix}(0.8631 - 0.1726)^2 + (0.7807 - 0.2832)^2 + \\ (0.4767 - 0.4767)^2 + (0.7071 - 0.7071)^2\end{matrix}}$$
$$= 0.8510$$

$$D^{-} = \sqrt{\begin{matrix}(0.1726 - 0.1726)^2 + (0.2263 - 0.2832)^2 + \\ (0.4099 - 0.4767)^2 + (0.3535 - 0.7071)^2\end{matrix}}$$
$$= 0.3643$$

Hasil perhitungan dari seluruh keseluruhan data tentang jarak solusi ideal pada pengguna pertama ditunjukkan pada Tabel 4.19, sedangkan untuk pengguna kedua ditunjukkan pada Tabel 4.20.

Tabel 4.19 Perhitungan Jarak Solusi Ideal Pengguna Pertama

Nama Lapangan Tenis	D+	D-
Lapangan Tenis PELTI	0.8510	0.3643
Lapangan Tenis Istana Dieng Club House	0.5633	0.7010
Lapangan Tenis Cakrawala UM	0.9127	0.088
Lapangan Tenis Wijaya Kusuma	0.8325	0.1637
Lapangan Tenis Araya	0.6306	0.5806

Tabel 4.20 Perhitungan Jarak Solusi Ideal Pengguna Kedua

Nama Lapangan Tenis	D+	D-
Lapangan Tenis PELTI	2.0334	0.7546
Lapangan Tenis Istana Dieng Club House	1.4935	1.4279
Lapangan Tenis Cakrawala UM	2.2066	0.2307
Lapangan Tenis Wijaya Kusuma	1.8998	0.4741

Lapangan Tenis Araya	1.2701	1.6986

F. Menghitung nilai preferensi dan melakukan perangkingan

Tahap selanjutnya yaitu menghitung nilai preferensi untuk menentukan *ranking* dari masing-masing alternatif. Salah satu contoh perhitungan di bawah ini menggunakan baris Lapangan Tenis Istana Dieng Club House oleh pengguna pertama:

$$\frac{0.7010}{0.7010 + 0.5633} = 0.5544$$

Maka didapat hasil keseluruhan nilai preferensi beserta perangkingannya dalam Tabel 4.21 dan Tabel 4.22.

Tabel 4.21 Nilai Preferensi dan Perangkingan Pengguna Pertama

Nama Lapangan Tenis	Ci+	Ranking
Lapangan Tenis PELTI	0.2997	3
Lapangan Tenis Istana Dieng Club House	0.5544	1
Lapangan Tenis Cakrawala UM	0.0879	5
Lapangan Tenis Wijaya Kusuma	0.1643	4
Lapangan Tenis Araya	0.4793	2

Tabel 4.22 Nilai Preferensi dan Perangkingan Pengguna Kedua

Nama Lapangan Tenis	Ci+	Ranking
Lapangan Tenis PELTI	0.2706	3
Lapangan Tenis Istana Dieng Club House	0.4887	2
Lapangan Tenis Cakrawala UM	0.0946	5
Lapangan Tenis Wijaya Kusuma	0.1997	4
Lapangan Tenis Araya	0.5721	1

G. Memberi nilai BORDA

Pada tahap ini, nilai BORDA diberikan sesuai dengan perangkingan yang telah dilakukan di tahap sebelumnya. Nilai BORDA dijelaskan pada Tabel 4.23 dan Tabel 4.24.

Tabel 4.23 Nilai BORDA Pengguna Pertama

Nama Lapangan Tenis	Ranking	Nilai BORDA
Lapangan Tenis PELTI	3	3
Lapangan Tenis Istana Dieng Club House	1	5
Lapangan Tenis Cakrawala UM	5	1
Lapangan Tenis Wijaya Kusuma	4	2
Lapangan Tenis Araya	2	4

Tabel 4.24 Nilai BORDA Pengguna Kedua

Nama Lapangan Tenis	Ranking	Nilai BORDA
Lapangan Tenis PELTI	3	3
Lapangan Tenis Istana Dieng Club House	2	4
Lapangan Tenis Cakrawala UM	5	1
Lapangan Tenis Wijaya Kusuma	4	2
Lapangan Tenis Araya	1	5

H. Menghitung nilai TOPSIS-BORDA dan melakukan perangkingan

Pada tahap terakhir, peneliti melakukan pencarian nilai TOPSIS-BORDA pada setiap alternatif. Nilai TOPSIS-BORDA kedua pengguna akan dijumlah karena rekomendasi yang dicari merupakan hasil masukkan lebih dari satu pengguna. Langkah pertama yaitu melakukan perkalian nilai preferensi dengan nilai BORDA. Hasil keseluruhan perhitungan dapat dilihat pada Tabel 4.25 dan Tabel 4.26.

Tabel 4.25 Nilai TOPSIS-BORDA Pengguna Pertama

Nama Lapangan Tenis	Ci+	Nilai BORDA	TOPSIS-BORDA
Lapangan Tenis PELTI	0.2997	3	0.8991
Lapangan Tenis Istana Dieng Club House	0.5544	5	2.772
Lapangan Tenis Cakrawala UM	0.0879	1	0.0879
Lapangan Tenis Wijaya Kusuma	0.1643	2	03286
Lapangan Tenis Araya	0.4793	4	1.9172

Tabel 4.26 Nilai TOPSIS-BORDA Pengguna Kedua

Nama Lapangan Tenis	Ci+	Nilai BORDA	TOPSIS-BORDA
Lapangan Tenis PELTI	0.2706	3	0.8118
Lapangan Tenis Istana Dieng Club House	0.4887	4	1.9548
Lapangan Tenis Cakrawala UM	0.0946	1	0.0946
Lapangan Tenis Wijaya Kusuma	0.1997	2	0.3994
Lapangan Tenis Araya	0.5721	5	2.8605

Langkah selanjutnya yaitu melakukan perhitungan antara hasil jumlah TOPSIS-BORDA dari pengguna pertama dan kedua lalu diberikan perangkingan untuk setiap alternatif. Hasil penjumlahan dapat dilihat pada Tabel 4.27.

Tabel 4.27 Hasil Perangkingan Nilai TOPSIS-BORDA

Nama Lapangan Tenis	Jumlah TOPSIS-BORDA	Ranking
Lapangan Tenis PELTI	1.7109	3
Lapangan Tenis Istana Dieng Club House	4.7268	2

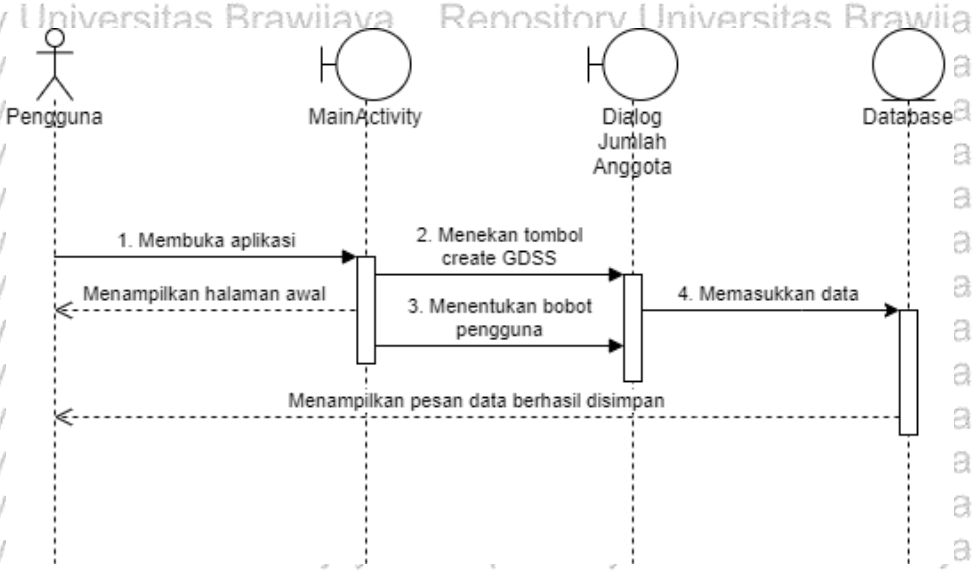
Lapangan Tenis Cakrawala UM	0.1825	5
Lapangan Tenis Wijaya Kusuma	0.728	4
Lapangan Tenis Araya	4.7777	1

Berdasarkan hasil akhir dari Tabel 4.27 maka dapat disimpulkan bahwa nilai terbesar dimiliki oleh Lapangan Tenis Araya sehingga mendapatkan peringkat pertama dalam rekomendasi.

4.3.2 Perancangan Sequence Diagram

Pada tahap selanjutnya adalah perancangan *sequence diagram*. Dalam tahap ini, *sequence diagram* bekerja untuk menjelaskan perilaku pada sebuah skenario dan menggambarkan interaksi antara pengguna dengan objek, termasuk pesan yang dipakai saat interaksi.

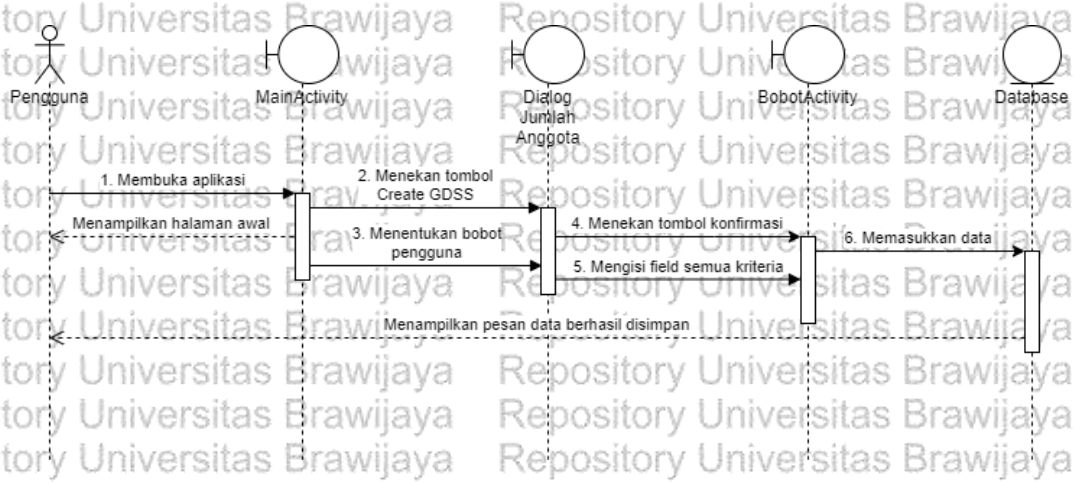
Sequence diagram sangat berkaitan erat dengan *use case diagram*, karena 1 *use case* dapat diwakilkan oleh 1 *sequence diagram*. Maka dari itu, di penelitian ini akan terdapat 6 gambar *sequence diagram* yang dapat dilihat pada Gambar 4.6 hingga Gambar 4.11.



Gambar 4.6 Sequence Diagram Menentukan Jumlah Anggota

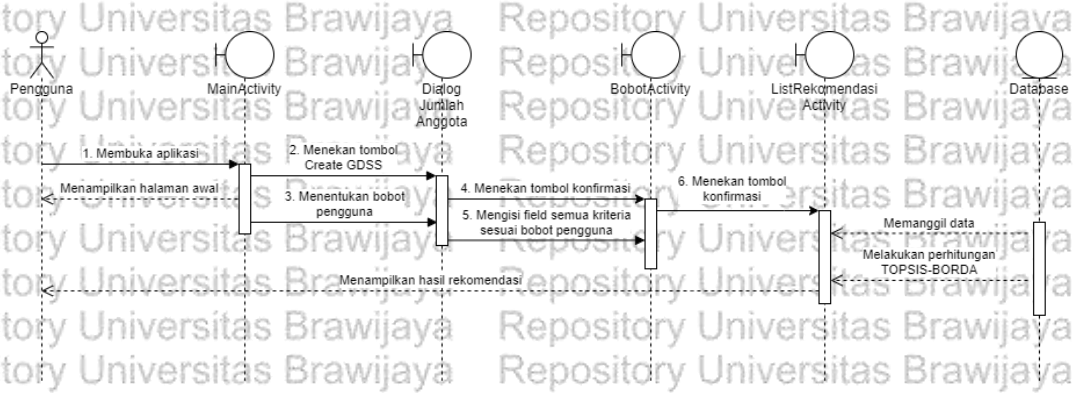
Pada Gambar 4.6 menjelaskan tentang *sequence diagram* menentukan jumlah anggota. Pengguna harus membuka aplikasi terlebih dahulu. Kemudian pengguna menekan tombol Create GDSS di halaman awal aplikasi dan akan memanggil dialog interaktif untuk menentukan jumlah anggota. Setelah jumlah anggota telah ditentukan, pengguna harus menekan tombol konfirmasi untuk ke halaman selanjutnya dan akan muncul pesan "data berhasil disimpan" yang

menandakan bahwa jumlah anggota dari pengguna sudah tersimpan di dalam *database*.



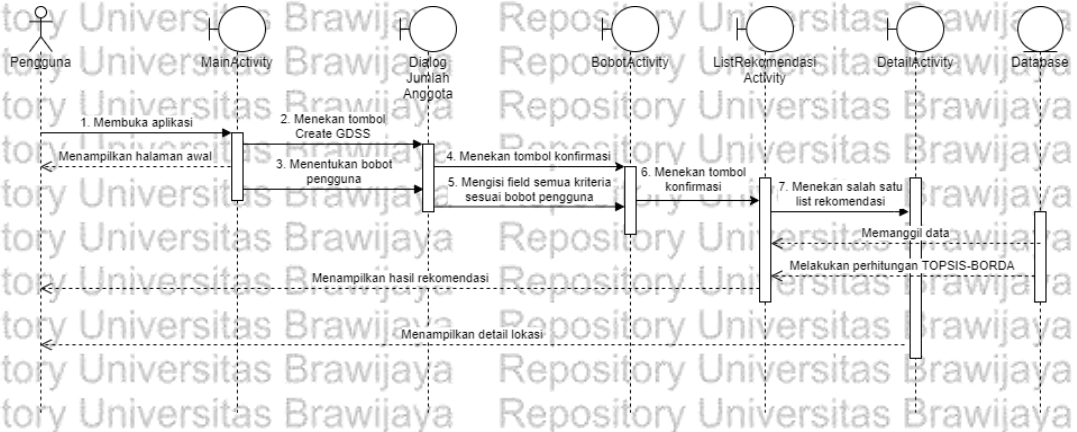
Gambar 4.7 Sequence Diagram Memasukkan Bobot Kriteria

Gambar 4.7 menjelaskan tentang *sequence diagram* memasukkan bobot kriteria. Setelah pengguna berhasil menyimpan data jumlah anggota, maka selanjutnya pengguna masuk ke halaman bobot kriteria. Pada halaman ini, pengguna harus mengisikan setiap bobot kriteria yang ada. Halaman bobot kriteria akan terus berulang sebanyak masukkan jumlah anggota dari pengguna di halaman sebelumnya. Setelah setiap bobot kriteria telah terisi, maka pengguna harus menekan tombol konfirmasi untuk ke halaman selanjutnya dan akan muncul pesan "data berhasil disimpan" yang menandakan setiap bobot kriteria setiap jumlah anggota dari pengguna sudah tersimpan di dalam *database*.



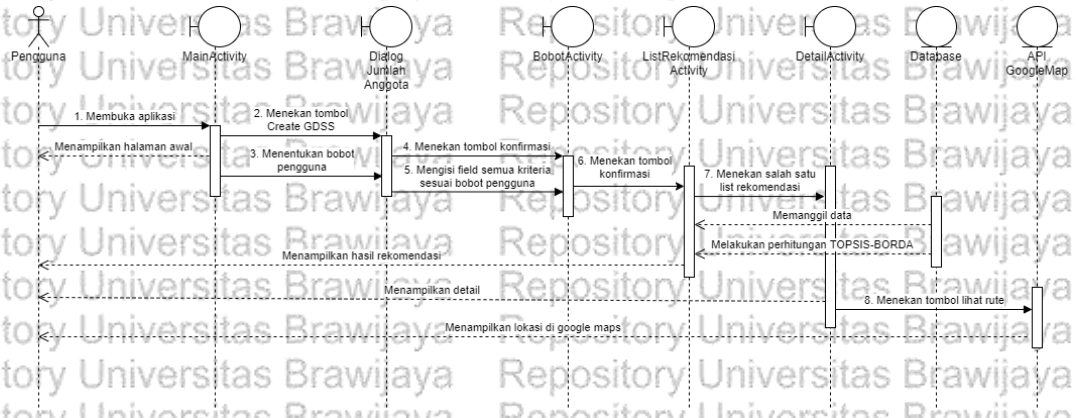
Gambar 4.8 Sequence Diagram Melihat Hasil Rekomendasi

Gambar 4.8 menjelaskan tentang *sequence diagram* melihat hasil rekomendasi. Setelah pengguna berhasil menyimpan data jumlah anggota beserta bobot kriterianya, maka selanjutnya pengguna masuk ke halaman list rekomendasi lapangan tenis dan memanggil *ListRekomendasiActivity*. Pada kelas ini, daftar rekomendasi lapangan telah dibuat berdasarkan data yang diambil dalam *database* dan melakukan perhitungan TOPSIS-BORDA pada aplikasi untuk meningkatkan akurasi dari hasil rekomendasi.



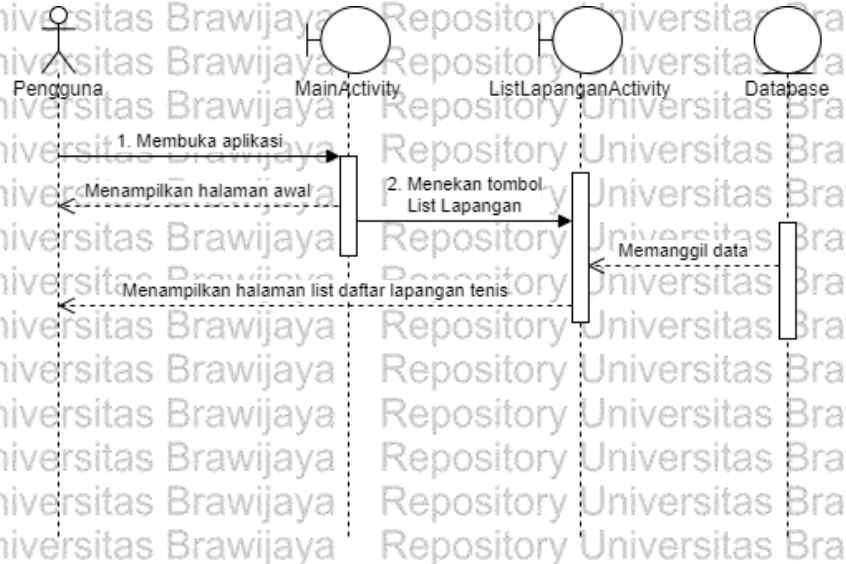
Gambar 4.9 Sequence Diagram Melihat Detail Lapangan Tenis

Gambar 4.9 menjelaskan tentang *sequence diagram* melihat detail lapangan tenis. Saat ini pengguna sedang berada dalam halaman list rekomendasi lapangan tenis. Pengguna memilih salah satu dari daftar yang ada, lalu aplikasi akan memanggil kelas *DetailActivity*. Kelas ini berfungsi untuk menampilkan keseluruhan informasi tentang lapangan tenis yang dipilih.



Gambar 4.10 Sequence Diagram Melihat Detail Lokasi

Gambar 4.10 menjelaskan tentang *sequence diagram* melihat detail lokasi. Saat ini pengguna sedang berada dalam halaman detail lapangan tenis. Pengguna menekan tombol lihat rute. Kemudian aplikasi akan memanggil google maps dan menunjukkan lokasi dari lapangan tenis yang sedang dilihat oleh pengguna.



Gambar 4.11 Sequence Diagram Melihat Daftar Lapangan Tenis

Pada Gambar 4.11 menjelaskan tentang *sequence diagram* melihat daftar lapangan tenis. Pengguna harus membuka aplikasi terlebih dahulu dan mengaktifkan internet. Kemudian pengguna menekan tombol lihat lapangan tenis di halaman awal aplikasi dan akan memanggil *ListLapanganActivity*. Kelas ini digunakan untuk menampilkan daftar lapangan tenis yang ada di Malang berdasarkan *database* awal yang telah dibuat oleh peneliti dari hasil analisis kebutuhan di sub bab 4.1.

4.3.3 Perancangan Class Diagram

Pada tahap ini, peneliti mendeskripsikan setiap kelas, atribut, method, dan objek yang digunakan akan digunakan dalam proses implementasi di bab selanjutnya. Manfaat dari perancangan *class diagram* ini adalah untuk mempermudah melihat setiap kebutuhan spesifik terkait aplikasi yang dikembangkan. Proses perancangan *class diagram* dapat dilihat pada gambar di Lampiran B. Dalam gambar tersebut terdapat kelas *activity, adapter,* dan perhitungan TOPSIS-BORDA yang saling terkait satu sama lain.

4.3.4 Perancangan Basis Data

Pada tahap ini digunakan untuk mendeskripsikan setiap atribut yang digunakan dalam pengembangan aplikasi. Setiap atribut dimasukkan ke dalam tabel yang ada di dalam basis data. Basis data yang digunakan adalah Firebase yang mendukung format JSON. Firebase sangat berguna bagi pengembangan aplikasi android dikarenakan ringan digunakan. Gambaran basis data terdapat pada Tabel 4.28.

Tabel 4.28 Skema JSON

No	Skema JSON
1	{
2	"pengguna": {

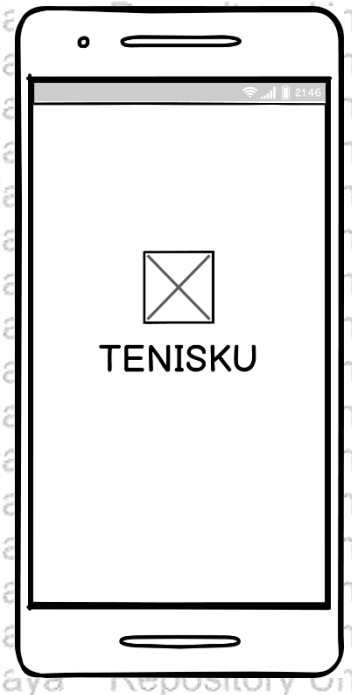
```
3           "1": {
4               "alamat": string,
5               "foto": string,
6               "harga": integer,
7               "jumlahlapangan": integer,
8               "lat": double,
9               "lng": double,
10              "nama": string,
11              "rating": double,
12              "sifat": string,
13              "telepon": string,
14              "waktubuka": string
15          }
16      }
17  }
```

4.3.5 Perancangan Antarmuka

Tahap terakhir dalam sub bab perancangan adalah merancang navigasi dan antarmuka. Perancangan dilakukan sebagai gambaran untuk implementasi dari aplikasi yang dikembangkan. Setiap antarmuka yang telah dibuat oleh peneliti diharapkan dapat digunakan untuk oleh setiap pengguna ponsel pintar android.

4.2.6.1 Halaman Splash Screen

Awal dari setiap aplikasi android akan selalu ada halaman splash screen. Halaman ini berisi nama dan logo dari aplikasi. Halaman ini akan ditampilkan jika aplikasi pertama kali dibuka oleh pengguna. Rancangan dari halaman splash screen dapat dilihat pada Gambar 4.12.



Gambar 4.12 Rancangan Halaman Splash Screen

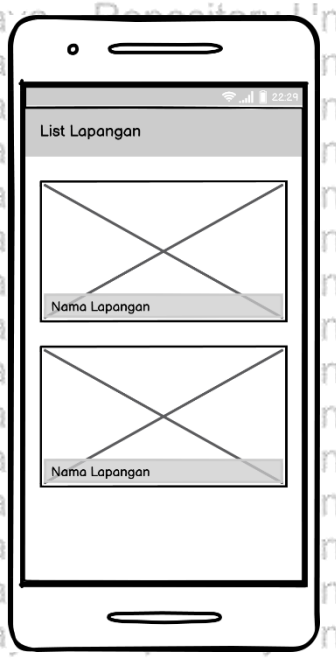
4.2.6.2 Halaman Home

Halaman home merupakan halaman utama dari aplikasi ini. Halaman ini ditampilkan setelah halaman splash screen. Pada halaman ini berisikan dua tombol yang nantinya akan menavigasikan ke halaman lain. Rancangan dari halaman home dapat dilihat pada Gambar 4.13.

Gambar 4.13 Rancangan Halaman Home

4.2.6.3 Halaman Daftar Lapangan Tenis di Malang

Halaman ini ditampilkan saat pengguna telah menekan tombol "List Lapangan" di halaman home. Pada halaman ini berisikan daftar lapangan tenis yang ada di Malang, namun tidak menjelaskan detail dari lapangan tenis tersebut. Rancangan dari halaman ini dapat dilihat pada Gambar 4.14.



Gambar 4.14 Rancangan Halaman Daftar Lapangan

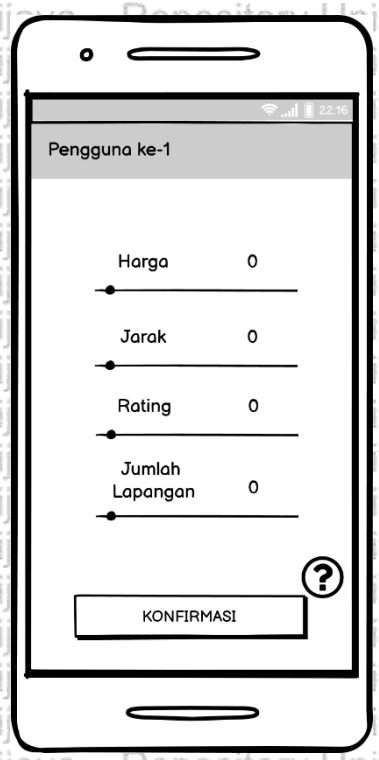
4.2.6.4 Halaman Tentukan Jumlah Anggota

Halaman ini ditampilkan saat pengguna telah menekan tombol "Create GDSS" di halaman home. Pada halaman ini pengguna diminta untuk menentukan jumlah anggota. Rancangan dari halaman ini dapat dilihat pada Gambar 4.15.

Gambar 4.15 Rancangan Halaman Tentukan Jumlah Anggota

4.2.6.5 Halaman Tentukan Bobot Kriteria

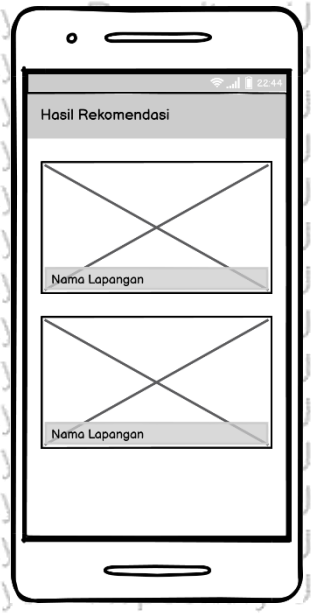
Halaman ini ditampilkan saat pengguna telah menentukan jumlah anggota. Pada halaman ini pengguna diminta untuk mengisikan setiap kriteria yang ada dengan bobot sebanyak jumlah anggota yang telah dimasukkan di halaman sebelumnya. Rancangan dari halaman ini dapat dilihat pada Gambar 4.16.



Gambar 4.16 Rancangan Halaman Tentukan Bobot

4.2.6.6 Halaman Hasil Rekomendasi

Halaman ini merupakan hasil dari keseluruhan data yang telah dimasukkan pengguna dan telah dihitung oleh sistem menggunakan metode TOPSIS-BORDA. Halaman ini berisikan daftar lapangan tenis yang sesuai dengan kriteria yang telah ditentukan oleh pengguna. Rancangan dari halaman rekomendasi dapat dilihat pada Gambar 4.17.



Gambar 4.17 Rancangan Halaman Hasil Rekomendasi

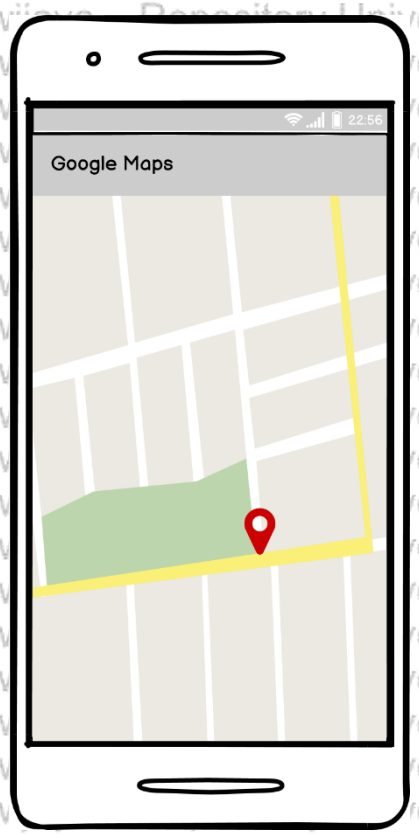
4.2.6.7 Halaman Detail Lapangan

Halaman detail lapangan ditampilkan saat pengguna menekan salah satu *item* yang ada pada daftar lapangan di halaman rekomendasi. Halaman ini berisi setiap rincian dari lapangan tenis, mulai dari harga, rating, jam operasional, hingga rute untuk sampai ke lapangan tersebut. Rancangan dari halaman ini dapat dilihat pada Gambar 4.18.

Gambar 4.18 Rancangan Halaman Detail Lapangan

4.2.6.8 Halaman Rute Google Maps

Halaman rute google maps akan terbuka apabila pengguna menekan tombol "lihat rute" pada halaman detail lapangan. Pada halaman ini akan menampilkan rute dari tempat pengguna saat ini menuju lokasi lapangan. Rancangan dari halaman ini dapat dilihat pada Gambar 4.19.



Gambar 4.19 Rancangan Halaman Rute Google Maps

BAB 5 IMPLEMENTASI

Pada bab ini menjelaskan tentang bagaimana tahap-tahap melakukan proses implementasi terhadap sistem berdasarkan hasil dari analisis kebutuhan dan perancangan yang telah dijelaskan pada bab sebelumnya.

5.1 Spesifikasi Sistem

Pada tahap ini membahas tentang perangkat yang digunakan dalam pengembangan sistem rekomendasi lapangan tenis di Malang. Tahap ini menjelaskan tentang spesifikasi dari perangkat tersebut yang dipecah menjadi spesifikasi dari perangkat keras dan perangkat lunak.

5.1.1 Spesifikasi Perangkat Keras

Perangkat keras yang digunakan dalam proses pengembangan aplikasi ini terbagi atas 2 bentuk, yakni menggunakan laptop dan peranti bergerak. Spesifikasi dari keduanya dapat dilihat pada Tabel 5.1 dan Tabel 5.2 di bawah ini.

Tabel 5.1 Spesifikasi Perangkat Keras Laptop

Komponen	Keterangan
Prosesor	AMD A8-610 APU
Memori	8192 MB
Kartu Grafis	AMD Radeon R5 Graphics 2.00 GHz

Tabel 5.2 Spesifikasi Perangkat Keras Peranti Bergerak

Komponen	Keterangan
Prosesor	Qualcomm Snapdragon 636
Memori	3.00 GB

5.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak dari pengembangan aplikasi ini dapat dilihat pada Tabel 5.3 di bawah ini.

Tabel 5.3 Spesifikasi Perangkat Lunak

Komponen	Keterangan
Sistem Operasi	Windows 10 Pro 64-bit
Alat Pengembangan	Android Studio 4.1.1
Basis Data	Google Firebase

Bahasa Pemrograman	Java
Platform Pengembangan	Android API 28

5.2 Batasan Implementasi

Pada tahap ini membahas mengenai Batasan implementasi dari pengembangan aplikasi rekomendasi lapangan tenis di Malang. Beberapa Batasan yang dapat dipatuhi agar sistem dapat berjalan secara maksimal adalah sebagai berikut:

1. Aplikasi ini bernama "TENISKU" yang hanya dapat dioperasikan menggunakan peranti bergerak dengan sistem operasi minimal Android 6.0.
2. Bahasa pemrograman yang digunakan dalam proses implementasi menggunakan Bahasa java.
3. Basis data yang digunakan dalam proses implementasi menggunakan Google Firebase.
4. Pengguna harus mengaktifkan fitur GPS dan koneksi internet untuk menggunakan aplikasi.

5.3 Implementasi Basis Data

Pada tahap implementasi basis data, peneliti menggunakan Google Firebase untuk menyimpan setiap data yang dibutuhkan dalam pengembangan aplikasi ini. Google Firebase dapat menampilkan basis data secara *realtime* terhadap pengguna. Google Firebase menggunakan format JSON untuk proses pengkodeannya. Struktur kode JSON dari pengembangan aplikasi ditampilkan pada Tabel 5.4.

Tabel 5.4 Implementasi Basis Data

```
{
  "pengguna": [ null, {
    "alamat": "Jl. Surabaya, Gading Kasri, Kec. Klojen, Kota Malang, Jawa Timur 65115",
    "foto": "https://firebasestorage.googleapis.com/v0/b/rekomendasi-tenis.appspot.com/o/lapangan%20tenis%20PELTI.PNG?alt=media&token=f96c8f5d-e35e-43cc-a141-ac95890550c0",
    "harga": 15000,
    "jumlahlapangan": 4,
    "lat": -7.966493168148484,
    "lng": 112.61822241391454,
    "nama": "Lapangan Tenis PELTI",
```

```
13      "rating": 4.7,
14      "sifat": "Umum",
15      "telepon": "0341-559413",
16      "waktubuka": "Senin-Jumat 08:00 - 16:00"
17    },
```

5.4 Implementasi Kode Program

Pada tahap ini menjelaskan tentang implementasi kode program dari setiap kelas yang digunakan untuk pengembangan aplikasi rekomendasi lapangan tenis di Malang. Kode program yang dibahas pada tahap ini hanya kelas *activity* yang memiliki peran penting saja dan bukan kelas sumber data.

5.4.1 Implementasi Kode Program Kelas Main

Kelas Main merupakan kelas yang pertama kali ditampilkan saat pengguna menggunakan aplikasi. Kode program dari kelas main dapat dilihat pada Tabel 5.5 di bawah ini.

Tabel 5.5 Implementasi Kode Program Kelas Main

```
1   package com.latihan.rekomendasitenis;
2
3   import ...
4
5   public class MainActivity extends AppCompatActivity
6   implements View.OnClickListener {
7
8       FusedLocationProviderClient mClient;
9       int totalUser = 2;
10
11      @Override
12      protected void onCreate(Bundle savedInstanceState) {
13          super.onCreate(savedInstanceState);
14          setContentView(R.layout.activity_main);
15
16          requestPermissions();
17
18          if(haveNetworkConnection()) {
19              alertInternetConnection();
20          }
21
22          mClient =
23  LocationServices.getFusedLocationProviderClient(this);
24
25          Button btnGdss = findViewById(R.id.btn_gdss);
26          Button btnList = findViewById(R.id.btn_list);
27          btnGdss.setOnClickListener(this);
28          btnList.setOnClickListener(this);
29      }
30
31      @SuppressLint({"NonConstantResourceId", "SetTextI18n"})
32      @Override
33      public void onClick(View v) {
```

```
        switch(v.getId()){
            case R.id.btn_list:
                Intent listLapangan = new
Intent(MainActivity.this, ListLapangan.class);
                startActivity(listLapangan);
                break;
            case R.id.btn_gdss:
                AlertDialog.Builder mBuilder = new
AlertDialog.Builder(MainActivity.this);
                View mView =
getLayoutInflater().inflate(R.layout.dialog_pengguna,
null);

                Button btnPlus =
mView.findViewById(R.id.btn_tambah);
                Button btnMinus =
mView.findViewById(R.id.btn_kurang);
                Button btnConfirm =
mView.findViewById(R.id.btn_konfirmasi);
                TextView jmlhPengguna =
mView.findViewById(R.id.jumlah_pengguna);

jmlhPengguna.setText(Integer.toString(totalUser));

                btnPlus.setOnClickListener(v1 -> {
                    if (totalUser == 9) {

Toast.makeText(getApplicationContext(), "Maksimum jumlah
pengguna adalah 9", Toast.LENGTH_LONG).show();
                        jmlhPengguna.setText("9");
                    } else {
                        totalUser++;

jmlhPengguna.setText(Integer.toString(totalUser));
                    }
                });

                btnMinus.setOnClickListener(v13 -> {
                    if (totalUser > 2) {
                        totalUser--;

jmlhPengguna.setText(Integer.toString(totalUser));
                    } else {

Toast.makeText(getApplicationContext(), "Minimum jumlah
pengguna adalah 2", Toast.LENGTH_LONG).show();
                    }
                });

                btnConfirm.setOnClickListener(v12 -> {
                    if
(ActivityCompat.checkSelfPermission(MainActivity.this,
ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED)
{
                        return;
                    }
mClient.getLastLocation().addOnSuccessListener(MainActivity
.this, location -> {
```

```
90
91                      if (location != null) {
92
93                          double mLat =
94  location.getLatitude();
95                          double mLong =
96  location.getLongitude();
97
98                          LatLng latLng = new LatLng(mLat,
99  mLong);
100
101                             Intent intent = new
102 Intent(MainActivity.this, BobotActivity.class);
103                             Bundle args = new Bundle();
104                     args.putParcelable("user_lat_lng",
105 latLng);
106                         intent.putExtras(args);
107                         intent.putExtra("jumlah_pengguna",
108 totalUser);
109                 Log.i("JUMLAH_USER",
110 String.valueOf(totalUser));
111
112                             startActivity(intent);
113                         }
114                     });
115                 });
116
117                 mBuilder.setView(mView);
118                 AlertDialog dialog = mBuilder.create();
119                 dialog.show();
120                 break;
121                 }
122         }
123
124     private void requestPermissions() {
125
126 ActivityCompat.requestPermissions(MainActivity.this, new
127 String[]{ACCESS_FINE_LOCATION}, 1);
128     }
129
130     private boolean haveNetworkConnection() {
131
132         boolean haveConnectedWifi = false;
133         boolean haveConnectedMobile = false;
134
135         ConnectivityManager cm = (ConnectivityManager)
136 getSystemService(Context.CONNECTIVITY_SERVICE);
137         assert cm != null;
138         NetworkInfo[] netInfo = cm.getAllNetworkInfo();
139         for (NetworkInfo info : netInfo) {
140             if
141 (info.getTypeName().equalsIgnoreCase("WIFI"))
142                 if (info.isConnected())
143                     haveConnectedWifi = true;
144             if
145 (info.getTypeName().equalsIgnoreCase("MOBILE"))
146                 if (info.isConnected())
147                     haveConnectedMobile = true;
148         }
```

```
149         return !haveConnectedWifi && !haveConnectedMobile;
150     }
151
152     private void alertInternetConnection() {
153         new AlertDialog.Builder(this)
154                 .setTitle("Connection Error")
155                 .setMessage("Mohon aktifkan koneksi
156 internet")
157                 .setPositiveButton("Coba Lagi",
157 (dialog, which) -> {
158                     if (haveNetworkConnection()) {
159                         alertInternetConnection();
160                     }
161                 })
162                 .setCancelable(false)
163                 .create().show();
164     }
165 }
```

Penjelasan dari kode program pada Tabel 5.5 adalah sebagai berikut:

1. Baris 1: Tempat package yang digunakan dalam pengembangan aplikasi.
2. Baris 3: Fungsi untuk memanggil library yang digunakan.
3. Baris 5 – 29: Fungsi untuk pengecekan fungsi yang pertama kali dijalankan beserta inisialisasi dari setiap variabel.
4. Baris 31 – 122: Fungsi untuk mengeksekusi pilihan tombol yang telah ditekan oleh pengguna.
5. Baris 124 – 128: Fungsi untuk mengaktifkan GPS dari pengguna.
6. Baris 130 – 150: Fungsi untuk menentukan apakah pengguna tersambung dengan internet atau tidak.
7. Baris 152 – 165: Fungsi untuk memunculkan dialog jika pengguna belum tersambung dengan internet.

5.4.2 Implementasi Kode Program Daftar Lapangan

Kelas ini berisikan keseluruhan daftar lapangan yang ada di kota Malang. Kelas ini dibuat agar pengguna dapat mengetahui daftar lapangan yang ada sebelum membuat keputusan dari hasil rekomendasi. Kode program dari kelas ini dapat dilihat pada Tabel 5.6.

Tabel 5.6 Implementasi Kode Program Daftar Lapangan

```
1   package com.latihan.rekomendasitenis;
2
3   import ...
4
5   public class ListLapangan extends AppCompatActivity {
6
7       ProgressBar progressBar;
8       RecyclerView rvList;
9
```

```
    ArrayList<LapanganTenis> listLapangan = new
ArrayList<>();
    ListLapanganAdapter adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.list_lapangan);

        setTitle("List Lapangan");

        progressBar = findViewById(R.id.progressBar);
        rvList = findViewById(R.id.rv_list);

        getData();

        setRecyclerView();
    }

    private void getData() {
        DatabaseReference mDatabase =
FirebaseDatabase.getInstance().getReference();

mDatabase.child("pengguna").addValueEventListener(new
ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot
dataSnapshot) {
                for (DataSnapshot snapshot :
dataSnapshot.getChildren()) {
                    LapanganTenis ltData =
snapshot.getValue(LapanganTenis.class);

                    listLapangan.add(ltData);
                }

                adapter = new
ListLapanganAdapter(listLapangan, ListLapangan.this);

                rvList.setAdapter(adapter);

                progressBar.setVisibility(View.INVISIBLE);

Toast.makeText(getApplicationContext(),"Data Berhasil
Dimuat", Toast.LENGTH_LONG).show();
            }

            @Override
            public void onCancelled(@NonNull DatabaseError
databaseError) {

Toast.makeText(getApplicationContext(),"Data Gagal Dimuat",
Toast.LENGTH_LONG).show();
                Log.e("MyListActivity",
databaseError.getDetails()+" "+databaseError.getMessage());
            }
        });
    }
```

```
69
70      private void setRecyclerView() {
71          rvList.setLayoutManager(new
72  LinearLayoutManager(this));
73          rvList.setHasFixedSize(true);
74      }
75  }
```

Penjelasan dari kode program pada Tabel 5.6 adalah sebagai berikut:

1. Baris 1: Tempat package yang digunakan dalam pengembangan aplikasi.
2. Baris 3: Fungsi untuk memanggil library yang digunakan.
3. Baris 5 – 27: Fungsi untuk pengecekan fungsi yang pertama kali dijalankan beserta inisialisasi dari setiap variabel.
4. Baris 29 – 68: Fungsi untuk mendapatkan data dari firebase.
5. Baris 70 – 75: Fungsi yang digunakan untuk mengubah bentuk data yang telah diambil dari firebase menjadi bentuk recyclerview.

5.4.3 Implementasi Kode Program Kelas Bobot

Kelas bobot dimunculkan ketika pengguna telah menekan tombol list rekomendasi dan telah menentukan jumlah anggota. Kelas ini berisikan kriteria-kriteria yang digunakan sebagai hasil rekomendasi pada hasil akhir nanti. Pengguna dapat mengisi setiap bobot kriteria yang ada, setelah itu dapat menekan tombol konfirmasi agar data tersebut dapat tersimpan dan diolah oleh aplikasi. Kode program dari kelas ini dapat dilihat pada Tabel 5.7.

Tabel 5.7 Implementasi Kode Program Kelas Bobot

```
1   package com.latihan.rekomendasitenis;
2
3   import ...
4
5   public class BobotActivity extends AppCompatActivity
6   implements View.OnClickListener {
7
8   protected void onCreate(Bundle savedInstanceState) {
9           super.onCreate(savedInstanceState);
10          setContentView(R.layout.activity_bobot);
11
12          CountTopsisBorda();
13
14          if (!haveNetworkConnection()) {
15              needInternetConnection();
16          }
17
18          prepare();
19  setTitle("Pengguna ke-"+ (counter+1));
20  countSeekBar();
21
22          ImageButton btnHelp = findViewById(R.id.btn_help);
23          Button btnConfirm = findViewById(R.id.btn_confirm);
24          btnHelp.setOnClickListener(this);
```

```
        btnConfirm.setOnClickListener(this);
    }
    private void prepare() {
        Intent intent = getIntent();
        LatLng latLng =
intent.getParcelableExtra("user_lat_lng");
        user_lat = latLng.latitude;
        user_lng = latLng.longitude;
        totalUser = intent.getIntExtra("jumlah_pengguna", 2);
    }

private void countSeekBar() {

        sbHarga.setMax(10);
        sbJarak.setMax(10);
        sbRating.setMax(10);
        sbJumlahLapangan.setMax(10);

        sbHarga.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
            @Override
            public void onProgressChanged(SeekBar seekBar, int
progress, boolean fromUser) {
                bobotHarga = seekBar.getProgress();
                tvHarga.setText(String.valueOf(bobotHarga));
            }

            @Override
            public void onStartTrackingTouch(SeekBar seekBar) {
            }

            @Override
            public void onStopTrackingTouch(SeekBar seekBar) {
            }
        });

        sbJarak.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
            @Override
            public void onProgressChanged(SeekBar seekBar, int
progress, boolean fromUser) {
                bobotJarak = seekBar.getProgress();
                tvJarak.setText(String.valueOf(bobotJarak));
            }

            @Override
            public void onStartTrackingTouch(SeekBar seekBar) {
            }

            @Override
            public void onStopTrackingTouch(SeekBar seekBar) {
            }
        });

        sbRating.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
            @Override
            public void onProgressChanged(SeekBar seekBar, int
progress, boolean fromUser) {
```

```
84                  bobotRating = seekBar.getProgress();
85          tvRating.setText(String.valueOf(bobotRating));
86                  }
87
88                  @Override
89                  public void onStartTrackingTouch(SeekBar seekBar) {
90                  }
91
92                  @Override
93                  public void onStopTrackingTouch(SeekBar seekBar) {
94                  }
95              });
96
97              sbJumlahLapangan.setOnSeekBarChangeListener(new
98          SeekBar.OnSeekBarChangeListener() {
99                  @Override
100                 public void onProgressChanged(SeekBar seekBar, int
101         progress, boolean fromUser) {
102                     bobotJumlahLapangan = seekBar.getProgress();
103         tvJumlahLapangan.setText(String.valueOf(bobotJumlahLapangan
104         ));
105                 }
106
107                 @Override
108                 public void onStartTrackingTouch(SeekBar seekBar) {
109                 }
110
111                 @Override
112                 public void onStopTrackingTouch(SeekBar seekBar) {
113                 }
114             });
115     }
116
117     @SuppressLint("NonConstantResourceId")
118     @Override
119     public void onClick(View v) {
120         switch (v.getId()) {
121             case R.id.btn_help:
122                 help();
123                 break;
124             case R.id.btn_confirm:
125                 if (bobotHarga == 0) {
126                     Toast.makeText(getApplicationContext(),
127         "Harga tidak boleh 0", Toast.LENGTH_SHORT).show();
128
129                 } else if (bobotJarak == 0) {
130                     Toast.makeText(getApplicationContext(),
131         "Jarak tidak boleh 0", Toast.LENGTH_SHORT).show();
132
133                 } else if (bobotRating == 0) {
134                     Toast.makeText(getApplicationContext(),
135         "Rating tidak boleh 0", Toast.LENGTH_SHORT).show();
136
137                 } else if (bobotJumlahLapangan == 0) {
138                     Toast.makeText(getApplicationContext(),
139         "Jumlah Lapangan tidak boleh 0",
140         Toast.LENGTH_SHORT).show();
141
142                 } else {
```

```
                    counter++;

                    //Normalisasi bobot kriteria
bobotNormalisasiHarga = ((double) bobotHarga) / ((double)
bobotHarga + (double) bobotJarak + (double) bobotRating +
(double) bobotJumlahLapangan);

bobotNormalisasiJarak = ((double) bobotJarak) / ((double)
bobotHarga + (double) bobotJarak + (double) bobotRating +
(double) bobotJumlahLapangan);

bobotNormalisasiRating = ((double) bobotRating) / ((double)
bobotHarga + (double) bobotJarak + (double) bobotRating +
(double) bobotJumlahLapangan);

                    if (counter == 1) {

                        KalkulasiTopsis();

GabungTopsisBordaKeFirebaseForFirstUser(ciPositif);

                    } else if (counter >= 1 && counter
< totalUser) {

                        KalkulasiTopsis();

GabungTopsisBordaKeFirebaseForMiddleUser(ciPositif);

                    } else if (counter == totalUser) {

                        KalkulasiTopsis();

GabungTopsisBordaKeFirebaseForLastUser(ciPositif);
                    }
                }
                break;
            }
        }

    private void help() {
        new AlertDialog.Builder(this)
                .setTitle("Keterangan")
                .setMessage("" +
                        "1. Harga: Semakin besar nilainya
maka semakin murah harga yang diinginkan\n\n" +
                        "2. Jarak: Semakin besar nilainya
maka semakin dekat lokasi yang diinginkan\n\n" +
                        "3. Rating: Semakin besar nilainya
maka semakin bagus lapangan yang diingikan\n\n" +
                        "4. Jumlah Lapangan: Semakin besar
nilainya maka semakin banyak jumlah lapangan yang ada")
                .setPositiveButton("OK", (dialog, which) ->
{
                })
                .setCancelable(true)
                .create().show();
    }
```

Penjelasan dari kode program pada Tabel 5.7 adalah sebagai berikut:

1. Baris 1: Tempat package yang digunakan dalam pengembangan aplikasi.
2. Baris 3: Fungsi untuk memanggil library yang digunakan.
3. Baris 5 – 26: Fungsi untuk pengecekan fungsi yang pertama kali dijalankan beserta inisialisasi dari setiap variabel
4. Baris 27 – 34: Fungsi untuk mengambil data dari kelas main.
5. Baris 36 – 115: Fungsi untuk menentukan bobot dari setiap kriteria yang dimasukkan oleh pengguna.
6. Baris 117 – 180: Fungsi untuk memasukkan data dari pengguna dan memanggil method untuk melakukan perhitungan dengan topsis dari setiap jumlah pengguna.
7. Barus 182 – 199: Fungsi untuk memunculkan dialog berupa bantuan pengisian kriteria ketika pengguna menekan tombol tanda tanya di sebelah tombol konfirmasi.

5.4.4 Implementasi Kode Program Hasil Rekomendasi

Hasil rekomendasi dibuat supaya dapat memenuhi keinginan pengguna berdasarkan kriteria yang telah dimasukkan di tahap sebelumnya. Pada kelas ini, setiap kriteria di kalkulasi dengan algoritma TOPSIS dan BORDA untuk mendapatkan hasil rekomendasi yang sesuai. Kode program dari kelas ini dapat dilihat pada Tabel 5.8.

Tabel 5.8 Implementasi Kode Program Hasil Rekomendasi

```
1  package com.latihan.rekomendasitenis;
2
3  import ...
4
5  public class ListRekomendasi extends AppCompatActivity {
6
7      @Override
8      protected void onCreate(Bundle savedInstanceState) {
9          super.onCreate(savedInstanceState);
10         setContentView(R.layout.list_rekomendasi);
11         setTitle("Hasil Rekomendasi");
12
13         RecyclerView rvListRekom =
14 findViewById(R.id.rv_listRekom);
15         rvListRekom.setHasFixedSize(true);
16         rvListRekom.setLayoutManager(new
17 LinearLayoutManager(this));
18
19
20         ArrayList<ListTopsis> listTopsis;
21
22         listTopsis =
23 getIntent().getParcelableArrayListExtra("array_hasil");
24         Log.i("LIST_FIREBASE", String.valueOf(listTopsis));
25
```

```
26          Collections.sort(listTopsis, new SortBorda());
27          ListRekomendasiAdapter adapter = new
28  ListRekomendasiAdapter(ListRekomendasi.this, listTopsis);
29          rvListRekom.setAdapter(adapter);
30      }
31
32      private static class SortBorda implements
33  Comparator<ListTopsis> {
34          @Override
35          public int compare(ListTopsis obj1, ListTopsis
36  obj2) {
37
38              return Double.compare(obj2.getCiPositif(),
39  obj1.getCiPositif());
40          }
41      }
42
43      @Override
44      public void onBackPressed() {
45          new AlertDialog.Builder(this)
46                  .setMessage("Kembali ke halaman awal?")
47                  .setPositiveButton("Tidak",
48  (dialogInterface, i) -> {
49
50                  })
51                  .setNegativeButton("Ya", (dialogInterface,
52  i) -> finish())
53                  .setCancelable(true)
54                  .create().show();
55      }
56  }
```

Penjelasan dari kode program pada Tabel 5.8 adalah sebagai berikut:

1. Baris 1: Tempat package yang digunakan dalam pengembangan aplikasi.
2. Baris 3: Fungsi untuk memanggil library yang digunakan.
3. Baris 5 – 30: Fungsi untuk pengecekan fungsi yang pertama kali dijalankan beserta inisialisasi dari setiap variabel.
4. Baris 32 – 41: Fungsi untuk mengurutkan data berdasarkan algoritma BORDA.
5. Baris 43 – 56: Fungsi untuk memunculkan dialog jika pengguna menekan tombol kembali pada layar *smartphone*.

5.4.5 Implementasi Kode Program Detail Lapangan

Kelas detail lapangan berfungsi agar pengguna dapat mengetahui setiap lapangan secara lebih rinci serta pengguna dapat mengetahui navigasi menuju lapangan tenis pilihannya. Kode program dari kelas ini dapat dilihat pada Tabel 5.9.

Tabel 5.9 Implementasi Kode Program Detail Lapangan

```
1  package com.latihan.rekomendasitenis;
2
3  import ...
4
5  public class DetailLapangan extends AppCompatActivity {
6      public static final String EXTRA_PERSON =
7  "extra_person";
8
9      @SuppressLint("SetTextI18n")
10     @Override
11     protected void onCreate(Bundle savedInstanceState) {
12         super.onCreate(savedInstanceState);
13         setContentView(R.layout.detail_lapangan2);
14         setTitle("");
15
16         TextView nama = findViewById(R.id.nama_lapangan);
17         TextView alamat = findViewById(R.id.alamat);
18         TextView telepon = findViewById(R.id.telepon);
19         TextView waktuBuka = findViewById(R.id.waktu_buka);
20         TextView harga = findViewById(R.id.harga);
21         TextView sifat = findViewById(R.id.sifat);
22         TextView rating = findViewById(R.id.rating);
23         ImageView fotoLapangan =
24 findViewById(R.id.foto_lapangan);
25
26         Button btnRute = findViewById(R.id.btn_rute);
27
28         ListTopsis listTopsis =
29 getIntent().getParcelableExtra(EXTRA_PERSON);
30
31         assert listTopsis != null;
32         nama.setText(listTopsis.getNama());
33         alamat.setText(listTopsis.getAlamat());
34         telepon.setText(listTopsis.getTelepon());
35         waktuBuka.setText(listTopsis.getWaktubuka());
36         sifat.setText(listTopsis.getSifat());
37         harga.setText("Rp " + listTopsis.getHarga());
38
39 rating.setText(String.valueOf(listTopsis.getRating()));
40
41 Picasso.get().load(listTopsis.getFoto()).fit().into(fotoLap
42 angan);
43
44         btnRute.setOnClickListener(v -> {
45             String map = "google.navigation:q=" +
46 listTopsis.getAlamat();
47             Intent intent = new Intent(Intent.ACTION_VIEW,
48 Uri.parse(map));
49             startActivity(intent);
50         });
51     }
52 }
```

Penjelasan dari kode program pada Tabel 5.9 adalah sebagai berikut:

1. Baris 1: Tempat package yang digunakan dalam pengembangan aplikasi.

2. Baris 3: Fungsi untuk memanggil library yang digunakan.
3. Baris 5 – 42: Fungsi untuk pengecekan fungsi yang pertama kali dijalankan beserta inisialisasi dari setiap variabel.
4. Baris 44 – 52: Fungsi yang digunakan untuk menavigasi pengguna ke tempat tujuan melalui aplikasi google map ketika pengguna menekan tombol lihat rute pada halaman detail lapangan.

5.4.6 Implementasi Kode Program Algoritma TOPSIS

Kelas algoritma TOPSIS digunakan agar hasil rekomendasi dapat sesuai dengan nilai dari masukkan kriteria pengguna. Kode program dari kelas ini dapat dilihat pada Tabel 5.10.

Tabel 5.10 Implementasi Kode Program Algoritma TOPSIS

```
1   package com.latihan.rekomendasitenis;
2
3   import android.util.Log;
4
5   import java.util.ArrayList;
6
7   public class CountTopsis {
8       public CountTopsis() {
9
10      }
11
12      public Double PembagiHarga(ArrayList<ListTopsis> list,
13  double pembagiHarga) {
14
15          for (int i = 0; i < list.size(); i++) {
16              pembagiHarga = pembagiHarga +
17  (Math.pow(list.get(i).getHarga(), 2));
18          }
19
20          return pembagiHarga;
21      }
22
23      public Double PembagiJarak(ArrayList<ListTopsis> list,
24  double pembagiJarak) {
25
26          for (int i = 0; i < list.size(); i++) {
27              pembagiJarak = pembagiJarak +
28  (Math.pow(list.get(i).getJarak(), 2));
29          }
30
31          return pembagiJarak;
32      }
33
34      public Double PembagiRating(ArrayList<ListTopsis> list,
35  double pembagiRating) {
36
37
38          for (int i = 0; i < list.size(); i++) {
39              pembagiRating = pembagiRating +
40  (Math.pow(list.get(i).getRating(), 2));
41          }
```

```
42
43          return pembagiRating;
44      }
45
46      public Double
47  PembagiJumlahLapangan(ArrayList<ListTopsis> list, double
48  pembagiJumlahLapangan) {
49
50          for (int i = 0; i < list.size(); i++) {
51              pembagiJumlahLapangan = pembagiJumlahLapangan +
52  (Math.pow(list.get(i).getJumlahlapangan(), 2));
53          }
54
55          return pembagiJumlahLapangan;
56      }
57
58      public ArrayList<Double>
59  NormalisasiHarga(ArrayList<ListTopsis> list,
60                                                      double
61  pembagiHarga, ArrayList<Double> normalisasiHarga) {
62
63          for (int i = 0; i < list.size(); i++) {
64              normalisasiHarga.add(list.get(i).getHarga() /
65  pembagiHarga);
66          }
67
68          return normalisasiHarga;
69      }
70
71      public ArrayList<Double>
72  NormalisasiJarak(ArrayList<ListTopsis> list,
73                                                      double
74  pembagiJarak, ArrayList<Double> normalisasiJarak) {
75
76          for (int i = 0; i < list.size(); i++) {
77              normalisasiJarak.add(list.get(i).getJarak() /
78  pembagiJarak);
79          }
80
81          return normalisasiJarak;
82      }
83
84      public ArrayList<Double>
85  NormalisasiRating(ArrayList<ListTopsis> list,
86                                                      double
87  pembagiRating, ArrayList<Double> normalisasiRating) {
88
89          for (int i = 0; i < list.size(); i++) {
90              normalisasiRating.add(list.get(i).getRating() /
91  pembagiRating);
92          }
93
94          return normalisasiRating;
95      }
96
97      public ArrayList<Double>
98  NormalisasiJumlahLapangan(ArrayList<ListTopsis> list,
99
100
```

```
101                                                          double
102 pembagiJumlahLapangan, ArrayList<Double>
103 normalisasiJumlahLapangan) {
104
105         for (int i = 0; i < list.size(); i++) {
106
107 normalisasiJumlahLapangan.add(list.get(i).getRating() /
108 pembagiJumlahLapangan);
109         }
110
111         return normalisasiJumlahLapangan;
112     }
113
114     public ArrayList<Double>
115 TerbobotHarga(ArrayList<Double> normalisasiHarga,
116                                                          double
117 bobotHarga,
118
119 ArrayList<Double> terbobotHarga) {
120
121         for (int i = 0; i < normalisasiHarga.size(); i++) {
122             terbobotHarga.add(normalisasiHarga.get(i) *
123 bobotHarga);
124         }
125
126         return terbobotHarga;
127     }
128
129     public ArrayList<Double>
130 TerbobotJarak(ArrayList<Double> normalisasiJarak,
131                                                          double
132 bobotJarak,
133 ArrayList<Double> terbobotJarak) {
134
135         for (int i = 0; i < normalisasiJarak.size(); i++) {
136             terbobotJarak.add(normalisasiJarak.get(i) *
137 bobotJarak);
138         }
139
140         return terbobotJarak;
141     }
142
143     public ArrayList<Double>
144 TerbobotRating(ArrayList<Double> normalisasiRating,
145                                                          double
146 bobotRating,
147
148 ArrayList<Double> terbobotRating) {
149
150         for (int i = 0; i < normalisasiRating.size(); i++)
151 {
152             terbobotRating.add(normalisasiRating.get(i) *
153 bobotRating);
154         }
155
156         return terbobotRating;
157     }
158
159
```

```
    public ArrayList<Double>
TerbobotJumlahLapangan(ArrayList<Double>
normalisasiJumlahLapangan,
                                        double
bobotJumlahLapangan,

ArrayList<Double> terbobotJumlahLapangan) {
        for (int i = 0; i <
normalisasiJumlahLapangan.size(); i++) {

terbobotJumlahLapangan.add(normalisasiJumlahLapangan.get(i)
* bobotJumlahLapangan);
        }

        return terbobotJumlahLapangan;
    }

    public ArrayList<Double> siPositif
            (ArrayList<Double> listHarga, ArrayList<Double>
listJarak, ArrayList<Double> listRating, ArrayList<Double>
listJumlahLapangan,
            double aPositifHarga, double aPositifJarak,
double aPositifRating, double aPositifJumlahLapangan,
ArrayList<Double> siPositif) {

        for (int i = 0; i < listHarga.size(); i++) {

siPositif.add(Math.sqrt((Math.pow((listHarga.get(i) -
aPositifHarga), 2))
                    + (Math.pow((listJarak.get(i) -
aPositifJarak), 2))
                    + (Math.pow((listRating.get(i) -
aPositifRating), 2))
                    + (Math.pow((listJumlahLapangan.get(i)
- aPositifJumlahLapangan), 2))));
        }

        return siPositif;
    }

    public ArrayList<Double> siNegatif
            (ArrayList<Double> listHarga, ArrayList<Double>
listJarak, ArrayList<Double> listRating, ArrayList<Double>
listJumlahLapangan,
            double aNegatifHarga, double aNegatifJarak,
double aNegatifRating, double aNegatifJumlahLapangan,
ArrayList<Double> siNegatif) {

        for (int i = 0; i < listHarga.size(); i++) {

siNegatif.add(Math.sqrt((Math.pow((listHarga.get(i) -
aNegatifHarga), 2))
                    + (Math.pow((listJarak.get(i) -
aNegatifJarak), 2))
                    + (Math.pow((listRating.get(i) -
aNegatifRating), 2))
                    + (Math.pow((listJumlahLapangan.get(i)
- aNegatifJumlahLapangan), 2))));
```

```
219             }
220
221             return siNegatif;
222         }
223
224         public ArrayList<Double> jumlahSi
225                 (ArrayList<Double> listSiPositif,
226 ArrayList<Double> listSiNegatif, ArrayList<Double>
227 jumlahSi) {
228
229             for (int i = 0; i < listSiPositif.size(); i++) {
230                 jumlahSi.add(listSiPositif.get(i) +
231 listSiNegatif.get(i));
232             }
233             return jumlahSi;
234         }
235
236         public ArrayList<Double> ciPositif
237                 (ArrayList<Double> listSiNegatif,
238 ArrayList<Double> jumlahSi, ArrayList<Double> ciPositif) {
239
240             for (int i = 0; i < listSiNegatif.size(); i++) {
241                 ciPositif.add(listSiNegatif.get(i) /
242 jumlahSi.get(i));
243             }
244             return ciPositif;
245         }
246 }
```

Penjelasan dari kode program pada Tabel 5.10 adalah sebagai berikut:

1. Baris 1: Tempat package yang digunakan dalam pengembangan aplikasi.
2. Baris 3 – 5: Fungsi untuk memanggil library yang digunakan.
3. Baris 8 – 56: Fungsi untuk menghitung nilai pembagi dari setiap kriteria.
4. Baris 58 – 112: Fungsi untuk menghitung nilai normalisasi dari setiap kriteria.
5. Baris 114 – 176: Fungsi untuk menghitung nilai terbobot dari setiap kriteria.
6. Barus 178 – 222: Fungsi untuk menghitung nilai siPositif dan siNegatif dari setiap kriteria.
7. Baris 224 – 234: Fungsi untuk menghitung nilai jumlah Si.
8. Baris 236 – 246: Fungsi untuk menghitung hasil dari ciPositif.

5.4.7 Implementasi Kode Program Algoritma BORDA

Kelas algoritma BORDA digunakan agar hasil rekomendasi dapat sesuai dengan nilai dari masukkan kriteria pengguna. Kode program dari kelas ini dapat dilihat pada Tabel 5.11.

Tabel 5.11 Implementasi Kode Program Algoritma BORDA

```
1   private static class SortBorda implements
2   Comparator<ListTopsis> {
3           @Override
4           public int compare(ListTopsis obj1, ListTopsis
5   obj2) {
6
7               return Double.compare(obj2.getCiPositif(),
8   obj1.getCiPositif());
9           }
10      }
11  private void
12  SortAndBordaForMiddleUser(ArrayList<ListTopsis> arrayBaru)
13  {
14
15          //sort ascending berdasarkan ciPositif
16          Collections.sort(arrayBaru, new SortBorda());
17
18          for (int i = 0; i < arrayBaru.size(); i++) {
19
20              int id = arrayBaru.get(i).getId();
21              double doubleCi =
22  arrayBaru.get(i).getCiPositif() * (i+1);
23  arrayBaru.set(i, new ListTopsis(doubleHarga, doubleJarak,
24  doubleRating, doubleJumlahLapangan, doubleCi, nama, alamat,
25  foto, id, waktubuka, sifat, telepon));
26
27          }
28
29          Collections.sort(arrayBaru, new SortById());
30
31          for (int i = 0; i < arrayBaru.size(); i++) {
32
33              int id = arrayBaru.get(i).getId();
34              double doubleCiGabungan =
35  arrayBaru.get(i).getCiPositif() +
36  listBorda.get(i).getCiPositif();
37
38  listBorda.set(i, new ListTopsis(doubleHarga, doubleJarak,
39  doubleRating, doubleJumlahLapangan, doubleCiGabungan, nama,
40  alamat, foto, id, waktubuka, sifat, telepon));
41          }
42      }
```

Penjelasan dari kode program pada Tabel 5.11 adalah sebagai berikut:

1. Baris 1 – 10: Implementasi dari kelas BORDA untuk mengurutkan data dari daftar lapangan yang ada.
2. Baris 11 – 42: Implementasi mengurutkan data dengan BORDA yang ditujukan pada antara pengguna pertama dengan pengguna terakhir.

5.5 Implementasi Antarmuka

Pada tahap ini membahas tentang antarmuka aplikasi dari keseluruhan kode program yang telah dikerjakan. Tentunya implementasi antarmuka telah dibuat sesuai dengan perancangan awal dari aplikasi ini.

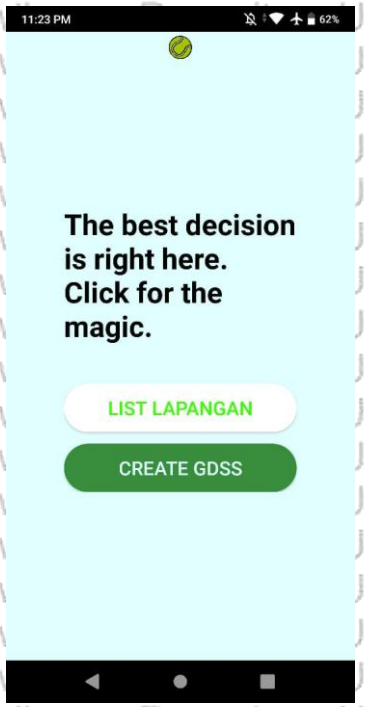
5.5.1 Implementasi Antarmuka Halaman Splash Screen

Halaman Splash Screen merupakan halaman yang muncul hanya dalam beberapa detik saja. Halaman ini ditunjukkan ketika ikon dari aplikasi pertama kali ditekan oleh pengguna. Implementasi dari halaman splash screen dapat dilihat pada Gambar 5.1.

Gambar 5.1 Implementasi Antarmuka Halaman Splash Screen

5.5.2 Implementasi Antarmuka Halaman Utama

Halaman utama adalah halaman yang terbuka jika halaman splash screen sudah menghilang. Halaman utama terdiri dari 2 pilihan tombol untuk memberi keputusan secara bebas kepada pengguna. Implementasi dari halaman utama dapat dilihat pada Gambar 5.2.



Gambar 5.2 Implementasi Antarmuka Halaman Utama

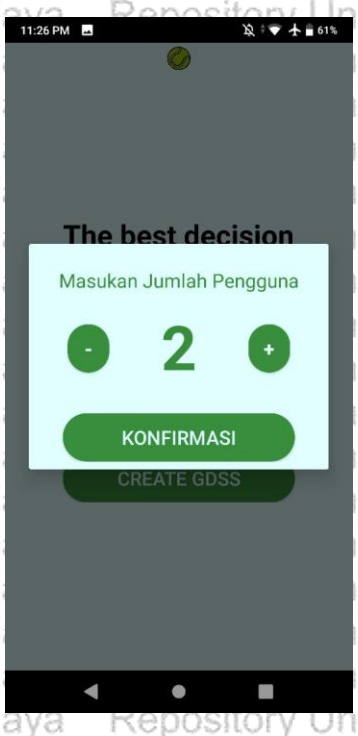
5.5.3 Implementasi Antarmuka Halaman Daftar Lapangan

Halaman ini akan muncul di layar pengguna ketika pengguna menekan tombol List Lapangan pada halaman utama. Implementasi dari halaman daftar lapangan dapat dilihat pada Gambar 5.3.

Gambar 5.3 Implementasi Antarmuka Halaman Daftar Lapangan

5.5.4 Implementasi Antarmuka Dialog Menentukan Jumlah Anggota

Antarmuka Dialog ini dimunculkan ketika pengguna menekan tombol Create GDSS pada halaman utama. Implementasi dari dialog tersebut dapat dilihat pada Gambar 5.4.



Gambar 5.4 Implementasi Antarmuka Dialog Menentukan Jumlah Anggota

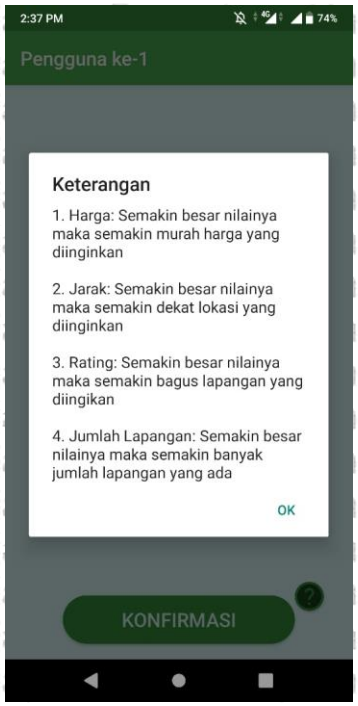
5.5.5 Implementasi Antarmuka Halaman Bobot Kriteria

Halaman bobot dimunculkan ketika pengguna telah menentukan jumlah anggotanya. Pada halaman ini, pengguna harus menentukan setiap bobot yang ada. Implementasi dari halaman bobot dapat dilihat pada Gambar 5.5.

Gambar 5.5 Implementasi Antarmuka Halaman Bobot Kriteria

5.5.6 Implementasi Antarmuka Dialog Bantuan

Antarmuka dialog bantuan dimunculkan ketika pengguna menekan tombol tanda tanya di sebelah tombol konfirmasi pada halaman bobot. Implementasi dari dialog tersebut dapat dilihat pada Gambar 5.6.



Gambar 5.6 Implementasi Antarmuka Dialog Bantuan

5.5.7 Implementasi Antarmuka Halaman Hasil Rekomendasi

Halaman hasil rekomendasi berisikan daftar rekomendasi lapangan yang ditujukan kepada pengguna berdasarkan kriteria yang telah dimasukkan. Implemenasi dari halaman ini dapat dilihat pada Gambar 5.7.

Gambar 5.7 Implementasi Antarmuka Halaman Hasil Rekomendasi

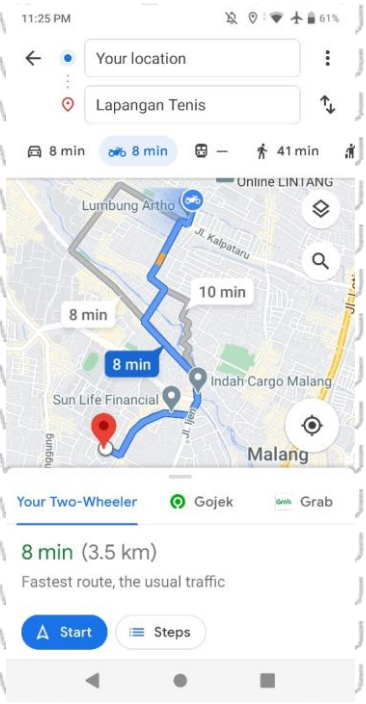
5.5.8 Implementasi Antarmuka Halaman Detail Lapangan

Halaman ini berisikan keterangan secara rinci dari setiap lapangan yang ada. Halaman detail dimunculkan ketika pengguna menekan salah satu *item* dari daftar hasil rekomendasi yang ada. Implementasi dari halaman detail lapangan dapat dilihat pada Gambar 5.8.

Gambar 5.8 Implementasi Antarmuka Halaman Hasil Rekomendasi

5.5.9 Implementasi Antarmuka Halaman Navigasi Maps

Halaman ini akan muncul ketika pengguna menekan tombol lihat rute pada halaman detail lapangan. Pengguna akan dialihkan ke halaman Google Maps. Implementasi dari halaman ini dapat dilihat pada Gambar 5.9.



Gambar 5.9 Implementasi Antarmuka Halaman Navigasi Maps

BAB 6 PENGUJIAN

Pada bab ini menjelaskan tentang bagaimana tahap-tahap melakukan proses pengujian terhadap aplikasi berdasarkan hasil dari implementasi aplikasi yang telah dijelaskan pada bab sebelumnya. Tahapan dari pengujian ini terbagi menjadi dua bagian, yaitu pengujian fungsional dan pengujian non-fungsional.

6.1 Pengujian Fungsional

Pengujian fungsional dilakukan untuk mengetahui aplikasi yang telah dibangun dapat dijalankan secara keseluruhan atau tidak dan bertujuan untuk menemukan kesalahan pada aplikasi.

6.1.1 Pengujian *Black-box*

Pengujian fungsional dilakukan menggunakan metode pengujian *black-box*. Pengujian ini dilakukan berdasarkan *usecase* yang telah dibuat sebelumnya. Berikut merupakan kasus pengujian dari setiap kebutuhan fungsional yang ada pada sistem rekomendasi lapangan tenis di Malang.

Tabel 6.1 hingga Tabel 6.6 akan menjelaskan tentang setiap kasus pengujian dari kebutuhan fungsional yang ada.

Tabel 6.1 Pengujian Menentukan Jumlah Anggota

Nama Kasus Uji	Menentukan Jumlah Anggota
Tujuan	Memastikan jumlah anggota pengguna yang berpartisipasi untuk pembuatan rekomendasi berdasarkan data kelompok
Prosedur Pengujian	1. Menekan tombol “Create GDDS” yang berwarna hijau pada halaman utama 2. Mengatur banyaknya jumlah anggota pada panel dialog yang muncul 3. Menekan tombol konfirmasi
Hasil yang Diharapkan	Pengguna dapat menentukan jumlah anggota

Tabel 6.2 Pengujian Memasukkan Bobot Kriteria

Nama Kasus Uji	Memasukkan Bobot Kriteria
Tujuan	Memastikan jumlah bobot kriteria yang dapat diisikan sesuai dengan jumlah anggota yang telah pengguna masukkan sebelumnya
Prosedur Pengujian	1. Menekan tombol “Create GDDS” yang berwarna hijau pada halaman utama

	2. Mengatur banyaknya jumlah anggota 3. Mengatur kriteria yang tersedia pada halaman bobot kriteria pengguna sesuai jumlah anggota
Hasil yang Diharapkan	Pengguna dapat memasukkan setiap bobot kriteria sesuai jumlah anggota dari pengguna

Tabel 6.3 Pengujian Melihat Hasil Rekomendasi

Nama Kasus Uji	Melihat Hasil Rekomendasi
Tujuan	Memastikan apakah sistem mampu menampilkan halaman hasil rekomendasi lapangan tenis berdasarkan kriteria yang telah diberikan
Prosedur Pengujian	1. Menekan tombol "Create GDDS" yang berwarna hijau pada halaman utama 2. Mengatur banyaknya jumlah anggota 3. Mengatur kriteria yang tersedia pada halaman bobot kriteria pengguna sesuai jumlah pengguna 4. Menekan tombol konfirmasi
Hasil yang Diharapkan	Pengguna dapat melihat daftar lapangan tenis hasil rekomendasi dari kriteria yang telah dimasukkan oleh pengguna dan diolah oleh sistem

Tabel 6.4 Pengujian Melihat Detail Lapangan Tenis

Nama Kasus Uji	Melihat Detail Lapangan Tenis
Tujuan	Memastikan apakah sistem mampu menampilkan informasi detail lapangan dari setiap daftar lapangan yang ada
Prosedur Pengujian	1. Menekan salah satu item pada halaman daftar lapangan atau halaman hasil rekomendasi
Hasil yang Diharapkan	Pengguna dapat melihat informasi detail lapangan secara lebih lengkap yang terdiri dari: foto, nama lapangan, alamat, harga, nomor telepon, jumlah lapangan, dan rute perjalanan menuju lapangan

Tabel 6.5 Pengujian Melihat Detail Lokasi

Nama Kasus Uji	Melihat Detail Lokasi
Tujuan	Memastikan sistem dapat memberikan rute perjalanan ke tempat lokasi lapangan dari lokasi pengguna
Prosedur Pengujian	1. Menekan tombol "Lihat Rute" pada halaman detail lapangan
Hasil yang Diharapkan	Pengguna mendapatkan rute perjalanan menuju lokasi lapangan menggunakan aplikasi Google Maps

Tabel 6.6 Pengujian Melihat Daftar Lapangan Tenis

Nama Kasus Uji	Melihat Daftar Lapangan Tenis
Tujuan	Memastikan apakah sistem mampu menampilkan seluruh daftar lapangan tenis kepada pengguna yang berasal dari basis data *firebase*
Prosedur Pengujian	1. Menekan tombol "List Lapangan" berwarna putih pada halaman utama
Hasil yang Diharapkan	Pengguna dapat melihat seluruh daftar lapangan tenis yang ada pada basis data *firebase*

Berdasarkan Tabel 6.1 hingga Tabel 6.6 didapatkan hasil dari setiap pengujian fungsional yang ada. Hasil tersebut dapat dilihat pada Tabel 6.7 di bawah ini.

Tabel 6.7 Hasil Pengujian Fungsional

Nama Kasus Uji	Hasil yang Diharapkan	Hasil yang Diperoleh	Status
Menentukan Jumlah Anggota	Pengguna dapat menentukan jumlah anggota	Jumlah anggota dari pengguna dapat diatur dengan mudah	Valid
Memasukkan Bobot Kriteria	Pengguna dapat memasukkan setiap bobot kriteria sesuai jumlah anggota dari pengguna	Halaman bobot kriteria berhasil ditampilkan sebanyak jumlah anggota pengguna	Valid
Melihat Hasil Rekomendasi	Pengguna dapat melihat daftar lapangan tenis hasil rekomendasi dari kriteria yang telah dimasukkan oleh	Hasil rekomendasi berhasil ditampilkan dan sesuai urutan dari kriteria masukkan pengguna	Valid

	pengguna dan diolah oleh sistem		
Melihat Detail Lapangan Tenis	Pengguna dapat melihat informasi detail lapangan secara lebih lengkap yang terdiri dari: foto, nama lapangan, alamat, harga, nomor telepon, jumlah lapangan, dan rute perjalanan menuju lapangan	Detail lapangan tenis yang dipilih oleh pengguna berhasil ditampilkan	Valid
Melihat Detail Lokasi	Pengguna mendapatkan rute perjalanan menuju lokasi lapangan menggunakan aplikasi Google Maps	Aplikasi Google Maps berhasil dibuka dan menunjukkan rute langsung ke tempat lokasi lapangan tenis	Valid
Melihat Daftar Lapangan Tenis	Pengguna dapat melihat seluruh daftar lapangan tenis yang ada pada basis data *firebase*	Daftar lapangan yang berasal dari data *firebase* berhasil ditampilkan	Valid

6.2 Pengujian Non-Fungsional

Pengujian non-fungsional dilakukan untuk mengetahui kualitas yang dimiliki oleh sistem. Pengujian non-fungsional dibagi menjadi 2 bagian yaitu pengujian validasi algoritme dan pengujian *usability* dari aplikasi yang telah dibangun.

6.2.1 Pengujian Validasi Algoritme

Pengujian validasi algoritme dilakukan dengan cara membandingkan nilai dari keluaran sistem dengan perhitungan manual yang telah dibuat oleh peneliti sebelumnya. Bobot kriteria yang diberikan pada pengujian validasi algoritme untuk pengguna pertama dan pengguna kedua dapat dilihat pada Tabel 6.8 dan Tabel 6.9.

Tabel 6.8 Bobot Kriteria Pengguna Pertama Pengujian Validasi Algoritme

Kriteria	Bobot
Harga	1
Jarak	1
Rating	1
Jumlah Lapangan	1

Tabel 6.9 Bobot Kriteria Pengguna Kedua Pengujian Validasi Algoritme

Kriteria	Bobot
Harga	2
Jarak	3
Rating	3
Jumlah Lapangan	2

Berdasarkan Tabel 6.8 dan Tabel 6.9, bobot kriteria dapat digunakan sebagai acuan dalam melakukan perhitungan algoritme ke dalam sistem ataupun melakukannya dengan manual. Data yang digunakan berupa kumpulan lapangan tenis. Hasil perhitungan untuk perbandingan antara perhitungan sistem dengan perhitungan manual dapat dilihat pada Tabel 6.10 dan 6.11 sebagai berikut.

Tabel 6.10 Perbandingan Perhitungan Pada Pengguna Pertama

No.	Nama Lapangan	Nilai Perhitungan Sistem	Nilai Perhitungan Manual
1	Lapangan Tenis PELTI	0,8028	0,8028
2	Lapangan Tenis Permata Jingga	0,7455	0,7455
3	Lapangan Tenis Istana Dieng Club House	0,1419	0,1419
4	Lapangan Tenis Puncak Dieng	0,5354	0,5354
5	Lapangan Tenis Gajayana	0,7002	0,7002
6	Lapangan Tenis Penjara	0,8885	0,8885
7	Lapangan Tenis Top	0,4723	0,4723
8	Lapangan Tenis Yon Bek-Ang	0,6114	0,6114
9	Lapangan Tenis Wijaya Kusuma	0,8233	0,8233
10	Lapangan Tenis Alumunium	0,6927	0,6927
11	Lapangan Tenis BCT	0,5776	0,5776
12	Lapangan Tenis Gossypium	0,4603	0,4603
13	Lapangan Tenis Araya	0,5311	0,5311
14	Lapangan Tenis Indoor UM	0,6865	0,6865
15	Lapangan Tenis UB SC	0,7070	0,7070
16	Lapangan Tenis Cakrawala UM	0,7975	0,7975

Tabel 6.11 Perbandingan Perhitungan Pada Pengguna Kedua

No.	Nama Lapangan	Nilai Perhitungan Sistem	Nilai Perhitungan Manual
1	Lapangan Tenis PELTI	0,3260	0,3260
2	Lapangan Tenis Permata Jingga	1,7034	1,7034
3	Lapangan Tenis Istana Dieng Club House	2,7201	2,7201
4	Lapangan Tenis Puncak Dieng	3,9987	3,9987
5	Lapangan Tenis Gajayana	5,0477	5,0477
6	Lapangan Tenis Penjara	6,5061	6,5061
7	Lapangan Tenis Top	8,1668	8,1668
8	Lapangan Tenis Yon Bek-Ang	10,7256	10,7256
9	Lapangan Tenis Wijaya Kusuma	12,3464	12,3464
10	Lapangan Tenis Alumunium	14,1635	14,1635
11	Lapangan Tenis BCT	16,0195	16,0195
12	Lapangan Tenis Gossypium	18,1518	18,1518
13	Lapangan Tenis Araya	20,4228	20,4228
14	Lapangan Tenis Indoor UM	22,5194	22,5194
15	Lapangan Tenis UB SC	25,0919	25,0919
16	Lapangan Tenis Cakrawala UM	27,9047	27,9047

Berdasarkan hasil yang ditunjukkan pada Tabel 6.10 dan Tabel 6.11, dapat disimpulkan bahwa sistem telah berjalan sesuai dengan perhitungan manual yang dilakukan oleh peneliti.

6.2.2 Pengujian *Usability*

Pengujian *usability* pada pengembangan aplikasi ini menggunakan metode *System Usability Scale* (SUS) dengan jumlah responden sebanyak 5 orang. Responden yang dimaksud adalah pengguna yang berumur 21 – 24 tahun, aktif menggunakan *smartphone* Android dan sering menyewa lapangan tenis setidaknya sebulan sekali. Perhitungan skor SUS dalam pengujian kali ini memiliki beberapa aturan yang harus dipenuhi yaitu sebagai berikut.

1. Setiap pernyataan dengan nomor ganjil, skor yang diberikan oleh responden dikurangi dengan 1.
2. Setiap pernyataan dengan nomor genap, skor akhir didapat dengan cara 5 dikurangi skor dari jawaban responden.
3. Hasil dari seluruh nilai ganjil dan genap dijumlahkan dan kemudian dikalikan dengan 2,5.

Berikut merupakan hasil dari pernyataan dan jawaban yang berasal dari 5 responden yang telah dipilih dapat diamati pada Tabel 6.12.

Tabel 6.12 Hasil Kuesioner Responden

No	Pertanyaan	Jawaban				
		STS	TS	N	S	SS
1	Saya pikir saya akan sering menggunakan aplikasi ini		3	1		1
2	Saya berpikir aplikasi ini sangat rumit	4	1			
3	Menurut saya, aplikasi ini cukup mudah digunakan				2	3
4	Saya membutuhkan bantuan orang lain untuk menggunakan aplikasi ini	2	2	1		
5	Saya merasa fungsi dalam aplikasi ini telah terintegrasi dengan baik	1			1	3
6	Saya merasa banyak hal yang tidak konsisten dalam aplikasi ini	1		3	1	
7	Saya pikir kebanyakan orang lain akan mudah menggunakan aplikasi ini			2	2	1
8	Saya merasa aplikasi ini sering membingungkan	1	3	1		
9	Saya merasa sangat yakin dapat menggunakan aplikasi ini			1	2	2
10	Saya harus mempelajari banyak hal sebelum menggunakan aplikasi ini	1	2	2		

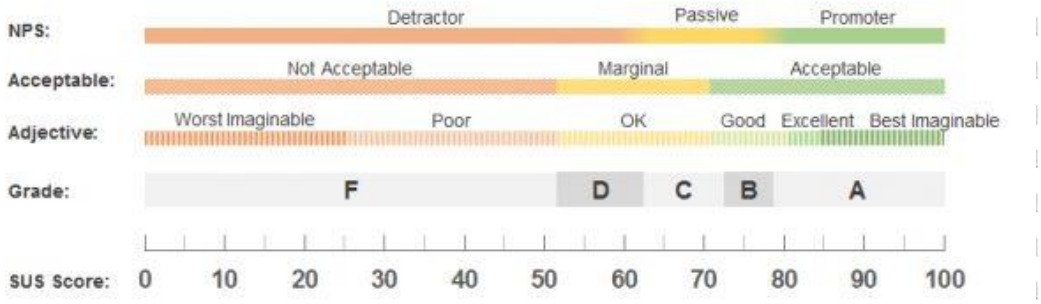
Setelah didapat data dari responden, lalu dilakukan perhitungan rata-rata skor SUS dari setiap peniliaian responden. Hasil akhir dari skor SUS yang telah dihitung dapat dilihat pada Tabel 6.13.

Tabel 6.13 Hasil Perhitungan Skor SUS

Responden	Total Penilaian	Hasil Akhir
1	30	30 * 2.5 = 75
2	39	39 * 2.5 = 97,5
3	20	20 * 2.5 = 50
4	26	26 * 2.5 = 65
5	32	32 * 2.5 = 80
Total Skor		367,5

Rata-rata	73,5

Pada Tabel 6.13 didapat jumlah rata-rata skor dari keseluruhan hasil akhir pengujian *usability* adalah 73,5. Penilaian akhir dengan pengujian SUS menggunakan sebuah *Grade Ranking* sebagai parameter dalam menentkan kelas aplikasi yang dikembangkan dapat dilihat pada Gambar 6.1 di bawah ini.



Gambar 6.1 *Grade Ranking* SUS

Sumber: Sauro (2018)

Berdasarkan Gambar 6.1, maka didapat bahwa sistem yang telah dibangun mendapat nilai di range *Grade* C pada *Grade Scale*, *Good* pada baris *Adjective*, dan *Acceptable* pada baris *Acceptable*. Dari hasil tersebut dapat disimpulkan jika aplikasi sudah cukup baik dan dapat diterima oleh pengguna.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil dari penelitian yang telah dilakukan dalam pengembangan aplikasi rekomendasi lapangan tenis di kota Malang, maka dapat ditarik kesimpulan sebagai berikut:

1. Sistem rekomendasi lapangan tenis dirancang dengan menetapkan algoritma TOPSIS dan BORDA sebagai sistem pendukung keputusan. Perancangan aplikasi ini juga menjabarkan tentang perancangan algoritme, *sequence diagram, class diagram,* basis data dan perancangan antarmuka yang dibuat berdasarkan analisis kebutuhan yang telah disiapkan.
2. Sistem rekomendasi lapangan tenis dapat diimplementasikan secara baik dan dapat berjalan secara maksimal. Sistem ini dibangun menggunakan Android studio sebagai perangkat lunak utama serta basis data Firebase sebagai sumber datanya. TOPSIS dan BORDA dapat berjalan dengan baik dalam penerapan GDSS pada sistem ini. Sistem ini telah berhasil menjalankan aplikasi pihak ketiga seperti Google Maps untuk mengetahui lokasi pengguna dengan menerapkan metode LBS.
3. Sistem rekomendasi lapangan tenis mendapat nilai grade C pada pengujian *usability*, yang artinya sistem sudah cukup baik diterima oleh pengguna. Selain itu, pengujian sistem dinyatakan valid untuk semua fungsi yang dijalankan melalui pengujian *black-box* serta pengujian algoritme.

7.2 Saran

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa saran untuk penelitian selanjutnya. Saran-saran yang perlu dilakukan adalah sebagai berikut:

1. Penggunaan metode lain seperti metode SAW untuk pengembangan sistem pendukung keputusan ke depannya agar dapat mengetahui hasil rekomendasi yang lebih akurat lagi.
2. Pengembangan fitur yang lebih memudahkan untuk pengguna seperti fitur dukungan lebih dari satu bahasa dan fitur dukungan untuk ukuran layar *smartphone* yang berbeda.
3. Pembuatan layar antarmuka yang lebih menarik.

DAFTAR REFERENSI

Belka, T. (2004). Designing recommender systems for tourism. *Designing recommender systems for tourism*.

Bhutia, P., & Phipon, R. (2012). Application of AHP and TOPSIS Method for Supplier Selection Problem. *IOSR Journal of Engineering*, 43-50.

H, N. S. (2011). *Pemrograman Aplikasi Mobile Smartphone dan Tablet Berbasis Android.* Bandung: Informatika Bandung.

Hermawan, S., & Stephanus. (2011). *Mudah Membuat Aplikasi Android.* Yogyakarta: Andi Offset.

Budiyanto, H., Dewi, R., & Sutrisno, S. (2019). Implementasi TOPSIS Pada Sistem Rekomendasi Pemilihan Lapangan Tenis Di Malang Berbasis Lokasi. Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer, 3(5), 4391-4396. Diambil dari https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/5217.

Insights, N. F. (2020). *Sports Fandom in Asia.* Nielsen Sports.

Kurniasih, D. (2013). *Sistem Pendukung Keputusan Pemilihan Laptop Dengan Metode TOPSIS.* Medan: STMIK Budi Darma.

Lardner, R. (2003). *Pedoman Lengka Bermain Tenis.* Semarang: Dahara Prize.

Limaye, M. (2009). *Software Testing: Principles, Techniques anda Tools*. New Delhi: Tata McGraw-Hill Education.

Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to Recommender Systems Handbook. In *Introduction to Recommender Systems Handbook* (p. Chapter 1).

Rickyanto, I. (2003). *Dasar Pemrograman Berorientasi Objek Dengan Java 2 (JDK 1.4)*. Yogyakarta.

Russel, N. (2007). Complexity of Control of Borda Count Elections. *MSCS Thesis. Rochester Institute of Technology*.

Saharuddin. (2012). Sistem Pendukung Keputusan Kelompok Dengan Metode TOPSIS dan Borda Untuk Penentuan Peringkat Terbaik Sekolah Menengah Atas (SMA) (Studi kasus : Dinas Pendidikan Pemuda dan Olahraga Kabupaten Pangkep Sulawesi Selatan).

Sauro, J. (2018, September 19). Retrieved from 5 WAYS TO INTERPRET A SUS SCORE: https://measuringu.com/interpret-sus-score/

Sharda, N. (2010). *Tourism Informatics: Visual Travel Recommender Systems, Social Communities, and User Interface Design.* New York: Information Science Reference.

Sparague, R., & Watson, H. (1993). *Decision Support Systems: Putting Theory Into Practice.* Englewood Clifts: Prentice Hall.

Subakti, I. (2002). *Sistem Pendukung Keputusan (Decision Support System).* Surabaya: Institut Teknologi Sepuluh November.

Turban, E., & Aronson, J. (2001). *Decision Support Systems and Intelligent Systems. 6th edition.* Prentice Hall: Upper Saddle River, NJ.

Ungkawa, U., Rosmala, D., & Aryanti, F. (2013). *Pembangunan Aplikasi Travel Recommender Dengan Metode Case Base Reasoning*.

Wikipedia. (n.d.). *Android*. Retrieved from Wikipedia.org: https://id.wikipedia.org/wiki/Android_(sistem_operasi) [Diakses 10 Februari 2020]

Yuliana. (2013). *Location Based Service.* Retrieved from http://yuliana.lecturer.pens.ac.id/Android/Materi/9.%20Google%20Map/LBS%202013.pdf [10 Februari 2020]

LAMPIRAN A DATA LAPANGAN TENIS

Nama Lapangan	Jarak	Harga / Jam	Rating	Sifat	Jumlah lapangan	Alamat
Lapangan Tenis PELTI	2.5495	15000	5	Umum	4	Jl. Surabaya No.3, Gading Kasri, Klojen, Kota Malang, Jawa Timur 65115
Lapangan Tenis Permata Jingga	2.564	25000	3.6	Umum	2	Jl. Raya Permata Jingga I No.1, Tunggulwulung, Kec. Lowokwaru, Kota Malang, Jawa Timur 65149
Lapangan Tenis Istana Dieng Club House	3.1252	75000	4.3	Umum	2	Jl. Istana Dieng Utara I No.22, Bandulan, Sukun, Kota Malang, Jawa Timur 65146
Lapangan Tenis Puncak Dieng	2.5764	30000	3.7	Umum	2	Jl. Puncak Dieng, Kunci, Kalisongo, Dau, Malang, Jawa Timur 65151
Lapangan Tenis Gajayana	3.6927	20000	4.7	Umum	5	Jl. Tenes No.30, Kauman, Klojen, Kota Malang, Jawa Timur 65119
Lapangan Tenis Jas Dam Brawijaya	5.4109	25000	4.2	Umum	2	Jl. Untung Suropati Utara, Kesatrian, Blimbing, Kota Malang, Jawa Timur 65126
Lapangan Tenis Penjara	4.0187	5000	5	Bukan umum	1	Jl. Taman Pemasyarakatan No.17, Bunulrejo, Blimbing, Kota Malang, Jawa Timur 65126
Lapangan Tenis Top	5.786	30000	4.1	umum	2	Jl. Andalas Sel., Kasin, Klojen, Kota Malang, Jawa Timur 65117
Lapangan Tenis Cakrawala UM	2.0374	20000	5	Bukan umum (harus izin)	2	Universitas Negeri Malang, Jl. Cakrawala, Sumbersari, Kec. Lowokwaru, Kota Malang, Jawa Timur 65145

Lapangan Tenis Yon Bek-Ang	5.2033	25000	4.8	Umum	1	Jl. Narotama Barat, Kesatrian, Blimbing, Kota Malang, Jawa Timur
Lapangan Tenis Wijaya Kusuma	3.3477	20000	4.8	Bukan umum (harus izin)	2	Jl. Wijaya Kusuma No.18, Lowokwaru, Kec. Lowokwaru, Kota Malang, Jawa Timur 65141
Lapangan Tenis Alumunium	4.9294	25000	4	Umum	1	Purwantoro, Blimbing, Kota Malang, Jawa Timur
Lapangan Tenis Blimbing	5.8459	25000	4	Umum	1	Pondok Blimbbing Indah Blok N Malang, Purwodadi, Blimbing, Kota Malang, Jawa Timur
Lapangan Tenis Araya	7.0259	30000	4.3	Umum	2	Boro Teronggo, Tirtomoyo, Pakis, Malang, Jawa Timur 65154
Lapangan Tenis Indoor UM	2.001	30000	4.5	Bukan umum (harus izin)	1	Universitas Negeri Malang Jl. Cakrawala, Sumbersari, Lowokwaru, Malang City, East Java 65145
Lapangan Tenis UB SC	1.8107	30000	4.4	Umum	2	Jl. Terusan Cibogo No.1, Penanggungan, Klojen, Kota Malang, Jawa Timur 65113
Lapangan Tenis Rampal	7.2132	20000	4.5	Bukan umum (hanya atlet)	2	Jl. Urip Sumoharjo, Kesatrian, Blimbing, Kota Malang, Jawa Timur 65126

LAMPIRAN B CLASS DIAGRAM