BAB II

LANDASAN TEORI

II.1. Sistem Informasi Akuntansi

Menurut (Anastasia Diana & Lilis setiawati : 2011; 04) Sistem informasi akuntansi adalah sistem yang bertujuan untuk mengumpulkan dan memproses data serta melaporkan informasi yang berkaitan dengan transaksi keuangan.

II.2. Penyusutan

Menurut (Rudianto : 2012 : 260) Penyusutan adalah pengalokasian harga perolehan asset tetap menjadi beban ke dalam periode akuntansi yang menikmati manfaat dari asset tetap tersebut.

II.2.1 Faktor yang berpengaruh

Menurut (Rudianto 2012 : 260-261) ada tiga faktor yang perlu dipertimbangkan dalam menentukan beban penyusutan setiap periode, yaitu :

1. Harga perolehan

   Yaitu keseluruhan uang yang dikeluarkan untuk memperoleh suatu asset tetap sampai siap digunakan oleh perusahaan.

2. Nilai sisa (Residu)

   Yaitu taksiran harga jual asset tetap pada akhir masa manfaatnya. Setiap perusahaan akan memiliki taksiran yang berbeda satu dengan laninnya atas suatu jenis asset tetap yang sama. Jumlah taksiran nilai residu juga akan sangat dipengaruhi oleh umur ekonomisnya, inflasi, nilai tukat mata uang, bidang usaha, dan sebaginya.

3. Taksiran umur kegunaan

   Yaitu taksiran masa manfaat dari asset tetap. Masa manfaat adalah taksiran umur ekonomis dari aset tetap, bukan umur teknis. Taksiran masa manfaat dapat dinyatakan dalam satuan periode waktu, satuan hasil produksi, atau satuan jam kerja.

II.3. Metode Garis Lurus

Menurut (Rudianto 2012 : 260) Metode garis lurus adalah metode perhitungan penyusutan aset tetap di mana setiap periode akuntansi diberikan beban yang sama secara merata. Beban penyusutan dihutung dengan cara mengurangi harga perolehan dengan nilai sisa dan dibagi dengan umur ekonomis aset tetap tersebut.

$$\textbf{Penyusutan} = \frac{\textbf{Harga Perolehan – Nilai Sisa}}{\textbf{Taksiran Umur Ekonomis Aset}}$$

(***Sumber: Rudianto 2012 : 260***)

Metode perhitungan penyusutan garis lurus akan menghasilkan beban penyusutan aset tetap yang sama dari tahun ke tahun. Metode ini juga dapat menghasilkan beban penyusutan berupa suatu persentase dari harga perolehan aset tetap.

II.4. Microsoft Visual Basic 2010

Menurut (Stefano,S.Kom: 2014 : 02) *Microsoft* visual basic. NET adalah sebuah alat untuk mengembangkan dan membangun aplikasi yang bergerak di atas sistem.NET *Framework,* dengan menggunakan bahasa basic. Dengan alat ini, para programmer dapat membangun aplikasi *Windows Forms,* Aplikasi web berbasis ASP.NET, dan aplikasi *command-line.* Alat ini dapat diperoleh secara terpisah dari beberapa produk lainya seperti *Microsoft* Visual C++, visual C#, atau Visual J# atau juga dapat diperolehan secara terpadu dalam *Microsoft* Visual Studio.NET.

Menurut (Christopher Lee ; 2014 :01) Visual basic 2010 adalah inkarnasi dari bahasa visual basic yang sangat populer dan telah dilengkapi dengan fitur serta fungsi yang setara dengan bahasa tingkat tinggi seperti C++.

II.5. SQL Server 2008

Menurut (Stefano,S.Kom: 2014 : 04) *Microsoft* SQL Server adalah sebuah system manajemen basis data relasional (RDBMS) produk *Microsoft.* Bahasa kueri utamnya adalah Transact-SQL yang merupakan implementasi dari SQL standar ANSI/ISO yang digunakan oleh *Microsoft* dan *Sybase.* Umumnya, SQL Server digunakan didunia bisnis yang memiliki basis data berskala kecil sampai menengah, tetapi kemudian berkembang dengan digunakannya SQL Server pada basis data besar.

Microsoft SQL Server juga mendukung ODBC (*Open Database Connectivity),* dan mempunyai *driver* JDBC untuk bahasa pemrograman Java. Fitur lain SQL Server adalah kemampuannya membuat basis data *mirroring* dan *clustering.*

II.6. Basis Data

Menurut (Edhy Sutanta :2011: 29) Basis data dapat dipahami sebagai suatu kumpulan data terhubung (*interrelated data)* yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data (kalaupun ada maka kerangkapan data tersebut harus seminimal mungkin dan terkontrol), data disimpan dengan cara-cara tertentu sehingga mudah digunakan/atau ditampilkan kembali data dapat digunakan oleh satu atau lebih program-program aplikasi secara optimal, data disimpan tanpa mengalami ketergantungan dengan program yang akan menggunakannya data disimpan sedemikian rupa sehingga proses penambahan, pengambilan, dan modifikasi data dapat dengan mudah dan terkontrol.

II.6.1. Pemodelan Data

Menurut (Yudi Priyadi; 2014 ;10) Terdapat beberapa penjelasan mengenai pemodelan basis data. Suatu basis data dapat digunakan secara bebas untuk menggambarkan dan memberikan deskripsi mengenai kumpulan informasi uang tersimpan dalam data *storage* computer. Secara sederhana, definisi untuk model basis data adalah sekumpulan notasi atau simbol untuk menggambarkan data dan relasinya berdasarkan suatu konsep dan aturan tertentu suatu pemodelan.

II.6.2. Notasi Diagram E-R

Menurut (Yudi Priyadi; 2014 ;20) Pemodelan basis data dengan menggunakan diagram relasi antar entitas, dapat dilakukan dengan menggunakan suatu pemodelan basis data yang bernama Diagram *Entity-Relational* (selanjutnya disingkat Diagram E-R). Pada Diagram E-R terdapat suatu simbol/notasi dasar yang digunakan pada Diagram E-R, yaitu entitas, relasi, atribut, dan garis penghubung.

1. Entitas

   Merupakan notasi untuk mewakili suatu objek dengan karakteristik sama, yang dilengkapi oleh atribut, sehingga pada suatu lingkungan nyata setiap objek akan berbeda dengan objek lainnya. Pada umumnya, objek dapat berupa benda, pekerjaan, tempat dan orang.

2. Atribut

   Merupakan notasi yang menjelaskan karakteristik suatu entitas dan juga relasinya. Atribut dapat sebagai key yang bersifat unik, yaitu *Primary Key* atau *Foreign Key.* Selain itu, atribut juga dapat sebagai pelengkap deskripsi suatu entitas dan relasi.

3. Relasi

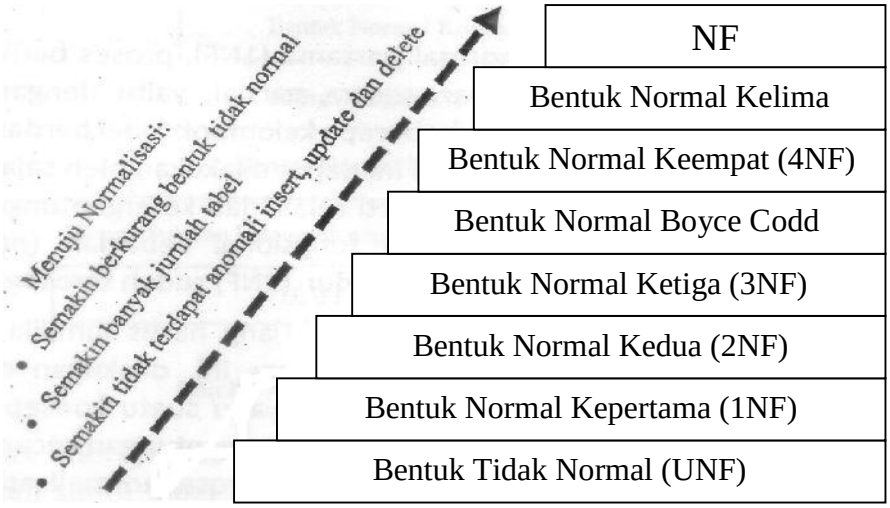
   Merupakan notasi yang digunakan untuk menghubungkan beberapa entitas berdasarkan fakta pada suatu lingkungan.

4. Garis Penghubung

   Merupakan notasi untuk merangkai keterkaitan antar notasi yang digunakan dalam Diagram E-R, yaitu entitas, relasi dan atribut.

II.6.3. Normalisasi

Menurut (Yudi Priyadi ; 2014 ;67) Normalisasi merupakan proses sistematis yang dilakukan pada struktur table basis data menjadi struktur table yang memiliki integritas data, sehingga tidak memiliki data anomaly pada saat melakukan *insert, delete,* dan *update.* Pada Gambar II.1, tahapan proses sistematis yang dilakukan mulai dari bentuk tidak normal menjadi bentuk normal memiliki suatu syarat yang harus dipenuhi pada saat menuju suatu bentuk yang lebih baik (*well structured relation).*



Gambar II.1 : Tahapan Proses Bentuk Normalisasi
(***Sumber : Yudi Priyadi ; 2014 : 67***)

Setiap syarat dalam tahapan suatu bentuk normal memiliki keterkaitan, hal ini disebabkan karena pada setiap bentuk normal mengalami penyempurnaan untuk bentuk normal selanjutnya. Bentuk tidak normal akan semakin berkurang, setelah melalui tahapan perubahan bentuk normalisasi, sehingga berdampak pada jumlah table yang semakin banyak, tetapi menuju perbaikan kedalam bentuk *well*

structured relation. Hal ini terjadi akibat dari pengelompakan data suatu table agar memiliki ketergantungan secara fungsional.

II.7. *Unified Modelling Language* (UML)

Menurut (Prabowo Pudjo Widodo & Herlawati ; 2011 ; 118) Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak. UML merupakan bahsa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram teks-teks pendukung.

UML hanya berfungsi untuk melakukan pemodelan, jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metode berorientasi objek.

II.7.1. Diagram-Diagram UML

Menurut (Prabowo Pudjo Widodo & Herlawati ; 2011 ; 120) Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokan dalam 3 kategori. Berikut ini penjelasan singkat dari pembagian kategori tersebut :

1. *Structure Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu stuktur statis dari sistem yang dimodelkan.

2. *Behavior* diagram yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction Diagrams* yaitu kumpulan diagram yang digunakanuntuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

A. *Class* Diagram

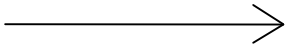
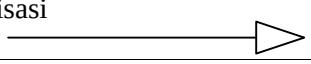
Diagram kelas atau diagram *class* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

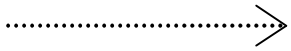
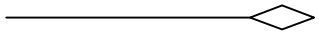
1) Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.

2) Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Berikut adalah keterangan dari simbol-simbol yang ada pada diagram kelas :

Tabel II.1. Diagram Kelas

Simbol	Deskripsi
Kelas Nama_kelas +atribut +operasi()	Kelas pada struktur sistem.
Antarmuka / *interface*  Nama_*interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
Asosiasi / *association* Asosiasi berarah/directed association 	Relasi antar kelas dengan makna umum, assosiasi biasanya juga disertai dengan *multiplicity.* Relasi antar kelas dengan makna kelas yang lain, assosiasi biasanya juga disertai dengan *multiplicity.*
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)

Kebergantungan/ dependency 	Relasi antar kelas dengan makna kebergantungan antar kelas.
Agregasi / *aggregation* 	Relasi antar kelas dengan makna semua bagian (*whole part)*

(***Sumber : Prabowo Pudjo Widodo & Herlawati ; 2011 : 124***)

B. *Object Diagram*

Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. Pada diagram objek harus dipastikan semua kelas yang sudah didefenisikan pada diagram kelas harus dipakai objeknya, karena jika tidak, pendefenisian kelas itu tidak dapat dipertanggungjawabkan. Untuk apa mendefenisikan sebuah kelas sedangkan pada jalannya sistem, objeknya tidak pernah dipakai.

C. *Component Diagram*

Diagram komponen atau component diagram dibuatuntuk menunjukkan organisasi dan ketergantungan di antara kumpulan komponen dalam sebuah sistem. Diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada didalam sistem. Komponen dasar yang biasanya ada dalam suatu sistem adalah sebagai berikut :

1. Komponen *user interface* yang menangani tampilan.
2. Komponen *bussiness procesiing* yang menangani fungsi-fungsi proses bisnis.
3. Komponen data yang menangani manipulasi data.
4. Komponen *security* yang menangani keamanan sistem.

Komponen lebih terfokus pada penggolongan secara umum fungsi-fungsi yang diperlukan.

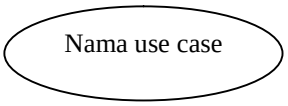
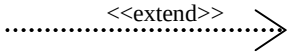
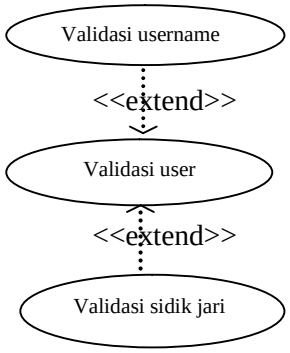
D. *Use Case* Diagram

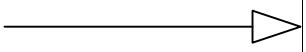
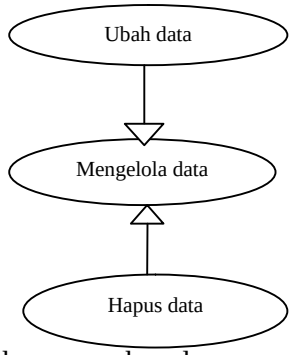
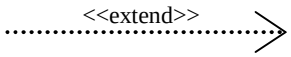
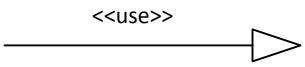
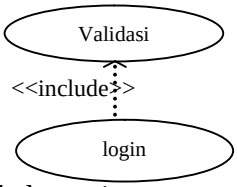
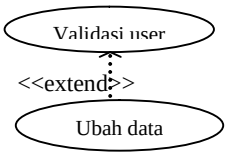
Use Ccase atau diagram *use case* merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua utama pada *use case* yaitu pendefinisian apa yang disebut actor dan *use case.*

1. Actor merupakan orang, proses, atau sistem lain yang berinertaksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tetapi actor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau actor.

Berikut adalah keterangan dari simbol-simbol yang ada pada diagram *use case :*

Tabel II.2. Diagram *Use Case*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal *frase* nama *use case.*
Aktor/ actor  Nama aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belumtentu merupakan orang, biasanya dinyatakan menggunakan kata benda diawal *frase* nama aktor.
Asosiasi / *association*	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi /extend 	Relasi *use case* akan tambah kesebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambah memiliki nama depan yang sama dengan *use case* yang ditambahkan, missal  Arah panah mengarah pada *use case* yang ditamabahkan..

Generalisasi / generalization 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yanglebih umum dari lainnya, misalnya :  Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
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. Ada dua sudut pandang yang cukup besar mengenai *include* di *use case :* • *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, missal pada kasus berikut :  • *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan, missal pada kasus berikut :  Kedua interprestasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

(***Sumber : Prabowo Pudjo Widodo & Herlawati ; 2011 : 131***)

E. Communication Diagram

Diagram komunikasi mengelompokkan message pada kumpulan diagram sekuen menjadi sebuah diagram. Dalam diagram komunikasi yang dituliskan adalah operasi / metode yang di jalankan antara objek yang satu dengan objek lainnya secara keseluruhan, oleh karna itu dapat di ambil dari jalanya interaksi pada semua diagram sekuen.

F. *Activity* Diagram

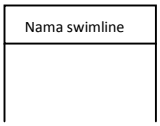
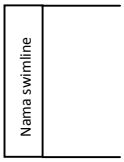
Diagram *activity* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

1. Rancangan proses bisnis di mana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari system/*user interface* di mana setiap aktivitas dianggap memiliki sebuah rancangan antar muka tampilan.
3. Rancangan pengujian di mana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikankasus ujinya.

Berikut adalah keterangan dari simbol-simbol yang ada pada diagram aktivitas :

Table II.3. Diagram *Activity*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja

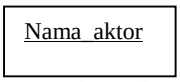
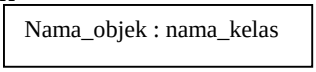
Percabangan / decesion 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / join	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane  Atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

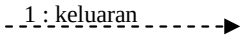
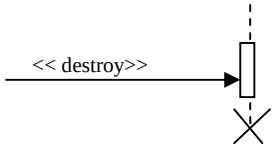
(***Sumber : Prabowo Pudjo Widodo & Herlawati ; 2011 : 134***)

G. *Sequence* Diagram

Sequence Diagram atau diagram sekuen menggambarkan kelakukan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *messasge* yang dikirimkan dan diterima antar objek. Banyaknya diagram sekuen yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dbuat juga semakin banyak. Berikut adalah keterangan dari simbol-simbol yang ada pada digram sekuen :

Tabel II.4. *Sequence* Diagram

Simbol	Deskripsi
Aktor/ actor  Atau  Tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belumtentu merupakan orang, biasanya dinyatakan menggunakan kata benda diawal *frase* nama aktor.
Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
Objek  	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.
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,  1 : nama_metode() 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 dikirim.

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 kembali.
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 : Prabowo Pudjo Widodo & Herlawati ; 2011 : 138***)