BAB II

LANDASAN TEORI

2.1. Konsep Dasar Sistem

Pemahaman mengenai konsep dasar sistem informasi ini memerlukan pendekatan-pendekatan mengenai sistem, sistem informasi dan sistem informasi akuntansi yang mencakup pengertian, karakteristik sistem, klasifikasi sistem, sistem informasi dan sistem informasi akuntansi.

2.1.1. Pengertian Sistem

Menurut Fauzi (2017:2) mendefinisikan bahwa "Sistem adalah suatu kesatuan yang terdiri dari interaksi subsistem untuk mencapai tujuan yang sama". Sedangkan menurut Tyoso (2016:1) mengemukakan bahwa, "Sistem merupakan suatu kumpulan dari komponen-komponen yang membentuk satu kesatuan".

Dapat ditarik suatu kesimpulan bahwa sistem merupakan sekumpulan elemen, komponen atau subsistem yang saling berhubungan, bekerja sama dan membentuk satu kesatuan dalam upaya mencapai tujuan.

2.1.2. Karakteristik Sistem

Sistem memiliki karakteristik atau ciri-ciri agar dikategorikan sebagai suatu sistem yang baik. Karakteristik dari sistem (Fauzi, 2017:2) diuraikan sebagai berikut:

1. Komponen sistem

   Suatu sistem terjadi dikarenakan adanya sejumlah komponen yang melakukan interaksi. Suatu sistem yang sekecil apapun akan selalu mengandung komponen-komponen.

2. Batas sistem

   Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lain atau dengan lingkungan luarnya.

3. Lingkungan luar sistem

   Lingkungan luar dari suatu sistem adalah daerah di luar batas dari suatu sistem yang mempengaruhi operasi sistem.

4. Penghubung sistem

   Penghubung merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari suatu subsistem ke subsistem yang lainnya. Keluaran dari sistem menjadi masukan untuk subsistem lainnya.

5. Masukan sistem

   Masukan sistem adalah energi yang dimasukkan ke dalam sistem.

6. Keluaran sistem

   Keluaran sistem adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisi pembuangan.

7. Pengolah sistem

   Suatu sistem dapat mempunyai suatu bagian pengolah atau sistem itu sendiri sebagai pengolahnya. Pengolah yang akan mengubah masukan mejadi keluaran.

8. Sasaran sistem

   Suatu sistem pasti mempunyai tujaun (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran maka sistem tersebut tidak berguna.

2.1.3. Klasifikasi Sistem

Beberapa aspek dari sistem mengizinkan pengguna untuk mengklarifikasikan sistem berdasarkan sudut pandang. Klasifikasi sistem yang dimaksud (Tyoso, 2016:5), yaitu:

1. Sistem Alamiah (*Natural System*) dan Sistem Buatan Manusia (*Artificial System*)
    a. Sistem alamiah merupakan sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia, misalnya sistem tata surya, sistem galaksi, sistem reproduksi dan lain-lain.
    b. Sistem buatan manusia merupakan sistem yang dirancang oleh manusia. Sistem buatan yang melibatkan interaksi manusia, misalnya sistem akuntansi, sistem informasi, dan lain-lain.
2. Sistem Deterministik (*Deterministic System*) dan Sistem Probabilistik (*Probabilistic System*)
    a. Sistem deterministik merupakan sistem yang beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi bagian-bagiannya dapat dideteksi dengan pasti sehingga keluaran dari sistem dapat diramalkan, misalnya sistem komputer, adalah contoh sistem yang tingkah lakunya dapat dipastikan berdasarkan program-program komputer yang dijalankan.
    b. Sistem probabilistik merupakan sistem yang kondisi masa depanya tidak dapat diprediksi karena mengandung unsur probabilitas, misalnya sistem manusia.

3. Sistem Terbuka (*Opened System*) dan Sistem Tertutup (*Closed System*)
    a. Sistem terbuka merupakan sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Lebih sepesifik dikenal juga yang disebut dengan sistem terotomasi, yang merupakan bagian dari sistem buatan manusia dan berinteraksi dengan kontrol oleh satu atau lebih komputer sebagai bagian dari sistem yang digunakan dalam masyarakat modern. Sistem ini menerima masukan dan menghasilkan keluaran untuk subsistem lainnya, misalnya sistem kebudayaan manusia.
    b. Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa danya campur tangan dari pihak luar. Secara teoritis sistem tersebut ada, tetapi kenyataannya tidak ada sistem yang benar-benar tertutup, yang ada hanyalah *relatively closed system* (secara relatif tertutup, tidak benar-benar tertutup).

2.1.4. Sistem Informasi

Suatu sistem informasi adalah suatu kombinasi dari orang-orang, fasilitas, teknologi, media, prosedur-prosedur dan pengendalian yang ditujukan untuk mendapatkan jalur komunikasi penting, memproses tipe transaksi rutin tertentu, memberi sinyal kepada manajemen dan yang lainnya terhadap kejadian-kejadian internal dan eksternal yang penting dan menyediakan sesuatu dasar untuk pengambilan keputusan (Fauzi, 2017:18).

Menurut Hutahaean (2015:13) mengemukakan bahwa “Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengelolaan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan

strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang dibutuhkan".

Berdasarkan pendapat dari beberapa ahli di atas, dapat disimpulkan bahwa sistem informasi merupakan sekumpulan dari orang, perangkat lunak, perangkat keras, dan prosedur yang saling berinteraksi, bekerja sama dalam menyelesaikan sesuatu untuk menghasilkan informasi yang dapat dijadikan sebagai dasar pengambilan keputusan.

Komponen-komponen dari sistem disebut dengan blok bangunan (*building block*). Penjelasan blok bangunan (Hutahaean, 2015:13) diuraikan sebagai berikut:

1. Blok masukkan (*input block*)

   Blok masukan merupakan blok yang bertugas dalam *input* data agar masuk ke dalam sistem informasi. Blok masukan bertugas dalam merekam data yang akan dimasukkan, biasanya berupa dokumen-dokumen dasar.

2. Blok model (*model block*)

   Blok model terbentuk dari kombinasi prosedur, logika dan model matematik yang memproses data *input* dan data yang tersimpan di basis data dengan cara yang sudah ditentukan untuk menghasilkan keluaran yang diinginkan.

3. Blok keluaran (*output block*)

   Sistem informasi menghasilkan keluaran (*output*) yaitu informasi yang berkualitas dan berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

4. Blok teknologi (*technology block*)

   Teknologi digunakan merupakan kotak alat dalam sistem informasi. Teknologi digunakan untuk menerima *input*, menjalankan model, menyimpan dan

mengakses data, menghasilkan dan mengirimkan keluaran berupa informasi dan membantu pengendalian dari sistem secara menyeluruh. Blok teknologi perangkat lunak (*software*) dan perangkat keras (*hardware*) yang dioperasikan oleh teknisi (*brainware*).

5. Blok basis data (*database block*)

   Basis data (*database*) merupakan media untuk menyimpan data yang saling berhubungan satu sama lainnya, tersimpan di perangkat keras komputer dan dapat dipergunakan kembali, diperlukan perangkat lunak untuk memanipulasinya.

6. Blok kendali (*control block*)

   Sistem informasi memiliki kontrol kendali untuk menanggulangi gangguan-gangguan terhadap sistem apabila terlanjur terjadi kesalahan maka dapat langsung diantisipasi atau diatasi.

2.1.5. Sistem Informasi Akuntansi

Menurut Mahatmyo (2014:9) mendefinisikan bahwa "Sistem informasi akuntansi merupakan sekelompok struktur dalam sebuah entitas yang mengelola sumber daya fisik dan sumber daya lain untuk mengubah data ekonomi menjadi informasi akuntansi, agar dapat memenuhi kebutuhan informasi berbagai pihak". Sedangkan menurut Mulyani (2016:21) mengemukakan bahwa "Sistem informasi akuntansi digunakan sebagai alat untuk melakukan analisis keputusan ataupun sebagai pembuat keputusan yang terkait dengan transaksi-transaksi perusahaan".

Maka dari itu dapat disimpulkan bahwa sistem informasi akuntansi adalah suatu komponen organisasi yang mengumpulkan, mengklasifikasikan, mengolah,

menganalisa dan mengkomunikasikan informasi finansial dan pengambilan keputusan yang relevan kepada pihak luar dan dalam perusahaan.

2.1.6. Penerimaan Kas

Menurut Bahri (2016:340) mendefinisikan bahwa "Jurnal penerimaan kas (*cash receipt jurnal*) yaitu berfungsi untuk mencatat seluruh transaksi penerimaan kas". Sedangkan menurut Shatu (2016:58) mengemukakan bahwa "Jurnal penerimaan kas ialah jurnal yang disediakan khusus untuk pencatat transaksi penerimaan kas".

Maka dari itu, dapat disimpulkan bahwa jurnal penerimaan kas adalah jurnal khusus yang dipakai untuk mencatat semua transaksi keuangan yang mengakibatkan bertambahnya kas atau uang tunai perusahaan.

2.1.7. Pengeluaran Kas

Menurut Bahri (2016:34) mengemukakan bahwa "Jurnal pengeluaran kas (*cash payments journal*) berfungsi untuk mencatat seluruh transaksi pengeluaran kas". Sedangkan menurut Shatu (2016:59) mendefinisikan bahwa "Jurnal pengeluaran kas adalah jurnal yang khusus untuk mencatat transaksi-transaksi pengeluaran kas".

Berdasarkan pendapat para ahli di atas, dapat disimpulkan bahwa jurnal pengeluaran kas adalah jurnal khusus yang digunakan untuk mencatat semua pengeluaran uang tunai atau kas dari berbagai jenis transaksi yang terjadi pada perusahaan.

2.1.8. Laporan Keuangan

Menurut Hery (2016:3) mengemukakan bahwa "Laporan keuangan pada dasarnya adalah hasil dari proses akuntansi yang digunakan sebagai alat untuk mengkomunikasikan data keuangan atau perusahaan kepada pihak-pihak yang berkepentingan".

Laporan keuangan merupakan satu dari beragam informasi yang digunakan pengguna untuk pengambilan keputusan yang berisikan tentang catatan keuangan perusahaan (Yadiati & Mubarok, 2017:6).

Maka dari itu dapat disimpulkan bahwa laporan keuangan merupakan catatan yang berisikan informasi tentang keuangan suatu perusahaan pada suatu periode akuntansi yang menggambarkan kinerja dari perusahaan tersebut.

Laporan keuangan memiliki beberapa tahapan berdasarkan urutan penyajiannya. Urutan laporan keuangan berdasarkan proses penyajiannya (Hery, 2016:3) adalah sebagai berikut:

1. Laporan Laba Rugi (*Income Statement*)

   Laporan laba rugi merupakan laporan yang sistematis tentang pendapatan dan beban perusahaan untuk satu periode tertentu. Laporan laba rugi ini pada akhirnya memuat informasi mengenai hasil kinerja manajemen atau hasil kegiatan operasional perusahaan.

2. Laporan Ekuitas Pemilik (*Statement Of Owner's Equity*)

   Laporan ekuitas pemilik adalah sebuah laporan yang menyajikan ikhtisar perubahan dalam ekuitas pemilik suatu perusahaan untuk satu periode tertentu.

   Neraca (*Balance Sheet*)

Neraca adalah sebuah laporan yang sistematis tentang posisi aset, kewajiban dan ekuitas perusahaan per tanggal tertentu.

3. Laporan Arus Kas (*Statement Of Cash Flows*)

   Laporan arus kas adalah sebuah laporan yang menggambarkan arus kas masuk dan arus kas keluar secara terperinci dari masing-masing aktivitas, yaitu mulai dari aktivitas operasi, aktivitas investasi, sampai pada aktivitas pendanaan atau pembiayaan untuk satu periode waktu tertentu.

2.1.9. Basis Data

Menurut Lubis (2016:3) mendefinisikan bahwa "Basis data adalah tempat berkumpulnya data yang saling berhubungan dalam suatu wadah (organisasi/perusahaan) bertujuan agar dapat mempermudah dan mempercepat untuk pemanggilan atau pemanfaatan kembali data tersebut". Sedangkan menurut Yanto (2016:11) mengemukakan bahwa "Basis data merupakan kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redundansi), untuk memenuhi berbagai kebutuhan".

Maka dari itu bahwa basis data merupakan suatu wadah yang menampung data-data yang saling berhubungan, dapat digunakan kembali, manipulasi, tanpa pengulangan untuk memenuhi berbagai kebutuhan pengguna data.

Komponen dasar sistem basis data digunakan untuk membantu kelancaran dari pembuatan dan manajemen basis data (Lubis, 2016:7), yang terdiri dari:

1. Data

   Data pada sistem data mempunyai dua (2) ciri, yaitu data yang tersimpan secara terintegrasi (*integrated*) dan data dapat dipakai bersama-sama (*shared*).

a. *Integrated* yaitu kumpulan dari berbagai macam *file* dari aplikasi-aplikasi yang berbeda yang disusun dengan cara menghilangkan bagian-bagian yang rangkap (*redundant*).
b. *Shared* yaitu masing-masing bagian dari *database* dapat diakses oleh pemakai dalam waktu yang bersamaan, untuk aplikasi yang berbeda.

2. Perangkat keras

   Perangkat keras ini terdiri dari semua peralatan perangkat keras komputer yang digunakan untuk pengelolaan sistem *database*, antara lain:

   a. Peralatan untuk penyimpanan, *disk*, *drum*, dan lain-lain.
   b. Peralatan *input* dan *output*.
   c. Peralatan komunikasi data.

3. Perangkat lunak

   Perangkat lunak berfungsi sebagai perantara (*interface*) antara pemakai dengan data fisik *database*, dapat berupa *database management system* (DBMS) atau program-program aplikasi dan prosedur-prosedur.

4. Pemakai

   Pemakai ini terbagi menjadi dua (2) bagian, yaitu:

   a. *Programmer*, orang/*team* yang membuat program aplikasi yang mengakses *database* dengan menggunakan bahasa pemrograman.
   b. *End user*, orang yang mengakses *database* melalui terminal dengan menggunakan *query language* atau program aplikasi yang dibuat oleh *programmer*.

Penggunaan basis data ini memiliki beberapa keuntungan (Lubis, 2016:8), diantaranya:

1. Terkontrolnya kerangkapan data dan inkonsistensi.
2. Terpeliharanya keselarasan data.
3. Data dapat dipakai secara bersama-sama.
4. Memudahkan penerapan standarisasi.
5. Memudahkan penerapan batasan-batasan penggunaan.
6. Terpeliharanya integritas data.
7. Terpeliharanya keseimbangan atas perbedaan kebutuhan data dari setiap aplikasi.
8. Program/data *independent*.

Basis data juga memiliki beberapa kerugian dalam penggunannya. Adapun kerugian basis data (Lubis, 2016:8), yaitu:

1. Mahal dalam implementasinya.
2. Rumit/kompleks.
3. Penanganan proses *recovery* & *back up* sulit.
4. Kerusakan pada sistem basis data dapat mempengaruhi departemen terkait.

2.1.10. *Website*

Media informasi khususnya *website* dan *internet* merupakan hal yang tidak asing lagi karena merupakan bagian teknologi di kalangan masyarakat, banyak pula aplikasi berbasis jaringan (*web-based application*). *Website* merupakan salah satu media pemasaran yang cukup menjanjikan. Situs *web* yang menarik dan informatif dapat dibuat dengan HTML dan PHP (Anna, 2016).

Menurut Ginanjar (2014:5) mengemukakan bahwa "*Website* adalah rangkaian atau sejumlah halaman di *internet* yang memiliki topik saling terkait untuk mempresentasikan suatu informasi". Sedangkan menurut Yuhefizar (2013:2),

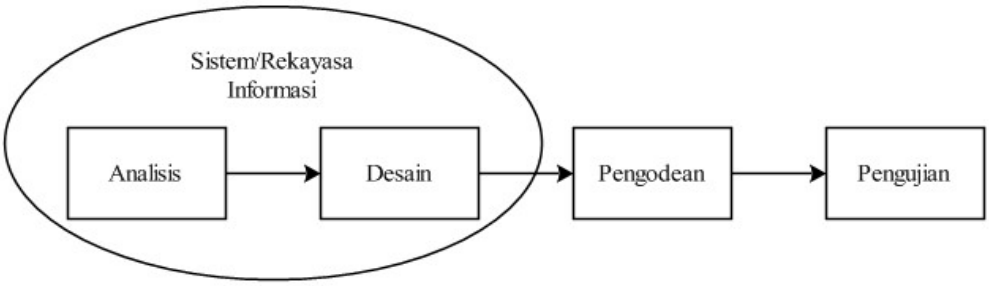
mendefinisikan bahwa "*Website* adalah keseluruhan halaman-halaman *web* yang terdapat dari sebuah domain yang mengandung informasi".

Dapat disimpulkan bahwa *website* merupakan rangkuman dari keseluruhan halaman-halaman *web* yang ada pada sebuah domainyang mengandung informasi teks, gambar diam atau gerak, animasi, suara, yang bersifat dinamis atau statis yang membentuk suatu rangkaian bangunan yang saling terkait dan memerlukan *internet*.

2.1.11. Model Pengembangan Perangkat Lunak

Menurut Rosa & Shalahuddin (2015:28) mengemukakan bahwa "Model *waterfall* adalah metode air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisa, desain, pengkodean, pengujian, dan pendukung (*support*)". Sedangkan menurut Muharto & Ambarita (2016:104) mendefinisikan bahwa "model *waterfall* ini melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem lalu menuju ke tahap analisis, desain, *coding*, *testing/verification*, dan *maintenance*".

Berdasarkan pendapat dari beberapa ahli di atas, dapat disimpulkan bahwa model *waterfall* merupakan model dari metode pengembangan perangkat lunak yang melakukan pendekatan sistematis dan sekuensial mulai dari tahapan analisis, desain, pengkodean, pengujian dan pendukung.



Sumber: Rosa & Shalahuddin (2015:29)

Gambar II.1. Tahapan Model *Waterfall*

Adapun penjelasan tahapan model *waterfall* (Rosa & Shalahuddin, 2015:29), yaitu:

1. Analisis kebutuhan perangkat lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user.* Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

2. Desain

   Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antar muka dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu untuk didokumentasikan.

3. Pembuatan Kode Program

   Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai desain yang telah dibuat pada tahap desain.

4. Pengujian

   Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error)* dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

5. Pendukung *(support)* atau pemeliharaan *(maintenance)*

   Tahap pendukung atau pemeliharaan dapat mengulangi proses mulai dari tahap analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.2. Teori Pendukung

Teknik atau alat bantu digunakan dalam memvisualisasikan rancangan sistem. Teori-teori lain yang digunakan untuk mendukung penulisan ini terdiri dari *unified modeling language* (UML) yang terdiri dari *use case diagram* dan *activity diagram*, *entity relationship diagram* (ERD) dan *black box testing*.

2.2.1. *Unified Modeling Language* (UML)

UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem (Hendini, 2016). Sedangkan Menurut Rosa & Shalahuddin (2015:137) mendefinisikan bahwa "UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung".

Maka dari itu dapat ditarik kesimpulan bahwa *unified modeling language* (UML) merupakan suatu bahasa standar yang digunakan untuk pemodelan dan komunikasi rancangan perangkat lunak dengan menggunakan diagram atau simbol-simbol tertentu.

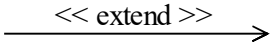
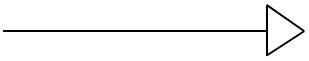
A. *Use Case Diagram*

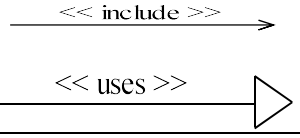
Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat". *Use case diagram* digunakan untuk mendeskripsikan tipikal interaksi antara pengguna dengan sistem informasi

(Maulana, 2014). Sedangkan menurut Rosa & Shalahuddin (2015:155) mendefinisikan bahwa "*Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat".

Berdasarkan beberapa pendapat ahli di atas, dapat disimpulkan bahwa *use case diagram* merupakan diagram UML yang berfungsi sebagai alat bantu pemodelan untuk menggambarkan tingkah laku (*behavior*) dari sudut pandang luar sistem untuk menjelaskan interaksi dan peran antara aktor dengan sistem yang dirancang.

Tabel II.1.
Simbol *Use Case Diagram*

Simbol	Deskripsi
use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan kata kerja di awal frase nama *use case*.
Aktor/*actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ektensi/*extend* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya.

Menggunakan/*include*/*uses*  	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

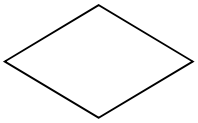
Sumber: (Rosa & Shalahuddin, 2015:156).

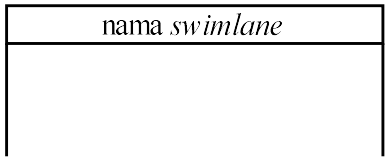
B. *Activity Diagram*

Activity diagram merupakan diagram yang menerangkan tentang aktifitas-aktifitas yang dapat dilakukan oleh seorang *entity* atau pengguna yang akan diterapkan pada aplikasi (Meilinda, 2016). Sedangkan menurut Rosa & Shalahuddin (2015:161) mendefinisikan bahwa "Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak"

Dapat disimpulkan bahwa *activity diagram* merupakan diagram yang menggambarkan aktifitas-aktifitas sistem dimana setiap urutan aktifitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.

Tabel II.2.
Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas  	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.

Penggabungan/*join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Sumber: (Rosa & Shalahuddin, 2015:162)

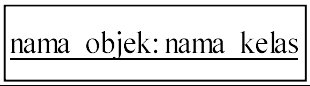
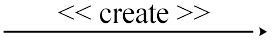
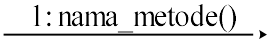
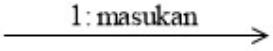
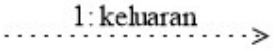
C. *Sequence Diagram*

Sequence diagram merupakan UML yang menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu (Meilinda, 2016).

> Rosa & Shalahuddin (2015:165) mengemukakan bahwa:
> Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima oleh objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Dapat disimpulkan bahwa *sequence diagram* dapat diartikan sebagai alat pemodelan rancangan sistem yang menggambarkan alur atau urutan sistem yang bersinkronisasi dengan *use case diagram* untuk mendeskripsikan waktu hidup objek dan *message* yang dikirm atau diterima oleh objek tersebut.

Tabel II.3.
Simbol *Sequence Diagram*

Simbol	Deskripsi
Aktor 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.
Garis hidup/*lifeline*	Menyatakan kehidupan suatu objek.
Objek 	Menyatakan objek yang berinteraksi pesan.
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.
Pesan tipe *create* 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri. Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
Pesan tipe *return* 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
Pesan tipe *destroy*	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*

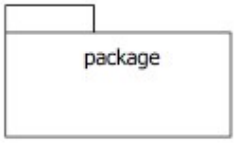
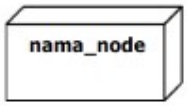
Sumber: Rosa & Shalahuddin (2015:165)

D. *Deployment Diagram*

Deployment diagram digunakan untuk menggambarkan detail bagaimana komponen disusun di infrastruktur sistem (Hendini, 2016). Sednagkan, menurut Rosa & Shalahuddin, 2015:154 "*Deployment diagram* menunjukan konfigurasi komponen dalam proses eksekusi aplikasi".

Dapat ditarik kesimpulan bahwa *deployment* diagram merupakan diagram UML yang berfungsi untuk menggambarkan konfigurasi komponen yang disusun sebagai infrastruktur aplikasi.

Tabel II.4.
Simbol *Deployment Diagram*

Simbol	Deskripsi
Package 	*Package* merupakan sebuah bungkusan dari satu atau lebih komponen.
Node 	Biasanya mengacu pada perangkat keras (*hardware*), perangkat lunak (*software*), jika di dalam node disertakan komponen untuk mengkonsistensikan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelumnya pada diagram komponen.
Kebergantungan/*dependency* 	Kebergantungan antar *node*, arah panah mengarah pada *node* yang dipakai.
Link	Relasi antar *node*.

Sumber: (Rosa & Shalahuddin, 2015:154)

2.2.2. *Entity Relationship Diagram* (ERD)

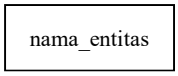
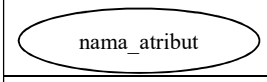
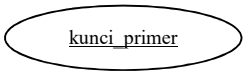
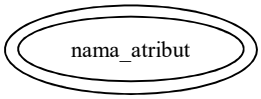
Menurut Rosa & Shalahuddin (2015:53) mendefinisikan bahwa "ERD adalah bentuk paling awal dalam melakukan perancangan basis data relasional. Jika menggunakan OODMBS maka perancangan ERD tidak perlu dilakukan". *Logical record structure* (LRS) merupakan representasi dari struktur *record-record* pada

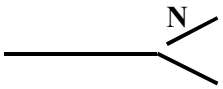
tabel dimana tabel-tabel tersebut terbentuk dari hasil himpunan antar entitas pada *entity relationship diagram* yang telah ditransformasikan menjadi bentuk LRS (Pratama, Sihombing & Putra, 2014).

Dapat disimpulkan bahwa *entity relationship diagram* (ERD) merupakan teknik pemodelan struktur data secara konseptual yang menggambarkan entitas lengkap dengan atributnya dan hubungan yang terjadi antar entitas tersebut.

Entity relationship diagram (ERD) digambarkan dengan simbol-simbol yang saling berhubungan (Rosa & Shalahuddin, 2015:50) yang diuraikan sebagai berikut:

Tabel II.5.
Komponen *Entity Relationship Diagram* (ERD)

Notasi	Komponen	Keterangan
	Entitas/*entity*	Entitas merupakan data inti yang akan disimpan, bakal tabel pada basis data, benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer. Penamaan entitas biasanya lebih ke kata benda dan belum merupakan nama tabel.
	Atribut	*Field* atau kolom data yang butuh disimpan dalam suatu entitas.
	Atribut kunci primer	*Field* atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan, biasanya berupa id. Kunci primer dapat lebih dari satu kolom, asalkan kombinasi dari beberapa kolom tersebut dapat bersifat unik (berbeda tanpa ada yang sama).
	Atribut multinilai/*multivalue*	*Field* atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki lebih dari satu.
	Relasi	Relasi yang menghubungkan antar entitas, diawali dengan kata kerja.

	Asosiasi/*association*	Penghubung antara relasi dan entitas dimana di kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian. Kemungkinan jumlah maksimum keterhubungan antara entitas satu dengan entitas yang lain disebut dengan kardinalitas. Misalkan ada kardinalitas 1 ke N atau sering disebut dengan *one to many* menghubungkan entitas A dan entitas B.

Sumber: (Rosa & Shalahuddin, 2015:50)

2.2.3. *Black Box Testing*

Black box testing merupakan teknik pengujian yang melakukan pendekatan kebutuhan dasar program dalam mengecek fungsional program (Swastika & Putra, 2016:73). Sedangkan menurut Rosa & Shalahuddin (2015:275) mengemukakan bahwa "*Black box testing* (pengujian kotak hitam) yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan".

Berdasarkan beberapa pendapat di atas, dapat ditarik kesimpulan bahwa *black box testing* merupakan suatu cara atau teknik yang digunakan dalam pengujian fungsional perangkat lunak agar terbebas dari kesalahan atau *error*.