BAB II

LANDASAN TEORI

2.1 Konsep Dasar

Sistem pada dasarnya adalah kelompok unsur yang erat hubungannya antara satu dengan yang lainnya, yang berfungsi bersama-sama untuk mencapai tujuan tertentu.

2.1.1 Pengertian Sistem

Menurut (Kadir, 2014), “Sistem adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan”.

Menurut (Rosa & Shalahuddin, 2018), “Sistem berarti kumpulan komponen yang saling terkait dan mempunyai satu tujuan yang ingin dicapai”.

Menurut (Mulyadi, 2016), “Sistem adalah sekelompok unsur yang erat berhubungan dengan satu dengan lainnya, yang berfungsi bersama – sama untuk mencapai tujuan tertentu”.

Penulis menyimpulkan bahwa sistem adalah suatu komponen yang saling terkait dan berinteraksi untuk mencapai tujuan.

2.1.2 Pengertian Informasi

Menurut (Chandra & Adriana, 2017), “Informasi adalah data yang telah diorganisir dan diproses sehingga bermanfaat bagi proses pengembalian keputusan”.

Menurut (Kadir, 2014), “Informasi adalah data yang diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau saat mendatang”.

Penulis menyimpulkan bahwa informasi adalah data yang diproses untuk pengambilan keputusan.

2.1.3 Karakteristik Informasi

Suatu informasi dikatakan bermanfaat menurut (Chandra & Adriana, 2017) yaitu :

1. Relevan, yaitu dapat mengurangi ketidakpastian, meningkatkan kualitas pengambilan keputusan, serta mengkonfirmasi atau mengkoreksi ekspektasi awal.
2. Andal, yaitu bebas dari kesalahan atau bias.
3. Lengkap, informasi dikatakan lengkap jika tidak menghilangkan aspek penting dari suatu kejadian atau aktivitas yang diukur.
4. Tepat waktu, yaitu tersedia saat diperlukan untuk mengambil keputusan.
5. Dapat dipahami. Disajikan dalam format yang mudah dipahami dan bermanfaat.
6. Dapat diverifikasi. Jika informasi tersebut dibaca oleh dua orang berbeda yang berpengetahuan memadai akan menghasilkan informasi yang sama.
7. Dapat diakses oleh pengguna jika diperlukan.

2.1.4 Sistem Informasi

Menurut (Kadir, 2014), “Sistem Informasi adalah sejumlah komponen (manusia, komputer, teknologi informasi, dan prosedur kerja), ada sesuatu yang diproses (data menjadi informasi), dan dimaksudkan untuk mencapai suatu sasaran atau tujuan.

2.1.5 Sistem Informasi Akuntansi (SIA)

Menurut(TM Books, 2017) merupakan sistem yang mengumpulkan, mencatat, menyimpan, dan memproses data sehingga menghasilkan informasi bagi para pengambil keputusan".

Menurut Bodnar dan Hopwood dalam (Kadir, 2014) menyebutkan bahwa " Sistem Informasi Akutansi sebagai kumpulan sunber daya yang dirancang untuk mentransformasikan data keuangan menjadi informasi.

Menurut Gelinas, Orams, dan Wiggins dalam (Kadir, 2014), "Sistem Informasi Akuntansi (SIA) adalah menghimpun, memproses dan melaporkan informasi yang berkaitan dengan transaksi keuangan"

Penulis menyimpulkan bahwa sistem informasi akuntansi adalah sebuah sistem yang berguna untuk pengambilan keputusan keuangan.

2.1.6 Komponen SIA

Menurut (Chandra & Adriana, 2017), SIA memiliki enam komponen yaitu:

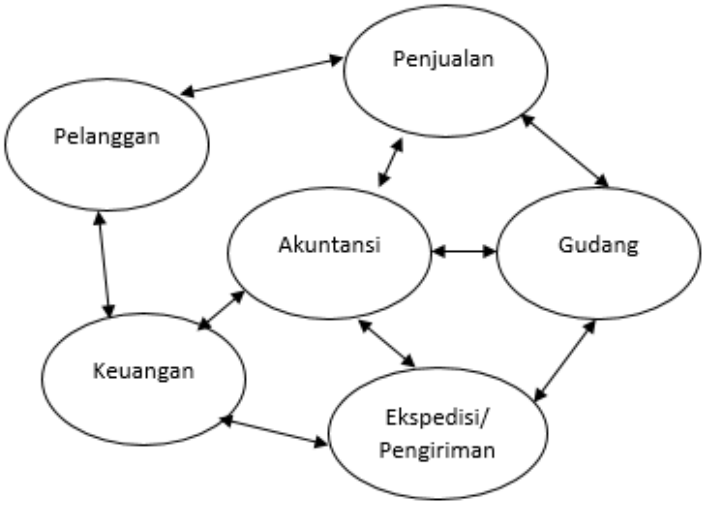
1. *User* yang menggunakan sistem.
2. Prosedur dan instruksi yang digunakan untuk mengumpulkan, memproses, dan menyimpan data.
3. Data mengenai organisasi dan aktivitas bisnisnya.
4. *Software* yang digunakan untuk memproses data
5. Infrastruktur teknologi informasi, yang terdiri dari komputer, *peripheral device*, dan perangkat jaringan
6. Pengendalian internal untuk menjaga keamanan data SIA

2.1.7 Siklus Penjualan

Menurut (Ardana & Lukman, 2016) memberikan batasan bahwa “Siklus penjualan merupakan suatu rangkaian kegiatan penjualan yang terjadi secara berulang-ulang dan diikuti dengan proses perekaman data dan informasi bisnis”.

Menurut (Ardana & Lukman, 2016) memberikan batasan bahwa,”Dalam siklus penjualan, kontrak pertama pelanggan adalah dengan fungsi penjualan”.

1. Melayani pertanyaan dan memberikan informasi produk tentang produk kepada calon pelanggan.
2. Menerima order pembelian dari pelanggan.
3. Berkoordinasi dengan fungsi keuangan untuk proses persetujuan kredit.
4. Menyiapkan kontrak penjualan dan/atau order penjualan.
5. Berkoordinasi dengan fungsi gudang untuk mengetahui informasi tentang status barang dan penyiapan barang.
6. Berkoordinasi dengan fungsi pengangkutan untuk proses pengiriman barang.
7. Menyiapkan faktur penjualan.



Sumber: (Ardana & Lukman, 2016)

Gambar II.1
Fungsi-Fungsi terkait dalam Siklus Penjualan

Menurut (Heri, 2016) Ada 2 ayat yang perlu dibuat sekaligus oleh penjual pada saat melakukan transaksi penjualan, yaitu:

Kas	xxx	
Penjualan		xxx

(apabila penjualan barang dagangan dilakukan secara tunai)

Piutang Usaha	xxx	
Penjualan		xxx

(apabila penjualan barang dagangan dilakukan secara kredit)

Harga Pokok Penjualan	xxx	
Persediaan Barang Dagangan		xxx

2.2 Peralatan Pendukung

Peralatan pendukung merupakan alat yang digunakan untuk menggambarkan bentuk logika model dari suatu sistem dengan menggunakan simbol-simbol lambang dan diagram. Adapun peralatan pendukung untuk merancang model sistem, terdiri dari:

2.2.1 Model SDLC *(Software Development Life Cycle)*

> Menurut (Rosa & Shalahuddin, 2018) mengemukakan bahwa :
> SDLC atau *Software Development Life Cycle* atau sering disebut juga *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (berdasarkan *best practice* atau cara-cara yang sudah teruji baik*).*

Menurut O'Brien dalam (Rosa & Shalahuddin, 2018) mengemukakan bahwa, "*System Development Life Cycle* (SDLC) adalah suatu metodologi yang digunakan untuk mengembangkan, memelihara, dan menggunakan sistem informasi".

Tahapan-tahapan yang ada pada SDLC secara global menurut (Rosa & Shalahuddin, 2018) adalah sebagai berikut :

1. Inisiasi (*initiation*)

   Tahap ini biasanya ditandai dengan pembuatan proposal proyek perangkat lunak

2. Pengembangan konsep sistem (*system concept development)*

   Mendefinisikan lingkup konsep termasuk dokumen lingkup sistem, analisis manfaat biaya, manajemen rencana, dan pembelajaran kemudahan sistem

3. Perencanaan (*planning)*

   Mengembangkan rencana manajemen proyek dan dokumen perencanaan lainnya. Menyediakan dasar untuk mendapatkan sumber daya *(resources)* yang dibutuhkan untuk memperoleh solusi.

4. Analisis kebutuhan (*requirements analysis)*

   Menganalisis kebutuhan pemakai sistem perangkat lunak *(user)* dan mengembangkan kebutuhan *user*. Membuat dokumen kebutuhan fungsional.

5. Desain (*design*)

   Mentransformasikan kebutuhan detail menjadi kebutuhan yang sudah lengkap, dokumen desain sistem fokus pada bagaimana dapat memenuhi fungsi-fungsi yang dibutuhkan.

6. Pengembangan (*development)*

   Mengonversi desain ke sistem informasi yang lengkap termasuk bagaimana memperoleh dan melakukan instalasi lingkungan sistem yang dibutuhkan; membuat basis data dan mempersiapkan prosedur kasus pengujian; mempersiapkan berkas atau *file* pengujian, pengodean, pengompilasian, memperbaiki dan membersihkan program; peninjauan pengujian.

7. Integrasi dan pengujian (*integration and test)*

   Mendemonstrasikan sistem perangkat lunak bahwa telah memenuhi kebutuhan yang dispesifikasikan pada dokumen kebutuhan fungsional. Dengan diarahkan oleh staf penjamin kualitas (*quality assurance)* dan *user*. Menghasilkan laporan analisis pengujian.

8. Implementasi (*implementation)*

   Termasuk pada persiapan implementasi, implementasi perangkat lunak pada lingkungan produksi (lingkungan pada *user*) dan menjalankan resolusi dari permasalahan yang teridentifikasi dari fase integrasi dan pengujian

9. Operasi dan pemeliharaan *(operations and maintenance)*

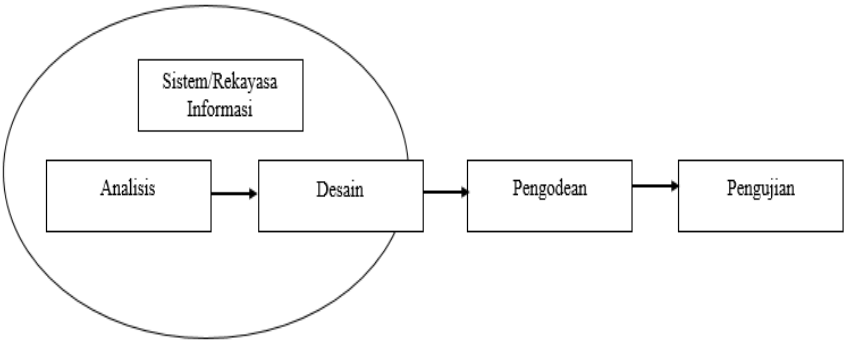
   Mendeskripsikan pekerjaan untuk mengoperasikan dan memelihara sistem informasi pada lingkungan produksi (lingkungan pada *user*), termasuk implementasi akhir dan masuk pada proses peninjauan.

10. Disposisi *(disposition)*

    Mendeskripsikan aktifitas akhir dari pengembangan sistem dan membangun data yang sebenarnya sesuai dengan aktifitas *user.*

2.2.2 Model *Waterfall*

Menurut (Rosa & Shalahuddin, 2018) Model SDLC air terjun (*Waterfall*) sering juga disebut model sekuensi linier (*sequential linear*) atau alur hidup klasik (*classic life cycle*). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengkodean, pengujian dan tahap pendukung (*support)*.Berikut adalah gambar model air terjun:



Sumber : (Rosa & Shalahuddin, 2018)

Gambar II.2
Ilustrasi Model *Waterfall*

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 multilangkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, 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 didokumentasikan

3. Pembuatan kode program

   Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan 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*)

   Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus adaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.2.3 *Unified Modelling Language (UML)*

Menurut(Rosa & Shalahuddin, 2018), "UML (*Unified Modelling Language)* adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement,* membuat analisis & desain, serta menggambarkan arsitektur dalam pemprograman berorientasi objek".

Menurut (Munawar, 2018), "UML (*Unified Modelling Language)* adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek".

Berikut ini adalah definisi mengenai 4 diagram UML yaitu:

1. *Use Case* Diagram

   Menurut (Rosa & Shalahuddin, 2018) mengatakan bahwa "*Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat".

   Menurut (Munawar, 2018) mengatakan bahwa "*Use case* adalah deskripsi fungsi dari sebuah *system* dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah *system*dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah *system* dipakai".

2. *Activity* Diagram

   Menurut (Rosa & Shalahuddin, 2018)mengatakan bahwa, "diagram aktivitas atau *activity* diagram mengambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak".

   Menurut (Munawar, 2018), " Activity Diagram adalah bagian penting dari UML yang menggambarkan aspek dinamis dari sistem".

3. *Deployment* Diagram

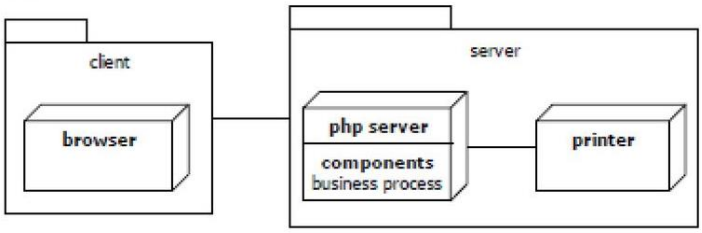
   Menurut(Rosa & Shalahuddin, 2018), "diagram *deployment* atau *deployment* diagram menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi

   Menurut (Munawar, 2018), " *Deployment* Diagram menunjukkan tata letak sebuah sistem secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian hadware".

   Menurut (Rosa & Shalahuddin, 2018) diagram *deployment* juga dapat digunakan untuk memodelkan hal-hal berikut :

a. sistem tambahan (*embedded system)* yang menggambarkan rancangan *device, node*, dan *hardware*

b. sistem *client/server*



Sumber : (Rosa & Shalahuddin, 2018)

Gambar II.3
Diagram *Deployment* Sistem *Client / Server*

c. sistem terdistribusi murni

d. rekayasa ulang aplikasi

4. *Sequence* Diagram

Menurut(Rosa & Shalahuddin, 2018), "diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Menurut (Munawar, 2018), "*Sequence* diagram digunakan untuk menggambarkan perilaku pada sebuah *scenario*".

2.2.4 *Entity Relationship Diagram* (ERD)

Menurut(Rosa & Shalahuddin, 2018), "*Entitiy Relationship Diagram* (ERD) adalah pemodelan awal basis data yang dikembangkan berdasarkan teori himpunan dalam bidang matematika untuk pemodelan basis data relasional".

Menurut (Lubis, 2016), "ER-D adalah suatu pemodelan berbasis pada persepsi dunia nyata yang mana terdiri dari kumpulan objek dasar yang disebut dengan entitas *(entity)* dan hubungan diantara objek-objek tersebut dengan menggunakan perangkat konseptual dalam diagram".

Menurut (Lubis, 2016), dalam ER-D hubungan antara entitas dapat dipetakan menjadi beberapa pembatas, yaitu:

1. Satu-ke-satu atau *one-to-one* (1-1)

   Pembacaan pemetaan satu-ke-satu dalam ER-D, berarti bahawa tiap entitas akan berhubungan dengan paling banyak satu entitas yang lain, demikian sebaliknya.

2. Satu-ke-banyak atau *one-to-many* (1-M / N)

   Pembacaan pemetaan satu-ke-banyak adalah atribut dapat berhubungan dengan lebih dari satu (banyak) atribut yang lain, tetapi tidak sebaliknya lebih dari satu (banyak) atribut hanya berhubungan dengan satu atribut yang lain.

3. Banyak-ke-satu atau *many-to-one* (M/N-1)

   Hubungan banyak-ke – satu merupakan kebalikan dari hubungan satu ke banyak, yaitu banyak (lebih dari satu) entitas yang satu akan berhubungan dengan hanya satu pada entitas yang lain, namun tidak sebaliknya.

4. Banyak-ke-banyak atau *many-to-many* (M-N)

   Pembacaan pemetaan banyak-ke – banyak, ini berarti banyak entitas dapat dihubungkan dengan banyak entitas yang lain, demikian sebaliknya.

2.2.5 *Logical Record Structure* (LRS)

Menurut (Tabrani, 2014) "*Logical Record Structure* dibentuk dengannomor dari tipe *record*. Beberapa tipe *record*digambarkan oleh kotak empat persegi panjang dandengan nama yang unik. Perbedaan LRS dengan E-Rdiagram adalah nama tipe *record* berada diluar kotak*field* tipe *record* ditempatkan".

Menurut(Tabrani, 2014) *"Logical Record Structure* terdiri dari *link-link* diantara tipe *record*. Link ini menunjukkan arah dari satu tipe *record* lainnya. Banyak *link* dari LRS yang diberi tanda *field-field* yang kelihatan pada kedua *link* tipe *record*. Penggambaran LRS mulai dengan menggunakan model yang dimengerti. Dua metode yang dapat digunakan, dimulai dengan hubungan kedua model yang dapat dikonverensikan ke LRS, metode yang lain dimulai dengan ER-diagram dan langsung dikonversikan ke LRS".

2.2.6 *Database*

Menurut(Rosa & Shalahuddin, 2018), "Basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat".

Menurut (Chandra & Adriana, 2017), "*database* merupakan sekumpulan file data yang saling berkaitan dan terkoordinasi yang disimpan dengan pengulangan data (*data redundancy*) sekecil mungkin".

Menurut (Enterprise, 2016), "*database* adalah sebuah sistem yang berfungsi untuk menyimpan dan mengolah sekumpulan data".

Penulis menyimpulkan bahwa *database* adalah sekumpulan file untuk menyimpan data yang saling keterkaitan dan dapat didapat mudah dan cepat.

2.2.7 Manfaat *Database*

Menurut (Chandra & Adriana, 2017), terdapat manfaat *database* antara lain:

1. Integrasi data

   Master file digabung menjadi satu kelompok data yang dapat diakses oleh beberapa program aplikasi.

2. *Sharing* data

   Data yang terintegrasi lebih mudah di*share* ke *user* yang memiliki otorisasi. *Database* lebih mudah di*browse* untuk menemukan masalah atau memperoleh informasi rinci yang menjadi dasar suatu laporan.

3. Meminimalkan pengulangan data *(data redundancy)* dan inkonsistensi.

   Oleh karena setiap *item* data disimpan hanya sekali, maka pengulangan data dan inkonsistensi dapat diminimalkan.

4. Independensi data

   Data dan program yang menggunakan data tersebut saling independen. Oleh karena itu, bila terjadi perubahan sebuah data, maka data yang lain tidak ikut berubah. Hal ini mempermudah pemprograman dan menyederhanakan manajemen data.

5. Dapat melakukan analisis lintas fungsi *(cross-functional)*

   Dalam sistem *database*, relasi dapat dibuat secara eksplisit dan digunakan dalam penyusunan laporan manajemen.

2.2.8 PHP

Menurut (Enterprise, 2018), “PHP merupakan bahasa pemrograman yang digunakan untuk membuat *website* dinamis dan interaktif”.

Menurut (Winarno et al., 2014), ”PHP adalah bahasa *strip* yang sangat cocok untuk pengembangan web dan dapat dimaksukkan kedalam HTML”,

Penulis menyimpulkan bahwa PHP adalah bahasa pemprograman yang digunakan untuk membuat *website* secara dinamis dan interaktif serta dapat dimasukkan kedalam HTML

2.2.9 XAMPP

Menurut (Enterprise, 2018) , “XAMPP merupakan *server* yang paling banyak digunakan untuk keperluan belajar PHP secara mandiri, terutama bagi *programmer* pemula.

2.2.10 MySQL

Menurut (Enterprise, 2018),“ My SQL merupakan *server* yang melayani *database* untuk membuat dan mengolah *database*”.

Menurut (Winarno, Ali, & SmithDev, 2014) , “MySQL merupakan tipe data relasional yang artinya MySQL menyimpan datanya dalam bentuk tabel-tabel yang saling berhubungan”.

Menurut (Enterprise, 2014), ”MySQL adalah RDBMS yang cepat dan mudah digunakan, serta sudah banyak digunakan untuk berbagai kebutuhan”.

Penulis dapat menyimpulkan bahwa MySQL adalah server yang melayani *database* untuk membuat dan megolah *database* yang saling berhubungan, serta cepat dan mudah digunakan.

2.2.11 Pengujian Perangkat Lunak

Menurut(Rosa & Shalahuddin, 2018), “Pengujian adalah satu set aktifitas yang direncanakan dan sistematis untuk menguji atau mengevaluasi kebenaran yang diinginkan”.

Menurut(Rosa & Shalahuddin, 2018), “ Black-Box Testing (pengujian kotak hitam) yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program”. Pengujian dimaksudkan untuk menguji perangkat dengan spesifikasi yang dibutuhkan.

Penulis dapat menyimpulkan bahwa pengujian perangkat lunak adalah menguji atau mengevaluasi desain dan kode program untuk mengetahui kualitas dari program yang sedang diuji.