

BAB II

LANDASAN TEORI

2.1. Konsep Dasar Sistem

Didalam sebuah perusahaan diperlukan adanya sistem untuk mengatur arus kegiatan dan informasi pada perusahaan yang bersangkutan. Penerapan sistem pada suatu perusahaan dilakukan untuk mendukung strategi bisnis perusahaan, proses bisnis, dan struktur perusahaan dalam rangka meningkatkan nilai bisnis dari perusahaan tersebut. Dukungan strategis dari sistem pada perusahaan tersebut dalam bentuk peningkatan efisiensi dan efektivitas dalam pelaksanaan berbagai aktifitas perusahaan.

Beberapa tahun yang lalu, sistem informasi dalam perusahaan mungkin masih dikembangkan secara sederhana. Tetapi memasuki era globalisasi dimana teknologi menjadi salah satu komponen penting dalam kehidupan manusia, tak dapat dipungkiri bahwa banyaknya perusahaan yang berkembang menjadi semakin besar dikarenakan sistem yang sudah terorganisir, baik sistem yang bersifat manual maupun yang sudah terkomputerisasi.

Dengan demikian, tak heran bahwa sistem itu sendiri sudah dipelajari dan dianalisa dengan menyeluruh. Sistem secara umum dapat didefinisikan sebagai suatu totalitas himpunan bagian-bagian yang satu sama lain berhubungan sedemikian rupa sehingga menjadi satu kesatuan yang terpadu untuk mencapai suatu tujuan tertentu.

2.1.1. Pengertian Sistem

Secara sederhana sistem adalah seperangkat unsur-unsur yang terdiri dari manusia, alat, konsep dan prosedur yang dihimpun menjadi satu kesatuan untuk maksud dan tujuan bersama. Namun ada beberapa pakar yang berlainan pendapat untuk mendefinisikan suatu sistem dengan pandangan yang berbeda-beda namun tetap satu tujuan.

Sedangkan menurut Sutarman (2009:5) mengemukakan bahwa “Sistem adalah kumpulan elemen yang saling berinteraksi dalam suatu kesatuan untuk menjalankan suatu proses pencapaian suatu tujuan utama”.

Menurut Mulyadi (2010:5) menyatakan bahwa:

Sistem adalah jaringan prosedur yang dibuat menurut pola yang terpadu untuk melaksanakan kegiatan-kegiatan pokok perusahaan sedangkan prosedur adalah suatu urutan kegiatan klerikal, biasanya melibatkan beberapa orang dalam satu departemen atau lebih yang dibuat untuk menjamin penanganan secara seragam transaksi perusahaan yang terjadi secara berulang-ulang.

Menurut Tata Sutabri (2012:6) mengemukakan bahwa “Sistem adalah sekelompok yang erat hubungannya satu sama lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu”.

Dan menurut Azhar Susanto (2013:22) mengemukakan bahwa “Sistem adalah kumpulan/grup dari subsistem/bagian/komponen apapun baik fisik ataupun non fisik yang saling berhubungan satu sama lain dan bekerja sama secara harmonis untuk mencapai satu tujuan tertentu”.

Berdasarkan pengertian sistem diatas, dapat disimpulkan bahwa sistem adalah sekelompok komponen dan elemen yang digabungkan menjadi satu untuk mencapai tujuan tertentu.

2.1.2. Karakteristik Sistem

Menurut Agus Mulyanto (2009:2) bahwa “Suatu sistem mempunyai karakteristik komponen maupun elemen (*component*), batasan sistem (*boundary*), lingkungan luar sistem (*environment*), penghubung (*interface*), masukan (*input*), pengolahan (*process*), keluaran (*output*), sasaran (*objective*), dan tujuan (*goal*)”.

1. Komponen Sistem

Suatu sistem tidak berada dalam lingkungan yang kosong, tetapi sebuah sistem berada dan berfungsi didalam lingkungan sistem lainnya. Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, bekerja sama membentuk satu kesatuan. Apabila suatu sistem merupakan sebuah komponen dari sistem lainnya yang lebih besar, maka akan disebut dengan subsistem, sedangkan yang lebih besar tersebut adalah lingkungannya. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempunyai suatu proses sistem secara keseluruhan.

2. Batasan Sistem

Batasan sistem merupakan pembatas atau pemisah antara suatu sistem dengan sistem lainnya atau dengan lingkungan luarnya. Batasan sistem menentukan konfigurasi ruang lingkup ataupun kemampuan sistem. Batasan sistem ini

memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batasan sistem juga menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem

Lingkungan luar sistem adalah apapun diluar batas dari sistem yang dapat mempengaruhi operasi sistem, baik yang menguntungkan ataupun yang merugikan. Pengaruh yang menguntungkan ini harus dijaga hingga akan mendukung kelangsungan operasi sebuah sistem. Sedangkan pengaruh dari lingkungan yang merugikan harus ditahan dan dikendalikan agar tidak mengganggu sebuah kelangsungan sebuah sistem

4. Penghubung Sistem

Penghubung merupakan hal yang sangat penting, sebab tanpa adanya penghubung, sistem akan berisi kumpulan subsistem yang berdiri sendiri dan tidak saling berkaitan. Penghubung merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. penghubung inilah yang akan menjadi media yang digunakan data dari masukan (*input*) hingga keluaran (*output*). Dengan adanya penghubung suatu subsistem dapat berinteraksi dan berintegrasi dengan subsistem lainnya membentuk satu kesatuan.

5. Masukan Sistem

Masukan atau *input* merupakan energi yang dimasukkan dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*), dan masukan sinyal (*signal input*). *Maintenance input* adalah bahan yang dimasukkan agar sistem dapat beroperasi. *Signal input* adalah masukan yang diproses mendapatkan keluaran. Sebagai contoh didalam sistem komputer, program merupakan

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

6. Keluaran Sistem

Keluaran (*output*) merupakan hasil dari pemrosesan. Keluaran dapat berupa informasi sebagai masukan dari perubahan untuk menjadikan keluaran yang diinginkan.

7. Pengolahan Sistem

Pengolahan sistem (*Process*) merupakan bagian yang melakukan perubahan dari masukan untuk menjadi keluaran yang diinginkan.

8. Sasaran Sistem

Suatu sistem pasti memiliki sasaran (*objective*) atau tujuan (*goal*). Jika suatu sistem tidak memiliki sasaran maka suatu sistem operasi tersebut tidak akan ada gunanya. Tujuan inilah yang mengarahkan suatu sistem. Secara umum suatu sistem memiliki tiga tujuan utama yaitu mendukung fungsi manajemen, mendukung pengambilan keputusan manajemen, dan mendukung kegiatan operasi perusahaan.

2.1.3. Klasifikasi Sistem

Sistem dapat diklasifikasikan ke melalui beberapa sudut pandang, diantaranya:

1. Sistem Abstrak dan Sistem Fisik

Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik, misalnya sistem teologi yaitu sistem pemikiran-pemikiran antara manusia dengan Tuhan. Sedangkan sistem fisik merupakan sistem yang ada secara fisik, misalnya sistem penjualan, sistem komputer, dan sebagainya.

2. Sistem Alamiah dan Sistem Buatan Manusia

Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia. Misalnya sistem perputaran bumi, sistem pergantian siang dan malam. Sedangkan sistem yang dirancang manusia adalah sistem dengan campur aduk atau buatan manusia.

3. Sistem Tertentu dan Sistem Tak Tentu

Sistem tertentu beroperasi dengan tingkah laku yang sudah diprediksi. Interaksi diantara bagian-bagiannya dapat diprediksi dengan pasti, sehingga keluaran dari sistem dapat diramalkan. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas.

4. Sistem Tertutup dan Sistem Terbuka

Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Secara teoritis sistem tertutup ini ada, namun pada kenyataannya sistem ini tidak ada. Yang ada hanya *relatively closed* (secara relatif tertutup, namun tidak benar-benar tertutup). Karena tidak ada sistem mana pun yang bersifat tertutup. Sedangkan Sistem terbuka merupakan sistem yang berhubungan dan terpengaruh dengan lingkungan

luarnya. Sistem ini menerima masukan dan menghasilkan keluaran untuk lingkungan luar atau subsistem lain.

2.1.4. Konsep Dasar Informasi

Sistem informasi berawal dari sebuah data yang merupakan kumpulan angka atau huruf yang belum di olah dan tidak memiliki arti. Jika sebuah data tersebut kemudian diolah maka data tersebut dapat dinamakan dengan informasi yang lebih berarti dan berguna bagi yang menerimanya. Dalam mengelola informasi tersebut dibutuhkan sebuah sistem, yaitu sistem informasi. Sistem Informasi itu sendiri adalah suatu sistem didalam organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat managerial dan kegiatan strategi dari luar organisasi dan menyediakan pihak luar tertentu dengan pihak luar tertentu. Dari pengertian informasi diatas dapat ditarik kesimpulan bahwa informasi merupakan hasil akhir dari data yang telah diproses pada suatu sistem sehingga menjadi bentuk yang lebih berguna bagi yang menerimanya untuk pengambilan keputusan.

2.1.5. Pengertian Informasi

Adapun informasi menurut Prasajo dan Riyanto dalam Kadir (2009:3) menyatakan bahwa, “Pengertian informasi sering disamakan dengan pengertian data. Data adalah sesuatu yang belum diolah dan belum dapat digunakan sebagai dasar yang kuat dalam pengambilan keputusan”.

Dan informasi menurut Sutarman dalam Mustakini (2009:14) menyatakan bahwa:

Informasi adalah sekumpulan fakta (data) yang diorganisasikan dengan cara tertentu sehingga mereka mempunyai arti bagi si penerima. Sebagai contoh, apabila kita memasukkan jumlah gaji dengan jumlah jam bekerja, kita akan mendapatkan informasi yang berguna. Dengan kata lain, informasi datang dari data yang akan diproses.

2.1.6. Tipe Informasi

Sistem Informasi dapat menyediakan tiga macam tipe informasi, masing-masing mempunyai arti yang berbeda untuk tingkatan manajemen yang berbeda, menurut Mustakini (2009:68) yaitu:

1. Informasi Pengumpulan Data (*Scorekeeping Information*), merupakan informasi yang berupa akumulasi atau pengumpulan data untuk menjawab pertanyaan.
2. Informasi Pengarahan Perhatian (*Attention Directing Information*), merupakan informasi untuk membantu manajemen memusatkan perhatian pada masalah-masalah yang menyimpang, ketidakberesan, ketidakefisienan dan kesempatan-kesempatan yang dapat dilakukan.
3. Informasi Pemecahan Masalah (*Problem Solving Information*), merupakan informasi untuk membantu manager atas mengambil keputusan memecahkan permasalahan yang dihadapinya.

2.1.7. Karakteristik Informasi

Untuk mendukung suatu keputusan dibutuhkan informasi yang berguna, dibutuhkan pula informasi dengan karakteristik yang berbeda berdasarkan tingkatan manajemen. Berikut karakteristik informasi menurut Mustakini (2009:71), yaitu:

1. Kepadatan Informasi

Untuk manajemen tingkat bawah, karakteristik informasinya adalah terperinci (*detail*) dan kurang padat, karena terutama digunakan untuk pengendalian operasi. Sedangkan untuk manajemen yang lebih tinggi tingkatannya, mempunyai karakteristik informasi yang semakin tersaring, lebih ringkas dan padat.

2. Luas Informasi

Untuk manajemen tingkat bawah, karakteristik informasinya adalah terfokus pada suatu masalah tertentu, karena digunakan oleh manager bawah yang mempunyai tugas yang khusus. Untuk manajemen yang lebih tinggi tingkatannya, membutuhkan informasi dengan karakteristik informasi yang semakin luas, karena manajemen atas berhubungan dengan masalah yang luas.

3. Frekuensi Informasi

Untuk manajemen tingkat bawah, frekuensi informasi yang diterima adalah rutin, karena digunakan oleh manager bawah yang mempunyai tugas yang terstruktur dengan pola yang berulang-rulang dari waktu ke waktu. Untuk manajemen yang lebih tinggi tingkatannya, frekuensinya adalah tidak rutin atau

(mendadak), karena manajemen atas berhubungan dengan pengambilan keputusan tidak terstruktur yang pola dan waktunya tidak jelas.

4. *Schedule* Informasi

Untuk manajemen tingkat bawah, informasi yang diterimanya mempunyai jadwal atau *schedule* yang jelas dan periodik, karena digunakan oleh manager bawah yang mempunyai tugas yang terstruktur. Untuk manajemen yang lebih tinggi tingkatannya, *schedule* informasinya adalah tidak ter-*schedule*, karena manajemen atas berhubungan dengan pengambilan keputusan tidak terstruktur.

5. Waktu Informasi

Untuk manajemen tingkat bawah, informasi yang dibutuhkan adalah informasi historis, karena digunakan oleh manager dalam pengendalian operasi yang memeriksa tugas-tugas rutin yang sudah terjadi. Untuk manajemen yang lebih tinggi tingkatannya, waktu informasinya lebih kemasa depan berupa informasi prediksi, karena digunakan oleh manajemen atas untuk pengambilan keputusan strategik yang menyangkut nilai masa depan.

6. Akses Informasi

Manajemen tingkat bawah membutuhkan informasi yang periodenya jelas dan berulang-ulang, sehingga dapat disediakan oleh bagian sistem informasi yang memberikan dalam bentuk laporan periodik. Dengan demikian, akses informasi untuk manajemen bawah dapat tidak secara *online*, tetapi dapat secara *offline*. Sebaliknya, untuk manajemen yang lebih tinggi tingkatannya, periode informasi yang dibutuhkannya tidak jelas, sehingga manager-manager tingkat

atas perlu disediakan akses online untuk mengambil informasi kapanpun mereka membutuhkannya.

7. Sumber Informasi

Karena manajemen tingkat bawah lebih berfokus pada pengendalian operasi internal perusahaan, maka manager-manager tingkat bawah lebih membutuhkan informasi dengan data yang bersumber dari internal perusahaan sendiri. Akan tetapi, manager-manager tingkat atas lebih berorientasi pada masalah perencanaan strategis yang berhubungan dengan lingkungan luar perusahaan, sehingga membutuhkan informasi dengan data yang bersumber pada eksternal perusahaan.

2.1.8. Pengertian Akuntansi

Suatu perusahaan sangat memerlukan akuntansi, karena dengan akuntansi kegiatan-kegiatan yang mengubah posisi keuangan perusahaan diproses menjadi suatu informasi yang berguna bagi manajemen perusahaan dan pengguna laporan keuangan lainnya.

Menurut Soemarso (2009:3) mengemukakan, “Akuntansi adalah proses mengidentifikasi, mengukur, dan melaporkan informasi ekonomi, untuk memungkinkan adanya penilaian dan keputusan yang jelas dan tegas bagi mereka yang menggunakan informasi tersebut”.

Menurut Munawir (2009:5) menyatakan bahwa:

Akuntansi adalah seni dari pencatatan, penggolongan, dan peringkasan dari pada peristiwa-peristiwa dan kejadian-kejadian yang setidaknya sebagian bersifat keuangan dengan cara yang setepat-tepatnya

dan dengan petunjuk atau dinyatakan dalam uang, serta penafsiran terhadap hal-hal yang timbul dari padanya.

Sedangkan menurut Ahmad dalam Munawir (2009:6) adalah:

Akuntansi adalah aktivitas-aktivitas yang menyediakan informasi biasanya bersifat kuantitatif dan seringkali disajikan dalam satuan moneter, untuk pengambilan keputusan, perencanaan, pengendalian sumber daya dan operasi, mengevaluasi prestasi dan pelaporan keuangan kepada para investor, kreditur, instansi yang berwenang serta masyarakat.

Dari pendapat-pendapat tersebut dapat disimpulkan bahwa akuntansi adalah suatu proses mencatat, mengklasifikasi, meringkas, mengolah dan menyajikan data, transaksi serta kejadian yang berhubungan dengan keuangan sehingga dapat digunakan oleh orang yang menggunakannya dengan mudah dimengerti untuk pengambilan suatu keputusan serta tujuan lainnya.

2.1.9. Pengertian Sistem Informasi Akuntansi

Menurut Moscovice dalam Zaki (2013:3) menyatakan bahwa:

Sistem Informasi Akuntansi adalah suatu komponen suatu organisasi yang mengumpulkan, menghasilkan, mengolah, menganalisa dan mengkomunikasikan informasi finansial dan pengambilan keputusan yang *relevan* kepada pihak diluar perusahaan (seperti kantor pajak, *investor*) dan pihak *intern* (terutama manajemen).

Menurut James A. Hall (2011:11) menyatakan bahwa “Sistem Informasi Akuntansi terdiri atas catatan-catatan dan metode yang digunakan untuk memulai, mengidentifikasi, menganalisis dan mencatat transaksi organisasi untuk memperhitungkan aktiva dan kewajiban terkait”.

Sedangkan Menurut Krismiaji dalam Pandji (2010:4) menyatakan bahwa, “Sistem Informasi Akuntansi sebuah sistem yang memproses data dan

transaksi guna menghasilkan informasi yang bermanfaat untuk merencanakan mengendalikan dan mengoperasikan bisnis”.

Dari pendapat-pendapat tersebut dapat dijelaskan bahwa sistem informasi akuntansi adalah kumpulan dari sumber-sumber seperti orang dan peralatan yang dirancang untuk memberikan informasi data keuangan dan data lainnya kepada para pembuat keputusan.

Untuk dapat menghasilkan informasi yang diperlukan oleh para pembuat keputusan, sistem informasi akuntansi harus melaksanakan tugas-tugas sebagai berikut:

1. Mengumpulkan transaksi dan data lain dan memasukkannya kedalam sistem.
2. Memproses data transaksi.
3. Menyimpan data untuk keperluan dimasa mendatang.
4. Menghasilkan informasi yang diperlukan dengan memproduksi laporan atau memungkinkan para pemakai untuk melihat sendiri data yang tersimpan di komputer.
5. Mengendalikan seluruh proses sedemikian rupa sehingga informasi yang dihasilkan akurat dan dapat dipercaya.

Jika dihubungkan dengan jenis-jenis sistem di atas, maka sistem informasi akuntansi merupakan jenis sistem yang relatif tertutup, karena sistem ini mengolah input menjadi output dengan memanfaatkan pengendalian intern untuk membatasi dampak lingkungan. Input sebuah sistem informasi akuntansi adalah transaksi atau kejadian ekonomi, misalnya penjualan secara tunai penjualan secara kredit, pembayaran biaya-biaya, dan sebagainya. Transaksi-transaksi tersebut

selanjutnya diproses dengan mencatatnya kedalam jurnal, diposting ke rekening-rekening buku besar dan diikhtisarkan dalam berbagai macam laporan output dari sistem informasi akuntansi adalah laporan keuangan dan laporan manajemen.

2.1.10. Pengertian Kas

Menurut Dwi Martani, dkk (2012:80) mengemukakan bahwa “kas adalah aset keuangan yang digunakan untuk kegiatan operasional perusahaan. Kas merupakan aset yang likuid karena digunakan untuk membayar kewajiban perusahaan”.

Kas meliputi uang logam, uang kertas, cek, wesel pos dan deposito. Perangko bukan merupakan kas melainkan biaya yang dibayar dimuka atau beban yang ditangguhkan. Pada umumnya perusahaan membagi kas menjadi dua kelompok, yaitu:

1. Kas kecil (*Petty Cash/Cash on Hand*)

Merupakan uang kas yang ada dalam brankas perusahaan yang digunakan untuk membayar dalam jumlah yang relatif kecil, misalnya pembelian perangko, biaya perjalanan, dan pembayaran lain dalam jumlah kecil.

2. Kas di Bank (*Cash in Bank*)

Merupakan uang kas yang dimiliki oleh perusahaan yang tersimpan di Bank dalam bentuk giro/bilyet dan kas ini dipakai untuk pembayaran yang jumlahnya besar dengan menggunakan cek.

2.1.11. Konsep Dasar Kas Kecil

Menurut Hery (2013:172) mengemukakan bahwa “Kas kecil merupakan sisa uang kas perusahaan yang tidak tersimpan di Bank umumnya tersedia dikasir perusahaan untuk memenuhi pembayaran-pembayaran yang jumlahnya relatif kecil (sebagai dana kas kecil / *petty cash fund*) dan juga untuk memenuhi keperluan pembayaran khusus”.

Dan menurut Henry Simamora (2010:213) mengemukakan bahwa “Kas kecil (*petty cash*) adalah dana kas yang dipakai untuk membayar pengeluaran-pengeluaran yang nilainya relatif kecil”.

Ada dua metode dalam mengelola kas kecil yaitu :

1. Metode Imprest

Metode ini dijalankan dengan menentukan jumlah dalam rekening kas kecil selalu tetap, yaitu sebesar *check* yang diserahkan kepada kasir kas kecil untuk dana kas kecil.

Oleh kasir kas kecil, *check* tersebut diuangkan ke bank dan uangnya digunakan untuk membayar pengeluaran-pengeluaran kecil. Apabila jumlah kas kecil tinggal sedikit dan juga pada akhir periode, kasir kas kecil akan minta pengisian kembali kas kecilnya sebesar jumlah yang sudah dibayar dari kas kecil.

2. Metode Fluktuasi

Dalam metode ini penentuan dana kas kecil dilakukan dengan yang sama seperti dalam sistem *imprest*. Perbedaannya adalah dalam sistem metode

fluktuasi saldo rekening kas kecil tidak tetap, tapi berfluktuasi sesuai dengan jumlah pengisian kembali dan pengeluaran-pengeluaran dari kas kecil.

Kalau dalam sistem *imprest* pencatatan terhadap pengeluaran-pengeluaran kas kecil baru dilakukan pada saat pengisian kembali dalam metode fluktuasi setiap terjadi pengeluaran uang dari kas kecil langsung dicatat.

2.1.12. Siklus Akuntansi

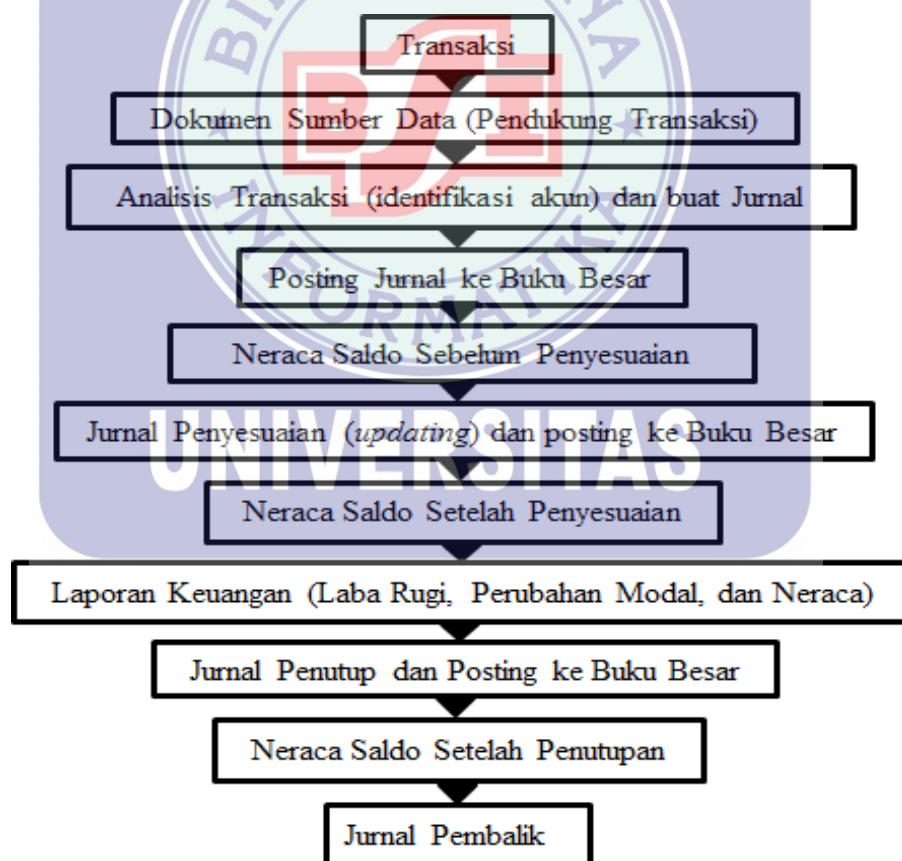
Menurut Hery (2013:66) mengemukakan bahwa “Proses akuntansi yang diawali dengan menganalisis dan menjurnal transaksi, dan yang diakhiri dengan membuat laporan dinamakan sebagai siklus akuntansi (*accounting cycle*)”.

Secara lebih rinci tahapan-tahapan dalam siklus akuntansi dapat diurutkan sebagai berikut :

1. Mula-mula dokumen pendukung transaksi dianalisis dan informasi yang terkandung dalam dokumen tersebut dicatat dalam jurnal.
2. Lalu data akuntansi yang ada dalam jurnal diposting ke buku-buku besar.
3. Seluruh saldo akhir yang terdapat pada masing-masing buku besar akun “didaftar” (dipindahkan) ke neraca saldo untuk membuktikan kecocokan antara keseluruhan nilai akun yang bersaldo normal kredit.
4. Menganalisis data penyesuaian dan membuat ayat jurnal penyesuaian.
5. Memposting data jurnal penyesuaian ke masing-masing buku besar akun yang terkait.

6. Dengan menggunakan pilihan (*optional*) bantuan neraca lajur sebagai kertas kerja (*worksheet*), neraca saldo setelah penyesuaian (*adjusted trial balance*) dan laporan keuangan disiapkan.
7. Membuat ayat jurnal penutup (*closing entries*).
8. Memposting data jurnal penutup ke masing-masing buku besar akun yang terkait.
9. Meyiapkan neraca saldo setelah penutupan (*post-closing trial balance*).
10. Membuat ayat jurnal pembalik (*reversing entries*).

Jika digambarkan dalam bagan arus, tahapan siklus akuntansi akan tampak sebagai berikut :



Sumber: Hery (2013:67)

Gambar II.1
Bagan Siklus Akuntansi

2.1.13. Penjurnalan

Menurut Mulyadi (2010:101) mengemukakan bahwa “Jurnal merupakan catatan akuntansi permanen yang pertama, yang digunakan untuk mencatat transaksi keuangan perusahaan.”

Contoh Jurnal :

Dalam sistem dana kas kecil dengan *fluctuating fund balance system* saldo rekening dana kas kecil didalam buku besar dibiarkan berfluktuasi sesuai dengan jumlah pengisian dan pemakaian dana kas kecil. Jurnal yang dibuat yang bersangkutan dengan pembentukan dana kas kecil adalah sebagai berikut :

1. Pembentukan dana kas kecil dicatat dalam register bukti kas keluar dengan cek dengan jurnal:

Register bukti kas keluar:

Dana kas kecil	xxx	
Bukti kas keluar yang akan dibayar		xxx

Register cek:

Bukti kas keluar yang akan dibayar	xxx	
Kas		xxx

2. Pengeluaran dana kas kecil dicatat dengan jurnal pengeluaran dana kas kecil dengan jurnal:

Biaya Overhead Pabrik	xxx	
Biaya administrasi dan umum	xxx	
Biaya Pelaksanaan	xxx	
Dana kas kecil		xxx

3. Pengisian dana kembali kas kecil dicatat dengan register bukti kas keluar dan register cek dengan jurnal:

Register bukti kas keluar:

Dana kas kecil	xxx	
	Bukti kas keluar yang akan dibayar	xxx

Register cek:

Bukti kas keluar yang akan dibayar	xxx
------------------------------------	-----

Kas

xxx

2.2. Peralatan Pendukung (*Tools System*)

Adapun peralatan pendukung yang dimaksud untuk merancang model sistem yang baru pada penulisan laporan Tugas Akhir (TA) ini adalah :

2.2.1. *Unified Modeling Language (UML)*

2.2.1.1. Konsep Dasar *Unified Modeling Language (UML)*

Menurut Rosa A.S dan M. Shalahuddin (2013:133) mengemukakan bahwa “*Unified Modeling Language (UML)* adalah salah satu standar bahasa yang banyak digunakan didunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek”.

Pada tahun 1996, *Object Management Group (OMG)* mengajukan proposal adanya standarisasi pemodelan berorientasi objek dan pada bulan September 1997 UML diakomodasi oleh OMG sehingga sampai saat ini UML

telah memberikan kontribusinya yang cukup besar didalam metodologi berorientasi objek dan hal-hal yang terkait didalamnya.

Secara fisik, UML adalah sekumpulan spesifikasi yang dikeluarkan oleh OMG. UML terbaru adalah UML 2.3 yang terdiri dari 4 macam spesifikasi, yaitu *Diagram Interchange Specification*, *UML Infrastructure*, *UML Superstructure*, dan *Object Constraint Language (OCL)*.

2.2.1.2. Diagram Unified Modeling Language (UML)

1. Use Case Diagram

Menurut Rosa A.S dan M.Shalahuddin (2013:155) mengemukakan bahwa:

Use Case Diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use Case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasidan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin sehingga dapat dengan mudah dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

Komponen-komponen yang ada pada *use case* adalah:

1. Aktor, merupakan orang, proses atau sistem lain yang berinteraksi dengan sistem yang akan dibuat. Jadi walaupun simbol aktor dalam *use case diagram* berbentuk orang, namun aktor belum tentu orang.
2. *Use case*, merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling berinteraksi atau bertukar pesan antar unit maupun aktor.
3. Relasi, merupakan hubungan yang terjadi pada sistem baik antar aktor maupun antara *use case* dan aktor.

2. Activity Diagram

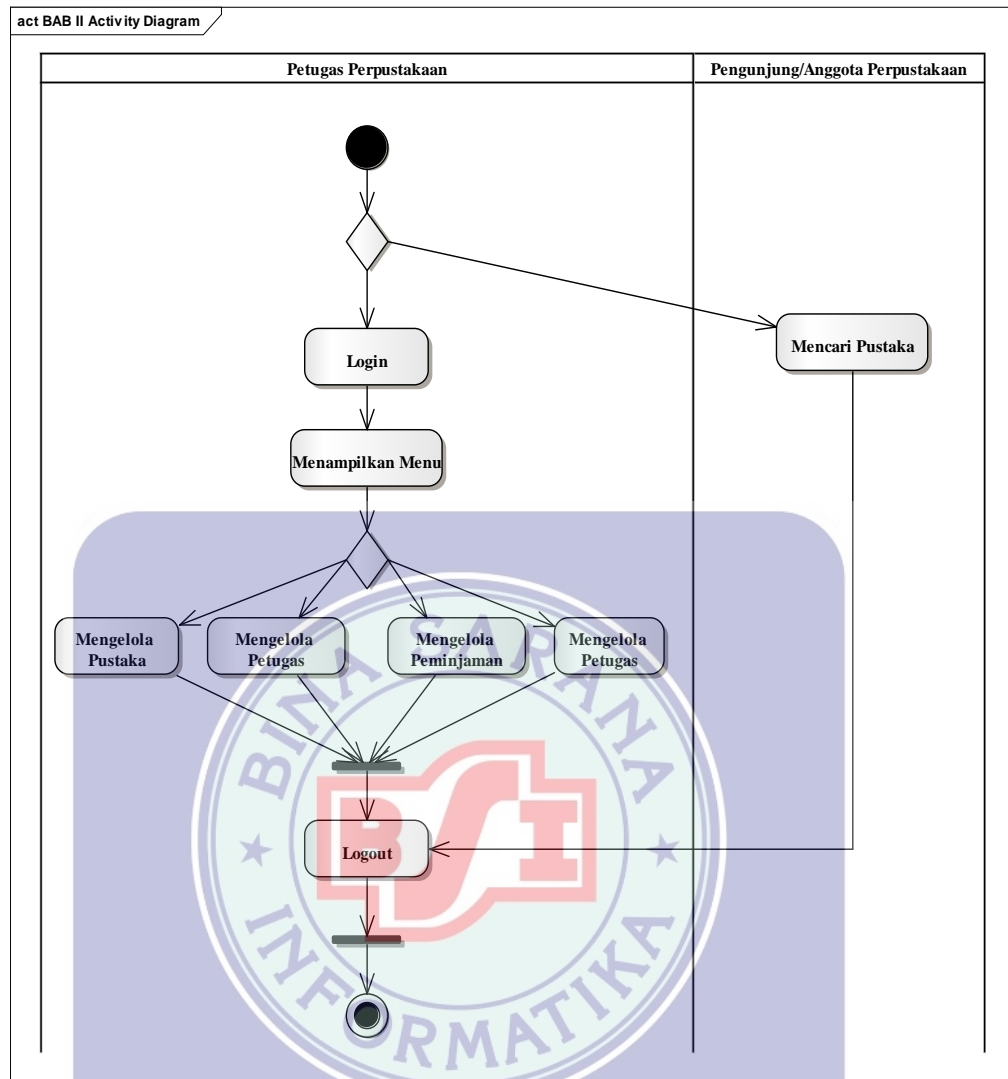
Menurut Rosa A.S dan M. Shalahuddin (2013:161) mengemukakan bahwa “Diagram aktifitas atau *activity diagram* adalah diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak”.

Sesuai dengan namanya diagram ini menggambarkan tentang aktifitas yang terjadi pada sistem. Dari pertama sampai akhir, diagram ini menunjukkan langkah-langkah dalam proses kerja sistem yang kita buat. Fungsi *activity diagram* adalah untuk menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses, memperlihatkan urutan aktivitas proses pada sistem, dan *activity diagram* dibuat berdasarkan sebuah atau beberapa *use case* pada *use case diagram*.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokkan tampilan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut ini adalah contoh *Activity Diagram* dari sistem informasi manajemen perpustakaan menurut Rosa A.S dan M.Shalahuddin(2013:234):



Sumber: Rosa A.S dan M. Shalahuddin (2013:235)

Gambar II.3

Activity Diagram

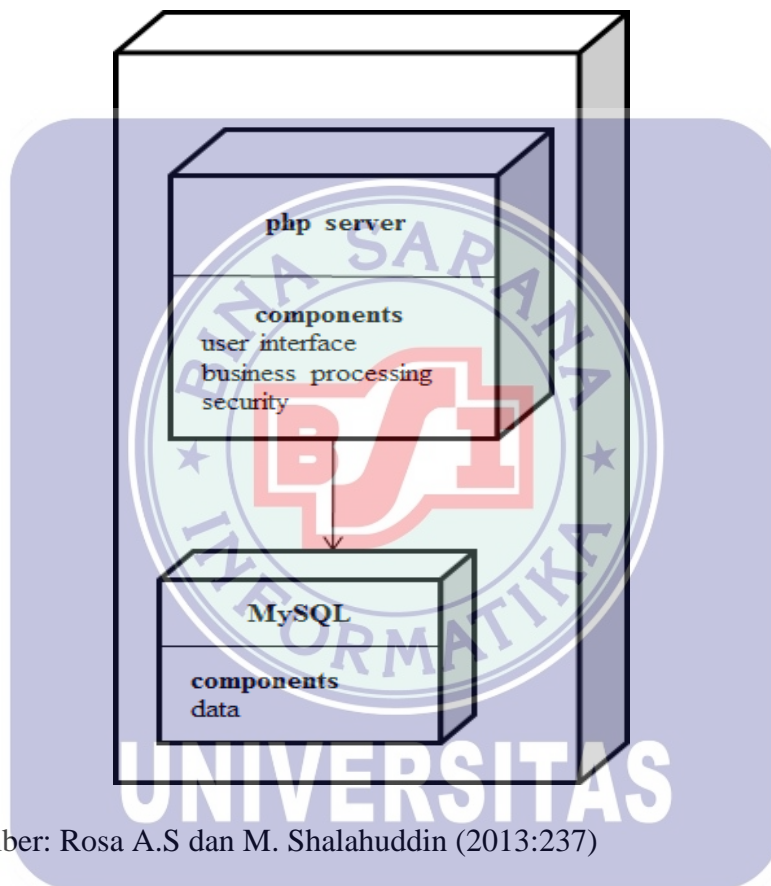
3. *Deployment Diagram*

Deployment diagram menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. *Deployment diagram* juga dapat digunakan untuk memodelkan hal-hal berikut:

1. Sistem tambahan (*embedded system*) yang menggambarkan rancangan *device*, *node*, dan *hardware*.

2. Sistem *client/server*.
3. Sistem terdistribusi murni.
4. Rekayasa ulang aplikasi.

Berikut ini adalah contoh *Deployment Diagram* dari sistem informasi manajemen perpustakaan menurut Rosa A.S dan M.Shalahuddin (2013:237):



Sumber: Rosa A.S dan M. Shalahuddin (2013:237)

Gambar II.4

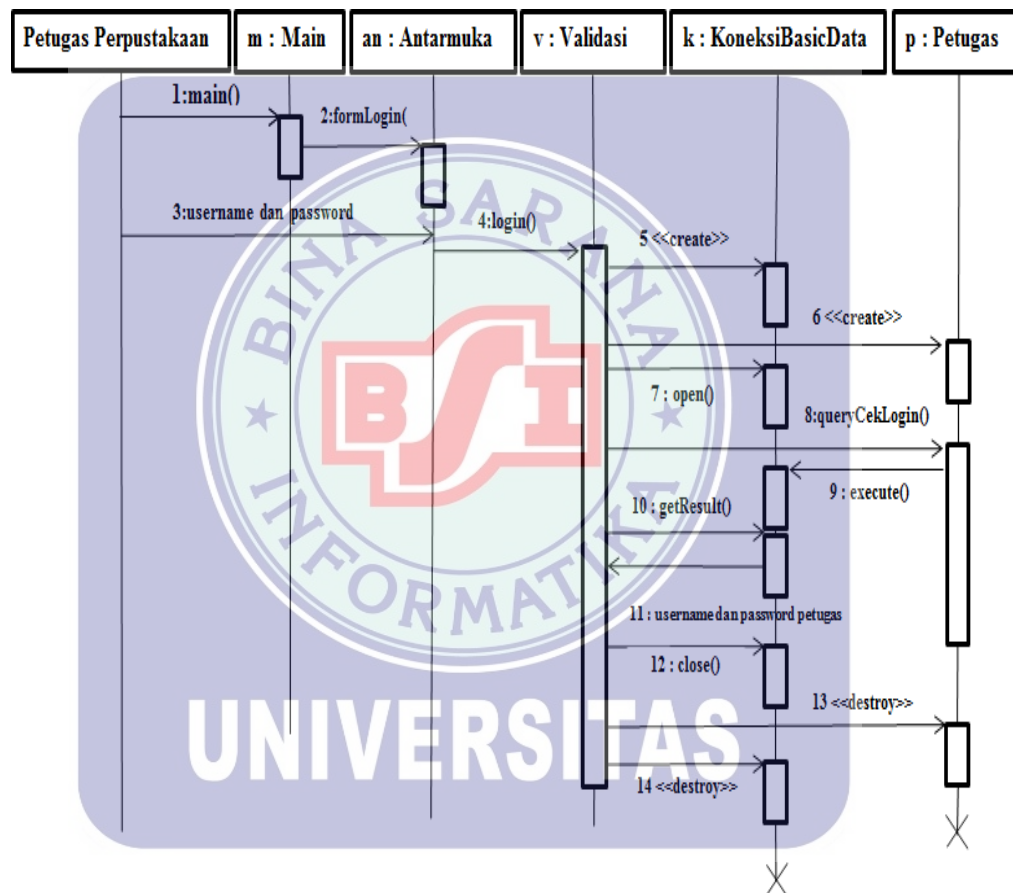
Deployment Diagram

4. *Sequence Diagram*

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar *sequence diagram*

maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat *sequence diagram* juga dibutuhkan untuk melihat skenario yang ada pada *use case*.

Berikut ini adalah contoh *Sequence Diagram* dari sistem informasi manajemen perpustakaan menurut Rosa A.S dan M.Shalahuddin (2013:209):



Sumber: Rosa A.S dan M. Shalahuddin (2013:209)

Gambar II.5

Sequence Diagram

5. Class Diagram

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa

yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Sedangkan operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga *programmer* dapat membuat kelas-kelas dalam program perangkat lunak sesuai dengan perancangan *class diagram*.

6. Object Diagram

Object diagram menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. Pada *object diagram* harus dipastikan semua kelas yang sudah didefinisikan pada *class diagram* harus dipakai objeknya, karena jika tidak dipakai, pendefinisian kelas itu tidak dapat dipertanggungjawabkan. *Object diagram* juga berfungsi untuk mendefinisikan contoh nilai atau isi dari atribut tiap kelas.

Hubungan *link* pada *object diagram* merupakan hubungan memakai dan dipakai dimana dua buah objek akan dihubungkan oleh *link* jika ada objek yang dipakai oleh objek lainnya.

Penulisan nilai sama dengan penulisan pada kamus data DFD. Misalkan jika sebuah atribut dapat berisi lebih dari satu string maka akan ditulis dengan lambang {"nilai1", "nilai2", ..., "nilain"}, atau misalkan ingin menuliskan bahwa nilai sebuah atribut string dapat diisi nilai1 atau nilai2 maka akan ditulis ["nilai1" | "nilai2"].

7. *Component Diagram*

Diagram komponen atau *component diagram* dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Komponen diagram berfokus pada komponen sistem yang dibutuhkan dan ada didalam sistem. Diagram komponen juga dapat digunakan untuk memodelkan hal-hal berikut:

1. *Source code* program perangkat lunak.
2. Komponen *executable* yang dilepas ke *user*.
3. Basis data secara fisik.
4. Sistem yang harus beradaptasi dengan sistem lain.
5. *Framework* sistem, *framework* pada perangkat lunak merupakan kerangka kerja yang dibuat untuk memudahkan pengembangan dan pemeliharaan aplikasi.

8. *Composite Structure Diagram*

Composite Structure Diagram mulai ada pada UML versi 2.0, pada versi 1.x diagram ini belum muncul. Diagram ini dapat digunakan untuk menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat berjalan (*runtime*) dari *instance* yang saling terhubung.

Composite Structure Diagram dapat menggambarkan struktur didalam kelas atau kolaborasi. Contoh penggunaan diagram ini misalnya untuk menggambarkan deskripsi dari setiap bagian mesin yang saling terkait untuk menjalankan fungsi mesin tersebut, menggambarkan aliran data *router* pada jaringan komputer, dan lain-lain.

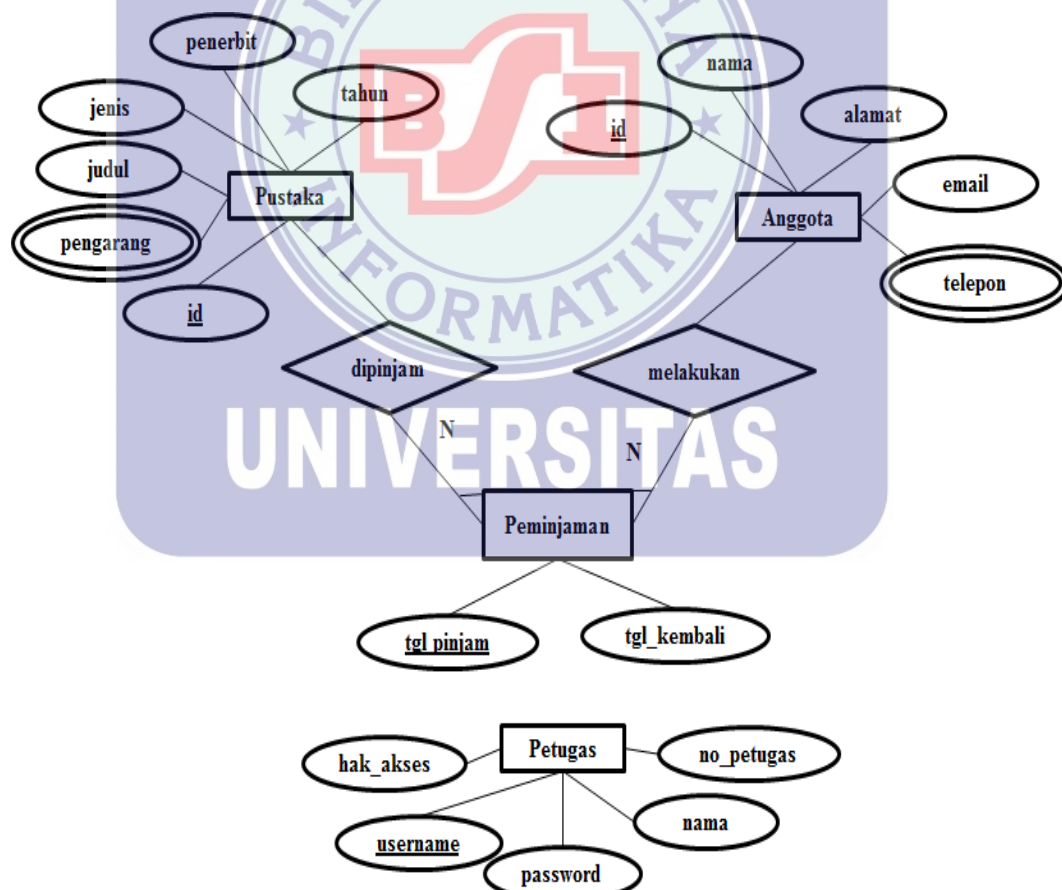
9. Package Diagram

Package Diagram menyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML. Hampir semua diagram dalam UML dapat dikelompokkan menggunakan *package diagram*.

2.2.2. Entity Relationship Diagram (ERD)

Menurut Rosa A.S dan M.Shalahuddin (2013:50) mengemukakan bahwa “Pemodelan awal basis data yang paling banyak digunakan adalah menggunakan ERD. ERD digunakan untuk pemodelan basis data relasional.”

Contoh *Entity Relationship Diagram* (ERD) adalah sebagai berikut:



Sumber: Rosa A.S dan M. Shalahuddin (2013:58)

Gambar II.6
Entity Relationship Diagram

2.2.3. Logical Record Structure (LRS)

Menurut Lestari (2013) “*Logical Record Structure* dibentuk dengan nomor *record type*. Beberapa *record type* digambarkan oleh kotak empat persegi panjang dan dengan nama yang unik”.

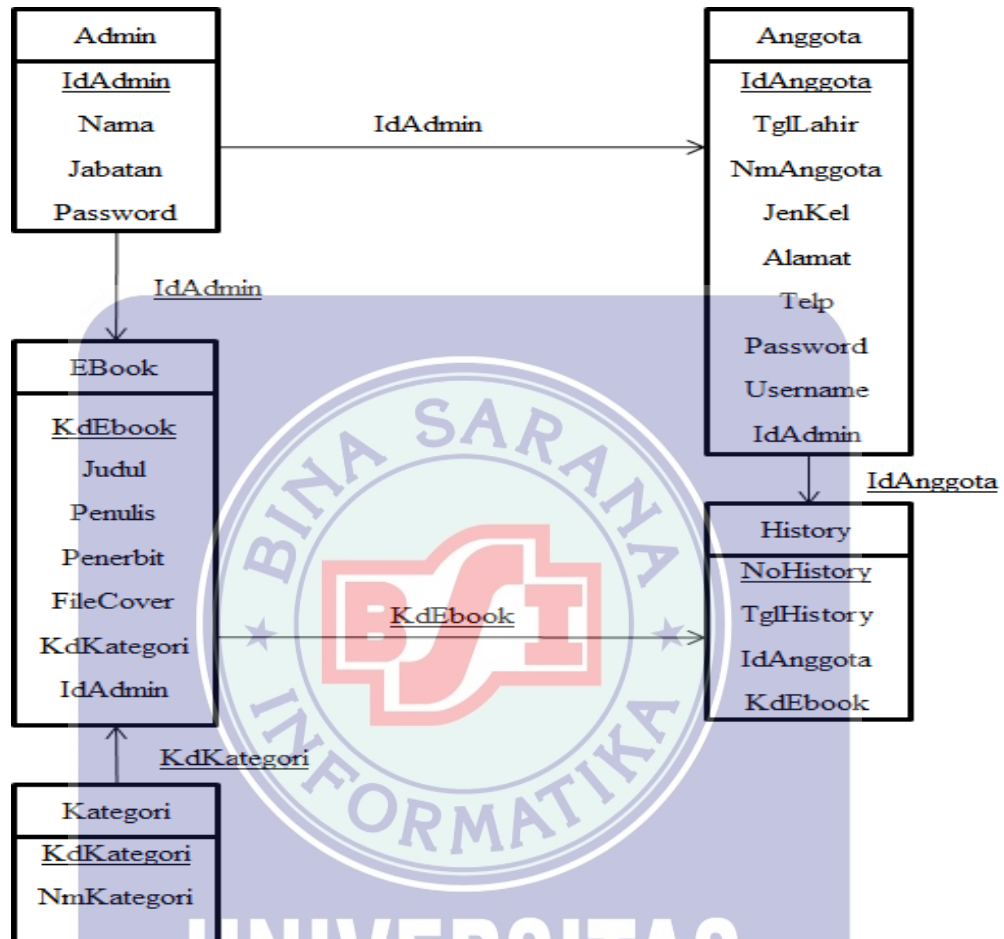
Menurut Hasugian dan Shidiq (2012:608) memberikan batasan bahwa “LRS adalah sebuah model sistem yang digambarkan dengan sebuah diagram ER akan mengikuti pola atau aturan permodelan tertentu dalam kaitannya dengan konvensi ke LRS”. Perubahan yang terjadi yaitu mengikuti aturan-aturan sebagai berikut:

1. Setiap entitas akan diubah kebentuk kotak.
2. Sebuah atribut relasi disatukan dalam sebuah kotak bersama entitas jika hubungan yang terjadi pada diagram ER 1:M (relasi bersatu dengan *cardinality* M) atau tingkat hubungan 1:1 (relasi bersatu dengan *cardinality* yang paling membutuhkan referensi).
3. Sebuah relasi dipisah dalam sebuah kotak tersendiri (menjadi entitas baru) jika tingkat hubungan M:M (*many to many*) dan memiliki *foreign key* sebagai *primary key* yang diambil dari kedua entitas yang sebelumnya saling berhubungan.

Perbedaan LRS dengan ERD dan tipe *record* berada diluar *field* tipe *record* ditempatkan. LRS 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 type record*. Penggambaran LRS mulai dengan menggunakan model yang dimengerti. Dua metode yang dapat digunakan yaitu dimulai dengan hubungan kedua model yang dapat dikonversikan

ke LRS. Metode yang lain dimulai dengan ERD dan langsung dikonversikan ke LRS.

Berikut ini Contoh *Logical Record Structure (LRS)*:



Sumber: Jurnal SIFOM Hermin Luisiah (2015:5)

Gambar II.7

Logical Record Structure

2.2.4. Pengkodean

Pengkodean (*encoding*) adalah proses perubahan karakter data yang akan dikirim dari suatu titik ke titik lain dengan kode yang dikenal oleh setiap terminal yang ada dan menjadikan setiap karakter data dalam sebuah informasi

digital kedalam bentuk *biner* agar dapat ditransmisikan. Suatu terminal yang berbeda menggunakan kode *biner* yang berbeda untuk mewakili setiap karakter.

Menurut Rosa A.S dan M. Shalahuddin (2013:29) mengemukakan bahwa “Pengkodean harus ditranslasikan kedalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain”.

1. Tipe Kode

Berdasarkan jenisnya struktur kode diantaranya yaitu:

1. Kode Mnemonik (*Mnemonic Code*)

Kode mnemonik digunakan untuk tujuan supaya mudah diingat. Kode mnemonik dibuat dengan dasar singkatan sebagian karakter dari item yang mewakili kode ini.

Contohnya:

KD : Kamus Data

SO : Solo

YG : Yogyakarta

2. Kode Urut (*Sequential Code*)

Kode urut disebut juga kode seri merupakan kode yang nilai urut antara suatu kode dengan kode berikutnya.

Contohnya:

00 Kas

002 Piutang Dagang

003 Persediaan Barang Dagang

3. Kode Blok (*Blok Code*)

Kode blok mengklasifikasikan item ke dalam kelompok blok tertentu yang mencerminkan suatu klasifikasi tertentu atas dasar pemakaian maksimum yang diharapkan.

1000-1999 Aktiva Lancar

2000-2999 Aktiva Tetap

3000-3999 Hutang Lancar

4. Kode Grup (*Group Code*)

Kode grup merupakan kode yang berdasarkan *field-filed* dan tiap-tiap *field* mempunyai arti.

Contohnya:

1 Aktiva Tetap

1.1 Aktiva Lancar

1.1.0 Kas

5. Kode Desimal (*Decimal Code*)

Kode desimal mengklasifikasikan kode atas dasar 10 unit angka desimal dimulai dari angka 0 sampai 9 atau dari 00 sampai 99 tergantung banyaknya kelompok.

Contohnya:

00 Aktiva Lancar

00100 Kas

00200 Piutang Dagang

00300 Persediaan Produk Selesai

2. Pedoman Umum Pengkodean

Pedoman umum pengkodean diantaranya yaitu:

1. Meringkas

Kode seharusnya diringkas. Kode yang terlalu panjang berarti banyak tombol dan akibatnya banyak kesalahan. Kode panjang juga berarti bahwa penyimpanan informasi dalam basisdata akan memerlukan banyak memori.

2. Menjaga Kode tidak Berubah

Kestabilan berarti bahwa identifikasi kode untuk pelanggan seharusnya tidak berubah setiap kali data diterima.

3. Memastikan Bahwa Kode adalah Unik

Bagi kode supaya bekerja, harus unik perhatikan bahwa semua kode yang digunakan dalam sistem dan memastikan bahwa tidak menggunakan nomor atau nama kode sama untuk item-item yang sama. Nomor dan nama kode merupakan bagian yang sangat penting dari masukan dalam kamus data.

4. Membiarkan Kode dapat Diurut

Membuat data agar bermanfaat, kode harus dapat diurut.

5. Menghindari Kode yang Buat Kekacauan

Menghindari penggunaan karakteristik kode yang terlihat atau terdengar serupa.

6. Menjaga Kode yang Seragam

Kode perlu untuk mengikuti bentuk banyak format sepanjang waktu.

7. Memperbolehkan Modifikasi Kode

Sistem pengkodean seharusnya mampu mencakup perubahan.

8. Membuat Kode Berarti

Kode yang berarti lebih mudah dimengerti.

9. Menggunakan kode-kode bisa digunakan.

2.2.5. Java

Menurut definisi *Sun Microsystem* didalam buku Rosa A.S dan M. Shalahuddin (2010:1) mengemukakan bahwa “*Java* adalah sekumpulan nama teknologi untuk membuat dan menjalankan perangkat lunak pada komputer yang berdiri sendiri (*standalone*) ataupun pada lingkungan jaringan.”

Java sebagai salah satu bahasa pemrograman yang sudah berumur dari era 1990-an, kian berkembang dan melebarkan dominasinya diberbagai bidang. Salah satu penggunaan yang terbesar *Java* adalah dalam pembuatan aplikasi *native* untuk *Android*. Selain itu *Java* pun menjadi pondasi bagi bebagai bahasa pemrograman seperti *Kotlin*, *Scala*, *Clojure*, *Groovy*, *JRuby*, *Jython*, dan lainnya yang memanfaatkan *Java Virtual Machine* sebagai rumahnya.

Setiap bahasa pemrograman pasti memiliki kelebihan dan kekurangan. Kelebihan dari *Java* adalah sebagai berikut:

1. *Multiplatform*

Kelebihan utama dari Java ialah dapat dijalankan di beberapa *platform* / sistem operasi komputer, sesuai dengan prinsip “tulis sekali, jalankan dimana saja”. Dengan kelebihan ini pemrogram cukup menulis sebuah program *Java* dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin/*bytecode*) sekali lalu hasilnya dapat dijalankan di atas beberapa *platform* tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis *java* dikerjakan di atas *operating system Linux* tetapi dijalankan dengan baik di atas *Microsoft Windows*. *Platform* yang didukung sampai saat ini adalah *Microsoft Windows, Linux, Mac OS* dan *Sun Solaris*. Penyebabnya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs *Java*) untuk meninterpretasikan *bytecode* tersebut.

2. OOP (*Object Oriented Programming* - Pemrogram Berorientasi Objek)

3. Perpustakaan Kelas Yang Lengkap

Java terkenal dengan kelengkapan *library*/perpustakaan (kumpulan program program yang disertakan dalam pemrograman *java*) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas *Java* yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. Bergaya C++

Java memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke *Java*. Saat ini pengguna *Java* sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke *Java*.

Universitas-universitas di Amerika Serikat juga mulai berpindah dengan mengajarkan *Java* kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.

5. Pengumpulan Sampah Otomatis

Java memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa *C++* yang dipakai secara luas).

Sedangkan kekurangan *Java* adalah sebagai berikut:

1. Tulis Sekali, Jalankan dimana Saja

Masih ada beberapa hal yang tidak kompatibel antara *platform* satu dengan *platform* lain. Untuk J2SE, misalnya *SWT-AWT bridge* yang sampai sekarang tidak berfungsi pada *Mac OS X*.

2. Mudah Didekompilasi

Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena kode jadi *Java* merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada *Microsoft NET Platform*. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/*reverse-engineer*.

3. Penggunaan Memori yang Banyak

Penggunaan memori untuk program berbasis *Java* jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti *C/C++* dan *Pascal* (lebih spesifik lagi, *Delphi* dan *Object Pascal*). Biasanya ini bukan merupakan

masalah bagi pihak yang menggunakan teknologi terbaru (karena *trend* memori terpasang makin murah), tetapi menjadi masalah bagi mereka yang masih harus berkuat dengan mesin komputer berumur lebih dari 4 tahun.

2.2.6. *NetBeans 8.1*

NetBeans adalah suatu serambi pengembangan perangkat lunak yang dituliskan dalam bahasa pemrograman *Java*. Serambi *NetBeans* pun memperkenalkan suatu pengembangan aplikasi dilakukan dengan dimulai dari sesetel pembentukan kesatuan perangkat lunak modular yang dinamai *modules*.

Menurut Ari Prabawati (2010:4) mengemukakan bahwa:

Netbeans merupakan salah satu *IDE* yang digunakan untuk melakukan pemrograman baik menulis kode, mengkompilasi, mencari kesalahan, dan mendidtribusi program. *Netbeans* juga menyediakan paket yang lengkap dalam pemrograman dari pemrograman standar (aplikasi desktop), pemrograman *enterprise*, dan pemrograman perangkat *mobile*.

Awal kemunculan *NetBeans* telah ada sejak tahun 1997 yaitu sebagai sebuah proyek mahasiswa. Pada tahun tersebut, suatu perusahaan dibangun oleh Roman Stenek disekitar proyek mahasiswa tersebut lalu perusahaan itu memulai memproduksi terbitan-terbitan *NetBeans IDE* yang bersifat perdagangan hingga akhirnya dibeli oleh *Sun Microsystems* ditahun 1999 lalu menjadikan *NetBeans IDE* sebagai serambi bersifat sumber terbuka pada bulan Juni 1999.

Netbeans juga digunakan oleh sang *programmer* untuk menulis, meng-*compile*, mencari kesalahan dan menyebarkan program netbeans yang ditulis dalam bahasa pemrograman *java* namun selain itu dapat juga mendukung bahasa pemrograman lainnya dan program ini pun bebas untuk digunakan dan untuk

membuat *professional desktop, enterprise, web, and mobile applications* dengan *Java language, C/C++*, dan bahkan *dynamic languages seperti PHP, JavaScript, Groovy, dan Ruby*.

2.2.7. XAMPP

Menurut Nugroho (2010:74) mengemukakan bahwa “XAMPP merupakan paket *PHP* yang berbasis *Open Source* yang dikembangkan oleh sebuah komunitas *Open Source*.”

XAMPP ialah perangkat lunak bebas yang mendukung banyak sistem operasi, merupakan campuran dari beberapa program. Yang mempunyai fungsi sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri dari program *MySQL database, Apache HTTP Server*, dan penerjemah ditulis dalam bahasa pemrograman *PHP* dan *Perl*.

Nama XAMPP merupakan singkatan dari X (empat sistem operasi), *Apache, MySQL, PHP* dan *Perl*. Program ini tersedia di bawah *GNU General Public License* dan bebas, adalah mudah untuk menggunakan *web server* yang dapat melayani tampilan halaman *web* yang dinamis. Jika ingin mendapatkan *xampp* dapat men-*download* langsung dari situs resminya.

2.2.8. PhpMyAdmin

Menurut Nugroho (2010:88) mengemukakan bahwa “*PhpMyAdmin* adalah suatu aplikasi *Open Source* yang berbasis *web*, aplikasi ini dibuat

menggunakan program *PHP*, fungsi aplikasi ini adalah untuk mengakses *database MySQL*.”

PhpMyAdmin adalah sebuah aplikasi/perangkat lunak bebas (*opensource*) yang ditulis dalam bahasa pemrograman *PHP* yang digunakan untuk menangani administrasi *database MySQL* melalui jaringan lokal maupun internet. *PhpMyAdmin* mendukung berbagai operasi *MySQL*, diantaranya (mengelola basis data, tabel-tabel, bidang (*field*), relasi (*relation*), indeks, pengguna (*user*), perijinan (*permission*), dan lain-lain.

Pada dasarnya, mengelola basis data dengan *MySQL* harus dilakukan dengan cara mengetikkan baris-baris perintah yang sesuai (*command line*) untuk setiap maksud tertentu. Jika seseorang ingin membuat basis data (*database*), ketikkan baris perintah yang sesuai untuk membuat basis data. Jika seseorang menghapus tabel, ketikkan baris perintah yang sesuai untuk menghapus tabel. Hal tersebut tentu saja sangat menyulitkan karena seseorang harus hafal dan mengetikkan perintahnya satu per satu.

Saat ini banyak sekali perangkat lunak yang dapat dimanfaatkan untuk mengelola basis data dalam *MySQL*, salah satunya adalah *phpMyAdmin*. Dengan *phpMyAdmin*, seseorang dapat membuat *database*, membuat tabel, mengisi data, dan lain-lain dengan mudah, tanpa harus menghafal baris perintahnya.

PhpMyAdmin merupakan bagian untuk mengelola basis data *MySQL* yang ada di komputer. Untuk membukanya, buka browser lalu ketikkan alamat

<http://localhost/phpmyadmin>, maka akan muncul halaman *phpMyAdmin*. Disitu nantinya seseorang bisa membuat (*create*) basis data baru, dan mengelolanya.

2.2.9. MySQL

Menurut Nugroho (2010:91) mengemukakan bahwa “*MySQL (My Structured Query Language)* atau biasa dibaca ai-se-kuel adalah sebuah program pembuatan dan pengelola *database* atau sering disebut *DBMS (Database Management System)*).

MySQL mulai berkembang pada tahun 1970-an. *MySQL* mulai digunakan sebagai standar yang resmi pada tahun 1986 oleh *American National Standart Institute* dan pada tahun 1987 oleh *ISO (International Organization for Standarization)* dan disebut sebagai *SQL-86*.

Meskipun *MySQL* diadopsi dan diacu sebagai bahasa standar oleh hampir sebagian besar *RDMS (Relational Database Management System)* yang beredar saat ini, tetapi tidak semua standar yang tercantum dalam *MySQL* diimplementasikan oleh *DBMS* tersebut. Sehingga terkadang ada perbedaan perilaku (hasil yang ditampilkan) oleh *DBMS* yang berbeda padahal *query* yang dimasukkan sama.

2.2.10. Spesifikasi *File* dan Dokumen

Spesifikasi *file* adalah hasil normalisasi data yang hanya menunjukkan atribut (*field*) apa saja yang terdapat pada sebuah *file*. Spesifikasi *file* memberikan rincian yang lebih lengkap berisi kode file, organisasi, *primary key*, panjang *record*, dan deskripsi *field* (no, nama *field*, *type field*, keterangan). Spesifikasi file

ini terdiri dari *file-file* yang digunakan untuk menyimpan data maupun proses pengolahan data.

Spesifikasi *file* merupakan format yang diperlukan dan harus diikuti dalam rangka untuk data *file* yang akan diproses. spesifikasi *file* ini terdiri dari *file-file* yang digunakan untuk menyimpan data maupun proses pengolahan data.

Tugas menentukan spesifikasi *file* berdasarkan parameter dari *file* meliputi:

1. Tipe dari *file*.
2. Media dari *file*.
3. Organisasi dari *file*.
4. *Field* kunci dari *file*.
5. Tabel dari spesifikasi *file*.

