BAB II
TINJAUAN PUSTAKA

2.1 Teori Yang Berhubungan Dengan Sistem Secara Umum

Teori yang berhubungan dengan sistem meliputi sistem dan data.

2.1.1 Sistem

Sistem dapat didefinisikan sebagai sekumpulan prosedur yang saling berkaitan dan saling terhubung untuk melakukan suatu tugas bersama-sama (Pratama, 2014:7).

Sistem adalah kumpulan dari obyek-obyek seperti orang, *resources*, konsep dan prosedur yang ditujukan untuk melakukan fungsi tertentu atau memenuhi suatu tujuan. Kemudian sistem juga merupakan kumpulan dari komponen yang berinteraksi bersama-sama secara kolektif untuk melaksanakan tugas (Pratiwi, 2016:4).

Berdasarkan uraian tersebut maka dapat disimpulkan bahwa pengertian sistem adalah sekumpulan komponen atau elemen yang dapat diartikan sebagai suatu kumpulan atau obyek-obyek yang terhubung satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan.

2.1.2 Data

Mengenai pengertian data, John J. Longkutoy dalam bukunya *Pengenalan Komputer* mendefinisikan sebagai beriku. "Istilah data adalah suatu istilah majemuk yang berarti fakta atau bagian dari fakta yang mengandung arti yang dihubungkan dengan kenyataan, simbol-simbol, gambar-gambar, angka-angka, huruf-huruf, atau simbol-simbol yang menunjukka suatu ide, objek, kondisi atau situasi dan lain-lain. Jelasnya, data itu berupa apa saja dan dapat ditemui di mana saja. Kegunaan data adalah sebagai bahan dasar yang objektif (relatif) di dalam proses kebijaksanaan dan keputusan oleh pimpinan organisasi" (Sutabri, 2012:2).

2.2 Teori yang Berhubungan dengan Penelitian

2.2.1 Keputusan

Keputusan merupakan hasil pemikiran berupa pemilihan satu diantara beberapa alternatif yang dapat digunakan untuk memecahkan masalah yang dihadapi (Pratiwi, 2016:2).

Jenis-jenis keputusan dibedakan menjadi tiga macam yaitu keputusan terstruktur, keputusan tidak terstruktur, dan keputusan semi terstruktur (Pratiwi, 2016:5-6).

1. Keputusan terstruktur

   Keputusan-keputusan yang berkaitan dengan persoalan yang telah diketahui sebelumnya. Proses pengambilan keputusan seperti ini biasanya didasarkan atas teknik-teknik tertentu dan sudah dibuat standarnya. Kategori keputusan ini juga dapat dikatakan suatu proses jawaban secara otomatis pada kebijakan yang sudah ditentukan sebelumnya. Secara alamiah hampir semua masalah rutin dan berulang memiliki parameter-parameter persoalan yang telah diketahui dan terdefinisi dengan baik, sehingga jawaban atau proses pengambilan keputusan pun bersifat rutin dan terjadwal.

2. Keputusan tak terstruktur

   Keputusan-keputusan yang berkaitan dengan berbagai persoalan baru. Keputusan tidak terstruktur biasanya juga berkaitan dengan persoalan yang cukup pelik, karena banyak parameter yang tidak diketahui atau belum diketahui. Oleh karena itu, untuk mengambil keputusan ini biasanya intuisi serta pengalaman seorang pelaku organisasi akan sangat membantu.

   Keputusan tak terstruktur, adalah "*fuzzy*", permasalahan kompleks dimana tak ada solusi yang mengikutinya. Masalah yang tak terstruktur adalah tak adanya 3 fase proses yang terstruktur. Keputusan tidak terstruktur (unstructured decision) bukan merupakan keputusan yang berulang dan rutin. Contohnya adalah memilih sampul depan sebuah majalah, mengontrak manajemen tingkat senior, dan memilih proyek penelitian awal yang akan dilakukan. Tidak ada kerangka atau model yang dapat memecahkan masalah sejenis ini. Bahkan, dibutuhkan banyak sekali pertimbangan dan intuisi. Walaupun demikian,

keputusan tidak terstruktur dapat didukung oleh bantuan dari keputusan yang diambil berdasar hasil komputer, yang berfungsi untuk memfasilitasi pengumpulan informasi dari berbagai sumber. Contohnya adalah keputusan untuk pengembangan teknologi baru, pengembangan jenis usaha baru, keputusan untuk bergabung dengan perusahaan lain, perekrutan eksekutif.

3. Keputusan semi terstruktur

   Terdapat beberapa keputusan terstruktur, tetapi tak semua dari fase-fase yang ada. Keputusan semi terstruktur (*semistructured decision*) ditandai dengan dengan peraturan-peraturan yang tidak lengkap untuk mengambil keputusan, dan adanya kebutuhan untuk membuat penilaian serta petimbangan subjektif sebagai pelengkap analisis data yang formal. Menetapkan anggaran pemasaran untuk suatu produk baru adalah contoh dari keputusan semi terstruktur. Walaupun keputusan seperti ini biasanya tidak dapat secara penuh diotomatisasikan, namun sering didukung dari komputer (*computer-based decision*). Contoh keputusan jenis adalah investasi keuangan, pengevaluasian kredit, penjadwalan produksi, pemberian dana rehabilitasi sekolah, dan pengendalian persediaan.

2.2.2 Sistem Pendukung Keputusan

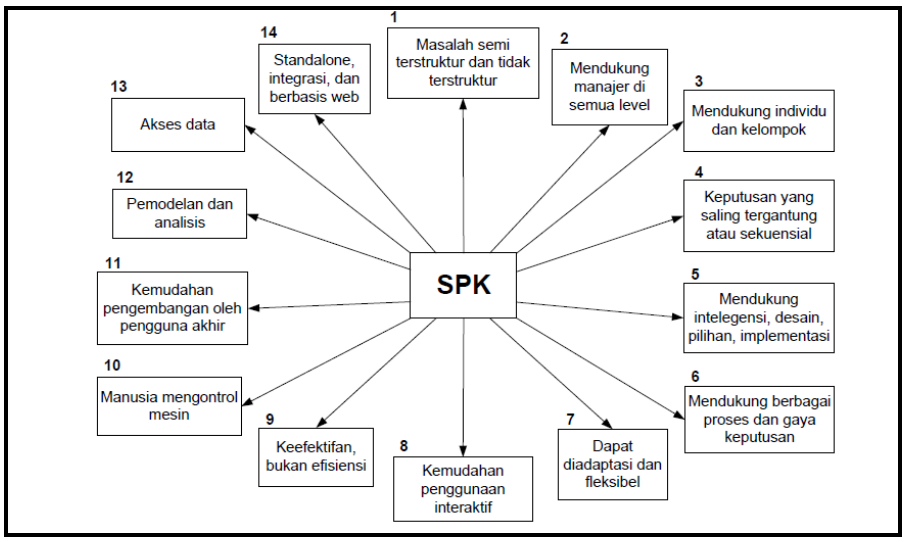
Pengertian sistem pendukung keputusan yang dikemukakan oleh McLeod (1998) yang menyatakan bahwa sistem pendukung keputusan merupakan sistem penghasil informasi yang ditujukan pada suatu masalah yang harus dibuat oleh manajer, sistem pendukung keputusan merupakan suatu sistem informasi yang ditujukan unyuk membantu manajemen dalam memecahkan masalah yang dihadapinya.

Definisi selengkapnya adalah sistem penghasil informasi spesifik yang ditujukan untuk memecahkan suatu masalah tertentu yang harus dipecahkan oleh manajer pada berbagai tingkatan. Sistem pendukung keputusan juga merupakan suatu sistem informasi berbasis komputer yang menghasilkan berbagai alternatif keputusan untuk membantu manajemen dalam menangani berbagai permasalahan

yang terstruktur ataupun tidak terstruktur dengan menggunakan data atau model (Pratiwi, 2016:4).

A. Karakteristik dan kapabilitas DSS

Karakteristik dan kapabilitas kunci dari DSS adalah (ditunjukkan pada gambar 2.1) (Turban, E., dkk, 2005:140-143):



Sumber: Turban, E., dkk (2005:142)

Gambar 2.1 Karakteristik dan kapabilitas kunci dari DSS

1. Dukungan untuk pengambil keputusan, terutama pada situasi semistruktur dan tak terstruktur, dengan menyertakan penilaian manusia dan informasi terkomputerisasi. Masalah-masalah tersebut tidak dapat dipecahkan (atau tidak dapat dipecahkan dengan konvenien) oleh sistem komputer lain atay oleh metode atau alat kuantitatif standar.
2. Dukungan untuk semua level manajerial, dari eksekutif puncak sampai manajer lini.
3. Dukungan untuk individu dan kelompok. Misalnya yang kurang terstruktur sering memerlukan keterlibatan individu dari departemen dan tingkat organisasional yang berbeda atau bahkan dari organisasi lain. DSS mendukung tim 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 di semua fase proses proses pengambilan keputusan: inteligensi, desain, pilihan dan implementasi.
6. Dukungan di berbagai proses dan gaya pengambilan keputusan.
7. Adaptivitas sepanjang waktu. Pengambil keputusan seharusnya reaktif, dapat menghadapi perubahan kondisi secara cepat, dan dapat mengadaptasikan DSS untuk memenuhi perubahan tersebut. DSS bersifat fleksibel dan karena itu pengguna dapat menambahan, menghapus, menggabungkan, mengubah, atau menyusun kembali elemen-elemen dasr. DSS juga fleksibel dalam hal dapat dimodifikasi untuk memecahkan masalah lain yang sejenis.
8. Pengguna merasa seperti rumah. Ramah-pengguna, kapabilitas grafis yang sangat kuat, dan antarmuka manusia mesin interaktif dengan satu bahasa alami dapat sangat meningkatan keefektifan DSS. Kebanyakan aplikasi DSS yang baru menggunakan antarmuka berbasis-Web.
9. Peningkatan terhadap keefektifan pengambilan keputusan (akurasi, timeliness, kualitas) ketimbang pada efisiensinya (biaya pengambilan keputusan). Ketika DSS disebarkan, pengambilan keputusan sering membutuhkan waktu lebih lama, namun keputusan lebih baik.
10. Kontrol penuh oleh pengambil keputusan terhadap semua langah proses pengambilan keputusan dalam langkah memecahkan suatu masalah. DSS secara khusus menekankan utnuk mendukung pengambilan keputusa, bukannya menggantikan.
11. Pengguna kahir dapat mengembangkan dan memodifikasi sendiri sistem sederhana. Sistem yang lebih besar dapat dibangun dengan bantuan ahli sistem informasi. Perangkat lunak OLAP dalam kaitannya dengan data warehouse membolehkan pengguna untuk membangun DSS yang cukup besar dan kompleks.
12. Biasanya model-model digunakan untuk menganalisis situasi pengambilan keputusan. Kapabilitas pemodelan memungkinkan eksperimen dengan berbagai strategi yang berbeda dibawah konfigurasi yang berbeda.
13. Akses disediakan untuk berbagai sumber data, format, dan tipe, mulai dari sistem informasi geografis (GIS) sampai sistem berorientasi-objek.

14. Dapat dilakukan sebagai alat standalone yang digunakan oleh seorang pengambil keputusan pada satu lokasi dan di beberapa organisasi sepanjang rantai persediaan. Dapat diintergrasikan dengan DSS lain dan atau aplikasi lain, dan dapat didistribusikan secara internal dan eksternal dengan menggunakan networking dan teknologi Web.

B. Tahapan Pengambilan Keputusan

Alur/ proses pemilihan alternatif tindakan/ keputusan biasanya terdiri dari langkah-langkah berikut (Pratiwi, 2016:10-11):

1. Tahap *Intelligence*

   Pencarian kondisi-kondisi yang dapat menghasilkan keputusan.

   Suatu tahap proses swseorang dalam rangka pengambil keputusan untuk permasalahan yang dihadapi, terdiri dari aktivitas penelusuran, pendeteksian serta proses pengenalan masalah. Data masukan diperoleh, diuji dalam rangka mengidentifikasi masalah.

2. Tahap *Design*

   Menemukan, mengembangkan dan menganalisis materi-materi yang mungkin untuk dikerjakan.

   Tahap proses pengambil keputusan adlaah tahap *intelligence* meliputi proses untuk mengerti masalah, mengenali solusi dan menguji kelayakan solusi. Aktifitas yang biasanya dilakukan seperti menemukan, mengembangkan dan menganalisa alternatif tindakan yang dapat dilakukan.

3. Tahap *Choice*

   Pemilihan dari alternatif pilihan yang tersedia, mana yang akan dikerjakan.

   Pada tahap ini dilakukan proses pemilihan diantara berbagai alternatif tindakan yang mungkin dijalankan. Hasil pemilihan tersebut kemudian diimplementasikan dalam proses pengambilan keputusan.

4. Tahap *Implementation*

   Implementasi dari SPK yang telah dpilih. Tahap implementasi adalah tahap pelaksanaan dari keputusan yang telah diambil. Pada tahap ini perlu disusun

serangkaian tindakan yang terencana, sehingga hasil keputusan dapat dipantau dan disesuaikan apabila diperlukan perbaikan.

C. Komponen-komponen DSS

Aplikasi DSS dapat terdiri dari subsistem sebagai berikut (Turban, E., dkk, 2005:143-144).

1. Subsistem manajemen data

   Subsistem manajemen data .Subsistem manajemen data memasukkan satu database yang berisi data yang relevan untuk situasi dan dikelola oleh perangkat lunak yang disebut **sistem manajemen database (DBMS)**. Subsistem manajemen data dapat diinterkoneksikan dengan **data warehouse** perusahaan, suatu repositori untuk data perusahaan yang relevan untuk pengambilan keputusan. Biasanya data disimpan atau diakses via server web database.

2. Subsisten manajemen model

   Subsisten manajemen model. Merupakan paket perangkat lunak yang memasukkan model keuangan, statistik, ilmu manajemen, atau model kuantitatif lainnya yang memberikan kapabilitas analitik dan manajemen perangkat lunak yang tepat. Bahasa-bahasa pemodelan untuk membangun model-model kustom juga dimasukkan. Perangkat lunak ini sering disebut **sistem manajemen basis model (MBMS)**. Komponen ini dapat dikoneksikan ke penyimpanan korporat atau eksternal yang ada pada model. Sistem manajemen dan metode solusi model diimplementasikan pada sistem pengembangan Web (seperti java) untuk berjalan pada server aplikasi.

3. Subsistem antarmuka pengguna

   Subsistem antarmuka pengguna. Pengguna berkomunikasi dengan dan memerintahkan DSS melalui subsistem ini. Pengguna adalah bagian yang dipertimbangkan dari sistem. Para peneliti menegaskan bahwa beberapa kontribuasi unik dari DSS berasal dari interaksi yang intensif antara komputer dan pembuat keputusan. Browser Web memberikan struktur antarmuka pengguna grafis yang familier dan konsisten bagi kebanyakan DSS.

4. **Subsistem manajemen berbasis**-pengetahuan

   Subsistem manajemen berbasis-pengetahuan. Subsistem ini dapat mendukung semua subsistem lain atau bertindak sebagai suatu komponen independen. Ia memberikan intelegensi untuk memperbesar pengetahuan si pengambil keputusan. Subsistem ini dapat diinterkoneksikan dengan repositori pengetahuan perusahaan (bagian dari sistem manajemen pengetahuan) yang kadang-kadang disebut basis pengetahuan organisasional. Pengetahuan dapat disediakan via server web. Banyak metode kecerdasan tiruan diimplementsikan dalam sistem pengembangan web seperti Java, dan mudah untuk diintergrasikan dengan komponen DSS lainnya.

2.3 Metode *Profile Matching*

Metode *profile matching* atau pencocokan profil adalah metode yang sering digunakan sebagai mekanisme dalam pengambilan keputusan dengan mengasumsikan bahwa terdapat tingkat variabel prediktor yang ideal yang harus dipenuhi oleh subyek yang diteliti, bukan tingkat minimal yang harus dipenuhi atau dilewati (Pratiwi, 2016:4).

Berikut adalah beberapa tahapan dan perumusan perhitungan dengan metode *profile matching* (Pratiwi, 2016:117-119) :

1. Pembobotan

   Pada tahap ini, akan ditentukan bobot nilai masing-masing aspek dengan menggunakan bobot nilai yang telah ditentukan bagi masing-masing aspek itu sendiri. Adapun inputan dari proses pemboboan ini adalah selisih dari profil karyawan dan profil jabatan. Dalam penentuan peringkat pada aspek kapasitas intelektual, sikap kerja dan perilaku untuk jabatan yang sama pada setiap gap, diberikan bobot nilai sesuai dengan tabel 2.1 berikut :

Tabel 2.1 Bobot Nilai Gap

No	Selisih Gap	Bobot Nilai	Keterangan
1	0	5	Konpetensi sesuai dengan yang dibutuhkan
2	1	4.5	Kompetensi individu kelebihan 1 tingkat/level

3	-1	4	Kompetensi individu kurang 1 tingkat/level
4	2	3.5	Kompetensi individu kelebihan 2 tingkat/level
5	-2	3	Kompetensi individu kurang 2 tingkat/level
6	3	2.5	Kompetensi individu kelebihan 3 tingkat/level
7	-3	2	Kompetensi individu kurang 3 tingkat/level
8	4	1.5	Kompetensi individu kelebihan 4 tingkat/level
9	-4	1	Kompetensi individu kurang 4 tingkat/level

Sumber: Pratiwi (2016:118)

2. Pengelompokan *Core* dan *Secondary factor*

   Setelah menentukan bobot nilai gap kriteria yang dibutuhkan, kemudian tiap kriteria dikelompokkan lagi menjadi dua kelompok yaitu *core factor* dan *secondary factor*.

   1) *Core Factor* (Faktor Utama)

      Core Factor merupakan aspek (kompetensi) yang paling menonjol/ paling dibutuhkan oleh suatu jabatan yang diperkirakan dapat menghasilkan kinerja optimal.

      Untuk menghitung *core factor* digunakan rumus

      $$NCI = \frac{\sum NC}{\sum IC} \quad \text{.....(1)}$$

      Keterangan :

      NCI = Nilai rata-rata *core factor* aspek kapasitas intelektual

      NC = Jumlah Total nilai *core factor* aspek kapasitas intelektual

      IC = Jumlah item *core factor*

   2) *Secondary Factor* (Faktor Pendukung)

      Secondary factor adalah item-item selain aspek yang ada pada *core factor.*

      Untuk menghitung *secondary factor* digunakan rumus :

      $$NSI = \frac{\sum NS}{\sum IS} \quad \text{.....(2)}$$

      Keterangan :

      NSI = Nilai rata-rata *secondary factor* aspek kapasitas intelektual

      NS = Jumlah Total nilai *secondary factor* aspek kapasitas intelektual

      IS = Jumlah item *secondary factor*

3. Perhitungan Nilai Total

   Dari perhitungan *core factor* dan *secondary factor* dari tiap-tiap aspek, kemudian dihitung nilai total dari tiap-tiap aspek diperkirakan berpengaruh pada kinerja tiap-tiap profile.

   Untuk menghitung nilai total dari masing-masing aspek, digunakan rumus :

   N = (X) % NCI + (X) % NSI .....(3)

   Keterangan :

   N = Nilai Total Tiap Aspek

   NCI = Nilai *Core Factor*

   NSI = Nilai *Secondary Factor*

   (X)% = Nilai Persentase

4. Perankingan

   Hasil akhir dari proses *profile matching* adalah ranking dari kandidat untuk mengisi jabatan/posisi tertentu. Penetuan mengacu ranking pada hasil perhitungan yang ditujukan pada rumus dibawah ini :

   Ranking = (X)% N1 + (X)% N2 + (X)% N3.....(4)

   Keterangan :

   N1 = Nilai Aspek 1

   N2 = Nilai Aspek 2

   N3 = Nilai Aspek 3

2.4 Teori yang Berhubungan Teknik Analisa yang Digunakan

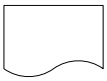
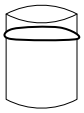
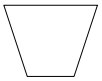
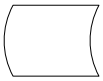
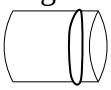
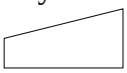
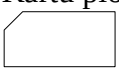
2.4.1 *Flowchart*

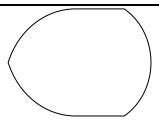
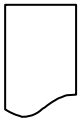
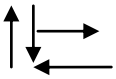
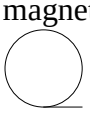
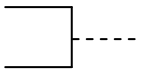
Bagan alir (*Flowchart*) adalah bagan (*Chart*) yang menunjukan alir (*Flow*) di dalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi serta pada waktu akan menggambarkan suatu bagan alir (Jogiyanto, 2005:795). Ada lima macam bagan alir yakni terdiri dari bagan alir sistem (*Systems flowchart),* bagan alir dokumen (*Document flowchart),* bagan alir skematik (*Scematic flowchart),* bagan alir program (*Program Flowchart*), bagan alir proses *(Process flowchart)* yang saya gunakan dalam penelitian ini adalah bagan alir sistem (*Systems flowchart).*

Bagan alir sistem (*Systems flowchart)* merupakan bagan yang menunjukkan arus pekerjaan secara keseluruhan dari sistem. Bagan ini menjelaskan urutan-urutan dari prosedur-prosedur yang ada didalam sistem, bagan alir sistem menunjukan apa yang dikerjakan di sistem, bagan alir sistem digambar dengan menggunakan simbol-simbol (Jogiyanto, 2005:796).

Berikut simbol bagan alir sistem (*systems flowchart*) dapat dilihat pada Tabel 2.2 :

Tabel 2.2 Simbol *Systems Flowchart*

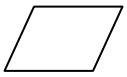
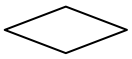
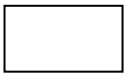
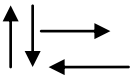
No	Simbol	Keterangan	No	Simbol	Keterangan
1	Dokumen 	Menunjukkan dokumen *input* dan *output* baik proses manual, mekanil atau *computer*	11	Hard disk 	Menunjukkan *input/ output* menggunakan hard disk
2	Kegiatan Manual 	Menunjukkan pekerjaan manual	12	Diskette 	Menunjukkan *input/ output* menggunakan diskette
3	Simpanan Offline 	File non-komputer yang diarsip urut angka (*numerical*)	13	Drum magnetik 	Menunjukkan *input/ output* menggunakan Drum magnetik
4	Simpanan Offline 	File non-komputer yang diarsip urut angka (*alphabetical*)	14	Pita kertas berlubang 	Menunjukkan *input /output* menggunakan Pita kertas berlubang
5	Simpanan Offline 	File non-komputer yang diarsip urut angka (*cronological*)	15	Keyboard 	Menunjukkan *input/ output* menggunakan on-line keyboard
6	Kartu plong 	Menunjukkan *input/output* yang	16	Display	Menunjukan output yang tampil di komputer

		menggunakan kartu plong			
7	Proses 	Menunjukkan proses dari operasi program komputer	17	Pita kontrol 	Menunjukkan penggunaan pita kontrol dalam *bacth control total* untuk pencocokan di proses *bacth processing*
8	Operasi luar 	Menujukkan operasi yang dilakukan diluar proses operasi komputer	18	Hubungan komunikasi 	Menunjukkan proses transmisi data melalui channel komunikasi
9	Pengurutan offline 	Menunjukan proses pengurutan data diluar proses komputer	19	Garis alir 	Menunjukkan arus proses
10	Pita magnetik 	Menunjukkan *input/ output* menggunakan pita magnetik	20	Penjelasan 	Penjelelasan dari suatu proses
			21	Penghubung 	Menunjukan penghubung ke halaman yang masi sama atau ke halaman lain

Sumber: Jogiyanto (2005:796)

Bagan alir program *program flowchart*) merupakan bagan yang menjelaskan secara rinci langkah-langkah dari proses program (Jogiyanto, 2005:802). Berikut simbol bagan alir program (*program flowchart*) dapat dilihat pada Tabel 2.3:

Tabel 2.3 Simbol *Program Flowchart*

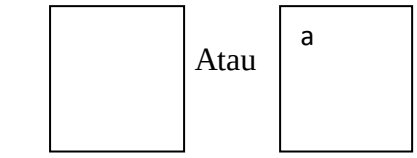
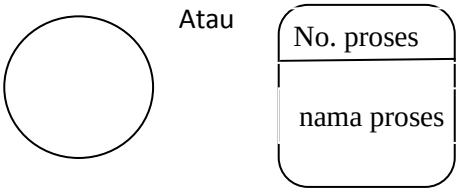
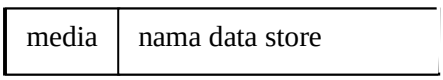
Simbol	Keterangan	Simbol	Keterangan
Input/output 	Simbol input / output digunakan untuk mewakili data input output	Keputusan 	Simbol keputusan digunakan untuk suatu penyeleksian kondisi di dalam program
Proses 	Simbol proses digunakan untuk mewakili proses	Proses terdefinisi 	Simbol proses terdefinisi digunakan untukmenunjukkan suatu operasi yang rinciannya ditunjukkan di tempat lain
Garis alir 	Simbol garis alir (*flow lines simbol*) digunakan untuk menunjukan arus dari proses	persiapan 	Simbol persiapan digunakan untuk member nilai awal suatu besaran
Penghubung 	Simbol Penghubung menunjukkkan sambungan dari bagan alir yang tertputus di halaman yang masih sama atau di halaman lainnya.	Titik terminal 	Titik terminal digunakan untuk menunjukkan awal dan akhir dari suatu proses

Sumber: Jogiyanto (2005:802)

2.4.2 DFD (*Data Flow Diagram*)

Diagram yang menggunakan notasi-notasi ini untuk menggambarkan arus dari data sistem sekarang dikenal dengan nama diagram arus data (*data flow diagram atau* DFD). DFD merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur (*structured Analysis and design*). DFD merupakan alat yang cukup populer sekarang ini, karena dapat menggambarkan arus data di dalam sistem dengan terstruktur dan jelas. Lebih lanjut DFD juga merupakan dokumentasi dari sistem yang baik (Jogiyanto, 2005:700). Berikut simbol digunakan di DFD untuk maksud mewakili dapat dilihat pada Tabel 2.4:

Tabel 2.4 Simbol DFD

Simbol	Keterangan
	*External entity* (kesatuan luar) : sistem akan menerima input dan menghasilkan output kepada lingkungan luarnya.
	Data flow (arus data) : menunjukkan arus dari data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem
	*Process* (proses) : kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari hasil suatu arus data yang masuk kedalam proses untuk dihasilkan arus data yang akan keluar dari proses
	*Data store* (simpanan data) : merupakan simpanan dari data

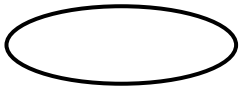
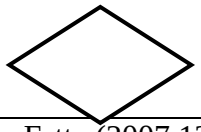
Sumber: Jogiyanto (2005:700-707)

2.4.3 ERD (*Entity Relational Diagram*)

ERD adalah notasi yang digunakan untuk melakukan aktivitas pemodelan data (Pressman, 2014:353). ERD adalah gambar atau diagram yang menunjukkan informasi dibuat, disimpan dan digunakan dalam sistem bisnis (Fatta, 2007:121). Berikut simbol-simbol ERD (Fatta, 2007:124):

Tabel 2.5 Simbol ERD

Simbol	Keterangan
	Entitas : Orang, tempat, atau benda memiliki nama tunggal

	Attribut : Property dari entitas harus digunakan oleh minimal 1 proses bisnis dipecah dalam detail
	Relationship: Menunjukan hubungan antar 2 entitas, dideskripsikan dengan kata kerja.

Sumber: Fatta (2007:124)

2.5 Teori Pendukung Lainnya

2.5.1 *Database*

Basis data (*database*) dapat didefinisikan sebagai himpunan kelompok data saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah (Hidayatullah dan Kawistara, 2014:147).

2.5.2 HTML (*Hypertext Markup Language*)

Hypertxt Markup Language (HTML) adalah bahasa standard yang digunakan untuk menampilkan halaman *web*. Yang bisa dilakukan HTML yaitu (Hidayatullah dan Kawistara, 2014:13).

1. Mengatur tampilan dari halaman *web* dan isinya.
2. Membuat tabel dalam halamn *web*.
3. Mempublikasikan halaman *web* secara *online*.
4. Membuat form yang bisa digunakan untuk menangani registrasi dan transaksi via *web*.
5. Menambahkan objek-objek seperti citra, audio, video, animasi, *java applet* dalam halaman *web*.
6. Menampilkan area gambar (canvas) di *browser*.

2.5.3 MySQL

MySQL adalah salah satu aplikasi DBMS yang sudah sangat banyak digunakan oleh para pemrogram aplikasi web. Kelebihan dari *MySQL* adalah gratis, handal, selalu di-*update* dan banyak form yang memfasilitasi para pengguna jika memiliki kendala. *MySQL* juga menjadi DBMS yang sering

dibundling dengan web *server* sehingga proses instalasinya jadi lebih mudah (Hidayatullah dan Kawistara, 2014: 180).

2.5.4 PHP (*Hypertext Preprocessor)*

PHP *Hypertext Preprocessor* atau disingkat dengan PHP ini adalah suatu bahasa *scripting* khusunya digunakan untuk web *development*. Karena sifatnya yang *server side scripting*, maka untuk menjalankan PHP harus menggunakan *web server* (Hidayatullah dan Kawistara, 2014: 231).

2.6 Metode Pengembangan *Extreme Programming* (XP)

Metode pengembangan sistem yang akan digunakan dalam penelitian ini adalah metode *Extreme Programming* (XP). Adapun tahapannya antara lain (Keina, 2013):

1. *Planning*, Aktivitas planning dimulai dengan membentuk *user stories*. Anggota *Extreme Programming* (XP) team kemudian menilai setiap *story* dan menentukan *cost* diukur dalam *development week*. *Customer* dan *Extreme Programming* (XP) team bekerja bersama untuk memutuskan bagaimana *group story* untuk release berikutnya (*software increment*) berikutnya untuk dibangun oleh *Extreme Programming* (XP) team. Jika komitmen telah dibuat, *Extreme Programming* (XP) team akan membangun story-story dengan cara:
   a. Semua *story* segera diimplementasikan (dalam beberapa minggu).
   b. *Story* dengan value tertinggi akan dipindahkan dari jadwal dan dimplementasikan pertama.
   c. *Story* dengan resiko paling tinggi akan diimplemetasikan lebih dulu. Setelah *project* pertama *di-release* dan *didelivery*, *Extreme Programming* (XP) team memperhitungkan kecepatan *project*. Selama *development*, *customer* dapat menambah *story*, merubah *value*, membagi *story* atau menghapusnya.
2. *Design*. *Extreme Programming* menggunakan CRC card, untuk mengenali dan mengatur *object oriented class* yang sesuai dengan software increment.
3. *Coding*. Sebelum membuat *code*, lebih baik membuat unit test tiap *story* untuk dimasukkan dalam *software increment*. *Extreme Programming* (XP)

menyarankan agar dua orang bekerja bersama pada satu komputer *workstation* untuk membuat *code* dari satu *story (pair programming*), untuk menyediakan *real time problem solving* dan jaminan *real time quality*. Setelah *pair programming* selesai, *code* diintegrasikan dengan kerja lainnya (*continuou sintegration*).

4. *Testing*. Unit test yang telah dibuat harus diimplementasikan menggunakan suatu *framework* dan diatur ke dalam *universal testing suite*, integrasi dan validasi sistem dapat dilakukan setiap hari. *Customer test* (*acceptance test*) dilakukan oleh *customer* dan fokus pada keseluruhan fitur dan fungsional sistem. *Acceptance test* diperoleh dari *customer stories* yang telah diimplemetasikan sebagai bagian dari *software release*.

2.7 Metode Pengujian *Black Box*

Metode pengujian digunakan untuk mengetahui fungsi yang telah ditentukan bahwa suatu sistem telah dirancang dapat menunjukan bahwa masing-masing fungsi sepenuhnya beroperasi. Pengujian kotak hitam *(black box)*, juga disebut pengujian prilaku, berfokus pada persyaratan fungsional perangkat lunak.

Artinya, teknik pengujian kotak hitam memungkinkan untuk membuat beberapa kumpulan kondisi masukan yang sepenuhnya akan melakukan semua kebutuhan fungsional untuk program. Pengujian kotak hitam *(black box)* bukan teknik alternatif untuk kotak putih *(white box).*

Pengujian kotak hitam *(black box)* berupaya untuk menemukan kesalahan dalam kategori berikut: (1) fungsi yang salah atau hilang, (2) kesalahan dalam struktur data atau akses basis eksternal, (4) kesalahan perilaku atau kinerja, dan (5) kesalahan inisialisasi dan penghentian (Roger S. Pressman, 2012:597).