BAB II

LANDASAN TEORI

2.1 Keputusan

Keputusan merupakan hasil dari proses memilih pilihan terbaik diantara beberapa alternatif yang telah tersedia. Pada proses pengambilan keputusan, kita akan berusaha mencurahkan segala pemikiran dan melakukan kegiatan yang diperlukan untuk mendapatkan pilihan yang terbaik. Kegiatan yang diperlukan adalah mengumpulkan data dan informasi yang diperlukan serta menentukan metode pengambilan keputusan yang akan digunakan sebagai dasar untuk mengambil keputusan.(Diana, S.Si, 2018)

Proses Pengambilan Keputusan dapat di pandang sebagai suatu sistem.Komponen sistem terdiri dari masukan,proses dan keluaran.

1. Masukan (Input)

   Masukan dalam proses pengambilan keputusan adalah data dan informasi.Data dapat berupa suatu keadaan,gambar,suara,huruf,angka atau bahasa yang dapat digunakan sebagai bahan untuk melihat lingkungan,objek,kejadian ataupun suatu konsep.Data ini masih memerlukan pengolahan data agar menjadi informasi yang lebih berdaya guna dan hasil pengolahan data dinamakan informasi.

2. Proses

   Proses pengambilan keputusan merupakan langkah yang diambil oleh seorang pengambil keputusan untuk mendapatka keputusan yang terbaik.

3. Keluaran (Output)

Keluaran dari proses pengambilan keputusan adalah keputusan yang dipilih oleh seorang pengambil keputusan,dimana keputusan ini tentunya merupakan keputusan terbaik.

Keputusan yang di ambil untuk menyelesaikan suatu masalah dapat dilihat dari ke terstrukturannya yang bisa di bagi menjadi.(Dini MH Hutagalung, 2017)

1. Keputusan terstruktur *(structured decision)*

Keputusan terstuktur adalah keputusan yang dilakukan secara berulang-ulang dan bersifat rutin, prosedur pengambilan keputusan sangatlah jelas, keputusan tersebut terutama dilakukan pada menajemen tingkat bawah.

2. Keputusan semiterstruktur *(semistructured decision)*

Keputusan semiterstruktur adalah keputusan yang memiliki dua sifat, sebagian sifat bisa ditangani oleh komputer dan yang lain tetap harus dilakukan oleh pengambil keputusan, prosedur dalam pengambil keputusan tersebut secara garis besar sudah ada, tetapi ada beberapa hal yang masih memerlukan kebijakan dari pengambil keputusan. Biasanya, keputusan semacam ini di ambil oleh manajer level menengah dalam suatu organisasi.

3. Keputusan tak terstruktur *(unstructured decision)*

Keputusan tak terstruktur adalah keputusan yang penanganannya rumit karena tidak terjadi beulang-ulang atau tidak selalu terjadi, keputusan tersebut menuntut pengalaman dan berbagai sumber yang bersifat eksternal. Keputusan tersebut umumnya terjadi pada manajemen tingkat atas.

Jika mengamati proses keluarnya suatu keputusan, maka dapat di catat tahap-tahap penting dalam proses tersebut, yaitu :

1. Indentifikasi masalah
2. Pemilihan model pemecahan masalah
3. Pengumpulan data yang dibutuhkan untuk melaksakan model keputusan tersebut
4. Mengimplementasikan model tersebut
5. Mengevaluasi sisi positif dari setiap alternatif yang ada
6. Melaksanakan solusi terpilih

2.2 Sistem Pendukung Keputusan

Sistem Pendukung Keputusan (SPK) dapat didefinisikan sebagai suatu program komputer yang menyediakan informasi dalam domain aplikasi yang diberikan oleh suatu model analisis keputusan dan akses ke database, dimana hal ini ditujukan untuk mendukung pembuat keputusan (decision maker) dalam mengambil keputusan secara efektif baik dalam kondisi yang kompleks dan tidak terstruktur. (Burhannudin & Dini, 2017)

Sistem pengambilan keputusan merupakan bagian yang tak terpisahkan dari totalitas sistem organisasi keseluruhan. Bahwa sistem oraganisasi paling tidak mencakup sistem fisik (sistem operasional), sistem manajemen (sistem keputusan), dan system informasi.(Burhannudin & Dini, 2017)

Kebutuhan akan informasi yang akurat,kebutuhan akan informasi yang terbaru dan up to date,penyediaan informasi yang tepat wakt,pengurangan biaya,adanya kebutuhan tentang sistem yang mudah digunakan karena adanya perubahan

perilaku pengguna akhir (end user) merupakan alasan-alasan yang membuat sistem pendukung keputusan merupakan sistem yang dibutuhkan.Karena sistem pendukung keputsan harus memenuhi semua kebutuhan diatas untuk membantu pengambilan keputusan.

Tujuan implementasi sistem pendukung keputusan antara lain :

1. Sistem pendukung keputusan berbasis komputer dapat memungkinkan para pengamil keputusan untuk mengambil keputusan dalam waktu yang cepat karena dukungan sistem yang dapat memproses data dengan cepat dan dalam jumlah yang banyak.
2. Sistem pendukung keputusan ini dimaksudkan untuk membantu manajer dalam mengambil keputusan bukan mengantikan tugas manajer sehingga dengan dukungan data, informasi yang akurat diharapkan manajer dapat membuat keputusan yang lebih akurat dan berkualitas.
3. Menghasilkan keputusan yang efektif (sesuai tujuan) dan efisien dalam hal waktu; Tujuan pengembangan sisitem ini adalah untuk efisiensi, peningkatan kinerja dan peningkatan kualitas informasi. Terdapat 2 jenis efisiensi yang diperoleh, yakni efisiensi biaya dan efisiensi sumber daya. Efisiensi biaya dilakukan dengan memperoleh dengan mengoptimumkan keuntungan dengan biaya minimum, sedangkan efisiensi sumber daya dilakukan dengan pemanfaatan sumber daya semaksimal mungkin.
4. Meningkatkan tingkat pengendalian guna meningkatkan kemampuan untuk mendeteksiadanya kesalahan-kesalahan pada suatu sistem sehingga dapat dilakukan asntisipasi kesalahan.

5. Menghasilkan keputusan yang berkualitas karena keputusan yang diambil didasarkan pada data yang lengkap dan akurat. Peningktan pelayanan oleh suatu sistem pendukung keputusan untuk menghsilkan keputusan yang berkualitas.

2.3 Metode *Simple Additive Weight* (SAW)

Metode SAW (*Simple Additive Weighting*) merupakan salah satu metode dalam pengambilan keputusan multi kriteria yang sederhana dan klasik.Metode ini termasuk dalam metode pembobotan atau dikenal sebagai metode penjumlahan terbobot.Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut.(Diana, S.Si, 2018)

Adapun langkah-langkah dalam metode SAW adalah sebagi berikut :

1. Menetukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan,$C_{j,}$ j = 1,2,..., m.
2. Menentukan Bobot untuk masing-masing kriteria W_j, j = 1,2,....,m dengan catatan penting $\sum Wj = 1$
3. Melakukan normalisasi matriks keputusan dengan melakukan proses perbandingan pada semua nilai alternative yang ada, rumus normalisasi adalah

$$rij = \begin{cases} \frac{Xijj}{maxij}, jika\ j\ adalah\ atribut\ keuntungan\ (benfit) \\ \frac{minxij}{xijj}, jika\ j\ adalah\ atribut\ biaya\ (cost) \end{cases}$$

Keterangan :

Rij : Nilai rating kinerja ternormalisasi

Xij : Nilai atribut yang dimiliki dari setiap kriteria

Max Xij: Nilai terbesar dari setiap kriteria

Min Xij : Nilai terkecil dari setiap kriteria

Benefit : Jika nilai terbesar adalah terbaik

Cost : Jika nilai terkecil adalah terbaik

4. Menghitung nilai preferensi untuk tiap alternatif, V_i, diberikan sebagai

$$Vi = \sum \frac{n}{j} = 1\, Wj * rij$$

Keterangan :

V_i : Rangking untuk setiap alternatif

W_j : Nilai bobot dari setiap kriteria

5. Nilai V_i yang lebih besar mengindikasikan bahwa alternatif A_i lebit terpilih.

2.4 Metode *Weighted Product*

Metode WPM Menggunakan perkalian untuk menghubungkan rating atribut,dimana rating atribut harus dipangkatkan dulu dengan bobot atribut yang bersangkutan. Proses ini sama halnya dengan proses normalisasi.(Diana, S.Si, 2018)

Langkah-langkah pada metode WPM adalah sebagi berikut :

1. Menetukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan, C_j, J = 1,2,....,n.
2. Menetukan bobot awal untuk masing-masing kriteria.Nilai bobot awal (W) digunakan untuk menunjukan tingkat kepentingan relative dari setiap kriteia. Nilai bobot awal ini ditentukan oleh pengambil keputusan yang menentukan tingkat kepentingan relative setiap kriteria.Ada beberapa cara yang bisa dilakukan untuk menentukan bobot awal ini antara lain :
   a. Dengan memebrikan nilai parameter untuk setiap kriteia.

b. Memberikan bobot antara 0-100 yang berarti tingkat kepentingan setiap kriteria. Melakukan normalisasi nilai bobot awal dengan membagi setiap nilai w0 dengan total nilai wj.Normalisasi atau perbaikan bobot ini menghasikan nilai nomalisasi wj = 1 dimana j = 1,2,....,n adalah banyak alternative dan $\sum wj$ adalah jumlah keseluruhan nilai bobot.Terdapat 2 sifat yang dimiliki oleh bobot awal berdasarkan pada sifat masing-masing kriteria yaitu keuntungan (benefit) dan biaya (cost).Untuk mencapai solusi ideal,kriteria yang memeliki sifat benefit nilainya akan dimaksimumkan (bernilai positif) sedangkan kriteria yang bersifat cost nilainya akan di minimumkan (bernilai Negatif). Normalisasi $Wj = \frac{wj}{\sum wj}$

3. Menentukan nilai vektor (S)

   $Sj = \prod_j^n = 1^x ij^{wj},\ i = 1,2, ...$

   Keterangan :

   Sj = Preferensi alternatif ke j dianalogkann dengan vektor S

   Xij = Nilai setiap alternatif yang dimiliki dari setiap kriteria

   n = Banyak Kriteria

   Wj = Hasil normalisasi nilai bobot awal

   Nilai vektor (S) ini diperoleh dengan cara memangkatkan nilai atibut yang dimiliki setiap kriteria dengan hasil normalisasi bobot yang berpangkat posistif untuk kriteria keuntungan (benefit) dan yang berpangkat negatif untuk kriteria (cost).

4. Menentukan Nilai vektor (V)

$Vj = \frac{\prod_{j}^{n} = 1^{x}ij^{wj}}{\prod_{j}^{n} = 1^{(x}ij*)^{wj},}$ Vektor V merupakan preferensi alternatif yang akan digunakan untuk perangkingan dengan cara membagi masing-masing junlah nilai vektor S dengan jumlah seluruh vektor S.(Diana, S.Si, 2018)

2.5 *Unified Modelling Language* (UML)

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.(Dharwiyanti & Wahono, 2003)

Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan class dan operation dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasabahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C (Dharwiyanti & Wahono, 2003).

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan syntax/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (Object-Oriented Design), Jim Rumbaugh

OMT (Object Modeling Technique), dan Ivar Jacobson OOSE (Object-Oriented Software Engineering).(Dharwiyanti & Wahono, 2003)

Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi booch, metodologi coad, metodologi OOSE,metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock, dsb. Masa itu terkenal dengan masa perang metodologi (method war) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan (Dharwiyanti & Wahono, 2003).

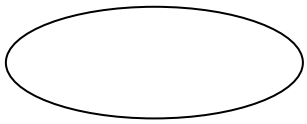
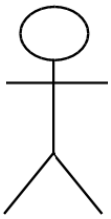
2.5.1 Use Case Diagram

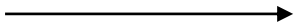
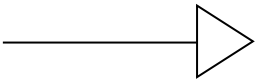
Use Case dan use case specification Use case adalah deskripsi fungsi dari sebuah sistem perspektif pengguna.Use case bekerja dengan cara mendeskripsikan tipikal interaksi antara user (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut skenario (Haviluddin, 2011).

Use case merupakan awal yang sangat baik untuk setiap fase pengembangan berbasis objek, design, testing, dan dokumentasi yang menggambarkan kebutuhan sistem dari sudut pandang di luar sistem (Haviluddin, 2011).

Perlu diingat bahwa use case hanya menetapkan apa yang seharusnya dikerjakan oleh sistem, yaitu kebutuhan fungsional sistem dan tidak untuk menentukan kebutuhan non-fungsional, misalnya: sasaran kinerja,bahasa pemrograman dan lain sebagainya (Haviluddin, 2011).Simbol-simbol yang digunakan untuk membuat use case diagram dapat dilihat pada tabel 2.1

Tabel 2.1 Simbol-simbol *Use Case Diagram*

Gambar	**Keterangan**
Use Case 	Use Case menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktir, yang dinyatakan dengan menggunakan kata kerja
Actor 	Actor atau Aktor adalah Abstraction dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifika sikan aktir, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan Use Case, tetapi tidak memiliki kontrol terhadap use case
Asosiasi	Asosiasi antara aktor dan use case,digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasika n data.

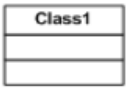
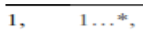
Include 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *case* ini. *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan.
Generalisasi 	Hubungan generalisasi spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.

2.5.2 Class Diagram

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku system (Hendini, 2016).

Class Diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan. Class Diagram secara khas meliputi : Kelas (Class), Relasi Assosiations, Generalitation dan Aggregation, attribut (Attributes), operasi (operation/method) dan visibility, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan Multiplicity atau Cardinality.Berikut simbol-simbol Class Diagram (Hendini, 2016). Simbol-simbol yang digunakan untuk membuat class diagram dapat dilihat pada tabel 2.2

Tabel 2.2 Simbol-simbol *Class Diagram*

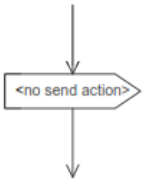
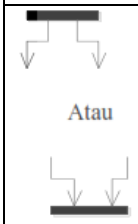
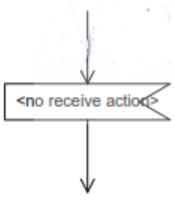
Simbol	**Deskripsi**
Class 	Menggambarkan sesuatu yang menyimpan informasi di *class* menampung nama *class*, atribut, dan *method.*
Package 	Berfungsi untuk mengelompokan kelas-kelas memiliki persamaan.
Asosiasi 	Asosiasi yang menghubungkan class dengan class Multiplycity.
Boundary Class 	*Boundary Class* menggambarkan *class* yang menjadi antar muka aktor dengan sistem.
Control Class 	*Control Class* menggambarkan *class* yang menjadi control atau perantara antar *class* dengan *database.*
Aggregation 	*Aggregation* menggambarkan suatu *class* terdiri dari *class* lain atau suatu *class* adalah bagian dari *class* lain.
Generalization 	*Generalization* merupakan sebuah *taxonomic relationship* antara *class* yang elbih umum dengan *class* yang lebih khusus.

2.5.3 Activity Diagram

Activity Diagram menggambarkan work flow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa

diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas dapat dilakukan oleh sistem. Simbol-simbol yang digunakan dalam activity diagram sebagai berikut (Pt & Rent, 2018). Simbol-simbol yang digunakan untuk membuat activity diagram dapat dilihat pada tabel 2.3

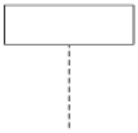
Tabel 2.3 Simbol-simbol *Activity Diagram*

Simbol	**Deskripsi**
	*Initial state* adalah sebuah diagram aktivitas yang memiliki sebuah status awal
	*Final state* adalah sebuah diagram aktivitas yang memiliki sebuah status akhir
	*Activity* adalah aktivitas yang dilakukan sistem diawali dengan kata kerja
	*Decision* adalah asosiasi percabangan jika ada pilihan aktivitas lebih dari satu
	Tanda Pengiriman
	Fork digunakan untuk menunjukkan kegiatan yang dilaksanakan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
	Tanda Penerimaan

2.5.4 Sequence Diagram

Sequence Diagram adalah tool yang sangat populer dalam pengembangan sistem informasi secara object-oriented untuk menampilkan interaksi antar objek.”Berdasarkan definisi tersebut, dapat disimpulkan bahwa Sequence Diagram adalah tool yang digunakan dalam pengembangan sistem.Berikut simbol-simbol dari Sequence Diagram (Pt & Rent, 2018). Simbol-simbol yang digunakan untuk membuat Sequence diagram dapat dilihat pada tabel 2.4

Tabel 2.4 Simbol-simbol *Sequence Diagram*

Simbol	**Deskrispsi**
Object Life Line 	Menyatakan objek yang berinteraksi pesan dan menyatakan kehidupan suatu objek
Activation 	Menyatakan suatu objek dalam keadaan aktif dan berinteraksi pesan
Messaage 	Menyatakan suatu objek mengirimkan data atau masukan atau informasi ke objek lainnya. Arah pada mengarah pada objek yang dikirim.
Message (call) 	Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau diri sendiri
Message (return) 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu

2.6 Database

Database adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Database atau basis data dapat dibayangkan ibarat sebuah lemari basis data (Alhadi Saputra, 2012).

Database Management System (DBMS) atau dalam bahasa Indonesia disebut dengan Manajemen Basis Data adalah perangkat lunak yang dirancang untuk mengelola dan memanggil kueri (query) basis data (Alhadi Saputra, 2012).

DBMS adalah perangkat lunak (Software) yang berfungsi untuk mengelola database, mulai dari membuat database itu sendiri, sampai dengan proses-proses yang berlaku dalam database tersebut, baik berupa entry, edit, hapus query terhadap data, membuat laporan dan lain sebagainya secara efektif dan efisien (Alhadi Saputra, 2012).

2.7 Visual Basic.NET

Visual Basic .NET (atau VB.NET) merupakan salah satu bahasa pemrograman yang bisa digunakan untuk membangun aplikasi-aplikasi.NET di platform Microsoft .NET. Tidak seperti generasi sebelumnya Visual Basic versi 6.0 ke bawah yang lebih difokuskan untuk pengembangan aplikasi desktop, Visual Basic .NET memungkinkan para pengembang membangun bermacam aplikasi, baik desktop maupun aplikasi web. Seiring dengan perkembangan aplikasi perangkat lunak yang semakin kompleks (S.Thya Safitri & PriyantoAgus, 2016).

2.8 Microsoft Access

Microsoft Access (atau Microsoft Office Access) adalah sebuah program aplikasi basis data komputer relasional yang ditujukan untuk kalangan rumahan dan perusahaan kecil hingga menengah. Aplikasi ini merupakan anggota dari beberapa aplikasi Microsoft Office.(Sutan Mohammad Arif, 2018)

Aplikasi ini menggunakan Microsoft Jet Database Engine dan juga menggunakan tampilan grafis yang intuitif sehingga memudahkan pengguna.Komponen utama (object) dari Microsoft Access adalah sebagi berikut:

1) tabel yang berfungsi sebagai tempat menyimpan sekumpulan data sejenis;
2) query berfungsi sebagai bahasa atau sintaks untuk melakukan manipulasi terhadap database;
3) form berfungsi untuk memasukkan dan mengubah data/informasi yang ada dalam suatu database dengan menggunakan tampilan formulir. Ini memudahkan pemasukan data, dan menghindari kesalahan pemasukan data;
4) report berfungsi untuk menampilkan, mencetak data/informasi dalam bentuk laporan..(Sutan Mohammad Arif, 2018)