

**SELEKSI PEMASOK KONSTRUKSI MENGGUNAKAN
METODE *WEIGHTED PRODUCT* BERBASIS *MOBILE* DI PT
BINARTHAMA KHARISMA**

SKRIPSI

Disusun sebagai salah satu syarat untuk memperoleh Gelar Sarjana Teknik pada
Program Studi Teknik Informatika Universitas Sangga Buana YPKP



Disusun Oleh:

ANDREW BALANGO

B1031511RB4001

FAKULTAS TEKNIK

JURUSAN TEKNIK INFORMATIKA

UNIVERSITAS SANGGA BUANA YPKP

BANDUNG

2019

LEMBAR PERSEMBAHAN

Kepada Bapak, Ir Riswandi Wiradipura & Ibu, Idah Farida S.Pd

Terima kasih atas segala dukungan kalian, baik dalam bentuk materi maupun moril. Tugas akhir ini saya persembahkan untuk kalian, sebagai wujud rasa terima kasih atas pengorbanan dan jerih payah kalian sehingga saya dapat menggapai cita - cita. Semoga Kebaikan dan kemurahan Hati Bpk Riswandi & Ibu Idah Farida beserta keluarga, di balas Oleh Allah SWT.

Tak lupa juga saya mengucapkan Terima Kasih sebesar-besarnya untuk pelajaran hidup yang saya dapatkan selama tinggal dengan Bapak & Ibu, Mohon Maaf dengan ke khilafan saya kepada Bapak & Ibu.

LEMBAR PERNYATAAN KEASLIAN TUGAS AKHIR

Saya yang bertanda tangan di bawah ini,

Nama : Andrew Balango
NPM : B1031511RB4001
Jurusan : Teknik Informatika
Tempat, tanggal lahir : Manado, 29 Desember 1993
Alamat orang tua : Jl. Gegerkalong Girang No.138

Menyatakan bahwa laporan tugas akhir ini adalah benar, bila terbukti tidak demikian, saya bersedia menerima segala akibatnya, termasuk pencabutan kembali Laporan Tugas Akhir yang telah saya kerjakan.

Bandung, 13 Februari 2020

Andrew Balango
B1031511RB4001

**LEMBAR PERSETUJUAN DAN
PENGESAHAN TUGAS AKHIR**

Karya Tulis Oleh : Andrew Balango
NPM : B1031511RB4001
Judul : SELEKSI PEMASOK KONTRUKSI
MENGGUNAKAN METODE WEIGHTED
PRODUCT BERBASIS MOBILE DI
PT.BINARTHAMA KHARISMA
.

Telah menyetujui laporan tugas akhir dengan prosedur dan ketentuan yang berlaku.

Disetujui Oleh :

PEMBIMBING 1 TUGAS AKHIR

PEMBIMBING 2 TUGAS AKHIR

Slamet Risnanto, ST.,M.Kom

Phitsa Mauliana, MM.,M.Kom

Mengetahui :

KETUA PRODI S1 TEKNIK INFORMATIKA

Riffa Haviani Laluma. S.Kom., MT

PERSETUJUAN DAN PENGESAHAN TUGAS AKHIR

Tugas Akhir ini diajukan oleh :

NPM : B1031511RB4001
Nama : ANDREW BALANGO
Program Studi : TEKNIK INFORMATIKA
Judul : **SELEKSI PEMASOK KONTRUKSI MENGGUNAKAN
METODE WEIGHTED PRODUCT BERBASIS MOBILE DI
PT.BINARTHAMA KHARISMA**

Untuk dipertahankan pada sidang Tugas Akhir Semester Genap Tahun 2020 dihadapan para penguji dan diterima sebagai bagian dari persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik (ST) pada Fakultas Teknik program Studi S1 Teknik Informatika Universitas Sangga Buana YPKP

Penguji I

Penguji II

TEGUH NURHADI S.,ST.,MT.

DHEMA YUNAUTAMA, ST.,M.KOM.

KATA PENGANTAR

Segala puji dan syukur penulis panjatkan kehadirat Allah SWT, atas segala limpahan Nikmat Sehat, Rahmat, Hidayat, dan Karunia-Nya sehingga penulis dapat menyelesaikan Laporan Tugas Akhir ini. Adapun judul dari Laporan Tugas Akhir ini adalah “APLIKASI SELEKSI PEMASOK KONTRUKSI MENGGUNAKAN METODE WEIGHTED PRODUCT BERBASIS MOBILE DI PT.BINARTHAMA KHARISMA” yang penulis buat untuk memenuhi salah satu syarat dalam menyelesaikan studi program S1 Teknik Informatika.

Penulis mengucapkan rasa hormat dan banyak terimakasih kepada semua pihak yang telah membantu dan membimbing penulis baik secara moril, dan materil, baik secara langsung maupun tidak langsung. Untuk itu pada kesempatan ini, penulis mengucapkan rasa hormat dan terimakasih kepada :

1. Dr. H. Asep Effendi, SE., M.Si., PIA., CFra., CRBC, selaku Ketua Rektor Universitas Sangga Buana YPKP.
2. Riffa Haviani Laluma. S.Kom., MT, selaku Ketua Prodi S1 Teknik Informatika
3. Slamet Risnanto, ST.,M.Kom selaku pembimbing 1 di Universitas Sangga Buana YPKP
4. Phitsa Mauliana, MM.,M.Kom selaku pembimbing 2 di Universitas Sangga Buana YPKP
5. Bpk, Ir Riswandi Wiradipura, Ibu, Idah Farida, S.pd. dan semua anggota keluarga yang telah memberi semangat dan pengorbanan dalam bentuk apapun.
6. Hartina, S.E., selaku orang spesial yang selalu memberi semangat serta bantuan baik doa maupun saran dalam pembuatan Laporan Tugas Akhir ini.
7. Rekan kerja di PT. Binarthama Kharisma yang telah mengerti dan mendukung penulis

dalam pembuatan Laporan Tugas Akhir ini.

Dalam menyusun laporan ini penulis telah berusaha semaksimal mungkin, tetapi penulis menyadari bahwa laporan ini masih jauh dari sempurna, untuk itu penulis mohon maaf atas keterbatasan yang ada dan menanti kritik dan saran dari pembaca yang sangat penulis harapkan untuk perbaikan.

Akhir kata, semoga laporan ini dapat bermanfaat bagi semua pembaca khususnya untuk mahasiswa/i Universitas Sangga Buana YPKP dan masyarakat pada umumnya.

Bandung, 13 Februari 2020

Andrew Balango

ABSTRAK

Andrew Balango (B1031511RB4001), Seleksi Pemasok Menggunakan Metode *Weighted Product* di PT Binarthama Kharisma

Sumber daya konstruksi meliputi usaha pemasok material, peralatan, teknologi, serta tenaga kerja konstruksi. Perhitungan keandalan rantai pasok harus didukung dengan data yang akurat tentang jumlah deposit, kualitas deposit, lokasi dan transportasi menuju proyek, pengaturan waktu kedatangan dan area transit. Sekarang ini banyak perusahaan yang melakukan pemilihan atau seleksi dari pemasoknya. Seleksi pemasok adalah tahap proses pembelian industri yang biasanya dilakukan dengan cara meninjau proposal dan memilih pemasok. Perusahaan pembeli akan mempertimbangkan banyak atribut dari pemasok beserta kepentingan relatifnya. PT Binarthama Kharisma merupakan salah satu perusahaan yang bergerak di bidang konstruksi. Terdapat beberapa kendala yang harus dibenahi pada perusahaan tersebut diantaranya Adm. Logistik mengalami kesulitan menerapkan beberapa kriteria yang diinginkan dalam memilih pemasok yang tepat dan dapat diandalkan untuk proses konstruksi. Beberapa atribut dan metode yang digunakan masih belum sesuai dengan kebutuhan perusahaan. Kemudian belum adanya suatu aplikasi *mobile* seleksi pemasok konstruksi yang digunakan pada PT Binarthama Kharisma. Tujuan dari penelitian ini yaitu membangun aplikasi seleksi pemasok konstruksi menggunakan metode *Weighted Product* berbasis *mobile* di PT Binarthama Kharisma. Hasil penelitian menunjukkan bahwa Adm. Logistik dimudahkan dalam memilih pemasok konstruksi terbaik dari urutan *ranking* yang dihasilkan. Beberapa kriteria yang digunakan pada metode *Weighted Product* sangat sesuai dengan kebutuhan perusahaan. Aplikasi *mobile* seleksi pemasok konstruksi berhasil dibangun menggunakan pemrograman Android di PT Binarthama Kharisma.

Kata Kunci: Aplikasi Android, Seleksi Pemasok, Metode *Weighted Product*, PT Binarthama Kharisma

ABSTRACT

Andrew Balango (B1031511RB4001), *Supplier Selection Application Using Weighted Product Method in PT Binarthama Kharisma*

Construction resources include the business of supplying materials, equipment, technology, and construction labor. Calculation of the reliability of the supply chain must be supported by accurate data on the amount of the deposit, deposit quality, location and transportation to the project, time of arrival and transit area. Nowadays many companies do the selection or selection from their suppliers. Supplier selection is the stage of the industry purchasing process that is usually done by reviewing proposals and selecting suppliers. The buying company will consider many of the attributes of the supplier and their relative importance. PT Binarthama Kharisma is one of the companies engaged in construction. There are several obstacles that must be addressed in the company including Adm. Logistics have difficulty applying several desired criteria in choosing the right and reliable supplier for the construction process. Some attributes and methods used are still not in accordance with company needs. Then there is no mobile application for construction supplier selection used at PT Binarthama Kharisma. The purpose of this study is to build a construction supplier selection application using the mobile-based Weighted Product method at PT Binarthama Kharisma. The results showed that Adm. Logistics are made easy in choosing the best construction supplier from the ranking sequence produced. Some of the criteria used in the Weighted Product method are very suitable to the needs of the company. The mobile application for construction supplier selection was successfully built using Android programming at PT Binarthama Kharisma.

Keywords: *Android Application, Supplier Selection, Weighted Product Method, PT Binarthama Kharisma*

DAFTAR ISI

LEMBAR PERSEMBAHAN	i
LEMBAR PERNYATAAN KEASLIAN TUGAS AKHIR	ii
LEMBAR PERSETUJUAN DAN PENGESAHAN TUGAS AKHIR	iii
KATA PENGANTAR	iv
ABSTRAKSI	v
ABSTRACT	vi
DAFTAR ISI	vii
DAFTAR TABEL	viii
DAFTAR GAMBAR	ix
BAB I PENDAHULUAN	1
1.1. Latar Belakang Masalah	1
1.2. Identifikasi Masalah	3
1.3. Perumusan Masalah	4
1.4. Maksud dan Tujuan	4
1.5. Metode Penelitian	5
1.5.1. Teknik Pengumpulan Data	5
1. Observasi	5
2. Wawancara	5
3. Studi Pustaka	5
1.5.2 Metode Pengembangan Aplikasi	5
1. Analisis Kebutuhan Sistem	6
2. Desain	6
3. Code Generation	6
4. Testing	6
5. Support	6
1.6. Ruang Lingkup	7
BAB II PENDAHULUAN	9
2.1. Tinjauan Jurnal	9
2.2. Konsep Dasar Program	18
2.2.1. Definisi Program Komputer	18
2.2.2. Pemrograman OOP	19
2.2.3. Pemrograman Android	21
2.3. Metode Algoritma	25
2.3.1. Definisi Algoritma	25
2.3.2. Metode Weighted Product	26
2.4. Pengujian Aplikasi	27
2.5. Peralatan Pendukung	30
2.3.3. UML	30
1. Use Case Diagram	30
2. Activity Diagram	31
3. Class Diagram	32
4. Sequence Diagram	33
2.3.4. Definisi Kriteria	37

BAB III PENDAHULUAN	24
3.1. Tinjauan Perusahaan	35
3.1.1. Sejarah Perusahaan	35
3.1.2. Struktur Organisasi dan Fungsi	36
3.2. Proses Bisnis Sistem	40
3.3. Spesifikasi Dokumen Sistem Berjalan	42
BAB IV RACANGAN SISTEM DAN PROGRAM USULAN	43
4.1 Analisa Kebutuhan Software	43
A. Tahapan Analisis	43
B. Use Case Diagram	43
C. Activity Diagram	52
4.2. Desain	59
4.2.2. Software Architecture	60
A. Class Diagram	60
B. Sequence Diagram	61
C. Component Diagram	72
D. Deployment Diagram	73
4.2.3 User Interface	74
1. Halaman Menu Utama (Frontend)	74
2. Halaman Login	75
3. Halaman Backend	76
4. Halaman Kriteria	77
5. Halaman Pemasok	78
6. Halaman Nilai	79
7. Halaman Ranking	80
8. Halaman Informasi	81
4.3 Code Genertion	82
1. <i>Class</i> Pengguna	82
2. <i>Class</i> Kriteria	84
3. <i>Class</i> Pemasok	87
4. <i>Class</i> Nilai	90
4.4. Testing	102
4.5. Support	105
4.5.1. Publikasi	105
1. Biaya Publikasi	105
2. Jadwal Publikasi	106
4.5.2. Spesifikasi Hardware dan Software	106
4.6. Spesifikasi Dokumen Sistem Usulan	107
BAB V PENUTUP	108
5.1. Kesimpulan	108
5.2. Saran	108

DAFTAR TABEL

Tabel II.1. Penelitian Terkait.....	14
Tabel II.2 Kriteria dan Bobot.....	28
Tabel II.3 <i>Rating</i> Alternatif	29
Tabel II.4 Hasil Perangkingan	31
Tabel IV.1. Deskripsi Use Case Melakukan Login.....	45
Tabel IV.2 Deskripsi Use Case Mengelola Kriteria.....	46
Tabel IV.3. Deskripsi Use Case Mengelola Pemasok.....	47
Tabel IV.4. Deskripsi Use Case Mengelola Nilai.....	48
Tabel IV.5. Deskripsi Use Case Melakukan Logout.....	49
Tabel IV.6. Deskripsi Use Case Mengakses Rangking.....	50
Tabel IV.7. Deskripsi Use Case Mengakses Informasi.....	51
Tabel IV.8. Jadwal Publikasi Aplikasi.....	106
Tabel IV.9. Spesifikasi Smartphone.....	106

DAFTAR GAMBAR

Gambar II.1. Pemrosesan Algoritma.....	18
Gambar II.2. Komponen-komponen Utama Android.....	21
Gambar II.4. Contoh Use Case Diagram.....	31
Gambar II.5. Contoh Activity Diagram.....	32
Gambar II.6. Contoh Class Diagram.....	33
Gambar II.7. Contoh Sequence Diagram.....	34
Gambar III.1. Struktur Organisasi PT Binarthama Kharisma Engineering Consultant.....	36
Gambar III.2. Activity Diagram Seleksi Pemasok	41
Gambar IV.1. Use Case Diagram Sistem Pendukung Keputusan.....	44
Gambar IV.2. Activity Diagram Melakukan Login.....	52
Gambar IV.3. Activity Diagram Mengelola Kriteria.....	53
Gambar IV.4. Activity Diagram Mengelola Pemasok.....	54
Gambar IV.5. Activity Diagram Mengelola Nilai.....	55
Gambar IV.6. Activity Diagram Melakukan Logout.....	56
Gambar IV.7. Activity Diagram Mengakses Rangkang.....	57
Gambar IV.8. Activity Diagram Mengakses Informasi.....	58
Gambar IV.9. ERD Aplikasi Seleksi Pemasok.....	59
Gambar IV.10. Class Diagram Aplikasi Seleksi pemasok konstruksi.....	60
Gambar IV.11. Sequence Diagram Melakukan Login.....	61
Gambar IV.12. Sequence Diagram Menambah Kriteria.....	62
Gambar IV.13. Sequence Diagram Mengubah Kriteria.....	63
Gambar IV.14. Sequence Diagram Menghapus Kriteria.....	64
Gambar IV.15. Sequence Diagram Menambah Pemasok.....	65
Gambar IV.16. Sequence Diagram Mengubah Pemasok.....	66
Gambar IV.17. Sequence Diagram Menghapus Pemasok.....	67
Gambar IV.18. Sequence Diagram Menambah Nilai.....	68

Gambar IV.19. Sequence Diagram Mengubah Nilai.....	69
Gambar IV.20. Sequence Diagram Menghapus Nilai.....	70
Gambar IV.21. Sequence Diagram Melakukan Logout.....	71
Gambar IV.22. Sequence Diagram Mengakses Rangking.....	72
Gambar IV.23. Sequence Diagram Mengakses Informasi.....	72
Gambar IV.24. Component Diagram Aplikasi Seleksi Pemasok Konstruksi.....	73
Gambar IV.25. Deployment Diagram Aplikasi Seleksi Pemasok Konstruksi.....	74
Gambar IV.26. Tampilan Halaman Frontend.....	75
Gambar IV.27. Tampilan Halaman Login.....	76
Gambar IV.28. Tampilan Halaman Backend.....	77
Gambar IV.29. Tampilan Halaman Kriteria.....	78
Gambar IV.30. Tampilan Halaman Pemasok.....	79
Gambar IV.31. Tampilan Halaman Nilai.....	80
Gambar IV.32. Tampilan Halaman Rangking.....	81
Gambar IV.33. Tampilan Halaman Informasi.....	82
Gambar IV.34. Flow Chart Aplikasi Seleksi Pemasok Konstruksi.....	103
Gambar IV.35. Grafik Alir Aplikasi Seleksi pemasok konstruksi.....	104

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Pembangunan infrastruktur skala besar yang menjadi prioritas pemerintah Indonesia harus didukung dengan industri konstruksi dan kesiapan rantai pasok sumber daya konstruksi yang handal. Sumber daya konstruksi meliputi usaha pemasok bahan bangunan/material, usaha pemasok peralatan konstruksi, usaha pemasok teknologi konstruksi, serta usaha pemasok tenaga kerja konstruksi. Perhitungan keandalan rantai pasok harus didukung dengan data yang akurat tentang jumlah deposit, kualitas deposit, lokasi dan transportasi menuju proyek, pengaturan waktu kedatangan dan area transit (Dewi *et al.*, 2018).

Sekarang ini banyak perusahaan yang melakukan pemilihan atau seleksi dari pemasoknya. Seleksi pemasok adalah tahap proses pembelian industri yang biasanya dilakukan dengan cara meninjau proposal dan memilih pemasok. Perusahaan pembeli akan mempertimbangkan banyak atribut dari pemasok beserta kepentingan relatifnya. Atribut-atribut dalam seleksi pemasok dapat mencakup kualitas produk dan layanan, reputasi, pengiriman yang tepat waktu, perilaku perusahaan yang etis, komunikasi yang jujur, serta harga yang kompetitif. Pada akhirnya, perusahaan pembeli dapat memilih pemasok tunggal atau beberapa pemasok saja (Kotler *et al.*, 2015).

PT. Binarthama Kharisma merupakan salah satu perusahaan yang melakukan manajemen anggaran sebagai antisipasi terhadap kebutuhan akan jasa teknik yang makin meningkat dimasa mendatang. Jasa teknik yang sangat berperan dalam menunjang lajunya pembangunan adalah jasa dalam bidang Informatika dan Telekomunikasi, Teknik Sipil dan Geoteknik, Teknik Lingkungan dan Geomatika yang saling terkait satu sama lain. Pengalaman konsultasi yang tajam dan mantap seiring dengan pengembangan usaha sektor lainnya akan memberikan saham yang besar dalam mencapai sasaran yang tepat serta nilai guna yang tinggi.

Terdapat beberapa kendala yang harus dibenahi dari hasil observasi yang telah dilakukan pada PT. Binarthama Kharisma. Bagian pembelian bahan baku kesulitan menerapkan beberapa kriteria yang diinginkan dalam memilih pemasok yang tepat dan dapat diandalkan untuk proses konstruksi. Beberapa atribut dan metode yang digunakan pada proses pemilihan pemasok masih belum sesuai dengan kebutuhan perusahaan. Kemudian belum adanya suatu aplikasi *mobile* seleksi pemasok konstruksi yang digunakan pada PT. Binarthama Kharisma. Oleh karena itu, pada permasalahan-permasalahan tersebut dibutuhkan proses kualifikasi dan optimasi untuk membuat keputusan yang objektif.

Suatu seleksi pemasok konstruksi dapat dioptimalkan menggunakan sistem penunjang keputusan (SPK) menggunakan metode *Weighted Product*. Terdapat penelitian mengenai SPK penilaian kinerja kasir Lotte Mart menggunakan metode *Weighted Product*. Hasil penelitian ini menunjukkan bahwa SPK memudahkan personalia melakukan penilaian kinerja Kasir menggunakan metode *Weighted Product* (Mirawati *et al.*, 2018). Kemudian penelitian SPK evaluasi kinerja pramuniaga Toserba Yogya Ciwalk menggunakan metode *Weighted Product* yang

memberikan kemudahan bagi personalia dalam melakukan penilaian kinerja terhadap Pramuniaga menggunakan metode *Weighted Product* (Mauliana *et al.*, 2018).

Dari permasalahan sebelumnya, maka penulisan skripsi ini berfokus pada seleksi pemasok konstruksi dengan menerapkan metode *Weighted-Product* menggunakan pemrograman *mobile*. Terdapat penelitian berbasis *mobile* yang dapat membantu dalam proses bisnis pada suatu perusahaan. Pada penelitian tersebut dilakukan pembangunan suatu aplikasi *mobile* yang berhasil menggunakan pemrograman Android (Wiguna & Alawiyah, 2019). Sehingga judul pada penulisan skripsi ini yaitu seleksi pemasok konstruksi menggunakan metode *Weighted Product* berbasis *mobile* di PT. Binarthama Kharisma.

1.2. Identifikasi Masalah

Berdasarkan uraian pada bagian latar belakang masalah dari kasus yang diteliti, maka dapat diidentifikasi permasalahan dari penelitian ini sebagai berikut:

1. Bagian pembelian bahan baku kesulitan dalam memilih pemasok yang tepat dan dapat diandalkan untuk proses konstruksi.
2. Belum adanya metode yang tepat beserta kriteria-kriteria yang sesuai dengan kebutuhan perusahaan pada seleksi pemasok konstruksi.
3. Belum adanya suatu aplikasi *mobile* seleksi pemasok konstruksi yang dapat digunakan di PT. Binarthama Kharisma.

1.3. Perumusan Masalah

Perumusan masalah pada penelitian ini merupakan rumusan konkrit dari masalah yang diteliti dalam bentuk pertanyaan penelitian, antara lain:

1. Bagaimana memudahkan bagian pembelian bahan baku dalam memilih pemasok yang tepat dan dapat diandalkan untuk proses konstruksi?
2. Bagaimana menerapkan metode yang tepat pada seleksi pemasok konstruksi dan menentukan beberapa atribut yang sesuai dengan kebutuhan perusahaan?
3. Bagaimana membangun suatu aplikasi *mobile* seleksi pemasok konstruksi yang dapat digunakan di PT. Binarthama Kharisma?

1.4. Maksud dan Tujuan

Maksud dari penyusunan skripsi ini menjelaskan kontribusi dari rancangan aplikasi dan metode yang akan diterapkan sebagai berikut:

1. Memudahkan bagian pembelian bahan baku dalam memilih pemasok yang tepat dan dapat diandalkan untuk proses konstruksi.
2. Menerapkan metode *Weighted Product* pada seleksi pemasok konstruksi dan menentukan beberapa atribut yang sesuai dengan kebutuhan perusahaan.
3. Membangun aplikasi seleksi pemasok konstruksi menggunakan pemrograman Android yang dapat digunakan di PT. Binarthama Kharisma.

Tujuan penulisan skripsi ini adalah untuk melengkapi salah satu syarat yang telah ditentukan dalam mencapai kelulusan Program Strata 1 (S1) Program Studi Teknik Informatika pada Universitas Sangga Buana YPKP.

1.5. Metode Penelitian

1.5.1. Teknik Pengumpulan Data

Pada tahap ini terdapat beberapa mekanisme dalam melakukan penelitian pada tempat riset diantaranya:

1. Observasi

Dilakukan pengamatan langsung atau peninjauan secara cermat pada PT. Binarthama Kharisma. Ditinjau dokumen operasional yang sedang berjalan dan analisa *Standard Operational Procedure* (SOP) pada perusahaan.

2. Wawancara

Melakukan tanya jawab secara tatap muka terhadap bagian pembelian atau *purchasing*, mandor konstruksi, serta *stakeholder* selaku pengambil keputusan pada tempat riset.

3. Studi Pustaka

Studi pustaka dan sumber referensi yang digunakan pada penelitian ini yaitu beberapa jurnal dan buku yang berkaitan dengan sistem penunjang keputusan, metode *Weighted Product*, dan pemrograman Android.

1.5.2. Metode Pengembangan Aplikasi

Pengembangan aplikasi yang akan dibangun pada penelitian ini yaitu menggunakan model *Waterfall* yang dibagi menjadi beberapa tahap yaitu:

1. Analisa Kebutuhan Sistem

Menjelaskan mengenai kebutuhan fungsional aplikasi yang akan dikembangkan di tempat riset. Pada bagian ini digambarkan *use case diagram* dan *activity diagram* terkait dengan proses bisnis yang dinarasikan.

2. Desain

Spesifikasi perancangan sesuai dengan kebutuhan yang terkait dengan pengembangan aplikasi. Pada perancangan *database* menggunakan *Entity Relationship Diagram* (ERD) dan *Logical Record Structure* (LRS). Sedangkan *software architecture* menggunakan *Unified Modelling Language* (UML).

3. Code Generation

Bahasa pemrograman yang akan digunakan pada pembuatan aplikasi ini menggunakan Java dan XML. Aplikasi yang akan dibangun termasuk ke dalam pemrograman berorientasi objek.

4. Testing

Teknik pengujian yang dilakukan terhadap aplikasi yang akan dibangun yaitu menggunakan *white box testing* untuk menguji alur atau *flow* dari aplikasi.

5. Support

Pada tahap ini melakukan upaya-upaya pengembangan terhadap aplikasi yang sedang dibuat dalam menghadapi dan mengantisipasi perkembangan maupun perubahan sistem bersangkutan terkait dengan *hardware* dan *software* yang akan digunakan. Perangkat yang digunakan untuk pemasangan aplikasi yang telah berhasil diujikan, dan disesuaikan dengan aplikasi yang nantinya dipilih.

1.6. Ruang Lingkup

Pada penelitian ini terdapat ruang lingkup batasan dari aplikasi yang akan dibangun diantaranya:

1. Pada aplikasi yang dibangun kinerja yang dinilai adalah pemasok konstruksi pada PT Binarthama Kharisma.
2. Kriteria-kriteria yang digunakan pada metode *Weighted Product* ditentukan dengan kebutuhan dan keinginan dari pihak manajemen perusahaan.
3. Mandor konstruksi dapat menentukan kriteria-kriteria beserta bobot untuk penilaian yang kemudian diserahkan datanya kepada bagian pembelian sebagai anjuran.
4. Beberapa ketentuan menu *front end* pada aplikasi yang akan dibangun yaitu:
 - a. Menu *ranking* yang memuat halaman hasil dari perhitungan metode *Weighted Product*.
 - b. Menu *profile* berisi tentang aplikasi dan informasi mengenai PT. Binarthama Kharisma.

- c. Menu *login* bagi pengguna aplikasi agar tidak terjadi penyalahgunaan data pada ponselnya.
5. Ketentuan beberapa menu pengelolaan data pada *back end* aplikasi diantaranya:
- a. Menu pemasok untuk menambahkan, mengubah, dan menghapus data pemasok konstruksi yaitu id, nama pemasok, dan keterangan.
 - b. Kriteria untuk menambahkan, mengubah, dan menghapus data kriteria yaitu id, nama kriteria, bobot, dan atribut (*cost-benefit*).
 - c. Nilai untuk menambahkan, mengubah, dan menghapus data penilaian yaitu id, nama pemasok, nama kriteria, dan nilai (0-100).
6. Aplikasi yang akan dibangun menggunakan pemrograman Android (*Java & XML*) dan *database* SQLite.
7. Versi android yang cocok untuk instalasi aplikasi yang dibangun yaitu Android OS, v5.1.1 (Lollipop).

BAB II

LANDASAN TEORI

Pada landasan teori dijelaskan tinjauan pustaka yang menguraikan teori-teori serta algoritma yang mendukung judul yang mendasari perubahan secara rinci. Landasan teori berupa definisi-definisi atau model dan algoritma yang berkaitan langsung dengan masalah yang diteliti. Pada bagian ini juga dituliskan *tools* atau *software* komponen yang digunakan untuk keperluan penelitian.

2.1. Tinjauan Jurnal

Berisikan analisa dari penulis mengenai aplikasi yang akan dibahas atau dikembangkan, kemudian diperbandingkan dengan penelitian serupa yang terdahulu yang diambil dari jurnal-jurnal yang mendukung.

Pada penelitian sistem pendukung keputusan evaluasi kinerja pramuniaga Toserba Yogya Ciwalk menggunakan Metode *Weighted Product*. Penelitian ini membahas pramuniaga yang kompeten dalam melayani konsumen sangat menentukan keberhasilan penjualan dari Toserba Yogya Ciwalk. Namun pihak manajemen kesulitan dalam melakukan evaluasi kinerja Pramuniaga menggunakan beberapa kriteria yang diinginkan. Banyak data penilaian kinerja Pramuniaga yang tercecer. Hasil penelitian menunjukkan bahwa SPK memberikan kemudahan bagi personalia dalam melakukan penilaian kinerja terhadap Pramuniaga sesuai dengan kriteria yang ditentukan. SPK berhasil dibangun

menggunakan *Web Programming*. Metode *Weighted Product* sangat tepat untuk menghitung peringkat dari kinerja Pramuniaga di Toserba Yogya Ciwalk (Mauliana *et al.*, 2018).

Pada penelitian sistem penunjang keputusan penilaian kinerja kasir Lotte Mart menggunakan metode *Weighted Product*. Penelitian ini membahas PT Lotte Mart Indonesia yang menempatkan Kasir pada tahap akhir proses belanja. Telah diidentifikasi beberapa permasalahan dalam melakukan penilaian kinerja terhadap Kasir. Pihak Personalia mengalami kendala dalam melakukan penilaian kinerja Kasir menggunakan beberapa kriteria penilaian yang diinginkan. Kemudian tahapan penilaian kinerja Kasir tidak cukup efektif. Hasil penelitian menunjukkan bahwa SPK memberikan kemudahan bagi Personalia dalam melakukan penilaian kinerja Kasir menggunakan beberapa kriteria yang diinginkan. SPK berhasil dibangun menggunakan pemrograman *Website*. Metode *Weighted Product* yang diimplementasikan sangat sesuai untuk menghitung peringkat dari kinerja Kasir di PT Lotte Mart Indonesia (Mirawati *et al.*, 2018).

Pada penelitian pemilihan *supplier* di JOB Pertamina Hulu *Energy-Jadestone Energy (Ogan Komering) Limited* menggunakan metode TOPSIS. Penelitian tersebut memaparkan banyaknya *supplier* membuat pengadaan barang dan jasa untuk lebih selektif dalam melakukan pemilihan berdasarkan kriteria yang sesuai dengan perusahaan yaitu mempertimbangkan aspek harga, kualitas, kesesuaian dokumen persyaratan, dan waktu pengiriman. Tujuan metode ini adalah melakukan pemilihan *supplier* untuk produk *Lamp Portable Explosion Proof* di perusahaan menggunakan metode TOPSIS. Hasil penelitian menunjukan dengan

dianalisa pendekatan tersebut PT Paramanusa menjadi *supplier* yang terpilih (Aprilia & Rosyidi, 2018).

Penelitian rancang bangun aplikasi sistem pendukung keputusan penentuan *supplier* obat terbaik menggunakan metode *Weighted Product* pada Rumah Sakit Ibu dan Anak Putri Surabaya. Permasalahan pada penelitian tersebut yaitu proses pembelian obat masih dilakukan dengan melihat kriteria harga termurah dan ketersediaan barang saja dari pihak pemasok. Sehingga kualitas yang didapat kurang memuaskan terhadap hasil pengembalian barang. Tujuan dari penelitian ini yaitu membangun aplikasi penentuan *supplier* obat terbaik menggunakan metode *Weighted Product* berdasarkan beberapa kriteria tambahan yang telah digunakan. Hasil dari penelitian tersebut yaitu aplikasi dapat menyediakan laporan *supplier* obat terbaik sesuai dengan kebutuhan RSIA Putri Surabaya. Laporan tersebut berguna sebagai informasi manajemen agar tidak melakukan pengembalian obat-obatan seperti yang terjadi pada periode sebelumnya (Jusak & Sudarmaningtyas, 2018).

Penelitian mengenai perancangan sistem pendukung keputusan untuk pemilihan pemasok Nata De Coco dengan metode *Weighted Product* (WP). Pada penelitian tersebut membahas pemasok nata de coco lembaran bagi perusahaan CV Agrindo Suprafood Yogyakarta. Terdapat permasalahan dikarenakan belum adanya kriteria yang secara rinci dapat dipergunakan untuk pemilihan pemasok. Penelitian ini bertujuan menemukan kriteria-kriteria yang berpengaruh dalam pemilihan pemasok. Digunakan metode *Weighted Product* (WP) untuk pengolahan data pada Sistem Pendukung Keputusan (SPK). Telah didapatkan beberapa kriteria yang mempengaruhi pemilihan pemasok. Hasil pengujian SPK diperoleh bahwa sistem

yang telah dibuat dapat dipergunakan oleh CV. Agrindo Suprafood Yogyakarta (Purnomo & Susetyo, 2016).

Pada penelitian implementasi DSS dengan metode SAW untuk menentukan prioritas pekerjaan operasi dan pemeliharaan sistem irigasi DPU Kabupaten Tegal. Permasalahan pada penelitian ini yaitu salah satu penyebab kurang terawatnya saluran irigasi di kabupaten Tegal adalah masih buruknya sistem manajemen pengembangan dan pengelolaan irigasi (PPI). Terdapat kesulitan dari DPU Kabupaten Tegal dalam penentuan prioritas pekerjaan operasi dan pemeliharaan (O&P) irigasi dengan kemungkinan adanya kesalahan proses perhitungan prioritas. Tujuan penelitian ini yaitu membuat *software* menggunakan metode *Simple Additive Weighting* (SAW). Hasil penelitian menggunakan metode SAW dapat diterapkan dalam evaluasi alternatif prioritas pekerjaan operasi dan pemeliharaan jaringan irigasi DPU Kabupaten Tegal (Faqih, 2014).

Pada penelitian Sistem Pendukung Keputusan (SPK) penerimaan beasiswa Bidik Misi di POLIBAN dengan metode SAW berbasis *web*. Penelitian tersebut memaparkan program beasiswa Bidik Misi yang merupakan program bantuan biaya pendidikan bagi calon mahasiswa tidak mampu secara ekonomi, namun memiliki potensi akademik yang baik. Tujuan penelitian ini yaitu membangun SPK untuk menentukan alternatif penerima Beasiswa Bidik Misi di Politeknik Negeri Banjarmasin yang paling mendekati kriteria yang telah ditentukan. Metode yang digunakan yaitu *Simple Additive Weighting* (SAW). Hasil perhitungan diperoleh penjumlahan dari matriks ternormalisasi dengan bobot perkriteria yang menunjukkan rangking alternatif penerima beasiswa dari yang paling mendekati kriteria hingga yang paling jauh dari kriteria (Fauzan *et al.*, 2018).

Penelitian sistem pendukung keputusan menentukan jumlah beras miskin menggunakan metode *Simple Additive Weighting* (SAW). Dalam penelitian ini digunakan metode SAW (*Simple Additive Weight*). Hasil penelitian ini yaitu aplikasi Sistem Pendukung Keputusan Menentukan Jumlah Beras Miskin Menggunakan Metode Simple Additive Weight (SAW) ini dapat membantu kepala desa dalam menentukan jumlah beras yang akan diterima oleh penerima Beras Miskin dan jumlah penerima Beras Miskin didapatkan berdasarkan nilai bobot yang telah ditentukan (Angrawati *et al.*, 2016).

Selanjutnya penelitian sistem pendukung keputusan penilaian kinerja karyawan menggunakan metode SAW (*Simple Additive Weighting*) di Universitas Muhammadiyah Purwokerto. Pada penelitian ini dibangun sebuah sistem pendukung keputusan menggunakan metode SAW (*Simple Additive Weighting*) berbasis *desktop*. Hasil dari penelitian mempermudah proses penilaian kinerja karyawan secara obyektif di LPM berdasarkan bobot dan kriteria-kriteria penilaian yang sudah ditentukan (Anto *et al.*, 2015).

Penelitian mengenai sistem informasi penjurusan dan penerimaan peserta didik baru menggunakan metode *Weighted Product*. Penulis membahas sistem informasi penjurusan dan penerimaan peserta didik baru menggunakan metode *Weighted Product* di MAN 2 Model Banjarmasin berbasis *web*. Penyeleksian berdasarkan beberapa kriteria yang ditetapkan oleh pihak sekolah MAN 2 Model Banjarmasin. Penelitian ini menghasilkan program yang berjalan dengan baik. Perhitungan data yang diproses menghasilkan perangkaan penjurusan dan menetapkan status siswa yang diterima sejak peserta mendaftarkan dirinya sebagai calon peserta didik di MAN 2 Model Banjarmasin (Fauzan *et al.*, 2017).

Perbandingan penelitian yang terkait dengan tujuan dari pembahasan pada penelitian ini dapat dilihat pada Tabel II.1.

Tabel II.1.**Penelitian Terkait**

No.	Penelitian	Metode	Hasil	Persamaan	Perbedaan
1	Seleksi Penyiar Radio Tanjung Puri Menggunakan Metode Weighted Product	Weighted Product	Dapat mempercepat kerja tim penilai dalam proses pemilihan penyiar Radio Tanjung Puri.	Menggunakan Metode WP (WEIGHTED PRODUCT)	a. Objek Penelitian Yaitu Radio Tanjung Puri b. Bahasa pemrograman berbasis web.
2	Seleksi Penerimaan Karyawan Baru Menggunakan Metode WP (WEIGHTED PRODUCT) Dengan Bahasa Pemrograman PHP dan MYSQL	Weighted Product	Untuk menentukan karyawan yang sesuai dengan kebutuhan dari kriteria perusahaan	Menggunakan Metode WP (WEIGHTED PRODUCT)	a. Objek Penelitian Yaitu Penerimaan Karyawan Baru. b. Bahasa pemrograman berbasis web.
3	Penerapan Metode Weighted Product (WP) untuk penerimaan pegawai baru di PT. TIGA SERANGKAI PUSTAKA MANDIRI SURAKARTA	Weighted Product	Seleksi Administrasi memiliki tingkat keakuratan sebesar 80% dan seleksi tes memiliki tingkat keakuratan sebesar 66.67%.	Menggunakan Metode WP (WEIGHTED PRODUCT)	a. Objek Penelitian Yaitu Di PT. TIGA SERANGKAI PUSTAKA

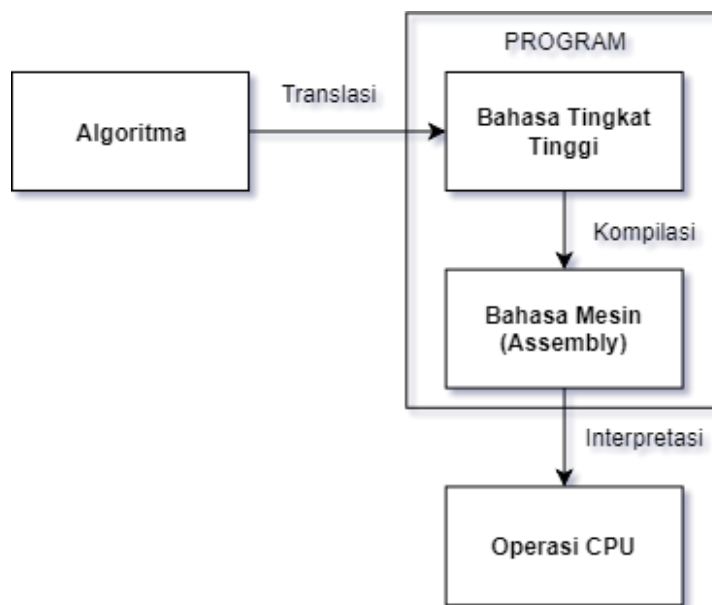
					<p>MANDIRI SURAKARTA</p> <p>b. Bahasa pemrograman berbasis web.</p>
4	<p>Sistem Seleksi Penerimaan Mahasiswa baru Menggunakan Metode WEIGHTED PRODUCT (WP)</p>	<p>Weighted Product</p>	<p>Dapat membantu proses seleksi mahasiswa baru di Universitas Nusantara PGRI Kediri Sehingga lebih mudah,cepat dan tepat.</p>	<p>Menggunakan Metode WP (WEIGHTED PRODUCT)</p>	<p>a. Objek Penelitian Yaitu di Universitas Nusantara PGRI Kediri</p> <p>b. Bahasa pemrograman berbasis web.</p>
5	<p>Seleksi Suplier Bahan Baku dengan pendekatan Multi Attribut Decision Making</p>	<p>Multi Attribut Decision Making</p>	<p>Logistik lebih mudah menentukan Suplier Bahan Baku.</p>		

2.2. Konsep Dasar Program

Berisikan mengenai definisi pengertian bahasa program yang akan digunakan dalam pembangunan atau pengembangan aplikasi yang akan dibahas dan tidak menggambarkan *tools* dalam bahasa program tersebut. Selain dari itu, dalam konsep dasar program juga dibahas tentang pemodelan pada program aplikasi yang digunakan yaitu membahas tentang pemrograman OOP.

2.2.1. Definisi Program Komputer

Pemrograman komputer adalah cara untuk memberikan instruksi algoritmik pada komputer. Jadi dalam hal ini komputer hanyalah sebagai pemroses saja yang dapat diperintah sesuai dengan algoritma yang diberikan. Cara memberikan instruksi atau algoritma ini adalah melalui pemrograman komputer dengan media bahasa pemrograman (Syaifudin *et al.*, 2018).



Sumber: (Syaifudin *et al.*, 2018)

Gambar II.1. Pemrosesan Algoritma

Urutan pemrosesan algoritma pada Gambar II.1 dapat dijelaskan menurut (Syarifudin *et al.*, 2018) sebagai berikut:

1. Algoritma dalam ide manusia ditranslasikan ke dalam bahasa tingkat tinggi menggunakan suatu *text editor* atau *software* pemrograman.
2. Instruksi dalam bentuk teks yang berisi bahasa tingkat tinggi dikompilasi oleh kompilator file berisi instruksi-instruksi bahasa mesin atau bahasa assembly.
3. Bahasa mesin atau bahasa *assembly* diinterpretasi oleh komputer atau CPU menjadi aksi-aksi dalam komputer.

2.2.2. Pemrograman OOP

Pemrograman Berorientasi Objek (PBO) atau *Object Oriented Programming* (OOP) merupakan teknik pemrograman menggunakan konsep objek. Tujuan dari OOP adalah untuk memudahkan programmer dalam pembuatan program menggunakan konsep objek yang ada dalam kehidupan sehari-hari. Jadi setiap permasalahan adalah objek itu sendiri yang merupakan gabungan dari beberapa objek yang lebih kecil (Abdulloh, 2017).

Pemrograman berorientasi objek merupakan suatu pendekatan pemrograman yang menggunakan *object* dan *class*. Saat ini konsep OOP sudah semakin berkembang. OOP bukanlah sekedar cara penulisan sintaks program yang berbeda, namun merupakan cara pandang dalam menganalisa sistem dan permasalahan pemrograman. Dalam OOP, setiap bagian dari program adalah *object* (Solichin, 2016).

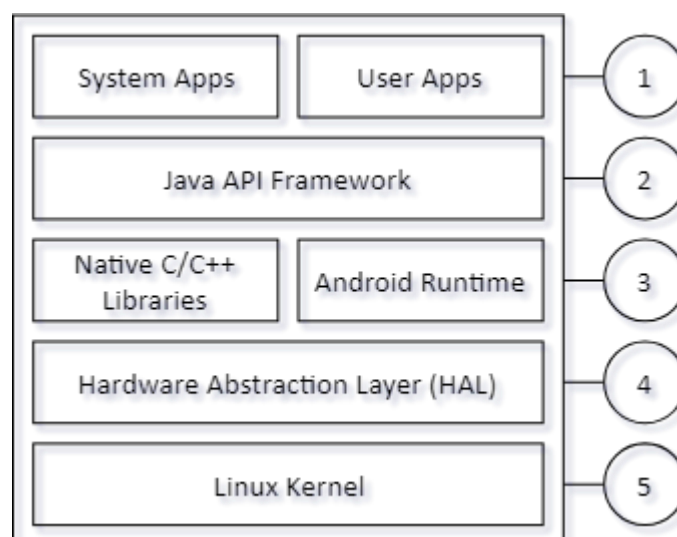
Konsep-konsep dasar pemrograman berbasis objek menurut (Enterprise, 2015) antara lain:

1. Objek (*object*), setiap objek menampung data dan kode untuk mengelola data-data tersebut. Objek dapat berinteraksi tanpa harus mengetahui informasi rinci mengenai data atau kode dalam objek lain.
2. Kelas (*class*) adalah sekumpulan objek dengan properti (atribut), perilaku (operasi), dan hubungan yang sama dengan antar objek. Sebuah objek merupakan variabel dengan tipe kelas. Kelas merupakan tipe data yang dibuat oleh pengguna. Setelah suatu kelas didefinisikan, maka dapat dibuat berapapun jumlah objek dari kelas tersebut.
3. Abstraksi (*abstraction*) data dilakukan dengan menyediakan deskripsi penting tanpa menyertakan informasi rinci atau penjelasan. Kelas-kelas menggunakan konsep abstraksi dan didefinisikan sebagai daftar atribut abstrak seperti ukuran, dan fungsi-fungsi yang mengelola atribut-atribut tersebut. Semua itu merangkum properti penting dari objek yang akan dibuat.
4. Enkapsulasi (*encapsulation*) data artinya membungkus data dan fungsi dalam sebuah unit tunggal. Dengan enkapsulasi, data tidak dapat diakses secara langsung dari luar. Untuk mengakses data, dapat menggunakan fungsi-fungsi yang ada dalam kelas tersebut.
5. Pewarisan (*inheritance*) adalah proses yang memungkinkan objek dari suatu kelas dapat memperoleh properti objek dari kelas yang lain. Pewarisan berarti suatu kelas mewarisi data dan perilaku dari kelas yang lain. Dengan pewarisan, dapat ditambahkan karakteristik pada suatu kelas tanpa harus memodifikasinya.

6. Polimorfisme (*polymorphism*) merupakan kemampuan untuk memiliki lebih dari satu bentuk. Polimorfisme berarti bahwa operasi yang sama dapat melakukan hal yang berbeda untuk tiap-tiap kelas.

2.2.3. Pemrograman *Android*

Android adalah sebuah sistem operasi untuk *smartphone* dan *tablet*. Pengguna dapat berinteraksi dengan perangkatnya dan menjalankan aplikasi-aplikasi yang tersedia pada *device* atau perangkat tersebut. *Android* adalah sistem operasi yang bersifat *open source* (sumber terbuka). *Source code* (kode sumber) dari sistem operasi *Android* dapat dilihat, diunduh, dan dimodifikasi secara bebas. Paradigma *open source* ini memudahkan pengembangan teknologi *Android*, karena semua pihak yang tertarik dapat memberikan kontribusi, baik pada pengembangan sistem operasi maupun aplikasi (Satyaputra, 2014).



Sumber: (Yudhanto & Wijayanto, 2018)

Gambar II.2. Komponen-komponen Utama *Android*

Pada Gambar II.2 dapat dijelaskan menurut mengenai komponen-komponen utama dari *Android* (Yudhanto & Wijayanto, 2018) sebagai berikut:

1. Aplikasi berada pada pada tingkat ini bersama dengan aplikasi sistem inti untuk *email*, perpesanan SMS, kalender, penjelajahan *internet*, atau kontak.
2. *Java API Framework*, semua fitur *Android* tersedia untuk *developer* melalui antarmuka pemrograman aplikasi.
 - a. Sistem tampilan digunakan untuk membangun UI aplikasi seperti daftar, tombol, dan menu.
 - b. Pengelola referensi digunakan untuk mengakses sumber daya non-kode seperti *string*, grafik, dan *file layout*.
 - c. Pengelola notifikasi digunakan untuk menampilkan peringatan khusus di bilah status.
 - d. Pengelola aktivitas yang mengelola daur hidup aplikasi.
 - e. Penyedia materi yang memungkinkan aplikasi untuk mengakses data dari aplikasi lain.
 - f. Semua API kerangka kerja yang digunakan aplikasi sistem Amdroid.
3. *Library* dan *Android Runtime*, setiap aplikasi berjalan dalam prosesnya sendiri dan dengan instance *Android Runtime* sendiri, yang memungkinkan beberapa mesin sekaligus virtual pada perangkat bermemori rendah. *Android* juga menyertakan rangkaian library waktu proses inti yang menyediakan sebagian

besar fungsionalitas bahasa pemrograman *Java*, termasuk beberapa fitur bahasa *Java 8* yang digunakan framework *Java API*. Banyak layanan dan komponen sistem *Android* inti dibangun dari kode asli yang memerlukan pustaka asli yang ditulis dalam C dan C++. Pustaka asli tersebut tersedia untuk aplikasi melalui kerangka kerja *Java API*.

4. *Hardware Abstraction Layer (HAL)*, lapisan ini menyediakan antarmuka standar yang menunjukkan kemampuan perangkat keras di perangkat ke kerangka kerja *Java API* yang lebih tinggi. HAL terdiri atas beberapa modul pustaka, masing-masing mengimplementasikan antarmuka untuk komponen perangkat keras tertentu, seperti modul kamera atau *bluetooth*.
5. *Kernel Linux* adalah fondasi dari *platform Android*. Lapisan-lapisan pada *Android* mengandalkan *kernel Linux* untuk fungsionalitas pokok seperti *threading* dan manajemen memori tingkat rendah.

A. Struktur Pemrograman *Android*

Java merupakan basis dari pemrograman *Android*. Pada *Android* disediakan *library-library* yang dibangun dengan pemrograman *Java*. Kemudian terdapat sebuah berkas XML yang berisikan kode-kode XML untuk membangun antar muka pada pemrograman *Android* (Wahana Komputer, 2012).

Sebuah aplikasi *Android* pada dasarnya terdiri dari empat komponen utama menurut (Supardi, 2015) yaitu:

1. *Activities* merupakan komponen untuk menyajikan tampilan pemakai (interface user) kepada pengguna.

2. *Service* merupakan komponen yang tidak memiliki tampilan pemakai, tetapi berjalan secara *backgrounds*.
3. *Broadcast receiver* merupakan komponen yang berfungsi menerima dan bereaksi untuk menyampaikan notifikasi.
4. *Content provider* merupakan komponen yang membuat kumpulan aplikasi dta secara spesifik sehingga dapat digunakan oleh aplikasi lain.

B. *Android Studio*

Android Studio merupakan perangkat lunak untuk para *developer Android* dalam membuat dan mengembangkan aplikasi *Android*. *Android Studio* menawarkan banyak fitur yang memungkinkan alur kerja pengembangan menjadi lebih mudah dan menyenangkan dalam satu *set* (Azis *et al.*, 2018).

Menurut (Suryana, 2018) *Android Studio* adalah Lingkungan Pengembangan Terpadu - *Integrated Development Environment* (IDE) untuk pengembangan aplikasi *Android* berdasarkan IntelliJ IDEA. Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, *Android Studio* menawarkan fitur lebih banyak untuk meningkatkan produktivitas saat membuat aplikasi *Android*, misalnya:

1. Sistem versi berbasis *Gradle* yang fleksibel.
2. *Emulator* yang cepat dan kaya fitur.
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat *Android*.

4. *Instant Run* untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru.
5. *Template* kode dan integrasi Github untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh.
6. Alat pengujian dan kerangka kerja yang ekstensif.
7. Alat *Lint* untuk meningkatkan kinerja, kegunaan, dan kompatibilitas versi.
8. Dukungan C++ dan NDK.
9. Dukungan bawaan untuk *Google Cloud Platform*, mempermudah pengintegrasian *Google Cloud Messaging* dan APP Engine.

2.3. Metode Algoritma

Berisikan mengenai definisi atau pengertian dari algoritma yang digunakan untuk menyelesaikan masalah yang dipaparkan dalam latar belakang masalah.

2.3.1. Definisi Algoritma

Definisi algoritma secara sederhana adalah urutan langkah-langkah logis yang disusun secara sistematis untuk menyelesaikan masalah (Syarifudin *et al.*, 2018) dengan algoritma yang mempunyai lima ciri penting yaitu:

1. Algoritma harus berhenti setelah mengerjakan sejumlah langkah terbatas.
Artinya algoritma harus punya garis finish dalam menjalankan langkah-langkahnya.

2. Setiap langkah harus didefinisikan dengan jelas dan tidak ambigu. pembaca maupun pemroses algoritma harus mengerti setiap perintah-perintah yang ada pada langkah-langkah algoritma.
3. Algoritma memiliki nol atau lebih masukan (*input*).
4. Algoritma mempunyai nol atau lebih keluaran (*output*).
5. Algoritma harus efektif.

2.3.2. Metode *Weighted Product*

Menurut Nofriansyah dan Defit (2017) menyatakan bahwa “metode *Weighted Product* (WP) merupakan salah satu metode yang sederhana dengan perkalian untuk menghubungkan rating atribut, dimana setiap rating atribut harus dipangkatkan dengan bobot atribut yang bersangkutan. Hal tersebut di atas dinamakan normalisasi”.

Langkah-langkah metode *Weighted Product* menurut Nofriansyah (2015) sebagai berikut:

- a. Menentukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan (C_i).
- b. Menentukan *rating* kecocokan setiap alternatif pada setiap kriteria.
- c. Memperbaiki nilai bobot terlebih dahulu dengan cara membagi bobot dengan rata-rata bobot yaitu nilai 100 adalah jumlah persentasi kebutuhan.

- d. Nilai seluruh atribut kriteria dipangkatkan dengan bobot yang telah diperbaiki.
Bagi sebuah alternatif dengan bobot pangkat positif untuk atribut manfaat (*benefit*) dan bobot pangkat negatif pada atribut biaya (*cost*).
- e. Seluruh nilai atribut kriteria dikalikan berdasarkan jumlah kriteria yang dimiliki setiap alternatif.
- f. Hasil perkalian dijumlahkan untuk menghasilkan nilai pada setiap alternatif.
- g. Mencari nilai hasil dengan melakukan pembagian dengan rata-rata dari nilai hasil setiap perkalian.
- h. Ditemukan urutan nilai hasil terbaik yang akan menjadi keputusan.

Dalam hal ini tahap awal dimulai dengan menganalisis kebutuhan-kebutuhan yang diperlukan dalam membangun suatu sistem mulai dari kebutuhan data sampai kebutuhan pembangunan aplikasi. Contoh perhitungan manual dari langkah-langkah dalam melakukan seleksi pemasok konstruksi menggunakan metode *Weighted Product* sebagai berikut:

A. Menentukan Kriteria dan Bobot

Terdapat beberapa kriteria beserta bobotnya yang ditentukan oleh bagian logistik yang dapat dilihat pada Tabel II.1.

Tabel II.2
Kriteria dan Bobot

No.	Kriteria (C)	Bobot (W)	Atribut
1	Harga	5	<i>Cost</i>
2	Kualitas	3	<i>Benefit</i>
3	<i>Stock</i>	4	<i>Benefit</i>
4	Pelayanan	5	<i>Benefit</i>

5	Pengiriman	3	Cost
---	------------	---	------

B. Menentukan *Rating* Alternatif

Contoh pemberian nilai *rating* terhadap tiga alternatif atau pemasok konstruksi ditunjukkan pada Tabel 2.

Tabel II.3
***Rating* Alternatif**

No.	Teknisi (A)	Kriteria (C)				
		C1	C2	C3	C4	C5
1	TB. Indah	70	70	80	100	80
2	TB. Sejahtera	100	80	80	80	90
3	TB. Sukasari	80	100	70	90	70

C. Perbaikan bobot

Dilakukan perbaikan bobot (W) sehingga $\sum W = 1$ yang dapat diperoleh sebagai berikut:

$$W_1 = \frac{5}{5+3+4+5+3} = \frac{5}{20} = 0,25 \quad \dots \text{Cost}$$

$$W_2 = \frac{3}{20} = 0,15 \quad \dots \text{Benefit}$$

$$W_2 = \frac{4}{20} = 0,2 \quad \dots \text{Benefit}$$

$$W_2 = \frac{5}{20} = 0,25 \quad \dots \text{Benefit}$$

$$W_2 = \frac{3}{20} = 0,15 \quad \dots \text{Cost}$$

D. Vektor S

Vektor S yang menyatakan nilai preferensi dari teknisi dapat dihitung sebagai berikut:

$$S_i = \prod_{j=1}^n x_{ij}^{w_j}$$

$$S_1 = (70^{-0,25}) (70^{0,15}) (80^{0,2}) (100^{0,25}) (80^{-0,15}) = 2,574214281$$

$$S_2 = (100^{-0,25}) (80^{0,15}) (80^{0,2}) (80^{0,25}) (90^{-0,15}) = 2,232120396$$

$$S_3 = (80^{-0,25}) (100^{0,15}) (70^{0,2}) (90^{0,25}) (70^{-0,15}) = 2,541224493$$

E. Vektor V

Nilai vektor V yang akan digunakan untuk perangkingan dapat dihitung sebagai berikut:

$$V_i = \frac{\prod_{j=1}^n x_{ij}^{w_j}}{\prod_{j=1}^n (x_j)^{w_j}}$$

$$V_1 = \frac{S_1}{S_1 + S_2 + S_3} = \frac{2,574214281}{2,574214281 + 2,232120396 + 2,541224493} \\ = \frac{2,574214281}{7,34755917} = 0,350349582$$

$$V_2 = \frac{2,232120396}{7,34755917} = 0,303790734$$

$$V_3 = \frac{2,541224493}{7,34755917} = 0,345859684$$

F. Perangkingan

Perhitungan metode *Weighted Product* menghasilkan urutan rangking yang dapat dilihat pada Tabel II.3.

Tabel II.4
Hasil Perangkingan

No.	Teknisi	V_i	Rangking
-----	---------	-------	----------

1	TB. Indah	0,323879205	1
2	TB. Sejahtera	0.347738456	3
3	TB. Sukasari	0.328382339	2

Pada Tabel 3 dapat dilihat bahwa nilai terbesar didapatkan oleh alternatif kedua, sehingga teknisi terbaik diurutkan dari TB Indah, TB Sukasari, dan diikuti oleh TB Sejahtera.

2.4. Pengujian Aplikasi

Menjelaskan secara umum mengenai langkah-langkah dalam melakukan pengujian terhadap aplikasi yang dibuat. Pengujian merupakan satu elemen dari verifikasi untuk memastikan bahwa perangkat lunak secara tepat mengimplementasikan suatu fungsi tertentu. Kemudian terdapat validasi untuk memastikan perangkat lunak dapat ditelusuri hingga ke persyaratan yang diminta oleh pelanggan. Pengujian perangkat lunak adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan spesifikasi, desain, dan pengkodean (Maturidi, 2014).

Terdapat dua pendekatan pengujian aplikasi beserta masing-masing perbedaannya yang dapat dijelaskan menurut Utami dan Asnawati (2015) sebagai berikut:

A. *Whitebox Testing*

Whitebox testing adalah pengujian yang didasarkan pada pengecekan terhadap rincian perancangan, menggunakan struktur kontrol dari desain program secara prosedural untuk membagi pengujian ke dalam beberapa kasus pengujian dari aplikasi.

Pelaksanaan atau mekanisme pengujian pada *whitebox testing* dapat dijelaskan sebagai berikut:

1. Menjamin seluruh *independent path* dieksekusi paling sedikit satu kali.
Independent path adalah jalur dalam program yang menunjukkan paling sedikit satu kumpulan proses ataupun kondisi baru.
2. Menjalani *logical decision* pada sisi *true* dan *false*.
3. Mengeksekusi pengulangan (*looping*) dalam batas-batas yang ditentukan.
4. Menguji struktur data internal.

Berdasarkan konsep pengujian bahwa *whitebox testing* yaitu memeriksa kalkulasi *internal path* untuk mengidentifikasi kesalahan.

B. *Blackbox Testing*

Pendekatan *blackbox* melakukan pengujian terhadap fungsi operasional software. Pendekatan ini biasanya dilakukan oleh penguji yang tidak ikut serta dalam pengkodean *software*. Kategori *error* yang akan diketahui melalui *blackbox testing* antara lain:

1. Fungsi yang hilang atau tidak benar.

2. *Error* dari antar muka.
3. *Error* dari struktur data atau akses eksternal basis data.
4. *Error* dari kinerja atau *behavior*.

C.Perbedaan *Black Box* dan *White Box* Testing

Berikut ini merupakan beberapa perbedaan antara *whitebox* dengan *blackbox testing*:

1. Kaidah *whitebox testing* antara lain:
 - a) Dilakukan oleh penguji yang mengetahui tentang QA.
 - b) Melakukan pengujian pada program aplikasi menyangkut keamanan dan kinerja aplikasi tersebut meliputi *code test*, desain implementasi, *security*, *data flow*, atau *software failure*.
 - c) Dilakukan seiring dengan tahapan pengembangan perangkat lunak atau pada tahap pengujian.
2. Kaidah *blackbox testing* diantaranya:
 - a) Dilakukan oleh penguji *independent*.
 - b) Melakukan pengujian berdasarkan apa yang dilihat, hanya fokus terhadap fungsionalitas dan *output*. Pengujian lebih ditujukan pada desain *software* sesuai standar dan reaksi, apabila terdapat celah-celah *bug/vulnerabilities* pada program aplikasi tersebut setelah dilakukan pengujian.
 - c) Dilakukan setelah *whitebox testing*.

2.5. Peralatan Pendukung

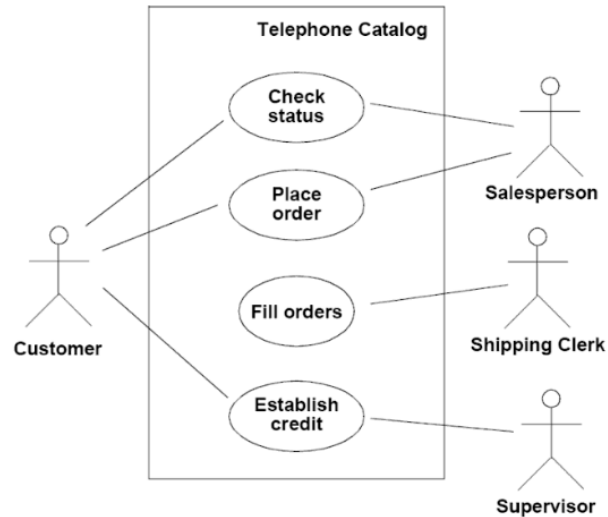
Pada bagian ini dijelaskan peralatan pendukung yang akan digunakan dalam merancang aplikasi menggunakan program berbasis OOP. Secara rinci akan dijelaskan mengenai teori UML (*Unified Modeling Language*).

2.3.3. UML

UML (*Unified Modeling Language*) memiliki banyak diagram yang digunakan untuk melakukan pemodelan data maupun sistem seperti *Use Case Diagram*, *Activity Diagram*, *Class Diagram*, dan *Sequence Diagram* yang dapat dijelaskan menurut (Mulyani, 2017) sebagai berikut:

A. *Use Case Diagram*

Use case diagram yaitu diagram yang menggambarkan *actor*, *use cases*, dan *dependencies* suatu proyek dengan tujuan untuk menjelaskan konsep hubungan antara sistem dengan dunia luar yang dibatasi dengan *boundary*. Contoh dari *use case diagram* dapat dilihat pada Gambar II.4.



Sumber: (Mulyani, 2017)

Gambar II.4. Contoh *Use Case Diagram*

Gambar II.4 contoh dari *use case diagram* dalam proses pemesanan barang. Ada beberapa *actor* yaitu *customer*, *sales person*, *shipping clerk* dan *supervisor* dengan *use case* *check status*, *place order*, *fill orders*, dan *establish credit*.

B. Activity Diagram

Activity diagram adalah diagram UML yang digunakan untuk menggambarkan alur aktivitas dari satu proses. Contoh *activity diagram* untuk memilih urutan dalam melakukannya dapat dilihat pada Gambar II.5.



Sumber: (Mulyani, 2017)

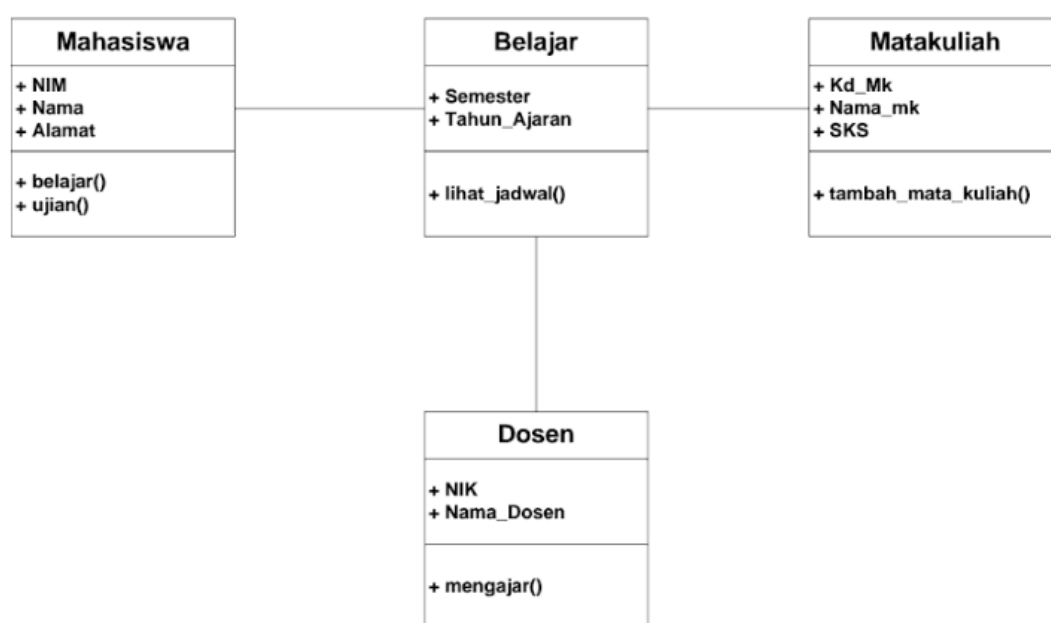
Gambar II.5. Contoh Activity Diagram

Pada Gambar II.5 menunjukkan *activity diagram* dalam proses pemesanan minuman pada *resto beverage*.

C. Class Diagram

Class diagram adalah diagram yang digunakan untuk merepresentasikan kelas, komponen-komponen kelas, dan hubungan antara masing-masing kelas.

Selain itu, *class diagram* mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantaranya. *Class diagram* juga menunjukkan *property* dan operasi sebuah kelas serta batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. UML menggunakan istilah fitur sebagai istilah umum yang meliputi *property* dan operasi sebuah kelas.



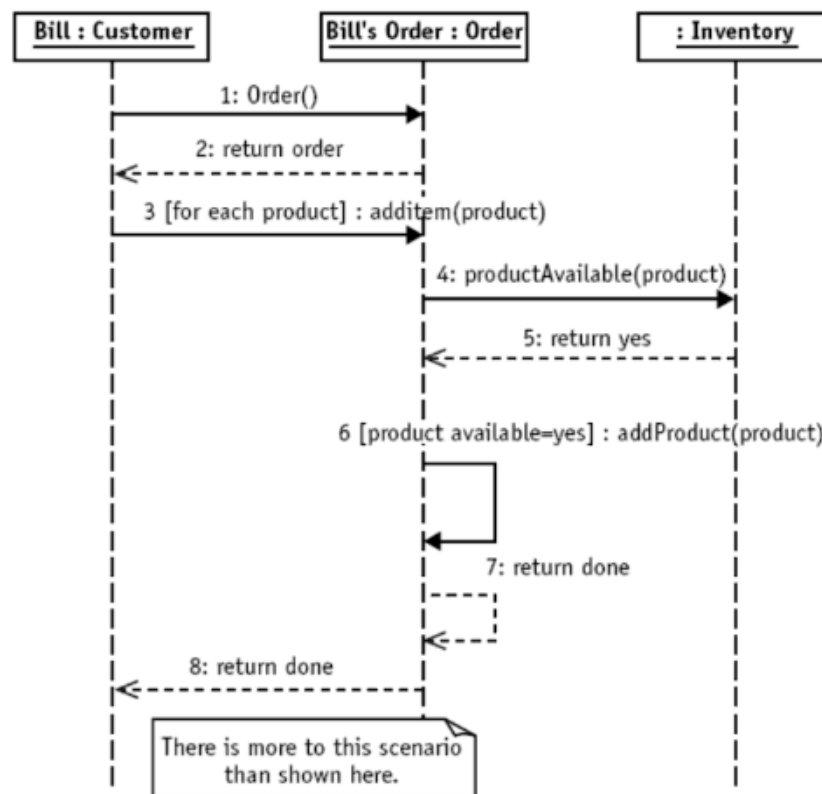
Sumber: (Mulyani, 2017)

Gambar II.6. Contoh Class Diagram

Pada Gambar II.6 merupakan contoh *class diagram* pada proses belajar suatu perkuliahan yang berisi *class* mahasiswa, belajar, mata kuliah, dan dosen.

D. Sequence Diagram

Sequence diagram adalah diagram yang menggambarkan interaksi antar objek. *Sequence diagram* secara khusus menjabarkan *behavior* sebuah skenario tunggal. Contoh *sequence diagram* ditunjukkan pada Gambar II.7.



Sumber: (Mulyani, 2017)

Gambar II.7. Contoh Sequence Diagram

Pada Gambar II.7 merupakan contoh *sequence diagram* dalam melakukan proses pemesanan produk. Urutan proses dimulai dengan pemesanan, kemudian

pengecekan produk/barang. Apabila barang tersedia di inventori, maka akan muncul penagihan pembayaran terhadap pelanggan.

2.3.4 Definisi Kriteria

- a. **Harga**
adalah suatu nilai tukar yang bisa disamakan dengan uang atau barang lain untuk manfaat yang diperoleh dari suatu barang atau jasa bagi seseorang atau kelompok pada waktu tertentu dan tempat tertentu. Istilah **harga** digunakan untuk memberikan nilai finansial pada suatu produk barang atau jasa.
- b. **Kualitas**
adalah totalitas fasilitas dan karakteristik dari produk atau jasa yang memenuhi kebutuhan, tersurat maupun tersirat.
- c. **Stock**
kegiatan penerimaan barang di gudang dan pengiriman barang untuk memastikan bahwa barang yang telah diterima atau dikirim telah sesuai dengan *purchasing order* (PO) atau *Delivery Order* (DO).
- d. **Pelayanan**
adalah pihak (individu / perusahaan) yang menjual atau memasok sumber daya dalam bentuk bahan baku kepada pihak lain (individu / perusahaan) untuk diolah menjadi barang atau jasa tertentu.
- e. **Pengiriman**
Mempersiapkan pengiriman fisik barang dari gudang ketempat tujuan yang disesuaikan dengan dokumen pemesanan dan pengiriman serta dalam kondisi yang sesuai dengan persyaratan penanganan barangnya.

BAB III

ANALISIS SISTEM BERJALAN

3.1. Tinjauan Perusahaan

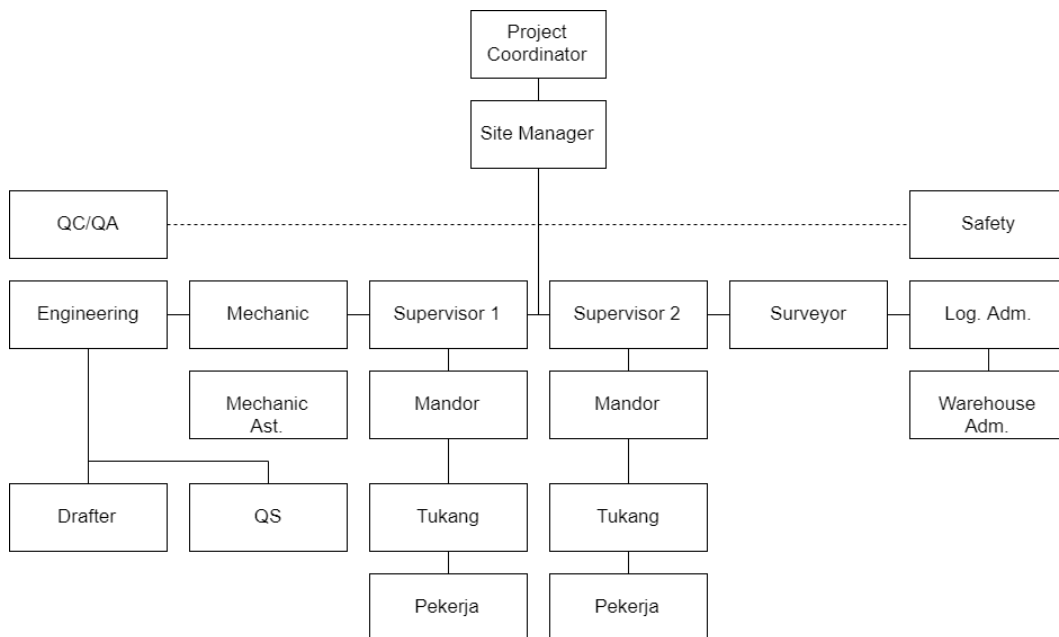
3.1.1. Sejarah Perusahaan

PT. Binarthama Kharisma *Engineering Consultant* didirikan pada tahun 1984 sebagai antisipasi terhadap kebutuhan akan jasa teknik yang makin meningkat dimasa mendatang mengikuti laju pembangunan di Indonesia sesuai dengan melajunya arus globalisasi. Jasa teknik yang sangat berperan dalam menunjang lajunya pembangunan adalah jasa dalam bidang Informatika dan Telekomunikasi, Teknik Sipil dan Geoteknik, Teknik Lingkungan dan Geomatika yang saling terkait satu sama lain. Oleh karena itu jasa-jasa tersebut dipilih sebagai obyek kegiatan kami.

Kami yakin, pengalaman konsultasi yang tajam dan mantap seiring dengan pengembangan usaha sektor lainnya akan memberikan saham yang besar artinya dalam mencapai sasaran hasil yang tepat serta nilai guna yang tinggi. PT. Binarthama Kharisma *Engineering Consultant* dalam rangka usaha serta pengabdian dirinya akan berusaha menjalankan landasan profesionalisme dalam Pembangunan Indonesia. PT. Binarthama Kharisma *Engineering Consultant* sekalipun berusia muda, akan tetapi para tenaga ahlinya telah mempunyai pengalaman Nasional maupun Internasional, selalu tetap membina dan meningkatkan segala kemampuan dan keahliannya.

3.1.2. Struktur Organisasi dan Fungsi

PT. Binarthama Kharisma *Engineering Consultant* memiliki struktur organisasi yang dibentuk untuk mempermudah dalam proyek seseorang pada bagian dan tugasnya masing-masing yang dapat dilihat pada Gambar III.1.



Sumber: PT Binarthama Kharisma *Engineering Consultant* (2019)

**Gambar III.1. Struktur Organisasi PT Binarthama Kharisma
*Engineering Consultant***

Pada Gambar III.1 merupakan struktur organisasi PT Binarthama Kharisma *Engineering Consultant* dengan deskripsi proyek yang dapat dijelaskan fungsi atau deskripsi pekerjaannya sebagai berikut:

1. *Project Coordinator*

Pemantauan rencana proyek, jadwal, jam kerja, anggaran, dan pengeluaran, pengorganisasian dan partisipasi dalam pertemuan pemangku kepentingan, dan memastikan tenggat waktu proyek terpenuhi tepat waktu.

2. *Site Manager*

- a. Merencanakan “*Time Schedule*” pelaksanaan proyek sesuai dengan kewajiban dari perusahaan terhadap pemilik proyek atau kepentingan perusahaan sendiri.
- b. Merencanakan pemakaian bahan dan alat dan pekerjaan instalasi untuk setiap proyek yang ditangani sesuai dengan volume dan waktu penggunaannya.

3. *Quality Control/Quality Assurance (QC/QA)*

- a. Membuat perencanaan kegiatan operasional *Quality Control*.
- b. Mengatur kegiatan operasional *Quality Control*.
- c. Melaksanakan kegiatan operasional *Quality Control*.
- d. Mengontrol pelaksanaan operasional *Quality Control*.

4. *Safety*

- a. Membuat perencanaan kegiatan Kesehatan dan keselamatan kerja.
- b. Mengatur kegiatan operasional Kesehatan dan keselamatan kerja.
- c. Melaksanakan kegiatan operasional Kesehatan dan keselamatan kerja.
- d. Mengontrol pelaksanaan operasional Kesehatan dan keselamatan kerja.

5. *Engineering*

Fungsi *Engineering* pada Proyek Konstruksi. *Safety Officer*, fokus pada masalah pengelolaan aspek keselamatan dan kesehatan kerja, serta pengelolaan proyek yang berwawasan lingkungan.

6. *Drafter*

Tugas dan tanggung jawab:

- a. Membuat perencanaan kegiatan operasional *drawing*.
- b. Mengatur kegiatan operasional *drawing*.
- c. Melaksanakan kegiatan operasional *drawing*.

7. *Quantity Surveyor*

Tugas dan tanggung jawab:

- a. Membuat perencanaan kegiatan operasional *Quantity Surveyor*.
- b. Mengatur kegiatan operasional *Quantity Surveyor*.
- c. Melaksanakan kegiatan operasional *Quantity Surveyor*.
- d. Mengontrol pelaksanaan operasional *Quantity Surveyor*.

8. *Mechanic*

Tugas dan tanggung jawab:

- a. Membuat perencanaan kegiatan operasional Mekanik
- b. Mengatur kegiatan operasional Mekanik

9. *Supervisor*

Tugas dan tanggung jawab:

- a. Membuat perencanaan kegiatan operasional BBS.
- b. Mengatur kegiatan operasional BBS.
- c. Melaksanakan kegiatan operasional BBS.
- d. Mengontrol pelaksanaan operasional BBS.

10. *Mandor*

- a. Membaca, memahami gambar kerja, dan menerjemahkannya ke dalam langkah-langkah operasional.
- b. Melakukan peninjauan dan pengukuran lapangan (*setting out*).
- c. Menghitung perkiraan volume pekerjaan, kebutuhan tenaga kerja, bahan dan alat.
- d. Menghitung harga satuan ongkos kerja.
- e. Merundingkan harga borongan pekerjaan.
- f. Membuat jadwal dan rencana kerja.
- g. Menyiapkan dan mengatur pembagian tugas para tukang dan pekerja.
- h. Mengawasi kegiatan para tukang dan pekerja dalam melakukan pekerjaan.
- i. Mengawasi kegiatan para tukang dan pekerja dalam melaksanakan pekerjaan.
- j. Menerapkan keselamatan dan kesehatan kerja.
- k. Mengukur dan Menghitung hasil kerja/*opname*.
- l. Melaporkan hasil kegiatan pelaksanaan pekerjaan dan menagih pembayaran.
- m. Membayar upah para tukang dan pekerja.

11. *Tukang*

Kepala tukang wajib memberikan contoh ‘bagaimana mengerjakan’ sebuah pekerjaan dengan baik dan benar, selanjutnya memastikan tukang bisa melakukan sesuai standar. Memberikan contoh bisa dilakukan juga oleh

seorang tukang senior, namun bila tidak ada tukang senior, maka menjadi tugas dari Kepala Tukang.

12. Pekerja

Memimpin dan mengendalikan pelaksanaan pekerjaan dilapangan sesuai dengan persyaratan waktu, mutu dan biaya yang telah ditetapkan. Membuat program kerja mingguan dan mengadakan pengarahan kegiatan harian kepada pelaksana pekerjaan. Mengadakan evaluasi dan membuat laporan hasil pelaksanaan pekerjaan dilapangan.

13. *Surveyor*

- a. Membantu kegiatan *survey* dan pengukuran diantaranya pengukuran topografi lapangan dan melakukan penyusunan dan penggambaran data-data lapangan.
- b. Mencatat dan mengevaluasi hasil pengukuran yang telah dilakukan sehingga dapat meminimalisir kesalahan dan melakukan tindak koreksi dan pencegahannya,
- c. Mengawasi *survey* lapangan yang dilakukan kontraktor untuk memastikan pengukuran dilaksanakan dengan akurat telah mewakili kuantitas untuk pembayaran sertifikat bulanan untuk pembayaran terakhir.
- d. Mengawasi survei lapangan yang dilakukan kontraktor untuk memastikan pengukuran dilaksanakan dengan prosedur yang benar dan menjamin data yang diperoleh akurat sesuai dengan kondisi lapangan untuk keperluan peninjauan desain atau detail desain.
- e. Mengawasi pelaksanaan *staking out*, penetapan elevasi sesuai dengan gambar rencana.
- f. Melakukan pelaksanaan survei lapangan dan penyelidikan Dan pengukuran tempat-tempat lokasi yang akan dikerjakan terutama untuk pekerjaan
- g. Melaporkan dan bertanggung jawab hasil pekerjaan ke kepala proyek

14. Logistik Adm.

- a. Mencari dan mensurvey data jumlah material beserta harga bahan dari beberapa *supplier* atau toko material bangunan sebagai data untuk memilih

harga bahan termurah dan memenuhi standar kualitas yang telah ditetapkan.

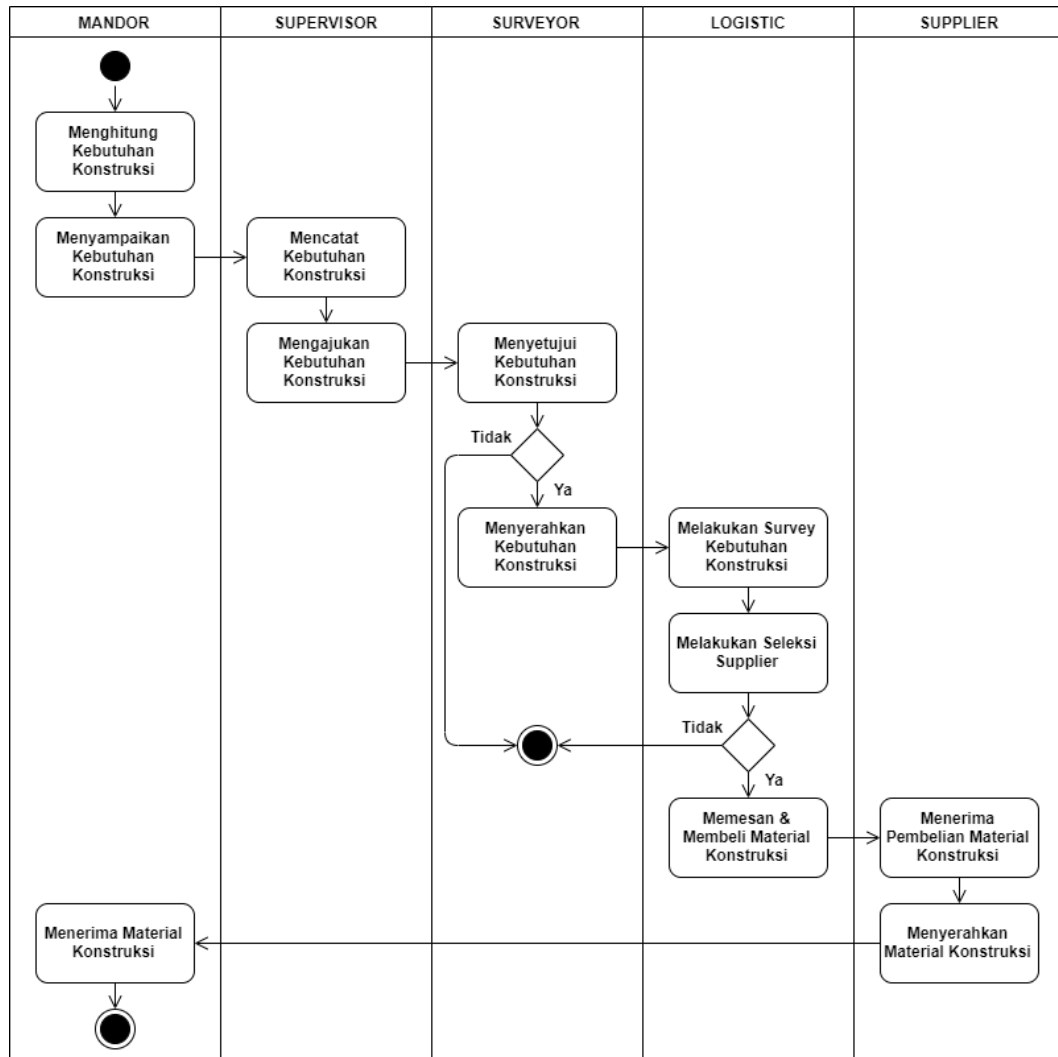
- b. Melakukan pembelian barang atau alat ke *supplier* atau toko bahan bangunan dengan melaksanakan seleksi sebelumnya sehingga bisa mendapatkan harga material termurah pada supplier terpilih.
- c. Menyediakan dan mengatur tempat penyimpanan material yang sudah didatangkan ke area proyek sehingga dapat tertata rapi dan terkontrol dengan baik jumlah pendatangan dan pemakaiannya.
- d. Membuat label keterangan pada barang yang disimpan untuk menghindari kesalahan penggunaan akibat tertukar dengan barang lain.
- e. Melakukan pencatatan keluar masuknya barang serta bertanggung jawab atas pendatangan dan ketersediaan material yang dibutuhkan dalam pelaksanaan pembangunan.

15. Gudang Adm.

- a. Membuat perencanaan pengadaan barang dan distribusinya.
- b. Mengawasi dan mengontrol operasional gudang.
- c. Mengawasi dan mengontrol semua barang yang masuk dan keluar sesuai dengan SOP.
- d. Melakukan pengecekan pada barang yang diterima sesuai SOP.

3.2. Proses Bisnis Sistem

Pada bagian ini diuraikan mengenai kumpulan aktivitas atau pekerjaan terstruktur yang saling terkait sesuai dengan SOP (*Standard Operating Procedure*) perusahaan. Proses bisnis yang sedang berjalan pada perusahaan dapat digambarkan dengan *activity diagram* pada Gambar III.2.



Gambar III.2. Activity Diagram Seleksi Pemasok

Pada Gambar III.2 menunjukkan *activity diagram* pada seleksi pemasok konstruksi untuk mencari *supplier* bahan bangunan yang cocok dengan kebutuhan proyek. Proses seleksi tersebut dimulai dari pihak mandor yang menyampaikan kebutuhan material konstruksi kepada *supervisor*. Selanjutnya *supervisor* mencatat dan mengajukan kebutuhan konstruksi kepada *surveyor*. Apabila *surveyor* tidak setuju maka prosesnya akan terhenti, namun apabila *surveyor* setuju dengan pengajuan tersebut maka catatan kebutuhan material konstruksi akan diserahkan kepada *adm logistik*. Selanjutnya *adm logistik* membeli barang ke *supplier* dengan melaksanakan seleksi sebelumnya sehingga bisa mendapatkan harga material termurah dari *supplier* terpilih.

3.3. Spesifikasi Dokumen Sistem Berjalan

Parameter-parameter dokumen yang digunakan pada tempat riset sebagai berikut:

1. Nama Dokumen : Formulir Material Konstruksi
 - Fungsi : Mencatat kebutuhan proyek
 - Sumber : Mandor & Supervisor
 - Tujuan : Surveyor & Logistic
 - Media : Kertas
 - Jumlah : 1 rangkap
 - Frekuensi : Setiap ada proyek
 - Bentuk : Lihat lampiran A.1
2. Nama Dokumen : Nota Pembelian Barang
 - Fungsi : Bukti pembelian material konstruksi
 - Sumber : *Supplier*
 - Tujuan : Logistic Adm.
 - Media : Kertas
 - Jumlah : 2 rangkap
 - Frekuensi : Setiap pembelian material konstruksi
 - Bentuk : Lihat lampiran A.2

BAB IV

RANCANGAN SISTEM DAN PROGRAM USULAN

Pada tahapan ini penulis mentranslasikan kebutuhan-kebutuhan aplikasi yang sudah dianalisis pada tahapan sebelumnya menjadi sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum pengkodean.

4.1. Analisa Kebutuhan *Software*

A. Tahapan Analisis

Pada tahapan ini perlu dirincikan spesifikasi kebutuhan dari sistem pendukung keputusan menggunakan metode *Weighted Product*.

Halaman *Frontend*:

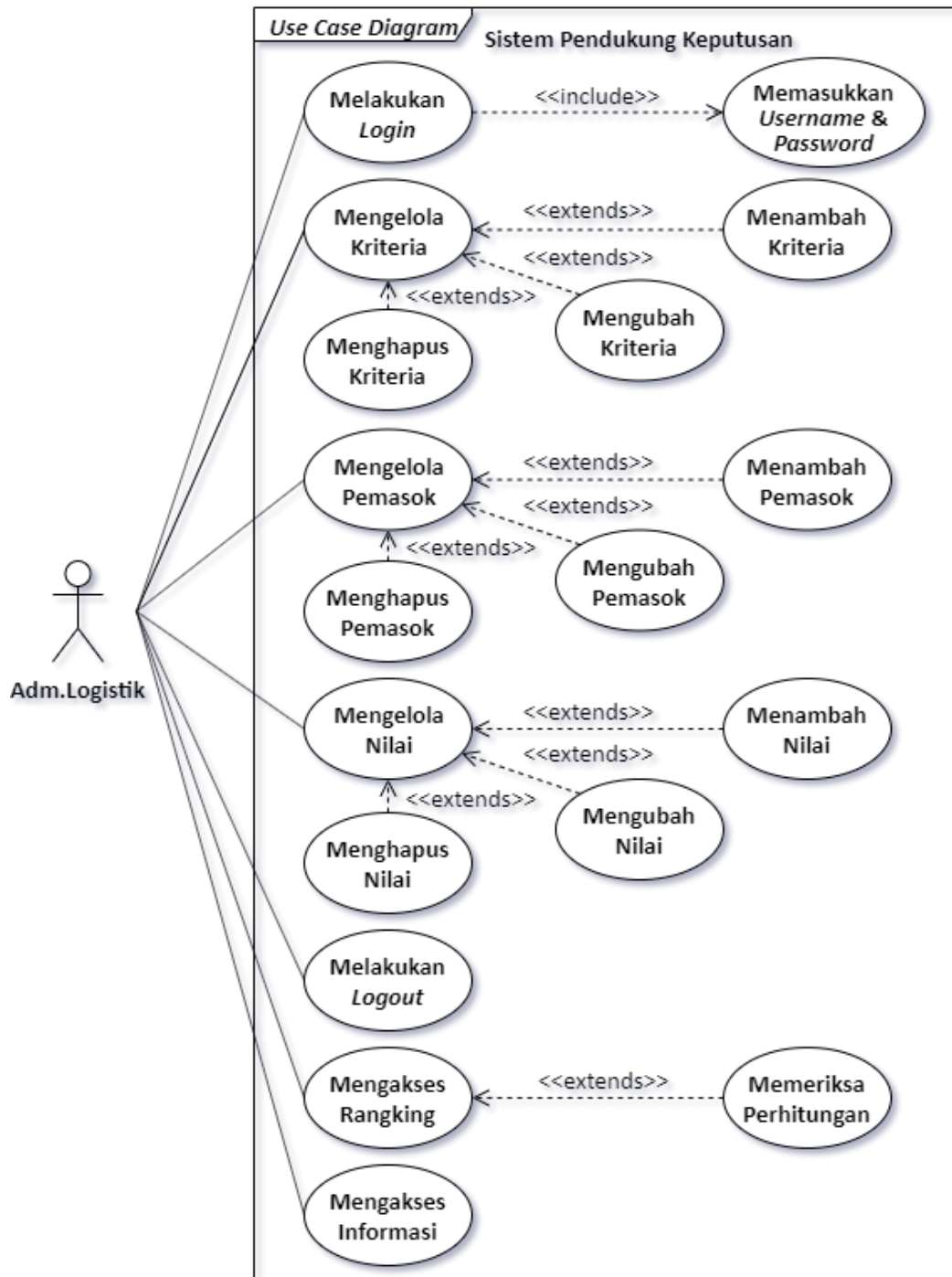
- A1. Adm. Logistik dapat melakukan *login*.
- A2. Adm. Logistik dapat memeriksa ranking dari hasil perhitungan seleksi.
- A3. Adm. Logistik dapat mengakses halaman informasi.

Halaman *Backend*:

- B1. Adm. Logistik dapat mengelola kriteria seleksi.
- B2. Adm. Logistik dapat mengelola alternatif pemasok konstruksi.
- B3. Adm. Logistik dapat mengelola nilai *rating*.
- B4. Adm. Logistik dapat melakukan *logout*.

B. *Use Case Diagram*

Use case diagram pada aplikasi yang akan dibangun digambarkan hanya yang terkait dengan proses bisnis utamanya saja yang dapat dilihat pada Gambar IV.1.



Gambar IV.1. Use Case Diagram Sistem Pendukung Keputusan

Pada gambar IV.1 merupakan *use case diagram* untuk seleksi pemasok material konstruksi menggunakan metode *Weighted Product*. Fungsi-fungsi yang terdapat pada sistem pendukung keputusan dapat dijelaskan pada deskripsi masing-masing *use case* sebagai berikut:

1. Deskripsi *Use Case* Melakukan *Login*

Kegiatan atau interaksi antara Adm. Logistik dengan aplikasi dalam melakukan *login* dapat dilihat pada Tabel IV.1.

Tabel IV.1.

Deskripsi *Use Case* Melakukan *Login*

<i>Use Case Name</i>	Melakukan <i>Login</i>
<i>Requirements</i>	A1
<i>Goal</i>	Adm. Logistik dapat melakukan <i>login</i> pada aplikasi <i>mobile</i> sistem penunjang keputusan.
<i>Pre-conditions</i>	Adm. Logistik telah menjalankan aplikasi.
<i>Post-conditions</i>	Aplikasi membuka halaman utama (<i>home page</i>).
<i>Failed end conditions</i>	Adm. Logistik salah memasukkan <i>username</i> atau <i>password</i> .
<i>Primary Actors</i>	Adm. Logistik
<i>Main Flow / Basic Path</i>	<ol style="list-style-type: none"> 1. Adm. Logistik memilih menu '<i>Login</i>'. 2. Aplikasi menampilkan halaman <i>login</i>. 3. Adm. Logistik memasukkan <i>username</i> dan <i>password</i>, kemudian menekan tombol '<i>Login</i>'. 4. Aplikasi memeriksa apakah <i>username</i> dan/atau <i>password</i> yang dimasukkan benar atau salah. Jika benar maka aplikasi akan menampilkan halaman <i>backend</i> (<i>home page</i>).

2. Deskripsi *Use Case* Mengelola Kriteria

Kegiatan atau interaksi antara Adm. Logistik dengan aplikasi dalam mengelola kriteria dapat dilihat pada Tabel IV.2.

Tabel IV.2.

Deskripsi *Use Case* Mengelola Kriteria

<i>Use Case Name</i>	Mengelola Kriteria
<i>Requirements</i>	B1
<i>Goal</i>	Adm. Logistik dapat mengelola data kriteria.
<i>Pre-conditions</i>	Adm. Logistik telah melakukan <i>login</i> .
<i>Post-conditions</i>	Aplikasi memproses pengelolaan data kriteria.
<i>Failed end conditions</i>	Adm. Logistik tidak lengkap memasukkan data kriteria atau tidak sesuai tipe datanya.
<i>Primary Actors</i>	Adm. Logistik
<i>Main Flow / Basic Path</i>	<ol style="list-style-type: none"> 1. Adm. Logistik memilih menu ‘Kriteria’. 2. Aplikasi menampilkan data kriteria yang sudah ada. 3. Adm. Logistik menekan tombol ‘Tambah’. 4. Aplikasi menampilkan formulir tambah kriteria. 5. Adm. Logistik mengisi data kriteria, kemudian menekan tombol ‘Simpan’. 6. Aplikasi menyimpan data kriteria ke dalam <i>database</i>.
<i>Invariant A:</i>	<ol style="list-style-type: none"> A3. Adm. Logistik menentukan data kriteria yang akan diubah, kemudian menekan tombol ‘Ubah’. A4. Aplikasi menampilkan formulir ubah kriteria. A5. Adm. Logistik mengedit data kriteria, kemudian menekan tombol ‘Simpan’. A6. Aplikasi menyimpan perubahan data kriteria ke dalam <i>database</i>.
<i>Invariant B:</i>	<ol style="list-style-type: none"> B3. Adm. Logistik menentukan kriteria yang akan dihapus, kemudian menekan tombol ‘Hapus’. B4. Aplikasi menampilkan konfirmasi penghapusan. B5. Adm. Logistik mengkonfirmasi penghapusan. B6. Aplikasi menghapus data kriteria dari <i>database</i>.

3. Deskripsi *Use Case* Mengelola Pemasok

Kegiatan atau interaksi antara Adm. Logistik dengan aplikasi dalam mengelola alternatif atau pemasok konstruksi dapat dilihat pada Tabel IV.3.

Tabel IV.3.

Deskripsi *Use Case* Mengelola Pemasok

<i>Use Case Name</i>	Mengelola Pemasok
<i>Requirements</i>	B2
<i>Goal</i>	Adm. Logistik dapat mengelola data pemasok konstruksi.
<i>Pre-conditions</i>	Adm. Logistik telah melakukan <i>login</i> .
<i>Post-conditions</i>	Aplikasi memproses pengelolaan data pemasok.
<i>Failed end conditions</i>	Adm. Logistik tidak lengkap memasukkan data pemasok atau tidak sesuai tipe datanya.
<i>Primary Actors</i>	Adm. Logistik
<i>Main Flow / Basic Path</i>	<ol style="list-style-type: none"> 1. Adm. Logistik memilih menu ‘Alternatif’. 2. Aplikasi menampilkan data pemasok yang tersedia. 3. Adm. Logistik menekan tombol ‘Tambah’. 4. Aplikasi menampilkan formulir tambah pemasok. 5. Adm. Logistik mengisi data pemasok, kemudian menekan tombol ‘Simpan’. 6. Aplikasi menyimpan data alternatif ke <i>database</i>.
<i>Invariant A:</i>	<ol style="list-style-type: none"> A3. Adm. Logistik menentukan data pemasok yang akan diubah, kemudian menekan tombol ‘Ubah’. A4. Aplikasi menampilkan formulir ubah pemasok. A5. Adm. Logistik mengedit data pemasok, kemudian menekan tombol ‘Simpan’. A6. Aplikasi menyimpan perubahan data pemasok ke dalam <i>database</i>.
<i>Invariant B:</i>	<ol style="list-style-type: none"> B3. Adm. Logistik menentukan pemasok yang akan dihapus, kemudian menekan tombol ‘Hapus’. B4. Aplikasi menampilkan konfirmasi penghapusan. B5. Adm. Logistik mengkonfirmasi penghapusan. B6. Aplikasi menghapus data pemasok dari <i>database</i>.

4. Deskripsi *Use Case* Mengelola Nilai

Kegiatan atau interaksi antara Adm. Logistik dengan aplikasi dalam mengelola nilai *rating* dari pemasok konstruksi yang dapat dilihat pada Tabel IV.4.

Tabel IV.4.

Deskripsi *Use Case* Mengelola Nilai

<i>Use Case Name</i>	Mengelola Nilai
<i>Requirements</i>	B3
<i>Goal</i>	Adm. Logistik dapat mengelola nilai <i>rating</i> pemasok.
<i>Pre-conditions</i>	Adm. Logistik telah melakukan <i>login</i> .
<i>Post-conditions</i>	Aplikasi memproses pengelolaan nilai <i>rating</i> .
<i>Failed end conditions</i>	Adm. Logistik tidak lengkap memasukkan nilai <i>rating</i> atau tidak sesuai tipe datanya.
<i>Primary Actors</i>	Adm. Logistik
<i>Main Flow / Basic Path</i>	<ol style="list-style-type: none"> 1. Adm. Logistik memilih menu ‘Nilai’. 2. Aplikasi menampilkan data nilai yang sudah ada. 3. Adm. Logistik menekan tombol ‘Tambah’. 4. Aplikasi menampilkan formulir tambah nilai. 5. Adm. Logistik mengisi nilai <i>rating</i>, kemudian menekan tombol ‘Simpan’. 6. Aplikasi menyimpan data nilai ke dalam <i>database</i>.
<i>Invariant A:</i>	<ol style="list-style-type: none"> A3. Adm. Logistik menentukan nilai <i>rating</i> yang akan diubah, kemudian menekan tombol ‘Ubah’. A4. Aplikasi menampilkan formulir ubah <i>rating</i>. A5. Adm. Logistik mengedit nilai <i>rating</i>, kemudian menekan tombol ‘Simpan’. A6. Aplikasi menyimpan perubahan data nilai ke dalam <i>database</i>.
<i>Invariant B:</i>	<ol style="list-style-type: none"> B3. Adm. Logistik menentukan nilai <i>rating</i> yang akan dihapus, lalu menekan tombol ‘Hapus’. B4. Aplikasi menampilkan konfirmasi penghapusan. B5. Adm. Logistik mengkonfirmasi penghapusan. B6. Aplikasi menghapus data nilai dari <i>database</i>.

5. Deskripsi *Use Case* Melakukan *Logout*

Kegiatan atau interaksi antara Adm. Logistik dengan aplikasi dalam melakukan proses *logout* atau keluar dari menu *admin* dapat dilihat pada Tabel IV.5.

Tabel IV.5.

Deskripsi *Use Case* Melakukan *Logout*

<i>Use Case Name</i>	Melakukan <i>Logout</i>
<i>Requirements</i>	B4
<i>Goal</i>	Adm. Logistik dapat keluar dari hak akses admin.
<i>Pre-conditions</i>	Adm. Logistik telah selesai melakukan seleksi.
<i>Post-conditions</i>	Aplikasi mengeluarkan hak akses admin.
<i>Failed end conditions</i>	-
<i>Primary Actors</i>	Adm. Logistik
<i>Main Flow / Basic Path</i>	1. Adm. Logistik memilih menu ' <i>Logout</i> '. 2. Aplikasi mengeluarkan hak akses admin.

6. Deskripsi *Use Case* Mengakses Rangking

Kegiatan atau interaksi antara Adm. Logistik dengan aplikasi dalam mengakses menu rangking dapat dilihat pada Tabel IV.6.

Tabel IV.6.

Deskripsi *Use Case* Mengakses Rangking

<i>Use Case Name</i>	Mengakses Rangking
<i>Requirements</i>	A2
<i>Goal</i>	Adm. Logistik dapat melihat hasil perangkingan.
<i>Pre-conditions</i>	Bobot kriteria dan nilai dari pemasok konstruksi telah ditentukan oleh pemegang keputusan.
<i>Post-conditions</i>	Aplikasi menampilkan hasil perangkingan seleksi pemasok konstruksi beserta rincian perhitungan metode <i>Weighted Product</i> .
<i>Failed end conditions</i>	Data alternatif pemasok konstruksi dan kriteria penilaian masih kosong.
<i>Primary Actors</i>	Adm. Logistik
<i>Main Flow / Basic Path</i>	<ol style="list-style-type: none"> 1. Adm. Logistik memilih menu ‘Rangking’. 2. Aplikasi menampilkan hasil perangkingan. 3. Adm. Logistik menekan tombol ‘Perhitungan’. 4. Aplikasi menampilkan rincian/<i>detail</i> perhitungan metode <i>Weighted Product</i>.

7. Deskripsi *Use Case* Mengakses Informasi

Kegiatan atau interaksi antara Adm. Logistik dengan aplikasi dalam mengakses menu informasi dapat dilihat pada Tabel IV.7.

Tabel IV.7.

Deskripsi *Use Case* Mengakses Informasi

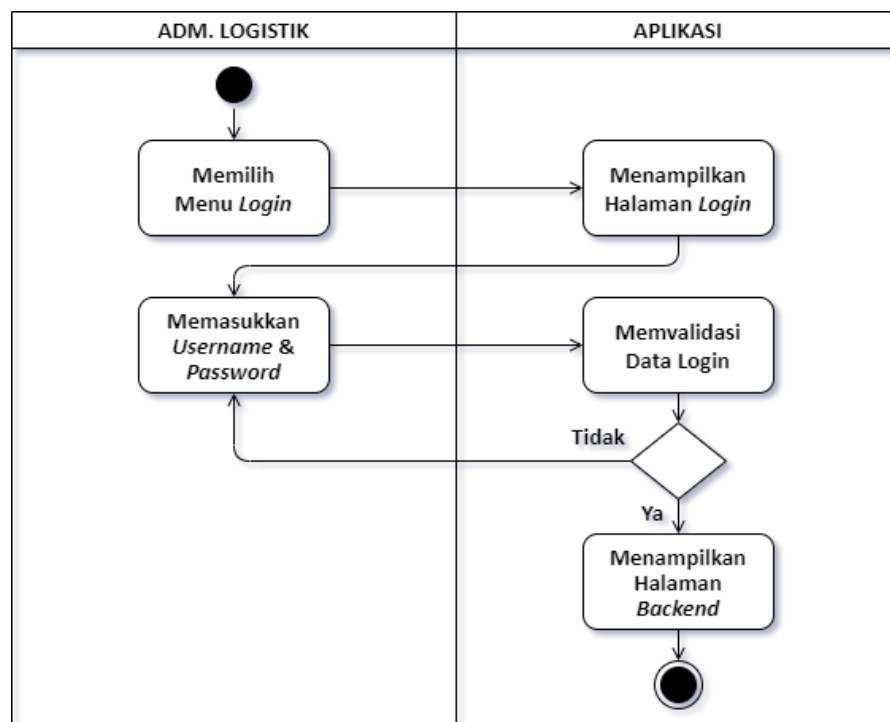
<i>Use Case Name</i>	Mengakses Informasi
<i>Requirements</i>	A3
<i>Goal</i>	Adm. Logistik dapat melihat deskripsi aplikasi.
<i>Pre-conditions</i>	Adm. Logistik menjalankan aplikasi.
<i>Post-conditions</i>	Aplikasi menampilkan deskripsi mengenai aplikasi seleksi pemasok konstruksi menggunakan metode <i>Weigted Product</i> pada PT Binarthama Kharisma.
<i>Failed end conditions</i>	Data alternatif dan kriteria masih kosong.
<i>Primary Actors</i>	Adm. Logistik
<i>Main Flow / Basic Path</i>	<ol style="list-style-type: none"> 1. Adm. Logistik memilih menu ‘Rangking’. 2. Aplikasi menampilkan hasil perbandingan. 3. Adm. Logistik menekan tombol ‘Perhitungan’. 4. Aplikasi menampilkan rincian/<i>detail</i> perhitungan metode <i>Weigted Product</i>.

C. Activity Diagram

Penggambaran *activity diagram* menggunakan partisi berdasarkan *actor* yang melakukan interaksi dengan aplikasi.

1. Activity Diagram Melakukan Login

Pada aplikasi seleksi pemasok konstruksi hanya Adm. Logistik yang dapat melakukan *login* yang dapat dilihat pada Gambar IV.2.

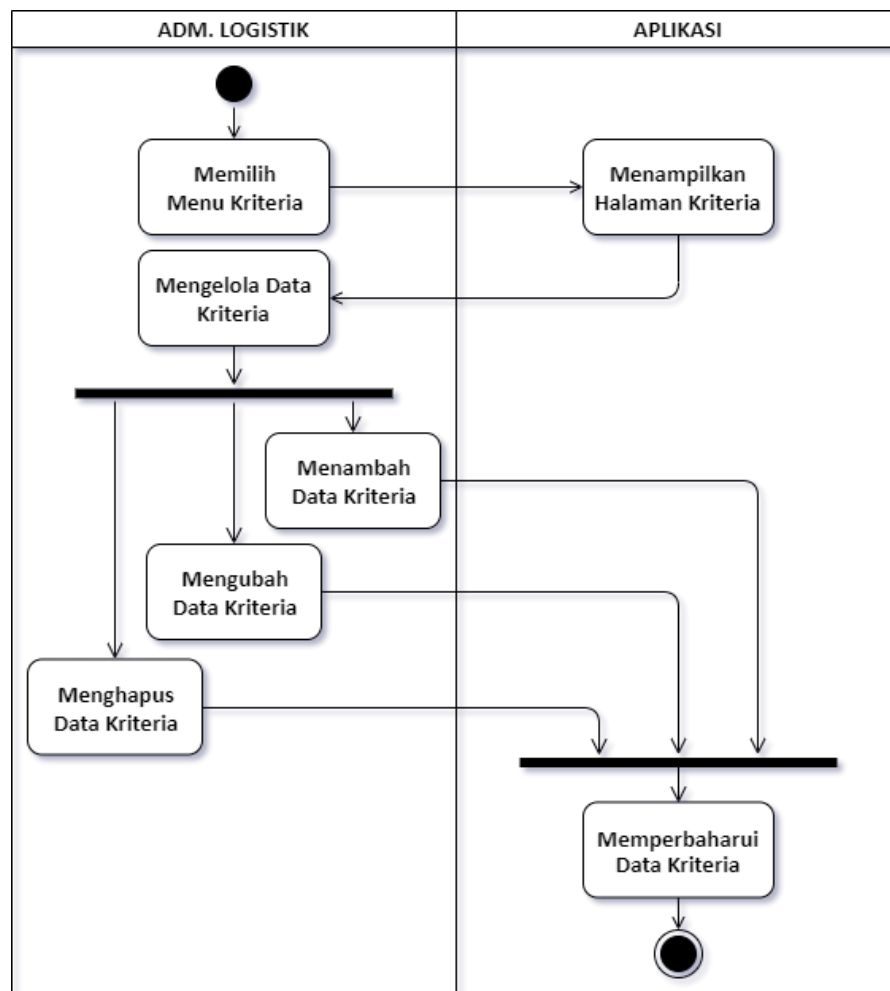


Gambar IV.2. Activity Diagram Melakukan Login

Pada Gambar IV.2 dapat dilihat ketika Adm. Logistik memilih menu *login* maka aplikasi akan menampilkan halaman *login*. Kemudian Adm. Logistik memasukkan *username* dan *password* yang divalidasi oleh sistem. Apabila *username* atau *password* salah maka Adm. Logistik harus mengulangi kembali memasukkan data *login* yang benar agar dapat melakukan pengolahan data.

2. Activity Diagram Mengelola Kriteria

Beberapa kriteria yang sudah ditentukan oleh *Surveyor* dapat ditambah, diubah, maupun dihapus oleh Adm. Logistik yang dapat dilihat pada Gambar IV.3.



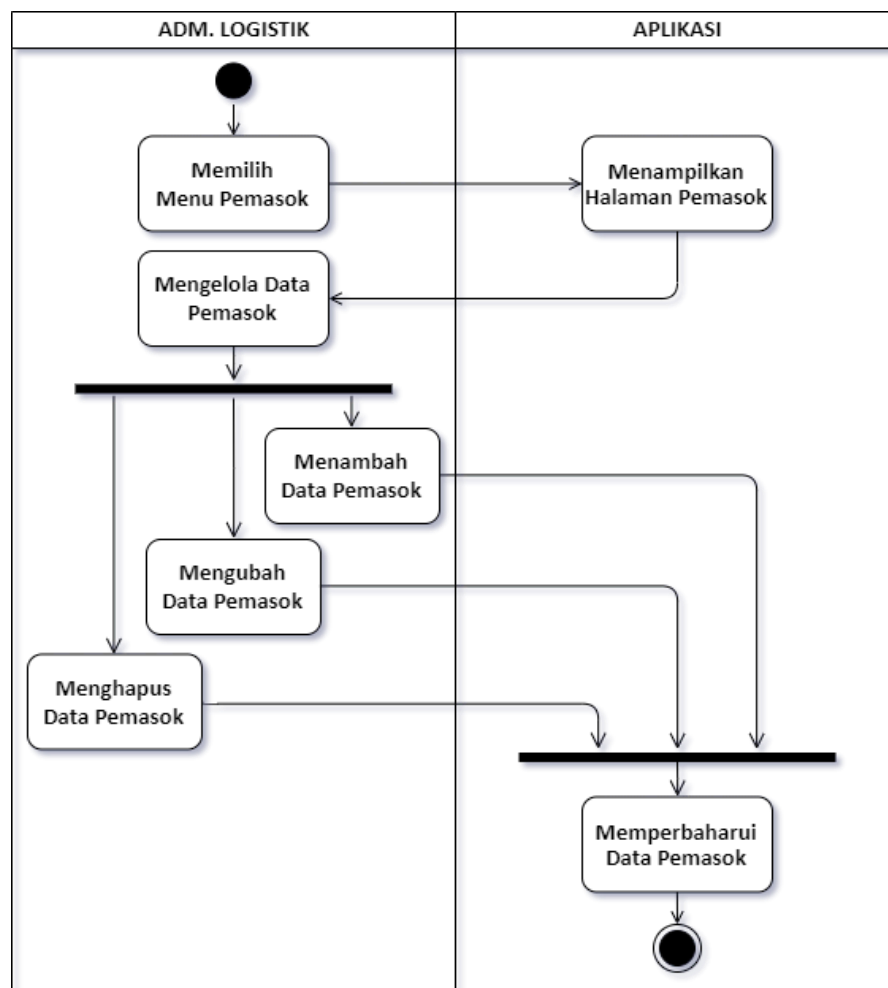
Gambar IV.3. Activity Diagram Mengelola Kriteria

Pada Gambar IV.3 dapat dijelaskan mengenai menu kriteria yang dipilih oleh Adm. Logistik yang akan menampilkan halaman kriteria pada aplikasi. Kemudian Adm. Logistik dapat mengelola data kriteria seperti id, nama, bobot,

dan atribut (*cost-benefit*) untuk diperbaharui melalui aplikasi ke dalam *database*.

3. Activity Diagram Mengelola Pemasok

Pengelolaan alternatif pemasok konstruksi yang dapat ditambah, diubah, maupun dihapus oleh Adm. Logistik yang dapat dilihat pada Gambar IV.4.



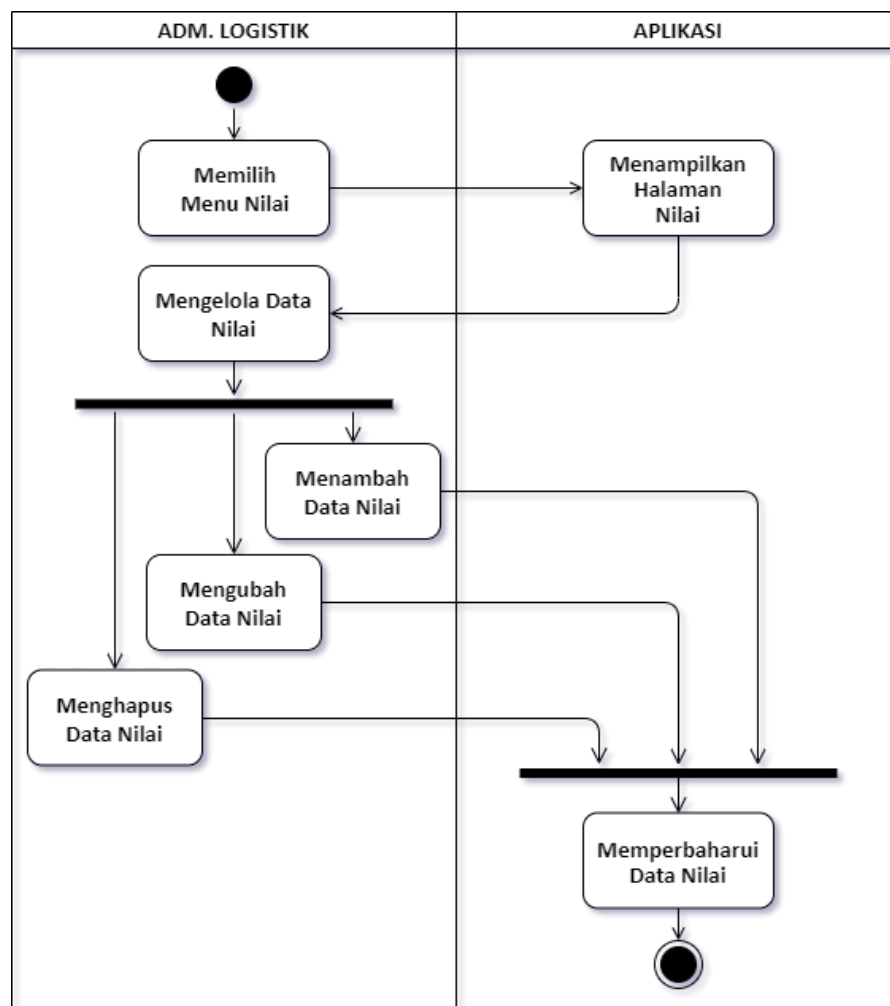
Gambar IV.4. Activity Diagram Mengelola Pemasok

Pada Gambar IV.4 dapat dijelaskan mengenai menu pemasok yang dipilih oleh Adm. Logistik yang akan menampilkan halaman dari data pemasok konstruksi.

Kemudian Adm. Logistik dapat mengelola data pemasok konstruksi untuk diperbaharui melalui aplikasi ke dalam *database*.

4. Activity Diagram Mengelola Nilai

Nilai *rating* pemasok konstruksi yang diberikan oleh Adm. Logistik dengan *activity diagram* yang dapat dilihat pada Gambar IV.5.



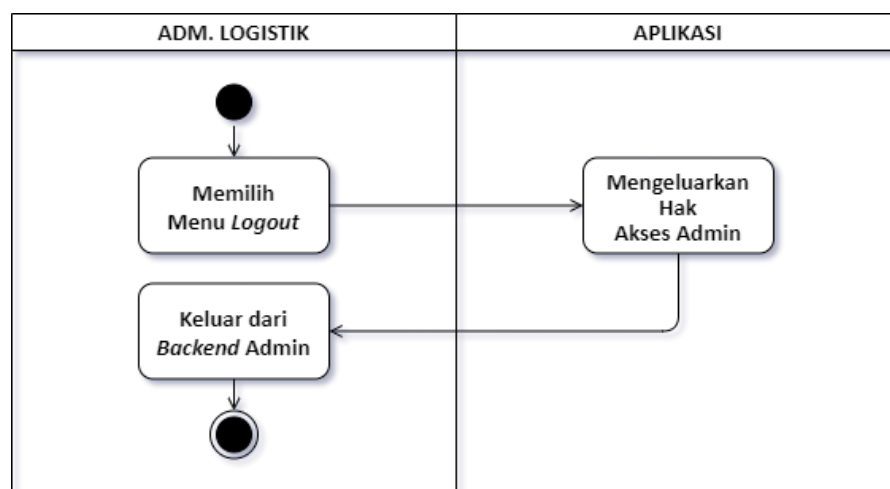
Gambar IV.5. Activity Diagram Mengelola Nilai

Pada Gambar IV.5 merupakan alur menu kriteria yang dipilih oleh Adm. Logistik yang akan menampilkan halaman *rating* pada aplikasi. Kemudian

Adm. Logistik dapat mengelola id, memilih alternatif, memilih kriteria, dan menentukan nilai untuk diperbaharui melalui aplikasi ke dalam *database*.

5. Activity Diagram Melakukan Logout

Kegiatan aktivitas Adm. Logistik dalam melakukan proses *logout* atau keluar dari *backend* pengelolaan data aplikasi seleksi pemasok konstruksi. Pada kegiatan tersebut dapat digambarkan dengan *activity diagram* proses *logout* yang dapat dilihat pada Gambar IV.6.

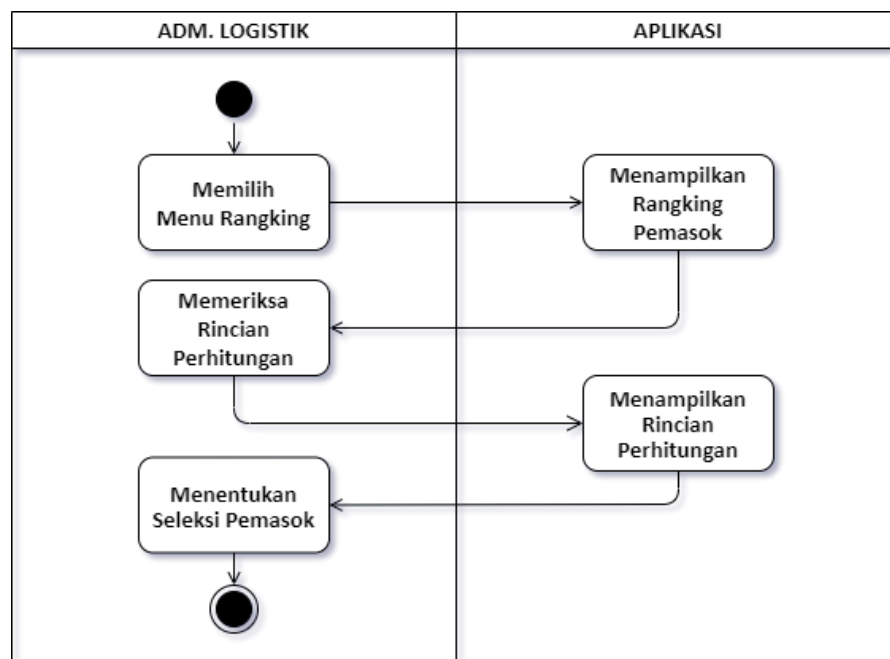


Gambar IV.6. Activity Diagram Melakukan Logout

Pada Gambar IV.6 merupakan alur yang dapat dilakukan oleh Adm. Logistik untuk melakukan *logout* aplikasi seleksi pemasok konstruksi. Adm. Logistik dapat memilih menu *logout* yang kemudian aplikasi akan mengeluarkan hak akses admin dari *backend* pengelolaan data. Pada proses tersebut aplikasi akan kembali menampilkan menu utama *frontend* dari aplikasi.

6. Activity Diagram Mengakses Rangking

Pada menu rangking berisi hasil dari perhitungan metode *Weighted Product* yang mengurutkan alternatif atau pemasok konstruksi berdasarkan nilai terbaik. Alur dari proses mengakses menu rangking yang dapat dilihat pada Gambar IV.7.



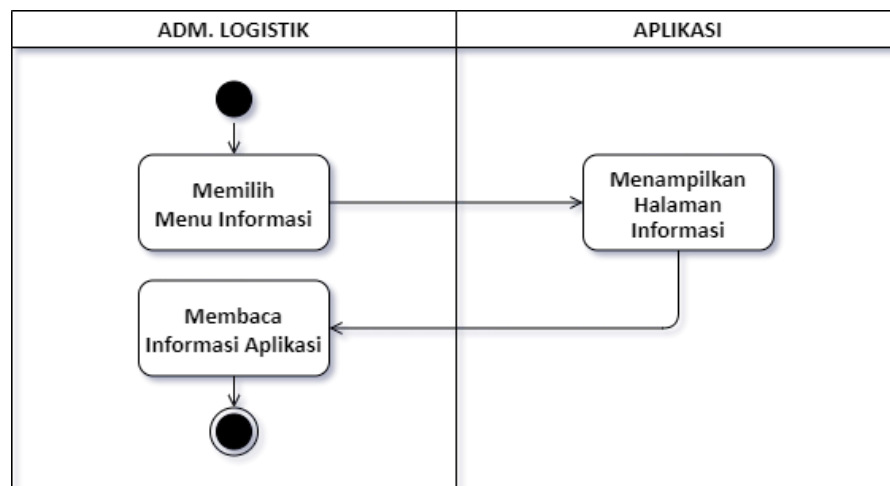
Gambar IV.7. Activity Diagram Mengakses Rangking

Pada Gambar IV.7 dapat dijelaskan bahwa menu rangking dapat diakses oleh Adm. Logistik selaku pelaksana dalam melakukan proses kerjasama dengan pemasok konstruksi. Ketika menu rangking dipilih atau diakses oleh Adm. Logistik maka sistem akan menampilkan urutan pemasok konstruksi dari yang nilainya tertinggi hingga terendah. Kemudian Adm. Logistik pun dapat melihat atau memeriksa rincian proses perhitungan yang dilakukan oleh aplikasi. Selain

itu, ditampilkan juga angka dari bobot kriteria dan nilai terhadap pemasok konstruksi yang telah ditentukan sebelumnya oleh Adm. Logistik.

7. *Activity Diagram* Mengakses Informasi

Menu informasi berisi keterangan dari aplikasi seleksi pemasok konstruksi beserta deskripsi mengenai perusahaan. *Activity diagram* pada kegiatan mengakses menu informasi yang dapat dilihat pada Gambar IV.8.



Gambar IV.8. *Activity Diagram* Mengakses Informasi

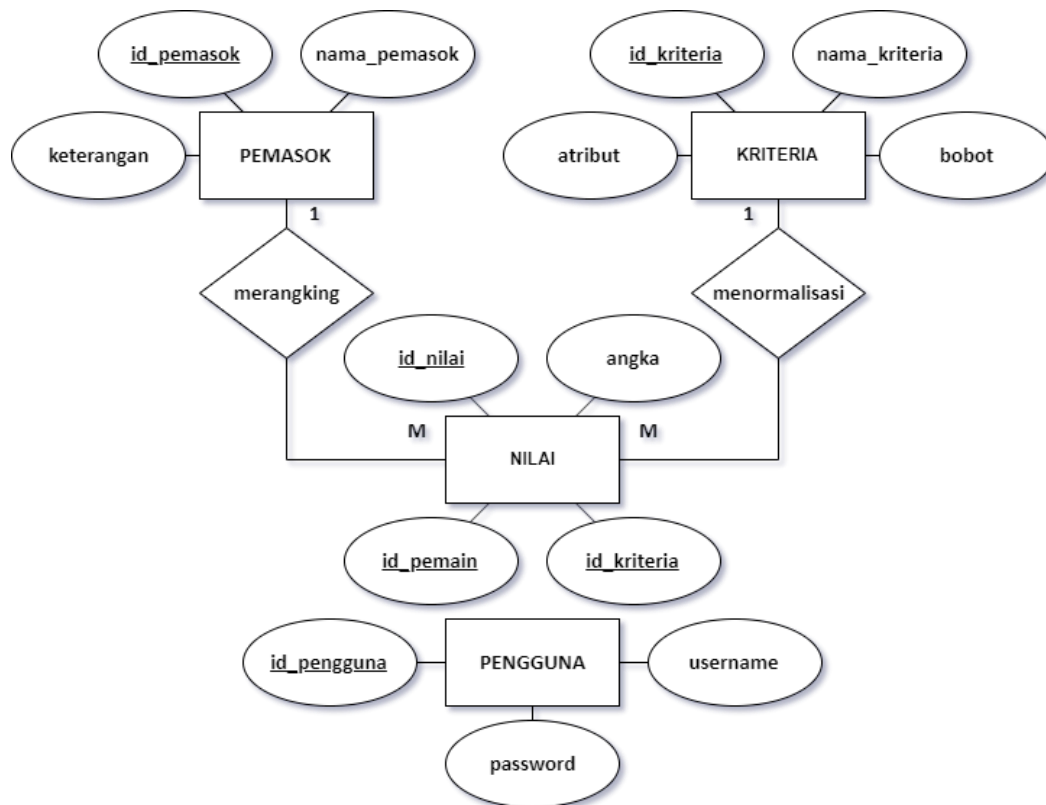
Pada Gambar IV.8 merupakan alur yang dapat dilakukan oleh Adm. Logistik untuk mengakses menu informasi. Adm. Logistik dapat memilih menu informasi yang kemudian aplikasi menampilkan halaman informasi. Halaman yang ditampilkan tersebut berisi keterangan aplikasi seleksi pemasok konstruksi menggunakan metode *Weighted Product* berupa teks dan gambaran atau deksripsi mengenai PT Binarthama Kharisma.

4.2. Desain

Pada tahapan ini menjelaskan tentang desain *database*, *software architecture*, serta *interface* dari sistem yang dibuat.

4.2.1. Database

Pada perancangan *database* menggambarkan tabel-tabel beserta relasinya menggunakan *Entity Relationship Diagram* (ERD).



Gambar IV.9. ERD Aplikasi Seleksi Pemasok

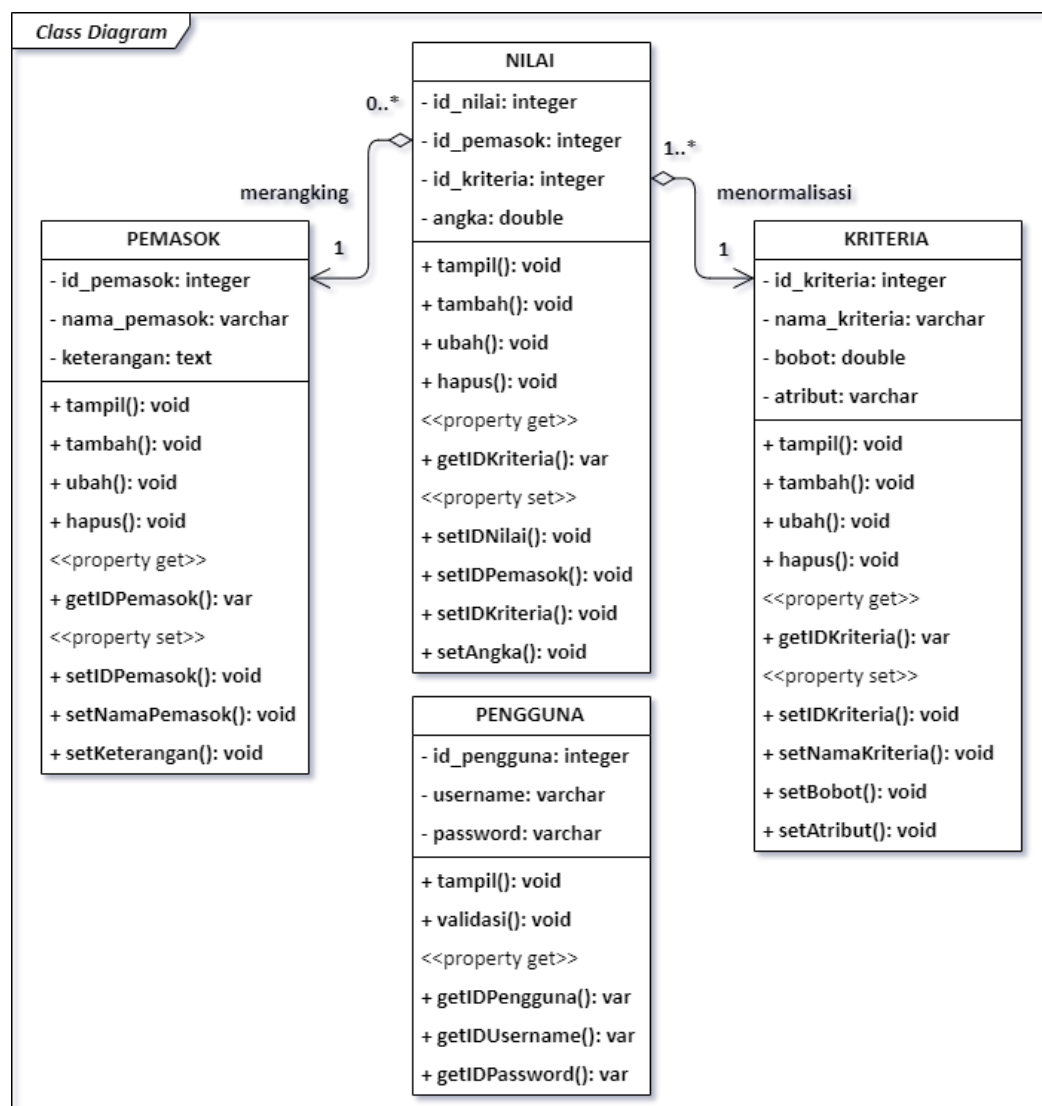
Pada Gambar IV.9 menunjukkan ERD yang merupakan rancangan *database* dari aplikasi untuk sistem pendukung keputusan yang akan dibangun.

Pada rancangan tersebut terdapat tabel pemasok, kriteria, dan nilai yang saling terkait. Namun untuk tabel pengguna tidak memiliki relasi dengan tabel lainnya.

4.2.2. Software Architecture

A. Class Diagram

Diagram struktur statis yang menggambarkan struktur aplikasi seleksi pemasok dapat dilihat menggunakan *class diagram* pada Gambar IV.10.



Gambar IV.10. Class Diagram Aplikasi Seleksi pemasok konstruksi

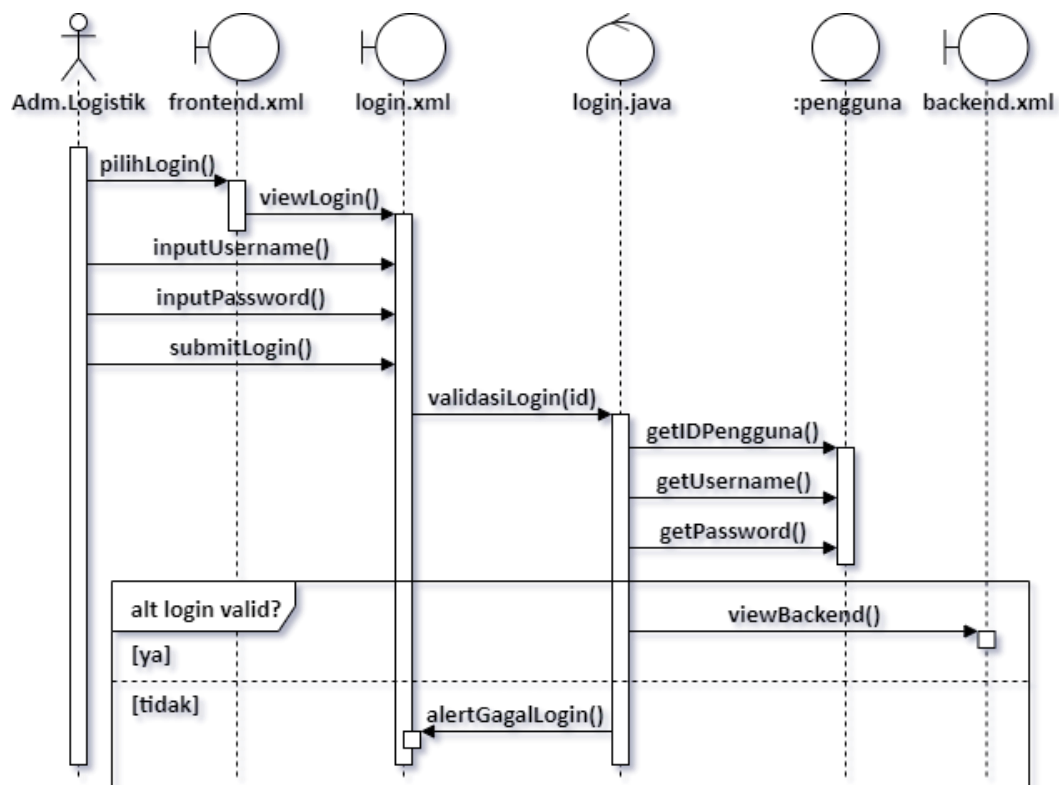
Pada Gambar IV.10 merupakan *class diagram* dari aplikasi seleksi pemasok konstruksi yang terdiri dari kelas alternatif, kriteria, nilai, dan pengguna.

B. Sequence Diagram

Penggambaran *sequence diagram* aplikasi seleksi pemasok konstruksi berdasarkan *fish level* dari *use case diagram* sebagai berikut:

1. Sequence Diagram Melakukan Login

Skenario dari proses *login* yang dilakukan oleh Adm. Logistik dapat dilihat urutannya menggunakan *sequence diagram* pada Gambar IV.11.



Gambar IV.11. Sequence Diagram Melakukan Login

Pada Gambar IV.11 menunjukkan interaksi Adm. Logistik melakukan *login* dengan memasukkan *username* dan *password*. Kemudian divalidasi

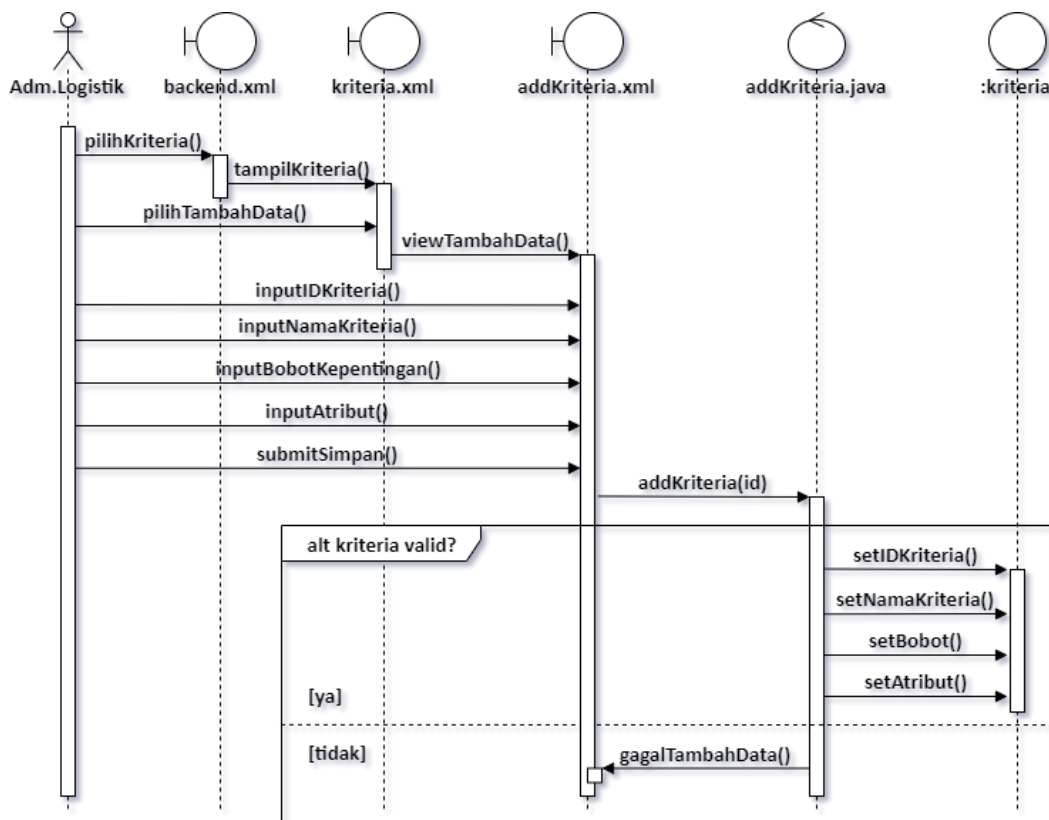
menggunakan *controller login* pada tabel pengguna. Jika *username* atau *password* benar maka akan dialihkan pada halaman *backend*.

2. Sequence Diagram Mengelola Kriteria

Pada *sequence diagram* mengelola kriteria akan dibagi menjadi beberapa diagram berdasarkan perluasan atau percabangan aktivitasnya yang dapat diuraikan antara lain:

a) Sequence Diagram Menambah Kriteria

Skenario menambah data kriteria dapat dilihat urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.12.

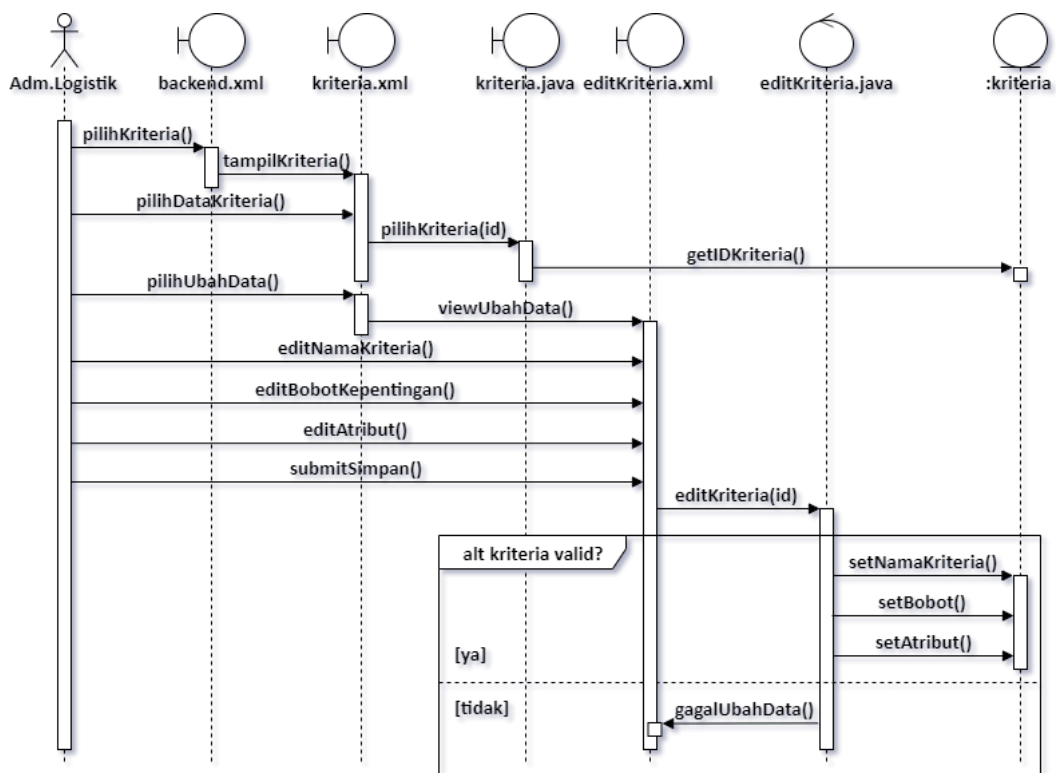


Gambar IV.12. Sequence Diagram Menambah Kriteria

Pada Gambar IV.12 dapat dijelaskan bahwa Adm. Logistik dapat menambah data kriteria yang meliputi id, nama, bobot, dan atribut untuk disimpan melalui aplikasi ke dalam *database*.

b) *Sequence Diagram* Mengubah Kriteria

Skenario mengubah data kriteria dapat dilihat urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.13.



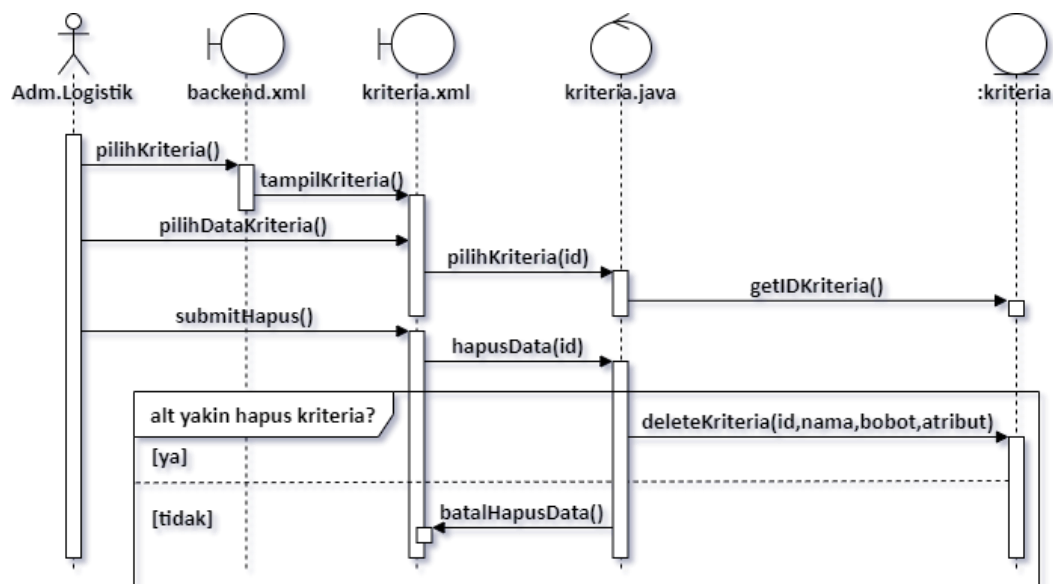
Gambar IV.13. *Sequence Diagram* Mengubah Kriteria

Pada Gambar IV.13 dapat dijelaskan mengenai skenario dalam mengubah data kriteria yang dilakukan oleh Adm. Logistik. Ketika Adm. Logistik memilih menu kriteria pada halaman *backend* maka aplikasi akan menampilkan halaman yang berisi data kriteria. Kemudian Adm. Logistik

memilih data kriteria yang akan diubah yang meliputi nama kriteria, bobot, dan atribut untuk diperbaharui melalui aplikasi ke dalam basis data.

c) *Sequence Diagram Menghapus Kriteria*

Skenario menghapus data kriteria dapat dilihat urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.14.



Gambar IV.14. *Sequence Diagram Menghapus Kriteria*

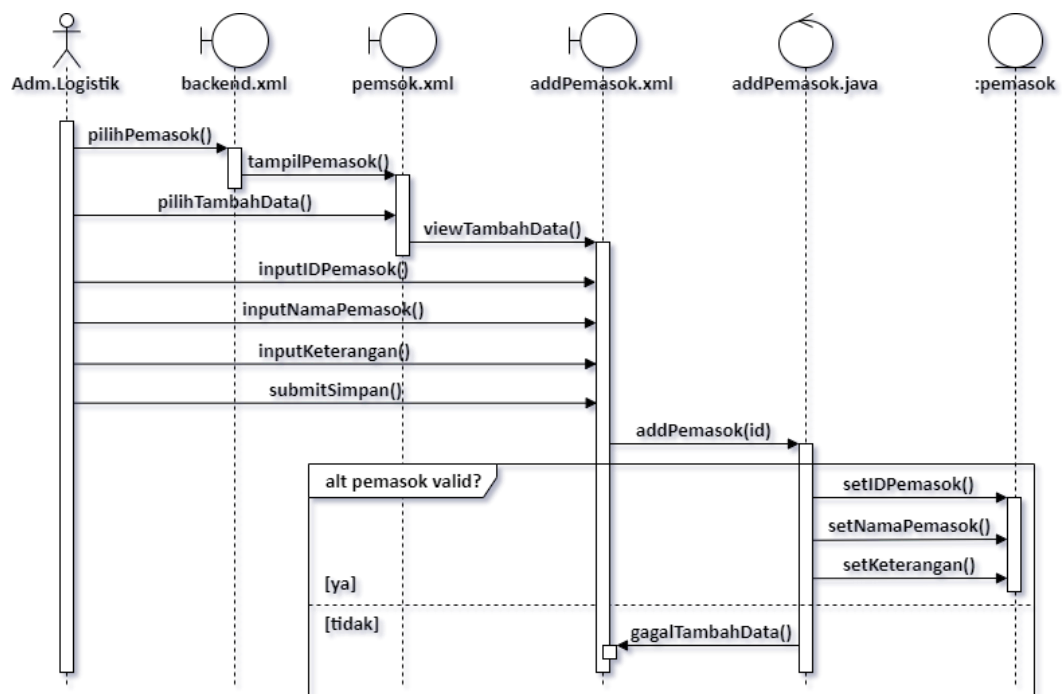
Pada Gambar IV.14 dapat dijelaskan mengenai skenario dalam menghapus salah satu data kriteria yang dilakukan oleh Adm. Logistik. Ketika Adm. Logistik memilih menu kriteria pada halaman *backend* maka aplikasi akan menampilkan halaman yang berisi data kriteria. Kemudian Adm. Logistik memilih data kriteria yang akan dihapus untuk diperbaharui melalui aplikasi ke dalam *database*.

3. Sequence Diagram Mengelola Pemasok

Pada *sequence diagram* mengelola alternatif pemasok angkulung akan dipecah menjadi beberapa diagram berdasarkan percabangan aktivitasnya antara lain:

a) Sequence Diagram Menambah Pemasok

Skenario menambah alternatif berupa data pemasok konstruksi yang dilakukan oleh Adm. Logistik dapat dilihat urutannya menggunakan *sequence diagram* pada Gambar IV.15.

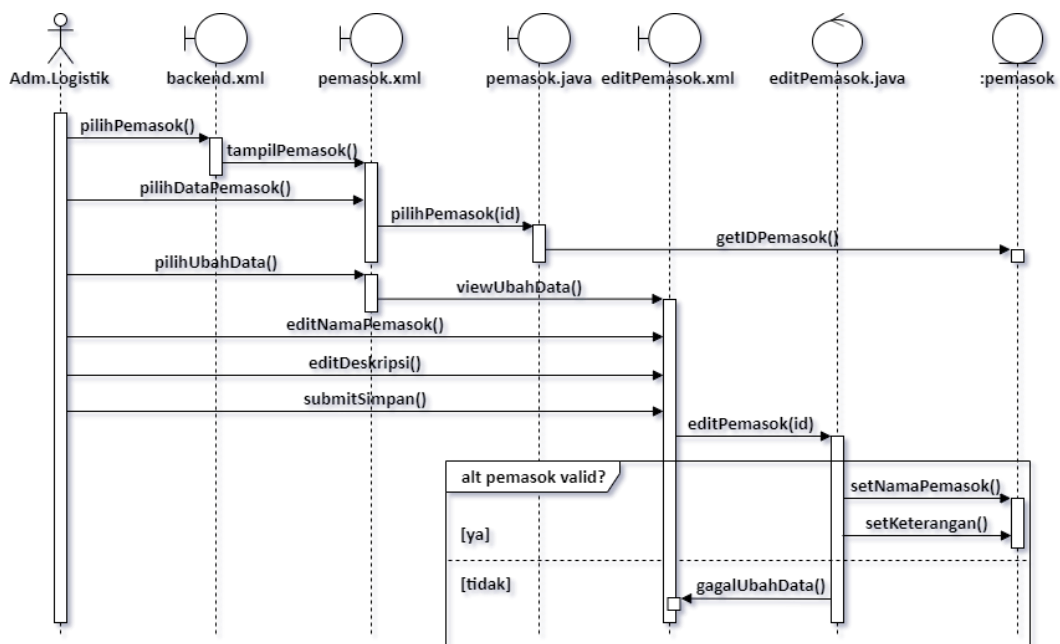


Gambar IV.15. Sequence Diagram Menambah Pemasok

Pada Gambar IV.15 menunjukkan skenario dari *sequence diagram* dalam menambah data alternatif pemasok konstruksi oleh Adm. Logistik. Data alternatif pemasok yang ditambahkan meliputi id, nama, dan deskripsi dari pemasok konstruksi untuk disimpan melalui aplikasi ke dalam *database*.

b) *Sequence Diagram* Mengubah Pemasok

Skenario mengubah alternatif atau data pemasok konstruksi yang diinisiasi oleh Adm. Logistik yang berinteraksi dengan aplikasi dapat dilihat urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.16.



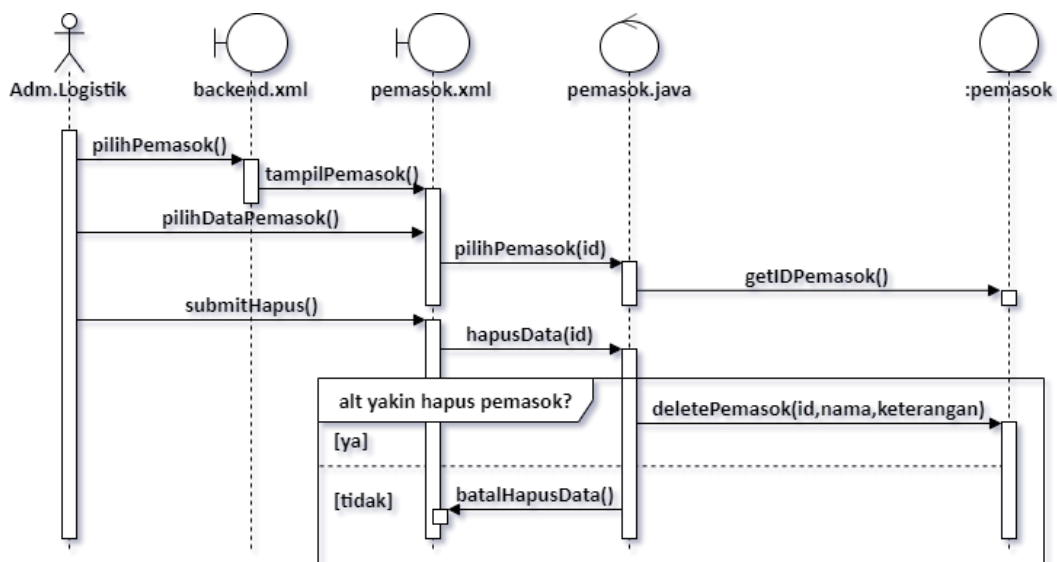
Gambar IV.16. *Sequence Diagram* Mengubah Pemasok

Pada Gambar IV.16 dapat dijelaskan mengenai skenario dalam mengubah alternatif atau data pemasok konstruksi yang dilakukan oleh Adm. Logistik. Ketika Adm. Logistik memilih menu alternatif pada halaman *backend* maka aplikasi akan menampilkan halaman yang berisi data alternatif. Kemudian

Adm. Logistik memilih data alternatif yang akan diubah yang meliputi nama dan deskripsi dari pemasok konstruksi saja untuk diperbaharui melalui aplikasi ke dalam *database*. Sedangkan id dari pemasok konstruksi tidak boleh diubah untuk kebutuhan pengenalan identitas dan *history*.

c) *Sequence Diagram* Menghapus Pemasok

Skenario menghapus alternatif atau data pemasok konstruksi yang diinisiasi oleh Adm. Logistik yang berinteraksi dengan aplikasi dengan menunjukkan serangkaian pesan (*message*) yang dikirim antara objek dapat dilihat urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.17.



Gambar IV.17. *Sequence Diagram* Menghapus Pemasok

Pada Gambar IV.17 dapat dijelaskan mengenai skenario dalam menghapus salah satu data pemasok konstruksi atau alternatif yang dilakukan oleh Adm. Logistik. Ketika Adm. Logistik memilih menu alternatif pada halaman

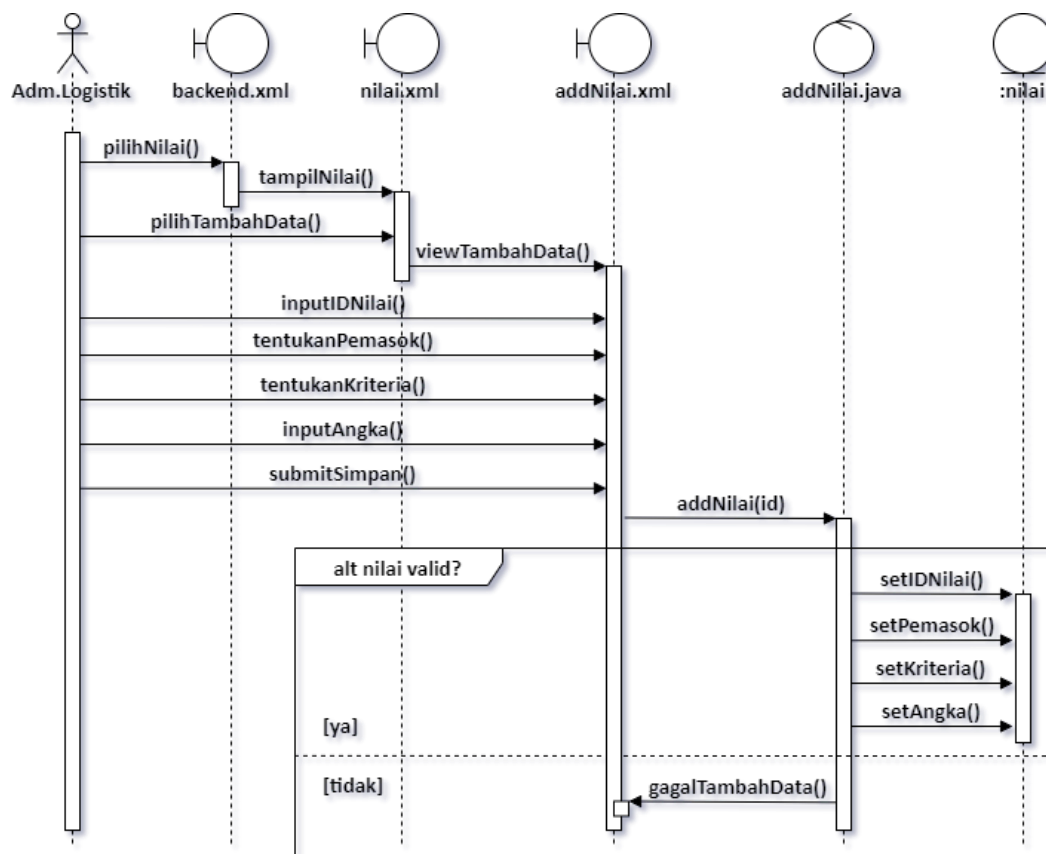
backend maka aplikasi akan menampilkan halaman yang berisi data alternatif. Kemudian Adm. Logistik memilih data alternatif yang akan dihapus untuk diperbaharui melalui aplikasi ke dalam *database*.

4. Sequence Diagram Mengelola Nilai

Pada *sequence diagram* mengelola nilai *rating* akan dipisahkan menjadi beberapa diagram berdasarkan perluasan aktivitasnya antara lain:

a) Sequence Diagram Menambah Nilai

Skenario menambah nilai *rating* dengan urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.18.

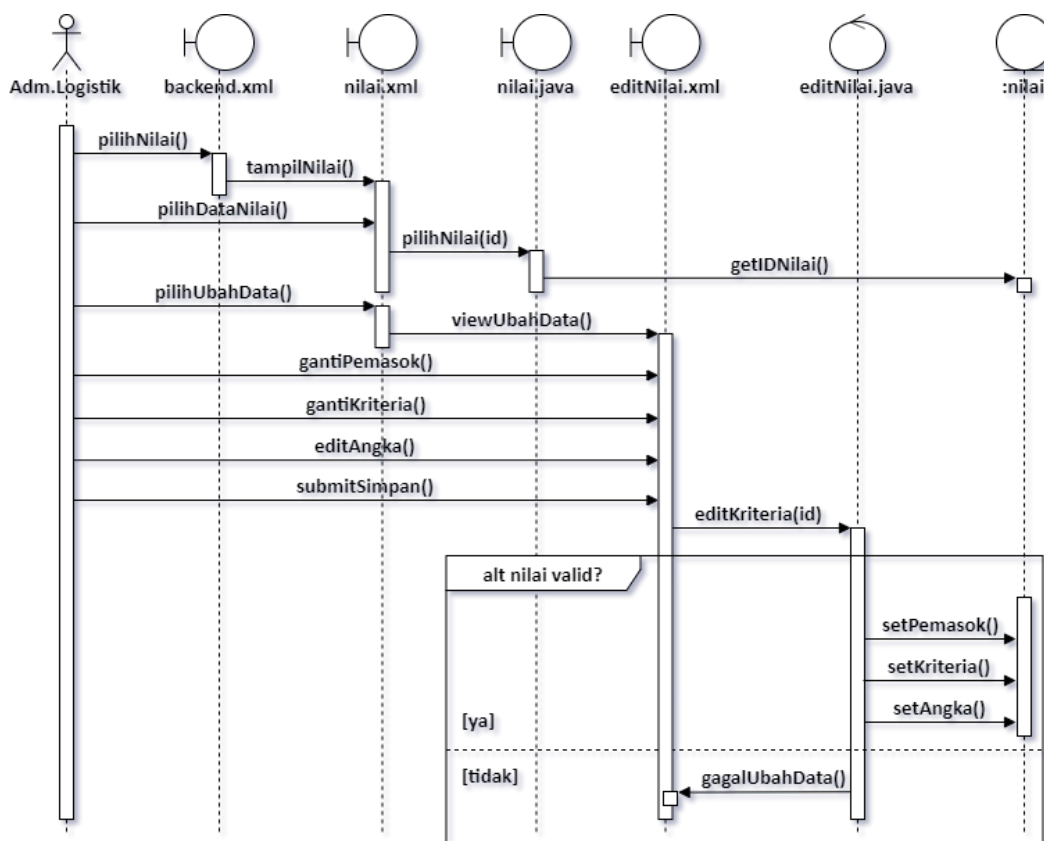


Gambar IV.18. Sequence Diagram Menambah Nilai

Pada Gambar IV.18 dapat dijelaskan bahwa Adm. Logistik dapat menambah data dari nilai *rating* yang meliputi id, pemasok, kriteria, dan nilai untuk disimpan melalui aplikasi ke dalam *database*.

b) *Sequence Diagram* Mengubah Nilai

Skenario mengubah data *rating* dengan urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.19.



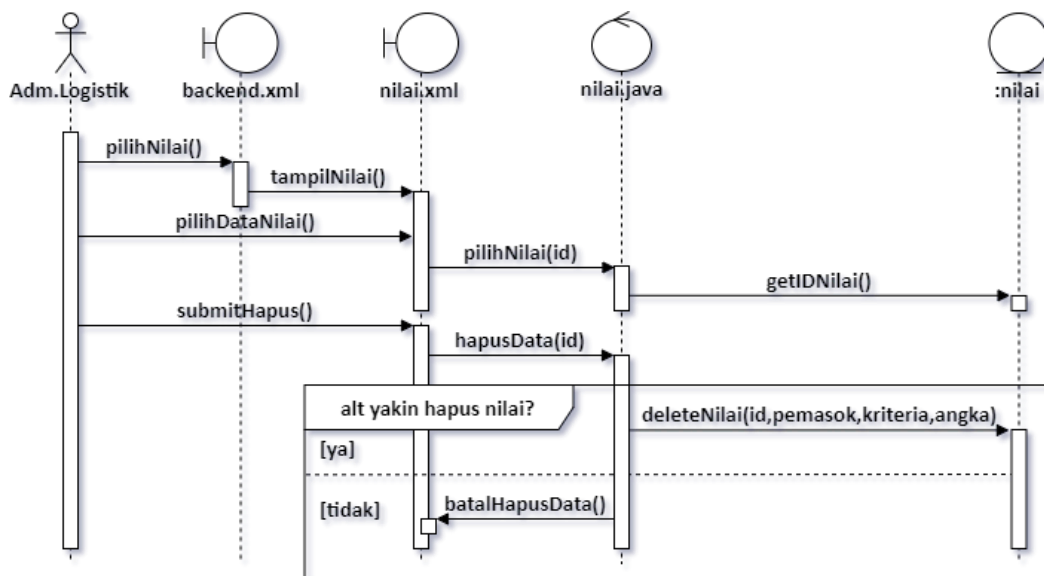
Gambar IV.19. Sequence Diagram Mengubah Nilai

Pada Gambar IV.19 dapat dijelaskan mengenai skenario dalam mengubah data *rating* yang dilakukan oleh Adm. Logistik. Ketika Adm. Logistik

memilih menu *rating* pada halaman *backend* maka aplikasi akan menampilkan halaman *rating* yang berisi data nilai. Kemudian Adm. Logistik memilih *rating* yang akan diubah meliputi alternatif, kriteria, dan nilai untuk diperbaharui melalui aplikasi ke dalam *database*.

c) Sequence Diagram Menghapus Nilai

Skenario menghapus data *rating* yang diinisiasi oleh Adm. Logistik yang berinteraksi dengan aplikasi dengan menunjukkan serangkaian pesan (*message*) yang dikirim antara objek dengan urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.20.



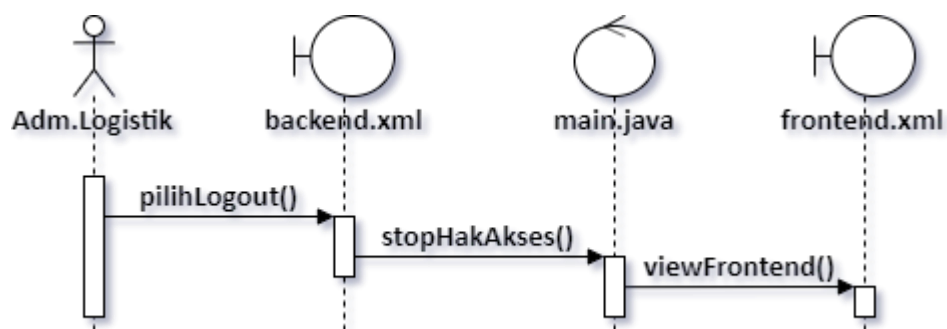
Gambar IV.20. Sequence Diagram Menghapus Nilai

Pada Gambar IV.20 dapat dijelaskan mengenai skenario dalam menghapus salah satu data *rating* yang dilakukan oleh Adm. Logistik. Ketika Adm. Logistik memilih menu *rating* pada halaman *backend* maka aplikasi akan menampilkan halaman yang berisi data *rating*. Kemudian Adm. Logistik

memilih data *rating* yang akan dihapus untuk diperbaharui melalui aplikasi ke dalam *database*.

5. Sequence Diagram Melakukan Logout

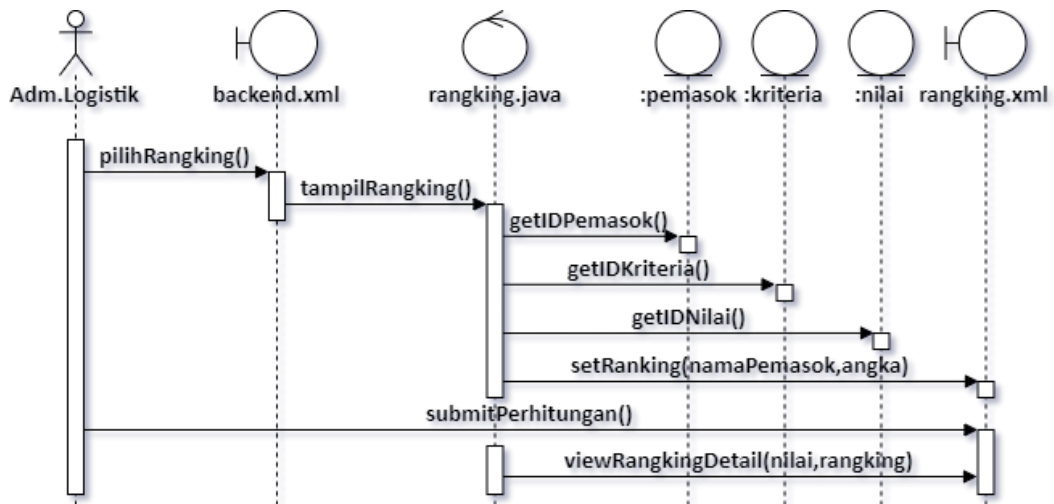
Skenario melakukan proses *logout* oleh Adm. Logistik dengan urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.21.



Gambar IV.21. Sequence Diagram Melakukan Logout

6. Sequence Diagram Mengakses Rangking

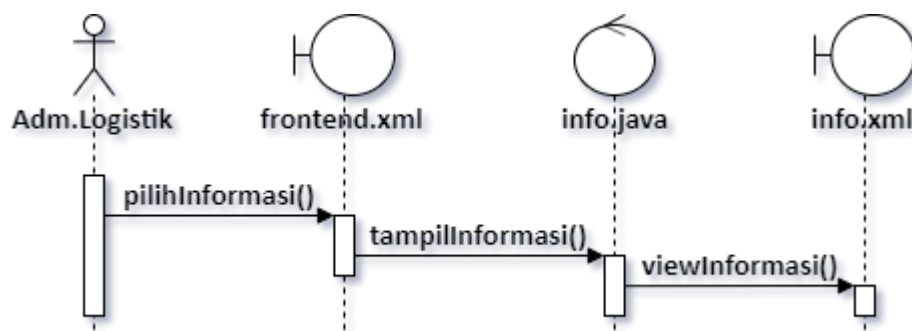
Skenario mengakses menu rangking oleh Adm. Logistik dengan urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.22.



Gambar IV.22. Sequence Diagram Mengakses Reranking

7. Sequence Diagram Mengakses Informasi

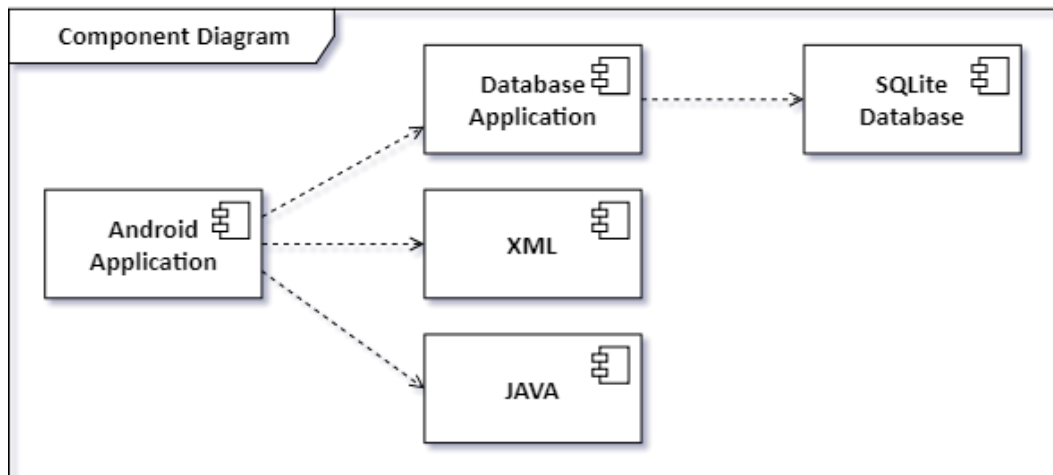
Skenario mengakses menu informasi oleh Adm. Logistik dengan urutannya menggunakan *sequence diagram* yang dapat dilihat pada Gambar IV.23.



Gambar IV.23. Sequence Diagram Mengakses Informasi

C. Component Diagram

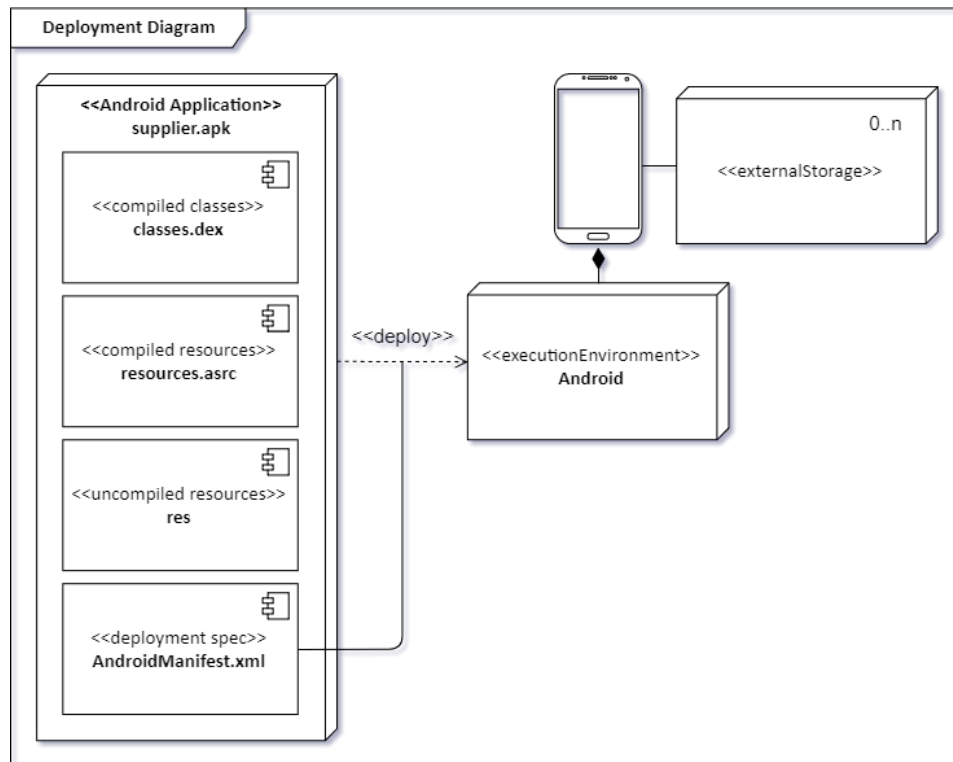
Menggambarkan struktur dan hubungan antar komponen perangkat lunak, termasuk ketergantungan diantaranya. *Component diagram* untuk aplikasi seleksi pemasok konstruksi dapat dilihat pada Gambar IV.24.



Gambar IV.24. *Component Diagram* Aplikasi Seleksi Pemasok Konstruksi

D. *Deployment Diagram*

Menggambarkan tata letak sistem secara fisik yang menampakkan bagian-bagian *software* yang berjalan pada *hardware* yang digunakan untuk mengimplementasikan sebuah sistem dan keterhubungan antara komponen-komponen *hardware*. *Deployment diagram* dari aplikasi pemasok konstruksi dapat dilihat pada Gambar IV.25.



Gambar IV.25. *Deployment Diagram* Aplikasi Seleksi Pemasok Konstruksi

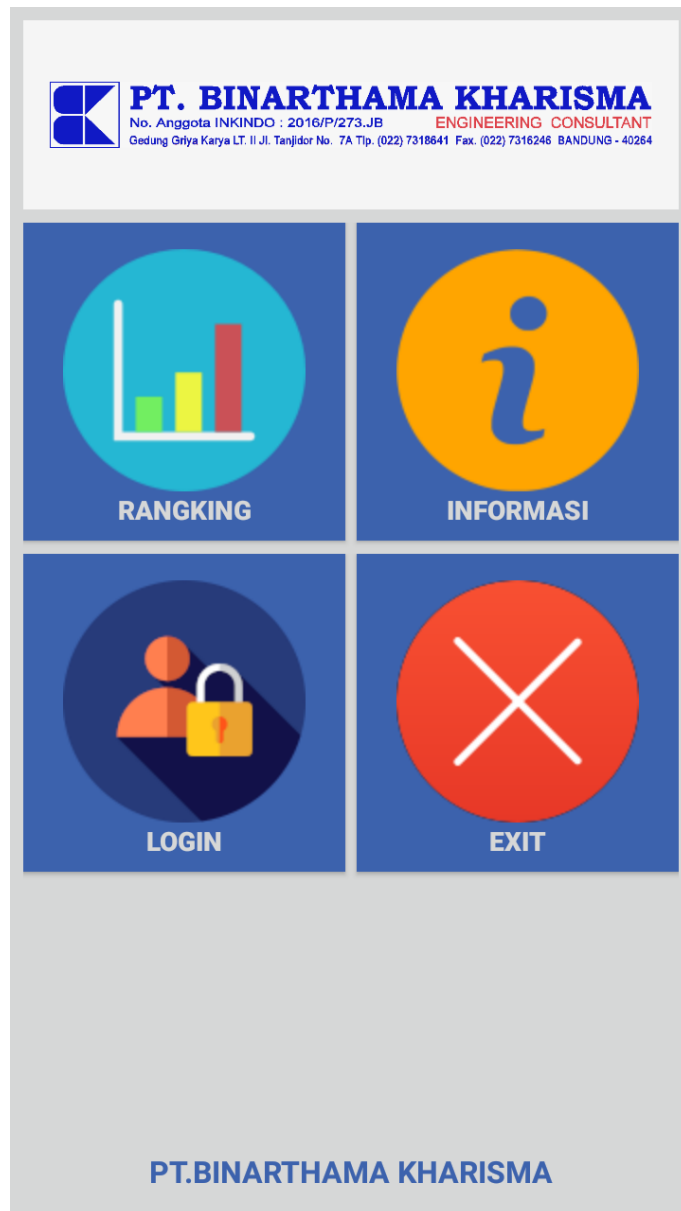
4.2.3. *User Interface*

User interface merupakan gambar antarmuka pengguna dari perancangan sistem yang dibuat. Perancangan antarmuka berfungsi untuk memudahkan interaksi antara pengguna dengan sistem.

1. Halaman Menu Utama (*Frontend*)

Tampilan halaman depan atau *frontend* merupakan halaman utama dari aplikasi seleksi pemasok konstruk ketika pertama kali menjalankan aplikasi tersebut.

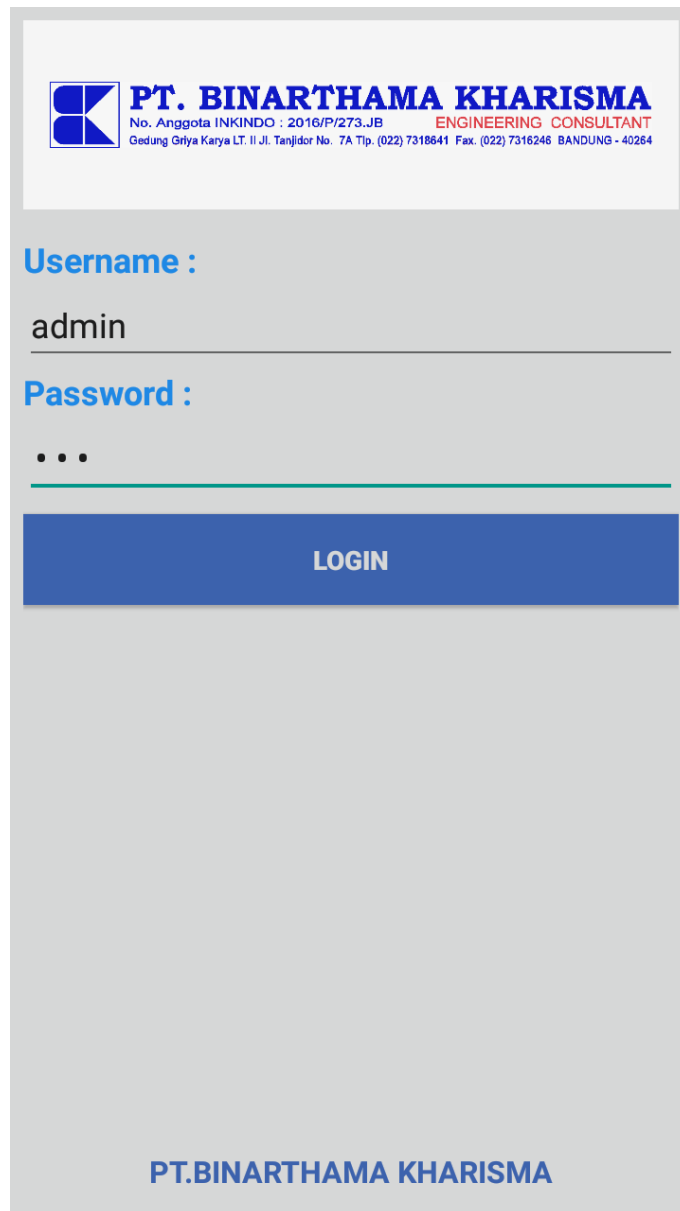
Tampilan dari *frontend* aplikasi dapat dilihat pada Gambar IV.26.



Gambar IV.26. Tampilan Halaman *Frontend*

2. Halaman *Login*

Untuk dapat melakukan pengelolaan data diperlukan otentikasi *login* pengguna agar tidak terjadi penyalahgunaan penilaian. Tampilan dari halaman *login* pengguna oleh Adm. Logistik dapat dilihat pada Gambar IV.27.



PT. BINARTHAMA KHARISMA
No. Anggota INKINDO : 2016/P/273.JB **ENGINEERING CONSULTANT**
Gedung Griya Karya LT. II Jl. Tanjidor No. 7A Tlp. (022) 7318641 Fax. (022) 7316246 BANDUNG - 40264

Username :

admin

Password :

...

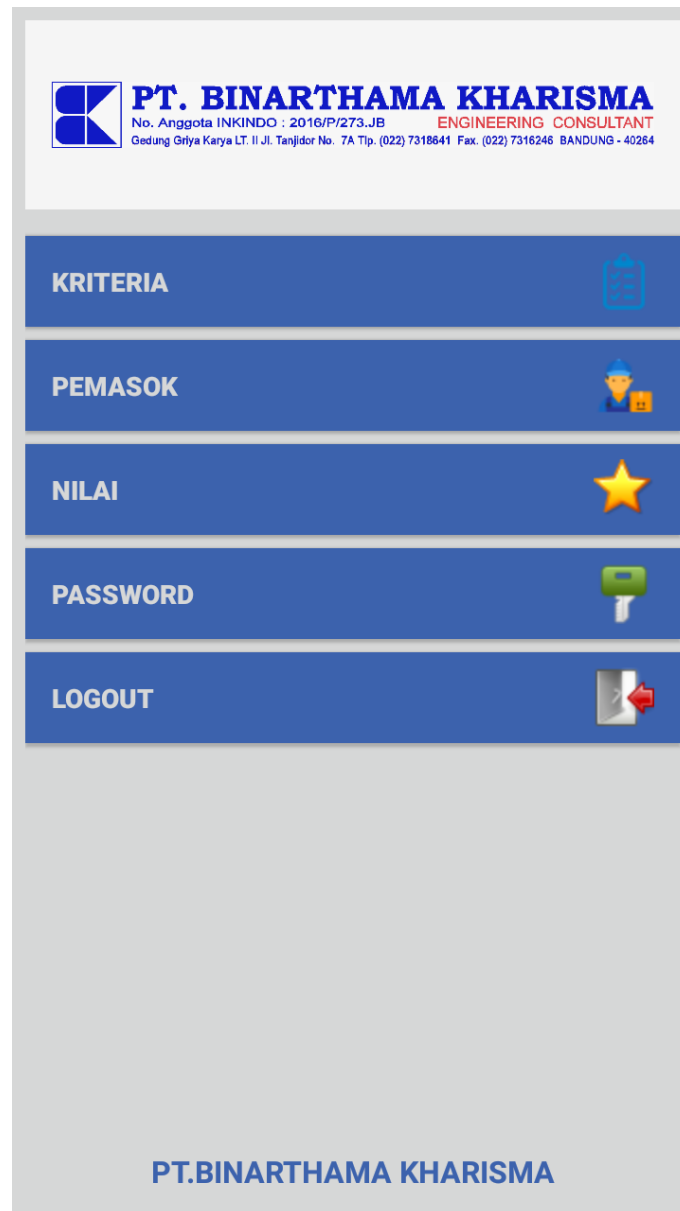
LOGIN

PT.BINARTHAMA KHARISMA

Gambar IV.27. Tampilan Halaman *Login*

3. Halaman *Backend*

Halaman *backend* atau *dashboard* terkadang disebut juga sebagai halaman administrator yang hanya dapat diakses oleh Adm. Logistik setelah *login* yang dapat dilihat pada Gambar IV.28.



Gambar IV.28. Tampilan Halaman *Backend*

4. Halaman Kriteria

Pada halaman kriteria terdapat beberapa data kriteria yang sudah ditentukan oleh *Surveyor*, namun dapat diperbaharui ketika terjadi kebijakan baru dari perusahaan. Tampilan dari halaman kriteria ditunjukkan pada Gambar IV.29.



PT. BINARTHAMA KHARISMA
No. Anggota INKINDO : 2016/P/273.JB **ENGINEERING CONSULTANT**
Gedung Griya Karya LT. II Jl. Tanjidor No. 7A Tlp. (022) 7318641 Fax. (022) 7316246 BANDUNG - 40264

1. Harga	Kepentingan: 3 (cost)
2. Kualitas	Kepentingan: 3 (benefit)
3. Stock	Kepentingan: 2 (benefit)
4. Pelayan	Kepentingan: 1 (benefit)
5. Pengiriman	Kepentingan: 1 (cost)

TAMBAH

Gambar IV.29. Tampilan Halaman Kriteria

5. Halaman Pemasok

Pada halaman ini ditampilkan data alternatif yang tersedia agar dapat digunakan untuk pengolahan data pemasok konstruksi dengan tampilan antarmuka pengguna pada Gambar IV.30.



PT. BINARTHAMA KHARISMA
No. Anggota INKINDO : 2016/P/273.JB **ENGINEERING CONSULTANT**
Gedung Griya Karya LT. II Jl. Tanjidor No. 7A Tlp. (022) 7318641 Fax. (022) 7316246 BANDUNG - 40264

1. TB.Indah
2. TB.Sejahtera
3. TB.Sukasari
4. TB.Setra Sari
5. TB.Makmur Sejahtera

TAMBAH

Gambar IV.30. Tampilan Halaman Pemasok

6. Halaman Nilai

Halaman nilai *rating* berisi penilaian yang diberikan oleh Adm. Logistik terhadap beberapa pemasok konstruksi yang sudah dimasukkan sebelumnya.

Tampilan dari halaman *rating* dapat dilihat pada Gambar IV.31.



PT. BINARTHAMA KHARISMA
 No. Anggota INKINDO : 2016/P/273.JB **ENGINEERING CONSULTANT**
 Gedung Griya Karya LT. II Jl. Tanjidor No. 7A Tlp. (022) 7318641 Fax. (022) 7316246 BANDUNG - 40264

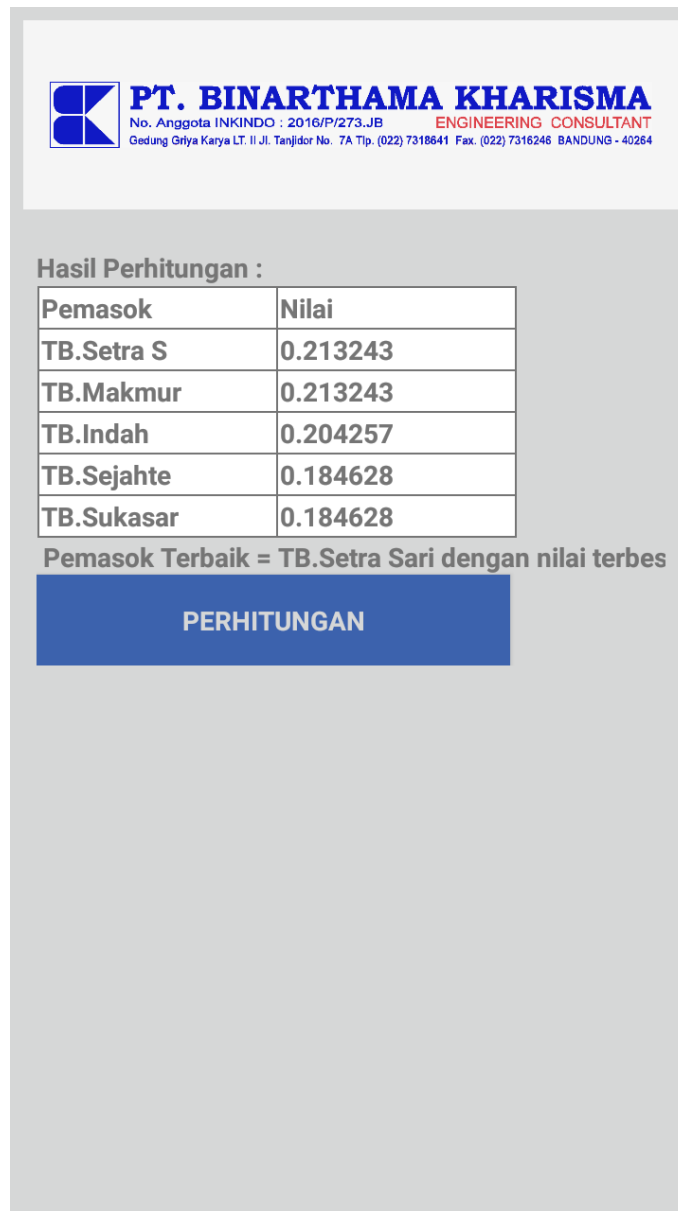
1. TB.Indah/Harga	Nilai = 70
2. TB.Indah/Kualitas	Nilai = 70
3. TB.Indah/Stock	Nilai = 80
4. TB.Indah/Pelayan	Nilai = 100
5. TB.Indah/Pengiriman	Nilai = 80
6. TB.Sejahtera/Harga	Nilai = 100
7. TB.Sejahtera/Kualitas	Nilai = 80
8. TB.Sejahtera/Stock	Nilai = 80
9. TB.Sejahtera/Pelayan	Nilai = 80
10. TB.Sejahtera/Pengiriman	

TAMBAH

Gambar IV.31. Tampilan Halaman Nilai

7. Halaman Ranging

Hasil akhir dari proses perhitungan metode *Weighted Product* yaitu menghasilkan peringkat atau ranging dari pemasok konstruksi dengan tampilan pada Gambar IV.32.



PT. BINARTHAMA KHARISMA
 No. Anggota INKINDO : 2016/P/273.JB **ENGINEERING CONSULTANT**
 Gedung Griya Karya LT. II Jl. Tanjidor No. 7A Tlp. (022) 7318641 Fax. (022) 7316246 BANDUNG - 40264

Hasil Perhitungan :

Pemasok	Nilai
TB.Setra S	0.213243
TB.Makmur	0.213243
TB.Indah	0.204257
TB.Sejahte	0.184628
TB.Sukasar	0.184628

Pemasok Terbaik = TB.Setra Sari dengan nilai terbes

PERHITUNGAN

Gambar IV.32. Tampilan Halaman Rangka

8. Halaman Informasi

Pada halaman informasi berisi deskripsi dari aplikasi seleksi pemasok konstruksi beserta keterangan mengenai sanggar tempat riset yaitu PT Binarthama Kharisma dengan tampilan pada Gambar IV.33.



Gambar IV.33. Tampilan Halaman Informasi

4.3. Code Generation

Tahap implementasi sistem adalah tahap penerapan sistem yang telah didesain atau dirancang, sehingga sistem yang telah dibuat dapat dioperasikan dan digunakan secara optimal sesuai dengan kebutuhan.

1. Class Pengguna

```

package com.wp_android;

import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class LoginActivity extends AppCompatActivity {

    EditText edloginusername;
    EditText edloginpassword;
    Button btnlogin;
    protected Cursor cursor;
    SQLHelper dbHelper;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //requestWindowFeature(Window.FEATURE_NO_TITLE);
        setContentView(R.layout.activity_login);

        dbHelper = new SQLHelper(this);

        edloginusername = (EditText)findViewById(R.id.edloginusername);
        edloginpassword = (EditText)findViewById(R.id.edloginpassword);

        btnlogin = (Button)findViewById(R.id.btnlogin);
        btnlogin.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                SQLiteDatabase db = dbHelper.getReadableDatabase();

                cursor = db.rawQuery("SELECT * FROM login WHERE username = '" + edloginusername.getText().toString() + "' AND password = '" + edloginpassword.getText().toString() + "'",null);

                if (cursor.getCount() > 0)
                {
                    Intent i = new Intent(LoginActivity.this, AdminActivity.class);
                    startActivity(i);
                    AdminActivity.userlogin = edloginusername.getText().toString();

```

```

        finish();
    }
    else
    {
        Toast.makeText(getApplicationContext(), "Login Gagal",
Toast.LENGTH_LONG).show();
    }

    }
    });
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is
present.
    getMenuInflater().inflate(R.menu.activity_login, menu);
    return true;
}
}

```

2. Class Kriteria

```

package com.wp_android;

import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;

public class KriteriaActivity extends AppCompatActivity {

    String[] array_id_kriteria;
    String[] array_nama_kriteria;
    String[] array_kepentingan;
    String[] array_cost_benefit;
    String[] array_kriteria;
    ListView listkriteria;
}

```

```

Button btnkriteriaadd;

protected Cursor cursor;
SQLHelper dbHelper;
public static KriteriaActivity obj;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //requestWindowFeature(Window.FEATURE_NO_TITLE);
    setContentView(R.layout.activity_kriteria);

    obj = this;

    dbHelper = new SQLHelper(this);

    btnkriteriaadd = (Button) findViewById(R.id.btnkriteriaadd);
    btnkriteriaadd.setOnClickListener(new Button.OnClickListener() {
        @Override
        public void onClick(View arg0) {
            Intent i = new Intent(KriteriaActivity.this,
AddKriteriaActivity.class);
            startActivity(i);
        }
    });

    RefreshList();
}

public void RefreshList()
{
    SQLiteDatabase db = dbHelper.getReadableDatabase();

    cursor = db.rawQuery("SELECT * FROM kriteria",null);

    array_id_kriteria = new String[cursor.getCount()];
    array_nama_kriteria = new String[cursor.getCount()];
    array_kepentingan = new String[cursor.getCount()];
    array_cost_benefit = new String[cursor.getCount()];
    array_kriteria = new String[cursor.getCount()];
    cursor.moveToFirst();
    for (int cc=0; cc < cursor.getCount(); cc++)
    {
        cursor.moveToPosition(cc);
        array_id_kriteria[cc] = cursor.getString(0).toString();
        array_nama_kriteria[cc] = cursor.getString(1).toString();
        array_kepentingan[cc] = cursor.getString(2).toString();
        array_cost_benefit[cc] = cursor.getString(3).toString();
        array_kriteria[cc] = cursor.getString(0).toString() + ". " +
cursor.getString(1).toString() + "\n Kepentingan: " +
cursor.getString(2).toString() + " (" + cursor.getString(3).toString() +
")";
    }
}

```

```

        listkriteria = (ListView)findViewById(R.id.listkriteria);
        listkriteria.setAdapter(new ArrayAdapter(this,
android.R.layout.simple_list_item_1, array_kriteria));
        listkriteria.setSelected(true);
        listkriteria.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> arg0, View arg1, int
arg2,
                                long arg3) {
                final int posisi = arg2;
                //.getItemAtPosition(arg2).toString();
                final CharSequence[] dialogitem = {"Edit", "Delete"};
                AlertDialog.Builder builder = new
AlertDialog.Builder(KriteriaActivity.this);
                builder.setTitle("Pilih ?");
                builder.setItems(dialogitem, new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int item) {
                        switch(item){
                            case 0 :
                                Intent i = new
Intent(getApplicationContext(), EditKriteriaActivity.class);
                                i.putExtra("id_kriteria",
array_id_kriteria[posisi]);
                                i.putExtra("nama_kriteria",
array_nama_kriteria[posisi]);
                                i.putExtra("kepentingan",
array_kepentingan[posisi]);
                                i.putExtra("cost_benefit",
array_cost_benefit[posisi]);
                                startActivity(i);
                                break;
                            case 1 :
                                SQLiteDatabase db =
dbHelper.getWritableDatabase();
                                db.execSQL("delete from kriteria where
id_kriteria = '" + array_id_kriteria[posisi] + "'");
                                RefreshList();
                                break;
                        }
                    }
                });
                builder.create().show();
            }
        });

        ((ArrayAdapter)listkriteria.getAdapter()).notifyDataSetChanged();
    }

    @Override

```



```

        public boolean onCreateOptionsMenu(Menu menu) {
            // Inflate the menu; this adds items to the action bar if it is
            present.
            getMenuInflater().inflate(R.menu.activity_kriteria, menu);
            return true;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
            switch (item.getItemId()) {
                case R.id.menu_kriteria_add:
                    Intent i = new Intent(KriteriaActivity.this,
                    AddKriteriaActivity.class);
                    startActivity(i);
                    return true;
                case R.id.menu_kriteria_refresh:
                    RefreshList();
                    return true;
                case R.id.menu_kriteria_exit:
                    finish();
                    return true;
            }
            return false;
        }
    }
}

```

3. Class Pemasok

```

package com.wp_android;

import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.widget.AdapterView;
import android.widget.AdapterView.Adapter;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;

public class AlternatifActivity extends AppCompatActivity {

    String[] array_id_alternatif;
}

```

```

String[] array_nama_alternatif;
String[] array_deskripsi;
String[] array_alternatif;
ListView listalternatif;

Button btnalternatifadd;

protected Cursor cursor;
SQLHelper dbHelper;
public static AlternatifActivity obj;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //requestWindowFeature(Window.FEATURE_NO_TITLE);
    setContentView(R.layout.activity_alternatif);

    obj = this;

    dbHelper = new SQLHelper(this);

    btnalternatifadd = (Button) findViewById(R.id.btnalternatifadd);
    btnalternatifadd.setOnClickListener(new Button.OnClickListener() {
        @Override
        public void onClick(View arg0) {
            Intent i = new Intent(AlternatifActivity.this,
AddAlternatifActivity.class);
            startActivity(i);
        }
    });

    RefreshList();
}

public void RefreshList()
{
    SQLiteDatabase db = dbHelper.getReadableDatabase();

    cursor = db.rawQuery("SELECT * FROM alternatif",null);

    array_id_alternatif = new String[cursor.getCount()];
    array_nama_alternatif = new String[cursor.getCount()];
    array_deskripsi = new String[cursor.getCount()];
    array_alternatif = new String[cursor.getCount()];
    cursor.moveToFirst();
    for (int cc=0; cc < cursor.getCount(); cc++)
    {
        cursor.moveToPosition(cc);
        array_id_alternatif[cc] = cursor.getString(0).toString();
        array_nama_alternatif[cc] = cursor.getString(1).toString();
        array_deskripsi[cc] = cursor.getString(2).toString();
        array_alternatif[cc] = cursor.getString(0).toString() + ". " +

```

```

cursor.getString(1).toString() + "\n" + cursor.getString(2).toString();
    }

    listalternatif = (ListView)findViewById(R.id.listalternatif);
    listalternatif.setAdapter(new ArrayAdapter(this,
android.R.layout.simple_list_item_1, array_alternatif));
    listalternatif.setSelected(true);
    listalternatif.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> arg0, View arg1, int
arg2,
                                long arg3) {
            final int posisi = arg2;
            //.getItemAtPosition(arg2).toString();
            final CharSequence[] dialogitem = {"Edit", "Delete"};
            AlertDialog.Builder builder = new
AlertDialog.Builder(AlternatifActivity.this);
            builder.setTitle("Pilih ?");
            builder.setItems(dialogitem, new
DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int item) {
                    switch(item){
                        case 0 :
                            Intent i = new
Intent(getApplicationContext(), EditAlternatifActivity.class);
                            i.putExtra("id_alternatif",
array_id_alternatif[posisi]);
                            i.putExtra("nama_alternatif",
array_nama_alternatif[posisi]);
                            i.putExtra("deskripsi",
array_deskripsi[posisi]);
                            startActivity(i);
                            break;
                        case 1 :
                            SQLiteDatabase db =
dbHelper.getWritableDatabase();
                            db.execSQL("delete from alternatif where
id_alternatif = '" + array_id_alternatif[posisi] + "'");
                            RefreshList();
                            break;
                    }
                }
            });
            builder.create().show();
        }
    });

((ArrayAdapter)listalternatif.getAdapter()).notifyDataSetChanged();
}

@Override

```

```

        public boolean onCreateOptionsMenu(Menu menu) {
            // Inflate the menu; this adds items to the action bar if it is
            present.
            getMenuInflater().inflate(R.menu.activity_alternatif, menu);
            return true;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
            switch (item.getItemId()) {
                case R.id.menu_alternatif_add:
                    Intent i = new Intent(AlternatifActivity.this,
                    AddAlternatifActivity.class);
                    startActivity(i);
                    return true;
                case R.id.menu_alternatif_refresh:
                    RefreshList();
                    return true;
                case R.id.menu_alternatif_exit:
                    finish();
                    return true;
            }
            return false;
        }
    }
}

```

4. Class Nilai

```

package com.wp_android;

import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.graphics.Color;
import android.graphics.Typeface;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Display;
import android.view.Menu;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.widget.Button;
import android.widget.HorizontalScrollView;
import android.widget.LinearLayout;
import android.widget.ScrollView;
import android.widget.TextView;
import android.view.ViewGroup.LayoutParams;

public class AnalisaWpActivity extends AppCompatActivity {

```

```

String[] alternatif;
String[] kriteria;
String[] costbenefit;
double[] kepentingan;
double[][] alternatifkriteria;
double[] bobotkepentingan;
double jumlahkepentingan = 0;
double[] pangkat;
String[] id_alternatif;
String[] id_kriteria;

protected Cursor cursor;
SQLHelper dbHelper;

public LinearLayout tampiltabel(double[][] data)
{
    LinearLayout.LayoutParams lpp = new
LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.WRAP_CONTENT);
    lpp.setMargins(1, 1, 1, 1);

    LinearLayout llv = new LinearLayout(this);
    llv.setOrientation(LinearLayout.VERTICAL);
    for (int i=0; i<data.length; i++)
    {
        LinearLayout llh = new LinearLayout(this);
        llh.setOrientation(LinearLayout.HORIZONTAL);
        llv.addView(llh);

        for (int j=0; j<data[i].length; j++)
        {
            TextView textview = new TextView(this);
            int maxLength = 10;
            if (maxLength > String.valueOf(data[i][j]).length()) {
maxLength = String.valueOf(data[i][j]).length(); }
            textview.setText(String.valueOf(data[i][j]).substring(0,
maxLength));

            textview.setWidth(100);
            textview.setTypeface(Typeface.DEFAULT_BOLD);
            textview.setBackgroundColor(Color.parseColor("#ffffff"));
            textview.setLayoutParams(lpp);
            textview.setPadding(2, 2, 2, 2);
            llh.addView(textview);
        }
    }
    llv.setBackgroundColor(Color.parseColor("#0000ff"));
    llv.setLayoutParams(lpp);
    llv.setPadding(1, 1, 1, 1);

    return llv;
}

```

```

    public LinearLayout tampilbaris(String[] data)
    {
        LinearLayout.LayoutParams lpp = new
        LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
        LayoutParams.WRAP_CONTENT);
        lpp.setMargins(1, 1, 1, 1);

        LinearLayout llv = new LinearLayout(this);
        llv.setOrientation(LinearLayout.VERTICAL);

        LinearLayout llh = new LinearLayout(this);
        llh.setOrientation(LinearLayout.HORIZONTAL);
        llv.addView(llh);

        for (int i=0; i<data.length; i++)
        {
            TextView textview = new TextView(this);
            int maxLength = 10;
            if (maxLength > String.valueOf(data[i]).length()) { maxLength =
String.valueOf(data[i]).length(); }
            textview.setText(data[i].substring(0, maxLength));
            textview.setWidth(100);
            textview.setTypeface(Typeface.DEFAULT_BOLD);
            textview.setBackgroundColor(Color.parseColor("#ffffff"));
            textview.setLayoutParams(lpp);
            textview.setPadding(2, 2, 2, 2);
            llh.addView(textview);
        }

        llv.setBackgroundColor(Color.parseColor("#0000ff"));
        llv.setLayoutParams(lpp);
        llv.setPadding(1, 1, 1, 1);

        return llv;
    }

    public LinearLayout tampilbaris(double[] data)
    {
        LinearLayout.LayoutParams lpp = new
        LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
        LayoutParams.WRAP_CONTENT);
        lpp.setMargins(1, 1, 1, 1);

        LinearLayout llv = new LinearLayout(this);
        llv.setOrientation(LinearLayout.VERTICAL);

        LinearLayout llh = new LinearLayout(this);
        llh.setOrientation(LinearLayout.HORIZONTAL);
        llv.addView(llh);

        for (int i=0; i<data.length; i++)
        {
            TextView textview = new TextView(this);

```

```

        int maxLength = 8;
        if (maxLength > String.valueOf(data[i]).length()) { maxLength =
String.valueOf(data[i]).length(); }
        textview.setText(String.valueOf(data[i]).substring(0,
maxLength));
        textview.setWidth(100);
        textview.setTypeface(Typeface.DEFAULT_BOLD);
        textview.setBackgroundColor(Color.parseColor("#ffffff"));
        textview.setLayoutParams(lpp);
        textview.setPadding(2, 2, 2, 2);
        llh.addView(textview);
    }

    llv.setBackgroundColor(Color.parseColor("#0000ff"));
    llv.setLayoutParams(lpp);
    llv.setPadding(1, 1, 1, 1);

    return llv;
}

public LinearLayout tampilkolom(String[] data)
{
    LinearLayout.LayoutParams lpp = new
LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.WRAP_CONTENT);
    lpp.setMargins(1, 1, 1, 1);

    LinearLayout llv = new LinearLayout(this);
    llv.setOrientation(LinearLayout.VERTICAL);
    for (int i=0; i<data.length; i++)
    {
        LinearLayout llh = new LinearLayout(this);
        llh.setOrientation(LinearLayout.HORIZONTAL);
        llv.addView(llh);

        TextView textview = new TextView(this);
        int maxLength = 10;
        if (maxLength > data[i].length()) { maxLength =
data[i].length(); }
        textview.setText(data[i].substring(0, maxLength));
        textview.setWidth(100);
        textview.setTypeface(Typeface.DEFAULT_BOLD);
        textview.setBackgroundColor(Color.parseColor("#ffffff"));
        textview.setLayoutParams(lpp);
        textview.setPadding(2, 2, 2, 2);
        llh.addView(textview);
    }

    llv.setBackgroundColor(Color.parseColor("#0000ff"));
    llv.setLayoutParams(lpp);
    llv.setPadding(1, 1, 1, 1);

    return llv;
}

```

```

    }

    public LinearLayout tampilkolom(double[] data)
    {
        LinearLayout.LayoutParams lpp = new
        LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
        LayoutParams.WRAP_CONTENT);
        lpp.setMargins(1, 1, 1, 1);

        LinearLayout llv = new LinearLayout(this);
        llv.setOrientation(LinearLayout.VERTICAL);
        for (int i=0; i<data.length; i++)
        {
            LinearLayout llh = new LinearLayout(this);
            llh.setOrientation(LinearLayout.HORIZONTAL);
            llv.addView(llh);

            TextView textview = new TextView(this);
            int maxLength = 8;
            if (maxLength > String.valueOf(data[i]).length()) { maxLength =
            String.valueOf(data[i]).length(); }
            textview.setText(String.valueOf(data[i]).substring(0,
            maxLength));
            textview.setWidth(100);
            textview.setTypeface(Typeface.DEFAULT_BOLD);
            textview.setBackgroundColor(Color.parseColor("#ffffff"));
            textview.setLayoutParams(lpp);
            textview.setPadding(2, 2, 2, 2);
            llh.addView(textview);
        }

        llv.setBackgroundColor(Color.parseColor("#0000ff"));
        llv.setLayoutParams(lpp);
        llv.setPadding(1, 1, 1, 1);

        return llv;
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //requestWindowFeature(Window.FEATURE_NO_TITLE);
        setContentView(R.layout.activity_analisa_wp);

        dbHelper = new SQLHelper(this);

        SQLiteDatabase db = dbHelper.getReadableDatabase();

        cursor = db.rawQuery("SELECT * FROM alternatif",null);

        alternatif = new String[cursor.getCount()];
        id_alternatif = new String[cursor.getCount()];
    }

```



```

        cursor.moveToFirst();
        for (int cc=0; cc < cursor.getCount(); cc++)
        {
            cursor.moveToPosition(cc);
            alternatif[cc] = cursor.getString(1).toString();
            id_alternatif[cc] = cursor.getString(0).toString();
        }

        cursor = db.rawQuery("SELECT * FROM kriteria",null);

        kriteria = new String[cursor.getCount()];
        kepentingan = new double[cursor.getCount()];
        costbenefit = new String[cursor.getCount()];
        id_kriteria = new String[cursor.getCount()];

        cursor.moveToFirst();
        for (int cc=0; cc < cursor.getCount(); cc++)
        {
            cursor.moveToPosition(cc);
            kriteria[cc] = cursor.getString(1).toString();
            kepentingan[cc] =
Double.parseDouble(cursor.getString(2).toString());
            costbenefit[cc] = cursor.getString(3).toString();
            id_kriteria[cc] = cursor.getString(0).toString();
        }

        alternatifkriteria = new
double[alternatif.length][kriteria.length];

        for (int i=0;i<alternatif.length;i++)
        {
            for (int j=0;j<kriteria.length;j++)
            {
                cursor = db.rawQuery("SELECT * FROM alternatif_kriteria
WHERE id_alternatif = '" + id_alternatif[i] + "' AND id_kriteria = '" +
id_kriteria[j] + "'",null);

                cursor.moveToFirst();
                if (cursor.getCount() > 0)
                {
                    cursor.moveToPosition(0);
                    alternatifkriteria[i][j] =
Double.parseDouble(cursor.getString(3).toString());
                }
            }
        }

        for (int i=0;i<kriteria.length;i++)
        {
            jumlahkepentingan = jumlahkepentingan + kepentingan[i];
        }

        bobotkepentingan = new double[kriteria.length];

```

```

for (int i=0;i<kriteria.length;i++)
{
    bobotkepentingan[i] = kepentingan[i] / jumlahkepentingan;
}

pangkat = new double[kriteria.length];

for (int i=0;i<kriteria.length;i++)
{
    if (costbenefit[i].equals("cost")) // == "cost")
    {
        pangkat[i] = -1 * bobotkepentingan[i];
    }
    else //benefit
    {
        pangkat[i] = 1 * bobotkepentingan[i];
    }
}

double[] nilai_s = new double[alternatif.length];
double total_s = 0;

for (int i=0;i<alternatif.length;i++)
{
    nilai_s[i] = 1; //karena perkalian bukan penambahan, kalo
    diberi nilai awal 0 hasilnya malah selalu nol
    for (int j=0;j<kriteria.length;j++)
    {
        //echo "$nilai_s[$i] * ".pow($alternatifkriteria[$i][$j],
        $pangkat[$j])."<br/>";
        nilai_s[i] = nilai_s[i] *
        Math.pow(alternatifkriteria[i][j], pangkat[j]);
    }
    total_s = total_s + nilai_s[i];
}

double[] hasil_v = new double[alternatif.length];

for (int i=0;i<alternatif.length;i++)
{
    hasil_v[i] = nilai_s[i] / total_s;
}

String[] alternatiffrangking = new String[alternatif.length];
double[] hasilrangking = new double[alternatif.length];

for (int i=0;i<alternatif.length;i++)
{
    hasilrangking[i] = hasil_v[i];
    alternatiffrangking[i] = alternatif[i];
}

```

```

        for (int i=0;i<alternatif.length;i++)
        {
            for (int j=i;j<alternatif.length;j++)
            {
                if (hasilrangking[j] > hasilrangking[i])
                {
                    double tmphasil = hasilrangking[i];
                    String tmpalternatif = alternatifrangking[i];
                    hasilrangking[i] = hasilrangking[j];
                    alternatifrangking[i] = alternatifrangking[j];
                    hasilrangking[j] = tmphasil;
                    alternatifrangking[j] = tmpalternatif;
                }
            }
        }

        Display display = getWindowManager().getDefaultDisplay();

        ScrollView sv = (ScrollView)findViewById(R.id.svAnalisa); // new
        ScrollView(this);
        //ViewGroup.LayoutParams lp = new
        LayoutParams(LayoutParams.FILL_PARENT, LayoutParams.FILL_PARENT);
        //sv.setLayoutParams(lp);
        LinearLayout l = new LinearLayout(this);
        l.setOrientation(LinearLayout.VERTICAL);
        sv.addView(l);
        HorizontalScrollView hv = new HorizontalScrollView(this);
        ViewGroup.LayoutParams lp = new
        LayoutParams(LayoutParams.MATCH_PARENT, LayoutParams.MATCH_PARENT);
        hv.setLayoutParams(lp);
        l.addView(hv);

        LinearLayout ll = new LinearLayout(this);
        ll.setOrientation(LinearLayout.VERTICAL);
        ll.setMinimumWidth(700);

        TextView tvhasilrankingfinal = new TextView(this);
        tvhasilrankingfinal.setText("Hasil Perhitungan :");
        tvhasilrankingfinal.setTextColor(Color.parseColor("#0000ff"));
        tvhasilrankingfinal.setTypeface(Typeface.DEFAULT_BOLD);
        //tvhasilrankingfinal.setWidth(100);
        //ViewGroup.LayoutParams lpp = new
        LayoutParams(LayoutParams.MATCH_PARENT, LayoutParams.WRAP_CONTENT);
        ViewGroup.LayoutParams lpp = new LayoutParams(display.getWidth(),
        LayoutParams.WRAP_CONTENT);
        tvhasilrankingfinal.setLayoutParams(lpp);
        ll.addView(tvhasilrankingfinal);
        LinearLayout llalternatifrangkingfinal = new LinearLayout(this); //
        = tampilkolom(alternatifrangking);

        LinearLayout.LayoutParams lppp = new
        LinearLayout.LayoutParams(LayoutParams.WRAP_CONTENT,

```

```

LayoutParams.WRAP_CONTENT);
    lppp.setMargins(1, 1, 1, 1);

    llalternatifrankingfinal.setOrientation(LinearLayout.VERTICAL);

    LinearLayout llhjudul = new LinearLayout(this);
    llhjudul.setOrientation(LinearLayout.HORIZONTAL);
    llalternatifrankingfinal.addView(llhjudul);

    TextView textviewAlternatifJudul = new TextView(this);
    textviewAlternatifJudul.setText("Alternatif");
    textviewAlternatifJudul.setWidth(100);
    textviewAlternatifJudul.setTypeface(Typeface.DEFAULT_BOLD);

textviewAlternatifJudul.setBackgroundColor(Color.parseColor("#ffffff"));
    textviewAlternatifJudul.setLayoutParams(lppp);
    textviewAlternatifJudul.setPadding(2, 2, 2, 2);
    llhjudul.addView(textviewAlternatifJudul);

    TextView textviewHasilJudul = new TextView(this);
    textviewHasilJudul.setText("Nilai");
    textviewHasilJudul.setWidth(100);
    textviewHasilJudul.setTypeface(Typeface.DEFAULT_BOLD);
    textviewHasilJudul.setBackgroundColor(Color.parseColor("#ffffff"));
    textviewHasilJudul.setLayoutParams(lppp);
    textviewHasilJudul.setPadding(2, 2, 2, 2);
    llhjudul.addView(textviewHasilJudul);

    for (int i=0; i<hasilranking.length; i++)
    {
        LinearLayout llh = new LinearLayout(this);
        llh.setOrientation(LinearLayout.HORIZONTAL);
        llalternatifrankingfinal.addView(llh);

        TextView textviewAlternatifRanking = new TextView(this);
        int maxLength = 10;
        if (maxLength > alternatiffranking[i].length()) { maxLength =
alternatiffranking[i].length(); }

textviewAlternatifRanking.setText(alternatiffranking[i].substring(0,
maxLength));
        textviewAlternatifRanking.setWidth(100);
        textviewAlternatifRanking.setTypeface(Typeface.DEFAULT_BOLD);

textviewAlternatifRanking.setBackgroundColor(Color.parseColor("#ffffff"));
        textviewAlternatifRanking.setLayoutParams(lppp);
        textviewAlternatifRanking.setPadding(2, 2, 2, 2);
        llh.addView(textviewAlternatifRanking);

        TextView textviewHasilRanking = new TextView(this);
        maxLength = 8;
        if (maxLength > String.valueOf(hasilranking[i]).length()) {
maxLength = String.valueOf(hasilranking[i]).length(); }

```

```

textViewHasilRanking.setText(String.valueOf(hasilrangking[i]).substring(0,
maxLength));
        textViewHasilRanking.setWidth(100);
        textViewHasilRanking.setTypeface(Typeface.DEFAULT_BOLD);

textViewHasilRanking.setBackgroundColor(Color.parseColor("#ffffff"));
        textViewHasilRanking.setLayoutParams(lppp);
        textViewHasilRanking.setPadding(2, 2, 2, 2);
        llh.addView(textViewHasilRanking);
    }

    llalternatifrankingfinal.setBackgroundColor(Color.parseColor("#0000ff"));
        llalternatifrankingfinal.setLayoutParams(lppp);
        llalternatifrankingfinal.setPadding(1, 1, 1, 1);

        llalternatifrankingfinal.setOrientation(LinearLayout.VERTICAL);
        ll.addView(llalternatifrankingfinal);

        TextView tvalternatifterbaikfinal = new TextView(this);
        tvalternatifterbaikfinal.setText("Alternatif Terbaik = " +
alternatifrangking[0] + " dengan nilai terbesar = " +
Double.valueOf(hasilrangking[0]));
        tvalternatifterbaikfinal.setTextColor(Color.parseColor("#0000ff"));
        tvalternatifterbaikfinal.setTypeface(Typeface.DEFAULT_BOLD);
        //tvalternatifterbaikfinal.setWidth(300);
        ll.addView(tvalternatifterbaikfinal);

        final LinearLayout llhasil = new LinearLayout(this);

        Button btnshow = new Button(this);
        btnshow.setText("Perhitungan");
        btnshow.setBackgroundColor(Color.parseColor("#0000ff"));
        btnshow.setTypeface(Typeface.DEFAULT_BOLD);
        btnshow.setTextColor(Color.parseColor("#0000ff"));
        btnshow.setBackgroundColor(Color.parseColor("#01fecf"));
        //Display display = getWindowManager().getDefaultDisplay();
        //btnshow.setWidth(display.getWidth());
        //btnshow.setWidth(Math.max(display.getWidth(),500));
        //btnshow.setWidth(display.getWidth());
        //btnshow.setMinimumWidth(300);//
Width(1000);//Math.max(display.getWidth(),500));
        //btnshow.setGravity(Gravity.LEFT);
        btnshow.setLayoutParams(new LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.WRAP_CONTENT)); // Width(50);
        //btnshow.setWidth(-2);
        //btnshow.setLayoutParams(new LayoutParams(10, 50)); // Width(50);
        btnshow.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub

```

```

        llhasil.setVisibility(View.VISIBLE);
    }
});
ll.addView(btnshow);

llhasil.setOrientation(LinearLayout.VERTICAL);
llhasil.setVisibility(View.GONE);
ll.addView(llhasil);

TextView tvalternatif = new TextView(this);
tvalternatif.setText("Alternatif :");
tvalternatif.setTextColor(Color.parseColor("#0000ff"));
tvalternatif.setTypeface(Typeface.DEFAULT_BOLD);
//tvalternatif.setWidth(100);
llhasil.addView(tvalternatif);
LinearLayout llalternatif = tampilbaris(alternatif);
llalternatif.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llalternatif);

TextView tvkriteria = new TextView(this);
tvkriteria.setText("Kriteria :");
tvkriteria.setTextColor(Color.parseColor("#0000ff"));
tvkriteria.setTypeface(Typeface.DEFAULT_BOLD);
//tvkriteria.setWidth(100);
llhasil.addView(tvkriteria);
LinearLayout llkriteria = tampilbaris(kriteria);
llkriteria.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llkriteria);

TextView tvcostbenefit = new TextView(this);
tvcostbenefit.setText("CostBenefit :");
tvcostbenefit.setTextColor(Color.parseColor("#0000ff"));
tvcostbenefit.setTypeface(Typeface.DEFAULT_BOLD);
//tvcostbenefit.setWidth(100);
llhasil.addView(tvcostbenefit);
LinearLayout llcostbenefit = tampilbaris(costbenefit);
llcostbenefit.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llcostbenefit);

TextView tvkepentingan = new TextView(this);
tvkepentingan.setText("Kepentingan :");
tvkepentingan.setTextColor(Color.parseColor("#0000ff"));
tvkepentingan.setTypeface(Typeface.DEFAULT_BOLD);
//tvkepentingan.setWidth(100);
llhasil.addView(tvkepentingan);
LinearLayout llkepentingan = tampilbaris(kepentingan);
llkepentingan.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llkepentingan);

TextView tvalternatifkriteria = new TextView(this);
tvalternatifkriteria.setText("Alternatif Kriteria :");
tvalternatifkriteria.setTextColor(Color.parseColor("#0000ff"));

```

```

tvalternatifkriteria.setTypeface(Typeface.DEFAULT_BOLD);
//tvalternatifkriteria.setWidth(100);
llhasil.addView(tvalternatifkriteria);
LinearLayout llalternatifkriteria =
tampiltabel(alternatifkriteria);
llalternatifkriteria.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llalternatifkriteria);

TextView tvbobotkepentingan = new TextView(this);
tvbobotkepentingan.setText("Bobot Kepentingan :");
tvbobotkepentingan.setTextColor(Color.parseColor("#0000ff"));
tvbobotkepentingan.setTypeface(Typeface.DEFAULT_BOLD);
//tvbobotkepentingan.setWidth(100);
llhasil.addView(tvbobotkepentingan);
LinearLayout llbobotkepentingan = tampilbaris(bobotkepentingan);
llbobotkepentingan.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llbobotkepentingan);

TextView tvpangkat = new TextView(this);
tvpangkat.setText("Pangkat :");
tvpangkat.setTextColor(Color.parseColor("#0000ff"));
tvpangkat.setTypeface(Typeface.DEFAULT_BOLD);
//tvpangkat.setWidth(100);
llhasil.addView(tvpangkat);
LinearLayout llpangkat = tampilkolom(pangkat);
llpangkat.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llpangkat);

TextView tvnilai_s = new TextView(this);
tvnilai_s.setText("Nilai S :");
tvnilai_s.setTextColor(Color.parseColor("#0000ff"));
tvnilai_s.setTypeface(Typeface.DEFAULT_BOLD);
//tvnilai_s.setWidth(100);
llhasil.addView(tvnilai_s);
LinearLayout llnilai_s = tampilkolom(nilai_s);
llnilai_s.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llnilai_s);

TextView tvhasil_v = new TextView(this);
tvhasil_v.setText("Hasil V:");
tvhasil_v.setTextColor(Color.parseColor("#0000ff"));
tvhasil_v.setTypeface(Typeface.DEFAULT_BOLD);
//tvhasil_v.setWidth(100);
llhasil.addView(tvhasil_v);
LinearLayout llhasil_v = tampilkolom(hasil_v);
llhasil_v.setOrientation(LinearLayout.VERTICAL);
llhasil.addView(llhasil_v);

TextView tvhasilranking = new TextView(this);
tvhasilranking.setText("Hasil Ranking :");
tvhasilranking.setTextColor(Color.parseColor("#0000ff"));
tvhasilranking.setTypeface(Typeface.DEFAULT_BOLD);

```

```

        //tvhasilranking.setWidth(100);
        llhasil.addView(tvhasilranking);
        LinearLayout llhasilranking = tampilkolom(hasilrangking);
        llhasilranking.setOrientation(LinearLayout.VERTICAL);
        llhasil.addView(llhasilranking);

        TextView tvalternatifranking = new TextView(this);
        tvalternatifranking.setText("Alternatif Ranking :");
        tvalternatifranking.setTextColor(Color.parseColor("#0000ff"));
        tvalternatifranking.setTypeface(Typeface.DEFAULT_BOLD);
        //tvalternatifranking.setWidth(100);
        llhasil.addView(tvalternatifranking);
        LinearLayout llalternatifranking = tampilkolom(alternatifrangking);
        llalternatifranking.setOrientation(LinearLayout.VERTICAL);
        llhasil.addView(llalternatifranking);

        TextView tvalternatifterbaik = new TextView(this);
        tvalternatifterbaik.setText("Alternatif Terbaik = " +
alternatifrangking[0] + " dengan nilai terbesar = " +
Double.valueOf(hasilrangking[0]));
        tvalternatifterbaik.setTextColor(Color.parseColor("#0000ff"));
        tvalternatifterbaik.setTypeface(Typeface.DEFAULT_BOLD);
        //tvalternatifterbaik.setWidth(300);
        llhasil.addView(tvalternatifterbaik);

        hv.addView(ll);
        //this.setContentview(sv);
    }

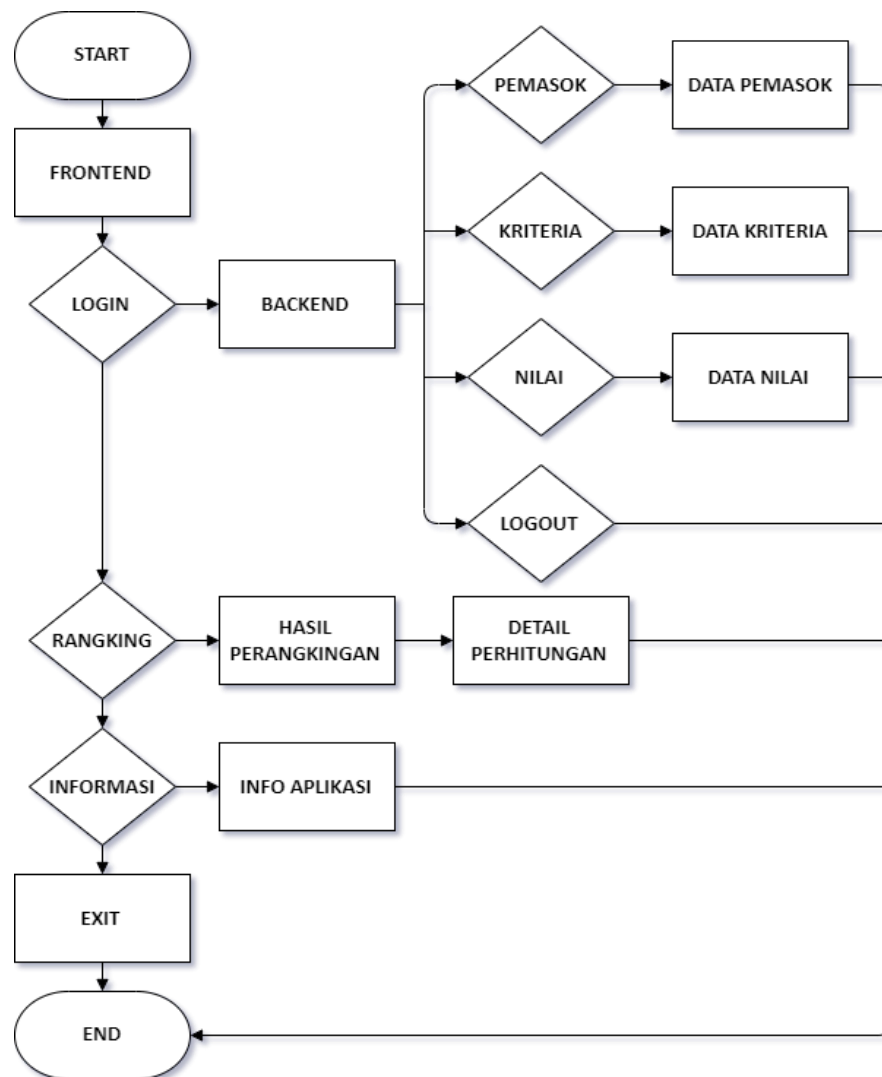
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.activity_analisa, menu);
        return true;
    }
}

```

4.4. Testing

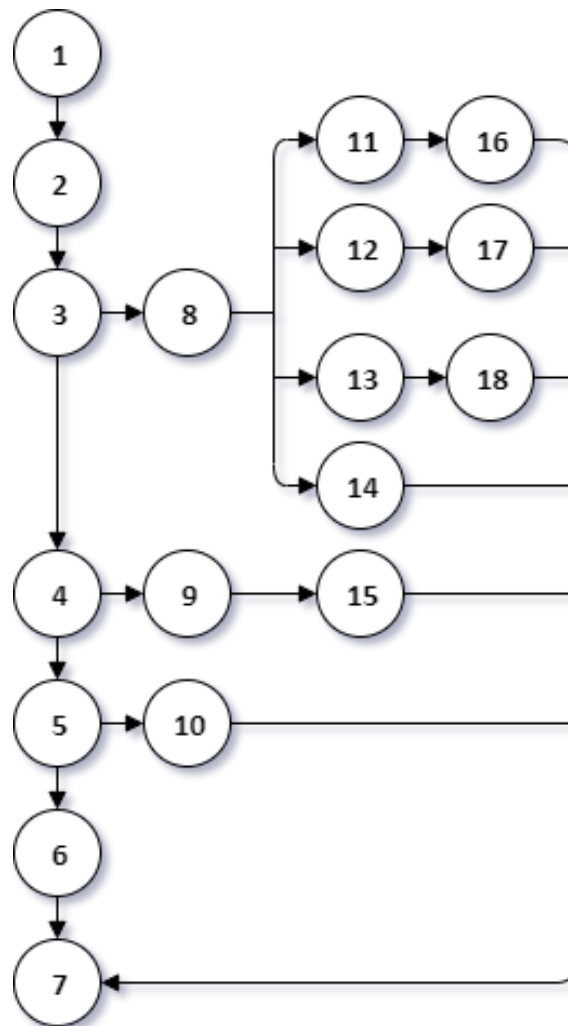
Proses pemeriksaan atau evaluasi komponen aplikasi untuk memverifikasi kebutuhan pengguna atau mengidentifikasi perbedaan-perbedaan antara hasil yang diharapkan dengan hasil yang terjadi. Aplikasi yang dikembangkan pada penelitian ini akan diujikan menggunakan *whitebox testing*.

Flowchart dari aplikasi pemasok konstruksi untuk analisa awal dari *whitebox testing* dapat dilihat pada Gambar IV.34.



Gambar IV.34. Flow Chart Aplikasi Seleksi Pemasok Konstruksi

Sedangkan kompleksitas siklomatis/*cyclomatic complexity* atau matrik perangkat lunak yang memberikan pengukuran kuantitatif terhadap kompleksitas logis untuk aplikasi seleksi pemasok konstruksi dapat digambarkan pada grafik alir/*flow graph* pada Gambar IV.35.



Gambar IV.35. Grafik Alir Aplikasi Seleksi pemasok konstruksi

Kompleksitas siklomatis dari grafik alir untuk *white box testing* dapat diperoleh dengan perhitungan berikut ini:

$$V(G) = E - N + 2$$

Dimana:

E = Jumlah *Edge* yang ditentukan gambar panah.

N = Jumlah simpul grafik alir ditentukan dengan gambar lingkaran.

$$V(G) = 20 - 18 + 2 = 4$$

$V(G) < 10$ berarti memenuhi syarat kekomplesitasi siklomatisnya.

Baris *set* yang dihasilkan dari jalur *independent* aplikasi seleksi pemasok konstruksi yaitu:

1. 1-2-3-4-5-6-7

2. 1-2-3-8-11-16-7

3. 1-2-3-8-12-17-7

4. 1-2-3-8-13-18-7

5. 1-2-3-8-14-7

6. 1-2-3-4-9-15-7

7. 1-2-3-4-5-10-12-13-14-7

8. 1-2-3-4-5-10-7

9. Ketika aplikasi dijalankan, maka terlihat bahwa satu set baris yang dihasilkan adalah 1-2-3-4-5-6-7-1-2-3-8-11-16-7-1-2-3-8-12-17-7-1-2-3-8-13-18-7-1-2-3-8-14-7-1-2-3-4-9-15-7-1-2-3-4-5-10-12-13-14-7-1-2-3-4-5-10-7 dan terlihat bahwa simpul telah dieksekusi satu kali.

4.5. Support

Perangkat yang digunakan untuk pemasangan aplikasi yang telah berhasil diujikan, serta disesuaikan dengan aplikasi yang dipilih oleh penulis.

4.5.1. Publikasi

1. Biaya Publikasi

- a. Biaya Registrasi *Google Play Store* : \$25 (Rp. 364.987,50,-)
- b. Biaya *Release* : Rp. 500.000,-
- c. Biaya *User Training* : Rp. 3.500.000,-

2. Jadwal Publikasi

Jadwal publikasi aplikasi seleksi pemasok konstruksi dapat dilihat pada Tabel IV.8.

Tabel IV.8.
Jadwal Publikasi Aplikasi

No.	Kegiatan Publikasi	Jadwal			
		Minggu 1	Minggu 2	Minggu 3	Minggu 4
1	Registrasi <i>Play Store</i> .				
2	Rilis & Sosialisasi Aplikasi.				
3	<i>User Training</i> Aplikasi.				

4.5.2. Spesifikasi *Hardware* dan *Software*

Menjelaskan tentang kebutuhan *hardware* dan *software* standard yang akan digunakan untuk dapat menjalankan sistem usulan.

Tabel IV.9.
Spesifikasi *Smartphone*

No.	Kebutuhan <i>Smartphone</i>	Keterangan
1	<i>Operating System</i>	Android OS, v5.1.1 (Lollipop)

2	CPU	Exynos 3475 Quad-Core 1.3 GHz Cortex-A7, GPU Mali-T720
3	RAM	1 GB
4	<i>Storage</i>	8 GB
5	Konektivitas	HSPA 42.2/5.76 Mbps, LTE Cat4 150/50 Mbps.
6	Layar	4.7 inches, 540x960 pixels.

4.6. Spesifikasi Dokumen Sistem Usulan

Parameter-parameter dokumen yang diusulkan pada aplikasi seleksi pemasok konstruksi yaitu:

Nama Dokumen : Hasil Perangkingan

Fungsi : Melihat dan memeriksa hasil perhitungan rangking.

Sumber : Adm. Logistik

Tujuan : Pengelola

Media : *Sheet*

Jumlah : 1 *page*

Frekuensi : Setiap diadakan proyek.

Bentuk : Lihat lampiran B.1

BAB V

PENUTUP

4.1. Kesimpulan

Pembahasan aplikasi seleksi pemasok konstruksi yang telah dilakukan untuk menjawab permasalahan yang telah dikaji dari skripsi yang dibahas maka dapat diuraikan kesimpulan sebagai berikut:

1. Adm. Logistik dimudahkan dalam memilih pemasok terbaik yang tepat dan dapat diandalkan terhadap proses konstruksi dari urutan *ranking* yang dihasilkan. Hal tersebut dikarenakan telah disediakan menu pengolahan data kriteria beserta bobot dan atributnya yang dapat ditentukan terlebih dahulu sesuai kebutuhan.
2. Pada proses pemilihan pemasok konstruksi diambil beberapa atribut dari beberapa referensi ilmiah dan digunakan metode *Weighted Product* yang sudah sesuai dengan kebutuhan perusahaan khususnya *Surveyor* dalam melakukan seleksi pemasok terbaik.
3. Aplikasi *mobile* seleksi pemasok konstruksi berhasil dibangun menggunakan bahasa pemrograman java dan xml, serta diimplementasikan pada *smartphone* dengan sistem operasi Android pada PT Binarthama Kharisma.

4.2. Saran

Berisi solusi mengenai kelemahan aplikasi yang dibuat yang dilihat dari aspek manajerial, teknis, dan penelitian selanjutnya. Agar aplikasi seleksi pemasok konstruksi dapat dikembangkan menjadi lebih baik lagi, maka terdapat beberapa saran yang dapat dipergunakan diantaranya:

1. Dalam membantu Adm. Konstruksi, diharapkan aplikasi seleksi pemasok konstruksi ini dapat terkoneksi secara daring atau terhubung dengan *server* perusahaan, sehingga dapat lebih memudahkan dalam pembuatan laporan ranking pemasok konstruksi.
2. Metode *Weighted Product* yang diterapkan lebih lanjut diharapkan dilengkapi atau digabungkan dengan metode lainnya. Hal tersebut memungkinkan pemeringkatan efektivitas maupun efisiensi yang lebih optimal dengan berbagai kriteria yang variatif.
3. Diharapkan dalam pengembangan selanjutnya, aplikasi yang dibangun untuk seleksi pemasok konstruksi ini tidak hanya untuk sistem operasi Android saja, namun juga dapat dijalankan dengan sistem operasi iOS pada *smartphone* iPhone.

DAFTAR PUSTAKA

- Abdulloh, R. (2017). *Membuat Toko Online dengan Teknik OOP, MVC, dan AJAX*. Jakarta: PT Elex Media Komputindo.
- Angrawati, D., Yamin, M., & Ransi, N. (2016). Sistem Pendukung Keputusan Menentukan Jumlah Beras Miskin Menggunakan *Metode Simple Additive Weight* (SAW). *semanTIK*, 2(1).
- Anto, A. G., Mustafidah, H., & Suyadi, A. (2015). Sistem Pendukung Keputusan Penilaian Kinerja Karyawan Menggunakan Metode SAW. *JUITA: Jurnal Informatika*, 3(4).
- Aprilia, S. P., & Rosyidi, C. N. (2018). Pemilihan Supplier di JOB Pertamina Hulu Energy-Jadestone Energy (Ogan Komering) Limited. Menggunakan Metode TOPSIS. In *Conference Proceedings of Seminar dan Konferensi Nasional IDEC (Surakarta, 2018)* (pp. 1-6).
- Azis, A., Utami, D. A. B., Umam, K., & Dzaki, I. N. (2018) *Bermain Android Studio Itu Mudah: Studi Kasus Pembuatan eM-Tilang*. Yogyakarta: Penerbit Deepublish.
- Dewi, S. M., Dobana, K., & Zacoeb, A. (2018). *Keandalan Struktur dan Infrastruktur*. Universitas Brawijaya Press.
- Faqih, H. (2014). Implementasi DSS dengan Metode SAW untuk Menentukan Prioritas Pekerjaan Operasi dan Pemeliharaan Sistem Irigasi DPU Kabupaten Tegal. *Bianglala Informatika*, 2(1).
- Fauzan, R., Indrasary, Y., & Muthia, N. (2018). Sistem Pendukung Keputusan Penerimaan Beasiswa Bidik Misi di POLIBAN dengan Metode SAW Berbasis Web. *Jurnal Online Informatika*, 2(2), 79-83.
- Fauzan, R., Fitri, R., & Fadliansyah, M. (2017). Sistem informasi penjurusan dan penerimaan peserta didik baru menggunakan metode weighted product. *Jurnal ELTIKOM*, 1(1), 11-22.
- Enterprise, J. (2015). *Mudah Membuat Aplikasi Absensi Dengan Java*. Jakarta: PT. Elex Media Komputindo.
- Jusak, J., & Sudarmaningtyas, P. (2018). Rancang Bangun Aplikasi Sistem Pendukung Keputusan Penentuan *Supplier* Obat Terbaik Menggunakan Metode *Weighted Product* pada Rumah Sakit Ibu dan Anak Putri Surabaya. *Jurnal JSIKA*, 7(2).
- Komputer, W. (2012). *Langkah Praktis Membangun Aplikasi Sederhana Platform Android*. Jakarta: PT Elex Media Komputindo.

- Kotler, P., Armstrong, G., Burton, S., & Deans, K. (2015). *Marketing* (9th ed.). French Forest: Pearson Australia.
- Maturidi, A. D. (2014). *Metode Penelitian Teknik Informatika*. Yogyakarta: Penerbit Deepublish.
- Mauliana, P., Wiguna, W., & Widyaman, D. (2018). Sistem Pendukung Keputusan Evaluasi Kinerja Pramuniaga Toserba Yogya Ciwalk Menggunakan Metode *Weighted Product*. *Infotronik: Jurnal Teknologi Informasi dan Elektronika*, 3(2), 85-94.
- Mirawati, M., Hikmah, A. B., & Wiguna, W. (2018). Sistem Penunjang Keputusan Penilaian Kinerja Kasir Lotte Mart Menggunakan Metode *Weighted Product*. *IJCIT (Indonesian Journal on Computer and Information Technology)*, 3(2).
- Mulyani, S. (2017). *Metode Analisis dan Perancangan Sistem* (2nd ed.). Bandung: Abdi Sistematika.
- Nofriansyah, D., & Defit, S. (2017). *Multi Criteria Decision Making (MCDM) pada Sistem Pendukung Keputusan*. Yogyakarta: Penerbit Deepublish.
- Nofriansyah, D. (2015). *Konsep Data Mining Vs Sistem Pendukung Keputusan*. Yogyakarta: Penerbit Deepublish.
- Purnomo, D. E. H., & Susetyo, A. E. (2016). Perancangan Sistem Pendukung Keputusan Untuk Pemilihan Pemasok *Nata De Coco* Dengan Metode *Weighted Product* (WP). *Tekinfo*, 4(2), 116-125.
- Sari, F. (2018). *Metode dalam Pengambilan Keputusan*. Yogyakarta: Penerbit Deepublish.
- Satyaputra, A. (2014). *Beginning Android Programming with ADT Bundle*. Jakarta: PT. Elex Media Komputindo.
- Solichin, A. (2016). *Pemrograman web dengan PHP dan MySQL*. Jakarta: Penerbit Budi Luhur.
- Supardi, I. Y. (2015). *Belajar coding android bagi pemula*. Jakarta: Elex Media Komputindo.
- Suryana, D. (2018). *Android Studio: Belajar Android Studio*. Bandung: Dayat Suryana Independent.
- Syaifudin, Y. W., Rozi, I. F., Mentari, M., & Lestari, V. A. (2018). *Dasar Pemograman: Dasar Pemograman* (Vol. 1). Malang: UPT Percetakan dan Penerbitan Polinema.

Utami, F. H., & Asnawati (2015). *Rekayasa Perangkat Lunak*. Yogyakarta: Penerbit Deepublish.

Wiguna, W., & Alawiyah, T. (2019). Sistem Reservasi Paket Wisata Pelayaran Menggunakan Mobile Commerce di Kota Bandung. *Jurnal VOI (Voice Of Informatics)*, 8(2).

Yudhanto, Y., & Wijayanto, A. (2018). *Mudah Membuat dan Berbisnis Aplikasi Android dengan Android Studio*. Jakarta: PT. Elex Media Komputindo.