IMPLEMENTASI TOPSIS PADA SISTEM REKOMENDASI KAFE DI KOTA MALANG BERBASIS LOKASI

SKRIPSI

Untuk memenuhi sebagian persyaratan   
memperoleh gelar Sarjana Komputer

Disusun oleh:

Wiandono Saputro

NIM: 155150201111111



PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2019

PENGESAHAN

IMPLEMENTASI TOPSIS PADA SISTEM REKOMENDASI KAFE DI KOTA MALANG BERBASIS LOKASI

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan

memperoleh gelar Sarjana Komputer

Disusun Oleh :

Wiandono Saputro

NIM: 155150201111111

Skripsi ini telah diuji dan dinyatakan lulus pada

13 Mei 2019

Telah diperiksa dan disetujui oleh:

|  |  |
| --- | --- |
| Dosen Pembimbing I  Ratih Kartika Dewi, S.T., M.Kom  NIK: 201503 890520 2 001 | Dosen Pembimbing II  Candra Dewi S.Kom., M.Sc  NIP: 19771114 200312 2 001 |

Mengetahui

Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D

NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

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

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

Malang, 12 April 2019

­

Wiandono Saputro

NIM: 155150201111111

PRAKATA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga skripsi yang berjudul “Implementasi TOPSIS Pada Sistem Rekomendasi Kafe Berbasis Lokasi” ini dapat terselesaikan untuk memenuhi sebagian persyaratan dalam memperoleh gelar Sarjana Komputer.

Penyusunan skripsi ini tidak akan berhasil tanpa bantuan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D. selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
2. Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Ketua Program Studi Teknik Informatika Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
3. Ibu Ratih Kartika Dewi, S.T., M.Kom dan Ibu Candra Dewi S.Kom., M.Sc selaku dosen pembimbing skripsi yang selalu membimbing dan menuntun penulis hingga terselesaikannya skripsi ini.
4. Ayahanda, Ibunda, dan seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya laporan ini.
5. Teman-teman kontrakan 5B-2 Moh. Rizky Hidayatullah, M. Rafsan Adhizar, Reihan Siregar, Paqsy Pria Islami, dan keluarga besar hamster yang selalu menghibur penulis dikala tidak bisa melanjutkan skripsi.
6. Teman-teman Naoyuki Academic Center yang selalu menjadi pengingat betapa tidak sempurnanya penulis dan menjadi pendorong penulis untuk menggapai mimpi.
7. Seluruh civitas akademika Teknik Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penyelesaian skripsi ini.

Penulis menyadari bahwa skripsi ini memiliki banyak kekurangan dan ketidaksempurnaan. Untuk itu penulis mengharapkan kritik dan saran yang dapat membangun. Semoga skripsi ini dapat memberikan manfaat baik bagi perkembangan ilmu pengetahuan dan teknologi.

Malang, 12 April 2019

Penulis

wiandonosaputro@gmail.com

ABSTRAK

**Wiandono Saputro, Implementasi TOPSIS Pada Sistem Rekomendasi Kafe Berbasis Lokasi**

**Pembimbing: Ratih Kartika Dewi, S.T., M.Kom, Candra Dewi S.Kom., M.Sc**

Di kota Malang, terdapat setidaknya 400 kafe dan restoran yang dapat dikunjungi mahasiswa. Banyaknya jumlah kafe yang tersedia ini menimbulkan masalah tersendiri bagi mahasiswa saat sedang mencari tempat berkumpul untuk refreshing atau mengerjakan tugas akhir, yaitu memilih kafe mana yang ingin dikunjungi. Terdapat beberapa faktor yang menjadi penentu dipilihnya sebuah kafe seperti rating, jarak, harga, dan fasilitas. Untuk memecahkan permasalahan ini dibutuhkan sebuah sistem yang dapat memberikan rekomendasi kafe kepada penggunanya berdasarkan lokasi dan kriteria kafe yang sedang ingin dicari juga memberikan informasi seperti harga, menu, biji kopi yang digunakan, metode seduh yang digunakan dan petunjuk arah menuju kafe.

Sistem rekomendasi kafe yang dikembangkan menggunakan metode TOPSIS. Metode TOPSIS digunakan untuk menghasilkan peringkat alternatif kafe berdasarkan pada konsep bahwa alternatif terbaik bukan hanya yang memiliki jarak terpendek dari solusi ideal positif tapi juga jarah terjauh dari solusi ideal negatif. Data alternatif disimpan menggunakan layanan basis data Firebase Realtime Database, dan petunjuk arah diberikan menggunakan aplikasi pihak ketiga yaitu Google Maps.

Dari hasil pengujian didapatkan nilai 100% valid pada pengujian fungsional, 100% valid pada pengujian validasi algoritme yang membandingkan hasil keluaran sistem dengan perhitungan manual, dan tidak adanya pembalikan peringkat pada pengujian *rank consistency* yang dilakukan dengan menambahkan alternatif baru, dan pengujian *usability* berdasarkan *usefulness, ease of use, ease of learning, dan satisfaction* yang menghasilkan persentase sebesar 80.55%.

Kata kunci: kafe, lokasi, TOPSIS, firebase, agile, DSDM

ABSTRACT

**Wiandono Saputro, Implementasi TOPSIS Pada Sistem Rekomendasi Kafe Berbasis Lokasi**

**Supervisors: Ratih Kartika Dewi, S.T., M.Kom, Candra Dewi S.Kom., M.Sc**

*In Malang, there are at least 400 cafes and restaurants that college students can visit. A large number of cafes available creates its own problems for college students while looking for a gathering place for refreshing or doing a final assignment, which is choosing which cafe to visit. There are several factors that determine the choice of a cafe such as rating, distance, price, and facilities. To solve this problem, we need a system that can provide cafe recommendations to its users based on the location and the criteria of the cafe they are looking for, also provides information such as prices, menus, coffee beans, brewing methods and directions to the cafe.*

*The cafe recommendation system developed using the TOPSIS. The TOPSIS method is used to generate alternative of cafe based on the concept that the best alternative is not only the the alternative with the shortest distance from the positive ideal solution but also with the farthest distance from the negative ideal solution. Alternative data is stored using the Firebase Realtime Database database service, and directions are given using third-party applications, namely Google Maps.*

*From the testing the system obtained 100% valid on functional testing, 100% valid in the validation of algorithm that compares the system output with manual calculation, and the absence of ranking reversal on rank consistency testing which is done by adding new alternatives, and testing usability based on usefulness, ease of use, ease of learning, and satisfaction which give results in a percentage of 80.55%.*

Keyword: cafe, location, TOPSIS, firebase, agile, DSDM

DAFTAR ISI

[PENGESAHAN ii](#_Toc8864228)

[PERNYATAAN ORISINALITAS iii](#_Toc8864229)

[PRAKATA iv](#_Toc8864230)

[ABSTRAK v](#_Toc8864231)

[ABSTRACT v](#_Toc8864232)

[DAFTAR ISI vii](#_Toc8864233)

[DAFTAR TABEL x](#_Toc8864234)

[DAFTAR GAMBAR xii](#_Toc8864235)

[BAB 1 PENDAHULUAN 1](#_Toc8864236)

[1.1 Latar belakang 1](#_Toc8864237)

[1.2 Rumusan masalah 2](#_Toc8864238)

[1.3 Tujuan 2](#_Toc8864239)

[1.4 Manfaat 2](#_Toc8864240)

[1.5 Batasan Masalah 2](#_Toc8864241)

[1.6 Sistematika Pembahasan 3](#_Toc8864242)

[BAB 2 LANDASAN KEPUSTAKAAN 5](#_Toc8864243)

[2.1 Kajian Pustaka Sistem Rekomendasi 5](#_Toc8864244)

[2.2 Sistem Rekomendasi 6](#_Toc8864245)

[2.2.1 Multiple-criteria Decision-making 6](#_Toc8864246)

[2.3 TOPSIS 6](#_Toc8864247)

[2.3.1 Langkah-Langkah TOPSIS 7](#_Toc8864248)

[2.4 Firebase 8](#_Toc8864249)

[2.4.1 Cloud Storage 9](#_Toc8864250)

[2.4.2 Realtime Database 9](#_Toc8864251)

[2.5 Pengukuran Jarak 9](#_Toc8864252)

[2.6 Pengujian Perangkat Lunak 10](#_Toc8864253)

[2.6.1 Pengujian Fungsional 10](#_Toc8864254)

[2.6.2 Pengujian Validasi Algoritme 10](#_Toc8864255)

[2.6.3 Pengujian *Rank Consistency* 10](#_Toc8864256)

[2.6.4 Pengujian *Usability* 11](#_Toc8864257)

[BAB 3 METODOLOGI PENELITIAN 12](#_Toc8864258)

[3.1 Tipe Penelitian 12](#_Toc8864259)

[3.2 Strategi Penelitian 12](#_Toc8864260)

[3.3 Subjek, Lokasi, dan Data Penelitian 13](#_Toc8864261)

[3.4 Metode Pengumpulan Data 13](#_Toc8864262)

[3.5 Peralatan Pendukung 13](#_Toc8864263)

[3.6 Metode Pengembangan Perangkat Lunak 13](#_Toc8864264)

[*3.6.1 Kick-Off* 14](#_Toc8864265)

[*3.6.2 Investigation* 14](#_Toc8864266)

[*3.6.3 Refinement* 14](#_Toc8864267)

[*3.6.4 Consolidation* 14](#_Toc8864268)

[*3.6.5 Close-out* 15](#_Toc8864269)

[BAB 4 PERANCANGAN 16](#_Toc8864270)

[4.1 Analisis Kebutuhan 17](#_Toc8864271)

[4.1.1 Gambaran Aplikasi 17](#_Toc8864272)

[4.1.2 Identifikasi Aktor 18](#_Toc8864273)

[4.1.3 Analisis Kebutuhan Fungsional 18](#_Toc8864274)

[4.1.4 Analisis Kebutuhan Non-Fungsional 19](#_Toc8864275)

[4.2 Pemodelan Kebutuhan 19](#_Toc8864276)

[4.2.1 Use Case Diagram 19](#_Toc8864277)

[4.2.2 Use Case Scenario 20](#_Toc8864278)

[4.3 Perancangan Aplikasi 23](#_Toc8864279)

[4.3.1 Perancangan Sequence Diagram 23](#_Toc8864280)

[4.3.2 Perancangan Diagram Kelas 26](#_Toc8864281)

[4.3.3 Perancangan Data 31](#_Toc8864282)

[4.4 Perancangan Algoritme 33](#_Toc8864283)

[4.4.1 Perancangan TOPSIS 33](#_Toc8864284)

[4.4.2 Perancangan Law of Cosine 37](#_Toc8864285)

[4.4.3 Perhitungan Manual Law of Cosine 38](#_Toc8864286)

[4.4.4 Perhitungan Manual TOPSIS 39](#_Toc8864287)

[4.5 Perancangan Antarmuka 42](#_Toc8864288)

[4.5.1 Antarmuka Beranda 42](#_Toc8864289)

[4.5.2 Antarmuka Rekomendasi 42](#_Toc8864290)

[4.5.3 Antarmuka Detail 43](#_Toc8864291)

[BAB 5 IMPLEMENTASI 44](#_Toc8864292)

[5.1 Spesifikasi Sistem 44](#_Toc8864293)

[5.1.1 Spesifikasi Perangkat Keras 44](#_Toc8864294)

[5.1.2 Spesifikasi Perangkat Lunak 45](#_Toc8864295)

[5.2 Batasan Implementasi 45](#_Toc8864296)

[5.3 Implementasi Basis Data 45](#_Toc8864297)

[5.4 Implementasi Kode Program 47](#_Toc8864298)

[5.4.1 Implementasi Kode Program Memberikan Rekomendasi 47](#_Toc8864299)

[5.4.2 Implementasi Kode Program Algoritme Law of Cosine Distance 50](#_Toc8864300)

[5.4.3 Implementasi Kode Program Algoritme TOPSIS 51](#_Toc8864301)

[5.5 Implementasi Antarmuka 53](#_Toc8864302)

[5.5.1 Antarmuka Beranda 53](#_Toc8864303)

[5.5.2 Antarmuka Rekomendasi 54](#_Toc8864304)

[5.5.3 Antarmuka Detail 55](#_Toc8864305)

[BAB 6 PENGUJIAN 56](#_Toc8864306)

[6.1 Pengujian Fungsional 56](#_Toc8864307)

[6.2 Pengujian Non-Fungsional 58](#_Toc8864308)

[6.2.1 Pengujian Validasi Algoritme 58](#_Toc8864309)

[6.2.2 Pengujian Rank Consistency 59](#_Toc8864310)

[6.2.3 Pengujian Usability 59](#_Toc8864311)

[6.3 Analisis Hasil Pengujian 61](#_Toc8864312)

[6.3.1 Analisis Pengujian Fungsional 61](#_Toc8864313)

[6.3.2 Analisis Pengujian Non-Fungsional 62](#_Toc8864314)

[BAB 7 PENUTUP 66](#_Toc8864315)

[7.1 Kesimpulan 66](#_Toc8864316)

[7.2 Saran 66](#_Toc8864317)

[DAFTAR RUJUKAN 67](#_Toc8864318)

DAFTAR TABEL

[Tabel 2.1 Penelitian Terdahulu 5](#_Toc8864319)

[Tabel 2.2 Skor Penilaian Jawaban Skala Likert 11](#_Toc8864320)

[Tabel 4.1 Identifikasi Aktor 18](#_Toc8864321)

[Tabel 4.2 Spesifikasi Kebutuhan Fungsional 18](#_Toc8864322)

[Tabel 4.3 Spesifikasi Kebutuhan Non-Fungsional 19](#_Toc8864323)

[Tabel 4.4 Skenario Menampilkan Beranda 20](#_Toc8864324)

[Tabel 4.5 Skenario Mendapatkan Rekomendasi 21](#_Toc8864325)

[Tabel 4.6 Skenario Menampilkan Detail Kafe 21](#_Toc8864326)

[Tabel 4.7 Skenario Menambahkan Kafe Ke Daftar Favorit 22](#_Toc8864327)

[Tabel 4.8 Skenario Menghapus Kafe Dari Daftar Favorit 22](#_Toc8864328)

[Tabel 4.9 Skenario Mendapatkan Petunjuk Arah 23](#_Toc8864329)

[Tabel 4.10 Perancangan Data JSON 31](#_Toc8864330)

[Tabel 4.11 Perancangan Data Firebase Storage 33](#_Toc8864331)

[Tabel 4.12 Data Lokasi 38](#_Toc8864332)

[Tabel 4.13 Sampel Data Alternatif 39](#_Toc8864333)

[Tabel 4.14 Nilai Pembagi Alternatif Kriteria 39](#_Toc8864334)

[Tabel 4.15 Hasil Normalisasi Matriks Alternatif 39](#_Toc8864335)

[Tabel 4.16 Hasil Solusi Ideal Positif dan Negatif 40](#_Toc8864336)

[Tabel 4.17 Hasil Perhitungan Jarak Terhadap Solusi Ideal Positif dan Negatif 40](#_Toc8864337)

[Tabel 4.18 Hasil Perhitungan Nilai Preferensi 41](#_Toc8864338)

[Tabel 4.19 Hasil Perangkingan 41](#_Toc8864339)

[Tabel 5.1 Spesifikasi Perangkat Keras 44](#_Toc8864340)

[Tabel 5.2 Spesifikasi Emulator 45](#_Toc8864341)

[Tabel 5.3 Spesifikasi Perangkat Lunak 45](#_Toc8864342)

[Tabel 6.1 Kasus Uji Menampilkan Beranda 56](#_Toc8864343)

[Tabel 6.2 Kasus Uji Mendapatkan Rekomendasi 57](#_Toc8864344)

[Tabel 6.3 Kasus Uji Menampilkan Detail Kafe 57](#_Toc8864345)

[Tabel 6.4 Kasus Uji Menambahkan Kafe Ke Daftar Favorit 57](#_Toc8864346)

[Tabel 6.5 Kasus Uji Menghapus Kafe Dari Daftar Favorit 58](#_Toc8864347)

[Tabel 6.6 Kasus Uji Mendapatkan Petunjuk Arah 58](#_Toc8864348)

[Tabel 6.7 Perbandingan Hasil Perhitungan 59](#_Toc8864349)

[Tabel 6.8 Daftar Kafe Pengujian *Rank Consistency* 59](#_Toc8864350)

[Tabel 6.9 Rekapitulasi Kuesioner Pengujian *Usability* 60](#_Toc8864351)

[Tabel 6.10 Rekapitulasi Kuesioner Pengujian *Usability* (Lanjutan) 61](#_Toc8864352)

[Tabel 6.11 Hasil Pengujian Fungsional 61](#_Toc8864353)

[Tabel 6.12 Hasil Pengujian Validasi Algoritme 62](#_Toc8864354)

[Tabel 6.13 Interpretasi Skor *Likert* 63](#_Toc8864355)

[Tabel 6.14 Hasil Perhitungan Pengujian Usability 64](#_Toc8864356)

[Tabel 6.15 Hasil Status Pengujian *Usability* 65](#_Toc8864357)

DAFTAR GAMBAR

[Gambar 3.1 Metode Penelitian 12](#_Toc8864358)

[Gambar 3.2 Struktur DSDM 14](#_Toc8864359)

[Gambar 4.1 Diagram Perancangan 16](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864360)

[Gambar 4.2 Gambaran Umum Sistem 17](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864361)

[Gambar 4.3 Use Case Diagram Kebutuhan Fungsional 20](#_Toc8864362)

[Gambar 4.4 Sequence Diagram Menampilkan Beranda 23](#_Toc8864363)

[Gambar 4.5 Sequence Diagram Mendapatkan Rekomendasi 24](#_Toc8864364)

[Gambar 4.6 Sequence Diagram Menampilkan Detail Kafe 25](#_Toc8864365)

[Gambar 4.7 Sequence Diagram Menambahkan Kafe Ke Daftar Favorit 25](#_Toc8864366)

[Gambar 4.8 Sequence Diagram Menghapus Kafe Dari Daftar Favorit 26](#_Toc8864367)

[Gambar 4.9 Diagram Kelas Sistem 26](#_Toc8864368)

[Gambar 4.10 Class Diagram Package Model 27](#_Toc8864369)

[Gambar 4.11 Class Diagram Package View 27](#_Toc8864370)

[Gambar 4.12 Diagram Kelas Package Presenter 28](#_Toc8864371)

[Gambar 4.13 Diagram Kelas Activity 28](#_Toc8864372)

[Gambar 4.14 Diagram Kelas Package Adapter 29](#_Toc8864373)

[Gambar 4.15 Diagram Kelas Package Util 30](#_Toc8864374)

[Gambar 4.16 Flowchart TOPSIS 33](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864375)

[Gambar 4.17 Flowchart TOPSIS 33](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864376)

[Gambar 4.18 *Flowchart* Normalisasi 35](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864377)

[Gambar 4.19 *Flowchart* Solusi Ideal Positif 35](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864378)

[Gambar 4.20 *Flowchart* Solusi Ideal Negatif 36](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864379)

[Gambar 4.21 *Flowchart* Jarak Matriks Keputusan Dengan Matriks 37](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864380)

[Gambar 4.22 *Flowchart* Kedekatan Relatif Dengan Solusi Ideal 37](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864381)

[Gambar 4.23 *Flowchart* Algoritme Haversine 38](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864382)

[Gambar 4.24 Antarmuka Beranda 42](#_Toc8864383)

[Gambar 4.25 Antarmuka Rekomendasi 42](#_Toc8864384)

[Gambar 4.26 Antarmuka Detail 43](#_Toc8864385)

[Gambar 5.1 Diagram Implementasi 44](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864386)

[Gambar 5.2 Implementasi Basis Data *Child* cafe di Firebase Realtime Database 46](#_Toc8864387)

[Gambar 5.3 Implementasi Basis Data *Child* menu di Firebase Realtime Database 47](#_Toc8864388)

[Gambar 5.4 Implementasi Antarmuka Beranda 54](#_Toc8864389)

[Gambar 5.5 Implementasi Antarmuka Rekomendasi 54](#_Toc8864390)

[Gambar 5.6 Implementasi Antarmuka Detail 55](#_Toc8864391)

[Gambar 6.1 Diagram Pengujian 56](file:///C:\Users\Theta\Dropbox\College\Semester%207\Induksi%20Riset\IMPLEMENTASI%20TOPSIS%20PADA%20SISTEM%20REKOMENDASI%20KAFE%20DI%20KOTA%20MALANG%20BERBASIS%20LOKASI.docx#_Toc8864392)

[Gambar 6.2 Peringkat Awal 63](#_Toc8864393)

[Gambar 6.3 Peringkat Setelah Penambahan Alternatif 63](#_Toc8864394)

# PENDAHULUAN

## Latar belakang

Kata kafe berasal dari bahasa Perancis, yang secara harfiah memiliki arti kopi, itulah alasan kenapa kata kafe identik dengan tempat minum kopi. Seperti yang dimuat pada Kamus Besar Bahasa Indonesia (KBBI), kafe juga memiliki pengertian tempat makan berkonsep sederhana, yang menyajikan minuman dan makanan ringan. Jadi dapat disimpulkan bahwa kafe adalah tempat kumpul yang selain menyajikan kopi tapi juga minuman dan makanan ringan.

Di Kota Malang, terdapat beragam jenis kafe menjamur di penjuru kotanya. Hal ini terjadi bukan tanpa sebab, pada tahun 2016 sendiri ada lebih dari 131.000 mahasiswa yang sedang menuntut ilmu di kota Malang dan jumlah ini terus bertambah sekitar 5-10% setiap tahun (Suryamalang, 2017). Selain jumlah mahasiswa yang banyak, pada tahun 2017 juga setidaknya ada 400 kafe dan restoran (Malang Post, 2017).

Namun dari banyaknya jumlah kafe yang tersedia ini justru menimbulkan masalah tersendiri bagi mahasiswa saat sedang mencari tempat berkumpul untuk *refreshing* atau mengerjakan tugas akhir, yaitu memilih kafe mana yang tepat. Dalam memilih kafe, ada beberapa hal yang menjadi pertimbangan, salah satunya adalah jarak, jarak sedikit berbeda jika dibandingkan dengan kriteria lainnya seperti harga, rating, dan fasilitas yang dapat pengguna sesuaikan tanpa membutuhkan pengetahuan terkait konteks lingkungan pengguna. Maka dari itu lokasi pengguna sangat penting pada proses pemberian rekomendasi.

Permasalahan ini dapat diselesaikan dengan sebuah solusi mengembangkan sistem rekomendasi kafe berbasis lokasi. Karena sistem yang dikembangkan membutuhkan data berupa lokasi, maka Android dipilih sebagai *platform* pengembangan. Keputusan ini diambil karena data lokasi menjadi lebih relevan dan akurat jika diambil melalui perangkat bergerak pengguna dan memudahkan pengguna untuk bisa mendapatkan rekomendasi dari mana saja.

Metode yang digunakan untuk mendapatkan rekomendasi kafe ini adalah TOPSIS. TOPSIS dapat menghasilkan keputusan yang tepat, karena TOPSIS merupakan metode pengambilan keputusan multikriteria (Hamka, Utami, Amborowati, 2014). Selain itu, TOPSIS juga memberikan rekomendasi berdasarkan pada konsep bahwa alternatif terbaik bukan hanya yang memiliki jarak terpendek dari solusi ideal positif tapi juga jarak terpanjang dari solusi ideal negatif untuk dapat menghasilkan rekomendasi yang sesuai dengan yang diharapkan (Kurniasih, 2013). TOPSIS memiliki konsep yang sederhana, mudah dipahami, dan proses komputasi yang sederhana serta mampu mengambil keputusan solusi paling ideal (Riandari, Hasugian, Taufik, 2017).

Berdasarkan penelitian yang telah dilakukan oleh Shih, Shyur, dan Lee (2006) TOPSIS dapat memberikan rekomendasi yang lebih rasional selayaknya manusia, dan mudah diterapkan bahkan kedalam bentuk *spreadsheets*. Juga pada penelitian yang dilakukan oleh Zanakis, Solomon, Wishart, dan Dublish (1998) TOPSIS memiliki peringkat *reversals* paling rendah jika dibandingkan dengan metode lainnya seperti ELECTRE, Multiplicative Exponential Weighting (MEW), Simple Additive Weighting (SAW), dan empat versi AHP (*original, geometric scale and right eigenvector*, dan *mean transformation solution*).

Setelah selesai diimplementasikan maka perlu dilakukan pengujian sistem demi memastikan sistem yang dihasilkan dapat bekerja sesuai dengan apa yang diharapkan, dan memberikan hasil rekomendasi yang sesuai dengan metode yang digunakan. Pengujian yang dilakukan antara lainnya adalah, pengujian fungsional, pengujian validasi algoritme, pengujian *rank consistency*,dan pengujian *usability*.

Berdasarkan pemaparan diatas, judul penelitian yang diambil adalah “Implementasi TOPSIS Pada Sistem Rekomendasi Kafe di Kota Malang Berbasis Lokasi” dengan harapan perangkat lunak yang dibangun dapat menyelesaikan permasalahan memilih kafe di Kota Malang.

## Rumusan masalah

Berdasarkan permasalahan yang diangkat pada bagian latar belakang, maka didapatkan rumusan masalah sebagai berikut:

1. Bagaimana hasil pengujian dari sistem rekomendasi kafe di kota Malang setelah diimplementasikan?

## Tujuan

Tujuan dari penelitian ini adalah:

1. Melakukan pengujian terhadap sistem rekomendasi kafe di kota Malang

## Manfaat

Hasil penelitian ini diharapkan dapat memberikan manfaat kepada masyarakat, yaitu:

Memberikan informasi kepada masyarakat berupa rekomendasi kafe dengan detail informasi berupa nama kafe, alamat, rating, menu, jarak, harga, dan fasilitas yang tersedia

Memberikan ruang publikasi baru kepada pemilik kafe melalui aplikasi yang dapat diakses pengguna

## Batasan Masalah

Agar penelitian ini dapat terfokus dan untuk menghindari meluasnya permasalahan, maka dari rumusan masalah yang telah dipaparkan, batasan masalah yang ditentukan adalah sebagai berikut:

Kriteria yang digunakan untuk merekomendasikan kafe adalah jarak, rating, harga, dan fasilitas

Kondisi internet dan sensor GPS dianggap sudah baik dan dapat digunakan pada aplikasi yang dibangun. Tidak perlu pembahasan perancangan dan implementasi pengaturan internet dan penentuan lokasi pengguna menggunakan sensor GPS

## Sistematika Pembahasan

Sistematika penulisan memberikan gambaran dan uraian dari penyusunan skripsi secara garis besar yang meliputi beberapa bab, antara lain:

**BAB 1 PENDAHULUAN**

Bab ini menguraikan latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika pembahasan mengenai perlunya penelitian tentang “Implementasi Topsis Pada Sistem Rekomendasi Kafe di Kota Malang Berbasis Lokasi”.

**BAB 2 LANDASAN KEPUSTAKAAN**

Bab ini berisi kajian pustaka sistem rekomendasi dan dasar teori mengenai teknologi yang diimplementasikan pada penelitian ini. Kajian pustaka berupa 3 macam penelitian sebelumnya yang berkaitan dengan penelitian yang ingin dikerjakan. Teori yang diambil mengenai sistem rekomendasi, metode TOPSIS, Firebase, pengukuran jarak, dan pengujian perangkat lunak.

**BAB 3 METODOLOGI PENELITIAN**

Bab ini menguraikan tentang metode dan langkah kerja yang dilakukan dalam penulisan skripsi yang terdiri dari studi literatur, pengumpulan data, analisis kebutuhan, perancangan sistem, implementasi sistem, pengujian sistem, dan pengambilan kesimpulan dan saran.

**BAB 4 PERANCANGAN**

Perancangan pada bab ini terdiri atas 2 bagian, yaitu analisis kebutuhan dan perancangan perangkat lunak. Analisis kebutuhan dipresentasikan dengan menggunakan *use case diagram* dan *use case scenario*, sedangkan perancangan perangkat lunak disesuaikan dengan analisis kebutuhan berupa arsitektur sistem, *sequence diagram*, *class diagram*, perancangan basis data, perancangan algoritme, dan perancangan antarmuka.

**BAB 5 IMPLEMENTASI**

Bab ini membahas tentang implementasi dari perancangan perangkat lunak. Implementasi terdiri dari spesifikasi sistem, batasan implementasi, implementasi basis data, implementasi *class* dan *assets*, implementasi kode program, dan implementasi antarmuka.

**BAB 6 PENGUJIAN**

Bab ini membahas mengenai teknik dan metode pengujian yang dilakukan pada penelitian ini. Pengujian yang dilakukan pada penelitian ini dengan menggunakan *black box*,pengujian validasi algoritme, pengujian *rank consistency*,dan pengujian akurasi.

**BAB 7 PENUTUP**

Pada bab ini membahas tentang kesimpulan dan saran dari pengerjaan penelitian dengan judul topik “Implementasi TOPSIS Pada Sistem Rekomendasi Kafe di Kota Malang Berbasis Lokasi”.

# LANDASAN KEPUSTAKAAN

## Kajian Pustaka Sistem Rekomendasi

Sebagai pertimbangan dan acuan perbandingan untuk landasan penelitian terdahulu, terdapat beberapa penelitian yang mempunyai sudah menggunakan metode yang sama yaitu TOPSIS. Penelitian terkait pengguna metode TOPSIS untuk mendapatkan rekomendasi dapat dilihat pada Tabel 2.1.

Tabel 2.1 Penelitian Terdahulu

|  |  |  |
| --- | --- | --- |
| No | Penelitian Terdahulu | Uraian |
| 1. | Judul Penelitian | Sistem Pendukung Keputusan Pemilihan Laptop Dengan Metode TOPSIS |
| Tujuan Penelitian | Mengetahui apakah TOPSIS dapat diterapkan untuk mendapatkan rekomendasi laptop.  Menguji penerapan metode TOPSIS pada sistem yang dirancang. |
| Hasil Penelitian | 1. Sistem rekomendasi laptop yang menerapkan metode TOPSIS dalam proses pemberian rekomendasi. 2. Hasil perhitungan metode TOPSIS yang diterapkan sudah sama dengan hasil perhitungan yang dilakukan secara manual. |
| 2. | Judul Penelitian | Sistem Pendukung Keputusan Kelompok Metode TOPSIS dan *Borda* Untuk Penentuan Bakal Calon Haji |
| Tujuan Penelitian | Mengkaji penerapan Sistem Pendukung Keputusan Kelompok untuk penentuan bakal calon haji di Universitas Muhammadiyah Purwokerto. |
| Hasil Penelitian | 1. TOPSIS dapat menghasilkan keputusan yang lebih tepat. 2. *BORDA* mampu menyatukan setiap keputusan yang dihasilkan oleh masing-masing pengambil keputusan. 3. Sistem dapat menyelesaikan permasalahan penentuan bakal calon haji di UMP. |
| 3. | Judul Penelitian | Sistem Pendukung Keputusan Menggunakan Metode TOPSIS Dalam Memilih Kepala Departemen Pada Kantor Balai Wilayah Sungai Sumatera II Medan |
| Tujuan Penelitian | 1. Merancang dan membuat aplikasi sistem pengambilan keputusan pemilihan kepala departemen pada kantor balai wilayah sungai sumatera II Medan. 2. Menerapkan metode TOPSIS pada aplikasi sistem pendukung keputusan pemilihan kepala departemen. |

Tabel 2.1 Penelitian Terdahulu (Lanjutan)

|  |  |  |
| --- | --- | --- |
| No. | Penelitian Terdahulu | Uraian |
| 3. | Hasil Penelitian | 1. Perancangan sistem menggunakan UML, bahasa pemrograman Visual Basic, Microsoft Access 2007, dan metode TOPSIS sebagai metode dalam pembuatan sistem pendukung keputusan. 2. Penerapan metode TOPSIS dengan menentukan kriteria yang digunakan sebagai acuan pengambilan keputusan. |

Dalam penelitian ini, sistem rekomendasi yang dibangun menerapkan metode *Technique for Order Preference by Similarity to Ideal Solution* (TOPSIS). Metode tersebut diharapkan dapat membantu proses pengambilan keputusan dalam memilih kafe yang sesuai dengan kriteria yang diinginkan. Selain itu, dengan aplikasi ini diharapkan juga dapat memberikan informasi kafe dan memberikan petunjuk arah ke lokasi kafe. Parameter yang digunakan dalam merekomendasikan objek kuliner adalah harga, rating, jarak, dan fasilitas.

## Sistem Rekomendasi

Sistem rekomendasi adalah model aplikasi yang memberikan hasil observasi terhadap keadaan dan keinginan pengguna (Darmastuti, 2013). Sistem rekomendasi merupakan sebuat alat personalisasi yang menyediakan pengguna sebuah informasi daftar item yang sesuai dengan keinginannya (Sebastia, Garcia, Onaindia, Guzman, 2009). Sistem rekomendasi menyimpulkan preferensi pengguna dengan menganalisis ketersediaan data, informasi, dan lingkungan pengguna. Proses analisis inilah yang disebut dengan *Multiple-criteria decision analysis* (MCDA) atau *Multiple-criteria decision-making* (MCDM).

### Multiple-criteria Decision-making

*Multiple-criteria decision making* berkaitan dengan situasi pengambilan keputusan dimana pengambil keputusan memiliki beberapa tujuan yang biasanya saling bertentangan (Habenicht, Scheubrein, Scheubrein, 2002). Di dalam permasalahan kehidupan nyata, tidak ada solusi ideal untuk setiap proses pembuatan keputusan, maka hal yang paling penting adalah untuk menemukan kriteria mana yang dapat dikompromikan. Solusi inilah yang memiliki kemungkinan terbaik untuk si pengambil keputusan, maka dari itu hasil keputusan yang dibuat bergantung pada preferensi si pengambil keputusan.

## TOPSIS

TOPSIS adalah metode yang dapat digunakan untuk menyelesaikan permasalahan pengambilan keputusan dengan banyak kriteria. Pertama kali diperkenalkan oleh Yoon dan Hwang pada tahun 1981. TOPSIS menggunakan prinsip bahwa alternatif yang terpilih harus mempunyai jarak terdekat dari solusi ideal positif dan terjauh dari solusi ideal negatif dari sudut pandang geometris dengan menggunakan jarak Euclidean untuk menentukan kedekatan relatif dari suatu alternatif dengan solusi optimal. Solusi ideal positif didefinisikan sebagai jumlah dari seluruh nilai terbaik yang dapat dicapai untuk setiap atribut, sedangkan solusi negatif-ideal terdiri dari seluruh nilai terburuk yang dicapai untuk setiap atribut (Kurniasih, 2013).

### Langkah-Langkah TOPSIS

Membuat matriks keputusan ternormalisasidari matriks keputusanR yang sudah ternormalisasi dengan metode Euclidean.

(2.1)

Dimana:

= hasil normalisasi matriks keputusan R;

i = 1,2,3,…,n;

j = 1,2,3,…,n;

Menentukan solusi ideal positif dan solusi ideal negatif. Solusi ideal positif dinotasikan dengan dan solusi ideal negatif dinotasikan dengan , menentukan solusi ideal (+) dan (-) adalah sebagai berikut:

(2.2)

(2.3)

Dimana:

= elemen matriks baris ke-*i* dan kolom ke-*j*;

= {j = 1,2,3,…,n dan j terkait dengan kriteria yang menguntungkan}

= {j = 1,2,3,…,n dan j terkait dengan kriteria yang merugikan}

Menghitung *separation measure*. *Separation measure* adalah pengukuran jarak dari suatu alternative ke solusi ideal positif dan solusi ideal negatif. Perhitungan matematis untuk *separation measure* ideal positif adalah sebagai berikut:

berhubungan *i* = 1,2,3,…,n

(2.4)

berhubungan *i* = 1,2,3,…,n

(2.5)

Dimana:

= {j = 1,2,3,…,n dan j terkait dengan kriteria yang menguntungkan}

= {j = 1,2,3,…,n dan j terkait dengan kriteria yang merugikan}

Menghitung kedekatan relatif terhadap solusi ideal. Kedekatan relatif dari alternatif dengan solusi ideal direpresentasikan dengan:

(2.6)

dengan 0 < < 1 dan *i* = 1,2,3,…,n

Membuat peringkat alternatif. Alternatif dapat diurutkan berdasarkan urutan . Maka dari itu, alternatif terbaik adalah salah satu yang berjarak terpendek terhadap solusi ideal dan berjarak terjauh dengan solusi ideal negatif.

## Firebase

Firebase adalah teknologi yang memungkinkan pengembang aplikasi untuk membuat aplikasi tanpa pemrograman dari sisi server. Jadi, proses pengembangan aplikasi menjadi lebih mudah dan cepat. Pengembang aplikasi tidak perlu repot-repot memikirkan pengembangan server atau membuat REST API sendiri, hanya dengan sedikit pengaturan pengembang bisa mengandalkan Firebase untuk melakukan hal-hal seperti: autentikasi pengguna, menyimpan data, dan mengimplementasikan aturan akses (Kumar, Akhi, Gunti, Reddy, 2016).

Firebase memiliki banyak produk untuk membantu pengembang aplikasi, terdapat 18 produk yang bisa digunakan pengembang aplikasi untuk melakukan proses pengembangan. 18 produk ini dibagi menjadi 3 bagian yaitu:

*Build Better Apps*, kategori produk ini berisi produk-produk yang diperlukan pengembang untuk memulai mengembangkan aplikasi diawal proses pengembangan, kategori ini terdiri dari beberapa produk sebagai berikut:

Cloud Firestore

ML Kit

Cloud Functions

Authentication

Hosting

Cloud Storage

Realtime Database

*Improve App Quality*, kategori produk ini berisi produk-produk yang dapat memberikan wawasan perihal performa aplikasi yang dikembangkan guna mengetahui bagaimana pengembang harus mengalokasikan sumber daya dengan lebih efektif, kategori ini terdiri dari beberapa produk sebagai berikut:

Crashlytics

Performance Monitoring

Test Lab

*Grow Your Business*, kategori produk ini berisi produk-produk yang dapat digunakan pengembang untuk meningkatkan jumlah pengguna dan meningkatkan tingkat retensi pengguna, kategori ini terdiri dari beberapa produk sebagai berikut:

In-App Messaging

Google Analytics

Predictions

A/B Testing

Cloud Messaging

Remote Config

Dynamic Links

App Indexing

Pada penelitian ini kategori produk yang digunakan adalah kategori yang pertama, karena mengingat penelitian ini adalah implementasi pengembangan. Pada kategori pertama pun hanya produk seperti Cloud Storage, dan Realtime Database saja yang digunakan.

### Cloud Storage

Produk ini bisa digunakan pengembang untuk menjadi tempat penyimpanan berkas statis seperti gambar. Fitur produk ini yang dapat memudahkan pengembang aplikasi dan pengguna adalah sistem download dan upload file yang memperhatikan keadaan konektivitas pengguna, ketika konektivitas perangkat pengguna hilang maka proses upload atau download secara otomatis terhenti dan dilanjutkan kembali ketika konektivitas pengguna telah kembali.

### Realtime Database

Produk ini memungkinkan aplikasi untuk melakukan sinkronisasi data secara *real-time*. Tipe basis data yang digunakan adalah NoSQL, inilah yang memungkinkan dilakukannya proses penyimpanan data dan sinkronisasi data secara *realtime.* Karena bertipe NoSQL maka format penyimpanan datanya adalah JSON.

Produk ini juga dapat melakukan optimasi untuk pengguna yang tidak dapat mengakses basis data karena sedang tidak memiliki koneksi. Aplikasi secara otomatis menggunakan data yang tersimpan di *cache* ketika hal ini terjadi. Sama seperti fitur keamanan yang ada di produk Cloud Storage, Realtime Database juga menerapkan fitur *user-based security* guna memastikan hanya pengguna bersangkutan saja yang dapat mengakses datanya.

## Pengukuran Jarak

Spherical Law of Cosine adalah salah satu alternatif selain Haversine untuk menghitung jarak garis lurus antara dua lokasi dengan menggunakan parameter garis bujur dan lintang. Metode ini memiliki proses perhitungan yang jauh lebih sederhana jika dibandingkan dengan Haversine.

Koordinat lokasi pengguna diambil menggunakan sensor GPS yang ada di perangkat bergerak pengguna. Koordinat ini lalu digabungkan dengan lokasi tujuan yang dalam konteks ini adalah kafe, dan dilakukan perhitungan menggunakan persamaan berikut:

(2.7)

## Pengujian Perangkat Lunak

Pengujian perangkat lunak sangat diperlukan dalam sebuah sistem, karena dengan melakukan pengujian kesalahan yang terdapat dalam proses pengembangan dapat ditemukan (Rouf, 2012).

### Pengujian Fungsional

Pengujian ini berfokus pada spesifikasi fungsional dari sistem (Mustaqbal, Firdaus, Rahmadi, 2015). Pengujian fungsional merupakan salah satu bagian dari pengujian *black box*. Pengujian *black box* atau biasa disebut sebagai *behavioral testing* merupakan pengujian yang berfokus kepada fungsionalitas dari kebutuhan-kebutuhan aplikasi. Pengujian ini membuat kondisi masukkan yang digunakan membuat semua kebutuhan fungsional dapat diuji secara keseluruhan. Pengujian *black box* cenderung untuk menemukan hal-hal berikut:

Fungsi yang salah atau tidak ada

Kesalahan antarmuka

Kesalahan pada struktur data dan akses basis data

Kesalahan performansi

Kesalahan inisisialisasi dan terminasi

### Pengujian Validasi Algoritme

Pengujian ini memeriksa apakah perhitungan yang dilakukan sistem sesuai dengan perhitungan yang dilakukan secara manual. Jika proses perhitungan yang dilakukan sistem sudah sesuai dengan perhitungan yang dilakukan secara manual maka penerapan algoritme yang ada di dalam sistem dapat dinyatakan sesuai dan berhasil.

### Pengujian *Rank Consistency*

Pengujian ini memeriksa apakah hasil peringkat yang diberikan sistem dapat bertahan tanpa ada pembalikan peringkat terhadap penambahan alternatif (Junior, Osiro, Carpinetti, 2014). Pengujian dilakukan dengan menambahkan alternatif baru. Test dilakukan sejumlah dengan alternatif awal yaitu 5, yang berarti dilakukan 5 kali percobaan dengan menambahkan 1 alternatif baru yang sama dengan masing-masing alternatif. Jika hasil peringkat baru yang dikeluarkan sistem setelah ditambahkan alternatif baru tidak mengalami pembalikan atau pengacakan peringkat maka dapat dinyatakan bahwa metode yang diterapkan tahan terhadap penambahan alternatif.

### Pengujian *Usability*

Pengujian *usability* dilakukan untuk mengevaluasi apakah sebuah aplikasi sudah sesuai dengan kebutuhan pengguna atau belum (Nurhadryani, et al, 2013). Salah satu cara untuk melakukan pengujian ini adalah dengan menggunakan kuesioner *USE* *(Usefulness, Satisfaction and Ease of Use).* Hasil dari kuesioner *USE* diolah dengan melakukan perhitungan menggunakan skala *likert*.

#### Skala Likert

Skala *likert* adalah sebuah tipe skala psikometri atau pengukuran psikologi yang menggunakan angket dan skala yang lebih luas dalam penelitian survei. Skala *likert* pada umumnya digunakan untuk mengukur sikap atau respon seseorang terhadap sesuatu. Pengungkapan sikap dengan menggunakan Skala *likert* sangat populer di kalangan para ahli psikologi sosial dan para peneliti. Hal ini dikarenakan selain praktis juga dirancang dengan baik dan pada umumnya memiliki reliabilitas yang memuaskan.

Tabel 2.2 Skor Penilaian Jawaban Skala Likert

|  |  |
| --- | --- |
| Jawaban | Skor |
| Sangat Tidak Setuju | 1 |
| Tidak Setuju | 2 |
| Netral | 3 |
| Setuju | 4 |
| Sangat Setuju | 5 |

Seperti yang dapat dilihat pada tabel 2.2 terdapat 5 kategori jawaban. Jawaban-jawaban ini kemudian diberikan skor yang dapat menggambarkan jawaban tersebut. Perhitungan skala *likert* digunakan untuk mendapatkan indeks persentase pengujian. Untuk mendapatkan nilai indeks membutuhkan beberapa variabel seperti total skor, dan Y yang dapat dilihat pada persamaan 2.8 hingga 2.10.

(2.9)

(2.8)

(2.10)

# METODOLOGI PENELITIAN

## Tipe Penelitian

Tipe penelitian skripsi ini adalah penelitian implementatif dengan kategori pengembangan atau *development*. Penelitian ini termasuk ke dalam tipe penelitian implementatif pengembangan karena produk atau artefak utama yang dihasilkan dari penelitian ini adalah sebuah perangkat lunak utuh beserta rancangannya yang menjadi solusi dari permasalahan yang diangkat. Perangkat lunak yang dikembangkan adalah sesuatu yang belum ada yang dikembangkan dengan menerapkan prinsip-prinsip rekayasa secara utuh meliputi analisis, perancangan, implementasi, dan pengujian hingga menjadi ada.

## Strategi Penelitian

Gambar 3.1 Metode Penelitian

Seperti yang dapat dilihat pada Gambar 3.1, penelitian dimulai dengan analisis permasalahan yang mencari permasalahan apa yang ingin diselesaikan dan solusi apa yang ingin diterapkan untuk dapat menyelesaikan permasalahan tersebut. Lalu dilanjutkan dengan studi literatur yang merupakan penelusuran pengetahuan dalam rangka menentukan data kajian pustaka dan menyusun dasar teori yang digunakan sebagai dasar pengetahuan dalam penelitian. Dilanjutkan dengan pengumpulan data yang bertujuan untuk mengumpulkan data-data kafe yang dibutuhkan sebagai bahan dalam proses pengerjaan penelitian.

Setelah permasalahan, dasar penelitian dan data yang dibutuhkan sudah didapatkan, proses perancangan sistem bisa dilakukan. Diawali dengan analisis kebutuhan yang dilakukan untuk menentukan kebutuhan apa saja yang diperlukan pada sistem yang dibangun. Perancangan sistem dimulai dengan memodelkan perancangan menjadi arsitektur sistem berupa arsitektur perangkat bergerak, *sequence diagram*, *class diagram*, basis data, algoritme, dan antarmuka. Setelah sistem selesai dirancang proses implementasi sistem dapat dilaksanakan berdasarkan pada hasil perancangan sistem sebelumnya.

Setelah sistem berhasil diimplementasikan pengujian terhadap sistem dapat dilaksanakan. Pengujian sistem dilakukan untuk mengetahui apakah sistem yang dibangun dapat bekerja sesuai dengan spesifikasi dari kebutuhan pengguna secara baik dan sesuai dengan yang diharapkan. Proses pengujian dibagi menjadi 2 bagian yaitu proses pengujian dan proses analisis. Proses analisis dilakukan setelah semua rangkaian pengujian selesai dilakukan pada sistem. Tahap terakhir dari penelitian ini adalah penarikan kesimpulan dan saran. Penarikan kesimpulan dilakukan setelah seluruh tahapan pengembangan sistem selesai dilakukan yang meliputi perancangan, implementasi, dan pengujian sistem. Kesimpulan didapatkan untuk menjawab rumusan masalah yang telah dipaparkan sebelumnya. Tahap akhir dari penelitian ini adalah penyertaan saran dari penulis agar dapat memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penelitian yang dilakukan selanjutnya.

## Subjek, Lokasi, dan Data Penelitian

Subjek dari penelitian ini adalah kafe atau warung kopi. Lokasi dari kafe atau warung kopi ini berada di kota Malang Data-data kafe dikumpulkan sebagai bahan dalam proses pengerjaan penelitian. Data kafe yang diperlukan terdiri dari nama, deskripsi, lokasi, waktu operasional, menu, dan fasilitas.

## Metode Pengumpulan Data

Pengumpulan data kafe dilakukan dengan beberapa cara. Untuk mendapatkan informasi seperti fasilitas dan harga menu dilakukan observasi menuju kafe untuk mengetahui fasilitas dan menu apa saja yang tersedia. Informasi seperti foto, jenis biji kopi dan metode penyeduhan yang digunakan dilakukan dengan observasi di akun sosial media Instagram kafe yang bersangkutan. Informasi seperti rating, *latitude,* dan *longitude* dilakukan observasi menggunakan Google Maps.

## Peralatan Pendukung

Peralatan yang digunakan dalam menyusun kode sumber dan melakukan pengujian penelitian ini, adalah sebuah laptop dengan spesifikasi sebagai berikut:

Sistem Operasi : Pop\_Os! 18.04 by System76

Prosesor : Intel Core i5-4200H @ 2.80Ghz

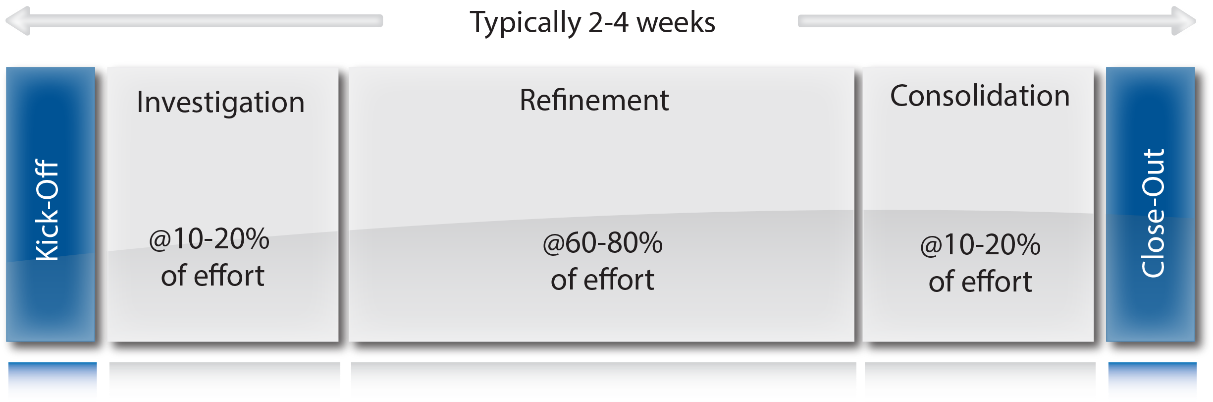
Memory : 8192MB

## Metode Pengembangan Perangkat Lunak

Metode pengembangan yang digunakan dalam pembuatan sistem pada penelitian ini adalah metode *Agile Development*. *Agile Development* adalah istilah umum untuk serangkaian metode yang berdasar pada nilai dan prinsip yang dinyatakan dalam *Agile Manifesto* (AgileBusiness, 2014). Salah satunya adalah *Dynamic Systems Development Method* (DSDM).

Pada DSDM terdapat sebuah teknik yaitu *Timeboxing*. *Timeboxing* adalah sebuah satuan periode pengerjaan objektif atau tugas. Objektif dari sebuah *Timebox* biasanya terdiri dari satu atau beberapa tugas. Teknik ini berguna untuk meningkatkan fokus demi menghasilkan pekerjaan yang berkualitas. Pada saat *Timebox* sudah selesai perlu dilakukan evaluasi yang mengukur kemajuan dan kesuksesan *Timebox*. Tingkat kesuksesan diukur berdasarkan sampai mana terselesaikannya suatu objektif (AgileBusiness, 2014).

Pada struktur *Timebox* di Gambar 3.2 terdiri dari 5 bagian, yaitu tahap *Kick-Off, Investigation, Refinement, Consolidation*, dan *Close-Out*. Kelima tahap ini adalah satu siklus DSDM, dalam pengerjaan sebuah sistem tidak berarti harus diselesaikan dalam satu siklus, melainkan melakukan siklus secara berulang sesuai dengan sifat *Agile* dan secara terus menerus juga melakukan perbaikan terhadap sistem yang dikembangkan. Struktur dari *Timebox* dijelakan pada sub-bab berikut:



Gambar 3.2 Struktur DSDM

### *Kick-Off*

Pada tahap ini dilakukan peninjauan objektif *Timebox* untuk mengetahui apa yang ingin diselesaikan dan dipastikan tujuan yang ingin diselesaikan benar-benar dapat diselesaikan.

### *Investigation*

Pada tahap ini objektif yang ingin diselesaikan pada tahap *Refinement* diperjelas lebih dalam lagi dan semua objektif diberikan skala prioritas dan syarat yang menyatakan objektif seperti apa yang dapat dinyatakan selesai.

### *Refinement*

Tujuan dari tahap ini adalah untuk menyelesaikan sebanyak-banyaknya objektif. Tahap ini diakhiri dengan dilakukannya peninjauan untuk mencari tahu apa yang perlu dilakukan agar objektif yang sedang dikerjakan dapat selesai. Setelah tahap ini selesai dilakukan tidak boleh ada objektif baru yang dikerjakan.

### *Consolidation*

Pada tahap ini keputusan apa yang ingin dilakukan untuk memastikan objektif memenuhi syarat pada tahap Refinement dilakukan, bersamaan dengan dilakukannya pengujian akhir. Tahap ini diakhiri dengan melakukan peninjauan yang memeriksa apakah objektif sudah memenuhi syarat. Setiap produk yang tidak memenuhi syarat tetap ada di daftar prioritas objektif untuk diselesaikan di *Timebox* selanjutnya.

### *Close-out*

Tujuan utama dari tahap ini adalah untuk mencatat objektif apa saja yang berhasil diselesaikan. Lalu memutuskan apakah objektif yang tidak selesai dilakukan lagi pada *Timebox* selanjutnya atau dihilangkan. Tujuan akhir dari tahap ini adalah untuk melakukan pengulasan pada proses *Timebox* yang telah dijalankan, mencari tahu apakah ada yang dapat dilakukan untuk membuat proses *Timebox* selanjutnya menjadi lebih baik.

# PERANCANGAN

Gambaran Aplikasi

Identifikasi Aktor

Analisis Kebutuhan

*Use Case Diagram*

Analisis Kebutuhan Fungsional

Analisis Kebutuhan Non-Fungsional

Perancangan

*Use Case Scenario*

Pemodelan Kebutuhan

Perancangan Aplikasi

Perancangan Algoritme

Perancangan Antarmuka

*Sequence Diagram*

Diagram Kelas

Perancangan Data

Perancangan TOPSIS

Perancangan Law of Cosine

Perhitungan Manual Law of Cosine

Perhitungan Manual TOPSIS

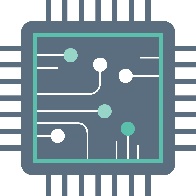
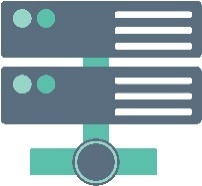
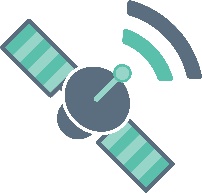
Gambar 4.1 Diagram Perancangan

## Analisis Kebutuhan

Pada tahap ini dilakukan identifikasi kebutuhan aplikasi rekomendasi kafe di Kota Malang. Terdapat 5 tahap utama yang dimulai dari analisis kebutuhan, pemodelan kebutuhan, perancangan aplikasi, perancangan algoritme, dan perancangan antarmuka. Masing-masing tahap tersebut memiliki tahap-tahapnya sendiri lagi seperti gambaran aplikasi, identifikasi aktor, analisis kebutuhan fungsional, dan analisis kebutuhan fungsional pada tahap analisis kebutuhan. Di tahap pemodelan kebutuhan terdapat *use case diagram* dan *use case scenario*. Di tahap perancangan aplikasi terdapat sequence diagram, class diagram, dan perancangan data. Di tahap perancangan algoritme terdapat perancangan TOPSIS, perancangan law of cosine, perhitungan manual law of cosine, dan perhitungan manual TOPSIS.

### Gambaran Aplikasi

Gambaran aplikasi merupakan representasi desain arsitektur sistem yang dibangun secara umum. Gambaran aplikasi secara umum ditunjukkan pada Gambar 4.2.



GPS

Hasil rekomendasi

Tampilan rekomendasi

Meminta rekomendasi

Meminta data kafe

Mengolah data kafe dengan kriteria pengguna



Gambar 4.2 Gambaran Umum Sistem

Data kafe

Request

GPS ke BTS

Pengguna menggunakan aplikasi pada perangkat bergerak Android. Aplikasi membutuhkan koneksi jaringan internet dan sensor GPS dalam keadaan siap untuk digunakan, hal ini diperlukan untuk mendapatkan data kafe yang disimpan di *database* dan lokasi pengguna melalui GPS untuk menjadi bahan pertimbangan hasil rekomendasi. Setelah aplikasi dibuka pengguna dapat melihat tampilan utama yang berisi kategori *Explore* yang berisi kafe-kafe baru, dan tombol yang dapat digunakan untuk meminta rekomendasi yang membawa pengguna ke tampilan pengaturan prioritas kriteria. Setelah prioritas kriteria selesai disesuaikan dan pengguna menekan tombol untuk menyimpan preferensi aplikasi mengambil data lokasi pengguna menggunakan sensor GPS lalu aplikasi meminta data kafe yang tersedia di *database.* Setelah hasil rekomendasi didapatkan aplikasi menampilkan daftar kafe yang tersedia berdasarkan peringkat yang didapatkan dari metode TOPSIS. Pengguna dapat melihat detail kafe dengan menekan tombol untuk melihat detail kafe.

### Identifikasi Aktor

Tahap ini berisi identifikasi aktor yang menggunakan aplikasi yang dibangun. Penjelasan aktor dapat dilihat pada tabel berikut.

Tabel 4.1 Identifikasi Aktor

|  |  |
| --- | --- |
| Aktor | Deskripsi |
| Pengguna | Pengguna dapat mendapatkan rekomendasi kafe yang berisi informasi detail kafe. |

### Analisis Kebutuhan Fungsional

Sub bab ini berisi spesifikasi kebutuhan fungsional aplikasi berupa kebutuhan fungsional, dan nama *use case* yang dapat dilihat pada Tabel 4.2.

Tabel 4.2 Spesifikasi Kebutuhan Fungsional

|  |  |  |
| --- | --- | --- |
| Nomor | Nama | Kebutuhan Fungsional |
| RK-F-01 | Menampilkan beranda | Aplikasi dapat menampilkan tampilan beranda yang di dalamnya terdapat 5 daftar kafe acak, daftar kafe favorit dan tombol meminta rekomendasi. |
| RK-F-02 | Mendapatkan rekomendasi | Aplikasi dapat menampilkan 5 rekomendasi kafe yang sesuai dengan kriteria yang diberikan oleh pengguna. |
| RK-F-03 | Menampilkan detail kafe | Aplikasi dapat menampilkan detail informasi kafe. |
| RK-F-04 | Menambahkan kafe ke daftar favorit | Aplikasi dapat menyimpan kafe ke dalam daftar kafe favorit pengguna. |
| RK-F-05 | Menghapus kafe dari daftar favorit | Aplikasi dapat menghapus kafe dari daftar kafe favorit pengguna. |
| RK-F-06 | Mendapatkan petunjuk arah | Aplikasi dapat memberikan petunjuk arah kafe dengan menggunakan Google Maps. |

Selanjutnya spesifikasi kebutuhan fungsional dijelaskan lebih lanjut menggunakan *use case diagram* dan *use case scenario.*

### Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional merupakan analisis yang tidak diminta oleh pengguna aplikasi, namun dibutuhkan oleh pengguna yang dapat dilihat pada Tabel 4.3.

Tabel 4.3 Spesifikasi Kebutuhan Non-Fungsional

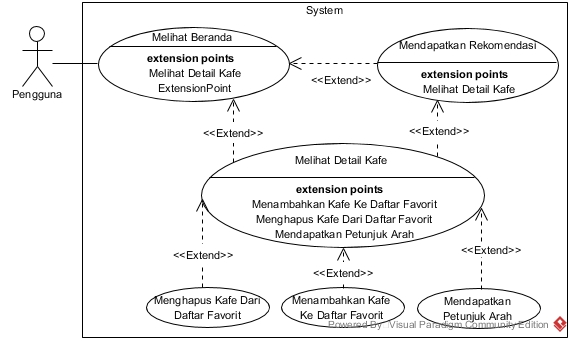
|  |  |  |
| --- | --- | --- |
| Nomor | Parameter | Deskripsi |
| RK-NF-01 | Validasi Algoritme | Memvalidasi hasil keluaran sistem dengan hasil perhitungan manual. Jika hasil perhitungan sama, maka penerapan metode TOPSIS dikatakan berhasil. |
| RK-NF-02 | *Rank Consistency* | Memeriksa hasil peringkat yang diberikan sistem dapat bertahan tanpa ada pembalikan peringkat terhadap penambahan alternatif. |
| RK-NF-03 | *Usability* | Aplikasi harus dapat mudah digunakan oleh pengguna berdasarkan kaidah *usability* yaitu *usefulness, easy to learn, easy use, dan satisfaction* dengan angka minimal 80%. |

## Pemodelan Kebutuhan

Sub bab ini berisi penjelasan-penjelasan terkait dengan pemodelan kebutuhan yang terdiri dari *use case diagram* dan *use case scenario*.

### Use Case Diagram

*Use case diagram* untuk aktor pengguna ditunjukkan pada Gambar 4.3. Pada awal penggunaan aplikasi pengguna ditampilkan tampilan beranda. Pada tampilan melihat beranda terdapat daftar 5 kafe yang diberikan oleh sistem secara acak, tombol meminta rekomendasi, dan daftar kafe favorit. Pengguna dapat melihat detail kafe atau meminta rekomendasi. Jika pengguna meminta rekomendasi tampilan kriteria ditampilkan. Saat pengguna selesai menentukan kriteria dan menekan tombol dapatkan rekomendasi maka sistem menampilkan 5 hasil rekomendasi dan pengguna dapat melihat detail masing-masing kafe dari hasil rekomendasi tersebut. Pada tampilan detail kafe pengguna dapat menambahkan kafe tersebut sebagai kafe favorit, menghapus dari daftar favorit jika sebelumnya sudah ditambahkan menjadi kafe favorit, dan mendapatkan petunjuk arah menuju kafe tersebut.



Gambar 4.3 Use Case Diagram Kebutuhan Fungsional

### Use Case Scenario

Pada sub bab ini dijelaskan secara detail spesifikasi *use case* yang ada pada *use case diagram*. Pada *use case scenario* terdapat nama spesifikasi, aktor yang terlibat, deskripsi, kondisi awal yang harus dipenuhi, aliran utama, aliran alternatif, dan kondisi akhir yang diharapkan.

#### Menampilkan Beranda

Menampilkan beranda adalah *use case scenario* dimana pengguna dapat melihat tampilan beranda dari aplikasi. Pada tampilan ini pengguna dapat melihat daftar kafe acak yang diberikan oleh aplikasi dan meminta rekomendasi kafe, detail dari *use case* ini dijelaskan pada Tabel 4.4.

Tabel 4.4 Skenario Menampilkan Beranda

|  |  |
| --- | --- |
| Nama | Menampilkan Beranda |
| Aktor | Pengguna |
| Deskripsi | Aplikasi dapat menampilkan tampilan beranda yang di dalamnya terdapat 5 daftar kafe acak dan tombol meminta rekomendasi. |
| Pre-condition | Pengguna belum membuka aplikasi. |
| Main flow | Pengguna membuka aplikasi |
| Alternative flow | - |
| Post-condition | Pengguna berhasil melihat tampilan beranda. |

#### Mendapatkan Rekomendasi

Mendapatkan rekomendasi adalah *use case scenario* dimana pengguna bisa mendapatkan rekomendasi. Skenario ini mulai terjadi ketika pengguna menekan tombol meminta rekomendasi, detail dari *use case* ini dijelaskan pada Tabel 4.5.

Tabel 4.5 Skenario Mendapatkan Rekomendasi

|  |  |
| --- | --- |
| Nama | Mendapatkan Rekomendasi |
| Aktor | Pengguna |
| Deskripsi | Aplikasi dapat menampilkan 5 rekomendasi kafe yang sesuai dengan kriteria yang diberikan pengguna. |
| Pre-condition | Pengguna berada pada tampilan beranda. |
| Main flow | Pengguna menekan tombol meminta rekomendasi  Aplikasi menampilkan tampilan prioritas kriteria  Pengguna menekan tombol untuk menyimpan prioritas kriteria  Aplikasi menampilkan 5 rekomendasi kafe yang sesuai dengan kriteria pengguna |
| Alternative flow | - |
| Post-condition | Pengguna berada pada tampilan hasil rekomendasi dan mendapatkan 5 rekomendasi kafe. |

#### Menampilkan Detail Kafe

Menampilkan detail kafe adalah *use case scenario* dimana pengguna dapat melihat detail informasi dari sebuah kafe. Skenario ini mulai terjadi ketika pengguna memilih kafe yang berada pada tampilan beranda atau hasil rekomendasi, detail dari *use case* ini dijelaskan pada Tabel 4.6.

Tabel 4.6 Skenario Menampilkan Detail Kafe

|  |  |
| --- | --- |
| Nama | Menampilkan Detail Kafe |
| Aktor | Pengguna |
| Deskripsi | Aplikasi dapat menampilkan detail informasi kafe. |
| Pre-condition | Pengguna sedang berada pada tampilan beranda, atau hasil rekomendasi. |
| Main flow | Pengguna memilih kafe yang ingin dilihat detailnya pada tampilan hasil rekomendasi  Aplikasi menampilkan detail informasi kafe yang dipilih |
| Alternative flow | - |
| Post-condition | Pengguna berada pada tampilan detail kafe dan melihat detail informasi terkait kafe tersebut. |

#### Menambahkan Kafe Ke Daftar Favorit

Menambahkan kafe ke dalam daftar favorit adalah *use case scenario* dimana pengguna dapat menambahkan kafe pilihannya ke daftar favoritnya. Skenario ini mulai terjadi ketika pengguna berada di tampilan detail sebuah kafe dan menekan tombol favorit, detail dari *use case* ini dijelaskan pada Tabel 4.7.

Tabel 4.7 Skenario Menambahkan Kafe Ke Daftar Favorit

|  |  |
| --- | --- |
| Nama | Menambahkan Kafe Ke Daftar Favorit |
| Aktor | Pengguna |
| Deskripsi | Aplikasi dapat menyimpan daftar kafe favorit pengguna. |
| Pre-condition | Pengguna sedang berada pada tampilan detail kafe. |
| Main flow | Pengguna menekan tombol favorit  Aplikasi menampilkan pesan berhasil |
| Alternative flow | - |
| Post-condition | Kafe ditambahkan ke dalam daftar kafe favorit pengguna. |

#### Menghapus Kafe Dari Daftar Favorit

Menghapus kafe dari daftar favorit adalah *use case scenario* dimana pengguna dapat menghapus kafe dari daftar favoritnya. Skenario ini mulai terjadi ketika pengguna berada di tampilan detail sebuah kafe yang sudah menjadi favorit dan menekan tombol favorit, detail dari *use case* ini dijelaskan pada Tabel 4.8.

Tabel 4.8 Skenario Menghapus Kafe Dari Daftar Favorit

|  |  |
| --- | --- |
| Nama | Menghapus Kafe Dari Daftar Favorit |
| Aktor | Pengguna |
| Deskripsi | Aplikasi dapat menghapus kafe dari daftar kafe favorit pengguna. |
| Pre-condition | Pengguna sedang berada pada tampilan detail kafe yang sudah dijadikan favorit. |
| Main flow | Pengguna menekan tombol favorit  Aplikasi menampilkan pesan berhasil |
| Alternative flow | - |
| Post-condition | Kafe dihapus dari daftar kafe favorit pengguna. |

#### Mendapatkan Petunjuk Arah

Mendapatkan petunjuk arah adalah *use case scenario* dimana pengguna dapat meminta petunjuk arah menuju kafe, detail dari *use case* ini dijelaskan pada Tabel 4.9.

Tabel 4.9 Skenario Mendapatkan Petunjuk Arah

|  |  |
| --- | --- |
| Nama | Mendapatkan Petunjuk Arah |
| Aktor | Pengguna |
| Deskripsi | Aplikasi dapat memberikan petunjuk arah kafe dengan menggunakan aplikasi Google Maps. |
| Pre-condition | Pengguna sedang berada pada tampilan detail kafe |
| Main flow | Pengguna menekan tombol untuk meminta petunjuk arah |
| Alternative flow | - |
| Post-condition | Pengguna berhasil mendapatkan petunjuk arah menuju kafe. |

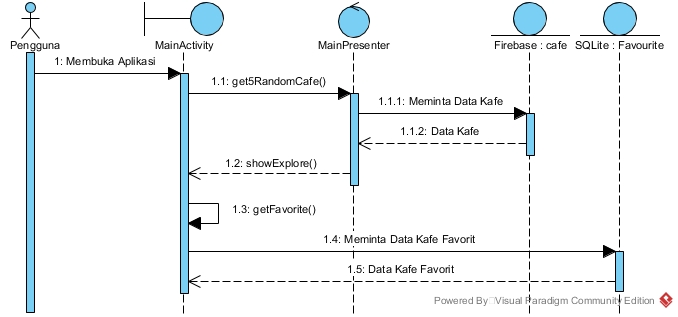
## Perancangan Aplikasi

Sub bab ini berisi pemodelan aplikasi yang terdiri dari *sequence diagram, class diagram*, perancangan data, perancangan TOPSIS, perancangan Law of Cosine, perhitungan manual, dan perancangan antarmuka.

### Perancangan Sequence Diagram

Perancangan *sequence diagram* menjelaskan interaksi dan hubungan antar objek dan kelas yang dirancang pada aplikasi perangkat bergerak rekomendasi kafe di kota Malang.

#### RK-F-01 (Menampilkan Beranda)

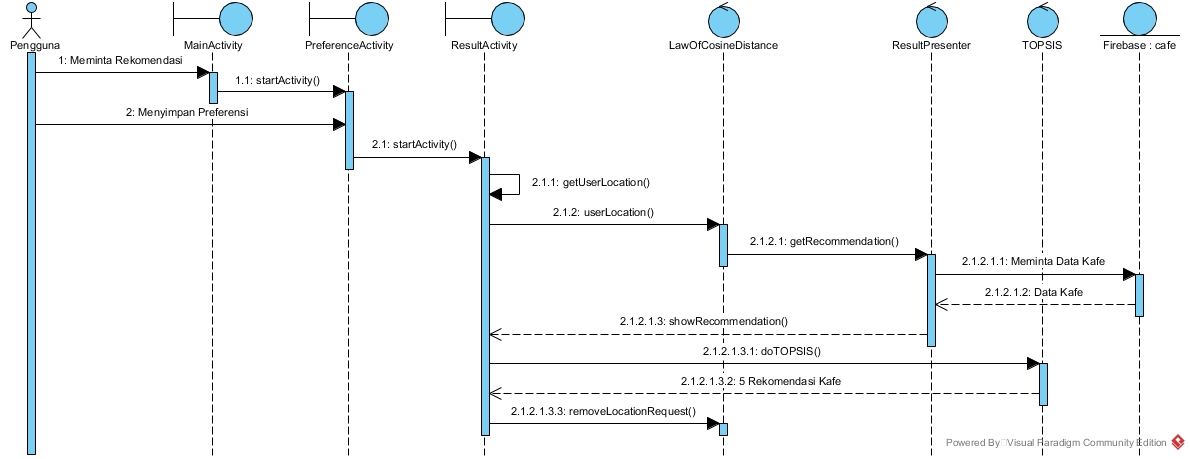


Gambar 4.4 Sequence Diagram Menampilkan Beranda

Gambar 4.4 menampilkan *sequence diagram* menampilkan beranda. Pengguna dapat sampai pada tampilan beranda dengan membuka aplikasi. Pada saat tiba di tampilan beranda, MainActivity memanggilfungsi get5RandomCafe di kelas MainPresenter untuk mendapatkan 5 kafe secara acak. Lalu MainActivity memanggil fungsi getFavorite untuk mendapatkan data kafe favorit pengguna dan menampilkannya.

#### RK-F-02 (Mendapatkan Rekomendasi)

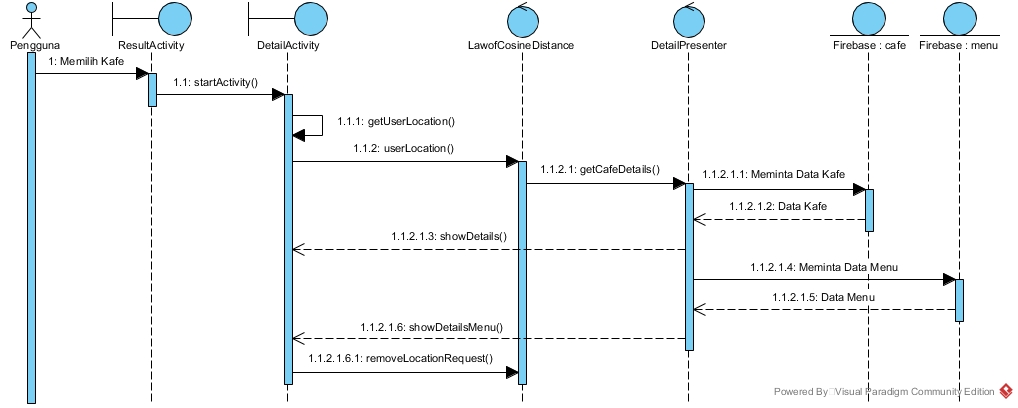
Gambar 4.5 menampilkan *sequence diagram* mendapatkan rekomendasi. Pengguna bisa mendapatkan rekomendasi kafe dengan cara menekan tombol untuk mendapatkan rekomendasi. Ketika tombol ditekan MainActivity memulai *activity* PreferenceActivity yang berisi tampilan pengaturan prioritas kriteria. Setelah pengguna mengatur kriterianya, lalu pengguna menekan tombol untuk menyimpan prioritas kriteria dan PreferenceActivity memulai *activity* ResultActivity. Selanjutnya ResultActivity memanggil fungsi getUserLocation yang memanggil fungsi userLocation untuk mendapatkan informasi lokasi pengguna pada saat ini, lalu fungsi getRecommendation pada kelas ResultPresenter dipanggil. ResultPresenter mengambil semua data kafe yang terdapat di *database* lalu memanggil fungsi showRecommendation pada *activity* ResultActivity untuk melakukan perangkingan dengan metode TOPSIS dengan fungsi doTOPSIS milik kelas TOPSIS. Setelah perangkingan dilakukan data kafe yang direkomendasikan dikembalikan lagi ke kelas ResultActivity untuk ditampilkan ke pengguna. Lalu fungsi removeLocationRequest dipanggil untuk memberhentikan permintaan sensor lokasi.



Gambar 4.5 Sequence Diagram Mendapatkan Rekomendasi

#### RK-F-03 (Menampilkan Detail Kafe)

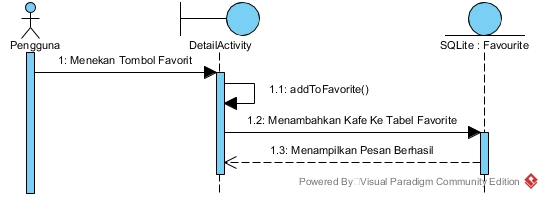
Gambar 4.6 menampilkan *sequence diagram* menampilkan detail kafe. Pengguna dapat menampilkan detail kafe dengan menekan tombol melihat detail. Setelah tombol ditekan ResultActivity memulai *activity* DetailActivity yang memanggil fungsi getUserLocation yang memanggil fungsi userLocation milik kelas LawofCosineDistance. Lalu fungsi getCafeDetails milik kelas DetailPresenter dipanggil untuk meminta data kafe dan menu yang terdapat di database dan menampilkannya di DetailActivity dengan fungsi showDetails dan showDetailsMenu. Baru setelah itu fungsi removeLocationRequest dipanggil untuk memberhentikan permintaan sensor lokasi.



Gambar 4.6 Sequence Diagram Menampilkan Detail Kafe

#### RK-F-04 (Menambahkan Kafe Ke Daftar Favorit)

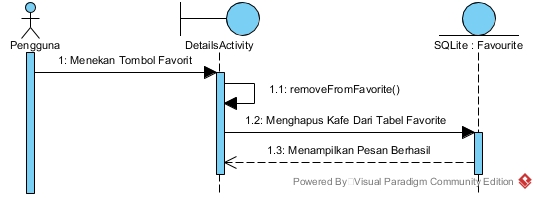
Gambar 4.7 menampilkan *sequence diagram* menambahkan kafe ke daftar favorit. Pengguna bisa menambahkan sebuah kafe ke dalam daftar kafe favoritnya dengan sudah berada di tampilan detail kafe yang ingin ditambahkan ke dalam daftar kafe favorit terlebih dahulu, lalu menekan ikon bintang. DetailActivity memanggil fungsi addToFavorite yang menambahkan kafe ke dalam tabel Favorit.



Gambar 4.7 Sequence Diagram Menambahkan Kafe Ke Daftar Favorit

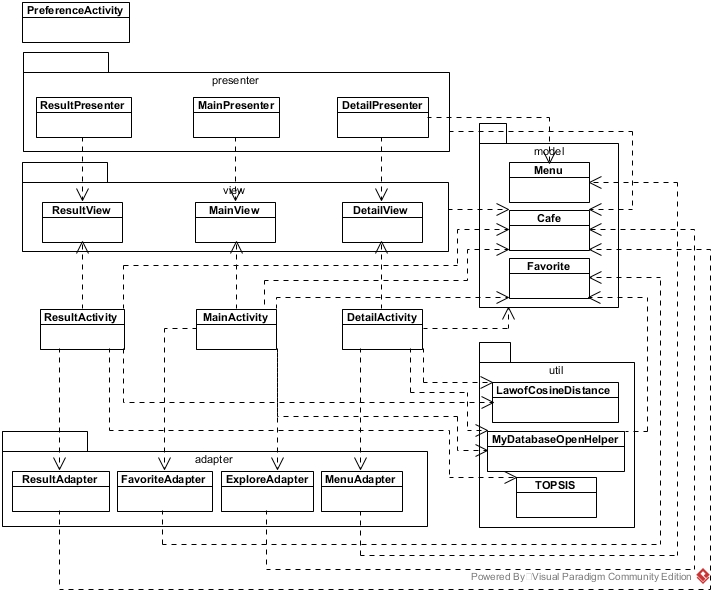
#### RK-F-05 (Menghapus Kafe Dari Daftar Favorit)

Gambar 4.8 menampilkan *sequence diagram* menghapus kafe dari daftar kafe favorit. Pengguna bisa menghapus sebuah kafe dari daftar kafe favoritnya dengan berada di tampilan detail kafe, lalu menekan ikon bintang. DetailActivity memanggil fungsi removeFromFavorite yang menghapus kafe dari daftar kafe favorit pengguna.



Gambar 4.8 Sequence Diagram Menghapus Kafe Dari Daftar Favorit

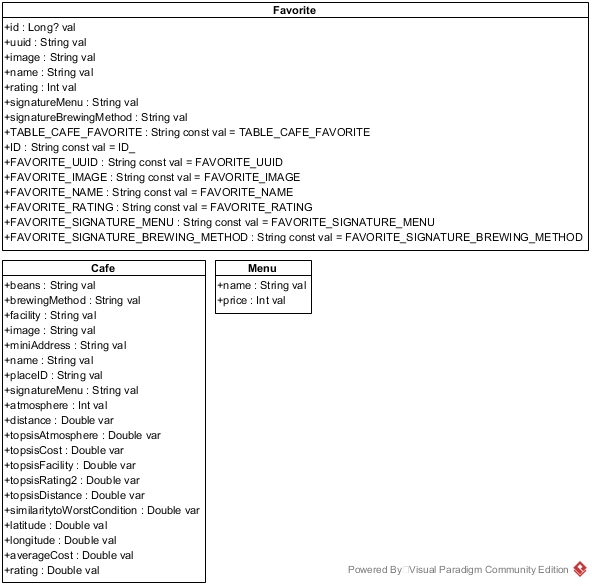
### Perancangan Diagram Kelas



Gambar 4.9 Diagram Kelas Sistem

Gambar 4.9 memaparkan garis besar dari diagram kelas sistem secara keseluruhan. Pada gambar ini kelas dibagi-bagi menjadi beberapa model, diantaranya adalah model, presenter, view, adapter, util, dan kelas-kelas activity. Pengelompokan kelas ini berdasarkan pada pola desain yang ingin digunakan, yaitu MVP atau Model, View, Presenter. Kelas adapter berfungsi untuk menjadi penghubung antara model dan recycler view yang ada pada tampilan aplikasi, sedangkan kelas util adalah kelas tambahan yang berfungsi untuk membantu berjalannya aplikasi.

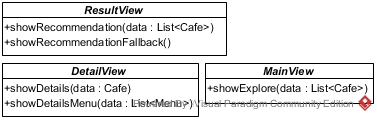
#### Diagram Kelas Package Model



Gambar 4.10 Class Diagram Package Model

Gambar 4.10 menjelaskan perancangan dari kelas-kelas yang terdapat pada *package* model. Terdapat tiga kelas yaitu Favorite, Cafe dan Menu. Ketiga kelas ini adalah representasi objek dari data yang terdapat pada sistem.

#### Diagram Kelas Package View

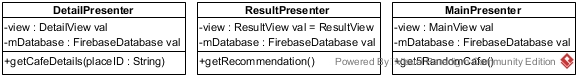


Gambar 4.11 Class Diagram Package View

Gambar 4.11 menjelaskan perancangan dari kelas-kelas yang terdapat pada *package* view. Terdapat tiga kelas utama yaitu ResultView, DetailView, dan MainView. Ketiga kelas ini adalah kelas *interface* yang diimplementasikan oleh kelas *activity*.

#### Diagram Kelas Package Presenter

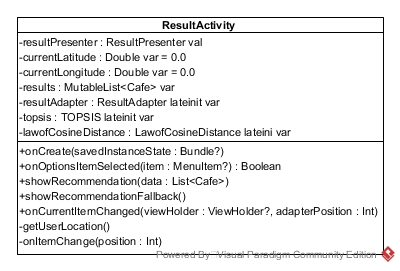
Gambar 4.12 menjelaskan perancangan dari kelas-kelas yang terdapat pada *package* presenter. Terdapat tiga kelas yaitu DetailPresenter, ResultPresenter, dan MainPresenter. Ketiga kelas ini adalah kelas yang bertugas untuk mengambil data dari *database* dan memberikan datanya kepada kelas view yang sudah diimplementasikan oleh kelas activity untuk kemudian proses penampilan data dapat dilakukan.



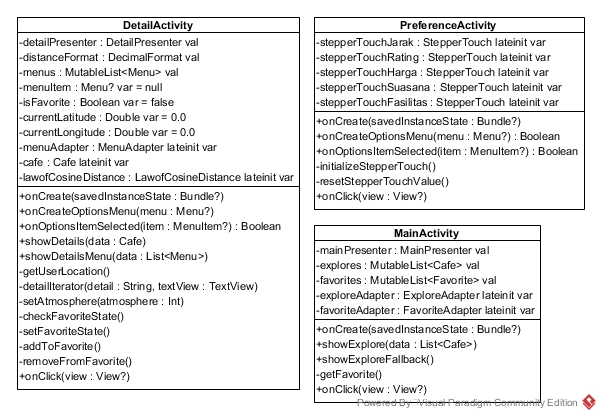
Gambar 4.12 Diagram Kelas Package Presenter

#### Diagram Kelas Activity

Gambar 4.13 menjelaskan perancangan dari kelas-kelas yang termasuk dalam kategori activity. Terdapat empat kelas yaitu DetailActivity, ResultActivity, MainActivity, dan PreferenceActivity. Keempat kelas ini adalah kelas yang bertugas untuk menjadi tempat data yang sudah didapatkan kelas presenter diolah untuk ditampilkan kepada pengguna. Masing-masing dari activity mengimplementasikan *interface* yang terdapat pada *package* view, kecuali kelas PreferenceActivity. Masing-masing kelas activity ini juga melakukan inisialisasi presenter masing-masing kelas kecuali kelas PreferenceActivity. PreferenceActivity tidak memiliki kelas view dan presenter karena pada PreferenceActivity tidak ada proses pengambilan data dari *database* dan pengolahan data untuk ditampilkan ke pengguna. Pada awal aplikasi dibuka pengguna ditampilkan MainActivity dan dapat melanjutkan ke activity lainnya sesuai dengan alur yang ada pada perancangan *use case scenario* dan *sequence diagram*.



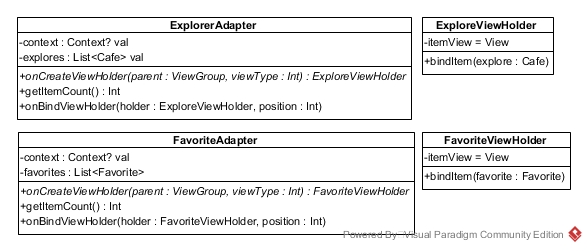
Gambar 4.13 Diagram Kelas Activity



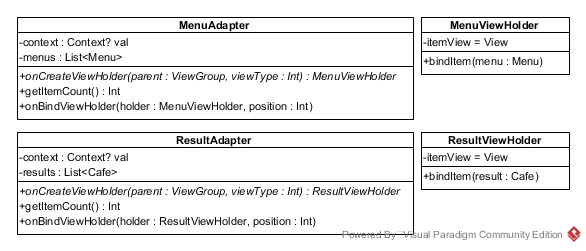
Gambar 4.13 Diagram Kelas Activity

#### Diagram Kelas Package Adapter

Gambar 4.14 menjelaskan perancangan dari kelas-kelas yang terdapat pada *package* adapter. Terdapat empat kelas yaitu ExplorerAdapter, FavoriteAdapter, MenuAdapter, dan ResultAdapter. Keempat kelas ini adalah kelas yang bertugas untuk melakukan pemasangan data kepada komponen *recycler view*. Masing-masing dari ada adapter memiliki kelas view holder masing-masing.

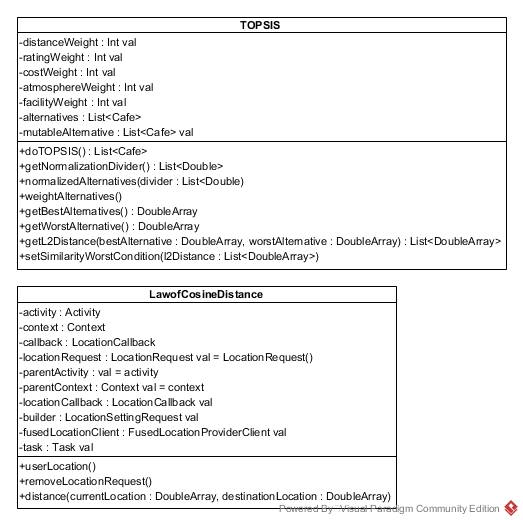


Gambar 4.14 Diagram Kelas Package Adapter

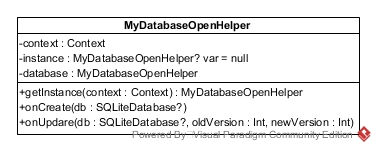


Gambar 4.14 Diagram Kelas Package Adapter

#### Diagram Kelas Package Util



Gambar 4.15 Diagram Kelas Package Util



Gambar 4.15 Diagram Kelas Package Util

Gambar 4.15 menjelaskan perancangan dari kelas-kelas yang terdapat pada *package* util. Terdapat tiga kelas yaitu TOPSIS, LawofCosineDistance, dan MyDatabaseOpenHelper. Ketiga kelas ini adalah kelas yang bertugas untuk melakukan tugas-tugas tertentu yang berguna untuk membantu berjalannya sistem. Seperti kelas TOPSIS yang berfungsi untuk melakukan proses perhitungan TOPSIS dan memberikan rekomendasi kafe. Kelas LawofCosineDistance yang berfungsi untuk melakukan perhitungan jarak antara pengguna dan kafe tujuan menggunakan rumus Law of Cosine. Kelas MyDatabaseOpenHelper yang berfungsi untuk mengatur database SQLite yang berguna untuk menyimpan data kafe favorit pengguna.

### Perancangan Data

Perancangan ini memberikan gambaran terkait penyimpanan data pada sistem. Basis data yang digunakan untuk menyimpan data adalah Firebase Realtime Database, maka dari itu data disimpan dalam bentuk JSON.

Tabel 4.10 Perancangan Data JSON

|  |  |
| --- | --- |
| 1 | { |
| 2 | "cafe" : { |
| 3 | "ID" : "Identifier kafe yang bernilai kombinasi angka dan huruf acak yang diberikan oleh Firebase", |
| 4 | "type" : "object", |
| 5 | "properties" : { |
| 6 | "atmosphere" : { |
| 7 | "type" : "integer" |
| 8 | }, |
| 9 | "averageCost" : { |
| 10 | "type" : "integer" |
| 11 | }, |
| 12 | "beans" : { |
| 13 | "type" : "string" |
| 14 | }, |
| 15 | "brewingMethod" : { |
| 16 | "type" : "string" |
| 17 | }, |
| 18 | "facility" : { |
| 19 | "type" : "string" |
| 20 | }, |
| 21 | "image" : { |
| 22 | "type" : "string" |
| 23 | }, |
| 24 | "latitude" : { |
| 25 | "type" : "float" |
| 26 | }, |
| 27 | "longitude" : { |
| 28 | "type" : "float" |
| 29 | }, |
| 30 | "miniAddress" : { |
| 31 | "type" : "string" |
| 32 | }, |
| 33 | "name" : { |
| 34 | "type" : "string" |
| 35 | }, |
| 36 | "rating" : { |
| 37 | "type" : "float" |
| 38 | }, |
| 39 | "signatureMenu" : { |
| 40 | "type" :"string" |
| 41 | }, |
| 42 | "uuid" : { |
| 43 | "type" :"string" |
| 44 | } |
| 45 | } |
| 46 | }, |
| 47 | "menu" : { |
| 48 | "ID" : "Identifier menu berisi value uuid dari kafe yang memiliki menu", |
| 49 | "type" : "object", |
| 50 | "properties" : { |
| 51 | "IDMenu" : "Identifier antar menu yang bernilai kombinasi angka dan huruf acak yang diberikan oleh Firebase", |
| 52 | "type" : "object", |
| 53 | "properties" : { |
| 54 | "name" : { |
| 55 | "type" :"string" |
| 56 | }, |
| 57 | "price" : { |
| 58 | "type" : "integer" |
| 59 | } |
| 60 | } |
| 61 | } |
| 62 | } |
| 63 | } |

Tabel 4.10 adalah struktur data yang ada pada basis data Firebase Realtime Database. Pada perancangan data diatas terdapat nama data, tipe data, dan properti jika data tersebut bertipe objek.

Tabel 4.11 Perancangan Data Firebase Storage

|  |  |  |
| --- | --- | --- |
| No. | Nama | Tipe Data |
| 1. | Name | String |
| 2. | Size | Double |
| 3. | Type | Image |
| 4. | Last Modified | Date |

Tabel 4.11 adalah struktur data Firebase Storage yang berfungsi untuk menyimpan gambar-gambar kafe.

## Perancangan Algoritme

Pada sub bab ini dijelaskan perancangan algoritme yang digunakan untuk memberikan rekomendasi kafe kepada pengguna.

### Perancangan TOPSIS

Mengambil data kafe di database

Start

Input data prioritas

Menghitung normalisasi matriks keputusan

Mencari nilai solusi ideal positif

Menghitung kedekatan relatif terhadap solusi ideal

Perangkingan hasil akhir

Menghitung jarak matriks keputusan dengan solusi ideal positif dan negatif

Menghitung nilai pembagi

End

Gambar 4.16 Flowchart TOPSIS

Mencari nilai solusi ideal negatif

Gambar 4.17 Flowchart TOPSIS

Gambar 4.16 menampilkan proses perhitungan TOPSIS. TOPSIS berfungsi untuk melakukan perangkingan alternatif kafe dan menghasilkan rekomendasi yang dapat sesuai dengan keinginan pengguna.

#### Perancangan Menghitung Nilai Pembagi Matriks Keputusan

Gambar 4.17 adalah sub proses dari *flowchart* TOPSIS. Sub proses ini menjelaskan proses perhitungan untuk mendapatkan nilai pembagi matriks keputusan yang digunakan untuk melakukan normalisasi.

Gambar 4.17 *Flowchart* Pembagi Normalisasi

Menghitung nilai akar dari jumlah nilai kuadrat setiap fitur

Menjumlahkan nilai kuadrat setiap kriteria alternatif i

i < jumlah alternatif

i = i + 1

No

Start

Yes

i = 0

j = 0

Menghitung nilai akar fitur j

j < jumlah kriteria

j = j + 1

No

Yes

Return

#### Perancangan Menghitung Matriks Keputusan Ternormalisasi

Gambar 4.18 adalah sub proses dari *flowchart* TOPSIS. Sub proses ini menjelaskan secara proses yang terjadi pada proses perhitungan matriks keputusan ternormalisasi. Setelah nilai pembagi didapatkan dari proses sebelumnya, dilanjutkan dengan mencari nilai normalisasi dari masing-masing fitur setiap alternatif yang ada. Nilai normalisasi matriks didapatkan dengan cara membagi nilai fitur dengan nilai pembagi yang sudah didapatkan pada proses perhitungan nilai pembagi sebelumnya.

i = 0

Membagi setiap nilai fitur alternatif i dengan nilai pembagi kriteia tersebut

Yes

i < jumlah alternatif

i = i + 1

No

Start

End

Set nilai matriks

Gambar 4.18 *Flowchart* Normalisasi

#### Perancangan Menghitung Matriks Solusi Ideal Positif

Kriteria semakin besar semakin baik

A+ < Nilai kriteria

A+ = Nilai kriteria

Yes

Yes

No

Kriteria semakin kecil semakin baik

A+ > Nilai kriteria

A+ = Nilai kriteria

Yes

Yes

No

Start

i = 0

i = i + 1

Yes

i < jumlah alternatif

No

Return

No

No

Gambar 4.19 *Flowchart* Solusi Ideal Positif

Gambar 4.19 adalah sub proses dari flowchart TOPSIS. Sub proses ini menjelaskan proses logika yang terjadi saat melakukan pencarian nilai matriks solusi ideal positif. Nilai matriks solusi ideal positif ditentukan pada sifat kriterianya. Jika nilai matriks kriteria semakin besar semakin baik, maka nilai kriteria yang dicari adalah yang terbesar, jika nilai matriks kriteria semakin kecil semakin baik, maka nilai kriteria yang dicari adalah yang terkecil.

#### Perancangan Menghitung Matriks Solusi Ideal Negatif

Gambar 4.20 adalah sub proses dari *flowchart* TOPSIS. Sub proses ini menjelaskan proses logika yang terjadi saat melakukan pencarian nilai matriks solusi ideal negatif. Nilai matriks solusi ideal negatif ditentukan pada sifat kriterianya. Jika nilai matriks kriteria semakin besar semakin baik, maka nilai kriteria yang dicari adalah yang terkecil, jika nilai matriks kriteria semakin kecil semakin baik, maka nilai kriteria yang dicari adalah yang terbesar. Berkebalikan dengan proses pencarian nilai solusi ideal positif.

Kriteria semakin besar semakin baik

A- > Nilai kriteria

A- = Nilai kriteria

Yes

Yes

No

Kriteria semakin kecil semakin baik

A- < Nilai kriteria

A- = Nilai kriteria

Yes

Yes

No

Start

i = 0

i = i + 1

Yes

i < jumlah alternatif

No

Return

No

No

Gambar 4.20 *Flowchart* Solusi Ideal Negatif

#### Perancangan Menghitung Jarak Matriks Keputusan Dengan Matriks Solusi Ideal Positif dan Negatif

Gambar 4.21 adalah sub proses dari *flowchart* TOPSIS. Sub proses ini menjelaskan secara proses perhitungan jarak matriks keputusan dengan matriks solusi ideal positif dan negatif.

Start

i = 0

Menghitung nilai akar kuadrat jarak matriks keputusan dengan matriks solusi ideal positif

No

i = i + 1

i < jumlah alternatif

Yes

Return

Gambar 4.21 *Flowchart* Jarak Matriks Keputusan Dengan Matriks

#### Perancangan Menghitung Kedekatan Relatif Dengan Solusi Ideal

Gambar 4.22 adalah sub proses dari *flowchart* TOPSIS. Sub proses ini menjelaskan secara mendetail proses yang terjadi saat melakukan perhitungan jarak matriks keputusan dengan matriks solusi ideal positif dan negatif.

Start

i = 0

Membagi jarak matriks keputusan solusi ideal negatif dengan jumlah jarak matriks keputusan solusi ideal positif dan negatif

No

i < jumlah alternatif

i = i + 1

Yes

Return

Gambar 4.22 *Flowchart* Kedekatan Relatif Dengan Solusi Ideal

### Perancangan Law of Cosine

Metode Law of Cosine adalah persamaan untuk menghitung jarak antara 2 lokasi dengan menggunakan *latitude* dan *longitude*. Proses perhitungan metode Law of Cosine untuk mendapatkan jarak antara 2 lokasi terdapat pada Gambar 4.23.

Start

Input data latitude dan longitude pengguna

Mengambil data latitude dan longitude kafe di database

Return

Menghitung nilai acos dari nilai sin latitude pengguna dikalikan nilai sin latitude kafe ditambah nilai cos latitude pengguna dikalikan nilai cos latitude kafe dikalikan nilai cos perbedaan longitude kafe dengan pengguna

Gambar 4.23 *Flowchart* Algoritme Haversine

### Perhitungan Manual Law of Cosine

Tabel 4.12 Data Lokasi

|  |  |  |
| --- | --- | --- |
| Lokasi | Latitude (Dalam Radian) | Longitude (Dalam Radian) |
| Pengguna | -0.138868465838453 | 1.965697315877940 |
| Kafe Tujuan | -0.139361905324577 | 1.965749867741710 |

Tabel 4.12 menjabarkan data lokasi pengguna dan kafe tujuan dalam bentuk latitude dan longitude. Dengan data ini perkiraan jarak dari dua lokasi bisa diketahui dengan menggunakan rumus Haversine. Dengan menggunakan persamaan 2.7 jarak antara pengguna dengan kafe tujuan bisa menghitung.

Berdasarkan perhitungan algoritme Law of Cosine didapatkan hasil jarak antara lokasi pengguna dan kafe tujuan sebesar 3161m atau 3.16km.

### Perhitungan Manual TOPSIS

Menentukan data alternatif yang digunakan.

Tabel 4.13 Sampel Data Alternatif

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Alternatif | Harga | Jarak (m) | Fasilitas | Rating |
| 1. | Kopi Cilik | 10000 | 3161.14049471563 | 1 | 5 |
| 2. | By Coffee | 15000 | 892.715441990626 | 2 | 4.6 |
| 3. | Nomaden Coffee | 10000 | 713.565422170724 | 2 | 4.6 |
| 4. | Bedjana Tiny House | 10000 | 2909.09765636547 | 2 | 4.8 |
| 5. | Labore Coffee Eatery | 25500 | 2000.42624177768 | 2 | 4.5 |

Menghitung normalisasi matriks alternatif

Normalisasi matriks alternatif dihitung dari nilai matriks pada Tabel 4.13 dengan menggunakan persamaan 2.1. Hal pertama yang perlu dilakukan adalah menjumlahkan nilai alternatif tiap kriteria yang dipangkatkan dua, kemudian diakar kuadrat. Langkah selanjutnya adalah membagi tiap data dengan hasil jumlah alternatif tiap kriteria.

Diatas adalah contoh perhitungan untuk mendapatkan jumlah nilai pembagi alternatif untuk setiap kriteria, detail dari hasil penjumlahan alternatif kriteria terdapat pada Tabel 4.14.

Tabel 4.14 Nilai Pembagi Alternatif Kriteria

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Alternatif | Harga | Jarak (m) | Fasilitas | Rating |
| 1. | Kopi Cilik | 10000 | 3161.14049471563 | 1 | 5 |
| 2. | By Coffee | 15000 | 892.715441990626 | 2 | 4.6 |
| 3. | Nomaden Coffee | 10000 | 713.565422170724 | 2 | 4.6 |
| 4. | Bedjana Tiny House | 10000 | 2909.09765636547 | 2 | 4.8 |
| 5. | Labore Coffee Eatery | 25500 | 2000.42624177768 | 2 | 4.5 |
|  | | 34281 | 4874.77999734152 | 4.1231 | 10.5171 |

Diatas adalah contoh perhitungan untuk mendapatkan nilai matriks alternatif yang sudah dinormalisasi, detail dari hasil normalisasi matriks alternatif setiap fitur dijabarkan pada Tabel 4.15.

Tabel 4.15 Hasil Normalisasi Matriks Alternatif

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Alternatif | Harga | Jarak (Km) | Fasilitas | Rating |
| 1. | Kopi Cilik | 0.2916989528 | 0.648468340405017 | 0.242535 | 0.475414 |

Tabel 4.15 Hasil Normalisasi Matriks Alternatif (Lanjutan)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Alternatif | Harga | Jarak (Km) | Fasilitas | Rating |
| 2. | By Coffee | 0.4375484292 | 0.18312938070589 | 0.4850712 | 0.437381 |
| 3. | Nomaden Coffee | 0.2916989528 | 0.1463790001928 | 0.4850712 | 0.437381 |
| 4. | Bedjana Tiny House | 0.2916989528 | 0.59676491204771 | 0.4850712 | 0.456398 |
| 5. | Labore Coffee Eatery | 0.7438323297 | 0.41036236360792 | 0.4850712 | 0.427873 |

Menghitung solusi ideal positif dan solusi ideal negatif

Solusi ideal positif dan solusi ideal negatif dihitung sesuai dengan persamaan 2.2 dan 2.3, apakah kriteria tersebut jika nilai semakin besar semakin menguntungkan atau jika semakin kecil semakin menguntungkan. Untuk kriteria menguntungkan (fasilitas, dan rating), solusi ideal positif merupakan nilai kriteria tertinggi dan solusi ideal negatif merupakan nilai kriteria terendah. Sedangkan kriteria merugikan (harga dan jarak), solusi ideal positif merupakan nilai kriteria terendah dan solusi ideal negatif merupakan nilai kriteria tertinggi.

Di atas juga terdapat contoh perhitungan untuk menghitung solusi ideal positif dan negatif, detail dari hasil perhitungan solusi ideal positif dan negatif terdapat pada Tabel 4.16.

Tabel 4.16 Hasil Solusi Ideal Positif dan Negatif

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Solusi | Harga | Jarak (Km) | Fasilitas | Rating |
| 1. | A+ | 0.2916989528 | 0.1463790001928 | 0.48507125 | 0.4754149 |
| 2. | A- | 0.7438323297 | 0.648468340405 | 0.242535 | 0.427873 |

Menghitung jarak alternatif terhadap solusi ideal positif dan negatif

Perhitungan dimulai dengan mengakar hasil kuadrat penjumlahan antara nilai normalisasi tiap baris dikurangi dengan solusi ideal positif dan mengakar hasil kuadrat penjumlahan antara nilai normalisasi tiap baris dikurangi dengan solusi ideal negatif seperti pada persamaan 2.4 dan 2.5.

Tabel 4.17 Hasil Perhitungan Jarak Terhadap Solusi Ideal Positif dan Negatif

|  |  |  |
| --- | --- | --- |
| No | Jarak | |
| **Si+** | **Si-** |
| 1. | 0.557599529202188 | 0.454625982523236 |
| 2. | 0.155142463763745 | 0.607671055967590 |
| 3. | 0.038033193692342 | 0.717936092711643 |
| 4. | 0.450787200964036 | 0.516463971713848 |
| 5. | 0.525710947332445 | 0.339879369185987 |

Berikut adalah contoh perhitungan untuk mendapatkan jarak alternatif terhadap solusi ideal positif, detail dari hasil perhitungan jarak terhadap solusi ideal positif dan negatif terdapat pada Tabel 4.17.

Menghitung kedekatan relatif

Perhitungan dimulai dengan cara membagi nilai solusi ideal negatif dengan hasil penjumlahan solusi ideal positif dan solusi ideal negatif seperti pada persamaan 2.6.

Berikut adalah contoh perhitungan untuk mendapatkan nilai preferensi, detail dari hasil perhitungan nilai preferensi terdapat pada Tabel 4.18.

Tabel 4.18 Hasil Perhitungan Nilai Preferensi

|  |  |  |
| --- | --- | --- |
| No | Alternatif | Preferensi |
| 1. | Kopi Cilik | 0.449135076380646 |
| 2. | By Coffee | 0.796618098983895 |
| 3. | Nomaden Coffee | 0.949689498797948 |
| 4. | Bedjana Tiny House | 0.533950215106993 |
| 5. | Labore Coffee Eatery | 0.392656159270642 |

Perangkingan nilai preferensi

Setelah mendapatkan nilai preferensi untuk masing-masing alternatif makanan dilakukan pengambilan keputusan rekomendasi kuliner. Langkahnya dengan mengurutkan hasil nilai preferensi dari alternatif yang tertinggi sampai alternatif terendah.

Tabel 4.19 Hasil Perangkingan

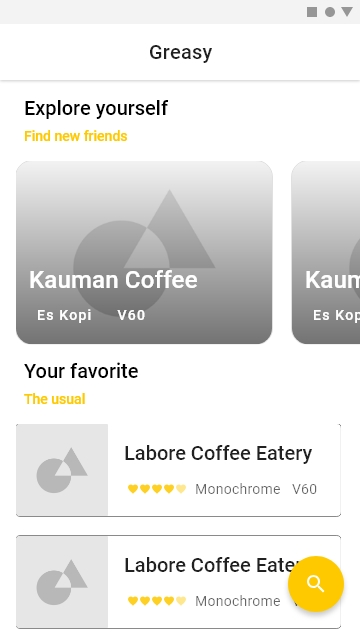
|  |  |  |
| --- | --- | --- |
| No | Alternatif | Preferensi |
| 1. | Nomaden Coffee | 0.949689498797948 |
| 2. | By Coffee | 0.796618098983895 |
| 3. | Bedjana Tiny House | 0.533950215106993 |
| 4. | Kopi Cilik | 0.449135076380646 |
| 5. | Labore Coffee Eatery | 0.392656159270642 |

Tabel 4.19 berisi hasil pengurutan nilai preferensi. Berdasarkan perhitungan menggunakan metode TOPSIS, maka diperoleh hasil rekomendasi alternatif kafe dengan nilai preferensi tertinggi yaitu Labore Coffee Eatery dengan nilai 0.733618769952074.

## Perancangan Antarmuka

Pada sub bab ini dijelaskan bagaimana rancangan antarmuka dari masing-masing tampilan aplikasi yang dibangun. Di sub bab ini juga dijelaskan alur dan tahapan yang terdapat pada tampilan aplikasi perangkat bergerak yang dibangun.

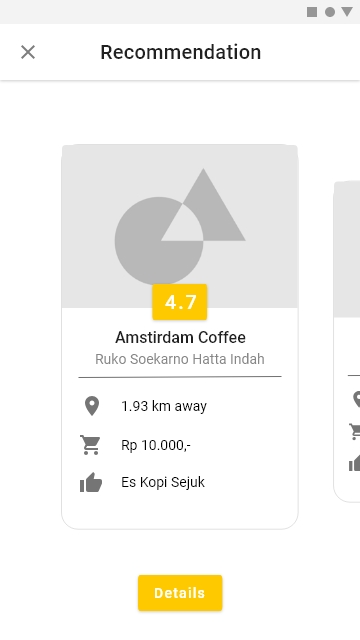
### Antarmuka Beranda



Gambar 4.24 Antarmuka Beranda

Gambar 4.24 menampilkan rancangan antarmuka beranda. Pengguna dapat melihat daftar kafe yang diberikan oleh sistem secara acak dan daftar kafe favorit. Pengguna juga dapat meminta rekomendasi kafe.

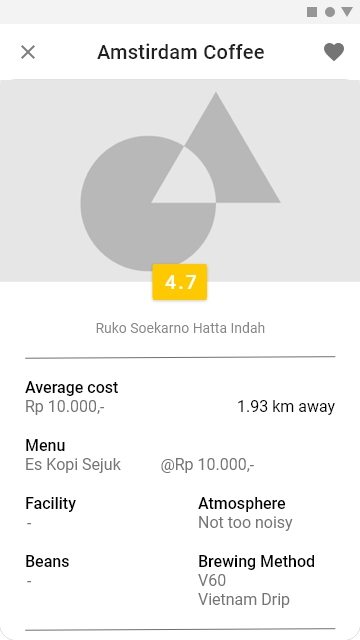
### Antarmuka Rekomendasi



Gambar 4.25 Antarmuka Rekomendasi

Gambar 4.25 menampilkan rancangan antarmuka rekomendasi. Antarmuka rekomendasi adalah antarmuka yang dilihat pengguna setelah memasukkan prioritas kriteria. Antarmuka ini menampilkan 5 rekomendasi kafe yang sesuai dengan preferensi pengguna. Pengguna dapat menggeser ke kanan dan ke kiri untuk melihat rekomendasi kafe.

### Antarmuka Detail



Gambar 4.26 Antarmuka Detail

Gambar 4.26 menampilkan antarmuka detail. Antarmuka detail adalah antarmuka yang dilihat pengguna saat ingin melihat detail kafe yang dipilih. Antarmuka ini berisi informasi lengkap mengenai harga, menu, fasilitas, dan alamat. Pengguna bisa menyimpan kafe ke dalam daftar favorit. Pengguna juga bisa mendapatkan petunjuk arah menuju kafe.

# IMPLEMENTASI

Spesifikasi Perangkat Keras

Spesifikasi Sistem

Spesifikasi Perangkat Lunak

Batasan Implementasi

Basis Data

Implementasi

Implementasi Mendapatkan Rekomendasi

Kode Program

Implementasi Algoritme Law of Cosine

Antarmuka

Implementasi Algoritme TOPSIS

Gambar 5.1 Diagram Implementasi

## Spesifikasi Sistem

Spesifikasi sistem dibagi menjadi 2 jenis, yaitu spesifikasi perangkat keras dan spesifikasi perangkat lunak.

### Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan pada proses implementasi sistem rekomendasi kafe berbasis lokasi di kota Malang dijabarkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras

|  |  |
| --- | --- |
| Spesifikasi | Detail |
| Prosesor | Intel Core i5-4200H CPU 2.80Ghz |
| Memori | 8192MB |
| Kartu Grafis | NVIDIA GeForce 750m |

Untuk proses instalasi dan pengujian dilakukan pada *emulator* dengan spesifikasi yang dijabarkan pada Tabel 5.2.

Tabel 5.2 Spesifikasi Emulator

|  |  |
| --- | --- |
| Spesifikasi | Detail |
| Prosesor | Google APIs Intel Atom (x86) |
| Memori | 512MB |

### Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan pada proses implementasi sistem rekomendasi kafe berbasis lokasi di kota Malang dijabarkan pada Tabel 5.3.

Tabel 5.3 Spesifikasi Perangkat Lunak

|  |  |
| --- | --- |
| Spesifikasi | Detail |
| Sistem Operasi | Pop\_Os! 18.04 by System76 |
| Bahasa Pemrograman | Kotlin |
| Alat Pengembangan | Android Studio |
| Basis Data | Firebase |
| Platform Pengembangan | Android API 22 |

## Batasan Implementasi

Tahap implementasi aplikasi perangkat bergerak ini memiliki beberapa batasan dalam proses implementasinya, antara lain:

1. Aplikasi perangkat bergerak rekomendasi kafe di kota Malang diimplementasikan untuk dapat berjalan pada perangkat bergerak dengan sistem operasi minimal Android 5.0.
2. Aplikasi perangkat bergerak diimplementasikan menggunakan bahasa pemrograman Kotlin.
3. Penyimpanan data menggunakan *database* Firebase Realtime Database.
4. Aplikasi harus berjalan dengan koneksi internet untuk melakukan pengambilan data.
5. Untuk mendapatkan rekomendasi pengguna harus mengaktifkan sensor GPS untuk mendapatkan informasi lokasi pengguna.

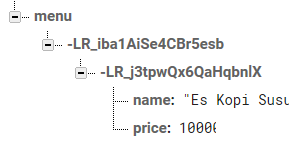
## Implementasi Basis Data

Berdasarkan perancangan basis data pada Tabel 4.10, implementasi penyimpanan data pada aplikasi menggunakan *database* Firebase Realtime Database untuk menyimpan data kafe. Pada Firebase Realtime Database data disimpan menggunakan struktur data JSON. Berikut adalah implementasi dari basis data yang digunakan aplikasi.



Gambar 5.2 Implementasi Basis Data *Child* cafe di Firebase Realtime Database

Gambar 5.1 menggambarkan susunan dari struktur data yang ada pada Firebase Realtime Database. *Child* pertama dari basis data adalah cafe yang digunakan untuk menyimpan data kafe. Setiap kafe memiliki kode uniknya sendiri, seperti pada contoh gambar diatas *“-LR\_iba1AiSe4CBr5esb”*. Kode ini adalah kode unik yang diberikan secara otomatis oleh Firebase Realtime Database ketika data diinputkan. Ada juga informasi seperti rata-rata harga, jenis biji kopi yang di sangria, metode penyeduhan, dan foto kafe. Data *image* menyimpan alamat dari foto kafe yang disimpan pada Firebase Storage. Data *latitude* dan *longitude* yang disimpan pada *database* adalah nilai yang sudah diubah menjadi radian atau nilai *latitude* dan *longitude* yang dikalikan dengan variabel konstan Pi lalu dibagi 180. Juga terdapat data seperti alamat, nama kafe, rating, dan menu andalan dari kafe tersebut. Data yang terakhir adalah uuid, data ini bernilai sama dengan identifier yang diberikan Firebase Realtime Database ketika data dimasukkan, data ini digunakan untuk menjadi pembeda saat melakukan pencarian data secara spesifik.



Gambar 5.3 Implementasi Basis Data *Child* menu di Firebase Realtime Database

Gambar 5.3 menggambarkan susunan dari struktur data yang ada pada Firebase Realtime Database. *Child* ini adalah yang kedua setelah kafe yang digunakan untuk menyimpan data menu kafe. *Identifier* dari objek-objek yang terdapat di *child* menu adalah kode unik dari kafe yang memiliki menu tersebut. Seperti pada contoh gambar diatas *“-LR\_iba1AiSe4CBr5esb”* adalah kode unik milik kafe dengan nama Kopi Cilik, maka menu yang terdapat pada objek ini adalah menu dari kafe Kopi Cilik. Data yang disimpan adalah nama menu dan harganya.

## Implementasi Kode Program

Pada sub bab ini dijelaskan tentang implementasi program berdasarkan fitur aplikasi rekomendasi kafe di kota Malang yang terdapat di perancangan. Kode program menggunakan bahasa pemrograman kotlin dan menggunakan Firebase Realtime Database untuk penyimpanan datanya.

### Implementasi Kode Program Memberikan Rekomendasi

Pada algoritme 1 terdapat kode program yang berfungsi untuk memberikan rekomendasi kafe kepada pengguna setelah pengguna memberikan prioritas kriteria kafe yang diinginkan pada tampilan Preference atau PreferenceActivity. Kode program ini terdapat pada kelas ResultActivity.

|  |  |
| --- | --- |
| Algoritme 1: Fungsi Memberikan Rekomendasi | |
| 1 | class ResultActivity : AppCompatActivity(), ResultView, |
| 2 | DiscreteScrollView.OnItemChangedListener<RecyclerView.ViewHolder> { |
| 3 | private val resultPresenter = ResultPresenter(this, FirebaseDatabase.getInstance()) |
| 4 | private var currentLatitude: Double = 0.0 |
| 5 | private var currentLongitude: Double = 0.0 |
| 6 | private var results: MutableList<Cafe> = mutableListOf() |
| 7 | private lateinit var resultAdapter: ResultAdapter |
| 8 | private lateinit var topsis: TOPSIS |
| 9 | private lateinit var lawofCosineDistance: LawofCosineDistance |
| 10 | override fun onCreate(savedInstanceState: Bundle?) { |
| 11 | super.onCreate(savedInstanceState) |
| 12 | setContentView(R.layout.activity\_result) |
| 13 | setSupportActionBar(toolbar\_result) |
| 14 | supportActionBar?.setHomeAsUpIndicator(R.drawable.ic\_home) |
| 15 | supportActionBar?.setDisplayHomeAsUpEnabled(true) |
| 16 | resultAdapter = ResultAdapter(this, results) |

|  |  |
| --- | --- |
| 17 | discrete\_scroll\_view\_result.setOrientation(DSVOrientation.HORIZONTAL) |
| 18 | discrete\_scroll\_view\_result.addOnItemChangedListener(this) |
| 19 | discrete\_scroll\_view\_result.adapter = resultAdapter |
| 20 | discrete\_scroll\_view\_result.setItemTransitionTimeMillis(150) |
| 21 | discrete\_scroll\_view\_result.setItemTransformer(ScaleTransformer.Builder().setMinScale(0.8f).build()) |
| 22 | getUserLocation() |
| 23 | } |
| 24 | override fun onOptionsItemSelected(item: MenuItem?): Boolean { |
| 25 | return when (item?.itemId) { |
| 26 | android.R.id.home -> { |
| 27 | onBackPressed() |
| 28 | true |
| 29 | } |
| 30 | else -> super.onOptionsItemSelected(item) |
| 31 | } |
| 32 | } |
| 33 | override fun showRecommendation(data: List<Cafe>) { |
| 34 | results.addAll(data) |
| 35 | data.forEachIndexed **{** i, it **->** |
| 36 | results[i].topsisCost = it.averageCost |
| 37 | results[i].topsisRating = it.rating |
| 38 | results[i].topsisAtmosphere = it.atmosphere.toDouble() |
| 39 | results[i].topsisFacility = results[i].facility.split(",").size.toDouble() |
| 40 | results[i].distance = lawofCosineDistance.distance( |
| 41 | doubleArrayOf(currentLatitude, currentLongitude), |
| 42 | doubleArrayOf(it.latitude, it.longitude) |
| 43 | ) |
| 44 | results[i].topsisDistance = results[i].distance |
| 45 | **}** |
| 46 | topsis = TOPSIS( |
| 47 | intent.getIntExtra("distance\_weight", 1), |
| 48 | intent.getIntExtra("rating\_weight", 1), |
| 49 | intent.getIntExtra("cost\_weight", 1), |
| 50 | intent.getIntExtra("atmosphere\_weight", 1), |
| 51 | intent.getIntExtra("facility\_weight", 1), |
| 52 | results |
| 53 | ) |
| 54 | results.clear() |
| 55 | results.addAll(topsis.doTOPSIS().dropLast(25)) |
| 56 | resultAdapter.notifyDataSetChanged() |
| 57 | lawofCosineDistance.removeLocationRequest() |
| 58 | result\_progress.visibility = View.GONE |
| 59 | discrete\_scroll\_view\_result.visibility = View.VISIBLE |
| 60 | button\_details.visibility = View.VISIBLE |
| 61 | } |
| 62 | override fun showRecommendationFallback() { |
| 63 | result\_progress.visibility = View.GONE |
| 64 | view\_result\_fallback.visibility = View.VISIBLE |
| 65 | } |
| 66 | override fun onCurrentItemChanged(viewHolder: RecyclerView.ViewHolder?, adapterPosition: Int) { |
| 67 | onItemChange(adapterPosition) |
| 68 | } |
| 69 | private fun getUserLocation() { |
| 70 | lawofCosineDistance = LawofCosineDistance(this, this, object : LocationCallback() { |
| 71 | override fun onLocationResult(locationResult: LocationResult?) { |
| 72 | locationResult ?: return |
| 73 | for (location in locationResult.locations) { |
| 74 | currentLatitude = location.latitude.times(Math.PI).div(180) |
| 75 | currentLongitude = location.longitude.times(Math.PI).div(180) |
| 76 | resultPresenter.getRecommendation() |
| 77 | } |
| 78 | } |
| 79 | }) |
| 80 | lawofCosineDistance.userLocation() |
| 81 | } |
| 82 | private fun onItemChange(position: Int) { |
| 83 | button\_details.onClick **{** |
| 84 | startActivity<DetailActivity>("place\_id" to results[position].placeID) |
| 85 | **}** |
| 86 | } |
| 87 | } |

Algoritme diatas menampilkan implementasi kode program pada kelas ResultActivity yang berfungsi untuk memberikan rekomendasi kafe kepada pengguna. Penjelasan potongan program, yaitu:

Baris 33, merupakan fungsi untuk menampilkan rekomendasi kafe.

Baris 34, merupakan proses menyalin semua keluaran fungsi getRecommendation objek resultPresenter yang berisi data semua kafe yang ada pada *database*.

Baris 35-44, merupakan proses pengisian variabel yang digunakan untuk melakukan perhitungan TOPSIS.

Baris 46-53, merupakan inisialisasi objek untuk melakukan perhitungan TOPSIS, informasi prioritas kriteria yang diberikan pengguna pada tampilan sebelumnya adalah argumen yang dibutuhkan untuk membuat objek TOPSIS, juga daftar kafe pada objek results.

Baris 54, merupakan proses untuk menghapus semua kafe yang ada pada objek results.

Baris 55, merupakan proses pengisian objek results dengan daftar kafe baru dengan keluaran fungsi doTOPSIS objek topsis.

Baris 56, merupakan pemanggilan fungsi untuk memberitahu objek resultAdapter bahwa ada perubahan data.

Baris 57, merupakan pemanggil fungsi untuk menghapus permintaan pembaharuan lokasi pengguna.

Baris 58-60, merupakan proses penampilan rekomendasi kafe.

Baris 62-65, merupakan fungsi untuk menampilkan pesan gagal ketika proses pengambilan data kafe gagal dilakukan.

Baris 66-68, merupakan sebuah *listener* yang berfungsi untuk memberitahu posisi rekomendasi yang sedang dilihat pengguna saat ini kepada fungsi onItemChange.

Baris 69-81, merupakan fungsi untuk mendapatkan lokasi pengguna.

Baris 82-86, merupakan fungsi untuk membuka tampilan detail kafe dari kafe yang dipilih pengguna.

### Implementasi Kode Program Algoritme Law of Cosine Distance

Pada algoritme 2 terdapat kode program yang berfungsi untuk mendapatkan jarak pengguna dengan kafe. Status lokasi pengguna didapatkan menggunakan sensor GPS pada perangkat bergerak pengguna. Sedangkan lokasi dari kafe berada di *database.* Kode program ini terdapat pada kelas LawofCosineDistance.

|  |  |
| --- | --- |
| Algoritme 2: Fungsi Mendapatkan Jarak Pengguna Dengan Kafe | |
| 1 | class LawofCosineDistance(activity: Activity, context: Context, callback: LocationCallback) { |
| 2 | private val locationRequest = LocationRequest().apply **{** |
| 3 | interval = 100000 |
| 4 | fastestInterval = 50000 |
| 5 | priority = LocationRequest.PRIORITY\_HIGH\_ACCURACY |
| 6 | **}** |
| 7 | private val parentActivity = activity |
| 8 | private val parentContext = context |
| 9 | private val builder = LocationSettingsRequest.Builder().addLocationRequest(locationRequest) |
| 10 | private val fusedLocationClient = LocationServices.getFusedLocationProviderClient(activity) |
| 11 | private val locationClient = LocationServices.getSettingsClient(context) |
| 12 | private val task = locationClient.checkLocationSettings(builder.build()) |
| 13 | private val locationCallback = callback |
| 14 | fun userLocation() { |
| 15 | task.addOnSuccessListener **{** |
| 16 | val permission = |
| 17 | ContextCompat.checkSelfPermission(parentContext, android.Manifest.permission.ACCESS\_FINE\_LOCATION) |
| 18 | if (permission == PackageManager.PERMISSION\_GRANTED) { |
| 19 | fusedLocationClient.requestLocationUpdates(locationRequest, locationCallback, null) |
| 20 | } |
| 21 | **}** |
| 22 | task.addOnFailureListener **{** |
| 23 | if (it is ResolvableApiException) { |
| 24 | try { |
| 25 | it.startResolutionForResult(parentActivity, 0) |
| 26 | } catch (sendEx: IntentSender.SendIntentException) { |
| 27 | } |
| 28 | } |
| 29 | **}** |
| 30 | } |
| 31 | fun removeLocationRequest() { |
| 32 | fusedLocationClient.removeLocationUpdates(locationCallback) |
| 33 | } |
| 34 | fun distance(currentLocation: DoubleArray, destinationLocation: DoubleArray): Double { |
| 35 | return Math.acos( |
| 36 | Math.sin(currentLocation[0]) \* |
| 37 | Math.sin(destinationLocation[0]) + |
| 38 | Math.cos(currentLocation[0]) \* |
| 39 | Math.cos(destinationLocation[0]) \* |
| 40 | Math.cos(destinationLocation[1] - currentLocation[1]) |
| 41 | ) \* 6371 |
| 42 | } |
| 43 | } |

Algoritme diatas menampilkan implementasi kode program pada kelas LawofCosineDistance yang berfungsi untuk mendapatkan informasi lokasi pengguna dan mendapatkan jarak antara pengguna dengan kafe. Penjelasan potongan program, yaitu:

Baris 14, merupakan fungsi untuk mendapatkan lokasi pengguna.

Baris 15-21, merupakan proses penambahan listener kepada objek task ketika proses pengecekan apakah sensor GPS pengguna aktif berhasil. Objek fusedLocationClient meminta pembaharuan lokasi pengguna dan menjalankan fungsi locationCallback yang berisi proses pengisian latitude dan longitude pengguna dan pemanggilan fungsi getRecommendation melalui objek resultPresenter.

Baris 22-29, merupakan proses penambahan listener kepada objek task ketika proses pengecekan apakah sensor GPS pengguna aktif gagal. Akan ditampilkan dialog yang meminta pengguna untuk mengaktifkan GPSnya.

Baris 31-33, merupakan fungsi untuk menghapus permintaan pembaharuan lokasi pengguna.

Baris 34-41, merupakan proses perhitungan jarak antara pengguna dengan kafe menggunakan persamaan 2.8.

### Implementasi Kode Program Algoritme TOPSIS

Pada algoritme 3 terdapat kode program yang berfungsi melakukan perhitungan TOPSIS untuk mendapatkan rekomendasi kafe. Menggunakan masukkan berupa daftar kafe dan memberikan keluaran yang juga daftar kafe namun sudah dalam bentuk terurut sesuai dengan rekomendasi.

|  |  |
| --- | --- |
| Algoritme 3: Fungsi TOPSIS | |
| 1 | class TOPSIS( |
| 2 | private val distanceWeight: Int, |
| 3 | private val ratingWeight: Int, |
| 4 | private val costWeight: Int, |
| 5 | private val atmosphereWeight: Int, |
| 6 | private val facilityWeight: Int, |
| 7 | alternatives: List<Cafe> |
| 8 | ) { |
| 9 | private val mutableAlternative = alternatives.toMutableList() |
| 10 | fun doTOPSIS(): List<Cafe> { |
| 11 | normalizedAlternatives(getNormalizationDivider()) |
| 12 | weightAlternatives() |
| 13 | setSimilaritytoWorstCondition( |
| 14 | getL2Distance( |
| 15 | getBestAlternative(), |
| 16 | getWorstAlternative() |
| 17 | ) |
| 18 | ) |
| 19 | return mutableAlternative.sortedByDescending **{** |
| 20 | it.similaritytoWorstCondition |
| 21 | **}** |
| 22 | } |
| 23 | private fun getNormalizationDivider(): List<Double> { |
| 24 | val divider = mutableListOf<Double>() |
| 25 | for (i in 0..4) { |
| 26 | divider.add(0.0) |
| 27 | } |
| 28 | mutableAlternative.forEach **{** |
| 29 | divider[0] += it.topsisDistance.pow(2) |
| 30 | divider[1] += it.topsisRating.pow(2) |
| 31 | divider[2] += it.topsisCost.pow(2) |
| 32 | divider[3] += it.topsisAtmosphere.pow(2) |
| 33 | divider[4] += it.topsisFacility.pow(2) |
| 34 | **}** |
| 35 | for (i in 0..4) { |
| 36 | divider[i] = Math.sqrt(divider[i]) |
| 37 | } |
| 38 | return divider |
| 39 | } |
| 40 | private fun normalizedAlternatives(divider: List<Double>) { |
| 41 | mutableAlternative.forEachIndexed **{** i, it **->** |
| 42 | mutableAlternative[i].topsisDistance = it.topsisDistance.div(divider[0]) |
| 43 | mutableAlternative[i].topsisRating = it.topsisRating.div(divider[1]) |
| 44 | mutableAlternative[i].topsisCost = it.topsisCost.div(divider[2]) |
| 45 | mutableAlternative[i].topsisAtmosphere = it.topsisAtmosphere.div(divider[3]) |
| 46 | mutableAlternative[i].topsisFacility = it.topsisFacility.div(divider[4]) |
| 47 | **}** |
| 48 | } |
| 49 | private fun getBestAlternative(): DoubleArray { |
| 50 | return doubleArrayOf( |
| 51 | mutableAlternative.minBy **{** it.topsisDistance **}**!!.topsisDistance, |
| 52 | mutableAlternative.maxBy **{** it.topsisRating **}**!!.topsisRating, |
| 53 | mutableAlternative.minBy **{** it.topsisCost **}**!!.topsisCost, |
| 54 | mutableAlternative.maxBy **{** it.topsisAtmosphere **}**!!.topsisAtmosphere, |
| 55 | mutableAlternative.maxBy **{** it.topsisFacility **}**!!.topsisFacility |
| 56 | ) |
| 57 | } |
| 58 | private fun getWorstAlternative(): DoubleArray { |
| 59 | return doubleArrayOf( |
| 60 | mutableAlternative.maxBy **{** it.topsisDistance **}**!!.topsisDistance, |
| 61 | mutableAlternative.minBy **{** it.topsisRating **}**!!.topsisRating, |
| 62 | mutableAlternative.maxBy **{** it.topsisCost **}**!!.topsisCost, |
| 63 | mutableAlternative.minBy **{** it.topsisAtmosphere **}**!!.topsisAtmosphere, |
| 64 | mutableAlternative.minBy **{** it.topsisFacility **}**!!.topsisFacility |
| 65 | ) |
| 66 | } |
| 67 | private fun getL2Distance(bestAlternative: DoubleArray, worstAlternative: DoubleArray): List<DoubleArray> { |
| 68 | val l2Alternatives = mutableListOf<DoubleArray>() |
| 69 | mutableAlternative.forEach **{** |
| 70 | l2Alternatives.add( |
| 71 | doubleArrayOf( |
| 72 | Math.sqrt( |
| 73 | Math.pow(bestAlternative[0] - it.topsisDistance, 2.0) |
| 74 | + Math.pow(bestAlternative[1] - it.topsisRating, 2.0) |
| 75 | + Math.pow(bestAlternative[2] - it.topsisCost, 2.0) |
| 76 | + Math.pow(bestAlternative[3] - it.topsisAtmosphere, 2.0) |
| 77 | + Math.pow(bestAlternative[4] - it.topsisFacility, 2.0) |
| 78 | ), |
| 79 | Math.sqrt( |
| 80 | Math.pow(worstAlternative[0] - it.topsisDistance, 2.0) |
| 81 | + Math.pow(worstAlternative[1] - it.topsisRating, 2.0) |
| 82 | + Math.pow(worstAlternative[2] - it.topsisCost, 2.0) |
| 83 | + Math.pow(worstAlternative[3] - it.topsisAtmosphere, 2.0) |
| 84 | + Math.pow(worstAlternative[4] - it.topsisFacility, 2.0) |
| 85 | ) |
| 86 | ) |
| 87 | ) |
| 88 | **}** |
| 89 | return l2Alternatives |
| 90 | } |
| 91 | private fun setSimilaritytoWorstCondition(l2Distance: List<DoubleArray>) { |
| 92 | l2Distance.forEachIndexed **{** i, it **->** |
| 93 | mutableAlternative[i].similaritytoWorstCondition = it[1].div(it[0].plus(it[1])) |
| 94 | **}** |
| 95 | } |
| 96 | } |

Algoritme diatas menampilkan implementasi kode program pada kelas TOPSIS yang berfungsi untuk melakukan perhitungan TOPSIS dan memberikan rekomendasi kafe. Penjelasan potongan program, yaitu:

Baris 9, merupakan proses penyalinan daftar kafe.

Baris 10-22, merupakan fungsi perhitungan TOPSIS yang menghasilkan keluaran daftar kafe yang terurut sesuai dengan hasil perangkingan TOPSIS.

Baris 23-39, merupakan fungsi untuk mendapatkan nilai pembagi yang dibutuhkan untuk melakukan proses normalisasi.

Baris 40-48, merupakan fungsi perhitungan normalisasi setiap kriteria kafe.

Baris 49-57, merupakan fungsi untuk mencari nilai solusi ideal positif.

Baris 58-66, merupakan fungsi untuk mencari nilai solusi ideal negatif.

Baris 67-90, merupakan fungsi untuk mendapatkan jarak nilai topsis kafe menuju solusi ideal positif dan negatif.

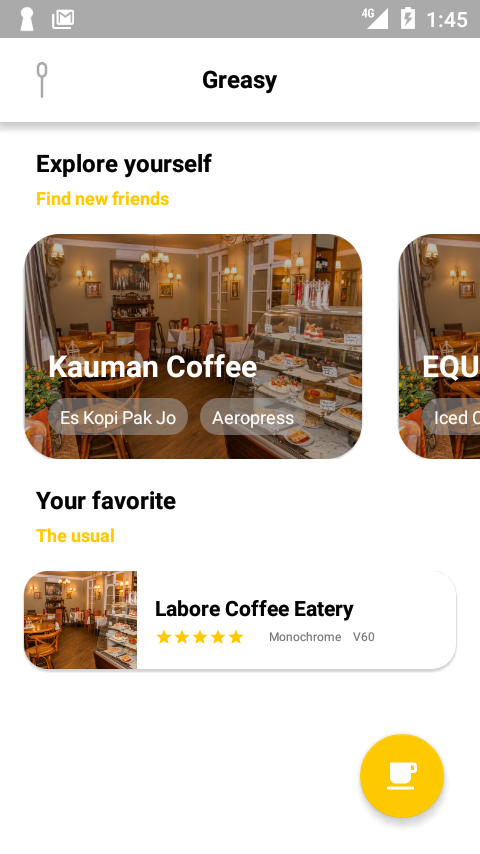
Baris 91-95, merupakan fungsi untuk mencari nilai kedekatan relatif terhadap solusi ideal.

## Implementasi Antarmuka

Sub bab ini menjelaskan hasil implementasi dari perancangan antarmuka aplikasi perangkat bergerak berdasarkan perancangan pada bab sebelumnya.

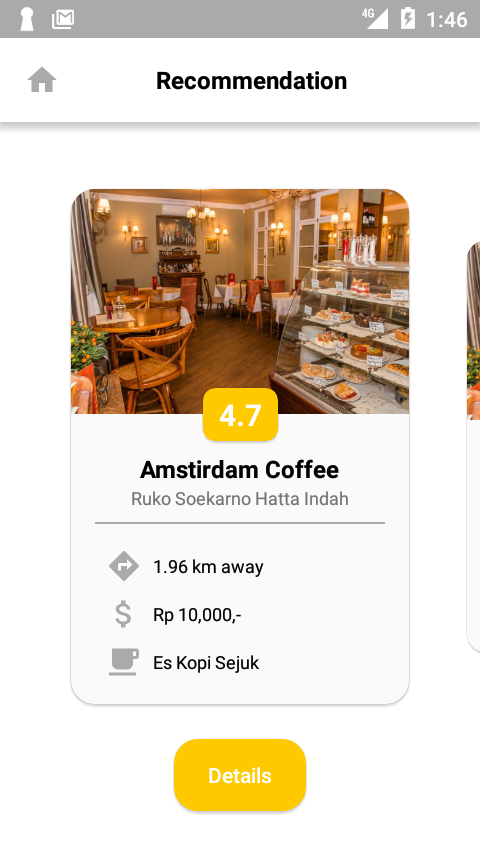
### Antarmuka Beranda

Implementasi dari halaman beranda dapat dilihat pada Gambar 5.4. Antarmuka ini diimplementasikan berdasarkan perancangan pada Gambar 4.24. Terdapat 2 bagian pada tampilan beranda, yaitu *Explore Yourself* dan *Your Favorite*. Pada bagian *Explore Yourself* pengguna dapat melihat 5 kafe yang diberikan oleh sistem secara acak, terdapat informasi seperti foto, nama kafe, biji kopi dan metode seduh yang digunakan. Pada bagian *Your Favorite* pengguna dapat melihat daftar kafe favoritnya.



Gambar 5.4 Implementasi Antarmuka Beranda

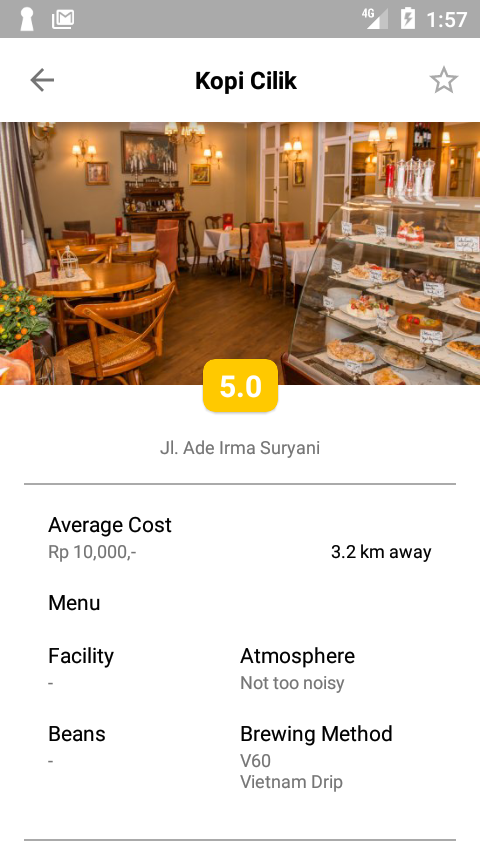
### Antarmuka Rekomendasi



Gambar 5.5 Implementasi Antarmuka Rekomendasi

Implementasi dari antarmuka rekomendasi dapat dilihat pada Gambar 5.5. Antarmuka ini diimplementasikan berdasarkan perancangan Gambar 4.25.

### Antarmuka Detail



Gambar 5.6 Implementasi Antarmuka Detail

Implementasi dari antarmuka detail dapat dilihat pada Gambar 5.6. Antarmuka ini diimplementasikan berdasarkan perancangan halaman detail pada Gambar 4.26.

# PENGUJIAN

Pengujian Validasi Algoritme

Pengujian Fungsional

Pengujian dan Analisis

Pengujian *Rank Consistency*

Pengujian Non-Fungsional

Pengujian *Usability*

Analisis Pengujian Non-Fungsional

Analisis Pengujian *Usability*

Analisis

Analisis Pengujian Fungsional

Analisis Pengujian Validasi Algoritme

Gambar 6.1 Diagram Pengujian

## Pengujian Fungsional

Pengujian fungsional dilakukan untuk mengetahui apakah sistem yang dibangun memiliki fungsi yang sesuai dengan kebutuhan. Pengujian ini dilakukan dengan menggunakan teknik pengujian *black box* yang mengacu pada *use case scenario* yang sudah dirancang sebelumnya. Berikut adalah kasus uji setiap kebutuhan fungsional dari sistem rekomendasi kafe di kota Malang. Tabel 6.1 menjelaskan prosedur kasus uji untuk pengujian fungsional melihat beranda pada sistem.

Tabel 6.1 Kasus Uji Menampilkan Beranda

|  |  |
| --- | --- |
| Nama Kasus Uji | Menampilkan Beranda |
| Tujuan | Memastikan apakah sistem mampu menampilkan konten tampilan beranda kepada pengguna sehingga pengguna dapat melihat 5 kafe acak, daftar kafe favorit, dan meminta rekomendasi. |
| Prosedur Pengujian | Membuka aplikasi |
| Hasil Yang Diharapkan | Pengguna dapat melihat 5 kafe acak, daftar kafe favoritnya dan tombol untuk meminta rekomendasi. |

Tabel 6.2 menjelaskan prosedur kasus uji untuk pengujian fungsional mendapatkan rekomendasi pada sistem.

Tabel 6.2 Kasus Uji Mendapatkan Rekomendasi

|  |  |
| --- | --- |
| Nama Kasus Uji | Mendapatkan Rekomendasi |
| Tujuan | Memastikan apakah sistem mampu menampilkan halaman pengaturan prioritas kriteria kafe dan memberikan rekomendasi kafe berdasarkan prioritas kriteria yang diberikan. |
| Prosedur Pengujian | Menekan tombol meminta rekomendasi pada halaman beranda.  Mengatur prioritas kriteria pada halaman *preference*.  Menekan tombol menyimpan pengaturan prioritas kriteria. |
| Hasil Yang Diharapkan | Pengguna dapat mengatur prioritas kafe yang sedang diinginkan dan mendapatkan 5 rekomendasi berdasarkan prioritas kriteria. |

Tabel 6.3 menjelaskan prosedur kasus uji untuk pengujian fungsional melihat detail kafe pada sistem.

Tabel 6.3 Kasus Uji Menampilkan Detail Kafe

|  |  |
| --- | --- |
| Nama Kasus Uji | Menampilkan Detail Kafe |
| Tujuan | Memastikan apakah sistem mampu menampilkan halaman detail kafe yang berisi detail informasi dari kafe yang dipilih pengguna. |
| Prosedur Pengujian | Menekan tombol melihat detail kafe pada tampilan rekomendasi kafe. |
| Hasil Yang Diharapkan | Pengguna dapat melihat detail informasi kafe yang terdiri dari: foto, rating, alamat, jarak antara pengguna dengan kafe, ragam biji kopi, metode seduh, rata-rata harga, menu, dan fasilitas dan tombol untuk mendapatkan petunjuk arah. |

Tabel 6.4 menjelaskan prosedur kasus uji untuk pengujian fungsional menambahkan kafe ke daftar favorit pada sistem.

Tabel 6.4 Kasus Uji Menambahkan Kafe Ke Daftar Favorit

|  |  |
| --- | --- |
| Nama Kasus Uji | Menambahkan Kafe Ke Daftar Favorit |
| Tujuan | Memastikan bahwa aplikasi dapat menambahkan kafe yang dipilih pengguna ke dalam daftar kafe favorit. |
| Prosedur Pengujian | Menekan ikon favorit pada bagian atas kanan aplikasi.  Kembali ke halaman beranda.  Melakukan pembaharuan tampilan. |
| Hasil Yang Diharapkan | Pengguna dapat menambahkan kafe pilihannya ke dalam daftar kafe favorit dan dapat melihatnya di daftar kafe favorit yang berada di tampilan beranda. |

Tabel 6.5 menjelaskan prosedur kasus uji untuk pengujian fungsional menghapus kafe dari daftar favorit pada sistem.

Tabel 6.5 Kasus Uji Menghapus Kafe Dari Daftar Favorit

|  |  |
| --- | --- |
| Nama Kasus Uji | Menghapus Kafe Dari Daftar Favorit |
| Tujuan | Memastikan bahwa aplikasi dapat menghapus kafe yang dipilih pengguna dari dalam daftar kafe favorit. |
| Prosedur Pengujian | Menekan ikon favorit pada bagian atas kanan aplikasi.  Kembali ke halaman beranda.  Melakukan pembaharuan tampilan. |
| Hasil Yang Diharapkan | Pengguna dapat menghapus kafe pilihannya dari dalam daftar kafe favorit. |

Tabel 6.6 menjelaskan prosedur kasus uji untuk pengujian fungsional mendapatkan petunjuk arah pada sistem.

Tabel 6.6 Kasus Uji Mendapatkan Petunjuk Arah

|  |  |
| --- | --- |
| Nama Kasus Uji | Mendapatkan Petunjuk Arah |
| Tujuan | Memastikan bahwa aplikasi dapat memberikan petunjuk arah menuju kafe yang dipilih pengguna menggunakan aplikasi Google Maps. |
| Prosedur Pengujian | Menekan tombol meminta petunjuk arah pada halaman detail kafe. |
| Hasil Yang Diharapkan | Pengguna mendapatkan petunjuk arah menuju kafe pilihan menggunakan aplikasi Google Maps. |

## Pengujian Non-Fungsional

Pengujian non-fungsional dilakukan untuk melakukan pengujian validasi algoritme dan *Usability* dari sistem yang dibangun seperti yang terdapat pada Tabel 4.3.

### Pengujian Validasi Algoritme

Pengujian validasi algoritme dilakukan dengan menguji hasil urutan rekomendasi kafe keluaran sistem dengan perhitungan manual. Apabila hasil rekomendasi yang diberikan oleh sistem sama dengan hasil rekomendasi dari perhitungan manual, maka hasil dari implementasi metode TOPSIS dapat dikatakan berhasil.

Data yang digunakan adalah semua data kafe yang ada pada basis data yang berarti 30 data kafe yang ada di kota Malang. Preferensi nilai prioritas pada setiap kriteria adalah sama untuk perhitungan manual dan sistem yaitu 1 di setiap kriteria. Dari 30 alternatif yang digunakan, hasil keluaran adalah 5 rekomendasi kafe.

Tabel 6.7 Perbandingan Hasil Perhitungan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Sistem | | Perhitungan Manual | |
| **Kafe** | **Nilai** | **Kafe** | **Nilai** |
| 1. | Nomaden Coffee | 0.94924771412 | Nomaden Coffee | 0.949689498797 |
| 2. | By Coffee | 0.802825217056 | By Coffee | 0.796618098983 |
| 3. | Bedjana Tiny House | 0.542543610006 | Bedjana Tiny House | 0.533950215106 |
| 4. | Kopi Cilik | 0.452937945829 | Kopi Cilik | 0.44913507638 |
| 5. | Labore Coffee Eatery | 0.399071939162 | Labore Coffee Eatery | 0.39265615927 |

Tabel 6.7 menampilkan hasil yang diperoleh dari perhitungan metode TOPSIS sistem dan perhitungan manual. Data alternatif yang digunakan adalah sebanyak 30 data dan memberikan keluaran berupa 5 rekomendasi kafe yang diberikan kepada pengguna. Hasil perhitungan adalah nilai kedekatan relatif terhadap solusi ideal yang kemudian diurutkan berdasarkan nilai tersebut mulai dari yang terbesar menuju terendah.

### Pengujian Rank Consistency

Pengujian *rank consistency* dilakukan dengan menambahkan alternatif dari sebuah set alternatif yang terdiri dari 5 kafe untuk melihat apakah terdapat perubahan peringkat yang signifikan dari set alternatif sebelum dan sesudah ditambahkan. Pengujian dilakukan sejumlah kafe yang terdapat pada set alternatif, yaitu 5. Di masing-masing pengujian ditambahkan satu alternatif yang bernilai sama dengan salah satu alternatif yang sudah ada. Pada Tabel 6.8 terdapat daftar kafe yang digunakan pada pengujian *rank consistency*. Pada pengujian ini preferensi prioritas terhadap kafe adalah konsisten yaitu bernilai 1 pada setiap kriteria.

Tabel 6.8 Daftar Kafe Pengujian *Rank Consistency*

|  |  |
| --- | --- |
| No. | Nama Kafe |
| 1. | Nomaden Coffee |
| 2. | By Coffee |
| 3. | Bedjana Tiny House |
| 4. | Kopi CIlik |
| 5. | Labore Coffee Eatery |

### Pengujian Usability

Pengujian *usability* dilakukan untuk mengetahui seberapa besar tingkat kebergunaan, kemudahan, dan kepuasan pengguna terhadap sistem yang dihasilkan. Proses pengujian dilakukan dengan cara memberikan aplikasi kepada responden dan meminta responden untuk mengisi kuesioner yang berisi pernyataan pendapat pengguna terhadap aplikasi. Rekapitulasi dari jawaban kuesioner yang digunakan pada pengujian *usability* dapat dilihat pada Tabel 6.9 sampai Tabel 6.10.

Tabel 6.9 Rekapitulasi Kuesioner Pengujian *Usability*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Pernyataan | Jawaban | | | | | Total |
| **STS** | **TS** | **N** | **S** | **SS** |
| **1** | **2** | **3** | **4** | **5** |
| *Usefulness* (Kebergunaan) | | | | | | | |
| 1. | Aplikasi ini membantu saya mendapatkan rekomendasi kafe. | 1 | 0 | 8 | 5 | 11 | 100 |
| 2. | Aplikasi ini membantu saya menyimpan kafe yang saya sukai. | 0 | 0 | 6 | 9 | 10 | 104 |
| 3. | Aplikasi ini membantu saya mendapatkan informasi sebuah kafe. | 0 | 0 | 7 | 9 | 9 | 102 |
| 4. | Aplikasi ini mempermudah saya dalam mendapatkan rekomendasi kafe. | 0 | 0 | 3 | 11 | 11 | 108 |
| 5. | Aplikasi ini memenuhi kebutuhan saya dalam mendapatkan rekomendasi kafe. | 0 | 0 | 5 | 18 | 2 | 97 |
| *Ease of Use* (Kemudahan Penggunaan) | | | | | | | |
| 6. | Aplikasi ini mudah untuk digunakan. | 0 | 1 | 3 | 12 | 9 | 104 |
| 7. | Aplikasi ini tidak mempersulit saya pada saat melakukan hal yang saya inginkan. | 0 | 0 | 3 | 12 | 10 | 107 |
| 8. | Semua orang dapat menggunakan aplikasi ini. | 0 | 2 | 5 | 13 | 5 | 96 |
| 9. | Saya tidak menemukan inkonsistensi pada aplikasi ini. | 1 | 1 | 5 | 11 | 7 | 97 |
| 10. | Kesalahan penggunaan aplikasi ini dapat saya atasi dengan mudah. | 0 | 1 | 6 | 12 | 6 | 98 |
| *Ease of Learning* (Kemudahan Belajar) | | | | | | | |
| 11. | Saya belajar menggunakan aplikasi ini dengan cepat. | 1 | 0 | 7 | 3 | 14 | 104 |
| 12. | Saya dapat dengan mudah mengingat bagaimana cara menggunakan aplikasi ini. | 0 | 0 | 3 | 18 | 4 | 101 |
| *Satisfaction* (Kepuasan) | | | | | | | |
| 13. | Saya merasa puas dengan apa yang dilakukan aplikasi ini. | 0 | 0 | 5 | 17 | 3 | 98 |

**Tabel 6.10 Rekapitulasi Kuesioner Pengujian *Usability* (Lanjutan)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Pernyataan | Jawaban | | | | | Total |
| **STS** | **TS** | **N** | **S** | **SS** |
| **1** | **2** | **3** | **4** | **5** |
| *Satisfaction* (Kepuasan) | | | | | | | |
| 14. | Saya ingin merekomendasikan aplikasi ini kepada teman saya. | 0 | 0 | 3 | 17 | 5 | 102 |
| 15. | Aplikasi ini bekerja sesuai dengan apa yang saya inginkan. | 0 | 1 | 3 | 18 | 3 | 98 |
| 16. | Saya membutuhkan aplikasi ini. | 0 | 0 | 5 | 16 | 4 | 99 |
| 17. | Aplikasi ini nyaman untuk digunakan. | 0 | 2 | 5 | 17 | 1 | 92 |

## Analisis Hasil Pengujian

Sub bab ini menjelaskan hasil dan kesimpulan dari pengujian aplikasi yang telah dilakukan.

### Analisis Pengujian Fungsional

Berdasarkan kasus uji masing-masing kebutuhan fungsional pada Tabel 6.1 sampai Tabel 6.6 diperoleh hasil pengujian fungsional yang dijabarkan pada Tabel 6.11.

Tabel 6.11 Hasil Pengujian Fungsional

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Kasus Uji | Hasil Yang Diharapkan | Hasil Yang Diperoleh | Status |
| Menampilkan Beranda | Pengguna dapat melihat 5 kafe acak, daftar kafe favorit dan tombol meminta rekomendasi kafe. | 5 kafe acak, daftar kafe favorit pengguna, dan tombol meminta rekomendasi berhasil ditampilkan. | Valid |
| Mendapatkan Rekomendasi | Pengguna dapat mengatur prioritas kafe yang diinginkan dan mendapatkan 5 rekomendasi kafe berdasarkan prioritas kriteria tersebut. | Prioritas kriteria pengguna berhasil disimpan dan 5 rekomendasi kafe berhasil ditampilkan. | Valid |
| Menampilkan Detail Kafe | Pengguna dapat melihat detail informasi kafe. | Detail informasi kafe berhasil ditampilkan. | Valid |
| Menambahkan Kafe Ke Daftar Favorit | Pengguna dapat menambahkan kafe ke daftar kafe favorit dan dapat melihatnya di halaman beranda. | Pesan kafe berhasil ditambahkan berhasil ditampilkan dan ketika pengguna berada di tampilan beranda kafe yang ditambahkan pengguna ada di daftar kafe favorit. | Valid |

**Tabel 6.11 Hasil Pengujian Fungsional (Lanjutan)**

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Kasus Uji | Hasil Yang Diharapkan | Hasil Yang Diperoleh | Status |
| Menghapus Kafe Dari Daftar Favorit | Pengguna dapat menghapus kafe dari daftar kafe favorit. | Pesan kafe berhasil dihapus dari daftar kafe favorit berhasil ditampilkan dan ketika pengguna berada di tampilan beranda kafe sudah tidak ada di daftar kafe favorit. | Valid |
| Mendapatkan Petunjuk Arah | Pengguna mendapatkan petunjuk arah menuju kafe pilihan melalui aplikasi Google Maps. | Aplikasi Google Maps berhasil dibuka dan menampilkan petunjuk arah menuju kafe. | Valid |

### Analisis Pengujian Non-Fungsional

Berdasarkan pengujian non-fungsional yang telah dilakukan pada sub bab sebelumnya, didapatkan analisis untuk pengujian validasi algoritme dan pengujian *usability.*

#### Analisis Pengujian Validasi Algoritme

Berdasarkan hasil pengujian validasi yang telah dilakukan pada Tabel 6.7 didapatkan hasil perbandingan antara keluaran sistem dengan perhitungan manual. Tabel 6.12 menampilkan analisis dari hasil pengujian validasi algoritme. Didapatkan status valid karena hasil keluaran sistem dengan perhitungan manual memiliki urutan rekomendasi yang sama. Sehingga penerapan metode TOPSIS dapat dikatakan berhasil.

Tabel 6.12 Hasil Pengujian Validasi Algoritme

|  |  |  |  |
| --- | --- | --- | --- |
| No | Perhitungan | | Status |
| **Sistem** | **Manual** |
| 1. | Nomaden Coffee | Nomaden Coffee | Valid |
| 2. | By Coffee | By Coffee | Valild |
| 3. | Bedjana Tiny House | Bedjana Tiny House | Valid |
| 4. | Kopi Cilik | Kopi Cilik | Valid |
| 5. | Labore Coffee Eatery | Labore Coffee Eatery | Valid |

#### Analisis Pengujian Rank Consistency

Proses perhitungan TOPSIS dengan data pada Tabel 6.8 menghasilkan urutan peringkat A1 > A2 > A3 > A4 > A5 seperti yang diilustrasikan pada Gambar 6.2. Untuk menguji konsistensi peringkat pada metode TOPSIS maka ditambahkan sebuah alternatif baru (A6). Pengujian dilakukan sebanyak lima kali dengan alternatif baru yang sama dengan kelima alternatif yang ada. Pada pengujian pertama ditambahkan alternatif yang nilainya sama dengan alternatif pertama, pengujian kedua dengan alternatif kedua, dan begitu seterusnya.

Gambar 6.2 Peringkat Awal

Gambar 6.3 Peringkat Setelah Penambahan Alternatif

Gambar 6.3 menampilkan hasil urutan peringkat setelah ditambahkan alternatif baru. Dapat dilihat bahwa tidak ada perubahan yang sangat signifikan dan tidak ada pembalikan peringkat yang terjadi pada hasil peringkat setelah alternatif baru ditambahkan. Pembalikan peringkat yang dimaksudkan disini adalah jika melihat pada hasil peringkat awal A5 berada pada peringkat ke-5, pada semua percobaan tidak ada yang menghasilkan A5 pada peringkat ke-1. Perubahan peringkat yang terjadi hanyalah pergeseran 1 peringkat keatas atau kebawah peringkat aslinya. Bahkan pada percobaan kelima peringkat yang dihasilkan sama.

#### Analisis Pengujian Usability

Tabel 6.13 Interpretasi Skor *Likert*

|  |  |  |
| --- | --- | --- |
| Skor *Likert* | Interpretasi skor | Pilihan |
| 1 | 0% - 19.99% | Sangat Tidak Setuju |
| 2 | 20% - 39.99% | Tidak Setuju |
| 3 | 40% - 59.99% | Netral |
| 4 | 60% - 79.99% | Setuju |
| 5 | 80% - 100% | Sangat Setuju |

Pada Tabel 6.13 terdapat interpretasi skala *likert* yang digunakan untuk mendapatkan indeks persentase pengujian *usability*. Proses pengujian ini dilakukan untuk melihat apakah aplikasi yang dibuat memiliki tingkat *usability* yang baik atau tidak. Pada proses pengujian ini skala *likert* digunakan untuk mendapatkan nilai *usability* aplikasi berdasarkan masukan dari 25 responden. Responden adalah kalangan mahasiswa di Fakultas Ilmu Komputer Universitas Brawijaya.

Berdasar pada persamaan 2.8 sampai 2.10 dapat didapatkan nilai indeks persentase skala *likert* untuk setiap pertanyaan.

Tabel 6.14 Hasil Perhitungan Pengujian Usability

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Pernyataan | Jawaban | | | | | Total | Indeks |
| **STS** | **TS** | **N** | **S** | **SS** |
| **1** | **2** | **3** | **4** | **5** |
| *Usefulness* (Kebergunaan) | | | | | | | | |
| 1. | Aplikasi ini membantu saya mendapatkan rekomendasi kafe. | 1 | 0 | 8 | 5 | 11 | 100 | 80% |
| 2. | Aplikasi ini membantu saya menyimpan kafe yang saya sukai. | 0 | 0 | 6 | 9 | 10 | 104 | 83.2% |
| 3. | Aplikasi ini membantu saya mendapatkan informasi sebuah kafe. | 0 | 0 | 7 | 9 | 9 | 102 | 81.6% |
| 4. | Aplikasi ini mempermudah saya dalam mendapatkan rekomendasi kafe. | 0 | 0 | 3 | 11 | 11 | 108 | 86.4% |
| 5. | Aplikasi ini memenuhi kebutuhan saya dalam mendapatkan rekomendasi kafe. | 0 | 0 | 5 | 18 | 2 | 97 | 77.6% |
| Rata-rata *Usefulness* (Kebergunaan) | | | | | | | | 81.7% |
| *Ease of Use* (Kemudahan Penggunaan) | | | | | | | | |
| 6. | Aplikasi ini mudah untuk digunakan. | 0 | 1 | 3 | 12 | 9 | 104 | 83.2% |
| 7. | Aplikasi ini tidak mempersulit saya pada saat melakukan hal yang saya inginkan. | 0 | 0 | 3 | 12 | 10 | 107 | 85.6% |
| 8. | Semua orang dapat menggunakan aplikasi ini. | 0 | 2 | 5 | 13 | 5 | 96 | 76.8% |
| 9. | Saya tidak menemukuan inkonsistensi pada aplikasi ini. | 1 | 1 | 5 | 11 | 7 | 97 | 77.6% |
| 10. | Kesalahan penggunaan aplikasi ini dapat saya atasi dengan mudah. | 0 | 1 | 6 | 12 | 6 | 98 | 78.4% |
| Rata-rata *Ease of Use* (Kemudahan Penggunaan) | | | | | | | | 80.3% |

**Tabel 6.14 Hasil Perhitungan Pengujian Usability (Lanjutan)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Pernyataan | Jawaban | | | | | Total | Indeks |
| **STS** | **TS** | **N** | **S** | **SS** |
| **1** | **2** | **3** | **4** | **5** |
| *Ease of Learning* (Kemudahan Belajar) | | | | | | | | |
| 11. | Saya belajar menggunakan aplikasi ini dengan cepat. | 1 | 0 | 7 | 3 | 14 | 104 | 83.2% |
| 12. | Saya dapat dengan mudah mengingat bagaimana cara menggunakan aplikasi ini. | 0 | 0 | 3 | 18 | 4 | 101 | 80.8% |
| Rata-rata *Ease of Learning* (Kemudahan Belajar) | | | | | | | | 82% |
| *Satisfaction* (Kepuasan) | | | | | | | | |
| 13. | Saya merasa puas dengan apa yang dilakukan aplikasi ini. | 0 | 0 | 5 | 17 | 3 | 98 | 78.4% |
| 14. | Saya ingin merekomendasikan aplikasi ini kepada teman saya. | 0 | 0 | 3 | 17 | 5 | 102 | 81.6% |
| 15. | Aplikasi ini bekerja sesuai dengan apa yang saya inginkan. | 0 | 1 | 3 | 18 | 3 | 98 | 78.4% |
| 16. | Saya membutuhkan aplikasi ini. | 0 | 0 | 5 | 16 | 4 | 99 | 79.2% |
| 17. | Aplikasi ini nyaman untuk digunakan. | 0 | 2 | 5 | 17 | 1 | 92 | 73.6% |
| Rata-rata *Satisfaction* (Kepuasan) | | | | | | | | 78.2% |

Berikut adalah contoh perhitungan untuk mendapatkan nilai indeks di Tabel 6.14. Pada nomor 17 didapatkan skor menggunakan persamaan 2.9 sejumlah 95 dan Y bernilai 125 menggunakan persamaan 2.10. Jika kedua nilai tersebut dihitung dengan persamaan 2.11 maka dihasilkan nilai 73.6%.

Tabel 6.15 menampilkan hasil status pengujian *usability*. Hasil status pengujian menghasilkan rata-rata penilaian sebesar 80.55% yang sama dengan status sangat setuju yang berarti dalam konteks pengujian ini adalah sangat setuju bahwa aplikasi dibutuhkan, mudah digunakan, mudah dipelajari, dan dapat memberikan kepuasan ketika aplikasi digunakan.

Tabel 6.15 Hasil Status Pengujian *Usability*

|  |  |  |
| --- | --- | --- |
| Aspek penilaian | Status | Rata-rata penilaian (%) |
| *Usefulness* | Sangat setuju | 81.7% |
| *Ease of Use* | Sangat setuju | 80.3% |
| *Ease of Learning* | Sangat setuju | 82% |
| *Satisfaction* | Sangat setuju | 78.2% |
| Rata-rata | | 80.55% |

# PENUTUP

## Kesimpulan

Berdasarkan hasil analisis perancangan, implementasi, dan pengujian yang telah dilakukan, dapat diambil beberapa kesimpulan sebagai berikut:

Pada pengujian fungsional didapatkan hasil dengan nilai persentase 100% valid yang berarti bahwa aplikasi telah berjalan sesuai dengan kebutuhan yang ada. Pada pengujian validasi algoritme didapatkan hasil bahwa keluaran sistem dengan perhitungan manual adalah sama, yang berarti bahwa metode TOPSIS berhasil diterapkan dan dapat memberikan hasil yang sesuai. Pada pengujian *rank consistency* didapatkan hasil tidak adanya perubahan peringkat yang signifikan terjadi ketika dilakukan proses penambahan alternatif. Sedangkan pengujian *usability* didapatkan nilai persentase rata-rata sebesar 80.55% yang membuktikan bahwa aplikasi dibutuhkan, mudah digunakan, mudah dipelajari, dan dapat memberikan kepuasan ketika digunakan.

## Saran

Saran yang dapat diberikan untuk pengembangan sistem rekomendasi kafe di kota Malang berbasis lokasi selanjutnya, antara lain:

Sistem dapat menerapkan metode tambahan yang dapat membantu pengguna dalam menentukan prioritas kriterianya.

Dapat diberikan fitur untuk pengguna dapat memberikan review dan rating kepada kafe.

DAFTAR RUJUKAN

Agile Business, 2014. *The DSDM Agile Project Framework* [online] Tersedia di: <https://www.agilebusiness.org/resources/dsdm-handbooks/the-dsdm-agile-project-framework-2014-onwards> [Diakses 9 November 2018]

Darmastuti, D., 2013. Implementasi Metode Simple Additive Weighting (SAW) dalam Sistem Informasi Lowongan Kerja Berbasis Web untuk Rekomendasi Pencari Kerja Terbaik. *Jurnal Sistem dan Teknologi Informasi (JustIN), 1*(2), pp.114-119.

Eniyati, S., 2011. Perancangan Sistem Pendukung Pengambilan Keputusan untuk Penerimaan Beasiswa dengan Metode SAW (Simple Additive Weighting). *Dinamik-Jurnal Teknologi Informasi, 16*(2).

Habenicht, W., Scheubrein, B. and Scheubrein, R., 2002. Multiple criteria decision making. Theme, 6(5).

Hamka, M., Utami, E. and Amborowati, A., 2014. Sistem Pendukung Keputusan Kelompok Metode Topsis dan Borda untuk Penentuan Bakal Calon Haji. *SEMNASTEKNOMEDIA ONLINE, 2*(1), pp.2-04.

Hasanah, N.U. 2017. Selain Penambahan Pendatang, Mahasiswa Juga Punya Sumbangan Lain ke Kota Malang [online] Tersedia di: <http://suryamalang.tribunnews.com/2017/04/01/selain-penambahan-pendatang-mahasiswa-juga-punya-sumbangan-lain-ke-kota-malang-yaitu> [Diakses 27 Agustus 2018]

Junior, F.R.L., Osiro, L. and Carpinetti, L.C.R., 2014. A comparison between Fuzzy AHP and Fuzzy TOPSIS methods to supplier selection. Applied Soft Computing, 21, pp.194-209.

Kumar, K.N., Akhi, K., Gunti, S.K. and Reddy, M.S.P., 2016. Implementing smart home using firebase. *International Journal of Research in Engineering and Applied Sciences, 6*(10), pp.193-198.

Kurniasih, D.L., 2017. Sistem pendukung keputusan pemilihan laptop dengan metode TOPSIS. *Pelita Informatika: Informasi dan Informatika, 3*(2).

Malang Post, 2017. Disertifikasi, Ikuti Standar Kafe dan Resto [online] Tersedia di: <https://www.malang-post.com/berita/kota-malang/disertifikasi-ikuti-standar-kafe-dan-resto> [Diakses 27 Agustus 2018]

Mustaqbal, M.S., Firdaus, R.F. and Rahmadi, H., 2016. Pengujian Aplikasi Menggunakan Black Box Testing Boundary Value Analysis (Studi Kasus: Aplikasi Prediksi Kelulusan SNMPTN). *Jurnal Ilmiah Teknologi Informasi Terapan, 1*(3).

Putra, R.H.D., Sujaini, H. and Safriadi, N., 2015. Penerapan Metode Haversine Formula Pada Sistem Informasi Geografis Pengukuran Luas Tanah. *Jurnal Sistem dan Teknologi Informasi (JUSTIN), 4*(1), pp.157-162.

Riandari, F., Hasugian, P.M. and Taufik, I., 2017. Sistem Pendukung Keputusan Menggunakan Metode Topsis Dalam Memilih Kepala Departemen Pada Kantor Balai Wilayah Sungai Sumatera II Medan. *Journal Of Informatic Pelita Nusantara, 2*(1).

Rouf, A., 2012. Pengujian perangkat lunak dengan menggunakan metode white box dan black box. *Himsyatech, 8*(1).

Sebastia, L., Garcia, I., Onaindia, E. and Guzman, C., 2009. e-Tourism: a tourist recommendation and planning application. *International Journal on Artificial Intelligence Tools, 18*(05), pp.717-738.

Shih, H.S., Shyur, H.J. and Lee, E.S., 2007. An extension of TOPSIS for group decision making. *Mathematical and Computer Modelling, 45*(7-8), pp.801-813.

Subakti, Irfan. 2002. *Sistem Pendukung Keputusan (Decision Support System)*. Surabaya: Jurusan Teknik Informatika FTI, Institut Teknologi Sepuluh Nopember.

Wibisono, W. and Baskoro, F., 2002. Pengujian Perangkat Lunak dengan Menggunakan Model Behaviour UML. *JUTI: Jurnal Ilmiah Teknologi Informasi, 1*(1), pp.43-50.

Zanakis, S.H., Solomon, A., Wishart, N. and Dublish, S., 1998. Multi-attribute decision making: a simulation comparison of select methods. *European journal of operational research, 107*(3), pp.507-529.