BAB II

TINJAUAN PUSTAKA

II.1. Sistem, Informasi dan Sistem Informasi

II.1.1. Pengertian Sistem

Terdapat dua kelompok pendekatan di dalam mendefinisikan sistem, yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Pendekatan sistem yang lebih menekankan pada prosedur mendefinisikan sistem adalah suatu suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu. Sedangkan pendekatan yang menekankan pada elemen atau komponennya mendefinisikan sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu (Jogiyanto; 2005:1-2).

Selain itu, sistem dapat juga diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen atau variabel yang terorganisir, saling berinteraksi, saling bergantung satu sama lain dan terpadu (Tata Sutabri; 2005:2).

II.1.2. Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu yaitu mempunyai komponen – komponen (*components*), batas sistem (*boundary*), lingkungan luar sistem (*environment*), penghubung (*interface*), masukan (*input*), keluaran (*output*), pengolah (*process*) dan sasaran (*objectives*) atau tujuan (*goal*).

1. Komponen Sistem

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. Setiap sistem tidak perduli betapapun kecilnya, selalu mengandung komponen-komponen atau subsistem-subsistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai suatu sistem yang lebih besar yang disebut dengan *supra system.* Misalnya suatu perusahaan dapat disebut dengan suatu sistem dan industri yang merupakan sistem yang lebih besar dapat disebut dengan *supra system.* Jika dilihat industri sebagai suatu sistem, maka perusahaan dapat disebut sebagai subsistem. Demikian juga bila perusahaan dilihat sebagai suatu sistem, maka sistem akuntansi adalah subsistemnya. Jika sistem akuntansi dilihat sebagai suatu sistem, maka perusahaan adalah *supra system* dan industri adalah *supra* dari *supra system.*

2. Batas Sistem

   Batas sistem (*boundary*) merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem

   Lingkungan luar (*environment*) dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi dari sistem dan dengan demikian harus tetap dijaga dan dipelihara. Sedangkan lingkungan luar yang merugikan harus ditahan dan dikendalikan, kalau tidak maka akan mengganggu kelangsungan hidup dari sistem.

4. Penghubung Sistem

   Penghubung (*interface*) merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lainnya. Keluaran (*ouput*) dari satu subsistem akan menjadi masukan (*input*) untuk subsistem yang lainnya dengan melalui penghubung. Dengan penghubung satu subsistem dapat berintegrasi dengan subsistem yang lainnya membentuk satu kesatuan.

5. Masukan Sistem

   Masukan (*input*) adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan sinyal (*signal input*). *Maintenance input* adalah energi yang dimasukkan agar sistem tersebut dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapatkan keluaran. Sebagai contoh di dalam

komputer, program adalah *maintenance input* yang digunakan untuk mengoperasikan komputernya dan data adalah *signal input* untuk diolah menjadi informasi.

6. Keluaran Sistem

   Keluaran sistem (*output*) adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan. Keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada *supra system*. Misalnya untuk sistem komputer, panas yang dihasilkan adalah keluaran yang tidak berguna dan merupakan hasil pembuangan, sedang informasi adalah keluaran yang dibutuhkan.

7. Pengolah Sistem

   Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah masukan menjadi keluaran. Suatu sistem produksi akan mengolah masukan berupa bahan baku dan bahan-bahan yang lain menjadi keluaran berupa barang jadi. Sistem akuntansi akan mengolah data-data transaksi menjadi laporan-laporan keuangan dan laporan-laporan lain yang dibutuhkan oleh manajemen.

8. Sasaran Sistem

   Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu

sistem dikatakan berhasil bila mengenai sasaran atau tujuannya (Jogiyanto; 2005:3-5).

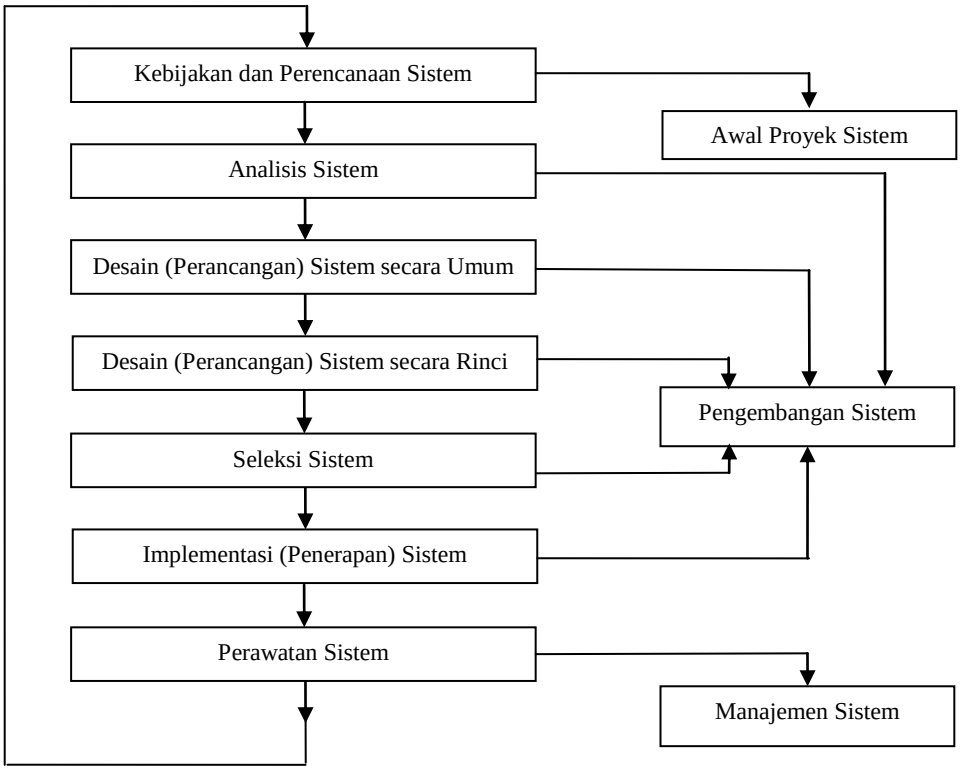
II.1.3. Pengembangan Sistem

Pengembangan sistem (*system development*) dapat berarti menyusun suatu sistem yang baru menggantikan sistem yang lama secara keseluruhan atau memperbaiki sistem yang telah ada (Jogiyanto; 2005:35).

Sebelum melakukan pengembangan sistem haruslah membuat skedul kerja yang berisi tahapan-tahapan kerja dan tugas-tugas pekerjaan yang akan dilakukan, sehingga proses pengembangan sistem dapat dilakukan dan selesai dengan berhasil sesuai dengan waktu dan anggaran yang direncanakan. Skedul kerja ini diwujudlkan dalam siklus hidup pengembangan sistem (*System Development Life Cycle* atau SDLC).

Dalam siklus ini tiap-tiap bagian dalam pengembangan sistem dibagi menjadi beberapa tahapan kerja. Tiap-tiap tahapan ini mempunyai karakteristik tersendiri. Tahapan utama siklus hidup pengembangan sistem dapat terdiri dari tahapan perencanaan sistem (*system planning*), analisis sistem (*system analysis*), desain sistem (*system design*), seleksi sistem (*system selection*), implementasi sistem (*system implementation*) dan perawatan sistem (*system maintenance*). Siklus hidup pengembangan sistem dari beberapa sistem informasi terkadang dapat sama atau mirip. Hal ini tidaklah kebetulan, karena proses pengembangan sistem informasi adalah proses teknik dan proses semacam ini harus mengikuti langkah-langkah yang sama serta prinsip-prinsip umum pengembangan sistem (Jogiyanto: 2005:41).

Siklus hidup pengembangan sistem dengan langkah-langkah utamanya dapat dilihat pada gambar II.1. berikut ini:



Gambar II.1. Siklus Hidup Pengembangan Sistem

Sumber : Jogiyanto; (2005:52)

II.1.4. Pengertian Informasi

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya. Sumber dari informasi adalah data. Data sendiri merupakan bentuk jamak dari bentuk tunggal datum atau data-*item*. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata (Jogiyanto; 2005:8).

Informasi adalah data yang telah diklasifikasikan, diolah atau diinterpretasi untuk digunakan dalam proses pengambilan keputusan. Sistem pengolahan

informasi mengolah data menjadi informasi atau tepatnya mengolah dari bentuk tak berguna menjadi berguna bagi penerimanya (Tata Sutabri; 2005:23).

II.1.5. Nilai dan Kualitas Informasi

Nilai dari informasi (*value of information*) ditentukan dari dua hal, yaitu manfaat dan biaya mendapatkannya. Suatu informasi dikatakan bernilai bila manfaatnya lebih efektif dibandingkan dengan biaya mendapatkannya. Akan tetapi perlu diperhatikan bahwa informasi yang digunakan di dalam suatu sistem informasi umumnya digunakan untuk beberapa kegunaan. Sehingga tidak memungkinkan dan sulit untuk menghubungkan suatu bagian informasi pada suatu masalah tertentu dengan biaya untuk memperolehnya, karena sebagian besar informasi dinikmati tidak hanya oleh satu pihak di dalam perusahaan. Lebih lanjut sebagian besar informasi tidak dapat persis ditaksir keuntungannya dengan suatu nilai uang, tetapi dapat ditaksir nilai efektivitasnya. Pengukuran nilai informasi biasanya dihubungkan dengan analisis *effectiveness* atau *cost benefit.*

Kualitas informasi dari suatu informasi (*quality of information*) tergantung dari tiga hal, yaitu :

a. Akurat, berarti informasi harus bebas dari kesalahan-kesalahan dan tidak bias atau menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya.
b. Tepat pada waktunya, berarti informasi yang datang pada penerima tidak boleh terlambat. Informasi yang telah usang tidak mempunyai nilai guna lagi.

c. Relevan, berarti informasi tersebut mempunyai manfaat untuk pemakainya. Relevansi informasi untuk tiap-tiap orang satu dengan yang lainnya berbeda (Jogiyanto; 2005:10-11).

II.1.6. Sistem Informasi

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategis dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan (Jogiyanto; 2005:11).

Sistem informasi dapat didefinisikan sebagai kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan untuk mengintegrasikan data, memproses dan menyimpan serta mendistribusikan informasi. Dengan kata lain, Sistem Informasi merupakan kesatuan elemen-elemen yang saling berinteraksi secara sistematis dan teratur untuk menciptakan dan membentuk aliran informasi yang akan mendukung pembuatan keputusan dan melakukan kontrol terhadap jalannya perusahaan (Budi Sutedjo Dharma Oetomo; 2006:11).

II.2. Pengolahan Data

Seperti yang dijelaskan pada pembahasan sebelumnya, informasi merupakan data yang diolah. Pengolahan data menjadi informasi ini merupakan suatu siklus, yang terdiri dari tahap-tahap berikut:

1. Pengumpulan data

   Pada tahap ini dilakukan suatu proses pengumpulan data yang asli dengan cara tertentu, seperti *sampling*, data transaksi, data *warehouse* dan lain sebagainya yang biasanya merupakan proses pencatatan data ke dalam suatu *file*.

2. *Input*

   Tahap ini merupakan proses pemasukan data dan prosedur pengolahan data ke dalam komputer melalui alat input seperti *keyboard*. Prosedur pengolahan data ini merupakan urutan langkah untuk mengolah data yang ditulis dalam suatu bahasa pemrograman yang disebut program.

3. Pengolahan Data

   Tahap ini merupakan tahap di mana data diolah sesuai dengan prosedur yang telah dimasukkan. Kegiatan pengolahan data ini meliputi pengumpulan data, klasifikasi (pengelompokan), kalkulasi, pengurutan, penggabungan, peringkasan baik dalam bentuk tabel maupun grafik, penyimpanan dan pembacaan data dari tempat penyimpan data.

4. *Output*

   Hasil pengolahan data akan ditampilkan pada alat *output* seperti *monitor* dan *printer* sebagai informasi.

5. Distribusi

   Setelah proses pengolahan data dilakukan, maka informasi yang dihasilkan harus segera didistribusikan. Proses pendistribusian ini tidak boleh terlambat dan harus diberikan kepada yang berkepentingan, sebab hasil

pengolahan data tersebut akan menjadi bahan pertimbangan dalam pengambilan keputusan atau menjadi data dalam pengolahan data selanjutnya (Budi Sutedjo Dharma Oetomo; 2006:11).

II.3. Lelang Barang Jaminan

Lelang barang jaminan adalah penjualan barang jaminan yang telah jatuh tempo di muka umum untuk mengembalikan kewajiban nasabah kepada Pegadaian. Barang jaminan merupakan salah satu syarat permohonan pinjaman (kredit gadai). Untuk dapat diketahui, bahwa kredit gadai adalah pemberian pinjaman (kredit) dalam jangka waktu tertentu kepada nasabah atas dasar hukum gadai dan persyaratan tertentu yang telah ditetapkan oleh perusahaan. Jangka waktu kredit gadai yang saat ini ditetapkan perusahaan adalah 120 hari (4 bulan). Maka jika dilihat dari pengertian lelang barang jaminan di atas, barang jaminan yang dapat dilelang adalah barang kasep dari segala jenis golongan kredit yang tidak dapat terlunasi melewati jangka waktu 120 hari (4 bulan).

Selain jangka waktu hal-hal yang harus diketahui yang berkaitan dengan barang jaminan dan kredit gadai di antaranya, barang yang diterima dan tidak diterima sebagai jaminan dan pelunasan kredit. Barang yang diterima sebagai jaminan diantaranya:

a. barang perhiasan (logam dan permata) seperti emas dan berlian,
b. kendaraan seperti mobil dan sepeda motor,
c. elektronik seperti TV dan *handphone*,
d. barang rumah tangga seperti mesin cuci, kulkas, blender dan lain-lain.

Sedangkan barang yang tidak diterima sebagai jaminan adalah:

a. barang-barang milik pemerintah,
b. barang-barang yang mudah membusuk,
c. barang berbahaya dan mudah terbakar,
d. barang yang sukar ditaksir nilainya,
e. barang yang dilarang peredarannya,
f. barang yang disewabelikan dan
g. barang hutang dan belum lunas serta barang titipan sementara.

Ketentuan pelunasan kredit gadai adalah nasabah harus menyelesaikan pinjamannya kepada PT. Pegadaian (Persero) sebagai pemberi pinjaman (kreditur) dengan cara, mengembalikan uang pinjaman dan membayar sewa modal atau penjualan lelang barang jaminannya (jika tidak bisa melunasi kredit). Sisa hasil penjualan lelang barang jaminan (harga lelang dikurangi pokok pinjaman dan sewa modal) dikembalikan kepada nasabah. Apabila hasil penjualan lelang tidak mencukupi melunasi pokok pinjaman dan sewa modalnya, maka kekurangannya tetap menjadi kewajiban nasabah.

Pelaksanaan lelang barang jaminan wajib dilakukan di kantor cabang atau ditempat lain yang ditunjuk/ditentukan oleh pemimpin cabang dengan seijin pemimpin wilayah. Barang jaminan dari unit pembantu cabang (UPC) dibawa ke cabang atau tempat lain yang ditunjuk oleh panitia lelang. Dengan pertimbangan jarak dari cabang dan pertimbangan lain. UPC dapat melaksanakan lelang sendiri dengan seijin pemimpin wilayah dengan ketua panitia lelangnya adalah pemimpin cabang (manajer operasional cabang dan pengelola UPC dilarang menjadi ketua

lelang). Pelaksanaan lelang dilakukan dalam dua periode dalam satu bulan dengan ketentuan sebagai berikut:

a. Periode I untuk pinjaman tanggal 1 s/d 15, lelang lelang dilaksanakan antara tanggal 18 s/d 22.
b. Periode II untuk pinjaman tanggal 16 s/d 31, lelang dilaksanakan antara tanggal 03 s/d 07.

Lelang barang jaminan dilakukan oleh panitia lelang, panitia lelang terdiri dari satu orang ketua (pemimpin cabang/manajer operasional usaha gadai) lelang dan dua orang anggota/lebih. Dengan tugas sebagai berikut:

a. Ketua Lelang sebagai koordinator atau pemandu lelang bertugas, menghitung dan menetapkan kembali HLL (Harga Limit Lelang), menawarkan barang yang akan dilelang, dan memutuskan pemenang lelang.
b. Anggota Lelang sebagai petugas yang membantu kelancaran pelaksanaan lelang bertugas, mencatat nama-nama calon pembeli lelang, menerima uang muka dari calon pembeli lelang, menerima pembayaran dari pembeli lelang, mengisi DPRL (Daftar Rincian Penjualan Lelang) dan membantu pekerjaan lainnya demi kelancaran pelaksanaan lelang.

   Terdapat beberapa istilah yang berhubungan dengan lelang barang jaminan, diantaranya:

1. Harga Limit Lelang (HLL)

   Harga limit lelang (HLL) adalah harga penawaran terendah yang pertama kali ditawarkan pada setiap penjualan lelang barang jaminan. HLL dapat ditetapkan berdasarkan dua pilihan berikut ini:

   a. Nilai Pasar Barang Lelang (NPBL), yaitu harga limit lelang suatu barang yang ditetapkan berdasarkan hasil barang tersebut dan Harga Pasar Pusat untuk Lelang (HPPL), Harga Pasar Daerah untuk Lelang (HPDL) dan/atau Harga Pasar Setempat (HPS) pada waktu dilaksanakan lelang.
   b. Nilai Minimum Barang Lelang (NMBL), yaitu Harga limit lelang suatu barang yang ditetapkan berdasarkan nilai Uang Pinjaman (UP), Sewa Modal (SM), biaya-biaya lain dan bea-bea lelang. UP yang dimaksud adalah UP normal, artinya sesuai dengan ketentuan dari taksiran normal. Apabila NPBL lebih besar dari NMBL maka HLL sesuai dengan NPBL. Jika NPBL lebih kecil dari NMBL maka HLL sesuai dengan NMBL.

2. Nilai Penjualan Lelang (NJL)

   NJL adalah harga lelang yang terbentuk pada saat lelang.

3. Nilai Pendapatan Lelang (NDL)

   NDL adalah hasil dari penjumlahan nilai penjualan lelang (NJL) dengan bea lelang. Dengan demikian NDL adalah harga yang harus dibayar oleh pembeli lelang/pemenang lelang.

4. Uang Kelebihan (Ukel)

   Uang Kelebihan (Ukel) adalah uang yang dapat dikembalikan kepada nasabah atas hasil penjualan lelang barang jaminannya sebesar selisih antara Nilai Penjualan Lelang (NJL) dengan jumlah Uang Pinjaman (UP), Sewa Modal (SM) dan biaya-biaya lain.

II.4. *Database*

II.4.1. Pengertian *Database*

Database adalah suatu aplikasi terpisah yang menyimpan suatu koleksi data. Masing-masing *database* memiliki satu API atau lebih yang berbeda untuk menciptakan, mengakses, mengelola, mencari dan mereplikasi data (Janner Simarmata; 2007:1).

Database juga dapat didefinisikan dalam berbagai sudut pandang seperti berikut:

1. Himpunan kelompok data yang saling berhubungan yang diorganisasi sedemikian rupa sehingga kelak dapat dimanfaatkan dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa tanpa pengulangan (*redudancy*) yang tidak perlu, untuk memenuhi kebutuhan.
3. Kumpulan *file*/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpan elektronik (Kusrini; 2007:2).

Selain basis data (*database*), ada yang dikenal dengan sistem basis data (*database system*) yang merupakan perpaduan antara basis data dan sistem manajemen basis data (*database management system* atau DBMS). Komponen-komponen sistem basis data meliputi:

1. Perangkat Keras (*Hardware*) sebagai pendukung operasi pengolahan data.

   Perangkat keras komputer adalah semua bagian fisik komputer. Contoh dari perangkat keras komputer yaitu: *mouse*, *keyboard*, *monitor*, *CPU*, memori dan lain-lain.
2. Sistem Operasi (*Operating System*) atau perangkat lunak untuk mengelola basis data. Sistem operasi merupakan suatu *software* sistem yang bertugas untuk melakukan kontrol dan manajemen *hardware* serta operasi-operasi dasar sistem, termasuk menjalankan *software* aplikasi seperti program-program pengolah kata dan *web browser*. Contoh dari sistem operasi yang ada sekarang yaitu, *DOS, Windows 98*, *Windows XP*, *Windows 2000, Windows NT*, *Linux, Machintos* dan lain-lain.
3. Basis data (*Database*) sebagai inti dari sistem basis data.
4. *Database Management System* (DBMS).

   DBMS adalah *software* yang menangani semua akses ke basis data. Secara konsep apa yang terjadi adalah sebagai berikut:

   a. *User* melakukan pengaksesan basis data untuk informasi yang diperlukannya menggunakan suatu bahasa manipulasi data, biasanya disebut SQL.

   b. DBMS menerima *request* dari *user* dan menganalisa *request* tersebut.

c. DBMS memeriksa skema eksternal *user*, pemetaan eksternal/ konseptual, pemetaan konseptual/ internal dan struktur penyimpanan.

d. DBMS mengeksekusi operasi-operasi yang diperlukan untuk memenuhi permintaan *user*.

Contoh dari DBMS ini yaitu antara lain *Microsoft SQL Server*, *Oracle*, *MySQL, Interbase, Paradox, Microsoft Acces* dan lain-lain.

5. Pemakai (*User*).

Pemakai merupakan orang atau sistem yang akan mengakses dan merubah isi basis data (Kusrini; 2007:11).

II.4.2. Kamus Data

Kamus data (KD) atau *data dictionary* (DD) atau disebut juga dengan istilah *systems data dictionary* adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi (Jogiyanto; 2005:725).

Dengan menggunakan kamus data, analisis sistem dapat mendefinisikan data yang mengalir di sistem dengan lengkap. Kamus data dibuat pada tahap analisis sistem dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. Pada tahap analisis, kamus data dapat digunakan sebagai alat komunikasi antara analis sistem dengan pemakai sistem tentang data yang mengalir di sistem. Pada tahap perancangan sistem , kamus data digunakan untuk merancang input, merancangan laporan-laporan dan *database*.

Sejumlah simbol yang digunakan dalam penggambaran kamus data seperti terlihat pada tabel II.1. berikut :

Tabel II.1. Simbol-simbol Kamus Data

Simbol	Uraian
=	Terdiri atas, mendefinisikan, diuraikan menjadi, artinya. Contoh: nama=sebutan+nama²+nama2+gelar²+gelar2
+	Dan
( )	Opsional (pilihan boleh ada atau boleh tidak) Contoh: alamat=alamat rumah+(alamat surat)
{}	Pengulangan Contoh : nama1= {karakter_valid}
[ ]	Memilih slah satu dari sejumlah alternatif, seleksi. Contoh : sebutan = [Bapak|Ibu|Yang Mulia]
* *	Komentar Contoh : *seminar yang akan diikuti*
|	Pemisah sejumlah alternatif pilihan antara simbol [ ]

Sumber : Budi Sutedja Dharma Oetama; (2006:118)

II.4.3. Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel relasional (Janner Simarmata dan Imam Paryudi; 2006: 77). Dengan normalisasi kita ingin mendesain *database* relasional yang terdiri dari tabel-tabel berikut:

1. Berisi data yang diperlukan.
2. Memiliki sesedikit mungkin redudansi.

3. Mengakomodasi banyak nilai untuk tipe data yang diperlukan.
4. Mengefisiensikan *update*.
5. Menghindari kemungkinan kehilangan data secara tidak disengaja/tidak diketahui.

Alasan utama dari normalisasi *database* minimal sampai dengan bentuk normal ketiga adalah menghilangkan kemungkinan adanya *insertion anomalies*, *deletion anomalies* dan *update anomalies*. Tipe-tipe kesalahan tersebut sangat mungkin terjadi pada *database* yang tidak normal. *Insertion anomaly* adalah sebuah kesalahan dalam penempatan informasi *entry* data baru ke seluruh tempat dalam *database* di mana informasi tersebut disimpan. Dalam *database* yang telah dinormalisasi, proses pemasukan suatu informasi baru hanya perlu dimasukkan ke dalam satu tepat. *Deletion anomaly* adalah sebuah kesalahan dalam penghapusan suatu informasi dalam *database* harus dilakukan dengan penghapusan informasi tersebut dari beberapa tempat dalam *database*. Dalam *database* yang telah dinormalisasi, penghapusan suatu informasi hanya perlu dilakukan dalam satu tempat dalam *database* tersebut. Sedangkan dalam melakukan *update* satu informasi, kesalahan juga dapat terjadi ketika kita harus melakukan *update* ke seluruh tempat yang mengimpan informasi tersebut. Kesalahan ini disebut dengan *update anomaly*.

Normalisasi merupakan cara pendekatan dalam membangun desain logika basis data relasional yang tidak secara langsung berkaitan dengan model data, tetapi dengan menerapkan sejumlah aturan dan kriteria standar untuk menghasilkan struktur tabel yang normal. Pada dasarnya desain logika basis data

relasional dapat menggunakan prinsip normalisasi maupun transformasi dari model E-R ke bentuk fisik. Dalam perspekrif normalisasi sebuah *database* dikatakan baik jika setiap tabel yang membentuk basis data sudah berada dalam keadaan normal. Suatu tabel dikatakan normal, jika:

a. Jika ada dekomposisi/penguraian tabel, maka dekomposisinya dijamin aman (*lossless-join decomposition*)
b. Terpeliharanya ketergantungan *functional* pada saat perubahan data (*dependency preservation*).
c. Tidak melanggar *Boyce Code Normal Form* (BCNF), jika tidak bisa minimal tidak melanggar bentuk normalisasi ketiga.

Beberapa kondisi yang diujikan pada proses normalisasi:

a. Menambah data/*insert*
b. Meng*edit*/meng*update*
c. Menghapus/*delete*
d. Membaca/*retrieve*

Bentuk-bentuk dari normalisasi:

1. Bentuk tidak normal

   Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti format tertentu, dapat saja tidak lengkap dan terduplikasi. Data dikumpulkan apa adanya sesuai keadaannya.

2. Bentuk normal tahap pertama (*1st Normal Form*)

   Sebuah tabel disebut 1NF jika:

   a. Tidak ada baris yang duplikat dalam tabel tersebut

b. Masing-masing *cell* bernilai tunggal

Permintaan yang menyatakan tidak ada baris yang duplikat dalam sebuah tabel berarti tabel tesebut memiliki sebuah kunci, meskipun kunci tersebut dibuat dari kombinasi lebih dari satu kolom atau bahkan kunci tersebut merupakan kombinasi dari semua kolom. Berikut ini akan dicontohkan normalisasi dari tabel Kuliah yang memiliki atribut : kode_kul, nama_kul, sks, semester, waktu, tempat dan nama_dos. Tabel Kuliah tersebut tidak memenuhi normalisasi pertama, karena terdapat atribut waktu yang tergolong ke dalam atribut bernilai banyak. Agar tabel tersebut dapat memenuhi 1NF, maka solusinya adalah dengan mendekomposisi tabel Kuliah menjadi:

a. Tabel Kuliah (kode_kul, nama_kul, sks, semester, nama_dos)
b. Tabel Jadwal (kode_kul, waktu, ruang)

3. Bentuk normal tahap kedua (*2nd Normal Form*)

Bentuk Normal Kedua (2NF) terpenuhi jika pada sebuah tabel semua atribut yang tidak termasuk dalam *primary key* memiliki ketergantungan fungsional pada *primary key* secara utuh. Sebuah tabel dikatakan tidak memenuhi 2NF, jika ketergantungannya hanya bersifat parsial (hanya tergantung sebagian dari *primary key*). Bentuk normal kedua akan dicontohkan sebagai berikut. Misal tabel Nilai terdiri dari atribut kode_kul, nim dan nilai. Jika pada tabel Nilai, misalnya kita tambahkan sebuah atribut yang bersifat redudan, yaitu nama_mhs, maka tabel Nilai ini dianggap melanggar 2NF. *Primary key* pada tabel Nilai adalah

[kode_kul.nim]. Penambahan atribut baru (nama_mhs) akan menyebabkan adanya ketergantungan fungsional yang baru yaitu nim-> nama_mhs. Karena atribut nama_mhs ini hanya memiliki ketergantungan pasrial pada *primary key* secara utuh (hanya tergantung pada nim, padahal nim hanya bagian dari *primary key*). Bentuk normal kedua ini dianggap belum memadai karena meninjau sifat ketergantungan atribut terhadap *primary key* saja.

4. Bentuk normal tahap Ketiga (*3rd Normal Form*)

   Sebuah tabel dikatakan memenuhi bentuk normal ketiga (3NF), jika untuk setiap ketergantungan fungsional dengan notasi X -> A, dimana A mewakili semua atribut tunggal di dalam tabel yang tidak ada di dalam X, maka:

   a. X haruslah *superkey* pada tabel tersebut
   b. Atau A merupakan bagian dari *primary key* pada tabel tersebut

   Misalnya pada tabel Mahasiswa, atribut alamat_mhs dipecah ke dalam alamat_jalan, alamat_kota dan kode_pos. Bentuk ini tidak memenuhi 3NF, karena terdapat ketergantungan fungsional baru yang muncul pada tabel tersebut, yakni:

   Alamat_jalan nama_kota-> kode_pos

   Dalam hal ini (alamat_jalan, nama_kota) bukan *superkey* sementara kode_pos juga bukan bagian dari *primary key* pada tabel Mahasiswa. Jika tabel Mahasiswa didekomposisi menjadi tabel Mahasiswa dan tabel Alamat, maka telah memenuhi 3NF. Hal itu dapat dibuktikan dengan

memeriksa dua ketergantungan fungsional pada tabel alamat tersebut, yaitu:

Alamat_jalan nama_kota -> kode_pos

Kode_pos-> nama_kota

Ketergantungan fungsional yang pertama tidak melanggar 3NF, karena (alamat_jalan nama_kota) merupakan *superkey* (sekaligus sebagai *primary key*) dari tabel Alamat tersebut. Demikian juga dengan ketergantungan fungsional yang kedua meskipun (kode_pos) bukan merupakan *superkey*, tetapi nama_kota merupakan bagian dari *primary key* dari tabel Alamat. Karena telah memenuhi 3NF, maka tabel tersebut tidak perlu di-dekomposisi lagi.

5. Bentuk Normal Tahap Keempat dan Kelima

   Penerapan aturan normalisasi sampai bentuk normal ketiga sudah memadai untuk menghasilkan tabel berkualitas baik. Namun demikian, terdapat pula bentuk normal keempat (4NF) dan kelima (5NF). Bentuk Normal Keempat berkaitan dengan sifat ketergantungan banyak nilai (*multivalued dependency*) pada suatu tabel yang merupakan pengembangan dari ketergantungan fungsional. Adapun bentuk normal tahap kelima merupakan nama lain dari *Project Join Normal Form* (PJNF).

6. *Boyce Code Normal Form* (BCNF)

   a. Memenuhi 1st NF

b. Relasi harus bergantung fungsi pada atribut *superkey* (Kusrini; 2007:39-43).

II.5. *UML* (*Unified Modeling Language*)

II.5.1. Pengertian *UML*

UML singkatan dari *Unifed Modeling Language* yang berarti bahasa pemodelan standar. Sebagai suatu bahasa, berarti *UML* memiliki sintaks dan semantik. Ketika kita membuat model menggunakan konsep *UML* ada aturan – aturan yang harus diikuti. Bagaimana elemen pada model – model yang kita buat berhubungan satu dengan lainnya harus mengikuti standar yang ada. *UML* bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. Ketika pelanggan memesan sesuatu dari sistem, bagaimana transaksinya? Bagaimana sistem mengatasi *error* yang terjadi? Bagaimana keamanan terhadap sistem yang kita buat? Dan sebagainya dapat dijawab dengan *UML*.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk:

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses – proses dan organisasinya.

UML telah diaplikasikan dalam bidang investasi perbankan, lembaga kesehatan, departemen pertahanan, sistem terdistribusi, sistem pendukung alat kerja, retail, *sales* dan *supplier*. Blok pembangun utama *UML* adalah diagram.

Beberapa diagram ada yang rinci (jenis *timing diagram*) dan lainnya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorientasi objek menggunakan bahasa model untuk menggambarkan, membangun, dan mendokumentasikan sistem yang mereka rancang. *UML* memungkinkan para anggota *team* untuk bekerja sama dengan bahasa model yang sama dalam mengaplikasikan beragam sistem. Intinya, *UML* merupakan alat komunikasi yang konsisten dalam mensuport para pengembang sistem saat ini. Sebagai perancang sistem, mau tidak mau pasti akan menjumpai *UML*, baik kita sendiri yang membuta atau sekedar membaca diagram *UML* buatan orang lain (Prabowo Pudjo Widodo dan Herlawati; 2011:7).

II.5.2. Diagram-diagram *UML*

Beberapa literatur menyebutkan bahwa *UML* menyediakan sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa diagram yang digabung, misalnya diagram komunikasi, diagran urutan dan diagram pewaktuan di gabung menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokkan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain:

1. Diagram Kelas (*Class Diagram*)

   Bersifat Statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan sistem berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif.

2. Diagram *Use Case* (*Use Case Diagram*)

   Bersifat statis. Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (sesuatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.

3. Diagram Interaksi dan *Sequence* (Urutan)

   Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.

4. Diagram Aktivitas (*Activity Diagram*)

   Bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu sistem dan memberi tekanan pada aliran kendali antar objek.

5. Diagram Paket (*Package Diagram*)

   Bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas, merupakan bagian dari diagram komponen.

6. Diagram Komunikasi (*Communication Diagram*)

   Bersifat dinamis. Diagram ini sebagai pengganti diagram kolaborasi *UML* 1.4 yang menekankan organisasi struktural dari objek-objek yang menerima serta mengirim pesan.

7. Diagram *Statechart* (*Statechart Diagram*)

   Bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian serta aktivitas. Diagram ini

terutama penting untuk memperlihatkan sifat dinamis dari antarmuka (*interface*), kelas, kolaborasi dan terutama penting pada pemodelan sistem-sistem yang reaktif.

8. Diagram Komponen (*Component Diagram*)

   Bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen – komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas – kelas, antarmuka – antarmuka serta kolaborasi – kolaborasi.

9. Diagram *Deployment* (*Deployment Diagram*)

   Bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*). Memuat simpul-simpul beserta komponen-komponen yang ada di dalamnya. Diagram *deployment* berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*).

Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan (Prabowo Pudji Widodo dan Herlawati; 2011:10-12). Pada tugas akhir ini, penulis hanya menggunakan empat dari sembilan diagram yang ada. Pembahasan dari diagram-diagram tersebut akan dijelaskan pada penjelasan berikut.

II.5.3. Diagram *Use Case* (*Use Case Diagram*)

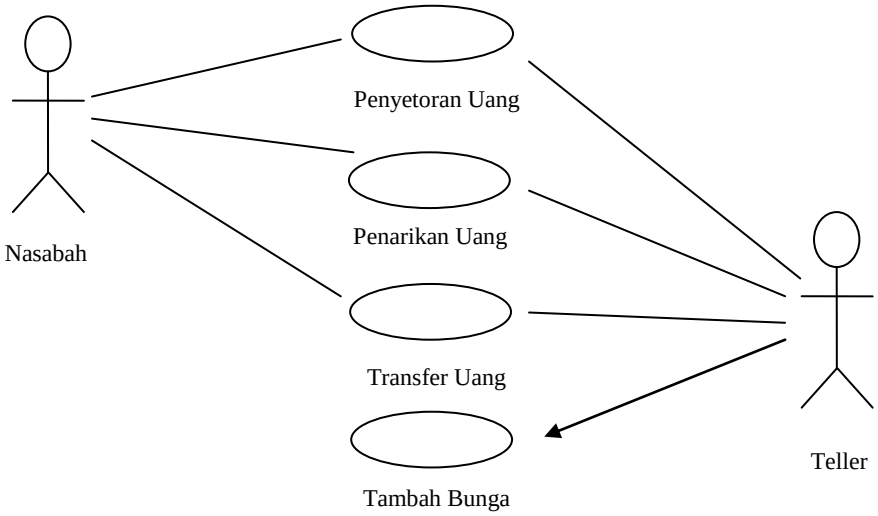
1. Diagram *Use Case*

Use case diagram menggambarkan *external view* dari sistem yang akan dibuat modelnya. Model *use case* dapat dijabarkan dalam diagram, tetapi yang perlu diingat, diagram tidak identik dengan model karena model lebih luas dari diagram (Prabowo Pudji Widodo dan Herlawati; 2011:16).

Komponen pembentukan *use case diagram* adalah:

1. Aktor (*actor*), menggambarkan pihak-pihak yang berperan dalam sistem.
2. *Use case*, aktivitas/sarana yang disiapkan oleh bisnis/sistem.
3. Hubungan (*link*), aktor mana saja yang terlibat dalam *use case* ini.

Gambar II.2. di bawah ini merupakan salah satu contoh bentuk *use case diagram*:

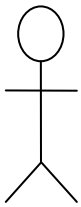


Gambar II.2. *Use Case Diagram*

Sumber: Prabowo Pudjo Widodo dan Herlawati (2011:17)

2. Aktor

Dalam membuat *use case* dan menentukan aktornya, agar mengidentifikasi siapa saja pihak yang terlibat dalam sistem. Pihak yang terlibat biasanya dinamakan *stakeholder*. Seperti yang terlihat pada gambar II.5 di atas terdapat *use case diagram* dengan dua aktor (nasabah dan teller) dan empat *use case* (penyetoran uang, penarikan uang, *transfer* uang dan tambah bunga) (Prabowo Pudji Widodo dan Herlawati; 2011:17). Simbol aktor adalah gambar orang, seperti yang terlihat pada gambar II.3. berikut ini:

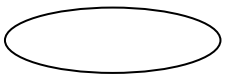


Gambar II.3. Simbol Aktor

Sumber: Prabowo Pudjo Widodo dan Herlawati; (2011:17)

3. *Use Case*

Use case menggambarkan fungsi tertentu dalam suatu sistem berupa komponen, kejadian atau kelas. Sedangkan *use case* juga dapat diartikan sebagai urutan langkah-langkah yang secara tindakan saling terkait (skenario), baik terotomatisasi maupun secara manual. *Use case* digambarkan dalam bentuk *ellips*/oval (Prabowo Pudji Widodo dan Herlawati; 2011:21). Simbol *use case* terlihat pada gambar II.4. berikut ini:



Gambar II.4. Simbol *Use case*

Sumber: Prabowo Pudjo Widodo dan Herlawati; (2011:22)

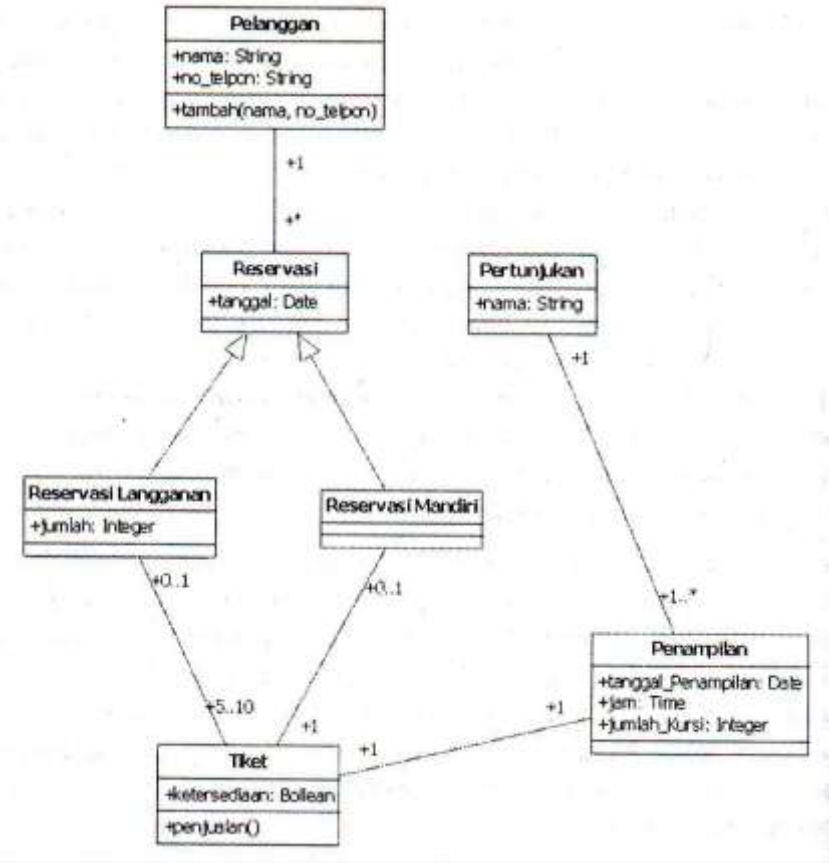
4. Relasi Antara *Use case / Aktor*

Pada *use case diagram*, relasi digambarkan sebagai sebuah garis antara dua simbol. Pemaknaan relasi berbeda-beda tergantung bagaimana garis tersebut digambar dan tipe simbol apa yang digunakan untuk menghubungkan garis tersebut. Relasi yang digunakan *UML* 2.0 adalah generalisasi, inklusi dan ekstensi, penjelasannya sebagai berikut.

a. Generalisasi (*Generalization*), generalisasi pada aktor dan *use case* dimaksudkan untuk menyederhanakan model dengan cara menarik keluar sifat-sifat pada aktor-aktor maupun *use case-use case* yang sejenis.
b. Ekstensi (*Extension*), ekstensi pada *use case* adalah *use case* yang terdiri dari langkah yang diekstraksi dari *use case* yang lebih kompleks untuk menyederhanakan masalah orisinal dan karena itu memperluas fungsinya. Hubungan antara ekstensi *use case* dan *use case* yang diperluas disebut *extend relationship*, diberi simbol "<<*extend*>>" dan hubungannya berupa garis putus-putus berpanah terbuka.
c. Inklusi (*Inclusion*), *use case* dasar yang akan diinklusi tidak lengkap, berbeda dengan *use case* dasar yang akan diekstensi. Sehingga *use case* inklusi bukan merupakan *use case* optional dan boleh tidak dijalankan. Simbol hubungan inklusi adalah garis putus-putus dengan anak panah terbuka dan diberi keterangan "<<*include*>>" (Prabowo Pudji Widodo dan Herlawati; 2011:24).

II.5.4. Diagram Kelas (*Class Diagram*)

Diagram kelas adalah inti dari proses pemodelan objek. Baik *forward engineering* maupun *reverse enginnering* memanfaatkan diagram ini *forward enginerring* adalah proses perubahan model menjadi kode program sedangkan *reverse enginnering* sebaliknya merubah kode program menjadi model (Prabowo Pudji Widodo dan Herlawati; 2011:37). Contoh *class diagram* terlihat pada gambar II.5. berikut ini:

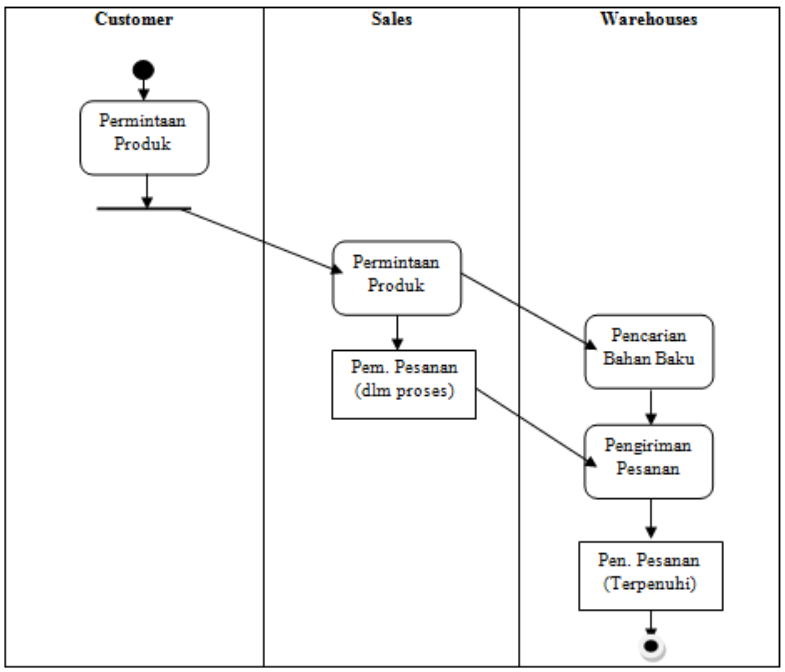


Gambar II.5. *Class Diagram*

Sumber: Adi Nugroho; (2010:12)

II.5.5. Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas lebih memfokuskan dari pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. Diagram ini tidak hanya memodelkan *software* melainkan memodelkan model bisnis juga. Diagram aktivitas menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi. Ketika digunakan dalam pemodelan *software*, diagram aktivitas merepresentasikan pemanggilan suatu fungsi tertentu misalnya *call*. Sedangkan bila digunakan dalam pemodelan bisnis, diagram ini menggambarkan aktivitas yang dipicu oleh kejadian-kejadian di luar seperti pemesanan atau kejadian-kejadian internal misalnya proses penggajian tiap jumat sore (Prabowo Pudji Widodo dan Herlawati; 2011:143-144). Contoh *activity diagram* terlihat pada gambar II.6. berikut ini:



Gambar II.6. *Activity Diagram*

Sumber: Adi Nugroho; (2010:61)

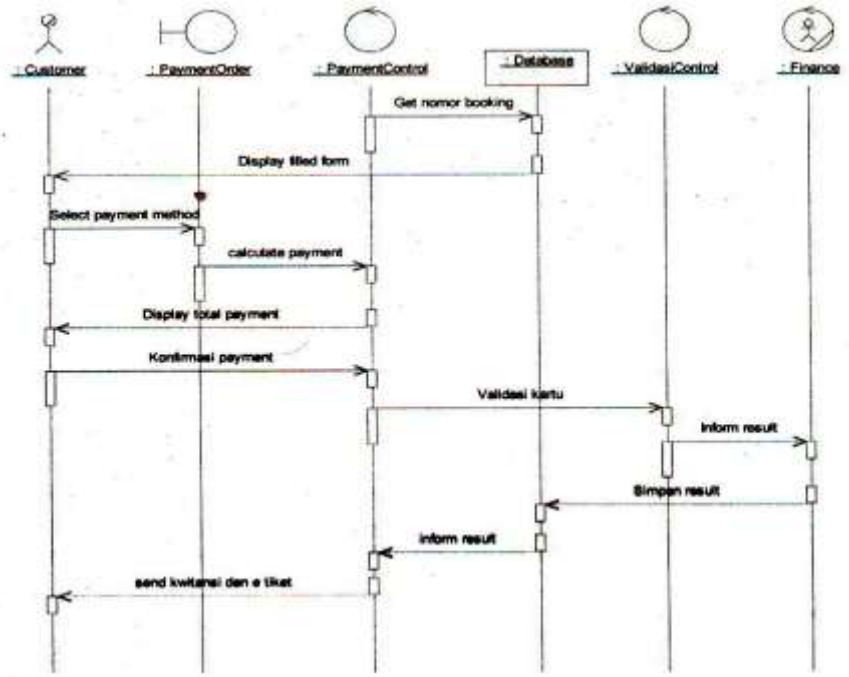
Aktivitas merupakan kumpulan aksi-aksi. Aksi-aksi melakukan langkah sekali saja tidak boleh dipecah menjadi beberapa langkah lagi. Contoh aksi yaitu:

a. Fungsi matematika
b. Pemanggilan perilaku
c. Pemrosesan data

Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu *classifier, classifier* dikatakan konteks daari aktivitas. Aktivitas dapat mengakses atribut dan operasi *classifier,* tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan untuk model proses bisnis, informasi itu biasanya disebut *process-relevant data*. Aktivitas diharapkan dapat digunakan ulang dalam suatu aplikasi, sedangkan aksi biasanya *specific* dan digunakan hanya untuk aktivitas tertentu.

II.5.6. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan/perilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima antar objek. Oleh karena itu, untuk menggambar *sequence diagram* maka harus diketahui objek-objek yang terlibat dalah sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu (Yuni Sugiarti; 2013:69). Contoh *sequence diagram* terlihat pada gambar II.7. berikut ini:



Gambar II.7. *Sequence Diagram*

Sumber: Yuni Sugiarti; (2013:72)

II.6. Visual Basic.Net

II.6.1. Pengertian Visual Basic.Net

Oleh karena adanya perubahan paradigma pemrograman dari pemrograman prosedural menjadi pemrograman berorientasi objek, *Microsoft* menjawabnya dengan memunculkan *.NET*. Teknologi *.NET* sendiri dapat dikatakan sebagai desain ulang dari Java dengan prinsip yang sama namun mempunyai tujuan yang berbeda (Wahana Komputer; 2012:2).

Sedangkan *Visual Basic* adalah bahasa pemrograman generasi ke tiga dari *Microsoft* dengan IDE (*Integrated Development Environment*) atau pemrograman pengembangan terpadu. *Visual Basic* dibuat dan dirancang untuk mudah

digunakan baik oleh programer pemula sekalipun. Selain digunakan untuk membuat program saderhana *Visual Basic* digunakan juga untuk membangun program kompleks, dengan menggunakan bahasa Basic, serta memungkinkan pengembangan dengan cepat dengan antarmuka grafis (GUI) (Eko Hari Atmoko; 2013:1).

Pada tugas akhir ini, penulis menggunakan program aplikasi *Visual Studio 2010. Visual Studio 2010* menyediakan berbagai *tool*, antara lain *tool Toolbox* yang berisi komponen visual, sehingga memudahkan dalam penulisan kode. Selain itu juga ada jendela *wizard* yang membantu untuk melakukan pemrograman dengan sangat mudah. Selain itu juga programer dapat menuliskan kode *Visual Studio 2010* pada lingkungan kerja lain, seperti *Visual Basic Express Edition* yang juga disediakan oleh *Microsoft* secara *free* bagi para pelajar dan pemula (Wahana Komputer; 2012:7).

II.7. Microsoft SQL Server

II.7.1. Pengertian SQL Server

SQL Server merupakan sebuah *Relational Database Manajement System* (RDBMS) buatan *Microsoft*, yang dirancang untuk mendukung program dengan arsitektur *client*/*server*, dimana *database* diletakkan pada komputer pusat yang disebut *server*, dan informasi digunakan bersama-sama oleh pengguna yang menjalankan program di dalam komputer yang sebut *client* (Eko Hari Atmoko; 2013:2).

II.7.2. Tipe Data SQL Server

Setiap kolom dalam *database* memiliki tipe data tertentu, dengan kata lain sebelum membuat *database* harus menentukan jenis tipe data yang cocok untuk tiap komponen yang dibangun , tipe data ini mengidentifikasikan ukuran data di dalam *table* atau *variable*, misal *text*, angka,data atau gambar.

SQL Server juga memiliki tipe data NULL, pilihan ini digunakan jika tidak ingin mendefinisikan suatu bilai pada data (Eko Hari Atmoko; 2013:91). Pada tabel II.2. berikut dapat dilihat tipe data yang ada pada SQL Server :

Tabel II.2. Tipe Data SQL Server

Tipe Data	Keterangan	Ukuran
Bigint	Integer (bilangan bulat)	8 bytes
Binary	Tipe data ini dapat menerima data binary dengan maksimum 8000 bytes data. Tipe data ini diinterpretasikan sebagai string dari bit misalnya (110011001011)	4 bytes
Bit	Tipe data integer yang hanya dapat bernilai 1 dan 0 atau NULL. Dapat digunakan untuk program dengan output Yes/No atau True/False	1 byte untuk kolom 8 bit, 2 bytes untuk kolom 9-16 bit, dst
Char	Data non-Unicode dengan jumlah karakter sampai dengan 8000 karakter	5 bytes
Cursor	Referensi pada cursor. Hanya dapat digunakan sebagai variabel dan parameter pada stored procedure	-
Datetime	Data tanggal dan waktu dari 1 Januari 1753 s/d 31 Desember 9999, dengan akurasi 3.33 mili detik	8 bytes
Decimal	Tipe data ini menerima nilai yang lebih presisi dibandingka tipe data integer. Tipe data ini menggunakan 2 parameter untuk menentukan tingkat presisi nilai yang diterima; precision dan scale. Precision adalah jumlah digit yang bisa diterima oleh field, sedangkan scale adalah jumlah angka di belakang koma yang bisa diterima oleh field. Jadi, jika membuat parameter precision sebanyak 5 dan scale sebanyak 2 maka field bisa menerima nilai seperti ini: 123,45. Tipe	5 – 17 bytes

	data ini bisa menerima nilai mulai dari -1038 hingga 1038-²	
Numeric	Tipe data ini pada dasarnya sama dengan tipe data decimal.	5 – 17 bytes
Float	Tipe data ini mirip dengan tipe data decimal,hanya saja parameter scale pada tipe data ini bisa menerima nilai yang tak terhingga, seperti pada nilai pi. Tipe data ini bisa menerima nilai mulai dari -1.79E + 308 hingga 1.79E + 308	4 – 8 bytes
Image	Mendefinisikan binary data untuk menyimpan image seperti GIF, JPG, TIFF	16 bytes
Integer atau int	Tipe data ini dapat menerima nilai mulai 2³¹ (-2,147,483,648) hingga 2³¹-1 (2,147,483,647)	4 bytes
Money	Tipe data ini dapat menerima nilai mulai dari -26³ (-9,223,372,036,854,775,808) hingga 26³-1 (-9,223,372,036,854,775,807)	8 bytes
Nchar	Tipe data ini mirip dengan tipe data char, namun tipe data ini bisa menerima nilai atau data Unicode (berbeda dengan tipe data char yang hanya bisa menerima nilai karakter non-Unicode). Tipe data ini bisa menerima nilai hingga 4000 karakter	2 bytes x jumlah karakter
Ntext	Data Unicode dengan ukuran bervariasi dengan jumlah karakter maksimal 2^30 -²(²,073,741,823)	16 bytes untuk pointer dan 2 bytes x karakter
Nvarchar	Data karakter Unicode dengan ukuran bervariasi 1s/d 4000	2 bytes x karakter
Real	Tipe data ini mirip dengan tipe data float, hanya saja menerima nilai yang lebih kecil dibandingkan dengan float, yaitu mulai dari -3.40E +38 hingga 3.40E +38	4 bytes
Smalldatetime	Tipe data ini dapat menerima tanggal dan waktu mulai dari ² Januari 1900 hingga 6 Juni 2079, dengan akurasi 1 menit	4 bytes
Smallint	Tipe data ini juga mirip dengan int, hanya saja nilai yang diterima lebih kecil daari int. Tipe data ini dapat menerima nilai mulai dari -2¹5 (-32,768) hingga 2¹5 -1(32767)	2 bytes
Smallmoney	Tipe dasar ini pada dasarnya sama dengan tipe data money, hanya saja nilai yang terima lebih kecil, yaitu mulai dari -214,748.3648 hingga 214,748.3647	4 bytes
Sql_variant	Mengizinkan untuk semua tipe data, jadi jika menggunakannya secara otomatis SQL Server akan	Sesuai jenis data

	mencari tipe data yang cocok untuk data tersebut.	
Sysname	Tipe data khusus, system-supplied, atau SQL user-defined data type. Tipe data sysname didefinisikan SQL Server sebagai nvarchar (128), yang mampu menampung 128 karakter Unicode atau 256 bytes. Gunakan sysname untuk mangacu pada kolom nama obyek	256 bytes
Uniqe-identifier	Tipe data ini berfungsi untuk membuat nilai yang unik yang mungkin bisa tampil seperti ini 6F9619FF-8B86-D011-B42D-00C04FC964FF. Tipe data ini berguna jika ingin membuat serial number atau id yang unik	16 bytes
Varbinary	Tipe data ini mirip dengan varchar,hanya saja nilai yang bisa diterima hanya data binary. Tipe data ini berguna untuk menyimpan data binary yang tidak diketahui dengan pasti jumlah bytes datanya	Panjang data + 4 bytes
Varchar	Tipe data ini mirip dengan tipe data char, namun tipe data ini berguna bagi yang tidak mengetahui secara pasti jumlah karakter yang akan dimasukkan oleh user. Tipe data ini juga bisa menerima nilai hingga 8000 karakter.	Panjang data actual yang dimasukkan
Xml	Tipe data ini berguna untuk menyimpan data dalam format XML Document. Tipe data ini dapat menyimpan data hingga 2 Gb. Tipe data ini merupakan tipe data baru yang terdapat di SQL Server	-

Sumber : Eko Hari Atmoko; (2013:91)

II.8. Gambaran Umum Perusahaan

II.8.1. Sejarah Singkat Perusahaan

Lahirnya PT. Pegadaian (Persero) di Indonesia ditandai dengan berdirinya Bank *Van Lening* pada masa *VOC* pada tahun 1746. Lembaga ini mempunyai tugas memberikan pinjaman uang kepada masyarakat dengan jaminan gadai.

Sampai sekarang Pegadaian telah mengalami 5 zaman pemerintahan yaitu:

A. Pegadaian pada masa *VOC* (1764 – 1811)

B. Pegadaian pada masa penjajahan Inggris (1811 – 1816)

C. Pegadaian pada masa penjajahan Belanda (1816 – 1942)

D. Pegadaian pada masa penjajahan Jepang (1942 – 1945)

E. Pegadaian pada masa Kemerdekaan (1945 – sekarang)

1. Pegadaian pada masa VOC (1764 – 1811)

Pegadaian sewaktu Indonesia di bawah kekuasaan *Vereenidge Qest Indische Compagnie (VOC),* Bank *Van Leening* pun ikut dibawa ke Indonesia. Dengan surat keputusan Gubernur Jenderal Van Imhoff tertanggal 20 Agustus 1746 dengan resmi didirikan suatu bank *Van Leening* yang pertama di Indonesia yaitu di Jakarta (Batavia). Bank ini didirikan dalam bentuk kerjasama antara *VOC* dengan swasta lainnya yaitu dengan £ 500.000 (2/3 dari *VOC* dan 1/3 dari swasta) disamping menjalankan usaha pemberian kredit berdasarkan gadai juga memberikan jasa Bank Wesel.

2. Pegadaian pada masa Penjajahan Inggris (1811 – 1816)

Pada masa penjajahan Inggris (1811) Bank *Van Leening* ini dihapuskan. Hal ini menurut keputusan Reffles yang berpendapat bahwa tidak wajar bagi suatu bank diusahakan oleh pemerintah. Sebagai gantinya diadakan suatu ketentuan bahwa setiap orang boleh mendirikan pegadaian swasta asal mendapat izin (*Licentie*) dari penguasa daerah setempat. *Licentiesel* ini diperkirakan akan menguntungkan pemerintah, namun yang terjadi sebaliknya dan pemegang *Licentie* menggunakan kesempatan ini untuk mengadakan praktik riba yang sangat merugikan rakyat, karena dari usahanya tersebut ingin memperoleh keuntungan sebesar-besarnya. Dengan kata lain *Licentiesel* itu justru malah menghidupkan

usaha-usaha lintah darat. Kemudian pada tahun 1814 *Licentiesel* tersebut diganti dengan *Pachstelsel*, yaitu hak mendirikan pegadaian diberikan kepada umum dengan penawaran yang paling tinggi (*Openbar Verpacht*) bahwa setiap orang boleh menerima gadai asal sanggup membayar sejumlah uang tertentu kepada pemerintah.

3. Pegadaian pada masa Penjajahan Belanda (1816 – 1942)

Kemudian *Pachstelsel* tersebut di atas (1843) telah dijalankan di seluruh Indonesia kecuali di daerah Priangan dan *Verstenladen* (Surakarta dan Yogyakarta). Pada tahun 1949 *rente-terief* (tarif bunga) ditetapkan oleh pemerintah dengan *Pachstelsel* ditetapkan sebagai monopoli, yang berarti bahwa seorang pemegang *Pach* dilarang menerima gadai sampai dengan jumlah £ 100. Larangan ini tercantum dalam KUHP (*Weetbook vanstratfrecht*) pada pasal 509 yang berbunyi : "Barang siapa yang dengan tidak berhak meminjamkan uang atau barang yang jumlahnya atau harganya tidak lebih dari seratus rupiah dengan menerima gadai atau dengan bentuk jual beli dengan hak membeli kembali atau dengan bentuk persetujuan komisi, dipidana dengan kurungan selama-lamanya 3 (tiga) bulan atau denda sebanyak-banyaknya Rp. 15.000,00 (lima belas ribu rupiah)."

Setelah mendapatkan suatu kesimpulan bahwa hasil uang pinjaman dari pegadaian menunjukkan hal-hal yang menguntungkan maka disarankan bahwa untuk membasmi lintah darat tersebut, harus dilakukan pemerintah. Dengan keputusan pemerintah (*Staatblad* No. 131 tanggal 12 Maret 1901) maka mulai

tanggal 1 April 1901 dibukalah Pegadaian Negara yang pertama di Indonesia yaitu Sukabumi.

Demikianlah sejarah timbulnya Pegadaian di Indonesia setelah Sukabumi diikuti pada tahun 1902 dibuka Pegadaian Negara kedua di Cianjur, kemudian pada tahun 1903 dibuka Pegadaian Negara di Purworejo, Bogor, Tasikmalaya, Cikaka (Bandung) dan Cimahi. Dengan *Staatsblad* tahun 1903 No. 266 Jawatan Pegadaian dijadikan perusahaan Negara dalam arti pasal 2 IBW (*Indonesce Bedrijivenwet) Staatsblad*. Tahun 1927 No. 419 sehingga dengan demikian bahwa kekayaan Negara yang tertanam di dalam usaha Jawatan Pegadaian diadministrasikan terpisah dari bagian kekayaan lainnya.

4. Pegadaian pada masa Penjajahan Jepang (1942 – 1945)

Setelah Jepang menduduki Indonesia pada tanggal 8 Maret 1942, maka pada pertengahan tahun 1942 Kantor Pusat Jawatan Pegadaian dipindahkan dari Jl. Kramat No. 162 ke Jl. Kramat No. 132 Jakarta dengan alasan akan dijadikan tempat tawanan perang. Barang-barang yang digadaikan saat itu adalah barang-barang emas dan permata kepunyaan rakyat yang harus dijual kepada tentara atau *Nippon*. Sedangkan lelang barang-barang emas dan permata dihapuskan dan pada tahun 1943 barang logam lainnya juga tidak dilelang. Akibatnya rakyat semakin melarat dan tidak mempunyai barang-barang berharga lagi, sehingga Pegadaian pada masa itu praktis hampir tidak berfungsi lagi. Pada waktu pemerintah Jepang mengeluarkan uang, sehingga uang yang beredar adalah uang Jepang.

5. Pegadaian pada masa Kemerdekaan (1945 – sekarang)

Pegadaian pada masa kemerdekaan dapat dibagi sebagai berikut :

a) Jawatan Pegadaian pada zaman Republik Indonesia (Perjuangan) tanggal 17 Agutus 1945 sampai dengan 27 Desesmber 1949 (Penyerahan Kedaulatan).
b) Jawatan Pegadaian pada zaman RIS tanggal 27 Desember 1949 sampai 17 Agustus 1950.
c) Jawatan Pegadaian dalam Negara Kesatuan RI 17 Agustus sampai sekarang.

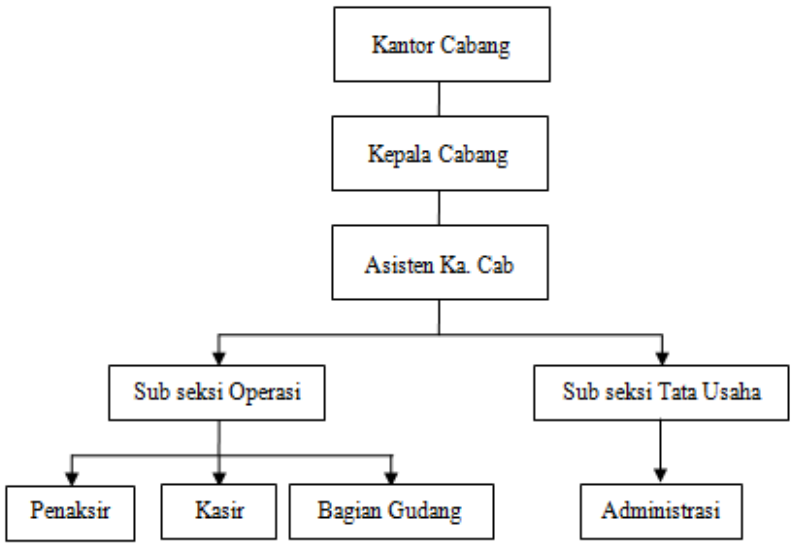
Berdasarkan peraturan pemerintah Republik Indonesia tahun 1961 No. 178 tanggal 3 Mei 1961 (Lembaran Negara Republik Indonesia tahun 1961 No. 209). Status sebagai Perusahaan Negara, Pegadaian ini hanya bertahan sampai tahun 1969. Pada tahun tersebut keluar Undang-undang Republik Indonesia tahun 1969 (Lembaran Negara tahun 1969 No. 40 Lembaran-lembaran Negara No. 2890), menjadi Undang-undang Lembaran tahun 1969 tambahan Lembaran Negara No. 2904. Undang-undang ini mengatur bentuk-bentuk usaha Negara menjadi 3 yaitu : PERJAN, PERUM dan PERSERO.

Sejalan dengan ketentuan dalam Undang-undang tersebut maka ditetapkan status Pegadaian melalui Peraturan Pemerintah No. 7 tahun 1969 menjadi Perusahaan Jawatan (PERJAN) Pegadaian. Dengan penyesuaian bentuk usaha tersebut, maka kekayaan Perusahaan Negara Pegadaian beralih kepada PERJAN Pegadaian. Pada tahun 1990 dikeluarkanlah Peraturan Pemerintah No. 10 1990 tanggal 10 April 1990 yang mengatur perubahan bentuk PERJAN Pegadaian menjadi Perusahaan Umum (PERUM) Pegadaian (Lembaga Negara tahun 1990 No. 14).

Untuk mengatur kembali peraturan tentang Perusahaan Umum Pegadaian dengan Peraturan Pemerintah, maka diterbitkanlah Peraturan Pemerintah No. 103 tahun 2000 tanggal 10 November 2000 tentang Perusahaan Umum Pegadaian. Namun pada tanggal 01 April 2012 Perusahaan Umum (PERUM) Pegadaian berubah menjadi Perseroan Terbatas (PT) Pegadaian sesuai dengan Peraturan Pemerintah.

II.8.2. Struktur Organisasi

Penelitian ini dilakukan untuk menganalisis dan merancang Sistem Informasi Lelang Barang Jaminan pada PT. Pegadaian (Persero) Cabang Labuhan Deli Medan, struktur organisasinya dapat dilihat pada gambar II.8. berikut :



Gambar II.8. Struktur Organisasi PT.Pegadaian (Persero)

Cabang Labuhan Deli Medan

Sumber : PT. Pegadaian (Persero)

II.8.3. Tugas dan Wewenang

Adapun tugas dan wewenang dari struktur organisasi pada subbab di atas adalah sebagai berikut :

1. Kepala Cabang

   Mengelola operasional cabang dengan menyalurkan lama pinjaman gadai dan melaksanakan usaha-usaha lainnya, serta mewakili kepentingan perusahaan dengan pihak lain atau masyarakat.

2. Asisten Kepala Cabang

   Melakukan pengawasan terhadap uang taksiran barang jaminan, uang pinjaman gadai, pengelolaan gudang barang jaminan, dan usaha lain serta mewakili kepala cabang dalam mengelola cabang apabila kepala cabang berhalangan, agar pelaksanaan operasional berjalan lancar, efektif dan efisien.

3. Penaksir

   Menaksir barang jaminan untuk menentukan mutu dan nilai barang sesuai dengan ketentuan yang berlaku.

4. Kasir

   Melakukan penerimaan-penerimaan dan pembayaran sesuai dengan aturan yang berlaku untuk kelancaran operasional.

5. Bagian Gudang

   Melakukan pemeriksaan, penyimpanan dan pengeluaran barang jaminan sesuai dengan peraturan yang berlaku.

6. Bagian Administrasi

Membuat laporan kantor cabang, dimana laporan itu berisi data-data kegiatan operasional cabang di dalam waktu kegiatan mingguan, untuk diberikan kepada kantor wilayah bagian keuangan oleh kepala cabang.