BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pendukung Keputusan

Sistem pendukung keputusan (SPK) adalah sistem berbasis komputer yang membantu para pengambil keputusan mengatasi berbagai masalah melalui interaksi Pengambilan keputusan (*Decision Making*) adalah melakukan penilaian dan menjatuhkan pilihan. Keputusan ini diambil setelah melalui beberapa perhitungan dan pertimbangan alternatif. *Decision support system* atau sistem pendukung keputusan secara umum didefinisikan sebagai sebuah sistem yang mampu memberikan kemampuan baik kemampuan pemecahan masalah maupun kemampuan pengkomunikasian untuk masalah semi terstruktur. Secara khusus, Sistem pendukung keputusan didefinisikan sebagai sebuah sistem yang mendukung kerja seorang manager maupun sekelompok manager dalam memecahkan masalah semi terstruktur dengan cara memberikan informasi ataupun usulan menuju pada keputusan tertentu (Muhammad Yudin Ritonga ; 2014 : 9).

Sistem pendukung keputusan (SPK) biasanya dibangun untuk mendukung solusi atas suatu masalah atau untuk suatu peluang. Aplikasi sistem pendukung keputusan (SPK) digunakan dalam pengambilan keputusan. Aplikasi sistem pendukung keputusan (SPK) menggunakan CBIS (*Computer Based Information System*) yang fleksibel, interaktif, dan dapat diadaptasi, yang dikembangkan untuk mendukung solusi atas masalah manajemen spesifik yang tidak terstruktur. Sistem

pendukung keputusan sebagai sistem berbasis komputer yang terdiri dari tiga komponen yang saling berinteraksi, sistem bahasa (mekanisme untuk memberikan komunikasi antara pengguna dan komponen sistem pendukung keputusan lain), sistem pengetahuan (respositori pengetahuan domain masalah yang ada pada sistem pendukung keputusan atau sebagai data atau sebagai prosedur) dan sistem pemrosesan atau lebih kapabilitas manipulasi masalah umum yang diperlukan untuk pengambilan keputusan (Dicky Nofriansyah ; 2014 : 1).

II.1.1. Karakteristik Sistem Pendukung Keputusan

Karakteristik dan kapabilitas sistem pendukung keputusan antara lain :

1. Dukungan untuk pengambilan keputusan, terutama pada situasi semi terstruktur dan tak terstruktur, dengan menyertakan penilaian manusia dan informasi terkomputerisasi. Masalah-masalah tersebut tidak dapat dipecahkan oleh sistem komputer lain atau oleh metode atau alat kuantitatif standar.
2. Dukungan untuk semua level manajerial, dari eksekutif puncak sampai manajer lini.
3. Dukungan untuk individu dan kelompok. Masalah yang kurang terstruktur sering memerlukan keterlibatan individu dari departemen dan tingkat organisasional yang berbeda atau bahkan dari organisasi lain. DSS mendukung im virtual melalui alat-alat Web kolaboratif.
4. Dukungan untuk keputusan independen dan atau sekuensial. Keputusan dapat dibuat satu kali, beberapa kali, atau berulang (dalam interval yang sama).

5. Dukungan disemua fase proses pengambilan keputusan : intelegensi, desain, pilihan, dan implementasi
6. Dukungan diberbagai proses dan gaya pengambilan keputusan.
7. Adaptivitas sepanjang waktu. Pengambil keputusan seharusnya reaktif, dapat menghadapi perubahan kondisi secara cepat, dan dapat mengadaptasi DSS untuk memenuhi perubahan tersebut. DSS bersifat fleksibel dan karena itu pengguna dapat menambahkan, menghapus, menggabungkan, mengubah, atau menyusun kembali elemen-elemen dasar, DSS juga fleksibel dalam hal dapat dimodifikasi untuk memecahkan masalah lain yang sejenis.
8. Peningkatan terhadap keefektifan pengambilan keputusan (akurasi, *timeliness*, kualitas).
9. Kontrol penuh oleh pengambil keputusan terhadap semua langkah proses pengambilan keputusan dalam memecahkan suatu masalah. DSS secara khusus menekankan untuk mendukung pengambil keputusan, bukannya menggantikan (Sri Hartati ; 2011 : 37).

II.2. Penilaian Kinerja

Organisasi juga membantu para karyawan dalam memperbaiki kinerja mereka dengan memberikan umpan baik tentang kinerja mereka di waktu sebelumnya. Penilaian kinerja (*performance appraisal*) adalah sebuah evaluasi terhadap kinerja dari pekerjaan karyawan, dengan cara membandingkan antara hasil *actual* dengan hasil yang diinginkan. Berdasarkan hasil evaluasi ini, manajer membuat keputusan objektif tentang kompensasi, promosi, kebutuhan pelatihan

tambahan, transfer, dan pemutusan hubungan kerja. Memberikan penilaian terhadap kinerja para karyawan serta mengkomunikasikan persepsi tentang kekuatan dan kelemahan mereka adalah elemen penting dalam meningkatkan produktivitas maupun laba perusahaan. Penilaian kinerja tidak hanya terbatas dalam dunia bisnis. Agen pemerintah, organisasi nirlaba, dan institusi pendidikan juga melaksanakan penilaian kinerja.

Beberapa perusahaan melaksanakan penilaian terhadap rekan sekerjanya, di mana karyawan memberikan penilaian terhadap kinerja rekan kerjanya, sementara perusahaan lain memberikan kesempatan kepada karyawan untuk memberikan penilaian terhadap para pengawas (*supervisor*) dan manajer. Salah satu jenis metode penilaian kinerja adalah penilaian kinerja 360 derajat, yaitu proses yang mengumpulkan umpan baik dari sebuah panel penilaian yang terdiri atas 8 sampai 12 orang, termasuk rekan sekerja, pengawas, anggota tim kerja, bawahan, dan kadang-kadang pelanggan. Idenya adalah mendapatkan umpan balik sebanyak-banyaknya dari berbagai perspektif. Bagaimanapun, pendekatan penilaian kinerja ini cenderung untuk menambah pekerjaan, baik bagi karyawan maupun manajer, setiap orang harus memberikan penilaian terhadap 20 orang atau lebih dan juga harus menghadapi tumpukan dokumen pekerjaan. Juga, karena evaluasi yang dilakukan tidak mencatumkan identitas, staf yang terlibat, yang memiliki itikad buruk dapat menggunakannya untuk mengambil keuntungan dalam perselisihan pribadi (Boone ; 2012 : 432).

II.3. Metode *TOPSIS*

TOPSIS adalah salah satu metode pengambilan keputusan multikriteria yang pertama kali diperkenalkan oleh Yoon dan Hwang (1981). TOPSIS menggunakan prinsip bahwa alternatif yang terpilih harus mempunyai jarak terdekat dari solusi ideal positif dan terjauh dari solusi ideal negatif dari sudut pandang geometris dengan menggunakan jarak Euclidean untuk menentukan kedekatan relatif dari suatu alternatif dengan solusi optimal. Solusi ideal positif didefinisikan sebagai jumlah dari seluruh nilai terbaik yang dapat dicapai untuk setiap atribut, sedangkan solusi negatif-ideal terdiri dari seluruh nilai terburuk yang dicapai untuk setiap atribut. TOPSIS mempertimbangkan keduanya, jarak terhadap solusi ideal positif dan jarak terhadap solusi ideal negatif dengan mengambil kedekatan relatif terhadap solusi ideal positif. Berdasarkan perbandingan terhadap jarak relatifnya, susunan prioritas alternatif bisa dicapai. Metode ini banyak digunakan untuk menyelesaikan pengambilan keputusan. Hal ini disebabkan konsepnya sederhana, mudah dipahami, komputasinya efisien, dan memiliki kemampuan mengukur kinerja relatif dari alternatif-alternatif keputusan.

Dalam melakukan perhitungan metode TOPSIS terdapat beberapa langkah-langkah yang harus diperhatikan, yaitu :

1. Membangun *normalized decision matrix* Elemen *rij* hasil dari normalisasi *decision matrix R* dengan metode *Euclidean length of a vector* adalah :

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

Dimana :

r_{ij} = hasil dari normalisasi matriks keputusan R
i = 1,2,3,...,m;
j = 1,2,3,...,n;

2. Membangun *weighted normalized decision matrix* Dengan bobot *W* = (*w1*,*w2*,....,*wn*), maka normalisasi bobot matriks *V* adalah :

$$V = \begin{bmatrix} w_{11}r_{11} & \cdots & w_{1n}r_{1n} \\ \vdots & \ddots & \vdots \\ w_{m1}r_{m1} & \cdots & w_{nm}r_{nm} \end{bmatrix}$$

3. Menentukan solusi ideal positif dan solusi ideal negatif Solusi ideal positif dinotasikan dengan A+ dan solusi ideal negatif dinotasikan dengan A-, sebagai berikut :

Menentukan solusi ideal (+) dan (-)

$$A^{+} = \left\{\left(\max v_{ij}\right)\left(\min v_{ij} \mid j \in J'\right), i = 1,2,3,...m\right\} = \left\{v_1^{+}, v_2^{+}, \cdots v_m^{+}\right\}$$
$$A^{-} = \left\{\left(\max v_{ij}\right)\left(\min v_{ij} \mid j \in J'\right), i = 1,2,3,...m\right\} = \left\{v_1^{-}, v_2^{-}, \cdots v_m^{-}\right\}$$

Dimana :

v_{ij} = elemen matriks *V* baris ke-*i* dan kolom ke-*j*
J = {j=1,2,3,...,n dan j berhubungan dengan *benefit criteria*}
J' = {j=1,2,3,...,n dan j berhubungan dengan *cost criteria*}

4. Menghitung separasi *Separation measure* ini merupakan pengukuran jarak dari suatu alternatif ke solusi ideal positif dan solusi ideal negatif. Perhitungan matematisnya adalah sebagai berikut :

Separation measure untuk solusi ideal positif

$$S_i^+ = \sqrt{\sum_{j=1}^{n}\left(v_{ij} - v_j^+\right)^2}$$, dengan i = 1,2,3,...,m

Dimana :

J = {j=1,2,3,...,n dan j merupakan *benefit criteria*}
J' = {j=1,2,3,...,n dan j merupakan *cost criteria*}

5. Menghitung kedekatan relatif terhadap solusi ideal Kedekatan relatif dari alternatif A^+ dengan solusi ideal A^- direpresentasikan dengan :

$$C_i = \frac{S_i^-}{S_i^- + S_i^+}$$, dengan $0 < C_i^+ < 1$ dan i = 1,2,3,...,m

6. Merangking alternatif Alternatif dapat diranking berdasarkan urutan C_i*.

Maka dari itu, alternatif terbaik adalah salah satu yang berjarak terpendek terhadap solusi ideal dan berjarak terjauh dengan solusi ideal negatif (Desi Leha Kurniasih ; 2013 : 8).

II.4. Pengertian Netbeans

NetBeans merupakan salah satu proyek *open source* yang disponsori oleh *Sun Microsystem*. Proyek ini bediri pada tahun 2000 dan telah menghasilkan 2 produk, yaitu *NetBeans* IDE dan *NetBeans Platform*. *NetBeans* IDE merupakan produk yang digunakan untuk melakukan pemrograman baik menulis kode, meng-*compile*, mencari kesalahan dan mendistribusikan program. Sedangkan *NetBeans Platform* adalah sebuah modul yang merupakan kerangka awal / pondasi dalam bangun aplikasi desktop yang besar.

NetBeans juga menyediakan paket yang lengkap dalam pemrograman dari pemrograman standar (aplikasi desktop), pemrograman enterprise, dan pemrograman perangkat mobile. Saat ini *NetBeans* telah mencapai versi 6.8 (Wahana Komputer ; 2010 : 15).

II.4. Pengertian *Database*

Database adalah sekompulan data mentah yang disusun menurut logika tertentu dan terorganisasi dalam bentuk yang dapat disimpan dan diproses oleh komputer. contoh *database* dapat berisi data pegawai, data penjualan, pembayaran, dan lain-lain. data internal dari akunting, keuangan, penjualan dan bidang-bidang bisnis lainnya yang disimpan dalam suatu sistem komputer dan disusun menurut logika tertentu disebut sebagai internal *database*. *Database* seringkali disimpan dalam suatu perangkat tertentu pada komputer, seperti hard disk, compact disk, dan sebagainya. hubungan antarsistem *database* dan sistem software sangat kuat karena sistem *database* yang dipakaiu sangat menentukan kemudahan aksesnya data sementara software sendiri memungkinkan peneliti memanipulasi data untuk dianalisis (Dermawan Wibisono ; 2012 : 129).

II.4.1. Teknik Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi

dari tabel rasional. Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang tlkah ditemukan yaitu :

1. Bentuk normal tahap pertama (1" Normal Form)

Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri.

2. Bentuk normal tahap kedua (2^{nd} normal form)

Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.

3. Bentuk normal tahap ketiga (3^{rd} normal form)

Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

4. *Boyce Code Normal Form* (BCNF)

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi

berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

5. Bentuk Normal Tahap Keempat dan Kelima

Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD). Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata ; 2010 : 76).

II.5. Pengertian MySQL

Mysql pertama kali dirintis oleh seorang programmer database bernama Michael Widenius, yang dapat anda hubungi di emailnya monty@analytikerna. Mysql *database server* adalah RDBMS (*Relasional Database Management System*) yang dapat menangani data yang bervolume besar. meskipun begitu, tidak menuntut *resource* yang besar. Mysql adalah *database* yang paling populer di antara *database* yang lain.

MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan *multi user*. MySQL memiliki dua bentuk lisensi, yaitu *free software* dan *shareware*. penulis sendiri dalam menjelaskan buku ini

menggunakan *database* ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensi, yang berada di bawah lisensi GNU/GPL (*general public license*) (Wahana Komputer; 2010 : 5).

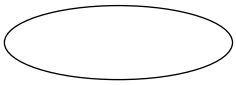
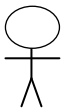
II.6. UML (*Unified Modeling Language*)

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.2. Simbol *Use Case*

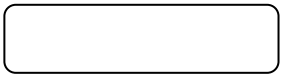
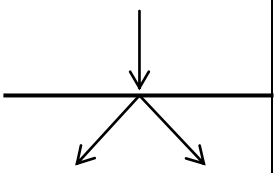
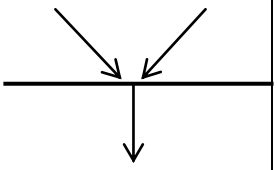
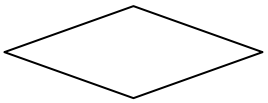
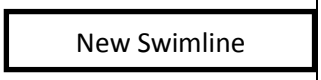
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, 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 control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Windu Gata ; 2013 : 4)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel II.3. Simbol *Activity Diagram*

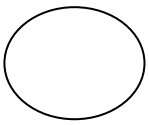
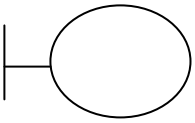
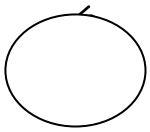
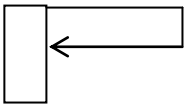
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*.
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Windu Gata ; 2013 : 6)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class,* berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive,* menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata ; 2013 : 7)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem. *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, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.5. *Multiplicity Class Diagram*

Multiplicity	**Penjelasan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata ; 2013 : 9)