BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Demi kesempurnaan penelitian, maka penulis perlu melakukan perbandingan untuk mengetahui gambaran dari penelitian terdahulu.

1. Menurut Deny Kurniawan (2013), dalam jurnalnya yang berjudul "Pengaruh Perubahan Laporan Modal Terhadap Perubahan *Profitabilitas* Pada Perusahaan Manufaktur Di Bursa Efek Jakarta (BEJ)", Tujuan penelitian ini adalah untuk mendapatkan bukti empiris mengenai masalah yang diteliti yaitu tentang pengaruh perubahan modal terhadap kerja dan terhadap perubahan *profitabilitas* pada perusahaan manufaktur *go* publik di Bursa Efek Jakarta (BEJ). Kelebihan dari penelitian ini adalah untuk mengetahui Pengelolaan modal kerja yang sangat penting karena menyangkut penetapan kebijakan modal kerja maupun pelaksanaan kebijakan modal kerja tersebut dalam operasi sehari-hari. Manajemen modal kerja berkepentingan terhadap keputusan investasi pada aktiva lancar dan hutang lancar terutama mengenai bagaimana menggunakan dan komposisi keduanya akan mempengaruhi resiko. Sedangkan kekurangan dari penelitian ini adalah Manajemen modal kerja berkepentingan kurang efektif terhadap keputusan investasi pada aktiva lancar dan hutang lancar terutama mengenai bagaimana menggunakan dan komposisi keduanya akan mempengaruhi resiko. Modal kerja dipergunakan perusahaan untuk membiayai kegiatan operasi perusahaan.

2. Menurut Zulia Hanum (2013), dalam jurnalnya yang berjudul "Pengaruh *Profitabilitas* Terhadap Modal Kerja Pada Perusahaan Makanan Dan Minuman Yang Terdaftar Di Bursa Efek Indonesia", Tujuan dari peneletian ini adalah untuk mengetahui apakah ada pengaruh antara *Profitabilitas* baik secara parsial maupun simultan terhadap Modal Kerja pada Perusahaan Perbankan yang terdaftar di Bursa Efek Indonesia. Pendekatan yang dilakukan dalam penelitian ini adalah pendekatan asosiatif. Populasi dalam penelitian ini adalah 16 Perusahaan Makanan dan Minuman yang terdaftar di Bursa Efek Indonesia (BEI) selama periode 2008 – 2012, sedangkan sampel yang memenuhi kriteria dalam pengambilan sampel untuk penelitian ini adalah 11 Perusahaan Makanan dan Minuman yang terdaftar di Bursa Efek Indonesia (BEI) selama periode tahun 2008 – 2012. Hasil penelitian menunjukkan bahwa secara parsial *Profitabilitas Return On Asset* (*ROA*) tidak berpengaruh signifikan terhadap Modal Kerja. *Profitabilitas Return On Equity* (*ROE*) tidak berpengaruh signifikan terhadap Modal Kerja. Secara simultan, *Profitabilitas* (*ROA* dan *ROE*) tidak berpengaruh signifikan terhadap Modal Kerja pada Perusahaan Makanan dan Minuman yang terdaftar di Bursa Efek Indonesia (BEI).
3. Menurut Cholifia Dwi Agustin Pangestuti (2016) dalam jurnalnya yang berjudul "Pengaruh Perputaran Modal Kerja, Ukuran Perusahaan, *Operating Leverage, Financial Leverage* Terhadap Profitabilitas" Penelitian ini bertujuan untuk mengetahui dan menganalisis pengaruh perputaran modal kerja, ukuran perusahaan, *operating leverage*, dan *financial leverage* terhadap *profitabilitas*. Populasi yang digunakan dalam penelitian ini sebanyak 23 perusahaan retail

yang terdaftar di Bursa Efek Indonesia. Berdasarkan uji t diketahui bahwa variabel perputaran modal kerja secara parsial berpengaruh signifikan terhadap *profitablitas*, karena nilai signifikansi lebih kecil dari 0,05 yaitu sebesar 0,011. Variabel ukuran perusahaan juga menunjukkan pengaruh signifikan terhadap *profitabilitas*, karena nilai signifikansi lebih kecil dari 0,05 yaitu sebesar 0,002. Variabel *operating leverage* berpengaruh tidak signifikan terhadap *profitabilitas*, Melihat dari hasil koefisien determinasi dapat disimpulkan bahwa variabel yang mempunyai pengaruh dominan adalah ukuran perusahaan karena mempunyai nilai paling tinggi yaitu sebesar 22,66.

4. Menutur Lisnawati Dewi (2016) dalam jurnal yang berjudul " Pengaruh Perputaran Modal Kerja Terhadap *Profitabilitas* Perusahaan Manufaktur Di Bursa Efek Indonesia" Penelitian ini bertujuan untuk menguji pengaruh perputaran kas, perputaran piutang, dan perputaran persediaan terhadap *profitabilitas* pada perusahaan manufaktur sektor industri barang konsumsi yang terdaftar di Bursa Efek Indonesia (BEI) periode 2010-2014. Variabel independen pada penelitian ini adalah perputaran kas, perputaran piutang dan perputaran persediaan, sedangkan untuk variabel dependen adalah *profitabilitas.* Sampel penelitian terdiri atas 19 perusahaan yang dipilih secara *purposive sampling*. Data laporan keuangan diperoleh dari Bursa Efek Indonesia (BEI). Metode penelitian yang digunakan adalah metode kuantitatif dengan teknik analisis regresi linier berganda dengan alat bantu aplikasi SPSS (*Statistical Product and Service Solutions*). Hasil penelitian ini menunjukkan bahwa perputaran kas dan perputaran persediaan tidak berpengaruh terhadap

profitabilitas, sedangkan perputaran piutang berpengaruh terhadap *profitabilitas.*

5. Menurut Ni Made Rustia Dewi (2015) dalam jurnal yang berjudul "Pengaruh Manajemen Modal Kerja Pada *Profitabilitas* Perusahaan Manufaktur" Tujuan penelitian ini untuk menguji pengaruh manajemen modal kerja pada *profitabilitas* perusahaan manufaktur yang terdaftar di Bursa Efek Indonesia periode 2011-2013. Variabel independen yang diujikan adalah periode pengumpulan piutang rata-rata *Average collection Period* (ACP), periode perputaran persediaan harian *Inventory Turnover In Days* (ITID), dan periode rata-rata pembayaran utang *Average Payment Period* (APP) sedangkan variabel dependen dalam penelitian ini adalah profitabilitas yang diproksikan dengan *gross profit margin* (GPM). Sampel penelitian berjumlah 61 perusahaan dengan metode *nonprobability* sampling. Data dianalisis menggunakan teknik analisis regresi linier berganda. Hasil pengujian menunjukkan bahwa periode pengumpulan piutang rata-rata *collection Period* (ACP), periode perputaran persediaan harian *Inventory Turnover In Days* (ITID), dan periode rata-rata pembayaran utang *Average Payment Period* (APP) berpengaruh negatif pada *profitabilitas.*

II.2. Landasan Teori

II.2.1. Sistem

Definisi sistem adalah "kumpulan dari bagian-bagian yang bekerja sama untuk mencapai tujuan yang sama. "Definisi sistem adalah "sekumpulan

objek-objek yang saling berelasi dan berinteraksi serta hubungan antar objek bisa dilihat sebagai suatu kesatuan yang dirancang untuk mencapai suatu tujuan. Sistem adalah penggabungan dari bagian-bagian atau komponen-komponen yang terpisah-pisah dan disatukan menjadi satu rangkaian dan menjadi suatu fungsi yang baru (Aris : 2015).

II.2.1.1. Karakteristik Sistem

Untuk memahami atau mengembangkan suatu sistem, maka perlu membedakan unsur-unsur dari sistem yang membentuknya. Berikut adalah karakteristik sistem yang dapat membedakan suatu sistem dengan sistem lainnya yaitu:

1. Batasan (*boundary*) : Penggambaran dari suatu elemen atau unsur mana yang termasuk di dalam sistem dan mana yang di luar sistem.
2. Lingkungan (*environment*) : Segala sesuatu di luar sistem, lingkungan yang menyediakan asumsi, kendala, dan input terhadap suatu sistem.
3. Masukan (*input*) : Sumber daya (data, bahan baku, peralatan, energi) dari lingkungan yang dikonsumsi dan dimanipulasi oleh suatu sistem.
4. Keluaran (*output*) : Sumber daya atau produk (informasi, laporan, dokumen, tampilan layar komputer, barang jadi) yang disediakan untuk lingkungan sistem oleh kegiatan dalam suatu sistem.
5. Komponen (*component*) : Kegiatan-kegiatan atau proses dalam sistem yang mentransformasikan *input* menjadi bentuk setengah jadi *(output).* Komponen ini bisa merupakan subsistem dari sebuah sistem.

6. Penghubung (*Interface*) : Tempat di mana komponen atau sistem dan lingkungannya bertemu atau berinteraksi.
7. Penyimpanan (*storage*): Area yang dikuasai dan digunakan untuk penyimpanan sementara dan tetap dari informasi, energi, bahan baku, dan sebagainya. Penyimpanan merupakan suatu media penyangga di antara komponen tersebut bekerja dengan berbagai tingkatan yang ada dan memungkinkan komponen yang berbeda dari berbagai data yang sama (Aris : 2015).

II.2.2. Informasi

Informasi adalah data yang berguna yang telah diolah sehingga dapat dijadikan dasar untuk mengambil keputusan yang tepat. Informasi sangat penting bagi organisasi. Pada dasarnya informasi adalah penting seperti sumber daya yang lain, misalnya peralatan, bahan, tenaga, dan sebagainya. Informasi yang berkualitas dapat mendukung keunggulan kompetitif suatu organisasi. Dalam sistem informasi akuntansi, kualitas dari informasi yang disediakan merupakan hal penting dalam kesuksesan sistem. Secara konseputal seluruh sistem organisasional mencapai tujuannya melalui proses alokasi sumberdaya, yang diwujudkan melalui proses pengambilan keputusan manajerial. Informasi memiliki nilai ekonomik pada saat ia mendukung keputusan alokasi sumberdaya, sehingga dengan demikian mendukung sistem untuk mencapai tujuan. (Mujilan : 2012 : 1)

II.2.3. Sistem Informasi

Sistem informasi berbasis komputer merupakan sekelompok perangkat keras dan perangkat lunak yang dirancang untuk mengubah data menjadi informasi yang bermanfaat. Jenis sistem informasi berbasis komputer

1. Pengolahan Data (*data processing)* merupakan pemanfaatan teknologi komputer untuk melakukan pengolahan data transaksi-transaksi dalam suatu organisasi. *Electronic Data Processing (*EDP) adalah aplikasi sistem informasi akuntansi paling dasar dalam setiap organisasi. Sehubungan dengan perkembangan teknologi komputer, istilah pengolahan data mulai dikenal dan mempunyai arti yang sama dengan istilah *Electronic Data Processing (*EDP).
2. Sistem Informasi Manajemen (SIM), menguraikan penggunaan teknologi komputer untuk menyediakan informasi bagi pengambilan keputusan para manajer.
3. Sistem Pendukung Keputusan (*Decision Support Systems)* diarahkan untuk melayani permintaan informasi tertentu, khusus, dan tidak rutin dari manajemen.
4. Sistem Pakar (*expert systems*) merupakan sistem informasi berbasis pengetahuan yang memanfaatkan pengetahuannya tentang bidang aplikasi tertentu untuk bertindak seperti seorang konsultan ahli bagi pemakainya.
5. *Executive Information Systems* (EIS) dibuat bagi kebutuhan informasi stratejik manajemen tingkat puncak.
6. Sistem Informasi Akuntansi sistem berbasis komputer yang dirancang untuk mengubah data akuntansi menjadi informasi. (Agustinus Mujilan : 2012)

II.2.4. Akuntansi

Akuntansi berasal dari bahasa Inggris yaitu "*To account*" yang artinya menghitung atau mempertanggungjawabkan sesuatu yang ada kaitannya dengan pengelolaan bidang keuangan dari suatu perusahaan kepada pemiliknya atas kepercayaan yang telah diberikan kepada pengelola tersebut untuk menjalankan kegiatan perusahaan. Pengertian lain akuntansi merupakan kumpulan prosedur berupa kegiatan mencatat, mengikhtisarkan, mengklasifikasikan dan melaporkan keuangan dalam bentuk laporan keuangan dalam periode waktu. Laporan keuangan yang dihasilkan harus dapat di pertanggungjawabkan kepada pihak-pihak yang berkepentingan. Akuntansi adalah proses dari transaksi yang dibuktikan dengan faktur, lalu dari transaksi dibuat jurnal, buku besar, neraca lajur kemudian akan menghasilkan Informasi dalam bentuk laporan keuangan yang digunakan pihak-pihak tertentu. (Wiratna Sujarweni : 2016 : 1)

II.2.5. Sistem Informasi Akuntansi

Sistem informasi akuntansi adalah kumpulan sumberdaya, seperti manusia dan peralatan, yang diatur untuk mengubah data menjadi informasi. Informasi ini dikomunikasikan kepada beragam pengambil keputusan. Sistem Informasi Akuntansi mewujudkan perubahan ini secara manual atau terkomputerisasi. Sistem Informasi Akuntansi juga merupakan sistem yang paling penting di organisasi dan merubah cara menangkap, memproses, menyimpan, dan mendistribusikan informasi. Saat ini, digital dan informasi *online* semakin digunakan dalam sistem informasi akuntansi. Organisasi perlu menempatkan

sistem di lini depan, dan mempertimbangkan baik segi sistem ataupun manusia sebagai factor yang terkait ketika mengatur sistem informasi akuntansi. Sistem Informasi Akuntansi pada umumnya meliputi beberapa siklus pemrosesan transaksi :

1. Siklus pendapatan. Berkaitan dengan pendistribusian barang dan jasa ke entitas lain dan pengumpulan pembayaran-pembayaran yang berkaitan.
2. Siklus pengeluaran. Berkaitan dengan perolehan barang jasa dari entitas lain dan pelunasan kewajiban yang berkaitan.
3. Siklus produksi. Berkaitan dengan pengubahan sumber daya menjadi barang dan jasa.
4. Siklus keuangan. Kejadian-kejadian yang berkaitan dengan perolehan dan manajemen dana-dana modal, termasuk kas. (Mujilan : 2012 : 3).

II.2.6. Laporan Perubahan Modal

Laporan perubahan modal adalah laporan yang berisi seberapa banyak modal awal bertambah ataupun berkurang selama periode tertentu. Perubahan modal itu terjadi dapat karena adanya laba atau rugi usaha, pengambilan pribadi dari pemilik atau *prive*, maupun penambahan modal pemilik. (Wiratna Sujarweni : 2016 : 59).

II.2.6.1. Modal

Modal atau sering disebut ekuitas adalah hak milik atas aktiva perusahaan yang dikurangi dengan semua kewajiban. Modal berasal dari investasi pemilik

yang ditahan di perusahaan. Modal juga dapat diartikan sebagai kewajiban perusahaan membayar hak pemilik bila diperlukan, misalnya ketika ada anggota yang keluar atau perusahaan dilikuidasi (dibubarkan). (Wiratna Sujarweni : 2016 : 30).

II.2.7. 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

1. Bentuk Nornal Pertama (1 NF)

Contoh yang kita gunakan di sini adalah sebuah perancangan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri. Masing-masing pemasok bisa menyediakan banyak barang. Tabel relasionalnya dapat dituliskan sebagai berikut :

PEMASOK (P#, Status, Kota, b#, qty) di mana

p# : kode pemasok (kunci utama)

status : kode status kota

Kota : nama kota

b# : barang yang dipasok

qty : jumlah barang yang dipasok.

Sebuah tabel relasional secara defenisi selalu berada dalam bentuk normal pertama. Semua nilai pada kolom-kolomnya adalah atomi. Ini berarti kolom-kolom tidak mempunyai nilai berulang. Tabel II.1. menunjukkan tabel pemasok dalam 1 NF.

Tabel II.1. Normalisasi Pertama Pemasok
Sumber : (Janner Simarmata, dkk : 2010:80)

P#	Status	Kota	B#	Qty
P1	20	Yogyakarta	B1	300
P1	20	Yogyakarta	B2	200
P1	20	Yogyakarta	B3	400
P1	20	Yogyakarta	B4	200
P1	20	Yogyakarta	B5	100
P1	20	Yogyakarta	B6	100
P2	10	Medan	B1	300
P2	10	Medan	B2	400
P3	10	Medan	B2	200
P4	20	Yogyakarta	B2	200
P4	20	Yogyakarta	B4	300
P4	20	Yogyakarta	B5	400

2. Bentuk Normal Kedua (2 NF).

Defenisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1 NF, tetapi tidak pada 2 NF, sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada 1 NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada kunci utama. Ini berarti bahwa setiap kolom bukan kunci harus tergantung pada seluruh kolom yang membentuk kunci utama. Tabel pemasok berada pada 1 NF, tetapi tidak pada 2 NF karena status dan kota tergantung secara fungsional hanya pada kolom p# dari

kunci gabungan (p#, b#). Ini dapat digambarkan dengan membuat daftar ketergantungan fungsional.

P# ⟶ Kota, Status

Kota ⟶ Status

(P#, B#) ⟶ qty

Proses mengubah tabel 1 NF ke 2 NF adalah :

a. Tentukan sembarang kolom penentu selain kunci gabungan dan kolom-kolom yang ditentukannya.
b. Buat dan beri nama tabel baru untuk masing-masing penentu dan kolom-kolom yang ditentukan.
c. Pindahkan kolom-kolom yang ditentukan dari tabel asal ke tabel baru penentu akan menjadi kunci utama pada tabel baru
d. Hapus kolom yang baru dipindahkan dari tabel asal, kecuali penentu yang akan berfungsi sebagai kunci tamu.
e. Tabel asal bisa diberi nama baru.

Pada contoh, kita memindahkan kolom p#, status, dan kota ke tabel baru yang disebut pemasok2. Kolom p# menjadi kunci utama tabel ini. Tabel II.3. menunjukkan hasilnya.

Tabel II.2. Tabel Bentuk Normal Kedua
Sumber : (Janner Simarmata, dkk : 2010:82)

Pemasok2

P#	Status	Kota
P1	20	Yogyakarta
P2	10	Medan
P3	10	Medan
P4	20	Yogyakarta
P5	30	Bandung

Barang

P#	B#	Qty
P1	B1	300
P1	B2	200
P1	B3	400
P1	B4	200
P1	B5	100
P1	B6	100
P2	B1	300
P2	B2	400
P3	B2	200
P4	B2	200
P4	B4	300
P4	B5	400

3. Bentuk Normal Ketiga (3 NF).

Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional hanya pada kunci utama. Secara defenisi, sebuah tabel berada pada bentuk normal ketiga (3 NF) jika tabel sudah berada pada 2 NF dan setiap kolom yang bukan kunci tidak tergantung secara transistif pada kunci utamanya. Dengan kata lain, semua atribut bukan kunci tergantung secara fungsional hanya pada kunci utama. Tabel barang sudah dalam bentuk normal ketiga. Kolom bukan kunci, qty, tergantung sepenuhnya pada kunci utama (p#, b#). Pemasok masih berada pada 2 NF, tetapi belum berada pada 3 NF karena dia mengandung ketergantungan transitif. Ketergantungan transitif terjadi ketika sebuah kolom bukan kunci, yang ditentukan oleh kunci utama, menentukan kolom lainnya. Konsep ketergantungan transistif daopat digambarkan dengan menunjukkan ketergantungan fungsional pada pemasok2, yaitu :

Pemasok2. p# ⟶ Pemasok2, status

Pemasok2. p# ⟶ Pemasok2, kota

Pemasok2. kota ⟶ Pemasok2, status

Perlu dicatat bahwa pemasok2, status ditentukan, baik oleh kunci utama p#, maupun kolom bukan kunci, kota

Proses mengubah tabel menjadi 3 NF adalah :

a. Tentukan semua penentu selain kunci utama dan kolom yang ditentukannya.
b. Buat dan beri nama tabel baru untuk masing-masing penentu dan kolom yang ditentukannya.
c. Pindahkan kolom yang ditentukan dari tabel asal ke tabel baru. Penentu menjadi kunci utama tabel baru.
d. Hapus kolom yang baru saja dipindahkan dari tabel asal, kecuali penentu yang akan berfungsi sebagai kunci tamu.
e. Tabel asal bisa diberi nama baru.

   Untuk mengubah PEMASOK2 menjadi 3 NF, kita membuat tabel baru yang disebut KOTA_STATUS dan memindahkan kolom kota dan status ke tabel baru. Status dihapus dari tabel diberi nama baru PEMASOK_KOTA. Tabel II.4 menunjukkan hasilnya.

Tabel II.3. Tabel Bentuk Normal Ketiga
Sumber : (Janner Simarmata, dkk : 2010:83)

PEMASOK_KOTA

P#	Kota
P1	Yogyakarta
P2	Medan
P3	Medan
P4	Yogyakarta
P5	Bandung

KOTA_STATUS

Kota	Status
Yogyakarta	20
Medan	10
Bandung	30
Semarang	40

4. Bentuk Normal Boyce Code (BCNF)

Setelah 3 NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3 NF sudah cukup karena sangat jarang entitas yang berada pada 3 NF bukan merupakan 4 NF dan 5 NF. Lebih lanjut, mereka berpendapat bahwa keuntungan yang didapat mengubah entitas ke 4 NF dan 5 NF sangat kecil sehingga tidak perlu dikerjakan. Bentuk Normal Boyce- Code (BCNF) adalah versi 3 NF lebih teliti dan berhubungan dengan tabel relasional yang mempunyai (a) banyak kunci kandidat (b) kunci kandidat gabungan, dan (c) kunci kandidat yang saling tumpang tindih. BCNF didasarkan pada konsep penentu. Sebuah kolom penentu adalah kolom di mana kolom-kolom lain sepenuhnya tergantung secara fungsional. Sebuah tabel relasional berada pada BCNF

5. Bentuk Normal Keempat (4 NF)

Sebuah tabel relasional berada pada bentuk normal keempat (4 NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional. Bentuk normal keempat (4 NF) didasarkan pada konsep

ketergantungan multivalue (MVD). Sebuah ketergantungan multivalue terjadi ketika dalam sebuah tabel relasional yang mengandung setidaknya tiga kolom, satu kolom mempunyai banyak baris bernilai sama, tetapi kolom lain bernilai berebeda. Defenisi secara formal diberikan oleh CJ. Date, yaitu :

Misalnya, ada sebuah tabel relasional R dengan kolom A, B dan C, Maka R.A R.B ⟶ (kolom A menentukan kolom B). Adalah benar jika dan hanya jika himpunan nilai B yang cocok dengan pasangan nilai A dan nilai C pada R hanya tergantung pada nilai A dan tidak tergantung pada nilai C. MVD selalu terjadi dalam pasangan, yaitu R.A ⟶ R.B dipenuhi jika dan hanya jika R.A R.C dipenuhi pula.

6. Bentuk Normal Kelima (5 NF).

Sebuah tabel berada pada bentuk normal kelima jika dia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*). Ketergantungan gabungan berarti sebuah tabel, setelah deskomposisi menjadi tiga atau lebih tabel yang lebih kecil, harus dapat digabungkan kembali untuk membentuk tabel asal. Dengan kata lain 5 NF menunjukkan ketika sebuah tabel tidak dapat dideskomposisi lagi (Janner Simarmata : 2012:86).

II.2.8. Basis Data (*Database*)

Secara sederhana database (basis data atau pangkalan data) dapat diungkapkan sebagai suatu pengorganisasian data dengan bantuan komputer yang memungkinkan data dapat diakses dengan mudah dan cepat (Kadir, 2004). Pengertian akses dapat mencakup pemerolehan data maupun pemanipulasian data seperti menambah serta menghapus data. Dengan memanfaatkan komputer, data dapat disimpan dalam media pengingat yang disebut *hard disk*. Dengan menggunakan media ini, keperluan kertas untuk menyimpan data dapat dikurangi. Selain itu, data menjadi lebih cepat untuk diakses terutama jika dikemas dalam bentuk database.

Pengaplikasian database dapat kita lihat dan rasakan dalam keseharian kita. Database ini menjadi penting untuk mengelola data dari berbagai kegiatan. Misalnya, kita bisa menggunakan mesin ATM (anjungan tunai mandiri atau *automatic teller machine*) bank karena bank telah mempunyai database tentang nasabah dan rekening nasabah. Kemudian data tersebut dapat diakses melalui mesin ATM ketika bertransaksi melalui ATM. Pada saat melakukan transaksi, dalam konteks database sebenarnya kita sudah melakukan perubahan (*update*) data pada database di bank.

Ketika kita menyimpan alamat dan nomor telepon di HP, sebenarnya juga telah menggunakan konsep database. Data yang kita simpan di HP juga mempunyai struktur yang diisi melalui formulir (*form*) yang disediakan. Pengguna dimungkinkan menambahkan nomor HP, nama pemegang, bahkan kemudian dapat ditambah dengan alamat *email*, alamat *web*, nama kantor, dan sebagainya.

Pemahaman tentang *database* ini dapat didekatkan pada konsep akuntansi. Kita bisa umpamakan bahwa ketika kita melakukan proses akuntansi secara manual, kita menuliskan suatu catatan ke dalam lajur dan kolom buku. Mulai dari jurnal, buku besar, buku pembantu kita memasukkan catatan satu demi satu. Melihat buku akuntansi tersebut, sebenarnya kita sudah melihat konsep *database*, yang jika dikelola dengan komputer masih diperlukan penyesuaian dalam membentuk kolom-kolomnya. (Mujilan : 2012:23)

II.2.8.1. *Model Database*

Model database yang saat ini banyak digunakan adalah *model database relational*. Imam (2008) menyebutkan "*Model database* ini disusun dalam bentuk tabel dua dimensi yang terdiri dari baris (*record*) dan (*field*), pertemuan antara baris dengan kolom disebut item data (*data value*). Tabel-tabel yang ada dihubungkan (*relationship*) sedemikian rupa menggunakan *field-field* kunci (*key field*) sehingga dapat meminimalkan duplikasi data."

Model database relational ini dapat kita kenal konsepnya mulai dari yang paling sederhana misalnya dengan penerapan program aplikasi *excel*. Meskipun untuk pengelolaan *database* secara luas *excel* jarang digunakan dan kurang mencukupi, namun untuk melihat konsep database dan konsep membangunnya program ini dapat dimanfaatkan. *Excel* mempunyai baris yang disebut *raw* dan mempunyai kolom. Kemudian item data merupakan sel atau pertemuan antara baris dan kolom. Tabel-tabel dapat diumpakan apabila kita menggunakan tabel dalam suatu *sheet* tertentu. Data dari berbagai tabel dapat diambil dari tabel lain

menggunakan perintah *look up* yang berdasarkan kode kunci tertentu. Kode kunci tersebut berada pada suatu kolom tertentu, yang dalam konsep *database relational* disebut sebagai *key field* tadi. (Mujilan : 2012:24)

II.2.8.2. Struktur *Database*

Untuk memahami konteks *database* kita perlu memahami istilah dan hal-hal yang terkait dengan database. Dalam berbagai program aplikasi *database* terdapat kesamaan ataupun sedikit perbedaan di dalamnya. Seseorang yang mempelajari *database* dengan program aplikasi tertentu harus memperhatikan struktur dan karakteristik sesuai dengan bahasa dalam aplikasi tersebut. Namun demikian, secara umum terdapat karakteristik sebagai berikut:

1. Nama *file*

Nama *file* adalah nama yang digunakan untuk mengidentifikasi adanya data yang disimpan dalam komputer dan digunakan untuk pemanggilan data. *File* yang dikelola akan muncul dalam komputer dengan ekstensi sesuai dengan program aplikasinya. *File* tersebut dapat digunakan untuk menandakan adanya *file database*, ataupun *file table*. *Database* dan *table* akan saling terkait, meskipun cara menyimpan dalam komputer akan mengalami sedikit perbedaan pada beberapa aplikasi. Misalnya Ms. *Access* akan menyimpan dengan *file* yang dapat kita lihat adalah *file* databasenya. Di program *MySql* nama *database* ini akan menjadi *folder*. Sementara di *FoxPro* nama *database* dapat menjadi *file* tersendiri. *Table* cara menyimpannya juga berbeda, dalam *Ms. Access* mungkin kita tidak melihat nama *table* secara kasat mata karena akan dikelola di dalam *file* database.

Di dalam *MySql* kita bisa melihat beberapa nama *file* terkait dengan pengelolaan *table*. Dan di dalam *FoxPro table* ini dapat menjadi nama *file* terpisah dan dapat dikenali pula sebagai *free table.*

2. *Database*

Database sebenarnya merupakan nama untuk menampung berbagai *table* di dalamnya. Konsep ini akan sama dalam berbagai program aplikasi. Misalnya kita membangun *database* akuntansi dengan nama database "*akun_base*". Di dalam *akun_base* akan diorganisasi berbagai *table* yang terkait dengan kegiatan akuntansi misalnya tabel: rekening, pelanggan, jurnal, buku induk, dan administrator program, dan sebagainya. Setiap data yang masuk tidaklah dicatat dalam *database*, namun di dalam masing-masing *table* yang sesuai.

3. *Table*

Table merupakan tempat untuk menyimpan data sesuai dengan kelompok data. Setiap isi *table* mengandung data yang mempunyai karakteristik dalam penggunaannya. Untuk mempermudah pengolahan biasanya pembangun *database* mengkategorikan *table* sesuai dengan data isinya sebagai berikut :

a. *Master table*

Master table berisi data tentang hal-hal utama dalam kegiatan *database*. *Table* ini berisi *record* yang relatif permanen atau seringkali menjadi acuan ketika mengoperasikan transaksi. Dalam master tabel identitas *record* menjadi penting dan diusahakan merupakan data atau kode yang bersifat unik. Unik dapat diartikan bahwa tidak ada dalam satu *table* berisi kode yang sama. Disain kode menjadi penting di sini. Misalnya dalam mendisain nama

akun dalam *database* akuntansi, maka kode akun menjadi sangat penting artinya. Dalam *table* berisi nama barang, maka kode barang menjadi hal penting. Contoh lain dalam *database* akademik, tabel *master* dapat berupa : mahasiswa, daftar dosen, daftar kurikulum.

b. *Transaction table*

Tabel transaksi digunakan untuk menyimpan data dalam menjalankan suatu kegiatan atau bisnis. Data ini seringkali akan bertambah dalam kesehariannya ketika terjadi transaksi yang sesuai dengannya. Secara lebih mudah dapat dipahami dalam akuntansi seringkali mencatat transaksi dalam jurnal. Terkait hal tersebut, transaksi ini dicatat dalam tabel jurnal. Dalam mencatat transaksi ini, kita harus menyesuaikan kode data tertentu dengan kode yang terdapat dalam *master table.*

c. *Tabulation table*

Tabulasi data dapat digunakan untuk menyimpan data seperti halnya *master* data namun bersifat sebagai data pembantu ketika menginput formulir baik untuk data master maupun transaksi. Misalnya untuk memetakan keterangan hobi, jenis kelamin, nama golongan, nama level manajemen, dan sebagainya. Dengan konsep penamaan *field* yang baik mungkin saja *table* tabulasi ini dapat digunakan untuk memuat berbagai kelompok data. Misalnya fieldnya berupa kode dan keterangan. Contoh kelompok *gender* dengan L = laki-laki; P = perempuan. Kelompok *level* dengan M = Manajer, O = operator, S = *seller*

d. *Temporary table*

Temporary adalah data sementara yang digunakan untuk membantu ketika terjadi proses transaksi. Data ini dapat saja langsung dihapus ketika transaksi selesai terproses. Misalnya digunakan untuk mempermudah perhitungan, penyimpanan data sementara sebelum diproses setuju ke *database.* Misalnya: ketika terjadi transaksi di depan kasir, data-data pertama akan ditangkap dan dimasukan dalam *file temporary* sebelum akhirnya kasir melakukan perintah “ok” yang menandakan data transaksi siap untuk disimpan atau diproses dalam komputer. Ketika masa tunggu ini, data masih dapat diedit, dibatalkan, ataupun ditambah. Sementara ketika sudah masuk ke sistem, edit atau penambahan akan membutuhkan prosedur tertentu. Seandainya dianalogikan dengan sistem akuntansi maka proses edit data yang telah masuk ke sistem dapat digunakan prosedur seperti halnya melakukan jurnal koreksi.

4. *Field*

Field adalah penanda untuk kolom data. Jika dalam *excel* penanda tersebut adalah kolom A, B, dan seterusnya, sementara dalam konsep *table* dalam *database* maka nama *field* memegang peranan penting. Dalam konsep *table* dalam *database*, ketika memanggil dengan nama *field* tertentu maka data-data di dalamnya akan muncul. Pengolahan dapat dilakukan dengan membuat *filter*, misalnya berdasarkan kode tertentu, berdasarkan *record* tertentu. Misalnya kita ingin memanggil *record* terkait nama karyawan Andi. Dalam *database* Andi ini diberi ID: 11001. Sehingga kita bisa menggunakan konsep filtrasi untuk

memanggil personalia dengan kode ID 11001. Apabila data ketemu, maka kita dapat menggunakan data berdasarkan *field-field* yang ada, misalnya nama, tempat lahir, tanggal lahir, alamat, dan sebagainya yang mengacu pada ID 11001.

Dalam mengatur setting *field*, biasanya akan terkait hal-hal sebagai berikut :

a. *Field type* : tipe *field* ini dapat terkait apakah *field* tersebut akan berisi data berupa *key field* (*primary key*, *secondary key*), atau *descriptor*. *Primary key* akan berisi ID atau kode pokok yang akan digunakan dalam mengidentifikasi *record*, sehingga data di dalam *field* tersebut tidak diijinkan untuk memiliki lebih dari satu data yang sama. *Secondary* adalah subset dari *key* utama. Misalnya saja kode mata kuliah dalam satu semester tidak boleh terdapat lebih dari satu pada ID atau NIM yang sama. *Descriptor* adalah *field* berisi data yang akan merupakan satu kesatuan dengan yang lainnya sebagai penjelasan akan adanya *record* atau ID tertentu.

b. Data *type*: tipe data merupakan jenis data yang dapat dimasukkan dalam *field*. Hal ini dapat dibagi secara umum sebagai karakter/ *text*, numerik, tanggal dan sebagainya.

c. *Field Size*. Penting untuk memahami ukuran field yang akan digunakan dalam menampung data. Dalam pengembangan sistem harus dapat memperkirakan berapa lebar ukuran *field* yang efektif. Apabila terlalu lebar akan terjadi banyak spasi kosong dan berpengaruh pada ukuran *file* yang disimpan. Sementara apabila terlalu sempit akan terdapat data yang tidak tersimpan. Misalnya ketika kita menghitung bahwa nama menggunakan ukuran 40 karakter sudah memenuhi untuk *field* data kita. Konsekuensinya,

apabila terdapat nama di atas 40 karakter maka akan terpotong menjadi 40 karakter. Konsekuensi lain adalah nama tersebut diinput hingga memuat maksimal 40 karakter yaitu dengan mengadakan singkatan nama.

5. *Records*

Records merupakan baris data. Karena satu baris data biasanya mengindikasikan satu kesatuan data tertentu, maka satu *record* ada yang menyebut satu data. Misalnya keterangan mengenai biodata Andi disimpan dalam satu *record* beridentitas ID 11001, maka untuk menyebutkan satu kesatuan data seputar Andi dalam baris tertentu ada yang menyebutkan sebagai *record* data Andi. Dalam konsep *database* masing-masing *record* memiliki nomor identitas tersendiri baik itu identitas yang diberikan komputer ataupun yang diinputkan secara manual. Sehingga dalam konteks tertentu dapat digunakan konsep nomor *record*, ID otomatis, ID *primary key*. (Mujilan : 2012:31)

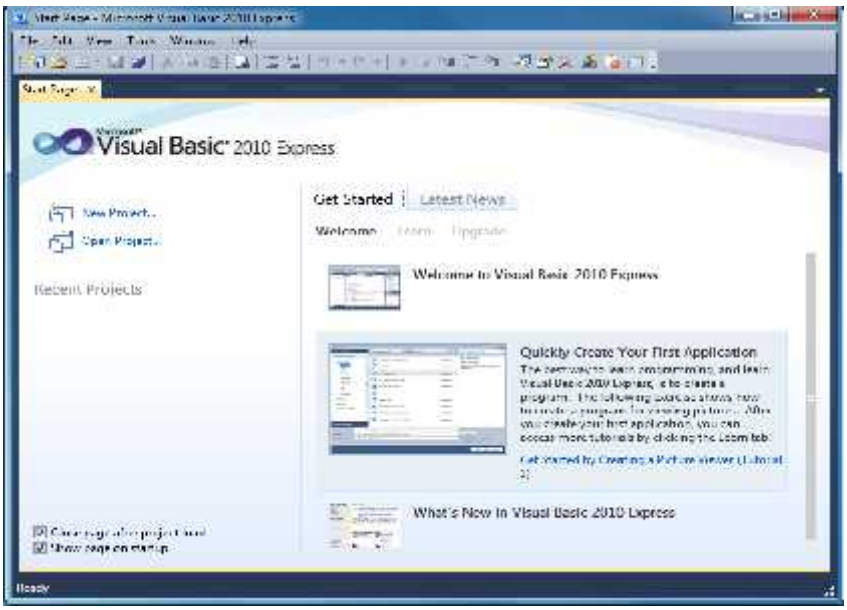
II.2.9. *Visual Studio Net. 2010*

Visual Basic 2010 adalah inkarnasi dari bahasa *visual basic* yang sangat populer dan telah dilengkapi dengan fitur serta fungsi yang setara dengan bahasa tingkat lainnya sseperti C++. Anda dapat menggunakan *visual basic* 2010 untuk membuat aplikasi *windows, mobile, web,* dan *office* atau kode yang telah ditulis oleh orang lain dan kemudian dimasukkan ke program lainnya. *Visual basic* menyediakan berbagai tools dan fitur canggih yang memungkinkan dapat menulis kode, menguji dan menjalankan program tunggal ataau terkadang serangkaian program yang terkait dengan satu aplikasi. (Christopher Lee:2014:1)

Tabel II.4. Jendela Aplikasi Visual Studio 2010
(Sumber : Christopher Lee ; 2014 : 2)

No	Bagian	Keterangan
1	Title Bar	Menampilkan nama aplikasi yang sedang terbuka.
2	Mneu Bar	Menampilkan daftar perintah yang memungkinkan anda dapat menulis, mengedit, menyimpan, mencetak, menguji dan menjalankan program *visual basic*.
3	Standard Toollbar	Berisi Tombol yang menjalankan perintah yang sering digunakan sepertin *open project, new project, save, cut, copy, paste,* dan *undo*
4	Toolbox	Berisi komponen NET yang dapat anda gunakan untuk mengembangkan antarmuka pengguna grafis untuk program visual studio 2010.
5	Area Kerja Utama	Menampilkan item yang sedang di kerjakan
6	Solution Explorer	Menampilkan elemen dari *visual bassic solution*, yaitu nama yang ddiberikan kepada program *visual basic* dan item lainnya yang dihasilkan oleh *visual basic* 2010 sehingga program akan mengeksekusi dengan benar.
7	Properties Window	Setiap Objek dalm program visual basic memiliki seperangkat karakteristik yang disebut sifat-sifat objek.

Untuk melihat tampilan *visual basic* 2010 dapat dilihat pada gambar II.1. sebagai berikut :



Gambar II.1. Tampilan Utama Visual Basic Net. 2010
(Sumber : Christopher Lee ; 2014 ; 2)

II.2.10. SQL Server 2008

SQL (*Structured Query Language*) adalah bahasa *non procedural* untuk mengakses data pada *database relasional*. SQL (*Structured Query Language*) adalah bahasa *database* yang dipergunakan dalam menyelesaikan permasalahan dalam *database* serta mempunyai kelebihan dalam mengolah data. Standar SQL (*Structured Query Language*) mula-mula didefenisikan oleh ISO *(International Standards Organization)* dan ANSI *(the American National Standards Institute)* yang dikenal dengan sebutan SQL86. (Eka Iswandy : 2015 : 73)

Dengan menggunakan SQL (*Structured Query Language*), kita dapat melakukan hal-hal berikut:

1. Memodifikasi struktur *database*.
2. Mengubah, mengisi, menghapus isi *database*.
3. Mentransfer data antara *database* yang berbeda. SQL (*Structured Query Language*) ada yang dikembangkan untuk PC dan ada juga yang dikembangkan untuk dapat mengakomodasi database yang sangat besar.

Beberapa contohnya antara lain:

1. *Microsoft Accsess*

Digunakan untuk PC, sangat mudah dipakai dimana perintah SQL (*Structured Query Language*) dapat langsung dimasukkan atau melalui fasilitas yang telah digunakan.

2. *Microsoft Query*

SQL (*Structured Query Language*) yang dipaket dengan produk lain dari *Microsoft Windows*, yaitu *Microsoft Visual Studio* seperti *Visual Basic* dan *Visual* C++. Untuk terhubung dengan *database* lain menggunakan ODBC.

3. *Oracle*

Digunakan untuk perusahaan yang mengggunakan *database* besar.

II.2.11. *UML* (*Unified Modelling Language*)

Unified Modelling Language (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak sebuah sistem. UML (*Unified Modelling Language*) lebih mengedepankan penggunaan diagram untuk menggambarkan aspek dari sistem, karena tergolong bahasa *visual* yang lebih mudah dan lebih cepat dipahami dibandingkan dengan bahasa pemrograman. *Unified Modelling Language* (UML) biasa digunakan untuk :

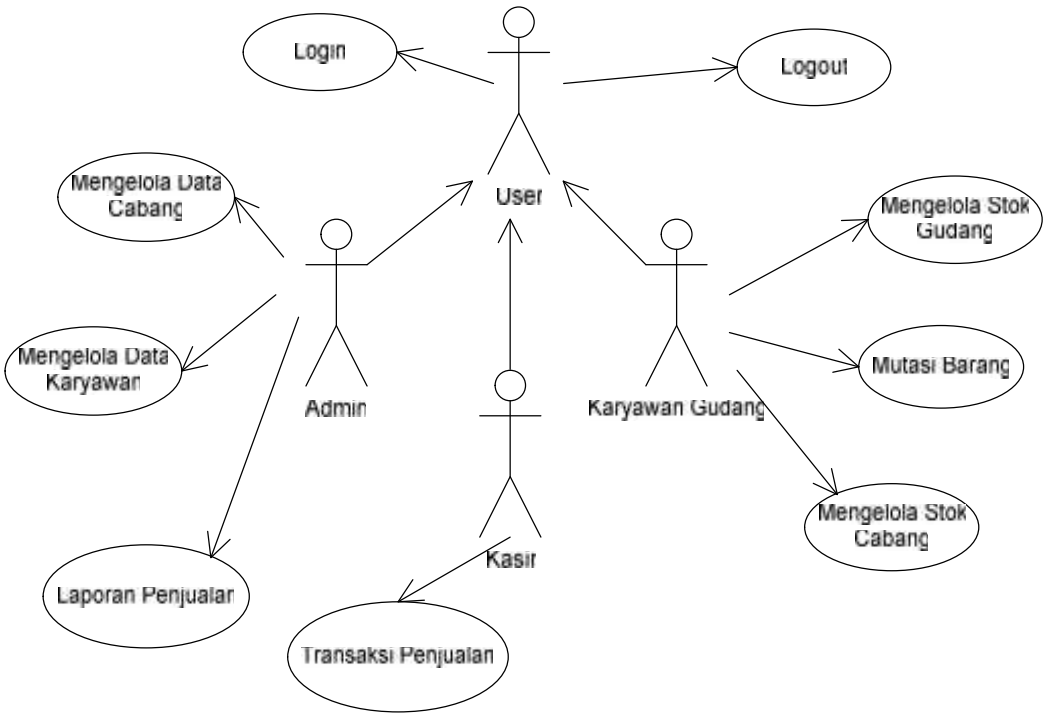
1. Menggambarkan batasan sistem dan fungsi-fungsi sistem secara umum, dibuat dengan *use case* dan *actor.*
2. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan *interaction diagrams*.
3. Menggambarkan representasi struktur *static* sebuah sistem dalam bentuk *class diagram*.
4. Membuat model *behavior* yang menggambarkan kebiasaan atau sifat sebuah sistem dengan *state transition diagrams* UML (*Unified Modelling Language*).

5. Menyatakan arsitektur implementasi fisik menggunakan *component and development diagrams*.
6. Menyampaikan atau memperluas *fungsionalty* dengan *stereo types*.

Pemodelan penggunaan UML merupakan metode pemodelan berorientasi objek dan berbasis *visual.* Karenanya pemodelan objek yang fokus pada pendefinisian struktur statis dan model sistem informasi yang dinamis daripada mendefinisikan data dan model proses yang tujuannya adalah pengembangan tradisional. UML (*Unified Modelling Language*) menawarkan diagram yang dikelompokkan manjadi lima perspektif berbeda untuk memodelkan suatu sistem. Seperti satu set *blue print* yang digunakan untuk membangun sebuah rumah (Saipul Anwar, et al., 2016 : 75-76).

II.2.11.1. *Use Case* Diagram

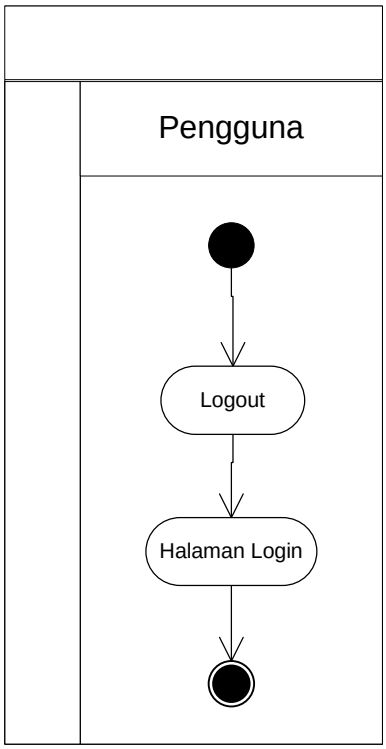
Use case diagram merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut (Ade Hendini, 2016 : 108).



Gambar II.2. *Use Case* Diagram
(Sumber : Ade Hendini, 2016 : 112)

II.2.11.2. *Activity* Diagram

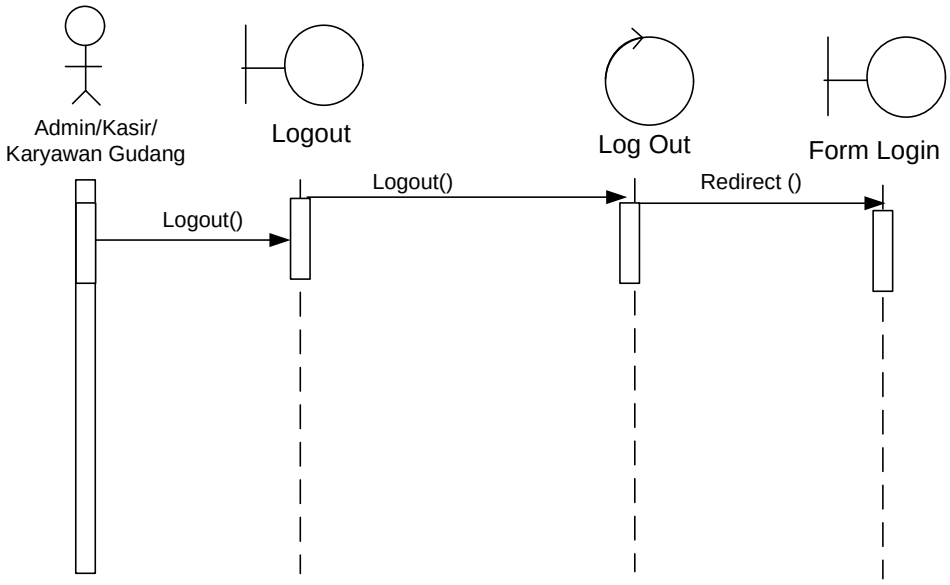
Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis (Ade Hendini, 2016 : 109).



Gambar II.3. *Activity* Diagram
(Sumber : Ade Hendini, 2016 : 112)

II.2.11.3. *Sequence* Diagram

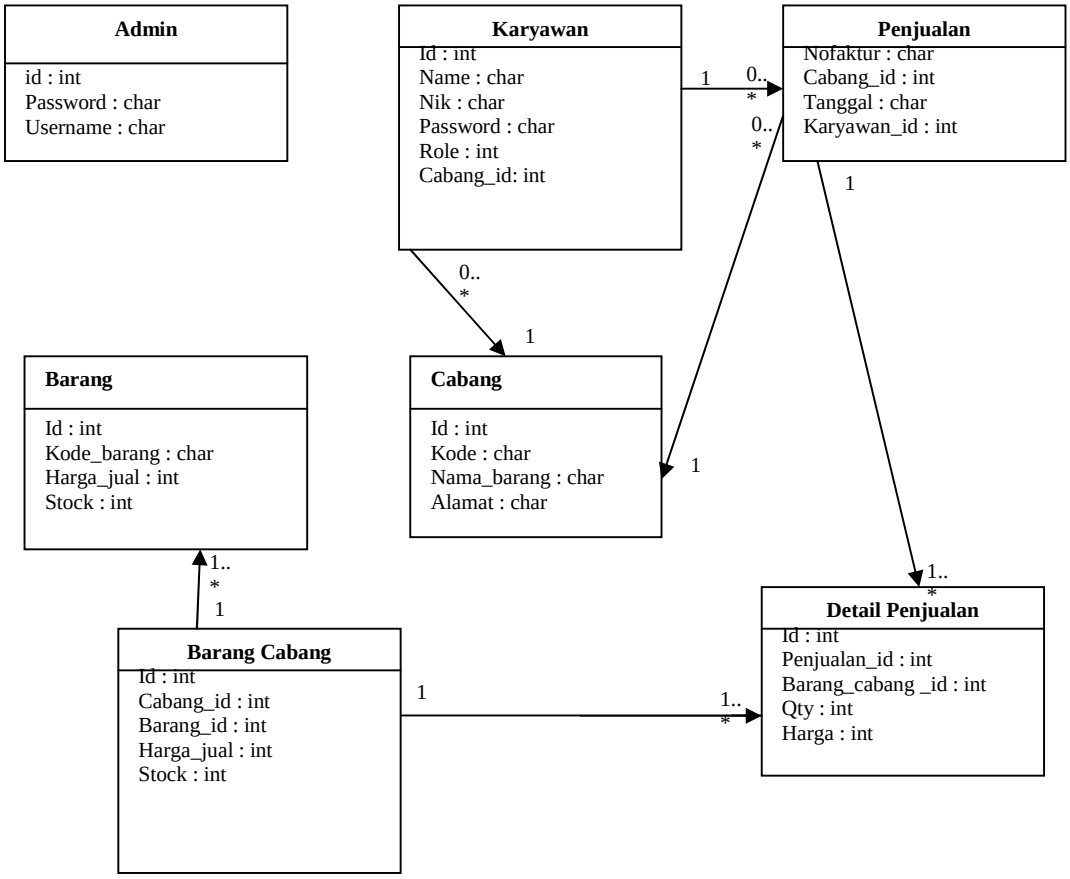
Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek (Ade Hendini, 2016 : 110).



Gambar II.4. *Sequence* Diagram
(Sumber : Ade Hendini, 2016 : 114)

II.2.11.4. *Class* Diagram

Merupakan hubungan antar kelas dan penjelasan *detail* tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class* diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class* diagram secara khas meliputi : Kelas (*Class*), Relasi *Assosiations*, *Generalitation* dan *Aggregation*, attribut (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality* (Ade Hendini, 2016 : 111).



Gambar II.5. *Class* Diagram
(Sumber : Ade Hendini, 2016 : 115)