

Digunakan untuk kalangan internal

MODUL PRAKTIKUM

PANDUAN PRAKTIS MENGUASAI VISIO, POWER DESIGNER DAN STAR UML SEBAGAI ALAT BANTU UNTUK ANALISIS DAN PERANCANGAN

**DIGUNAKAN UNTUK PRAKTIKUM
MATAKULIAH PRAKTIKUM ANALISIS DAN PERANCANGAN SISTEM INFORMASI**



**Disusun Oleh:
Z A I D I R**

**PROGRAM STUDI MANAJEMEN INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS RESPATI YOGYAKARTA
2014**

KETENTUAN KEGIATAN PRAKTIKUM

A. Kelengkapan Praktikum

1. Kegiatan praktikum dilengkapi dengan modul praktikum
2. Perlu memiliki flashdisk
3. Hasil kegiatan praktikum disimpan pada folder masing-masing (bisa dibuat pada drive komputer atau flashdisk atau dibuat di server)

B. Tata Tertib Praktikum

1. Peserta praktikum (praktikan) harus memenuhi atau membawa kelengkapan praktikum setiap kali mengikuti kegiatan praktikum.
2. Untuk mengikuti praktikum harus berpakaian rapi dan sopan.
3. Pada saat kegiatan praktikum, peserta praktikum harus menjaga ketenangan, ketertiban, kebersihan dan kerapian.
4. Peserta praktikum harus datang tepat waktu, dengan waktu toleransi keterlambatan 15 menit.
5. Alat komunikasi dinyalakan dalam mode silent atau dimatikan.
6. Selama praktikum, peserta praktikum tidak diperkenankan meninggalkan ruangan praktikum/lab tanpa seizin pengampu/dosen praktikum.
7. Peserta praktikum harus bertutur kata yang baik dan sopan kepada pengampu/dosen praktikum.
8. Peserta praktikum harus dapat menunjukkan sikap kejujuran. Apabila menemukan perlengkapan praktikum peserta praktikum lain yang tertinggal di komputer atau meja yang digunakan, maka dapat memberitahukan kepada petugas lab untuk selanjutnya akan dibuat pengumuman.
9. Jadwal kegiatan praktikum yang karena sesuatu sebab tidak bisa dilaksanakan, maka akan dicari jadwal pengganti dihari lain berdasarkan kesepakatan pengampu/dosen praktikum dengan peserta praktikum.
10. Peserta praktikum harus memenuhi minimal kehadiran yaitu 75% dari total pertemuan.
11. Pada saat dilaksanakan ujian responsi, peserta praktikum diwajibkan menggunakan jas almamater.
12. Hal-hal yang belum diatur pada tata tertib ini akan diatur kemudian.

C. Sangsi

1. Pengampu/dosen praktikum berhak memperingatkan bahkan mengeluarkan peserta praktikum yang tidak dapat menjaga ketenangan, ketertiban, kebersihan dan kerapian.
2. Apabila peserta praktikum datang lebih dari 15 menit, maka tidak diperkenankan mengikuti praktikum.
3. Peserta praktikum yang tidak memenuhi minimal kehadiran 75% dari total pertemuan, maka tidak diperkenankan mengikuti responsi/UAS. Dengan demikian maka nilai akhiri untuk mata kuliah praktikum adalah E.

D. Penilaian

Kegiatan praktikum akan dinilai dengan beberapa komponen sebagai berikut:

No	Komponen Penilaian	Bobot Penilaian
1	Praktikum Harian	30%
2	Tugas	30%
3	Responsi/UAS	40%

Pengantar Tools Untuk Perancangan

Tujuan Instruksional Khusus:

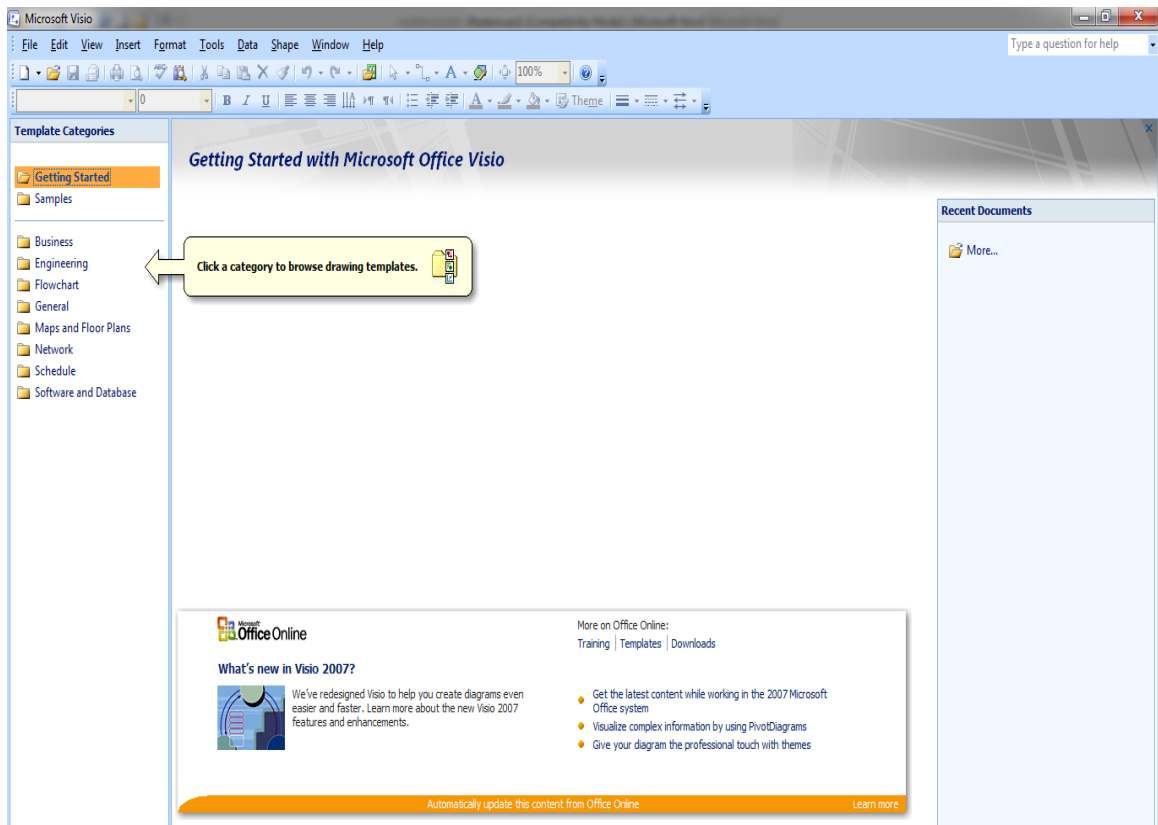
Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak yang sering dipakai untuk perancangan sistem informasi.

Pertemuan ini akan menjelaskan secara singkat tentang tiga perangkat lunak yang akan dijadikan tools dalam perancangan sistem informasi seperti Visio, Power Designer, dan Star UML.

1.1 Mengenal Perangkat Lunak

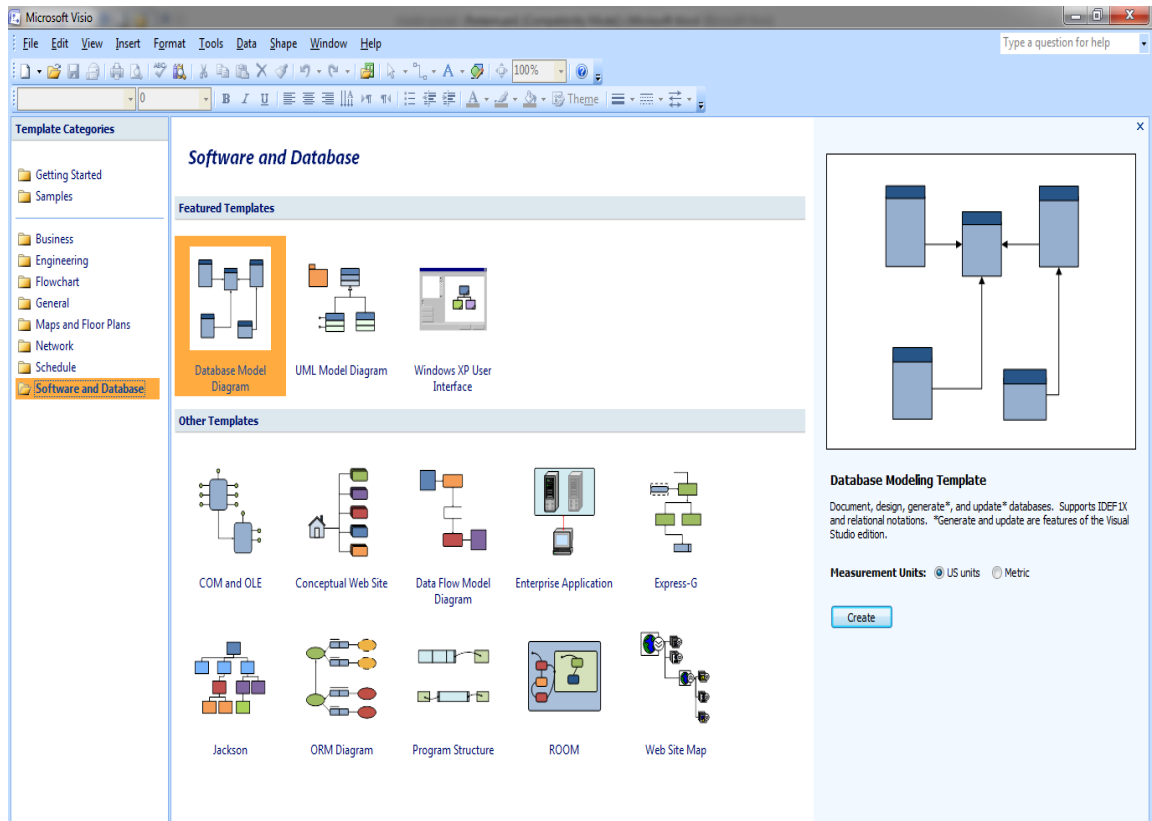
1. Visio

Perangkat lunak Visio bisa juga dijadikan alat untuk perancangan system informasi karena pada aplikasi ini telah disediakan simbol-simbol yang dapat membantu dalam perancangan sistem informasi. Pada saat memanggil aplikasi visio, akan ditampilkan menu seperti gambar 1.1.



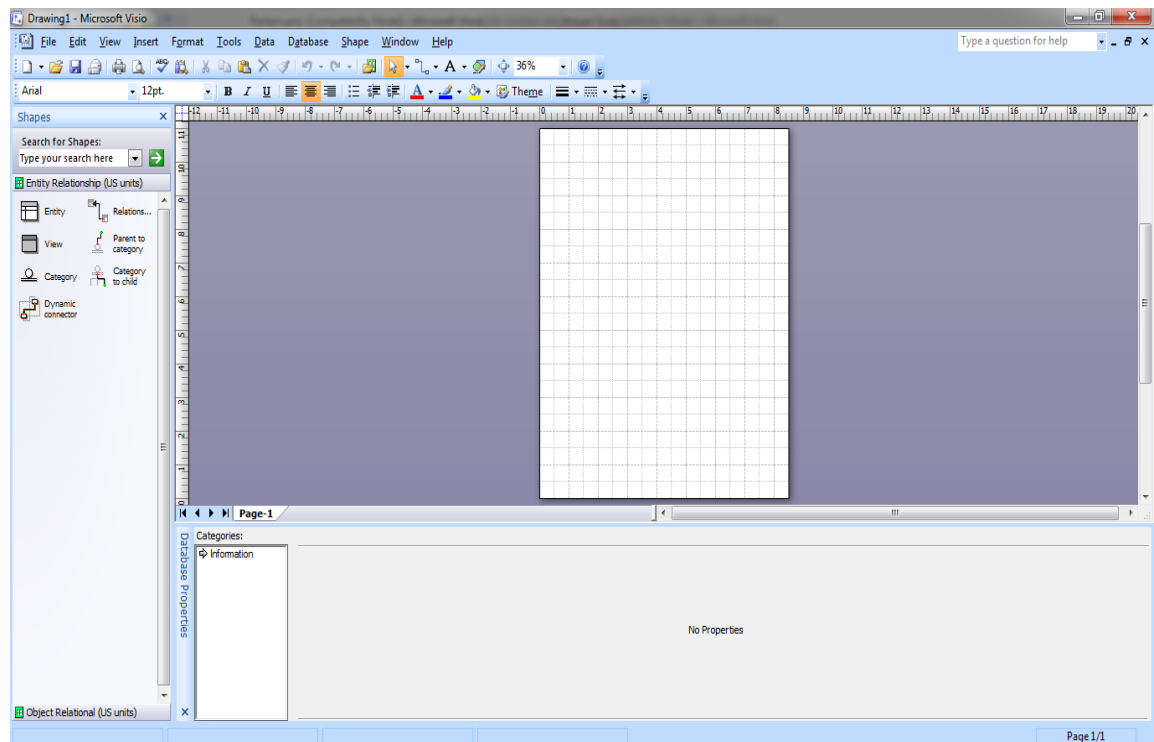
Gambar 1.1 Tampilan Pertama Visio

Untuk membangun rancangan sistem informasi salah satu fasilitas yang digunakan adalah pada pilihan **Software and Database**. Pada saat memilih fasilitas tersebut akan disajikan beberapa pilihan seperti terlihat pada Gambar 1.2.



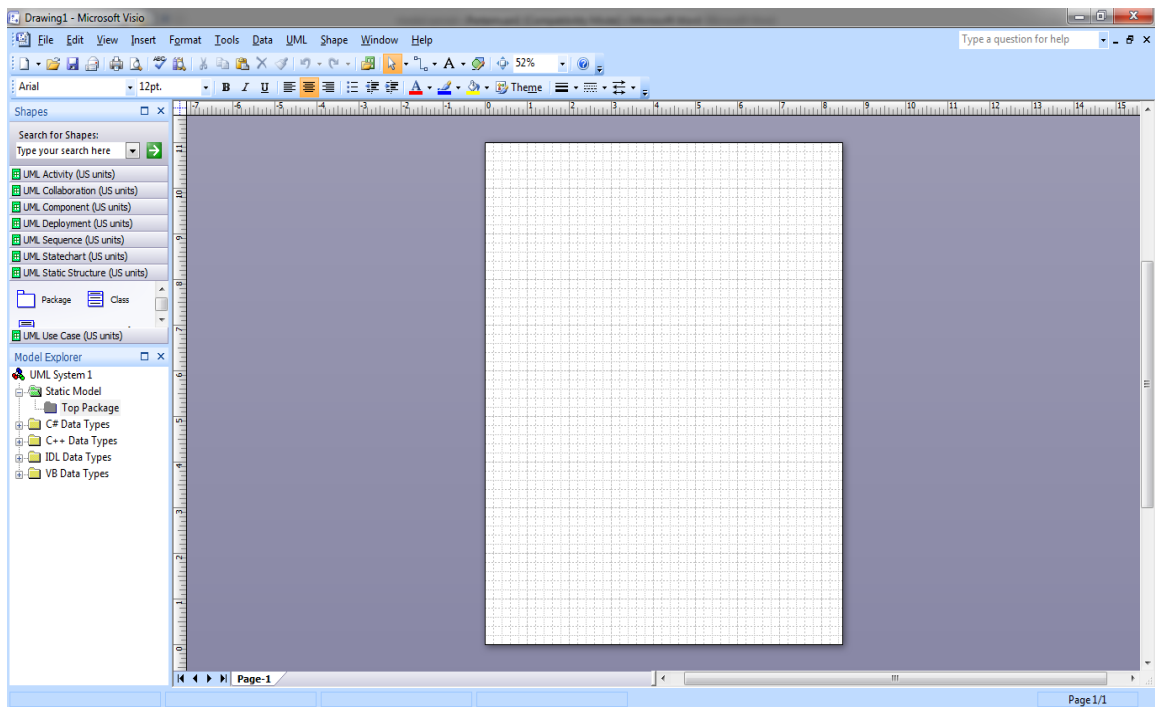
Gambar 1.2 Tampilan Pilihan Software and Database

Apabila yang dibutuhkan atau diklik pilihan Database Model Diagram dan dilanjutkan dengan mengklik tombol Create, maka akan disajikan lembar kerja seperti terlihat pada Gambar 1.3.



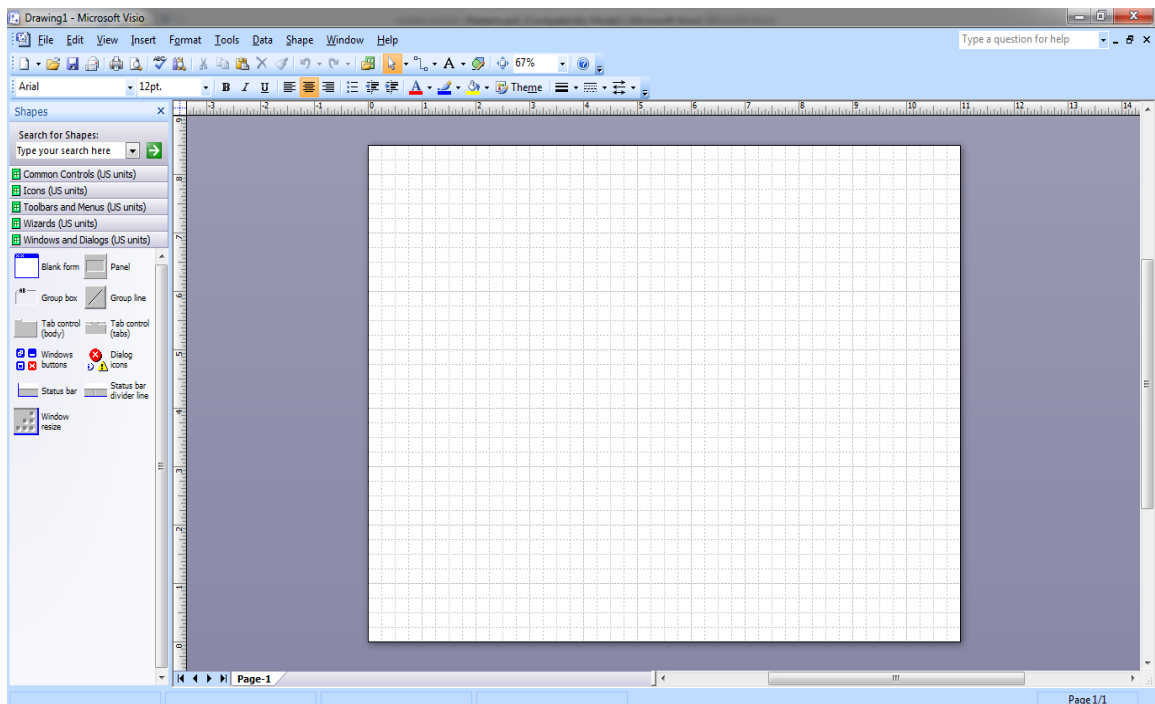
Gambar 1.3 Tampilan Pilihan Database Model Diagram

Apabila yang dibutuhkan atau diklik pilihan UML Model Diagram dan dilanjutkan dengan mengklik tombol Create, maka akan disajikan lembar kerja seperti terlihat pada Gambar 1.4.



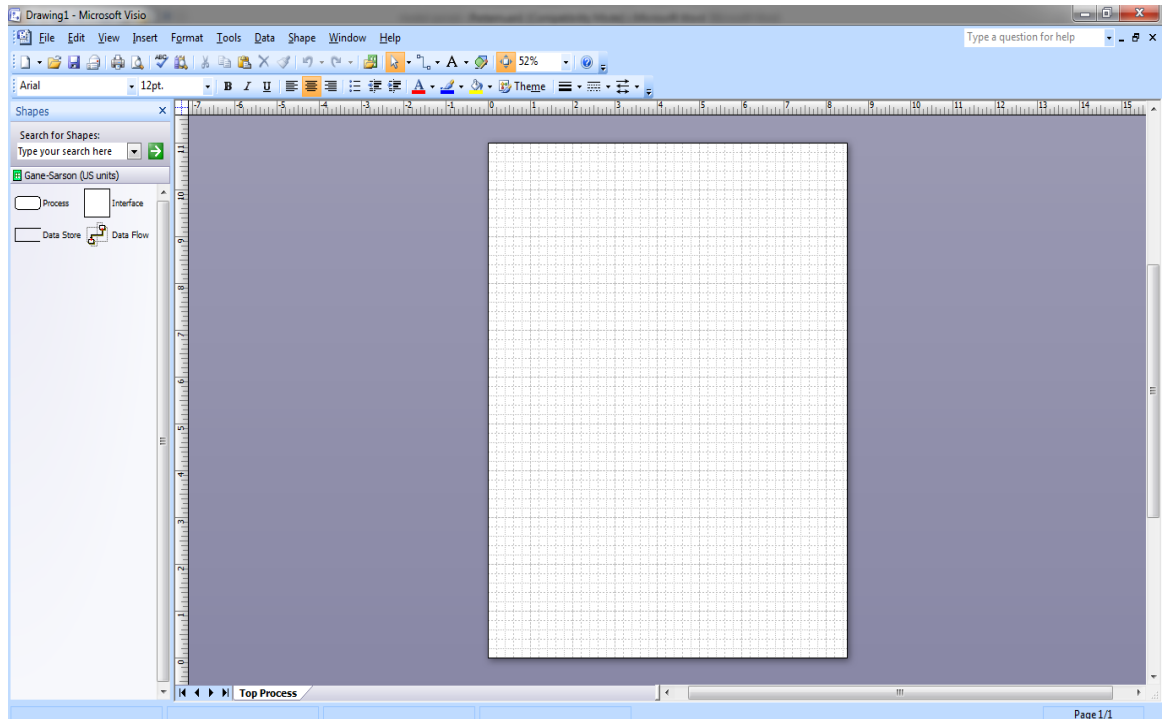
Gambar 1.4 Tampilan Pilihan UML Model Diagram

Apabila yang dibutuhkan atau diklik pilihan Windows XP User Interface dan dilanjutkan dengan mengklik tombol Create, maka akan disajikan lembar kerja seperti terlihat pada Gambar 1.5.



Gambar 1.5 Tampilan Pilihan Windows XP User Interface

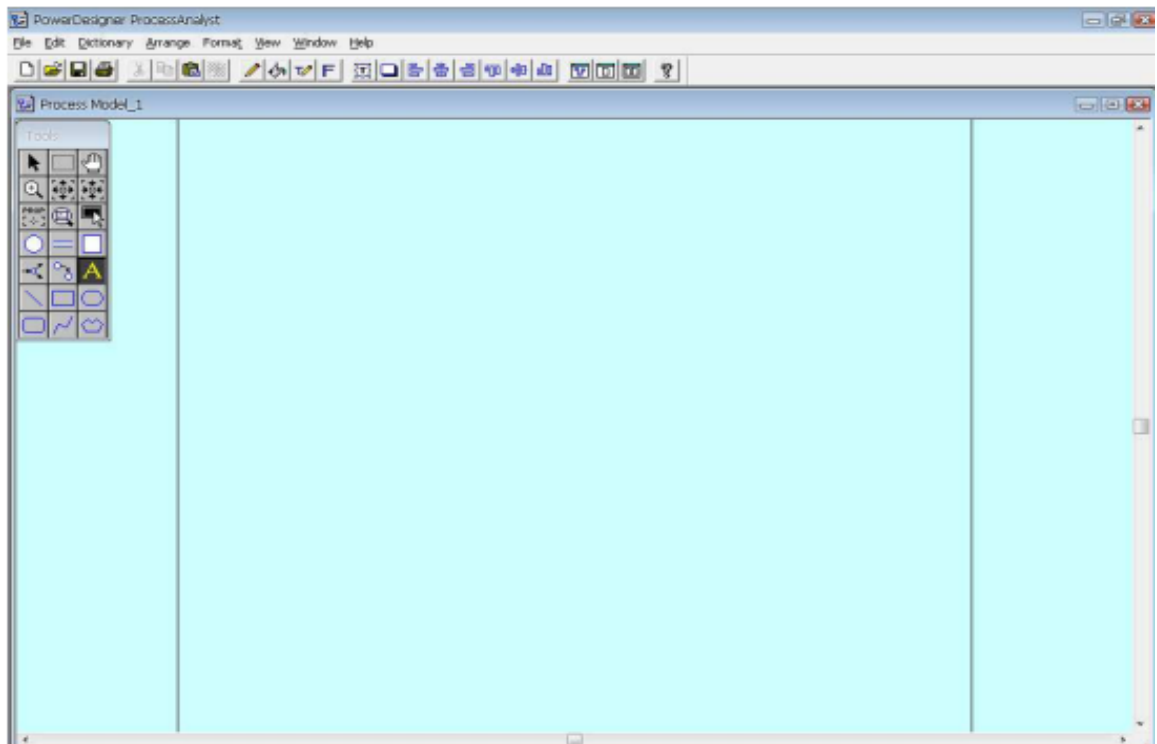
Apabila yang dibutuhkan atau diklik pilihan Data Flow Model Diagram dan dilanjutkan dengan mengklik tombol Create, maka akan disajikan lembar kerja seperti terlihat pada Gambar 1.6.



Gambar 1.6 Tampilan Pilihan Data Flow Model Diagram

2. Power Designer

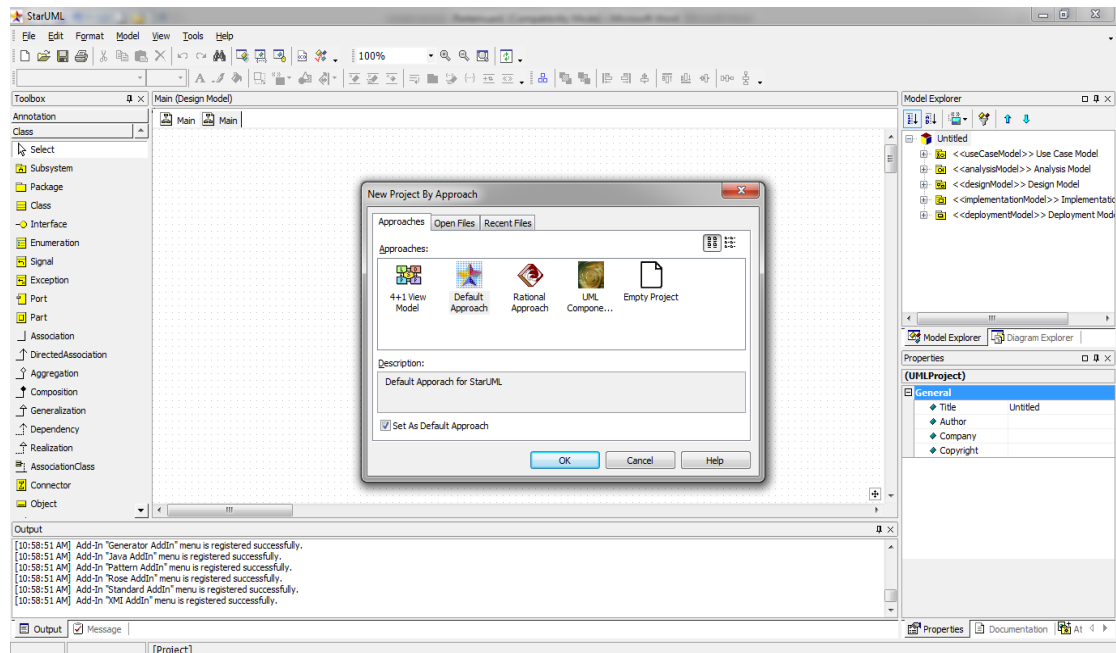
Pada saat memanggil aplikasi Power Designer dan memilih Analyst Process, akan ditampilkan menu seperti gambar 1.7.



Gambar 1.7 Tampilan Awal Power Designer

3. Star UML

Pada saat memanggil aplikasi Star UML, akan ditampilkan menu seperti gambar 1.8.



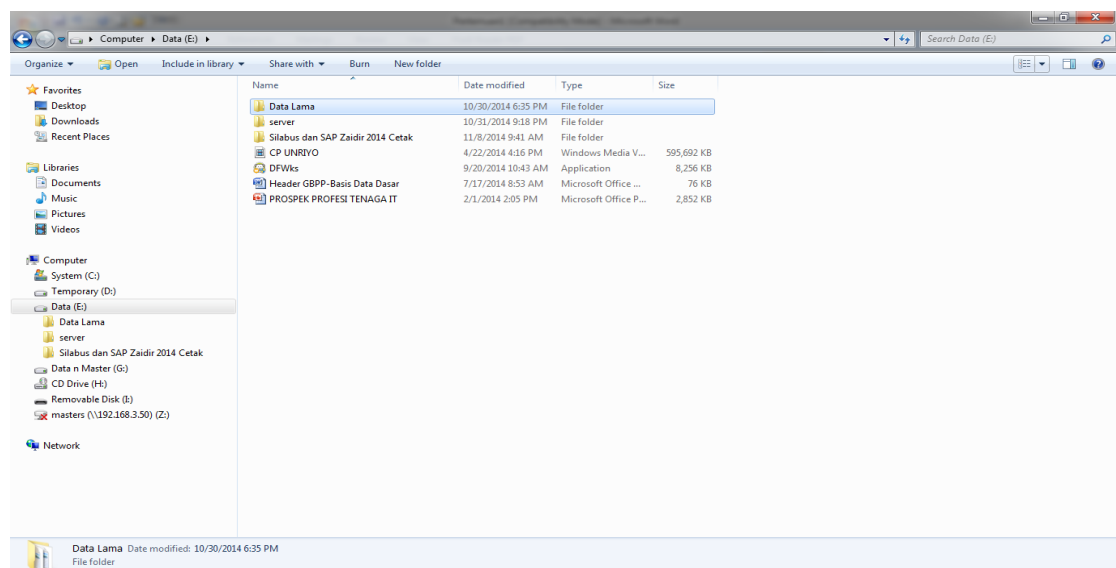
Gambar 1.8 Tampilan Awal Star UML

1.2 Praktek

1.2.1 Menyiapkan Folder Kerja

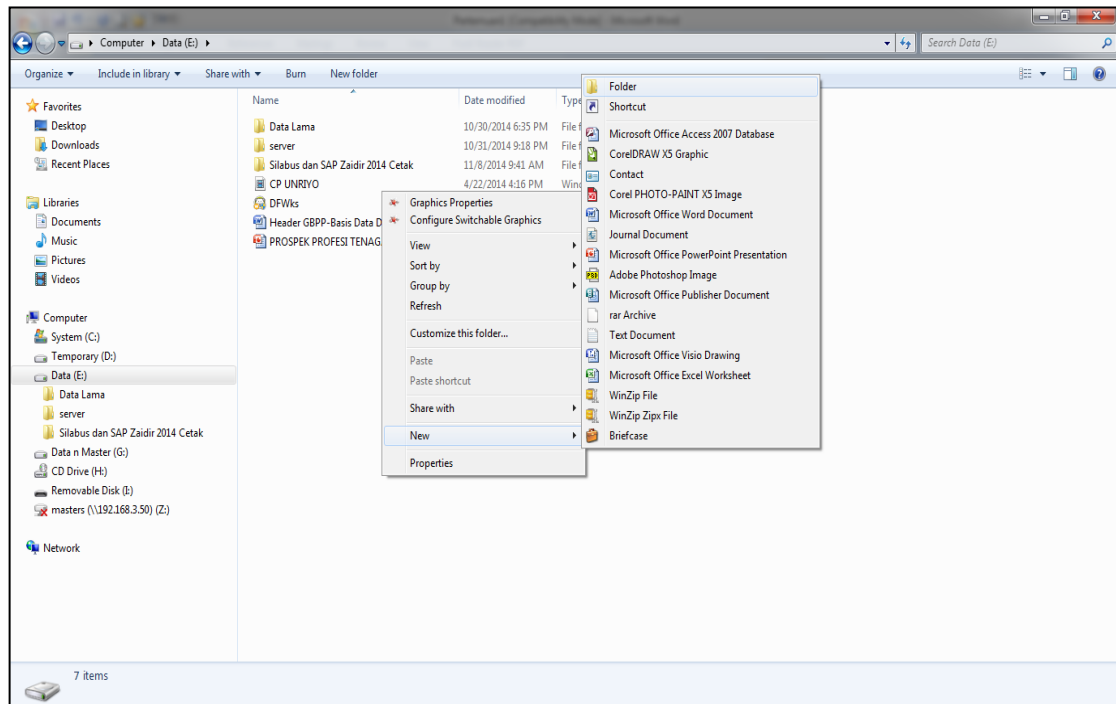
Folder kerja perlu disiapkan dengan seksama agar hasil praktek yang dilakukan dapat tersimpan dengan baik dan mudah untuk ditemukan kembali. Untuk keperluan praktek, mulai saat ini sampai beberapa pertemuan berikutnya, siapkanlah folder kerja dengan langkah sbb:

1. Aktifkan Windows Explorer, sehingga akan ditampilkan fasilitas Window Explorer seperti terlihat pada Gambar 1.9.



Gambar 1.9 Tampilan Window Explorer

2. Arahkan mouse ke layar bagian kanan dan klik kanan, pilih New, pilih Folder seperti terlihat pada Gambar 1.10.



3. Saat setelah mengklik pilihan Folder, maka lanjutkan dengan mengetik nama folder yang akan dibuat, misalnya ANSI-01 atau Pertemuan1, dll.

1.2.2 Memulai dan menjalankan Visio

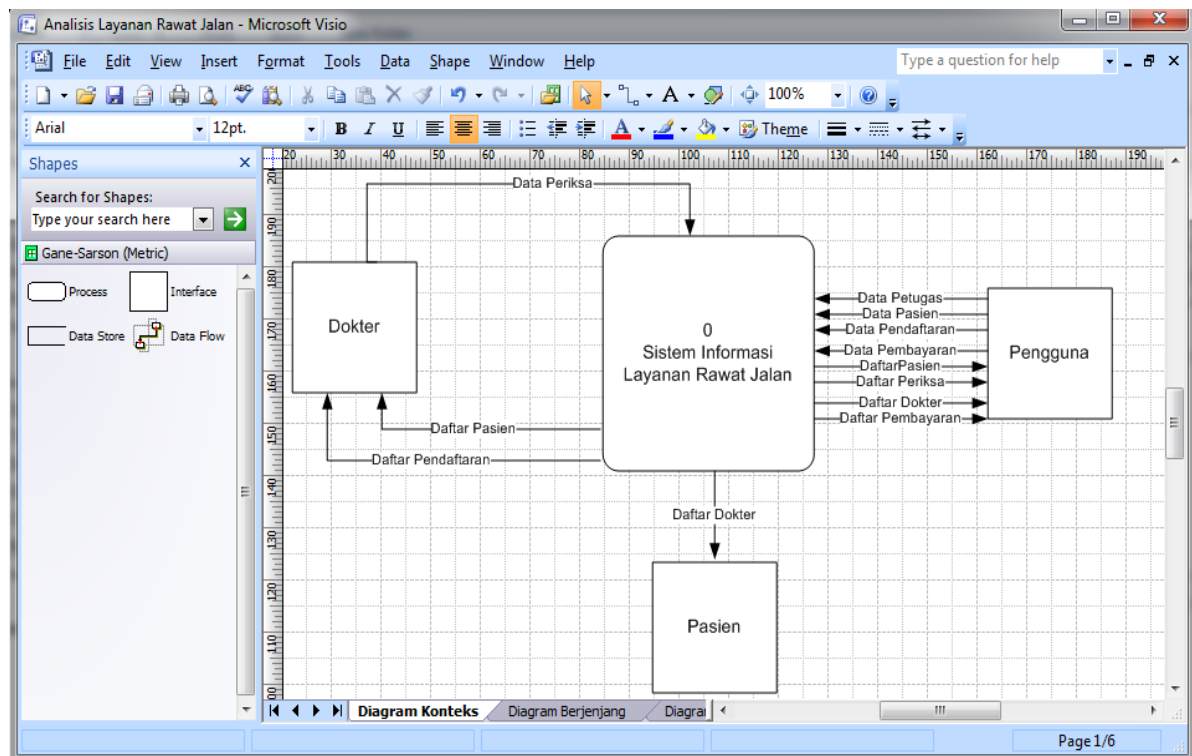
Untuk membuka program Visio, dapat mengikuti langkah sebagai berikut:

1. Klik tombol **Start** pada windows taskbar
2. Pilih **All Programs**, sehingga akan ditampilkan semua aplikasi yang telah diinstal pada komputer.
3. Pilih atau sorot kemudian klik **Microsoft Office**. Pilih atau klik **Microsoft Visio**. Tunggulah beberapa saat sampai ditampilkan seperti gambar 1.1.
4. Pilih fasilitas **Software and Database**, sehingga akan ditampilkan pilihan seperti terlihat pada Gambar 1.2.
5. Pilih fasilitas **Data Flow Model Diagram** sehingga akan ditampilkan lembar kerja seperti Gambar 1.6.

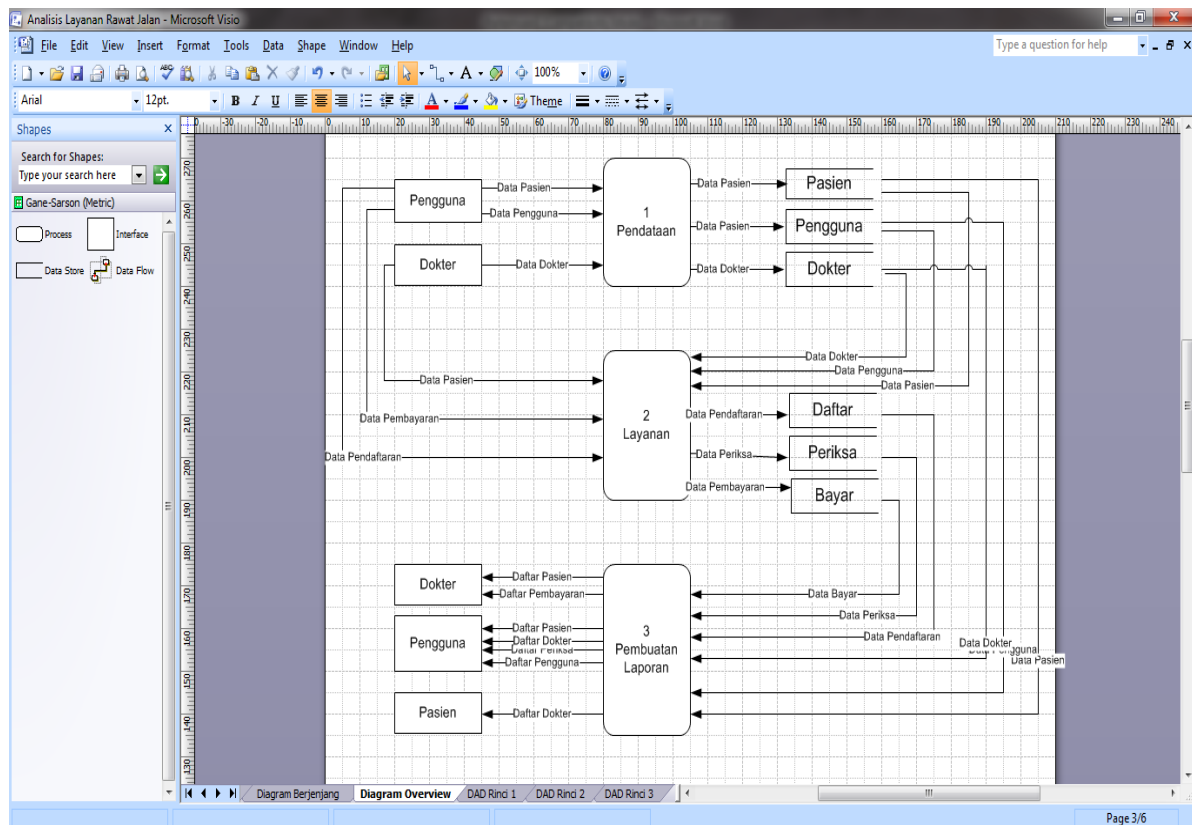
1.2.3 Membuat Diagram Arus Data Dengan Visio

Diagram Arus data (DAD) merupakan suatu bagan yang sering digunakan untuk merancang sistem secara modular. Pada contoh di bawah ini akan digambarkan mengenai cara pemanfaatan Visio untuk membuat Diagram Konteks, Diagram Overview, dan Diagram rinci dari permasalahan Sistem Informasi Rawat Jalan (Rajal)

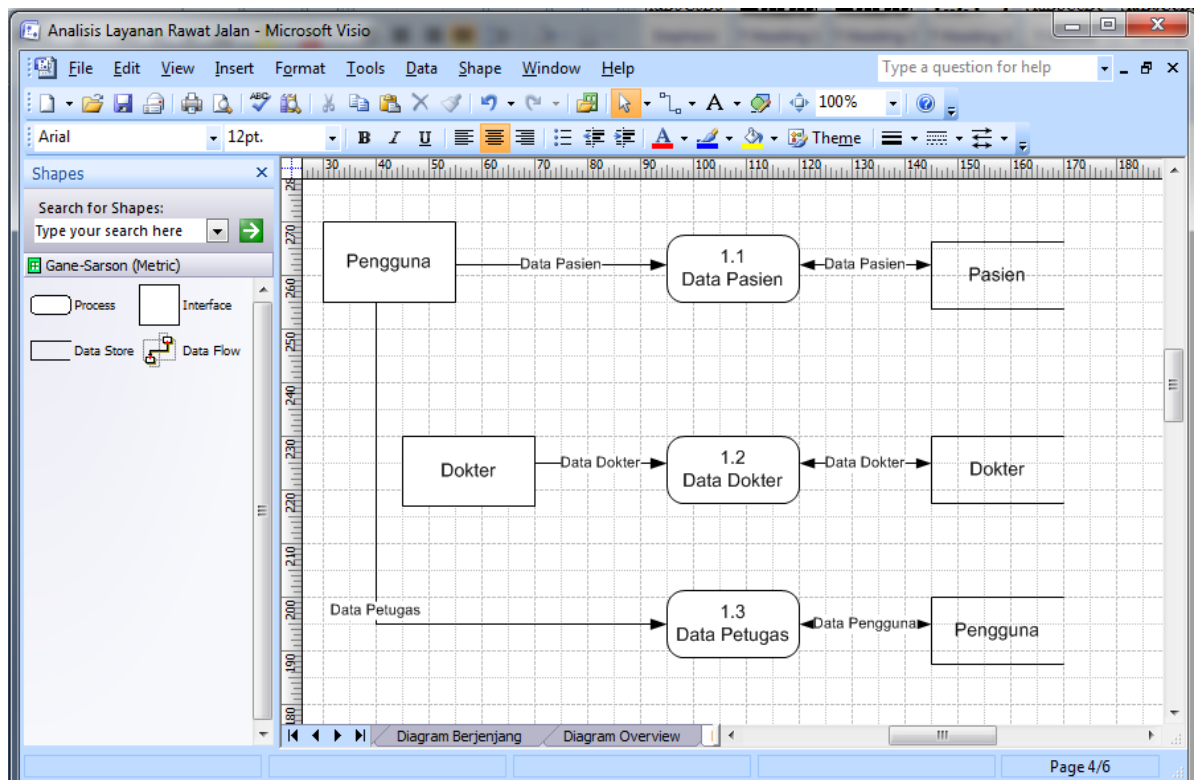
1. Diagram Konteks



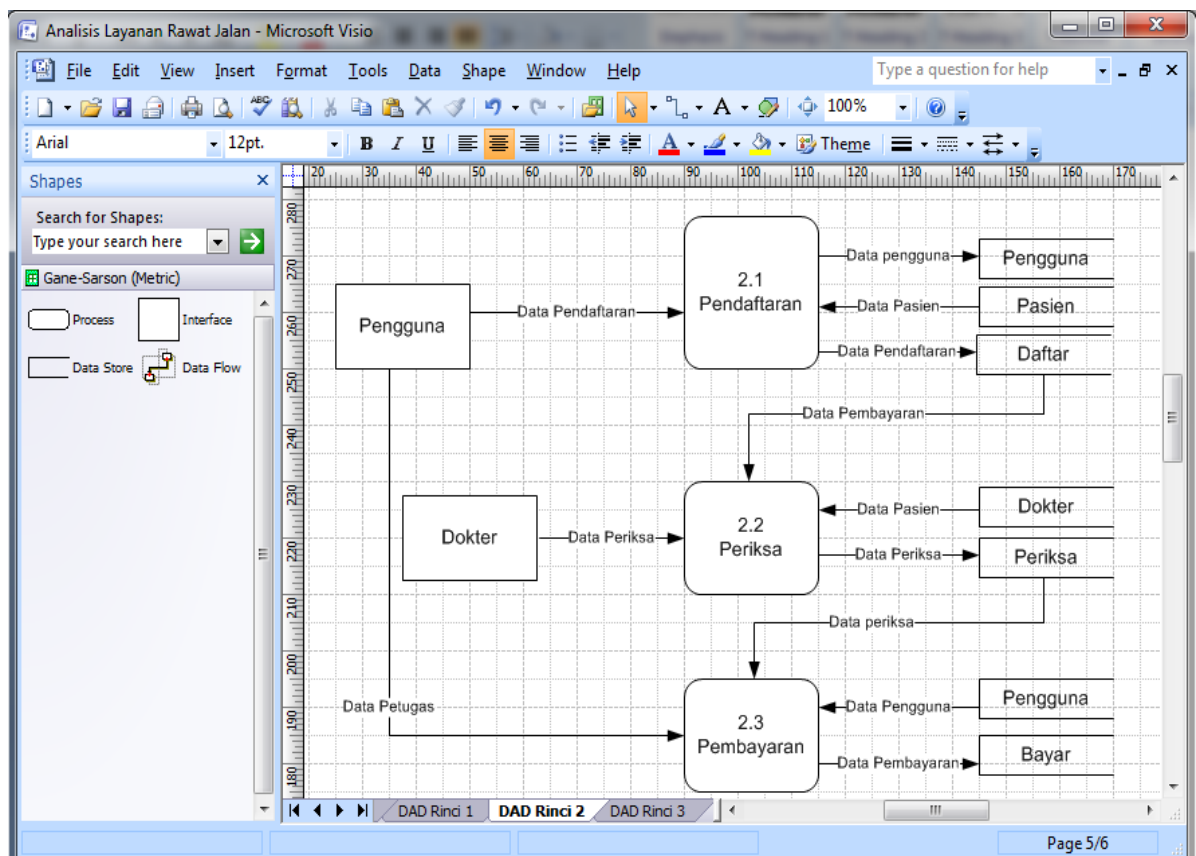
2. Diagram Overview



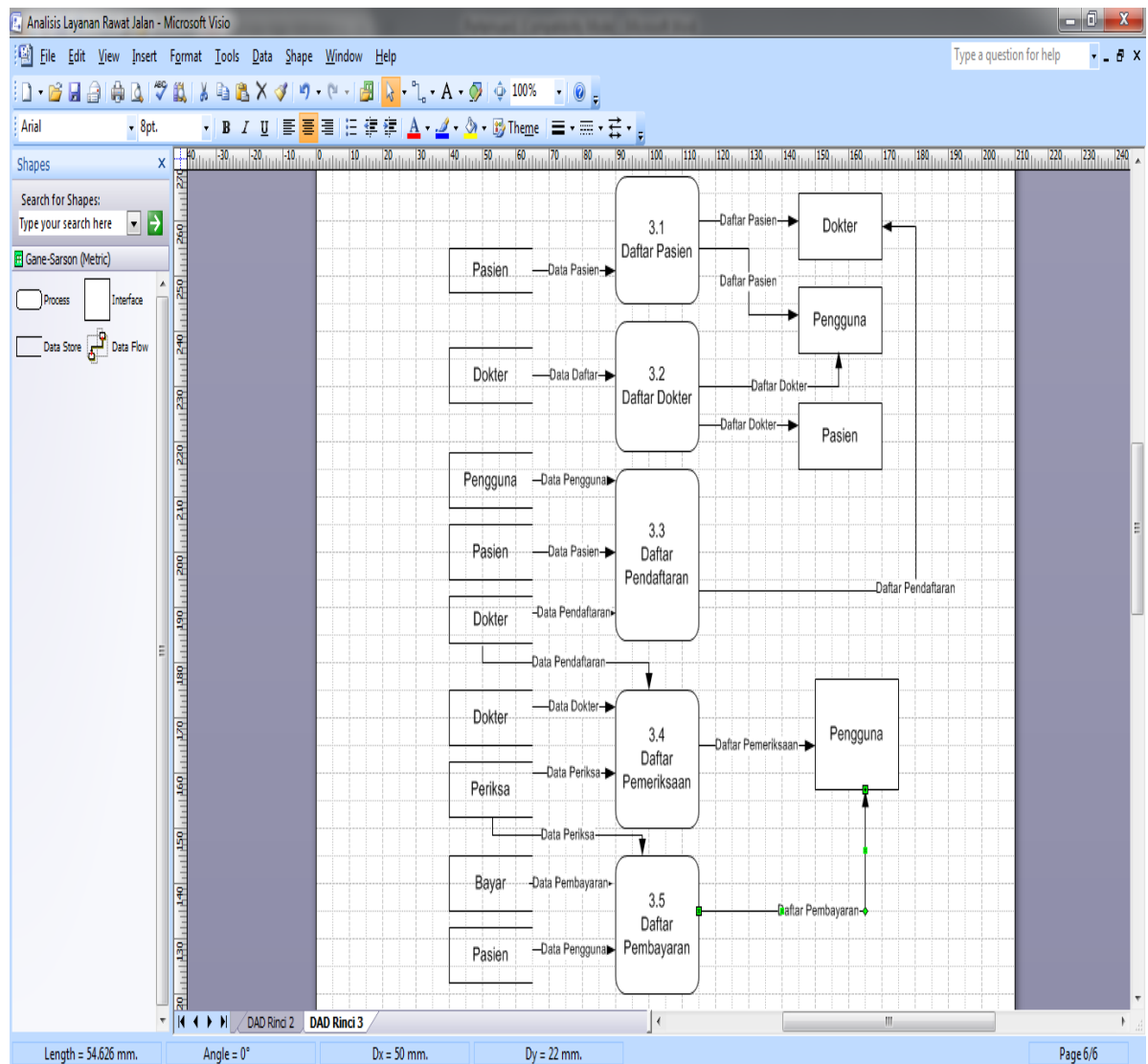
3. Diagram Arus Data Level 2 Proses 1



4. Diagram Arus Data Level 2 Proses 2



5. Diagram Arus Data Level 2 Proses 3



Penggunaan Power Designer Untuk Membuat Diagram Arus Data (DAD)

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Power Designer untuk membuat rancangan model dalam bentuk Diagram Arus Data (DAD).

Pertemuan ini akan menjelaskan secara singkat tentang pendahuluan, aturan pembuatan DAD, membuat model baru, membuat dan mendapatkan proses, membuat dan mendapatkan eksternal entitas, menjalankan Power Designer, membuat DAD dengan model Yourdan dan membuat DAD dengan model De marco.

2.1 Pendahuluan

Ide dari suatu bagan untuk mewakili arus data dalam suatu sistem dimulai pada tahun 1967 yang dikenalkan oleh Martin dan Estrin dalam bentuk suatu algoritma program dengan menggunakan simbol lingkaran dan panah untuk mewakili arus data. E. Yourdan dan L.L. Constantine juga menggunakan notasi simbol ini untuk menggambarkan arus data dalam perancangan program. G.E. Whitehouse tahun 1973 juga menggunakan notasi semacam ini untuk membuat model-model sistem matematika. Penggunaan notasi dalam diagram arus data ini sangat membantu sekali untuk memahami suatu sistem pada semua tingkat kompleksitasnya seperti yang diungkapkan oleh Cris Gane dan Trish Sarson. Pada tahap analisis, penggunaan notasi ini sangat membantu sekali di dalam komunikasi dengan pemakai sistem untuk memahami sistem secara logika. Diagram yang menggunakan notasi-notasi ini untuk menggambarkan arus dari data sistem sekarang dikenal dengan nama diagram arus data atau DAD (data flow diagram atau DFD).

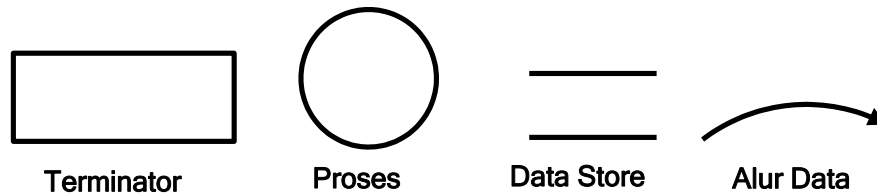
DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir (misalnya lewat telepon, surat dan sebagainya) atau lingkungan fisik dimana data tersebut akan disimpan (misalnya file kartu, microfiche, harddisk, tape, diskette dan lain sebagainya). DFD merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur (structured analysis and design). DFD merupakan alat yang cukup populer sekarang ini, karena dapat menggambarkan arus data di dalam sistem dengan terstruktur dan jelas. Lebih lanjut DFD juga merupakan dokumentasi dari sistem yang baik.

2.2 Simbol yang Digunakan

Beberapa simbol digunakan di DFD untuk maksud mewakili:

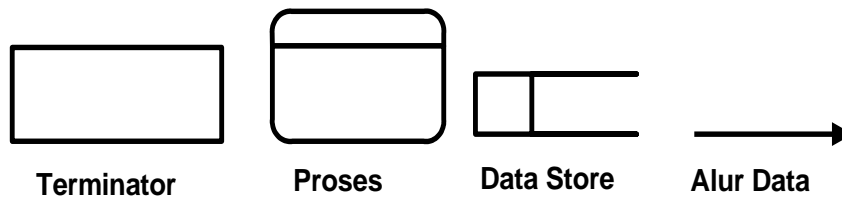
1. External entity (kesatuan luar) atau boundry (batas sistem);
2. Data flow (arus data);
3. Process (proses);
4. Data store (simpanan data)

2.2.1 Menurut Yourdan dan DeMarco



Gambar 2.1 Simbol DAD menurut Yourdan dan DeMarco

2.2.2 Menurut Gene dan Sarson



Gambar 2.1 Simbol DAD menurut Gene dan Sarson

1. Terminator / Entitas Luar (*External Entity*)
Terminator adalah entitas di luar sistem yang berkomunikasi/berhubungan langsung dengan sistem. Entitas luar ini dapat berupa orang, sekelompok orang, organisasi, perusahaan, departemen atau sistem lainnya yang berada di luar lingkungan sistem yang akan memberikan *input*/menerima *output* dari sistem.
2. Proses
Komponen proses menggambarkan kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari suatu arus data yang masuk kedalam proses (*input*) untuk menghasilkan arus data yang keluar dari proses (*output*).
3. Data Store/ Simpanan Data
Komponen ini merupakan simpanan dari data yang dapat berupa suatu *file* atau *database*, suatu arsip atau catatan manual suatu tabel atau agenda.
4. Flow Line/Alur Data
Alur data digunakan untuk menerangkan perpindahan data /paket data yang terjadi diantara proses, simpanan data dan terminator.

2.3 Aturan/Pedoman Pembuatan DAD

Pedoman untuk menggambar DAD adalah sebagai berikut:

1. Identifikasi terlebih dahulu semua kesatuan luar (*external entities*) yang terlibat di sistem. Misalmya sitem penjualan mempunyai kesatuan luar yang terlibat adalah:
 - a. Langgan

- b. Manajer Kredit
 - c. Bagian Gudang
 - d. Bagian Pengiriman
2. Identifikasi semua input dan output yang terlibat dengan kesatuan luar. Misalnya untuk sistem penjualan ini, input atau output yang terlibat dengan kesatuan luar adalah sebagai berikut:

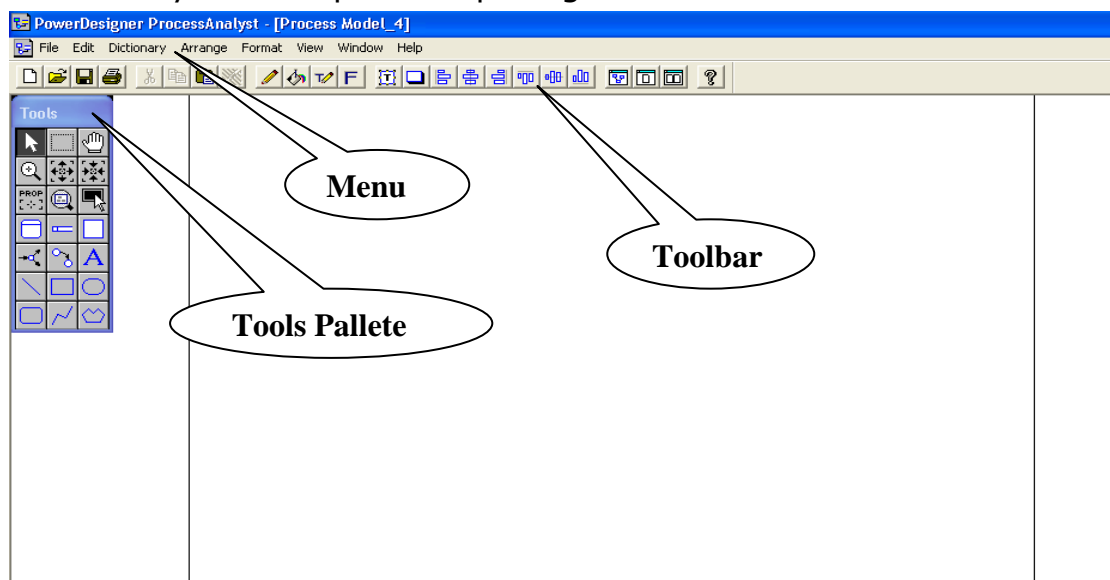
Kesatuan Luar	Input	Output
Langganan	Order langganan	-
Bagian Gudang	-	Tembusan permintaan persediaan
Bagian Pengiriman	Tembusan jurnal	Faktur, tembusan kredit dan tembusan jurnal
Manajer Kredit	-	Status piutang

3. Gambarlah terlebih dahulu suatu diagram konteks (*context diagram*). DAD merupakan alat untuk structure analysis. Pendekatan terstruktur ini mencoba untuk menggambarkan sistem pertama kali secara garis besar (disebut dengan top level) dan memecah-mecahnya menjadi bagian yang lebih terinci (disebut dengan lower level).
4. Gambarlah bagan berjenjang untuk semua proses yang ada di sistem terlebih dahulu. Bagan berjenjang (*hiachy chart*) digunakan untuk mempersiapkan penggambaran DAD ke level-level lebih bawah lagi.
5. Gambarlah sketsa DAD untuk overview diagram berdasarkan proses di bagan berjenjang
6. Gambarlah DAD untuk level-level berikutnya (DAD rinci)

2.4 Membuat Model Baru

Langkah langkah membuat Model Baru adalah sebagai berikut :

1. Klik dua kali icon program ProcessAnalyst sehingga window ProcessAnalyst di tampilkan seperti gambar dibawah ini




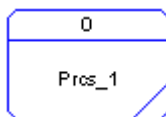
Sebelum menutup/mengakhiri program ProcessAnalyst, Anda akan menyimpan Model Baru.

2. Pilih Menu File – Save As
Window File Save As akan muncul
3. Ketikkan Nama File di kotak File Name
4. Klik OK


2.5 Membuat dan Mendapatkan Proses

Sebagai contoh Anda akan membuat Proses “Penerbitan”

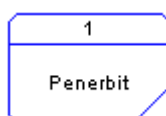
1. Klik Tool Process 
2. Klik pada layer kerja. Process mempunyai nama Prcs_n dimana n adalah nomor dari process yang dibuat



Anda dapat merubah nomor process dengan cara klik Menu – Dictionary - Renumber

3. Klik kanan mouse untuk menampilkan Process tool
4. Klik dua kali simbol Process untuk menampilkan Properti Process
5. Ketikkan “Penerbitan” dalam kotak Name
6. Klik tombol  pada baris Code untuk memberikan nama yang

sama pada code




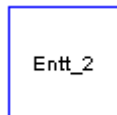
7. Klik tombol OK sehingga simbol Process menjadi seperti gambar berikut


2.6 Membuat dan Mendapatkan Eksternal Entity

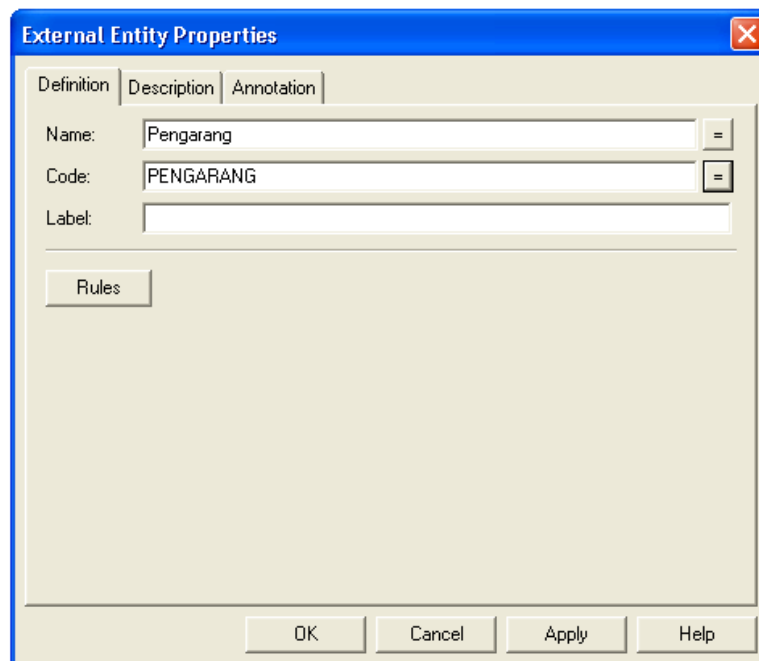
Process Penerbitan merubah data yang diterima dari "Pengarang" dan memberikan data kepada "Toko Buku"

Anda akan membuat External Entity "Pengarang"

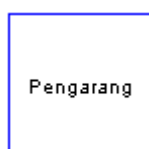
1. Klik Tool External Entity 
2. Klik Simbol kedalam layar kerja



3. Klik Kanan mouse untuk menampilkan Tool External Entity
4. Klik dua kali simbol External Entity untuk menampilkan property
5. Ketikkan "Pengarang" pada kotak Name
6. Klik tombol dibaris Code untuk memberikan nama yg sama pada code 



7. Klik OK sehingga External Entity Pengarang menjadi seperti gambar




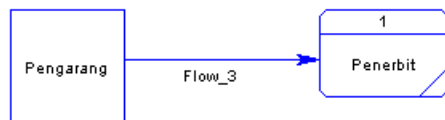
berikut

8. Ulangi langkah 1 sampai dengan 7 untuk membuat External Entity "Toko Buku"

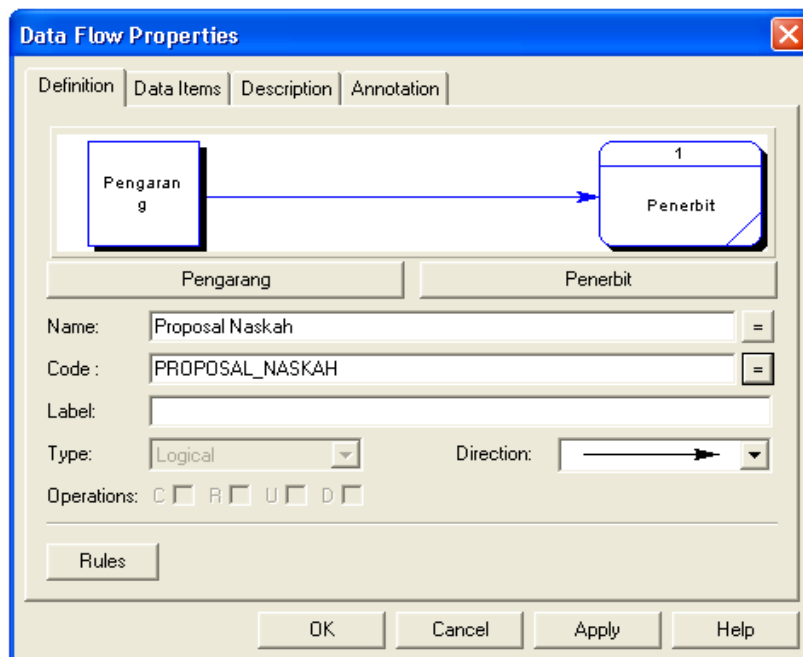
2.7 Membuat Data Flow Menggunakan Objects

Anda akan menghubungkan Object dalam Proses Utama. Dalam langkah ini anda tentukan arah data kedalam Model. Anda akan membuat Data Flow antara External Entity Pengarang dan Proses Penerbitan.

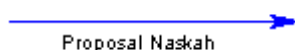
1. Klik Tool Data Flow 
2. Klik Pengarang dan tahan klik kiri mouse
3. Tarik data flow kearah Proses Penerbitan



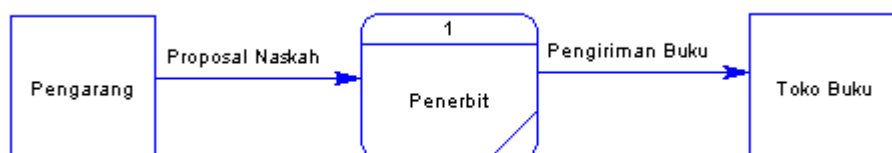
4. Klik kanan untuk menampilkan Data Flow tool
5. Klik dua kali simbol Data Flow untuk menampilkan property
6. Ketikan "Proposal Naskah" kedalam kotak Name
7. Klik tombol untuk memberikan nama yang sama



8. Klik OK sehingga akan menjadi seperti berikut



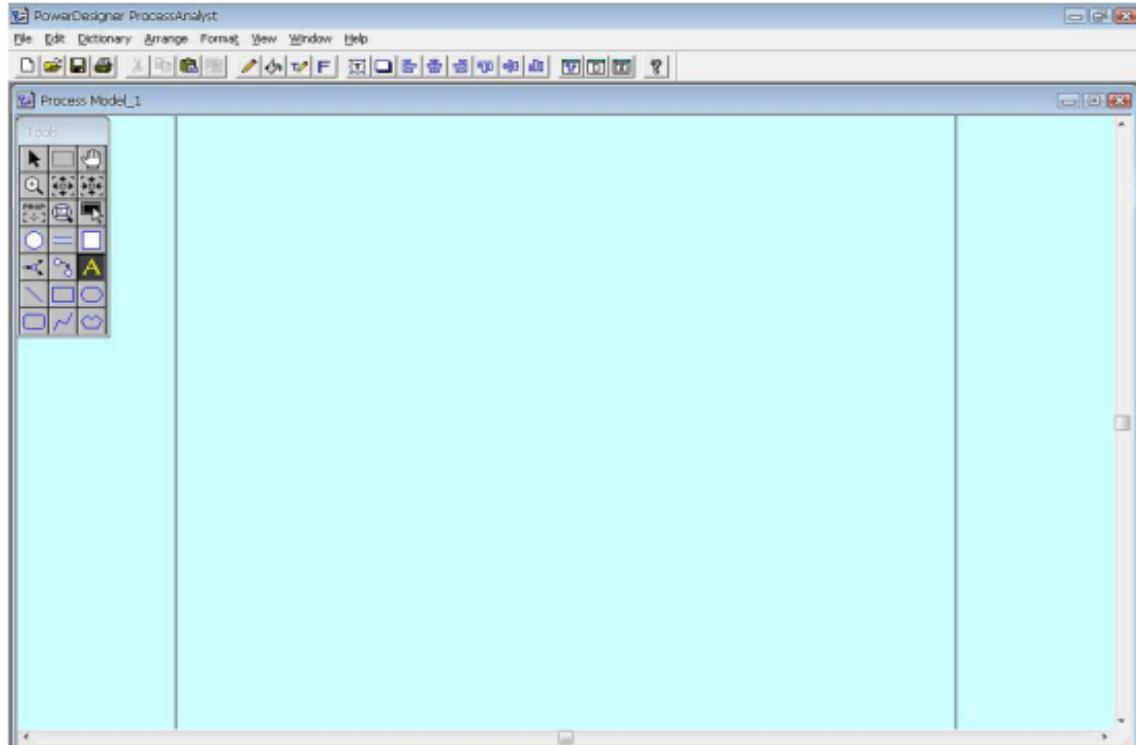
9. Ulangi langkah langkah diatas untuk membuat Data Flow dari Process Penerbitan ke External Entity Toko Buku sehingga akan menjadi seperti berikut :



2.7 Praktek

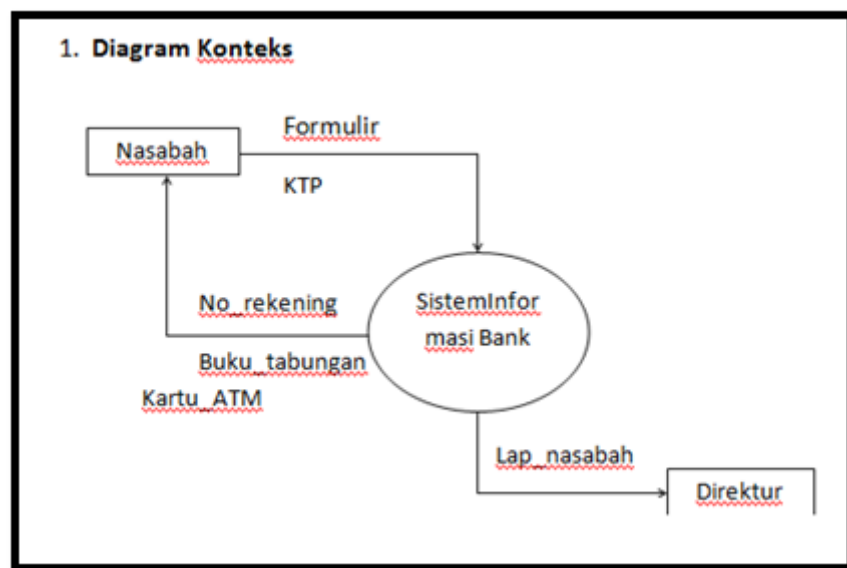
2.7.1 Menjalankan Power Designer

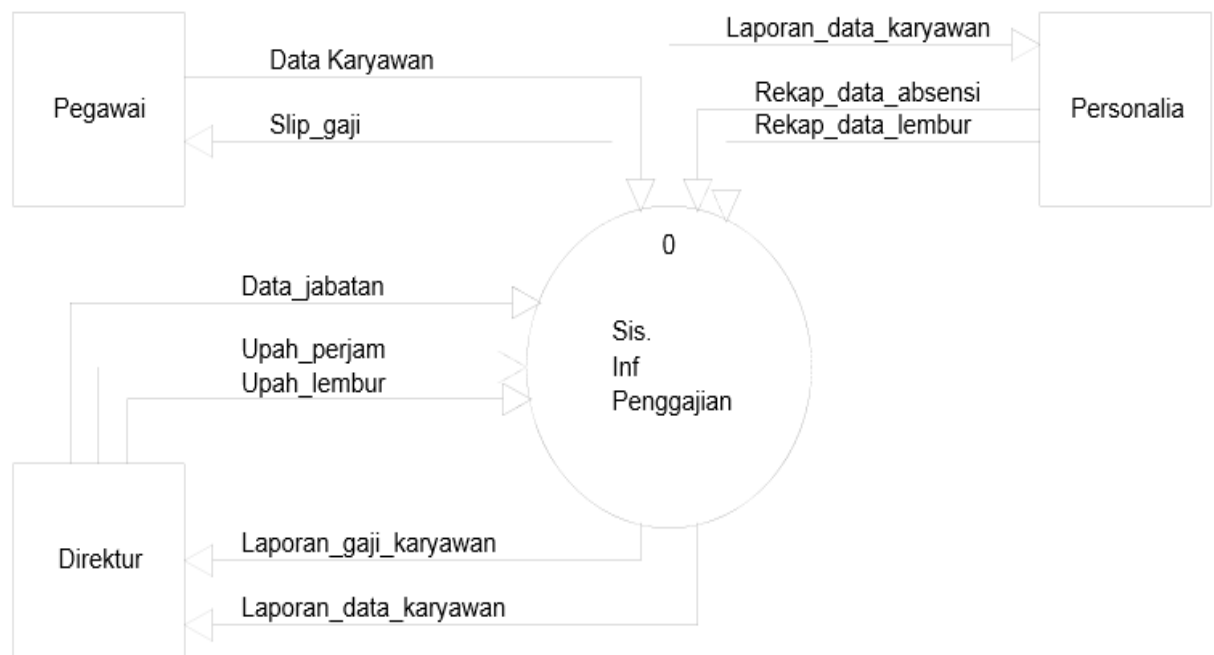
Pada saat memanggil aplikasi Power Designer dan memilih Analyst Process, akan ditampilkan menu seperti gambar 1.7.



Gambar Tampilan Awal Power Designer

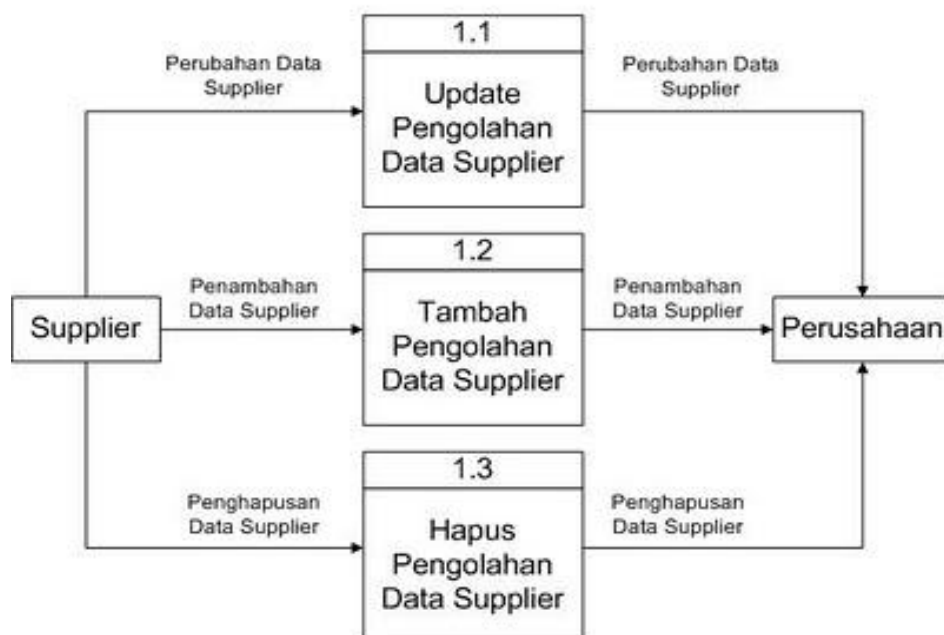
2.7.2 Membuat DAD dengan Model Yourdan dan DeMarco

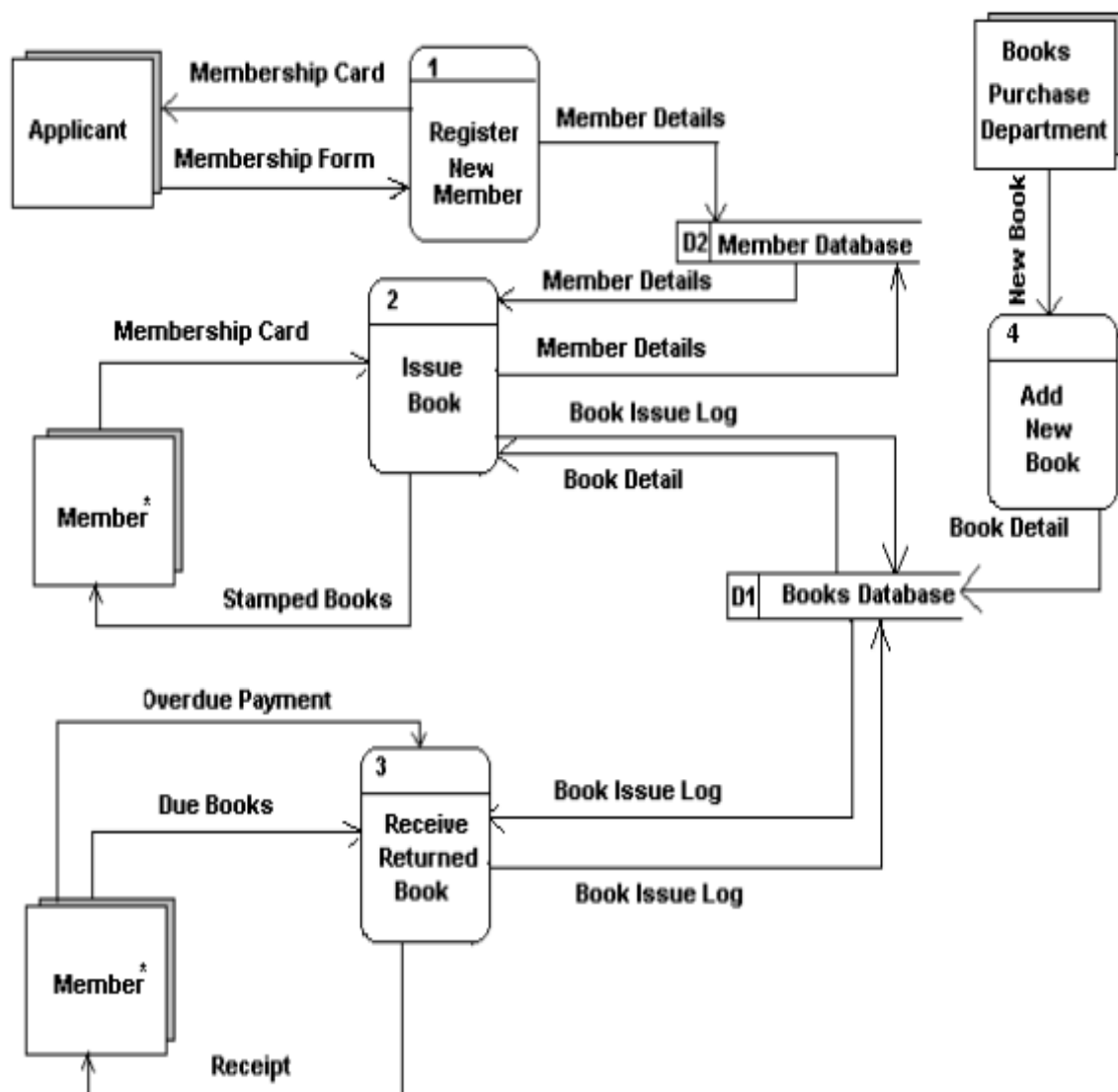




2.7.3 Membuat DAD dengan Model Gane dan Sarson

Data Flow Diagram (DFD) Level 2 Untuk Pengolahan Data Supplier





2.7.4 Soal Latihan

1. Gambarkanlah Diagram Aliran Data (data flow diagram) logic pada sebuah system pengolahan data penyimpanan / pengambilan uang disalah satu bank yang saudara ketahui. Proses-proses yang harus ada pada diagram tersebut adalah :

- Pembukaan Rekening
- Penyimpanan Uang
- Pengambilan Uang
- Penghitungan Uang.
- Penutupan Rekening
- Pembuatan Laporan.

2. Sebuah konter HP ingin melakukan perubahan pengolahan data dari manual ke sistem komputerisasi agar pelayanan pada pelanggan lebih meningkat. Sistem penjualan yang diterapkan bisa secara tunai maupun secara kredit. Pelanggan yang membeli perlu dicatat agar ada hubungan keberlanjutan untuk waktu mendatang. HP yang dijual ada beberapa merek dan jenisnya dengan harga yang berbeda. Bagi pelanggan yang melakukan pembelian tunai, akan diberi potongan 5% dari nilai transaksi. Jika pembelian secara kredit, maka ada pengolahan untuk melakukan angsuran. Untuk mempertanggungjawabkan transaksi yang terjadi, maka sistem ini dilengkapi dengan petugas yang melayani setiap shif kerja. Pimpinan bisa menikmati informasi tanpa harus meminta langsung kepada petugas, sehingga dibutuhkan sistem dapat bekerja pada jaringan.

Dari kasus di atas, kerjakanlah :

- a. Gambarkanlah rancangan proses dalam bentuk Diagram Konteks (nilai 35)
- b. Gambarkanlah flow chart sistemnya! (nilai 30)
- c. Gambarkanlah rancangan *database* secara terinci dalam bentuk Relasi Antar Tabel (nilai 35)

Penggunaan Power Designer Untuk Membuat Diagram Konteks dan Diagram Overview Untuk Suatu Masalah

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Power Designer untuk membuat Diagram Konteks dan Diagram Overview.

Pertemuan ini akan menjelaskan secara singkat tentang cara membuat Diagram Konteks dan Diagram Overview.

3.1 Pengertian Diagram Konteks

Context Diagram (CD) merupakan pendekatan terstruktur yang mencoba untuk menggambarkan sistem pertama kali secara garis besar (disebut dengan top level) dan memecah-mecahnya menjadi bagian yang lebih terinci. Context diagram ini menggambarkan hubungan input/output antara sistem dengan kesatuan luar.

Context Diagram adalah bagian dari Data Flow Diagram (DFD) yang berfungsi memetakan model lingkungan, yang dipresentasikan dengan lingkaran tunggal yang mewakili keseluruhan sistem. CD menyoroti sejumlah karakteristik penting sistem, yaitu:

1. Kelompok pemakai, organisasi atau sistem lain di mana sistem melakukan komunikasi (sebagai terminator).
2. Data masuk, yaitu data yang diterima sistem dari lingkungan dan harus diproses dengan cara tertentu.
3. Data keluar, yaitu data yang dihasilkan sistem dan diberikan ke dunia luar.
4. Penyimpanan data (*storage*), yaitu digunakan secara bersama antara sistem dengan terminator. Data ini dapat dibuat oleh sistem dan digunakan oleh lingkungan atau sebaliknya dibuat oleh lingkungan dan digunakan oleh sistem. Hal ini berarti pembuatan simbol data storage dalam CD dibenarkan, dengan syarat simbol tersebut merupakan bagian dari dunia diluar sistem.
5. Batasan, antara sistem dan lingkungan.

Aliran dalam CD memodelkan masukan ke sistem dan keluaran dari sistem, seperti halnya sinyal kontrol yang diterima atau dibuat sistem. Aliran data hanya digambarkan jika diperlukan untuk mendeteksi kejadian dalam lingkungan dimana sistem harus memberikan respon atau membutuhkan data untuk menghasilkan respon. Selain itu aliran data dibutuhkan untuk menggambarkan transportasi antara sistem dan terminator. Dengan kata lain aliran data digambarkan jika data tersebut diperlukan untuk menghasilkan respon pada kejadian tertentu.

Dalam hal ini seharusnya menggambar dengan asumsi bahwa masukan disebabkan dan diinisialisasi oleh terminator, sedangkan keluaran disebabkan dan diinisialisasi oleh sistem. Hal itu dilakukan dengan mencegah interaksi

yang tidak perlu (*extraneous prompts*) yang berorientasi pada implementasi masukan-keluaran, dan mengkonsentrasikan pemodelan pada aliran data yang esensial saja.

CD dimulai dengan penggambaran terminator, aliran data, aliran kontrol, penyimpanan, dan proses tunggal yang mempresentasikan keseluruhan sistem. Bagian termudah adalah menetapkan proses yang hanya terdiri dari satu lingkaran dan diberi nama yang mewakili sistem. Nama harus dapat menjelaskan proses.

Langkah yang dapat membantu dalam menggambarkan CD:

1. Identifikasikan seluruh informasi yang dibutuhkan.
2. Identifikasikan seluruh data yang dibutuhkan proses/informasi.
3. Identifikasikan seluruh tujuan setiap informasi bagi penggunanya.
4. Identifikasikan seluruh sumber data yang dibutuhkan proses/informasi

Cara menggambar context diagram dilakukan dengan meng-klik salah satu tool yang ada kemudian menempatkannya di lembar kerja PDPA. Langkah pertama, tempatkan dahulu satu buah proses yang merupakan context diagram dan semua entitas eksternal yang ada.

Langkah kedua adalah mengisikan properti untuk masing-masing entitas eksternal dan proses. Caranya adalah dengan meng-double-click objek yang akan diubah. Berikan nama dan code yang sesuai dengan keinginan namun merepresentasikan objek tersebut.

Langkah ketiga adalah menghubungkan entitas eksternal dan proses dengan menggunakan aliran (flow tool). Ingat, untuk menghubungkannya, tariklah garis dari tengah objek ke tengah objek yang lainnya. Jangan menarik garis dari luar objek karena program tidak akan mengenali sumber atau tujuan aliran data tersebut.

3.2 Pengertian Diagram Overview

Diagram Overview sering juga disebut sebagai Diagram Arus Data Level 1. Diagram ini merupakan dekomposisi dari Diagram Konteks dengan tujuan agar sistem lebih mudah dipahami.

3.3 Kasus

Komunitas ResiBisma merupakan sebuah komunitas yang bertujuan untuk mendirikan lembaga pendidikan gratis bagi anak-anak tidak mampu. Dalam mewujudkan tujuan itu, maka didirikanlah peternakan burung parkit yang di kemudian hari diharapkan bisa menjadi sumber dana untuk mendirikan lembaga pendidikan gratis. Sumber dana untuk pengembangan usaha parkit sendiri diperoleh dari investasi yang dilakukan oleh investor lokal.

Suatu hari datanglah Mr Tamagoci dari Negeri Antah Berantah dengan tujuan melihat-lihat koleksi burung yang ada, serta tertarik untuk berinvestasi. Cerita berlanjut ketika Mr Tamagoci mendengarkan paparan mengenai bagaimana prospek beternak burung parkit. Sebagai

intermezzo, seorang anggota komunitas memaparkan bahwa sistem keuntungan yang ditawarkan adalah sistem bagi hasil, dan bahwa bagian untuk Komunitas ResiBisma adalah ditujukan untuk mendirikan lembaga pendidikan gratis serta memberikan kesempatan belajar bagi anak-anak tidak mampu.

Rupanya Mr Tamagoci sangat tersentuh dan terharu. Dan bahkan selain berinvestasi, beliau juga dalam waktu dekat akan menyumbangkan koleksi bukunya yang berjumlah 1000 eksemplar kepada Komunitas ResiBisma dengan harapan bisa dikelola dengan baik menjadi sebuah perpustakaan yang bisa diakses oleh siapapun. Menyikapi hal ini maka Komunitas ResiBisma dengan segera mengadakan rapat koordinasi dan kemudian diputuskan untuk memanfaatkan salah satu ruangan di Mabes ResiBisma sebagai perpustakaan. Selain itu juga akan dikembangkan sebuah Sistem Informasi Perpustakaan yang berguna untuk mengelola koleksi buku yang ada serta mencatat transaksi peminjaman buku.

1. Rumusan Masalah

Komunitas ResiBisma akan mengembangkan sebuah Sistem Informasi Perpustakaan yang bisa dipergunakan untuk mendata koleksi buku serta mencatat transaksi peminjaman buku.

2. Batasan Masalah

Mempertimbangkan kondisi terkini dari Komunitas ResiBisma, maka pengembangan Sistem Informasi Perpustakaan ini dibatasi pada hal-hal berikut ini:

- a. Sistem yang dikembangkan adalah sebuah sistem berbasis desktop.
- b. *Database Management System* yang dipergunakan adalah MySQL Server 5.0.
- c. Fitur-fitur yang akan dikembangkan adalah sebagai berikut:
 - Entri Koleksi Buku
 - Pendaftaran Anggota
 - Peminjaman Buku
 - Pengembalian Buku
 - Penghitungan Denda
 - Pembuatan Laporan

3. Analisa Awal

Sebagai langkah awal, kemudian disepakati prosedur-prosedur standar yang akan dipergunakan sebagai acuan dalam kegiatan operasional perusahaan. Berikut adalah prosedur-prosedur standar yang disepakati:

- a. Koleksi buku dientrikan ke dalam sistem oleh pengelola perpustakaan.
- b. Pengelola perpustakaan adalah anggota Komunitas ResiBisma atau orang lain yang ditunjuk.
- c. Untuk dapat menggunakan sistem, pengelola harus login terlebih dahulu ke dalam sistem.
- d. Untuk dapat meminjam buku, harus terlebih dahulu mendaftar sebagai anggota perpustakaan dan akan mendapatkan kartu anggota.
- e. Seorang anggota yang akan meminjam harus menunjukkan kartu anggota.

- f. Seorang anggota perpustakaan diperbolehkan meminjam buku maksimal 4 eksemplar dalam satu transaksi peminjaman.
- g. Lama waktu peminjaman buku adalah 7 hari terhitung sejak peminjaman.
- h. Keterlambatan pengembalian buku dikenakan denda per hari.
- i. Besaran denda berbeda untuk setiap buku. Besaran denda ini dientrikan bersamaan dengan entri data buku.
- j. Baik anggota maupun pengelola perpustakaan bisa mencari data buku menggunakan fasilitas pencarian yang disediakan.
- k. Proses pencarian bisa dilakukan berdasarkan judul.
- l. Pengelola perpustakaan dapat menampilkan mencetak laporan transaksi peminjaman buku.
- m. Pengelola perpustakaan dapat menampilkan mencetak daftar koleksi buku.

3.4 Praktek

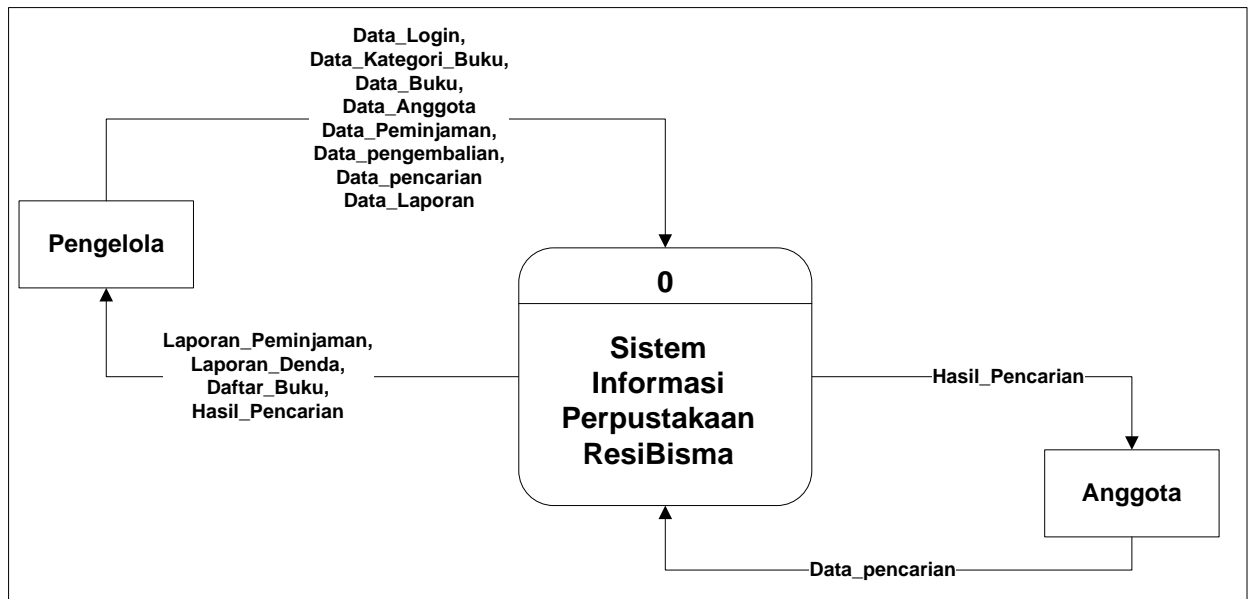
3.4.1 Membuat Diagram Konteks

Langkah awal untuk menggambarkan DFD adalah mengidentifikasi seluruh entitas yang terlibat di dalam sistem. Berikut ini adalah entitas-entitas yang terlibat di dalam sistem serta deskripsinya:

Tabel 1 Daftar Entitas yang terlibat dalam proses sistem

No	Entitas	Deskripsi
1	Pengelola	Pengelola merupakan pihak internal perpustakaan yang bertugas dalam mengelola data koleksi buku, mendata anggota perpustakaan, mencatat transaksi peminjaman dan pengembalian serta mencetak laporan.
2	Anggota	Anggota merupakan pihak eksternal yang terdaftar dan berhak untuk meminjam koleksi buku yang ada.

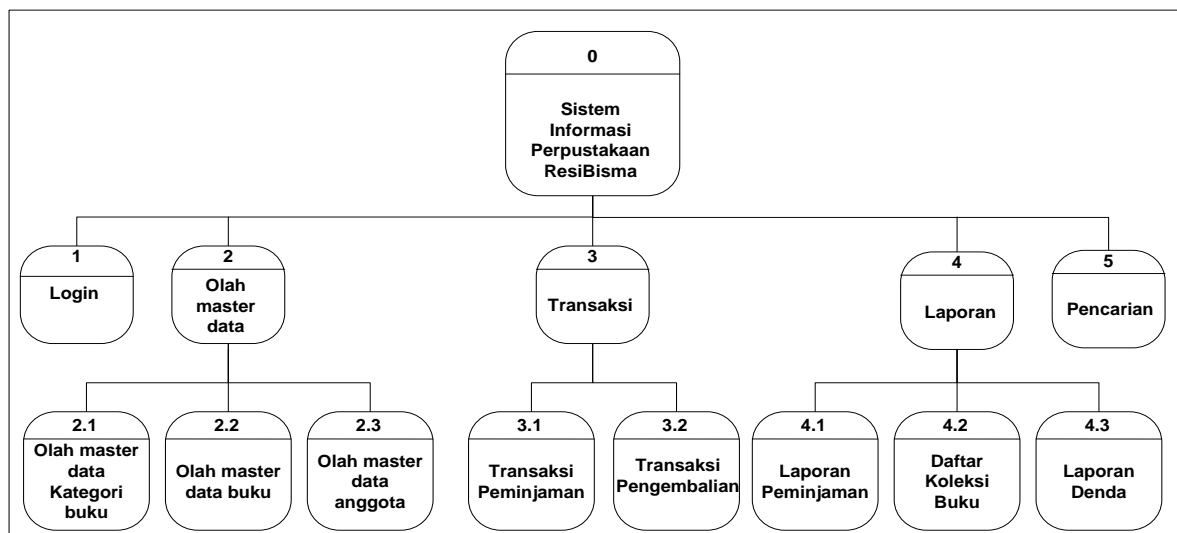
Setelah seluruh entitas yang terlibat di dalam sistem teridentifikasi, langkah selanjutnya adalah menggambarkan Diagram Konteks seperti gambar 3 di bawah ini.



Gambar 1 Diagram Konteks

3.4.2 Membuat *Diagram Overview*

Sebelum membuat *Diagram Overview*, disarankan untuk membuat Diagram Berjenjang. Namun diagram berjenjang ini hanya sifatnya untuk membantu saja dan dibuat pada kertas kerja saja. Sebagai contoh diagram berjenjang untuk sistem informasi perpustakaan dapat digambarkan sebagai berikut:

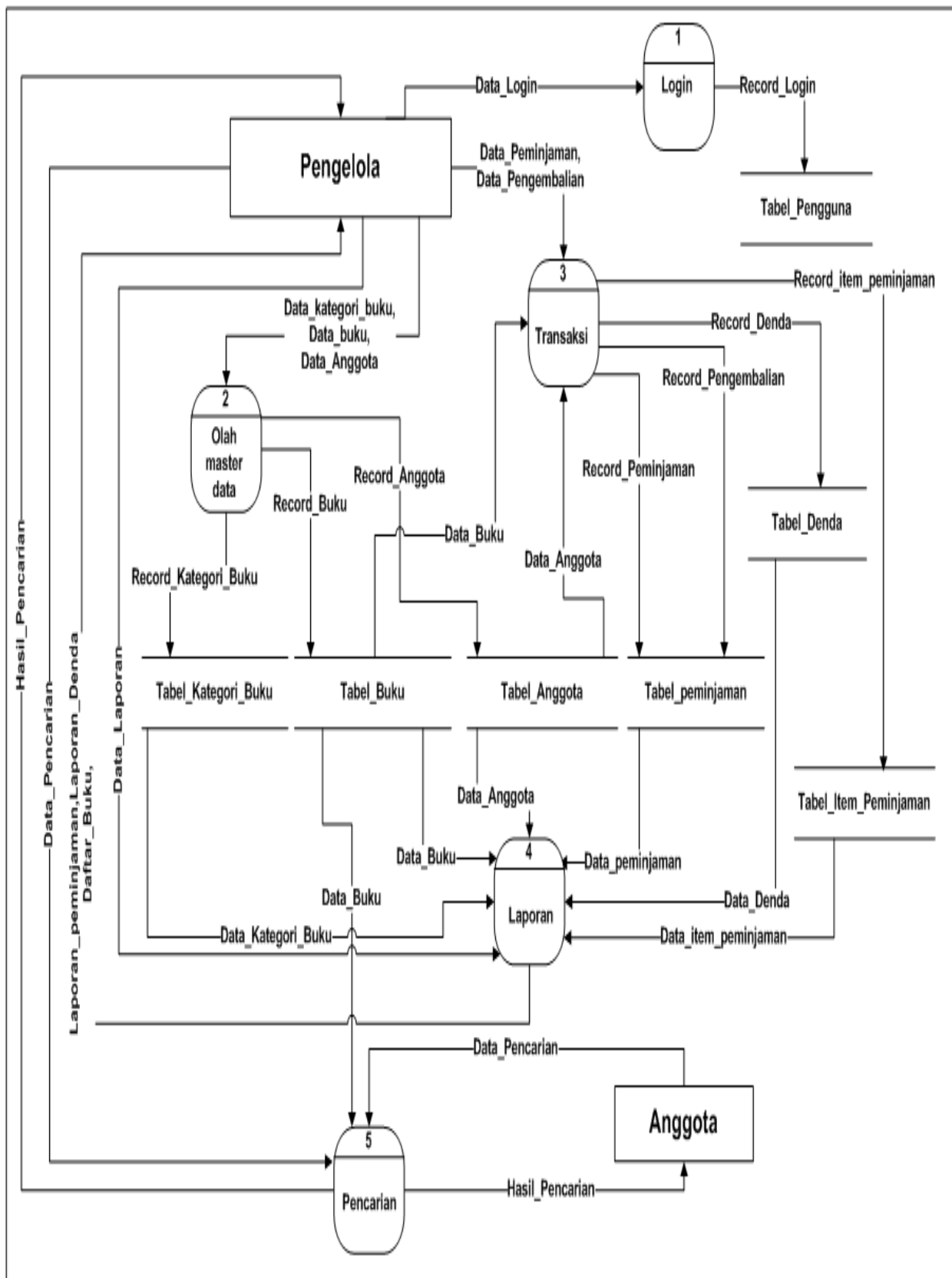


Gambar 2 Diagram berjenjang

Berdasarkan diagram berjenjang pada gambar 2, akan dibuat diagram overview. Pada tabel 2 ditampilkan daftar proses yang ada beserta deskripsi, input serta output untuk masing-masing proses. Sedangkan diagram overview dapat dilihat pada gambar 3.

Tabel 2 Proses-proses dalam sistem beserta deskripsinya

No Proses	Nama Proses	Deskripsi	Input	Output
1	Login	Proses login. Pengelola harus login terlebih dahulu sebelum dapat menggunakan sistem. Pada proses login, pengelola memasukkan username dan password	Data_login	Record_Login
2	Olah Master Data	Proses ini dipergunakan untuk mengelola data-data yang nantinya akan dipergunakan untuk seluruh proses di dalam sistem.	Data_kategori_buku, Data_Buku, Data_Anggota	Record_kategori_buku, Record_Buku, Record_Anggota
3	Transaksi	Transaksi merupakan proses yang terkait dengan sirkulasi buku dalam perpustakaan.	Data_peminjaman, Data_item_peminjaman, Data Pengembalian, Data_Buku, Data_anggota	Record_peminjaman, Record_item_peminjaman, Record_pengembalian
4	Laporan	Merupakan proses menampilkan dan mencetak laporan	Data_Laporan, Data_Kategori_Buku, Data_buku, Data_Anggota, Data_Peminjaman, Data_item_peminjaman	Laporan_Peminjaman, Daftar_Buku
5	Pencarian	Merupakan proses pencarian koleksi buku oleh pengelola dan anggota	Data_pencarian	Hasil_pencarian



Gambar 3 Diagram Overview

Penggunaan Power Designer Untuk Membuat Diagram Rinci

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Power Designer untuk membuat diagram rinci.

Pertemuan ini akan menjelaskan secara singkat tentang proses pembuatan diagram rinci berdasarkan diagram konteks dan diagram overview pada pertemuan 3.

4.1 Pengertian Diagram Rinci

Diagram Arus Data rinci merupakan turunan atau dekomposisi dari diagram overview untuk proses-proses yang masih memungkinkan untuk dipecah lagi. Diagram rinci tentu saja dimaksudkan agar sistem yang dikembangkan dapat lebih mudah dipahami sampai pada titik terkecil.

4.2 Beberapa Hal Yang Harus Diperhatikan

Untuk membuat diagram rinci, ada beberapa hal yang harus diperhatikan, yaitu:

1. Diagram rinci atau pecahan proses minimal dua proses.
2. Jangan terlalu banyak persimpangan, sehingga eksternal entitas maupun simpanan dapat digambarkan lebih dari satu kali.

4.3 Praktek

4.3.1 Membuat Diagram Rinci

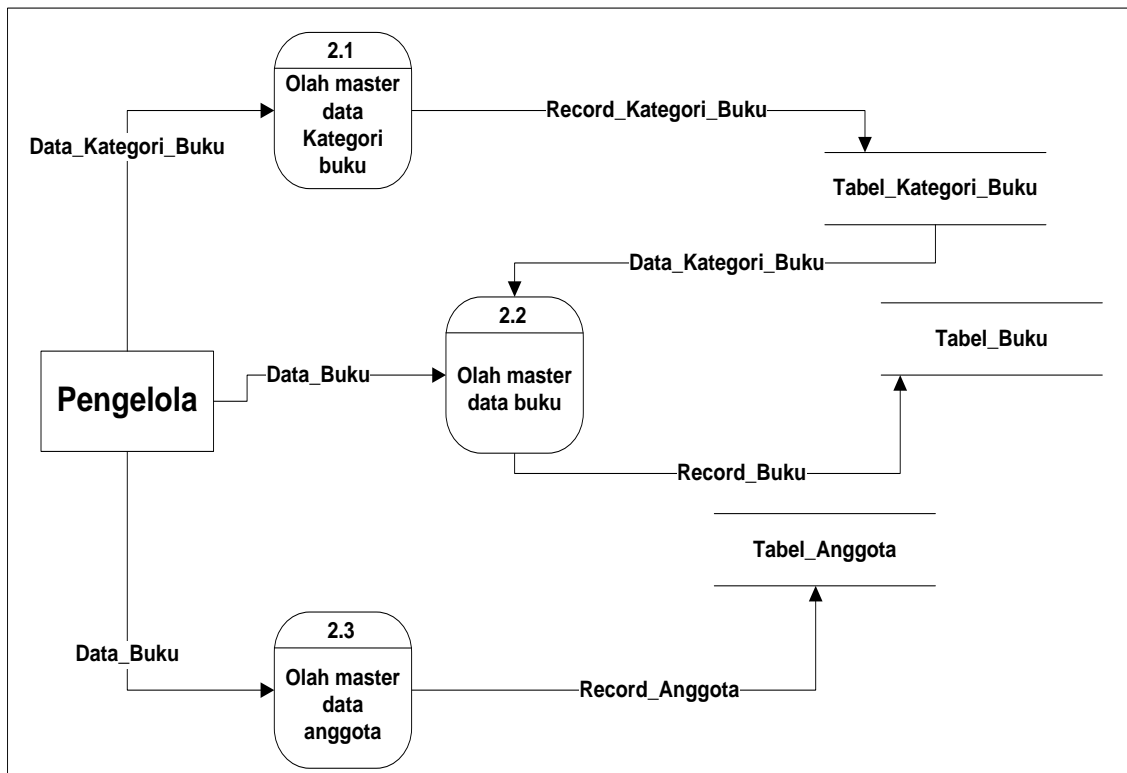
Setelah semua proses direpresentasikan dalam diagram *overview*, langkah selanjutnya adalah memecah proses-proses yang masih perlu dipecah ke dalam proses-proses yang lebih kecil yang lebih detail. Dari kasus pada pertemuan 3, terdapat tiga proses yang perlu dipecah lagi ke dalam proses-proses yang lebih kecil, yaitu olah master data, transaksi dan laporan.

Untuk memecah proses-proses ini, akan dimulai dari proses olah data master. Proses-proses yang ada pada proses transaksi dapat dilihat pada tabel 1.

Tabel 1 Proses-proses pada proses olah master data

No Proses	Nama Proses	Deskripsi	Input	Output
2.1	Olah master data kategori buku	Setiap buku termasuk dalam kategori tertentu. Untuk itu perlu didefinisikan kategori-kategori buku yang ada. Pada proses ini, pengelola dapat menambah, mengedit serta menghapus data kategori buku.	Data_kategori_buku	Record_kategori_buku
2.2	Olah Master Data Buku	Merupakan proses pengelolaan koleksi buku. Pada proses ini, pengelola dapat menambah, mengedit serta menghapus data buku. Penambahan dan pengeditan data buku akan membutuhkan data_kategori_buku yang langsung diakses dari tabel_kategori_buku.	Data_kategori_buku, Data_Buku,	Record_buku
2.3	Olah Master Data Anggota	Proses ini dipergunakan untuk mengelola data anggota. Ketika seseorang mendaftar menjadi anggota perpustakaan, pengelola akan menambahkan data anggota melalui proses ini. Selain itu, pengelola juga dapat mengedit dan menghapus data anggota melalui proses ini.	Data_Anggota	Record_anggota

Berdasarkan pemecahan proses olah master data di atas, kemudian digambarkan DFD Level 2 untuk proses olah master data seperti dapat dilihat pada gambar 1.



Gambar 1 DFD Level 2 Proses Olah Master Data

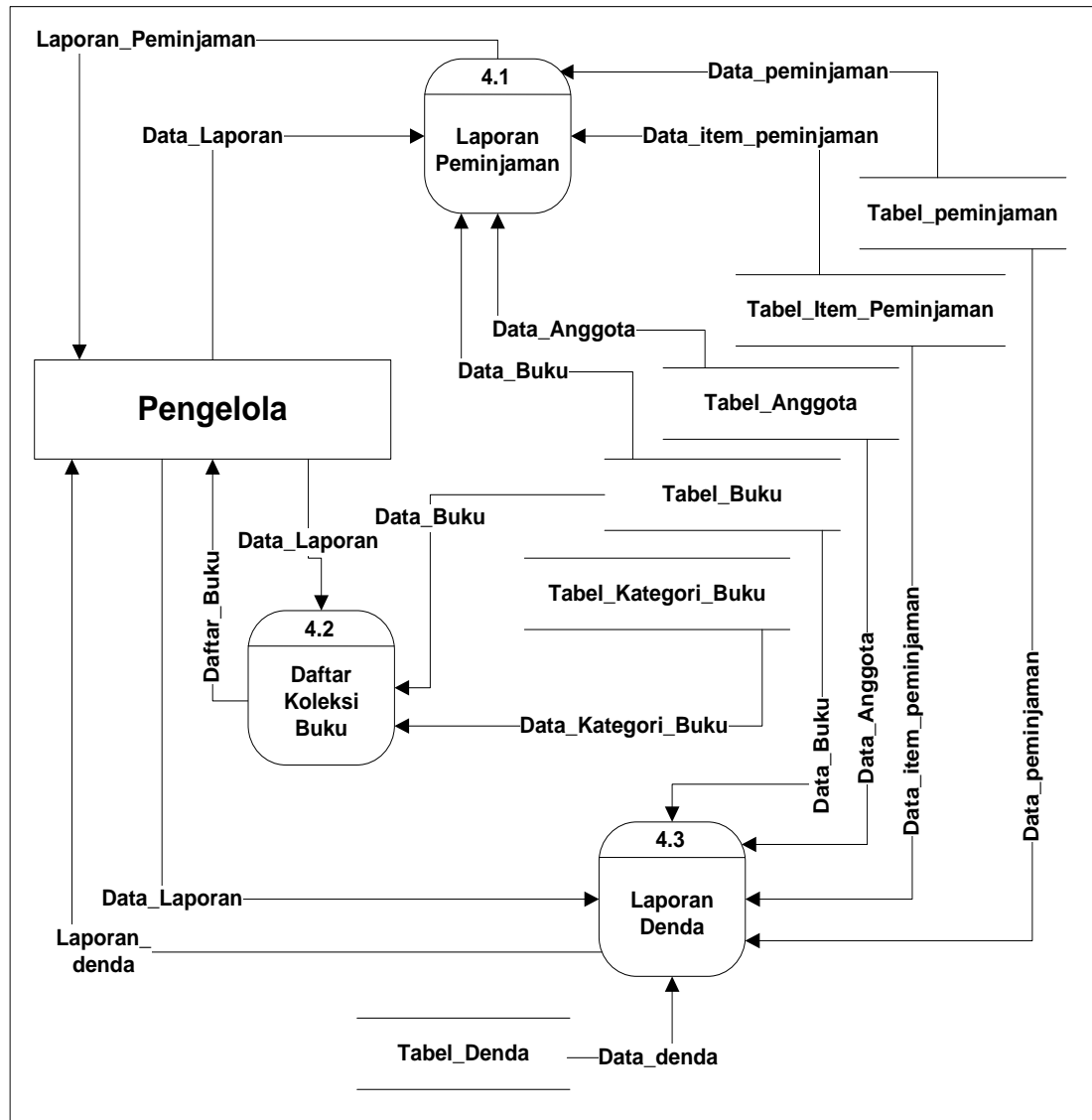
Untuk proses transaksi, proses-proses yang ada dapat dilihat pada tabel 2.

Tabel 2 Proses-proses pada proses transaksi

No Proses	Nama Proses	Deskripsi	Input	Output
3.a	Peminjaman	Merupakan transaksi peminjaman buku oleh anggota.	Data_Buku, Data Anggota, Data_Peminjaman	Record_peminjaman, Record_item_peminjaman
3.b	Pengembalian	Merupakan transaksi pengembalian buku yang dipinjam oleh anggota	Data_buku, data_anggota, data_Peminjaman, data_Item_peminjaman	Record_peminjaman, Record_denda

Gambar 7 merupakan DFD Level 1 untuk proses transaksi.

	Denda	untuk menampilkan dan mencetak pengembalian pinjaman buku yang terlambat sehingga anggota dikenakan denda.	Data_item_peminjaman, Data_Anggota, Data_buku, Data_laporan, Data_Denda	
--	-------	--	---	--



Gambar 8 DFD Level 1 Proses Laporan

4.3.2 Menguji Kebenaran Model

Process Analyst dapat kita check jika sebuah ProcessAnalyst Model (PAM) menggunakan metode yang benar. Pesan Kesalahan dan Peringatan (*Error and Warning*) menandakan masalah dalam PAM

1. Pilih Menu **Dictionary – Check Model**

Kotak dialog proses check tampil. Ini akan menampilkan Pesan Kesalahan dan Peringatan. Pesan terakhir menandakan apakah model sudah benar.

2. Klik **OK**

Anda dapat melihat Pesan Kesalahan dengan menggunakan Menu Dictionary – Display Messages.

Penggunaan Power Designer Untuk Pembuatan Rancangan Basis Data

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Power Designer untuk membuat rancangan basis data.

Pertemuan ini akan menjelaskan secara singkat tentang cara menemukan entitas, menentukan field kunci, dan menambahkan atribut.

5.1 Menemukan Entitas

- Buat ilustrasi/gambaran cerita tentang sistem yang akan dicari entitasnya
- Tandai setiap objek yang diwakili oleh kata benda yang ada di dalam ilustrasi tersebut
- Untuk setiap objek tersebut yakinkan bahwa ia memiliki karakteristik yang nanti disebut sebagai atribut
- Tentukan objek yang merupakan entitas (Jika memang ia memiliki karakteristik jadikan ia sebagai entitas)

5.2 Menentukan Field Kunci

5.3 Menambahkan Atribut

5.4 Praktek

5.4.1 Membuat Data Item

5.4.2 Membuat Data Store

5.4.3 Membuat Relasi

Analisa data dilakukan dengan mengidentifikasi entitas-entitas yang ada di dalam sistem, relasi yang terjadi antar entitas serta kardinalitas untuk setiap relasi yang terjadi. Untuk memodelkan data, dipergunakan *Entity Relation Diagram* (ERD). Berikut adalah langkah-langkah yang harus dilakukan dalam pemodelan data menggunakan ERD:

- 1) Identifikasi entitas-entitas yang ada beserta atributnya. Entitas merupakan objek yang terdapat di dalam sistem dan berbeda dari yang lain. Entitas dapat berupa orang, kejadian, perusahaan, dan lain-lain.
- 2) Tentukan relasi yang terjadi antar entitas.
- 3) Tentukan kardinalitas pada relasi-relasi yang ada.

Pada tabel 6 disajikan daftar entitas yang terdapat di dalam sistem beserta deskripsi dan atribut-atributnya.

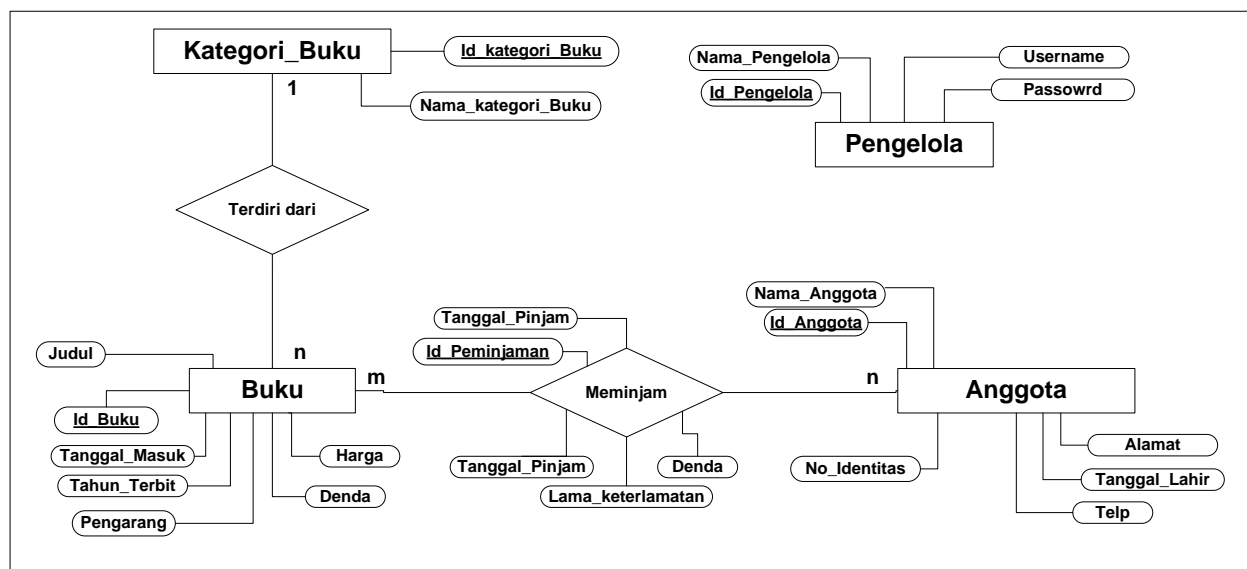
Tabel 6 Daftar Entitas serta deskripsinya

No	Entitas	Deskripsi	Atribut	Atribut Kunci
1	Pengelola	Merujuk pada orang yang mengelola sistem perpustakaan. Merupakan entitas bebas yang tidak berelasi dengan entitas lain.	Id_pengelola, nama_pengelola, username, password	Id_Pengelola
2	Anggota	Merujuk kepada orang yang menjadi anggota perpustakaan	Id_Anggota, nama_Anggota, no_identitas, alamat, telp, tanggal_lahir	Id_Anggota
3	Kategori_buku	Merujuk pada kategori-kategori buku yang dipergunakan untuk mengklasifikasi buku.	Id_kategori_buku, Nama_kategori_buku	Id_Kategori_buku
4	Buku	Merujuk kepada koleksi buku perpustakaan.	Id_buku, judul, pengarang, tahun_terbit, tanggal_masuk, harga, denda	Id_buku

Sedangkan relasi antar entitas beserta kardinalitasnya disajikan pada tabel 7 di bawah ini. Sedangkan pada gambar 20 merupakan diagram relasi entitas untuk entitas-entitas yang ada pada tabel 7.

Tabel 7 Relasi antar entitas

No	Relasi	Entitas yang berelasi			
		Entitas	Kardinalitas	Entitas	Kardinalitas
1	Terdiri dari	Kategori - Buku	One	Buku	Many
2	Meminjam	Anggota	Many	Peminjaman	Many



Gambar 20 Diagram Relasi Entitas

Berikut adalah makna logis dari relasi yang terjadi antar entitas:

- 1) **Terdiri dari**
Merupakan relasi *one to many* antara entitas kategori_buku dan entitas buku. Sebuah kategori buku bisa terdiri dari nol atau lebih buku, sedangkan satu buku pasti termasuk dalam satu kategori_buku tertentu.
- 2) **Meminjam**
Merupakan relasi *many to many* antara entitas buku dan anggota. Seorang anggota boleh meminjam lebih dari satu buku dan satu buku bisa dipinjam oleh lebih dari seorang anggota pada saat yang berbeda. Transaksi peminjaman akan selalu diikuti oleh transaksi pengembalian buku. Setiap pengembalian buku yang terlambat akan

dikenakan denda. Besarnya denda ditentukan oleh lamanya keterlambatan dan nominal denda untuk setiap buku.

Konversi ERD ke Tabel

Langkah selanjutnya dari proses analisa data adalah mengkonversikan ERD ke dalam tabel-tabel. Berikut adalah tabel-tabel hasil konversi ERD pada gambar 3.7:

- 1) Tabel Pengelola
(Id_Pengelola, Nama_Pengelola, Username, Password)
Merupakan hasil konversi dari entitas pengelola.
- 2) Kategori_Buku
(Id_Kategori_Buku, Nama_Kategori_Buku)
Merupakan hasil konversi dari entitas kategori_buku.
- 3) Tabel Buku
(Id_Buku, Id_Kategori_Buku, judul, pengarang, tahun_terbit, tanggal_masuk, harga, denda)
Merupakan hasil konversi relasi *one to many* antara entitas kategori_buku dan entitas buku. Pada tabel buku ini, atribut kunci pada entitas dengan kardinalitas *one* yaitu kategori_buku (id_kategori_buku) akan menjadi kunci tamu pada entitas dengan kardinalitas *many*.
- 4) Tabel Anggota
(Id_Anggota, nama_anggota, no_identitas, tanggal_lahir, alamat, telp)
Merupakan hasil konversi dari entitas anggota.
- 5) Tabel Peminjaman
(Id_Peminjaman, id_anggota, Id_Buku, tanggal_pinjam, tanggal_kembali, lama_keterlambatan, denda)
Tabel ini merupakan hasil dari relasi *many to many* antara entitas buku dan entitas anggota, sehingga atribut kunci dari entitas yang berelasi akan menjadi kunci tamu pada tabel ini.

Normalisasi Tabel

Sebelum tabel-tabel hasil konversi ERD dapat diimplementasikan, harus terlebih dahulu dilakukan normalisasi. Terdapat lima tahapan normalisasi, dari normalisasi bentuk pertama (1NF) sampai normalisasi bentuk kelima (5NF), namun biasanya hanya dilakukan normalisasi sampai bentuk ketiga (3NF) saja, karena sudah cukup memadai untuk

menghasilkan tabel-tabel dengan kualitas baik. Berikut adalah persyaratan dari 1NF sampai ke 3NF:

1) 1NF/ bentuk normal pertama

Dikatakan bentuk normal pertama jika:

- Setiap atribut sudah dalam bentuk atomic (tunggal, tidak dapat dipecah lagi).
- Tidak memiliki atribut bernilai banyak (*multivalued attribute*), atribut composite atau kombinasinya dalam domain data yang sama.

Contoh:

- Tabel data mahasiswa:

Nim	Nama	Hobi
983124001	Joe Satriani	Musik
983124001	Joe Satriani	Main Bola
983124002	Yngwie Malmsteen	Musik
983124002	Yngwie Malmsteen	Memasak
983124003	Paul Gilbert	Musik

Pada tabel di atas, terdapat grup berulang. Di mana mahasiswa dengan Nim 983124001 memiliki hobi music dan main bola, kemudian mahasiswa dengan nim 983124002 memiliki hobi musik dan memasak. Sehingga dapat dikatakan bahwa terdapat nilai yang tidak bernilai atomic, di mana terjadi pengulangan nim dan nama. Implementasi nilai yang tidak atomic dapat menyebabkan kompleksitas penyimpanan dan grup berulang. Sehingga tabel di atas harus didekomposisi/ dipecah lagi menjadi tabel-tabel berikut:

Tabel mahasiswa:

Nim	Nama
983124991	Joe Satriani
983124002	Yngwie Malmsteen
983124003	Paul Gilbert

Tabel Mahasiswa-Hobi

Nim	Hobi
983124001	Musik
983124001	Main Bola
983124002	Musik
983124002	Memasak
983124003	Musik

- Contoh composite

<u>Id_Mata_Kuliah</u>	<u>Id_dosen</u>	<u>Kelas</u>	<u>Jadwal</u>

Pada tabel di atas, atribut jadwal berisi gabungan hari dan jam. Jika asumsi hari dan jam memegang peranan penting dalam basisdata, maka perlu dipisah ke dalam atribut yang berbeda, menjadi:

Id_Mata_Kuliah	Id_dosen	Kelas	Jadwal_Hari	Jadwal_Jam

2) 2NF/ bentuk normal kedua

Dikatakan bentuk normal kedua jika sudah dalam bentuk normal pertama dan semua atribut bukan kunci sudah bergantung sepenuhnya terhadap atribut kunci. (ketergantungan fungsional).

3) 3NF/ bentuk normal ketiga

Dikatakan sudah dalam bentuk normal ketiga jika sudah tidak terdapat lagi ketergantungan transitif. Ketergantungan transitif adalah ketergantungan atribut bukan kunci terhadap atribut bukan kunci lainnya.

Berdasarkan persyaratan di atas, maka proses normalisasi dapat dilakukan. Berikut adalah hasil dari normalisasi tabel-tabel hasil konversi ERD:

1) Tabel Pengelola

(Id_pengelola, nama_pengelola, username, password)

Tabel ini sudah dalam bentuk normal ketiga.

2) Tabel Kategori_buku

(Id_Kategori_Buku, nama_kategori_buku)

Tabel ini sudah dalam bentuk normal ketiga.

3) Tabel_buku

(Id_Buku, Id_kategori_buku, judul, pengarang, tahun_terbit, tanggal_masuk, harga, denda)

Tabel buku ini sudah dalam bentuk 1NF, tetapi belum masuk dalam 2NF. Perhatikan bahwa primary key pada tabel ini adalah id_buku dan id_kategori_buku. Berdasarkan persyaratan bahwa seluruh atribut bukan kunci harus bergantung sepenuhnya pada atribut kunci, dalam hal ini harus bergantung sepenuhnya pada id_buku dan

id_kategori_buku. Pada kenyataannya, nilai pada atribut bukan kunci hanya bergantung kepada id_buku saja, sehingga dikatakan bahwa atribut bukan kunci hanya tidak bergantung sepenuhnya pada atribut kunci (ketergantungan parsial). Menyikapi hal ini, maka tabel buku seharusnya dipecah menjadi dua tabel, yaitu:

- Buku1 (id_buku, judul, pengarang, tahun_terbit, tanggal_masuk, harga, denda)
- Buku2 (id_buku, id_kategori_buku)

Namun dalam implementasinya, pada kasus ini tabel buku tidak harus dipecah menjadi dua tabel seperti ditunjukkan di atas. Dengan catatan harus diberi batasan (*constraint*). Maksud dari pemberian batasan ini adalah untuk mengantisipasi terjadinya inkonsistensi data akibat proses manipulasi. Untuk memperjelas perhatikan ilustrasi di bawah ini:

Tabel Kategori_buku

Id_kategori_buku	Nama_kategori_buku
CAT-001	Novel
CAT-002	Komik
CAT-003	Kamus

Tabel Buku

Id_buku	Id_kategori_buku	Judul	Pengarang	Tahun Terbit	Tanggal Masuk	Harga	Denda
BK-001	CAT-001	Gajah Tak Pernah Lupa	Agatha Christie	1987	01-12-2012	75.000	1.500
BK-002	CAT-002	Kungfu Boy	Hiroshi Kamagawa	1990	01-12-2012	25.000	750
BK-003	CAT-003	Twin	Tassuya Hirota	1993	01-12-2012	25.000	750

Perhatikan bahwa tabel kategori_buku disebut sebagai tabel induk sedangkan tabel buku merupakan tabel anak. Di mana nilai pada kolom id_kategori_buku pada tabel buku nilainya merujuk pada nilai id_kategori_buku pada tabel kategori_buku sebagai tanda bahwa kedua tabel ini berelasi. Sekarang mari kita ambil dua kasus dengan dua operasi manipulasi data yang berbeda. Perhatikan pada buku dengan kode BK-002, memiliki id_kategori_buku CAT-002, yang pada tabel kategori_buku merupakan Komik. Ketika pada tabel induk, nilai Id_kategori_buku pada Komik kita ganti nilainya dengan “CAT-004”, seharusnya nilai id_kategori_buku pada tabel buku juga

berganti menjadi “CAT-004”, sehingga relasi antar kedua tabel tetap terjaga serta data pada tabel buku yang memiliki id_kategori_buku tetap konsisten dan tidak kehilangan referensi terhadap tabel kategori_buku. Kasus lain adalah pada proses penghapusan, ketika record kategori_buku dengan id_kategori_buku “CAT-002” dihapus apa yang akan terjadi? Seluruh record pada tabel buku yang id_kategori_buku bernilai “CAT-002” akan kehilangan referensi pada tabel kategori_buku. Untuk itu proses penghapusan record pada tabel kategori_buku atau tabel induk seharusnya tidak diijinkan ketika atribut kunci pada record tersebut terelasi dengan record tertentu pada tabel buku/ tabel anak.

Batasan (*constraint*) yang dapat diberikan untuk mengantisipasi kasus-kasus di atas adalah bahwa pada proses update diperbolehkan dan pada tabel anak atribut yang berelasi juga ikut terupdate serta proses penghapusan record pada tabel induk tidak diperbolehkan. Atau dalam bahasa *script SQL* adalah sebagai berikut:

On update cascade on delete restrict;

Perintah di atas biasa ditambahkan ketika kita membuat primary key pada tabel anak. Tabel di atas sudah dalam bentuk 3NF, di mana tidak terdapat ketergantungan transitif.

4) Tabel_Anggota

(Id_Anggota, nama_anggota, no_identitas, tanggal_lahir, alamat, telp)

Tabel anggota sudah dalam bentuk normal ketiga (3NF).

5) Tabel_peminjaman

(Id Peminjaman, id anggota, Id Buku, tanggal_pinjam, tanggal_kembali, lama_keterlambatan, denda)

Untuk menganalisa tabel ini, perhatikan contoh data pada tabel di bawah ini:

Id_peminjaman	Id_anggota	Id_buku	Tanggal_Pinjam	Tanggal_Kembali	Lama_Keterlambatan	Denda
1	1	BK-002	01-12-2012	15-12-2012	1	1500
1	1	BK-003	01-12-2012	15-12-2012	1	1500

Asumsi : maksimal peminjaman 14 hari dihitung mulai tanggal pinjam.

Tabel di atas bermakna logis bahwa terjadi transaksi peminjaman dengan id_peminjaman “1” oleh anggota dengan id_anggota “1” pada tanggal “01-12-2012”

dan dikembalikan pada tanggal “15-12-2012” sehingga terjadi keterlambatan pengembalian selama 1 hari sehingga total denda adalah 15.000. Sedangkan buku yang dipinjam adalah buku dengan id_buku:

- “BK-002”
- “BK-003”

Tabel di atas mengandung grup berulang, yang mana keterangan mengenai id_peminjaman, id_anggota, tanggal_pinjam, tanggal_kembali, lama_keterlambatan serta denda akan diulang sebanyak jumlah buku yang dipinjam. Hal ini tidak memenuhi kaidah bentuk normal pertama. Untuk dapat memenuhi kaidah bentuk normal pertama, maka tabel di atas harus dipecah, di mana bagian yang berulang dipisahkan menjadi tabel tersendiri sehingga akan didapat dua tabel seperti di bawah ini:

a) Peminjaman

(Id Peminjaman, id anggota, tanggal_pinjam, tanggal_kembali, lama_keterlambatan, denda)

b) Item_peminjaman

(id_peminjaman, id buku)

Tabel Item_peminjaman merupakan tabel hasil dekomposisi/ pemecahan tabel peminjaman. Dengan id_peminjaman pada tabel ini merujuk pada id_peminjaman pada tabel peminjaman. Kedua tabel di atas sudah dalam bentuk normal pertama. Selanjutnya akan kita analisa, apakah tabel-tabel di atas sudah masuk dalam normal kedua. Tabel item_peminjaman sudah pasti masuk dalam bentuk normal kedua dan ketiga, sebab seluruh atribut yang ada merupakan atribut kunci, sehingga kemungkinan terjadi ketergantungan sebagian serta ketergantungan transitif tidak ada.

Bagaimana dengan tabel peminjaman? Tabel peminjaman sudah dalam bentuk normal kedua, karena atribut bukan kunci pasti bergantung sepenuhnya pada atribut kunci yaitu id_peminjaman dan id_anggota. Jika sudah dalam bentuk normal kedua, bagaimana dengan bentuk normal ketiga? Adakah ketergantungan transitif atau ketergantungan atribut bukan kunci terhadap atribut bukan kunci lainnya?

Perhatikan lama_keterlambatan dan denda. Nilai dari kedua atribut ini tidak bergantung kepada id_peminjaman dan id_anggota, tetapi pada tanggal_pinjam dan

tanggal_kembali. Untuk itu tabel peminjaman perlu dipecah lagi. Berikut adalah tabel-tabel hasil normalisasi tabel peminjaman:

- Peminjaman

(Id_Peminjaman, id_anggota, tanggal_pinjam, tanggal_kembali)

- Item_peminjaman

(id_peminjaman, id_buku)

Tabel ini hasil dekomposisi sebagai akibat adanya grup berulang.

- Denda

(Id_Peminjaman, lama_keterlambatan, denda)

Tabel ini hasil dekomposisi sebagai akibat adanya ketergantungan transitif.

2.2.2 Perancangan Basisdata

Pada perancangan basis data, akan dilakukan penentuan struktur tabel-tabel berdasarkan hasil normalisasi tabel-tabel konversi ERD. Pemilihan tipe data yang dipergunakan pada *field-field* dapat dilakukan sesuai dengan kebutuhan. Tentu saja dengan pertimbangan-pertimbangan tertentu. Misalnya untuk tipe data *field* id_kategori_buku dipergunakan tipe data *varchar* dengan ukuran 6, dengan aturan 3 digit pertama dipergunakan untuk inisial, dan 2 digit terakhir urutan jumlah kode yang sudah masukkan. Sedangkan digit ke-4 akan diisi “-“. Contoh:

- Komik → KMK-01
- Kamus → KMS-02
- Novel → NVL-03
- Majalah → MJL-04



Berikut ini adalah kamus data yang mendefinisikan struktur tabel serta tipe data dan keterangan lain:

1) Tabel Pengelola

Field	Tipe Data (Ukuran)	Keterangan	Tabel Referensi	Constraint/ Batasan
Id_pengelola	Integer	Primary key, AUTO_INCREMENT		
Nama_pengelola	Varchar (30)			
Username	Varchar(25)			
Password	Varchar(100)			

2) Tabel Kategori_Buku

Field	Tipe Data (Ukuran)	Keterangan	Tabel Referensi	Constraint/ Batasan
Id_kategori_buku	Varchar (6)	Primary Key		
Nama_kategori_buku	Varchar(25)			

3) Tabel Buku

Field	Tipe Data (Ukuran)	Keterangan	Tabel Referensi	Constraint/ Batasan
Id_buku	Varchar(10)	Primary key		
Id_kategori_buku	Varchar (6)	Foreign key	Kategori_buku	<i>On update cascade</i> <i>On delete restrict</i>
Judul	Varchar(100)			
Pengarang	Varchar(30)			
Tahun_terbit	Varchar(4)			
Tanggal_masuk	Date(8)			
Harga	Double			
Denda	Double			

Pemberian *constraint* pada *foreign key* id_kategori_buku bermakna logis bahwa ketika id_kategori_buku pada tabel kategori buku diubah nilainya, maka nilai id_kategori_buku tersebut juga akan berubah pada tabel buku, sedangkan proses penghapusan record tertentu pada tabel kategori_buku tidak diperbolehkan jika ternyata record tersebut berelasi dengan record tertentu pada tabel buku.

4) Tabel Anggota

Field	Tipe Data (Ukuran)	Keterangan	Tabel Referensi	Constraint/ Batasan
Id_Anggota	Integer	Primary key, AUTO_INCREMENT		
Nama_anggota	Varchar (30)			
No_Identitas	Varchar(25)			
Alamat	Varchar(100)			
Telp	Varchar(25)			
Tanggal_lahir	Date(8)			

Tipe data untuk id_anggota dipergunakan integer dengan nilai AUTO_INCREMENT, artinya nilainya akan bertambah dan teurut dengan sendirinya. Yang menjadi pertimbangan untuk hal ini adalah bahwa untuk mempermudah proses, di mana pengelola tidak harus mengentrikan secara manual lagi nilai id_anggota. Sistem secara otomatis akan memberikan nilai id_anggota. Untuk pemilihan tipe data ini bebas dilakukan, asalkan bisa menjawab kebutuhan.

5) Tabel Peminjaman

Field	Tipe Data (Ukuran)	Keterangan	Tabel Referensi	Constraint/ Batasan
Id_peminjaman	Integer	Primary key, AUTO_INCREMENT		
Id_anggota	Integer	Foreign Key	Anggota	<i>On update cascade On delete restrict</i>
Tanggal_pinjam	Date(8)			
Tanggal_kembali	Date(8)			

6) Tabel Item_Pengembalian

Field	Tipe Data (Ukuran)	Keterangan	Tabel Referensi	Constraint/ Batasan
Id_peminjaman	Integer	Foreign key	Peminjaman	<i>On update cascade On delete cascade</i>
Id_buku	Varchar(10)	Foreign Key	Buku	<i>On update cascade On delete restrict</i>

Pada tabel ini terdapat dua *foreign key*, yaitu id_peminjaman dan id_buku. Keduanya memiliki *constraint* berbeda. Pada *field* id_peminjaman, ketika salah satu record pada tabel peminjaman dihapus, maka record-record pada tabel item_peminjaman yang berelasi dengan record yang dihapus tersebut juga akan terhapus. Ketika sebuah transaksi peminjaman dihapus, maka seluruh informasi buku dipinjam pun pasti akan ikut terhapus. Pemberian *constraint*

7) Tabel Denda

Field	Tipe Data (Ukuran)	Keterangan	Tabel Referensi	Constraint/ Batasan
Id_peminjaman	Integer	Foreign key	Peminjaman	<i>On update cascade On delete cascade</i>
Denda	Double			
Lama_keterlambatan	Integer(2)			

Penggunaan Power Designer Untuk Perancangan Interface

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Power Designer untuk membuat perancangan interface bagi sistem yang akan dikembangkan.

Pertemuan ini akan menjelaskan secara singkat tentang pengertian interface, menentukan kebutuhan sistem, menentukan kebutuhan interface, symbol yang digunakan, menggambar arsitektur sistem dan menggambar rancangan interface.

6.1 Pengertian Interface

Antarmuka (Interface) merupakan mekanisme komunikasi antara pengguna (user) dengan sistem. Antarmuka (Interface) dapat menerima informasi dari pengguna (user) dan memberikan informasi kepada pengguna (user) untuk membantu mengarahkan alur penelusuran masalah sampai ditemukan suatu solusi.

Interface, berfungsi untuk menginput pengetahuan baru ke dalam basis pengetahuan sistem pakar (ES), menampilkan penjelasan sistem dan memberikan panduan pemakaian sistem secara menyeluruh / step by step sehingga pengguna mengerti apa yang akan dilakukan terhadap suatu sistem. Yang terpenting adalah kemudahan dalam memakai / menjalankan sistem, interaktif, komunikatif, sedangkan kesulitan dalam mengembangkan / membangun suatu program jangan terlalu diperlihatkan.

Interface yang ada untuk berbagai sistem, dan menyediakan cara :

- Input, memungkinkan pengguna untuk memanipulasi sistem
- Output, memungkinkan sistem untuk menunjukkan efek manipulasi pengguna.

1. Tujuan Interface

Tujuan sebuah interface adalah mengkomunikasikan fitur-fitur sistem yang tersedia agar user mengerti dan dapat menggunakan sistem tersebut. Dalam hal ini penggunaan bahasa amat efektif untuk membantu pengertian, karena bahasa merupakan alat tertua (barangkali kedua tertua setelah gesture) yang dipakai orang untuk berkomunikasi sehari-harinya. Praktis, semua pengguna komputer dan Internet (kecuali mungkin anak kecil yang memakai komputer untuk belajar membaca) dapat mengerti tulisan.

Meski pada umumnya panduan interface menyarankan agar ikon tidak diberi tulisan supaya tetap mandiri dari bahasa, namun elemen interface lain seperti teks pada tombol, caption window, atau teks-teks singkat di sebelah kotak input dan tombol pilihan semua menggunakan bahasa. Tanpa bahasa pun kadang ikon bisa tidak jelas maknanya, sebab tidak semua lambang ikon bisa bersifat universal.

Meskipun penting, namun sayangnya kadang penggunaan bahasa, seperti pemilihan istilah, sering sekali dianggap kurang begitu penting. Terlebih dari itu dalam dunia desain situs Web yang serba grafis, bahasa sering menjadi sesuatu yang nomor dua ketimbang elemen-elemen interface lainnya.

Tujuan sebuah interface adalah mengkomunikasikan fitur-fitur sistem yang tersedia agar user mengerti dan dapat menggunakan sistem tersebut. Dalam hal ini penggunaan bahasa amat efektif untuk membantu pengertian, karena bahasa merupakan alat tertua (barangkali kedua tertua setelah gesture) yang dipakai orang untuk berkomunikasi sehari-harinya. Praktis, semua pengguna komputer dan Internet (kecuali

mungkin anak kecil yang memakai komputer untuk belajar membaca) dapat mengerti tulisan. Interface ada dua jenis, yaitu :

Interface ada dua jenis, yaitu : 1) Graphical Interface : Menggunakan unsur-unsur multimedia (seperti gambar, suara, video) untuk berinteraksi dengan pengguna, 2) Text-Based : Menggunakan syntax/rumus yang sudah ditentukan untuk memberikan perintah.

2. Graphical Interface

Ada 5 tipe utama interaksi untuk interaction:

- a. Direct manipulation – pengoperasian secara langsung : interaksi langsung dengan objek pada layar. Misalnya delete file dengan memasukkannya ke trash. Contoh: Video games. Kelebihan : Waktu pembelajaran sangat singkat, feedback langsung diberikan pada tiap aksi sehingga kesalahan terdeteksi dan diperbaiki dengan cepat. Kekurangan : Interface tipe ini rumit dan memerlukan banyak fasilitas pada sistem komputer, cocok untuk penggambaran secara visual untuk satu operasi atau objek.
- b. Menu selection – pilihan berbentuk menu : Memilih perintah dari daftar yang disediakan. Misalnya saat click kanan dan memilih aksi yang dikehendaki. Kelebihan : tidak perlu ingat nama perintah. Pengetikan minimal. Kesalahan rendah. Kekurangan : Tidak ada logika AND atau OR. Perlu ada struktur menu jika banyak pilihan. Menu dianggap lambat oleh expert dibanding *command language*.
- c. Form fill-in – pengisian form : Mengisi area-area pada form. Contoh : Stock control. Kelebihan : Masukan data yang sederhana. Mudah dipelajari. Kekurangan : Memerlukan banyak tempat di layar. Harus menyesuaikan dengan form manual dan kebiasaan.
- d. Command language – perintah tertulis : Menuliskan perintah yang sudah ditentukan pada program. Contoh: operating system. Kelebihan : Perintah diketikkan langsung pada system. Misal UNIX, DOS command. Bisa diterapkan pada terminal yang murah. Kombinasi perintah bisa dilakukan. Misal copy file dan rename nama file. Kekurangan : Perintah harus dipelajari dan diingat cara penggunaannya, tidak cocok untuk biasa. Kesalahan pakai perintah sering terjadi. Perlu ada sistem pemulihan kesalahan. Kemampuan mengetik perlu.
- e. Natural language – perintah dengan bahasa alami : Menggunakan bahasa alami untuk mendapatkan hasil. Contoh: search engine di Internet. Kelebihan: Perintah dalam bentuk bahasa alami, dengan kosa kata yang terbatas (singkat), misalnya kata kunci yang kita tentukan untuk dicari oleh search engine. Ada kebebasan menggunakan kata-kata. Kekurangan: Tidak semua sistem cocok gunakan ini. Jika digunakan maka akan memerlukan banyak pengetikan.

6.2 Menentukan Kebutuhan Sistem dan Kebutuhan Interface

Hasil dokumentasi dari tahap analisis kelemahan system digunakan untuk rekomendasi fungsionalitas apa saja yang bias dilakukan system baru. Fungsionalitas inilah yang mencerminkan kebutuhan system.

System requirement (Kebutuhan Sistem)

Tujuan dari fase analisis adalah memahami dengan sebenar-benarnya kebutuhan dari sistem baru dan mengembangkan sebuah sistem yang memadai requirement tersebut-atau memutuskan bahwa sebenarnya pengembangan sistem baru tidak dibutuhkan. Penentuan kebutuhan sistem merupakan langkah yang paling crucial dalam tahapan SDLC. Kebutuhan Sistem bisa diartikan sebagai berikut:

- a. Pernyataan tentang apa yang harus dikerjakan oleh sistem
- b. Pernyataan tentang karakteristik yang harus dimiliki sistem

Tipe-tipe Kebutuhan Sistem

Untuk mempermudah system analis menentukan keseluruhan requirement secara lengkap, maka analis membagi kebutuhan system ke dalam 2 jenis

1. Kebutuhan Fungsional (*Functional requirement*)
Kebutuhan fungsional adalah jenis kebutuhan yang berisi proses-proses apa saja yang nantinya dilakukan oleh system. Kebutuhan fungsional juga berisi informasi-informasi apa saja yang harus ada dan dihasilkan oleh sistem.
2. Kebutuhan Non fungsional (Nonfunctional equirements).
Requirement jenis ini adalah tipe requirement yang berisi properti perilaku yang dimiliki oleh sistem, meliputi:

6.3. Praktek

6.3.1 Menggambar Arsitektur Sistem

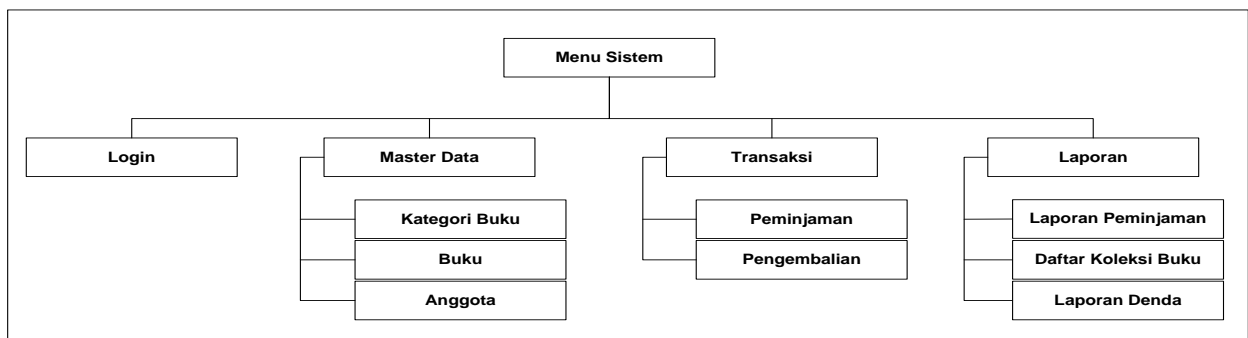
6.3.2 Menggambar Rancangan Interface

Kembali ke *Data Flow Diagram* (DFD), kita sudah mendapatkan proses-proses sebagai berikut:

1. Login
2. Olah Master Data
 - 2.1 Kategori Buku
 - 2.2 Buku
 - 2.3 Anggota

3. Transaksi
 - 3.1 Peminjaman
 - 3.2 Pengembalian
4. Laporan
 - 4.1 Laporan Peminjaman
 - 4.2 Daftar Buku
 - 4.3 Laporan Denda
5. Pencarian

Dari proses-proses di atas, maka dapat dibuat sebuah rancangan menu seperti ditunjukkan oleh gambar 9 seperti di bawah ini. Pada gambar 10 ditampilkan rancangan *user interface* untuk form login. Ketika login, pengelola akan memasukkan username dan password, kemudian sistem akan memverifikasi apakah username dan password terdaftar atau tidak. Setelah berhasil login, akan ditampilkan form menu.



Gambar 9 Rancangan Menu

Username

Password

Gambar 10 Rancangan Form Login

Sistem mempunyai tiga menu utama, yaitu Olah Data Master, Transaksi dan Laporan. Menu Transaksi dan Laporan dapat dijalankan setelah data-data yang dibutuhkan untuk

melakukan proses-proses pada menu-menu ini sudah didefinisikan terlebih dahulu. Pada gambar 11 ditampilkan rancangan form olah master data kategori buku. Pada gambar 12 ditampilkan rancangan form olah master data buku. Pada rancangan form ini terdapat *combo box* yang berisikan data kategori buku, yang menunjukkan adanya relasi antara buku dan kategori buku. Di mana sebuah buku pasti berelasi dengan kategori buku tertentu. Untuk rancangan form olah master data anggota ditunjukkan oleh gambar 13.

<u>Id kategori buku</u>	<u>Nama kategori buku</u>
CAT-001	Novel
CAT-002	Komik
CAT-003	Kamus

Gambar 11 Rancangan form olah master data kategori buku

<u>Id buku</u>	<u>Id kategori buku</u>	<u>Judul</u>	<u>Pengarang</u>	<u>Tahun Terbit</u>	<u>Tanggal Masuk</u>	<u>Harga</u>	<u>Denda</u>
BK-001	CAT-001	Gajah Tak Pernah Lupa	Agatha Christie	1987	01-12-2012	75.000	1.500
BK-002	CAT-002	Kungfu Boy	Hiroshi Kamagawa	1990	01-12-2012	25.000	750
BK-003	CAT-003	Twin	Tassuya Hirota	1993	01-12-2012	25.000	750

Gambar 12 Rancangan form olah master data kategori buku

<u>Id Anggota</u>	<u>No Identitas</u>	<u>Nama Anggota</u>	<u>Tanggal Lahir</u>	<u>Alamat</u>	<u>Telp</u>
ANG-001	983124001	Noel	16-06-2005	Jln Perdana II No.216C Maguwoharjo	085292951855
ANG-002	123456789	Joe Satriani	12-05-2007	Krg Ploso Maguwoharjo	085292951856
ANG-003	987654321	Archiless	05-05-1980	Jln. Paingan II No.123 Maguwoharjo	0817251547

Gambar 13 Rancangan form olah master data anggota

Untuk rancangan form peminjaman dan pengembalian ditunjukkan pada gambar 14 dan gambar 15. Pada proses peminjaman, pengelola akan mengentrikan Id Anggota serta id buku

yang akan dipinjam. Daftar buku yang dipinjam akan ditampilkan pada tabel yang terdapat pada form.

This form is used for borrowing books. It contains input fields for transaction details and a table of available books.

Input Fields:

- No Transaksi
- Id Anggota
- Nama
- Id Buku
- Judul
- Pengarang

Buttons: Transaksi Baru, Edit Transaksi, Batal Transaksi, Tambah, Hapus.

Kode Buku	Judul
BK-002	Kungfu Boy
BK-003	Twin

Gambar 14 Rancangan form peminjaman

This form is used for returning books. It includes fields for transaction details, return date, and a table of books.

Input Fields:

- No Transaksi
- Id Anggota
- Nama
- Tanggal Pinjam
- Terlambat (Ya/Tidak)
- Lama Keterlambatan
- Denda

Buttons: Buka Transaksi, Simpan.

Kode Buku	Judul
BK-002	Kungfu Boy
BK-003	Twin

Gambar 15 Rancangan form pengembalian

Pada proses pembuatan laporan, untuk rancangan form laporan peminjaman seperti dapat dilihat pada gambar 16 serta rancangan form laporan denda pada gambar 18, pengelola harus menentukan tanggal laporan. Sedangkan pada rancangan form laporan daftar koleksi buku dapat dilihat pada gambar 17. Pada modul workshop ini, jenis laporan yang dibuat hanya seperti yang ditunjukkan pada gambar saja. Untuk variasi jenis laporan dapat dikembangkan sesuai kebutuhan. Pada prinsipnya proses pembuatan laporan adalah proses mengkueri data sesuai dengan kondisi yang diberikan. Untuk rancangan form pencarian buku oleh pengelola dan anggota dapat dilihat pada gambar 19.

This form is used to generate a borrowing report. It includes a date selection field and two buttons.

Fields: Tanggal Laporan (01-12-2012)

Buttons: Cetak Laporan, Batal

Gambar 16 Rancangan form laporan peminjaman

This form is used to generate a book collection report. It includes a selection field and two buttons.

Fields: Pilihan (Urut Id Buku)

Buttons: Cetak Laporan, Batal

Gambar 17 Rancangan form daftar koleksi buku

Tanggal Laporan

01-12-2012

Cetak Laporan

Batal

Gambar 18 Rancangan form laporan denda

Opsi Pencarian

☒ Judul
 ☐ Id Buku
 ☐ Pengarang

Cari

Kata Kunci

Keluar

<u>Id buku</u>	<u>Id kategori buku</u>	<u>Judul</u>	<u>Pengarang</u>	<u>Tahun Terbit</u>	<u>Tanggal Masuk</u>	<u>Harga</u>	<u>Denda</u>
BK-001	CAT-001	Gajah Tak Pernah Lupa	Agatha Christie	1987	01-12-2012	75.000	1.500
BK-002	CAT-002	Kungfu Boy	Hiroshi Kamagawa	1990	01-12-2012	25.000	750
BK-003	CAT-003	Twin	Tassuva Hirota	1993	01-12-2012	25.000	750

Gambar 19 Rancangan form pencarian buku

Penggunaan Star UML Untuk Membuat Rancangan Use Case Secara Umum

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Star UML untuk membuat Use Case secara umum.

Pertemuan ini akan menjelaskan secara singkat tentang pengantar Star UML, pengertian use case, Simbol-simbol yang digunakan dan proses menggambar Use Case.

7.1 Pengantar UML

7.1.1 Pendahuluan

Unified Modeling Language (UML) adalah sebuah “bahasa” yang telah menjadi standar dalam industry untuk visualisasi, merancang dan mendokumentasikan system piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah system.

Dengan menggunakan UML dapat membuat model untuk semua jenis aplikasi piranti lunak dimana aplikasi tersebut dapat berjalan pada piranti keras, system operasi dan jaringan apapun serta ditulis dalam bahasa pemrograman apapun. Berhubung UML menggunakan class dan operation dalam konsep dasarnya, maka lebih cocok untuk penulisan piranti lunak dalam bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi procedural dalam VB atau C.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan syntax/semantic. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (Object-Oriented Design), Jim Rumbaugh OMT (Object Modeling Technique), dan Ivar Jacobson OOSE (Object-Oriented Software Engineering).

7.1.2 Diagram Dasar dalam UML

a. Model Use Case Diagram

Use case diagram secara grafis menggambarkan interaksi antara system, system eksternal, dan pengguna. Dengan kata lain Use Case diagram secara grafis mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna (user) mengharapkan interaksi dengan sistem itu. Use Case secara naratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi.

b. Diagram Struktur Statis

UML menawarkan dua diagram untuk memodelkan struktur statis sistem informasi, yaitu:

- 1) Class Diagram: menggambarkan struktur object sistem. Diagram ini menunjukkan class object yang menyusun sistem dan juga hubungan antara class object tersebut.

- 2) Object Diagram: serupa dengan class diagram, tetapi object diagram memodelkan instance object actual dengan menunjukkan nilai-nilai saat ini dari atribut instance. Object Diagram menyajikan "snapshot/potret" tentang object sistem pada point waktu tertentu. Dengan ini tidak digunakan sesering Class Diagram, tetapi saat digunakan dapat membantu seorang developer memahami struktur system secara lebih baik.

c. Diagram Iterasi

Diagram interaksi memodelkan sebuah interaksi, terdiri dari satu set objek, hubungan-hubungannya, dan pesan yang terkirim di antara objek. Model diagram ini memodelkan behavior (kelakuan) system yang dinamis dan UML memiliki dua diagram untuk tujuan ini, yaitu:

- 1) Diagram rangkaian/Sequence Diagram: secara grafis menggambarkan bagaimana objek berinteraksi dengan sama lain melalui pesan pada sekuensi sebuah use case atau operasi. Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima di antara objek dan dalam sekuensi atau timing apa.
- 2) Diagram kolaborasi/Collaboration Diagram: serupa dengan diagram rangkaian/sekuensi, tetapi tidak focus pada timing atau sekuensi pesan. Diagram ini justru menggambarkan interaksi (atau kolaborasi) antara objek dalam sebuah format jaringan.

Diagram rangkaian maupun diagram kolaborasi merupakan isomorphic artinya kita dapat mengubah dari satu diagram ke diagram lain.

d. Diagram State/State Diagram

UML memiliki sebuah diagram untuk memodelkan behavior objek khusus yang kompleks (statechart) dan sebuah diagram untuk memodelkan behavior dari sebuah use case atau sebuah metode, yaitu:

- 1) Diagram statechart: digunakan untuk memodelkan behavior objek khusus yang dinamis. Diagram ini mengilustrasikan siklus hidup objek berbagai keadaan yang dapat diasumsikan oleh objek dan event-event (kejadian) yang menyebabkan objek beralih dari satu state ke state lain.
- 2) Diagram aktivitas/Activity Diagram: secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun use case. Activity diagram dapat juga digunakan untuk memodelkan action yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari action tersebut.

e. Diagram Implementasi

Diagram implementasi juga memodelkan struktur system informasi, yaitu:

- 1) Diagram komponen/Component Diagram: digunakan untuk menggambarkan organisasi dan ketergantungan komponen-komponen software system. Komponen diagram dapat digunakan untuk menunjukkan bagaimana kode program dibagi menjadi modul-modul (atau komponen).
- 2) Diagram penguraian/Deployment: digunakan untuk mendeskripsikan arsitektur fisik dalam istilah "node" untuk hardware dan software dalam sistem. Diagram ini menggambarkan konfigurasi komponen-komponen software real-time, prosesor, dan peralatan yang membentuk arsitektur sistem.

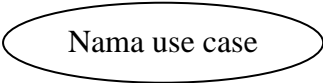
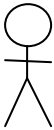

7.2 Pengertian Use Case

Dalam membuat sebuah system, langkah awal yang perlu dilakukan adalah menentukan kebutuhan. Terdapat dua jenis kebutuhan, yaitu kebutuhan fungsional dan kebutuhan non fungsional. Kebutuhan fungsional adalah kebutuhan pengguna dan stakeholder sehari-hari yang akan dimiliki oleh system, dimana kebutuhan ini akan digunakan oleh pengguna dan stakeholder. Kebutuhan non fungsional adalah kebutuhan yang memperhatikan hal-hal berikut yaitu performansi, kemudahan dalam menggunakan system, kehandalan system, keamanan system, keuangan, legalitas, dan operasional.

Kebutuhan fungsional akan digambarkan melalui sebuah diagram yang dinamakan diagram use case. Use Case Diagram atau diagram use case merupakan pemodelan untuk menggambarkan kelakuan (behavior) system yang akan dibuat. Diagram use case mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan system yang akan dibuat. Dengan pengertian yang cepat, diagram use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah system dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Terdapat beberapa symbol dalam menggambarkandiagram use case, yaitu use case, actor dan relasi. Hal yang perlu diingatmengenai diagram use case adalah diagram use case bukan menggambarkan tampilan antarmuka (user interface), arsitektur dari system, kebutuhan nonfungsional, dan tujuan performansi. Untuk penamaan use case adalah nama didefinisikan sesimpel mungkin, dapat dipahami dan menggunakan kata kerja.

7.3 Mengenal Simbol-simbol Pada Use Case Secara Umum

Berikut ini adalah simbol-simbol yang ada pada diagram use case secara umum:

Simbol	Deskripsi
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama use case.
Aktor/Actor  Nama aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi/Association 	Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan aktor.

7.4 Menemukan Aktor

Pekerjaan awal dalam mendesain sistem adalah menemukan aktor, menemukan fungsionalitas dan membatasi sistem yang akan dibuat. Pembatasan sistem ini penting

untuk menemukan aktor. Dari sinilah kita akan menentukan apakah sesuatu itu adalah aktor dan apakah aktor tersebut akan berbentuk orang atau sistem lain. Aktor adalah segala hal diluar sistem yang akan menggunakan sistem tersebut untuk melakukan sesuatu.

Dilihat dari pentingnya, yang perlu dipahami adalah memisahkan sistem yang akan dibangun dengan yang ada di luar sistem. Oleh karenanya, kita perlu membatasi sistem yang akan dibuat dan segala sesuatu yang berinteraksi dengan sistem adalah aktor. Cara mudah menemukan aktor adalah dengan bertanya hal-hal berikut:

- SIAPA yang akan menggunakan sistem?
- APAKAH sistem tersebut akan memberikan NILAI bagi aktor?

Namun, yang perlu diingat adalah tidak semua aktor adalah manusia, bisa saja sistem lain yang berinteraksi dengan sistem yang dibuat. Untuk menemukan sistem lain sebagai aktor, hal-hal di bawah ini bisa menjadi pertimbangan. Jika anda bergantung pada sistem lain untuk melakukan sesuatu, maka sistem lain itu adalah aktor. Jika sistem lain itu meminta (request) informasi dari sistem anda, maka sistem lain itu adalah aktor. Untuk penamaan aktor diberi nama sesuai dengan PERANnya. Contoh, pada sistem pencatatan di Supermarket.

Pertanyaan	Analisis
Siapa sajakah yang berinteraksi dengan sistem pencatatan penjualan di supermarket?	<ul style="list-style-type: none"> - Bagian yang akan mencatat penjualan barang - Bagian yang ingin tahu berapa besar keuntungan yang didapatkan - Bagian yang ingin tahu berapa banyak produk yang berkurang
Peran apa saja yang terlibat?	Kasir, Manajer, bagian gudang
Nilai apa sajakah yang akan diberikan sistem kepada aktor	<p>Nilai bagi kasir:</p> <ul style="list-style-type: none"> - Ia akan mendapat struk belanja - Lama aktivitas kerja akan terekam ke dalam sistem <p>Nilai bagi manajer:</p> <ul style="list-style-type: none"> - Ia perlu mengetahui laporan keuntungan dalam rentang waktu tertentu <p>Nilai bagi bagian gudang:</p> <ul style="list-style-type: none"> - Ia perlu mengetahui produk apa saja yang berkurang
Apakah sistem pencatatan penjualan bergantung pada sesuatu?	<p>Printer:</p> <p>Untuk mencetak struk Mesin debit ATM:</p> <p>Untuk menraik sejumlah uang pada account seseorang</p>

Jadi, aktor yang ada pada sistem pencatatan penjualan supermarket adalah kasir, manajer, bagian gudang, printer dan ATM

Jika anda perhatikan dari tabel di atas, pertanyaan yang akan muncul adalah mengapa struk belanja menjadi nilai bagi kasir, dan bukannya pelanggan? Struk belanja memang nilai bagi pelanggan, namun yang perlu diingat adalah pelanggan tidak berinteraksi langsung dengan sistem, kasirlah yang berinteraksi langsung dengan sistem. Pelanggan akan mendapatkan nilainya melalui kasir.

Sistem dibangun untuk menyediakan kebutuhan bagi aktor, jika suatu saat nanti stakeholder akan menentukan bahwa sistem pencatatan penjualan akan berinteraksi dengan pelanggan, maka aktor di atas pun tentu saja akan berubah. Inilah yang dimaksud dengan batasan sistem. Stakeholder dan pengguna akan menentukan batasan sistem yang akan dibuat.

7.5 Menemukan Use Case

Jika anda sudah berhasil menemukan aktor, maka untuk menemukan use case akan lebih mudah dilakukan. Sebuah use case harus mendeskripsikan sebuah pekerjaan dimana pekerjaan tersebut akan memberikan nilai yang bermanfaat bagi aktor.

Pengertian ini penting untuk diingat, karena dari hal inilah akan menentukan bahwa sebuah use case tidak akan menjadi terlalu kecil. Use case yang terlalu kecil tidak akan memberikan nilai bagi aktor.

Untuk menemukan use case, mulailah dari sudut pandang aktor, misalnya dengan bertanya:

- Informasi apa sajakah yang akan didapat aktor dari sistem?
- Apakah ada kejadian dari sistem yang perlu diberitahukan kepada aktor?

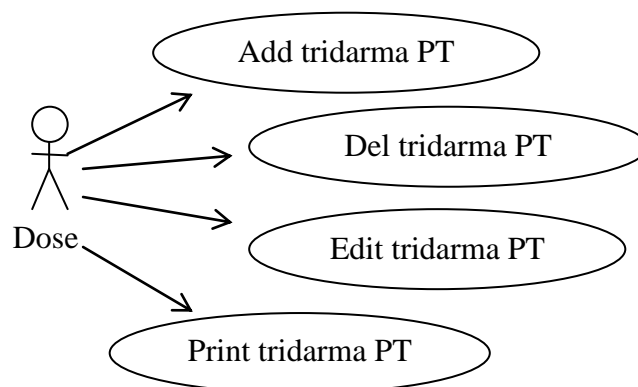
Sedangkan dari sudut pandang sistem, misalnya dengan pertanyaan sebagai berikut:

- Apakah ada informasi yang perlu disimpan atau diambil dari sistem?
- Apakah ada informasi yang harus dimasukkan oleh aktor?

Setiap use case harus dijelaskan alur prosesnya melalui sebuah deskripsi use case (use case description) atau scenario use case. Deskripsi use case berisi:

- Nama use case yaitu penamaan use case yang menggunakan kata kerja
- Deskripsi yaitu penjelasan mengenai tujuan use case dan nilai yang akan didapatkan oleh aktor
- Kondisi sebelum (pre-condition) yaitu kondisi-kondisi yang sudah dipenuhi ketika use case sudah dilaksanakan alur dasar (basic flow) yaitu alur yang menceritakan semua aksi yang dilakukan adalah benar atau proses yang harusnya terjadi.
- Alur alternatif (alternatif flow) yaitu alur yang menceritakan aksi alternatif, yang berbeda dari alur dasar.

Kesalahan yang sering muncul pada diagram use case, seringkali sebuah use case dianggap sebagai sebuah "function" atau item menu. Hal ini adalah salah. Perhatikan contoh pada Gambar 7.1.



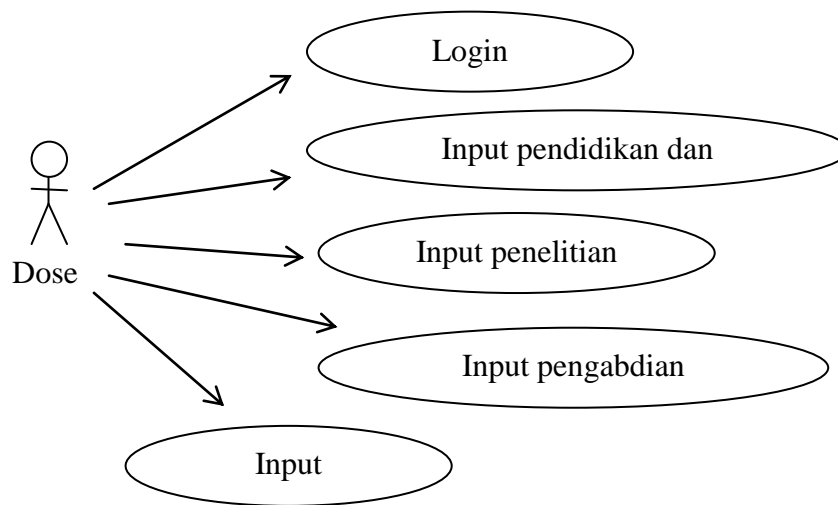
Gambar 7.1 Contoh 1 Use Case Diagram

Use case pada gambar 7.1 menggambarkan mengenai apa yang harus dilakukan oleh sistem yang terdiri dari beberapa proses yaitu menambah, menghapus, merubah dan cetak kegiatan tridarma perguruan tinggi dosen. Sebenarnya diagram tersebut memperlihatkan proses penguraian fungsi-fungsi (functional decomposition) yaitu mengurai proses ke dalam bagian yang lebih kecil. Hal ini adalah salah karena use case di atas tidak memberikan nilai kepada user.

Diagram use case adalah sebuah diagram yang menjelaskan apa yang harus dilakukan oleh sistem pada level konseptual sehingga kita akan memahami apakah keputusan yang diambil oleh sistem adalah benar atau salah. Cobalah bertanya seperti ini: Apakah saya akan menggunakan proses tambah tridarma PT jika saya tidak pernah menginput data kegiatan

tridarma PT, dan semua proses di atas sebenarnya berkaitan dengan melakukan input data tridarma PT.

Apa yang salah dari diagram di atas? Diagram di atas tidak memberikan nilai kepada aktor, atau dengan kata lain jika kita menggambarkan diagram seperti di atas, nilai akan menjadi hilang. Sebuah use case seharusnya dibuat untuk menghasilkan suatu nilai kepada aktor, pada level tertentu jika aktor melakukan input data maka proses tersebut akan memberikan nilai kepada aktor. Tapi jika proses input data saja tidak pernah dilakukan, apakah hal ini akan memberikan nilai? Tentu saja tidak. Oleh karena itu, gambarlah diagram use case yang berfokus pada nilai yang akan diberikan kepada aktor. Untuk itu gambar 7.1 dapat diubah menjadi gambar 7.2 supaya bisa memberikan nilai kepada aktor.



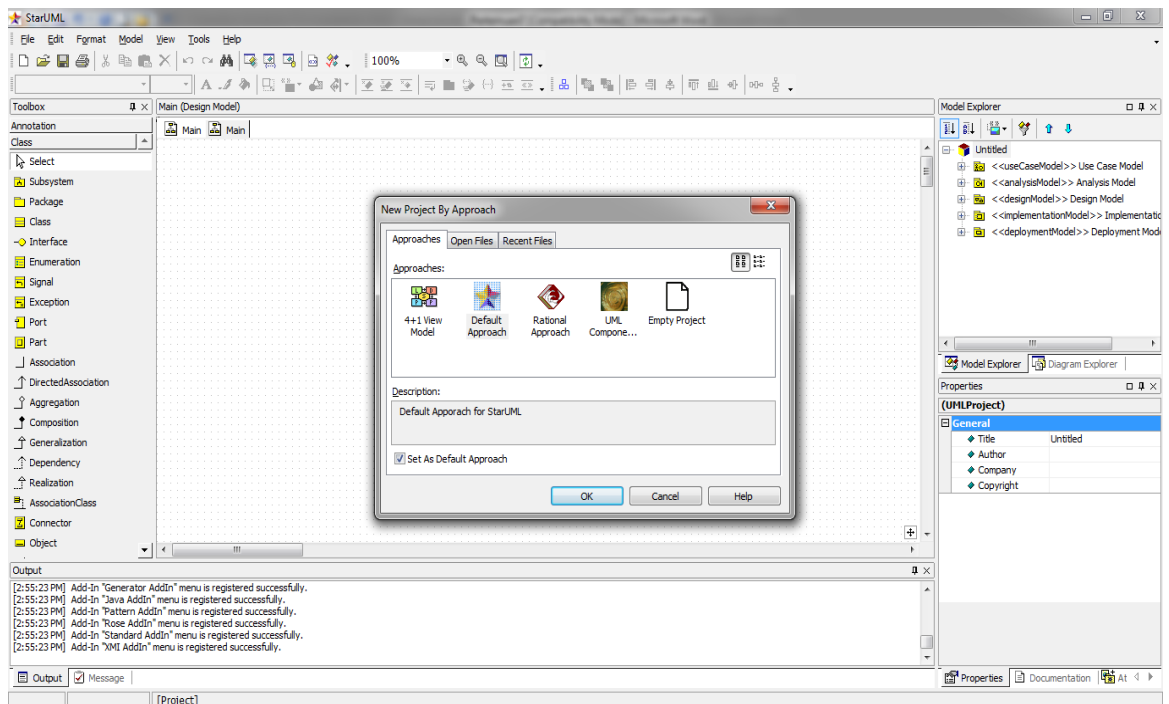
Gambar 7.2 Contoh 2 Use Case Diagram

7.6 Praktek

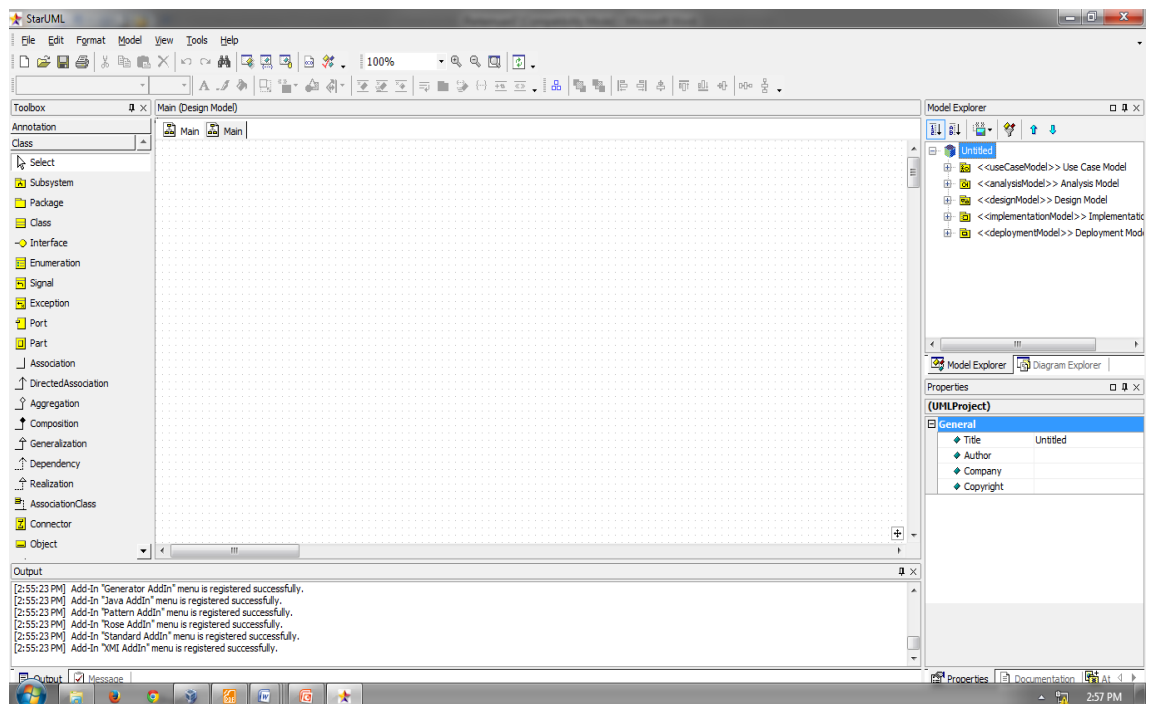
7.6.1 Pengantar Penggunaan Star UML

Untuk menggunakan perangkat lunak Star UML, lakukan langkah-langkah sebagai berikut:

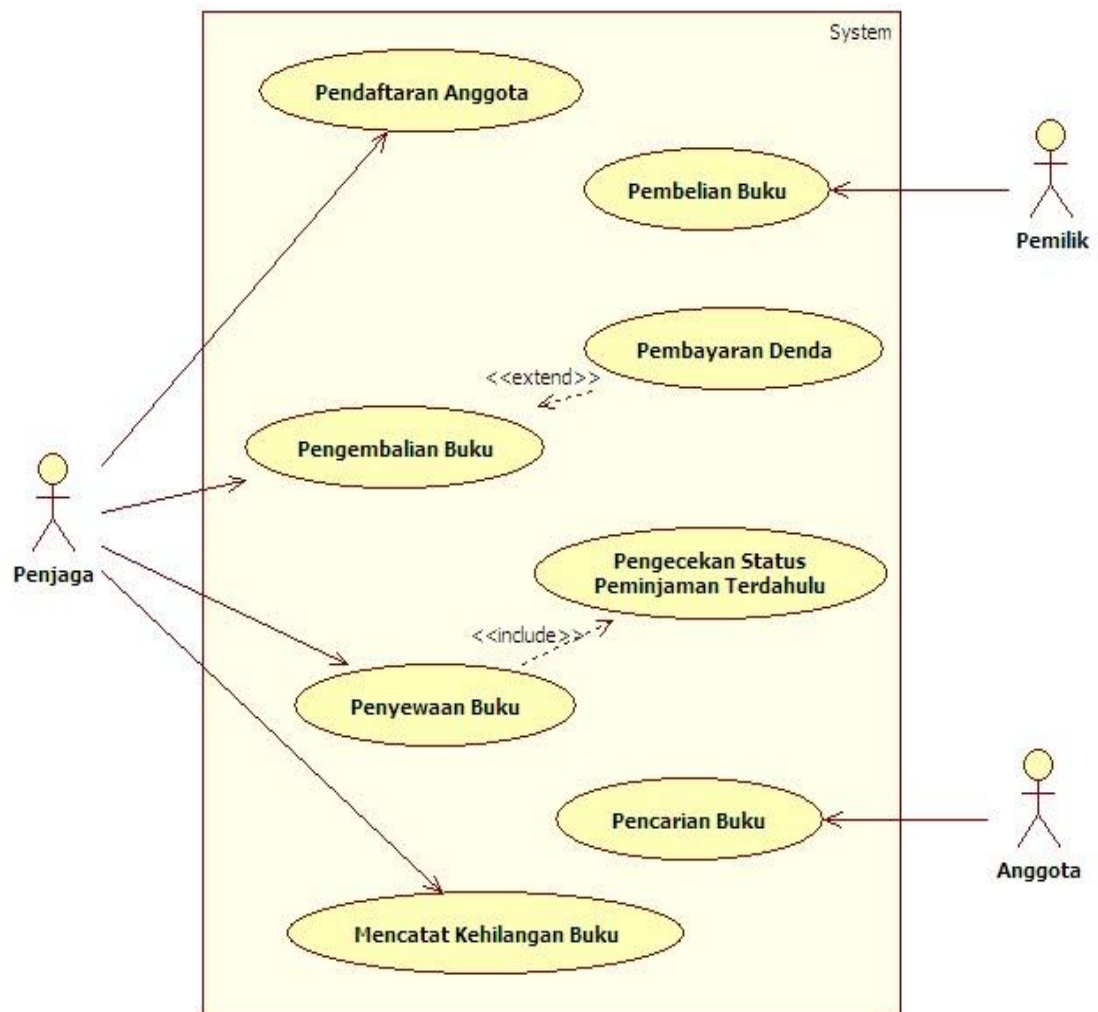
1. Panggil Star UML dengan cara mengklik tpmbol Star, pilih All Program, pilih Star UML dan lanjutkan dengan mengklik StarUML. Tunggu beberapa saat sampai ditampilkan menu pembuka seperti gambar di bawah ini.



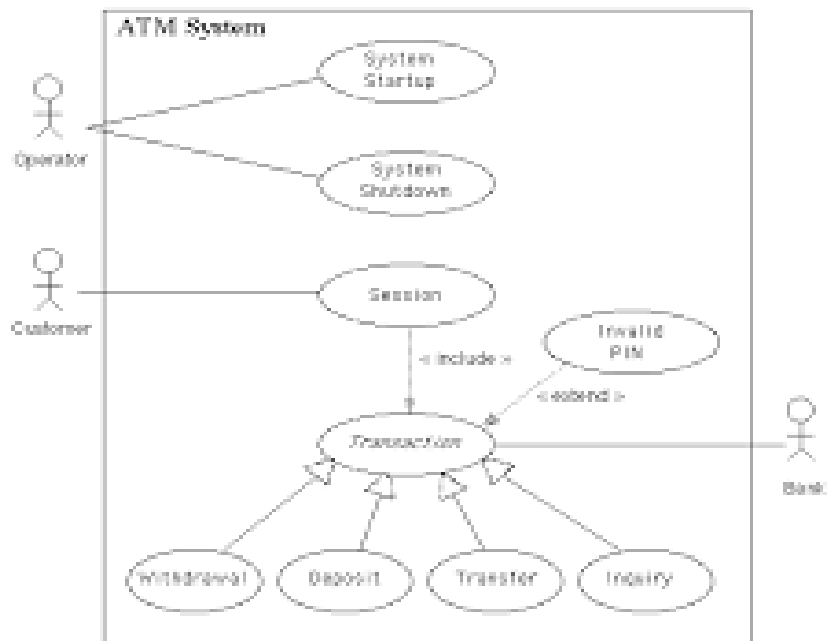
2. Pilih Default Approach lanjutkan dengan mengklik tombol OK, sehingga akan ditampilkan lembar kerja seperti terlihat pada gambar di bawah ini



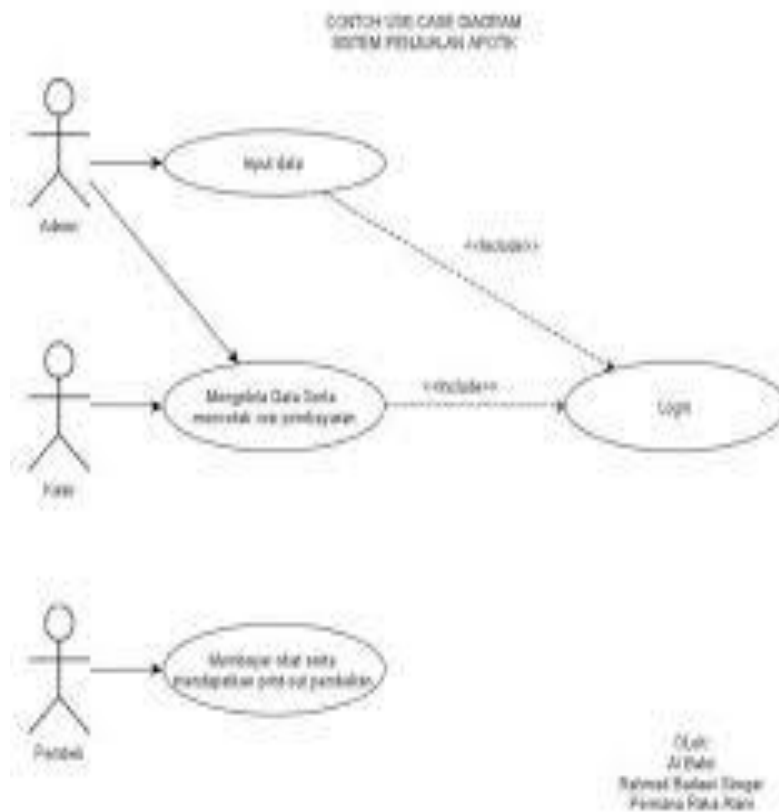
7.6.2 Menggambar Use Case Kasus 1



7.6.3 Menggambar Use Case Kasus 2



7.6.4 Menggambar Use Case Kasus 3



Penggunaan Star UML Untuk Membuat Rancangan Use Case Secara Rinci

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Star UML untuk pembuatan rancangan Use Case secara rinci.

Pertemuan ini akan menjelaskan secara singkat tentang proses include, extend dan generalisasi.

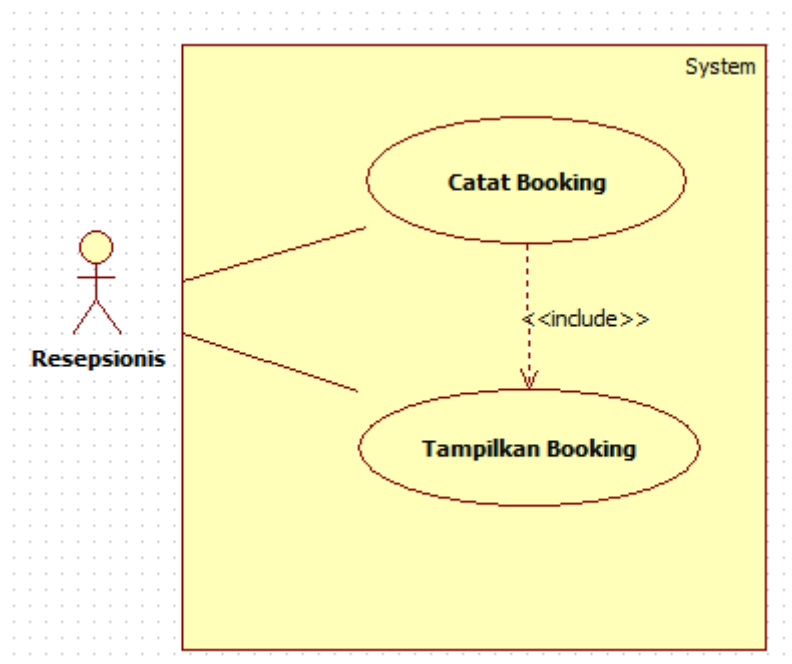
8.1 Pengantar

Dalam penggambaran Use Case terkadang akan ditemui apa yang disebut dengan Stereotype dimana istilah ini merupakan sebuah model khusus yang terbatas untuk kondisi tertentu. Untuk Menunjukkan stereotype digunakan symbol “<<” diawalnya dan ditutup “>>” diakhirnya.

8.2 Include

Stereotype <<include>> digunakan untuk menggambarkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya. Biasanya <<include>> digunakan untuk menghindari pengcopian suatu use case karena sering dipakai.

Contoh penggunaan stereotype <<include>>

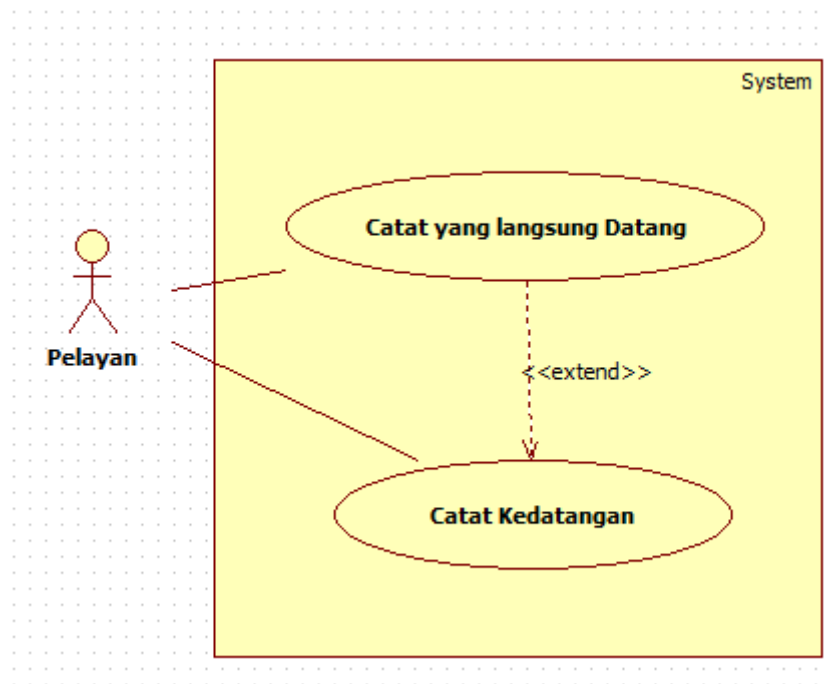


Gambar 8.1 Contoh Penggunaan Include

8.3 Extend

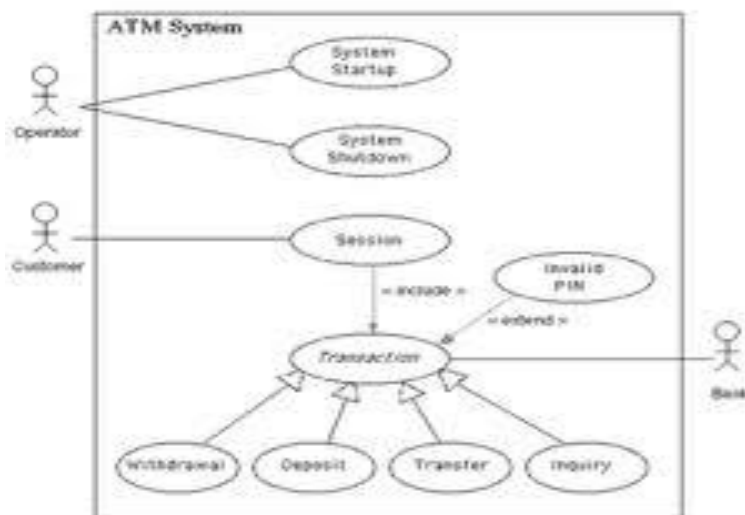
Stereotype <<extend>> digunakan untuk menunjukkan bahwa satu use case merupakan tambahan fungsionalitas dari use case yang lain jika kondisi atau syarat tertentu yang dipenuhi.

Contoh penggunaan stereotype <<extend>>



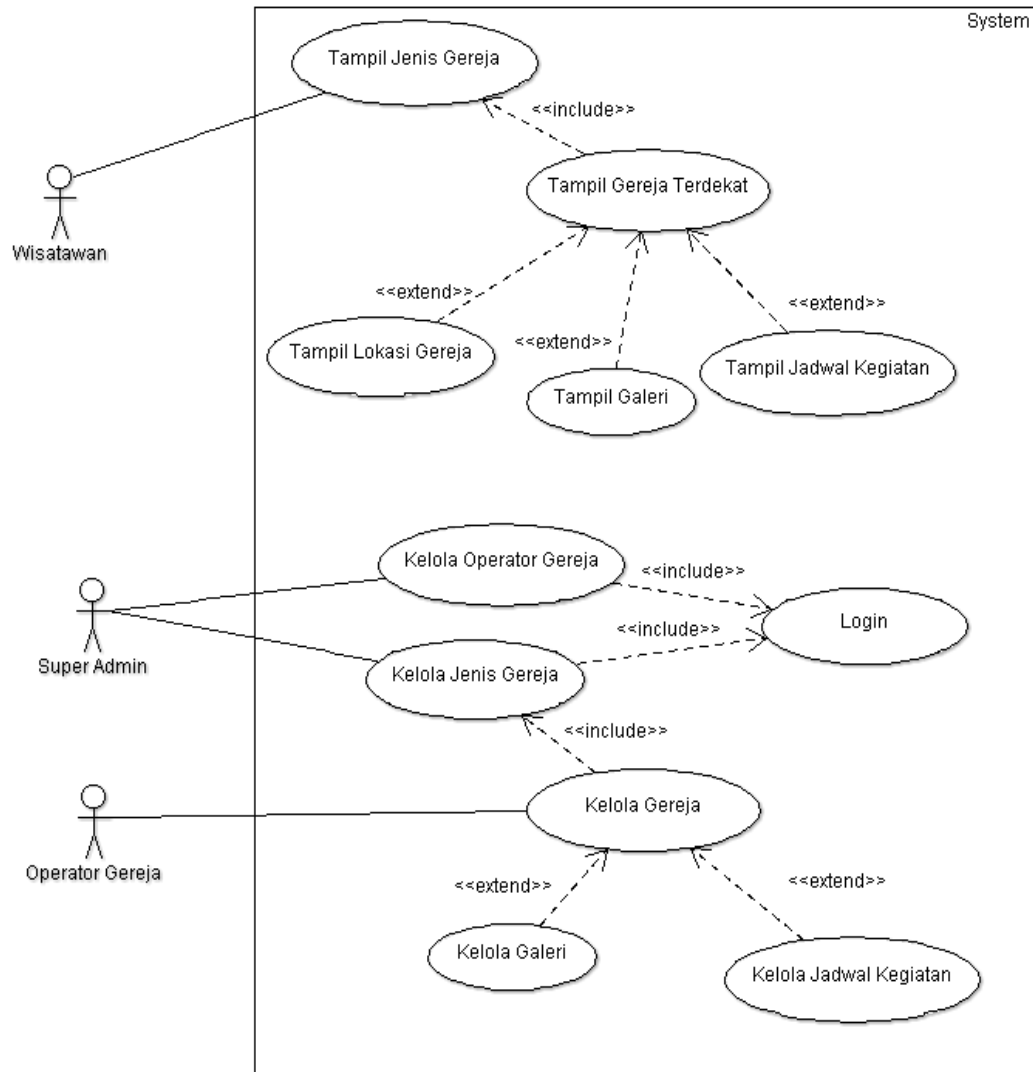
Gambar 8.2 Contoh Penggunaan Extend

8.4 Generalisasi

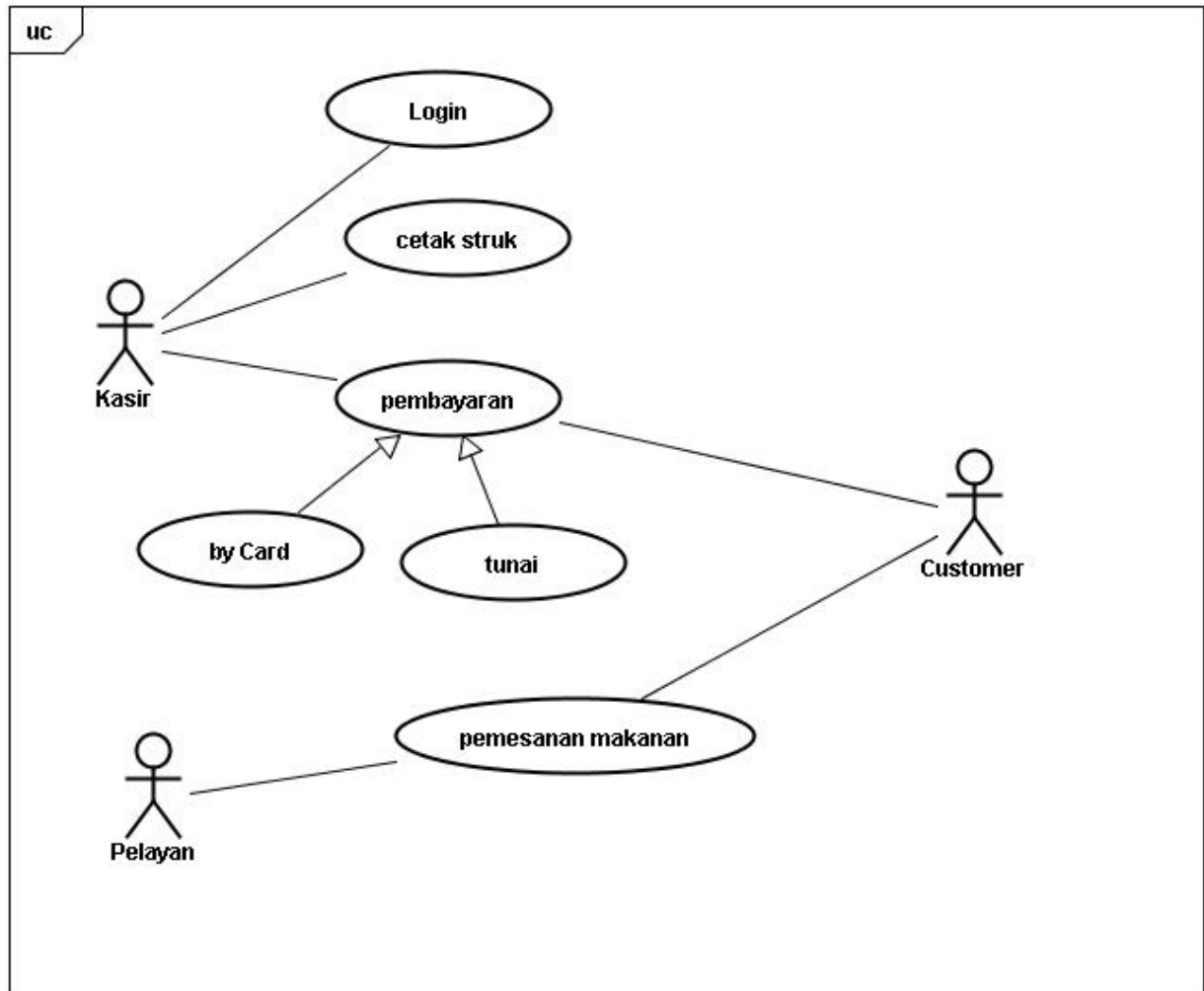


8.5 Praktek

8.5.1 Membuat Rancangan Use Case Dengan Include dan Extend



8.5.2 Membuat Rancangan Use Case Dengan Generalisasi



Penggunaan Star UML Untuk Membuat Rancangan Diagram Activity

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Star UML untuk membuat rancangan diagram activity.

Pertemuan ini akan menjelaskan secara singkat tentang pengertian diagram activity, simbol-simbol yang digunakan, memahami alur sistem dan menggambar masing-masing bagian.

9.1 Pengertian Diagram Activity

Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah system atau proses bisnis. Hal yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas system bukan apa yang dilakukan actor, jadi aktivitas yang dapat dilakukan oleh system. Diagram aktivitas mendukung perilaku parallel.

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



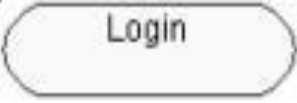


- Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis system yang didefinisikan.
- Urutan atau pengelompokkan tampilan dari system/user interface dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
- Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujiannya.

Activity diagram menggambarkan berbagai alir aktivitas dalam system yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses parallel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan state diagram khusus, dimana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (state processing). Oleh karena itu activity diagram tidak menggambarkan behavior internal sebuah system (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana actor menggunakan system untuk melakukan aktivitas.

9.2 Simbol-simbol yang Digunakan

Keterangan	Simbol
Titik Awal atau permulaan.	
Titik Akhir atau akhir dari aktivitas.	
Aktiviti, atau aktivitas yang dilakukan oleh aktor.	
Decision, atau pilihan untuk mengambil keputusan.	
Arah tanda panah alur	

9.3 Membuat Diagram Aktivitas

Diagram aktivitas mendeskripsikan aliran kerja dari perilaku sistem. Diagram ini hampir sama dengan diagram status karena kegiatan-kegiatannya merupakan status suatu pekerjaan dengan menunjukkan kegiatan yang dilakukan secara berurutan. Sebaiknya diagram aktivitas digunakan untuk melengkapi diagram lain seperti diagram interaksi dan diagram status, karena diagram aktivitas dapat mengetahui aliran sistem yang akan dirancang. Selain itu diagram aktivitas bermanfaat untuk menganalisis use case melalui penggambaran aksi-aksi yang dibutuhkan, penggambaran algoritma berurutan yang kompleks, dan pemodelan aplikasi dengan proses paralel. Tetapi diagram aktivitas tidak menunjukkan bagaimana objek berperilaku atau objek berkolaborasi secara detail.

1. Langkah-langkah Penggambaran

Diagram aktivitas dibaca dari atas ke bawah, mungkin bercabang untuk menunjukkan kondisi, keputusan dan atau memiliki kegiatan paralel.

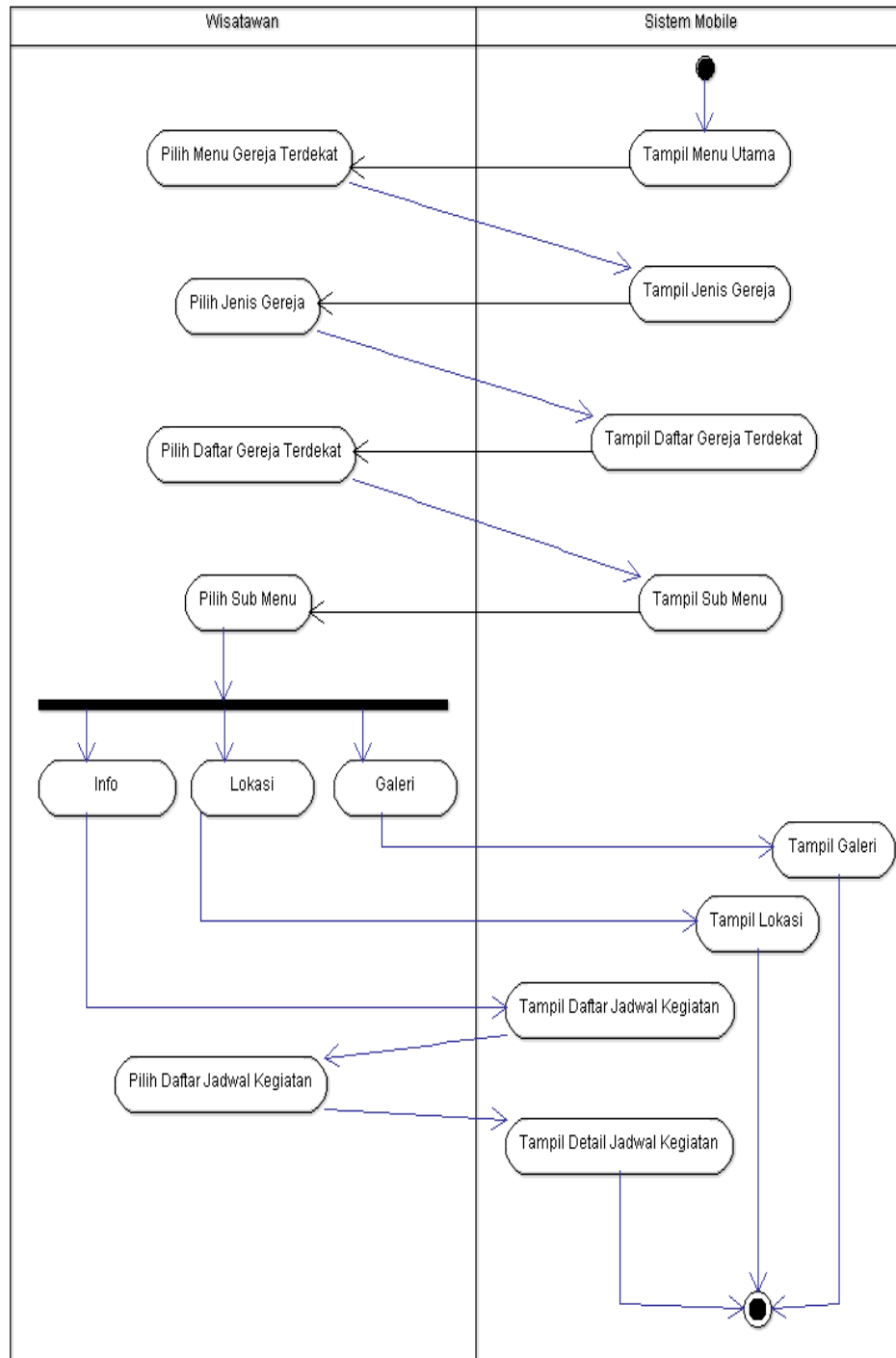
Berikut adalah langkah-langkah membuat diagram aktivitas:

- Buat simbol status awal ketika mengawali diagram.
- Gambarkan aksi pertama dan seterusnya sesuai aliran kegiatan sistem. Gunakan sebuah fork ketika berbagai aktivitas terjadi secara bersamaan. Setelah penggabungan seluruh kegiatan paralel, harus digabungkan dengan simbol join.
- Cabang keputusan digunakan untuk menunjukkan suatu kegiatan yang memenuhi kondisi tertentu. Seluruh pencabangan diakhiri tanda penggabungan (menggunakan tanda decision) sebagai akhir perilaku tersebut.
- Akhiri diagram dengan simbol status akhir.

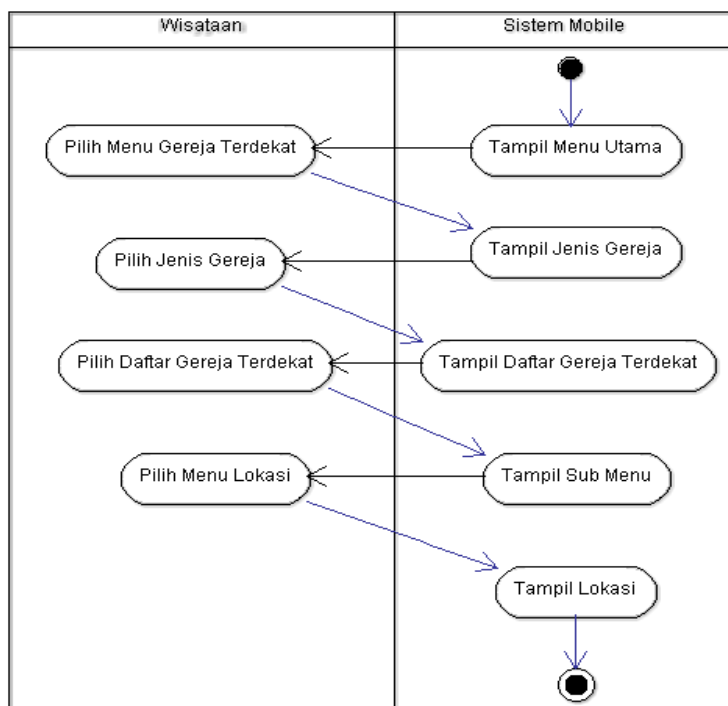
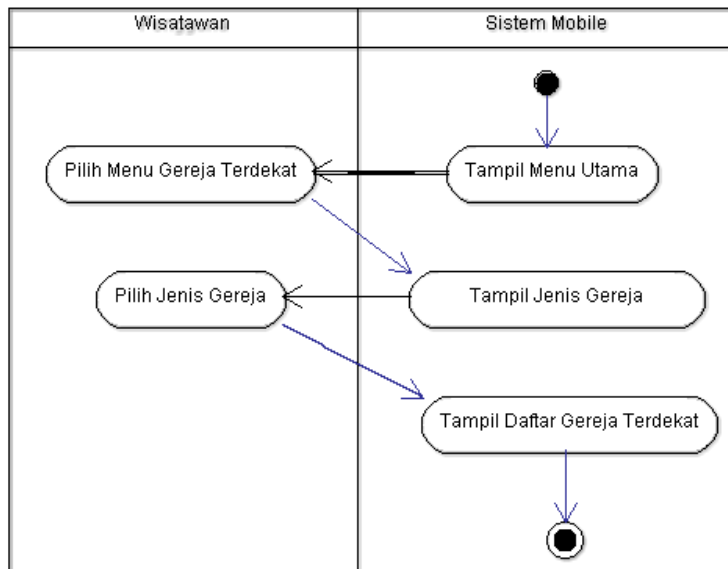
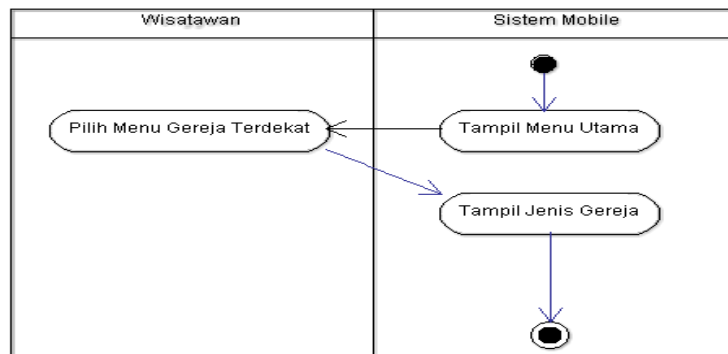
2. Contoh Diagram Aktivitas
 - a. Diagram aktivitas tanpa swimlane
 - b. Diagram aktivitas dengan swimlane

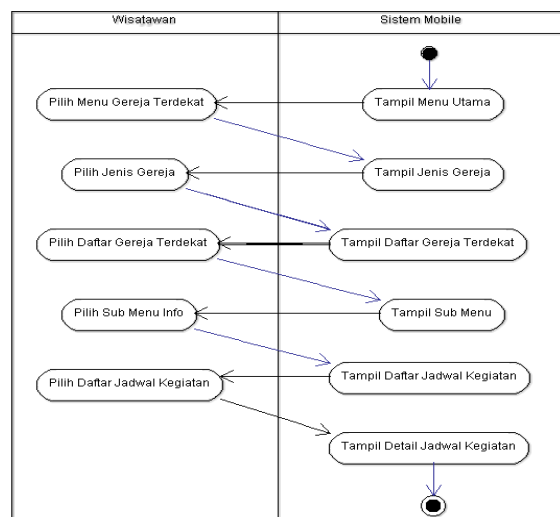
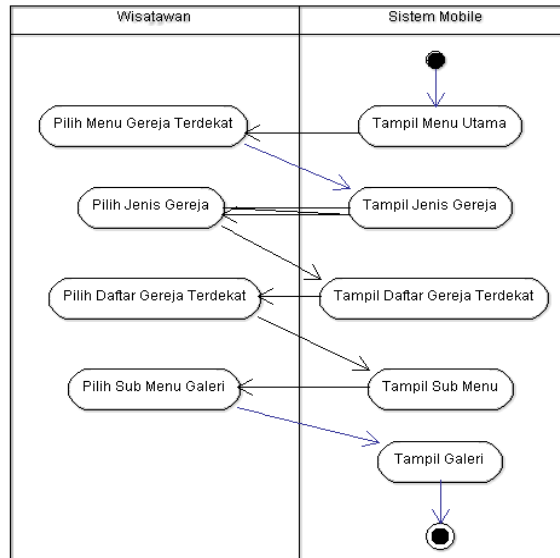
9.4 Praktek

9.4.1 Memahami Alur Sistem



9.4.2 Menggambar Alur Masing-masing Bagian





Penggunaan Star UML Untuk Membuat Rancangan Diagram Sequence

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Star UML untuk membuat rancangan diagram sequence.

Pertemuan ini akan menjelaskan secara singkat tentang pengertian diagram interaksi, diagram urutan (sequence), symbol-simbol yang digunakan, memahami fungsi yang digunakan dan menetapkan time line masing-masing fungsi.

10.1 Pengertian Diagram Interaksi

Diagram interaksi atau interaction diagram digunakan untuk memodelkan interaksi objek di dalam sebuah use case (proses). Diagram interaksi memperlihatkan interaksi yang memuat himpunan dari objek dan relasi yang terjadi antar objek tersebut, termasuk juga bagaimana message (pesan) mengalir diantara objek. Diagram interaksi terdiri dari dua buah diagram, yaitu diagram sekuen (sequence diagram) dan diagram kolaborasi (collaboration diagram). Diagram sekuen menggambarkan urutan even dan waktu dari suatu pesan yang terjadi antar objek dalam sebuah use case, sedangkan diagram kolaborasi menggambarkan bagaimana objek terkoneksi secara statik (tetap) dengan penekanan pada organisasi strktural objek-objek yang mengirim dan menerima pesan.

10.2 Pengertian Diagram Urutan (Sequence)

Diagram sequence 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 diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Banyaknya diagram sekuen yang harus digambar adalah sebanyak pendefinisian use case yang memiliki proses sendiri atau yang penting semua use case yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak use case yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

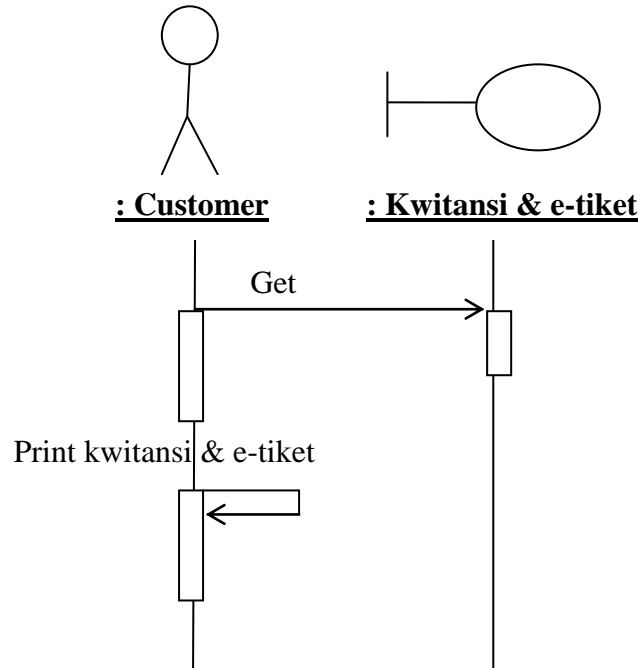
Penomoran pesan berdasarkan urutan interaksi pesan. Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu.

Diagram sekuen memiliki ciri yang berbeda dengan diagram interaksi pada diagram kolaborasi sebagai berikut:

1. Pada diagram sekuen terdapat garis hidup objek. Garis hidup objek adalah garis tegas vertikal yang mencerminkan eksistensi sebuah objek sepanjang periode waktu. Sebagian besar objek-objek yang tercakup dalam diagram interaksi akan eksis sepanjang durasi tertentu dari interaksi, sehingga objek-objek itu diletakkan di bagian atas diagram dengan garis hidup tergambar dari atas hingga bagian bawah diagram. Suatu objek lain dapat saja diciptakan, dalam hal ini garis hidup dimulai saat pesan create diterima suatu objek. Selain itu suatu objek juga dapat dimusnahkan dengan pesan Destroy, jika kasus ini terjadi, maka garis hidupnya juga harus berakhir.

2. Terdapat fokus kendali (focus of control), berupa empat persegi panjang ramping dan tinggi yang menampilkan aksi suatu objek secara langsung atau sepanjang sub ordinat. Puncak dari empat persegi panjang adalah permulaan aksi, bagian dasar adalah akhir dari suatu aksi. Pada diagram ini mungkin juga memperlihatkan penyaringan (nesting) dan fokus kendali yang disebabkan oleh proses rekursif dengan menumpuk fokus kendali yang lain pada induknya.

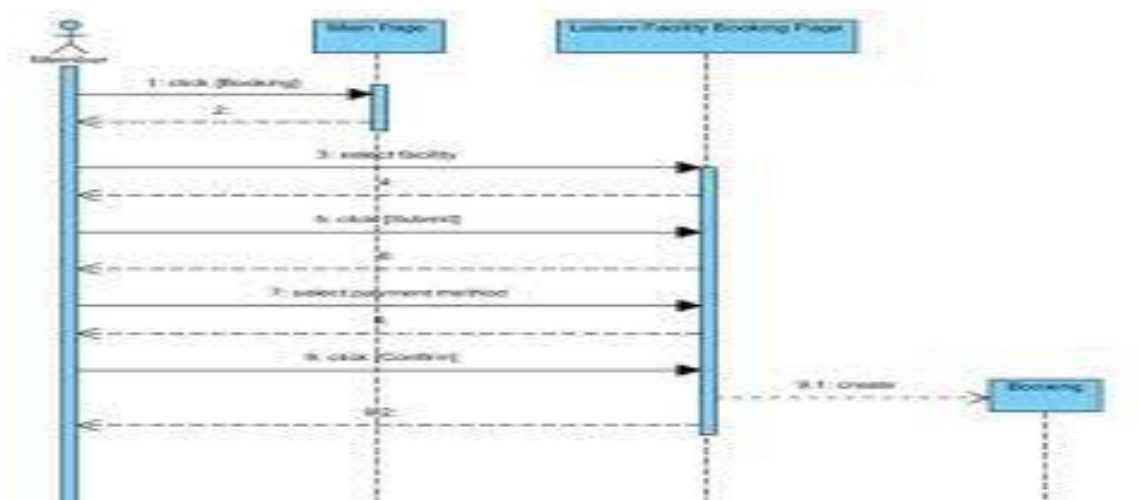
Contoh:



Gambar 10.1 Cetak Bukti Pembayaran Tiket

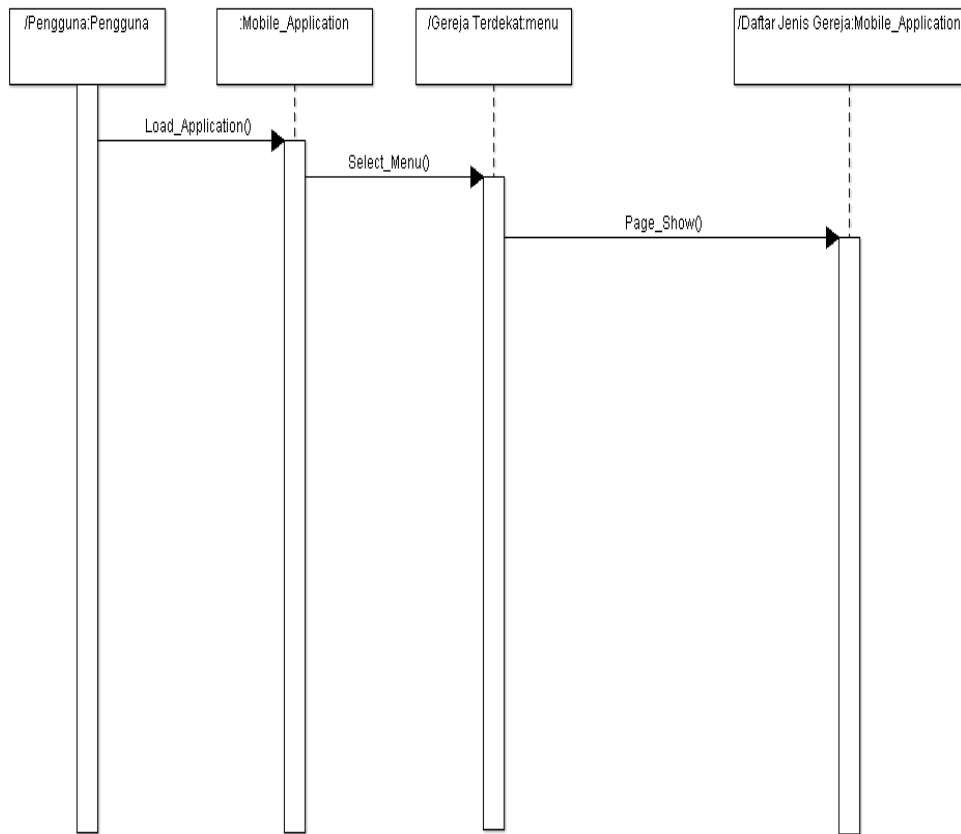
10.3 Simbol-simbol yang Digunakan

- Komponen utama sequence diagram terdiri atas obyek yang dituliskan dengan kotak segiempat bernama.
- Message diwakili garis dengan tanda panah dan waktu yang ditunjukkan dengan progres vertical.

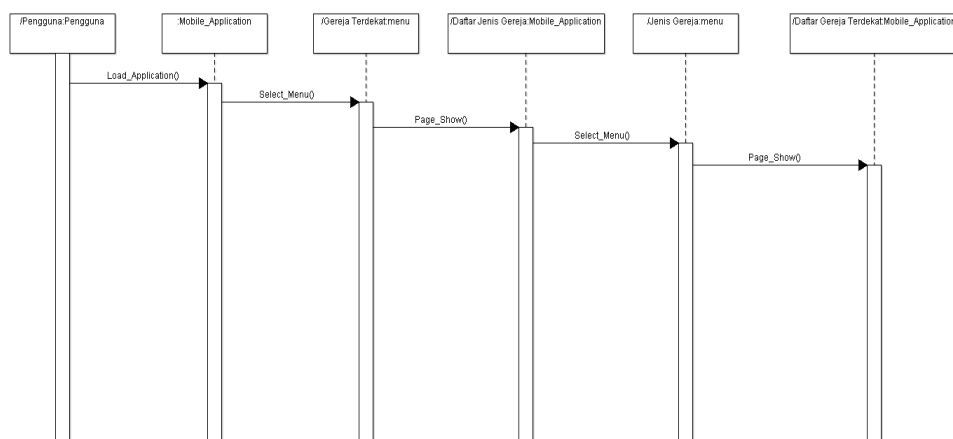


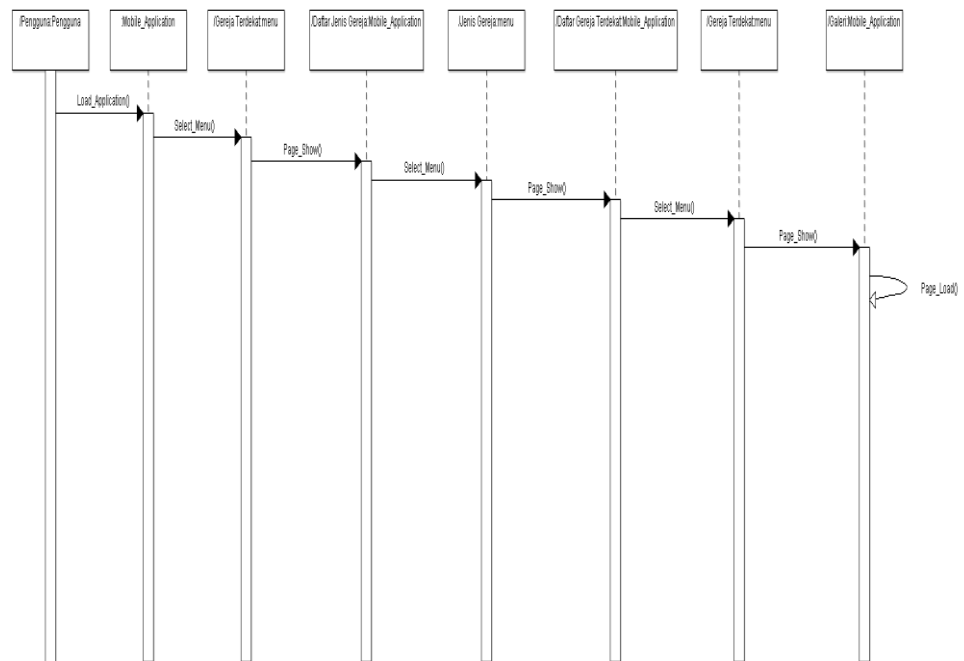
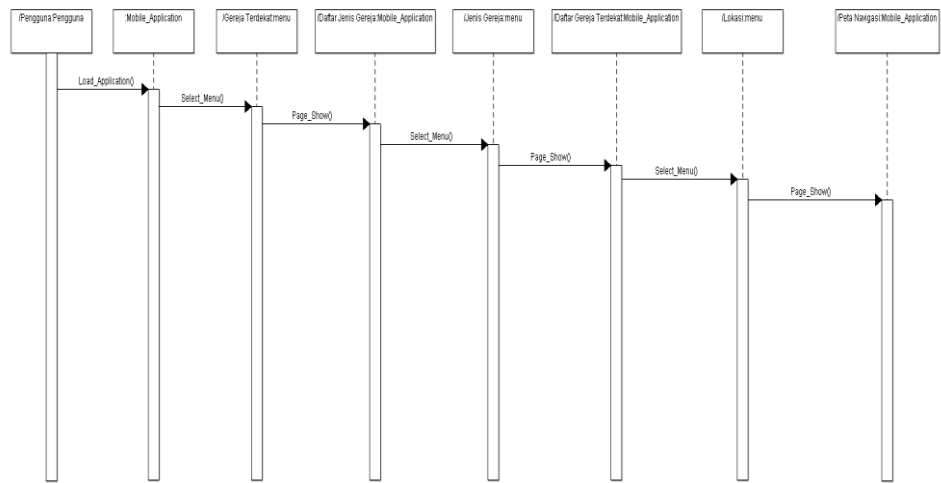
10.4 Praktek

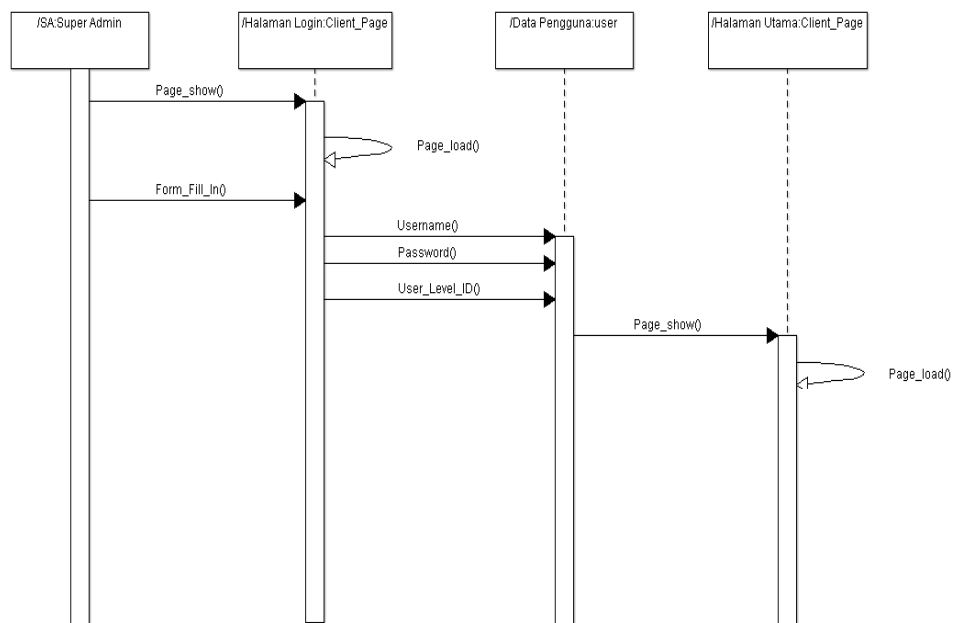
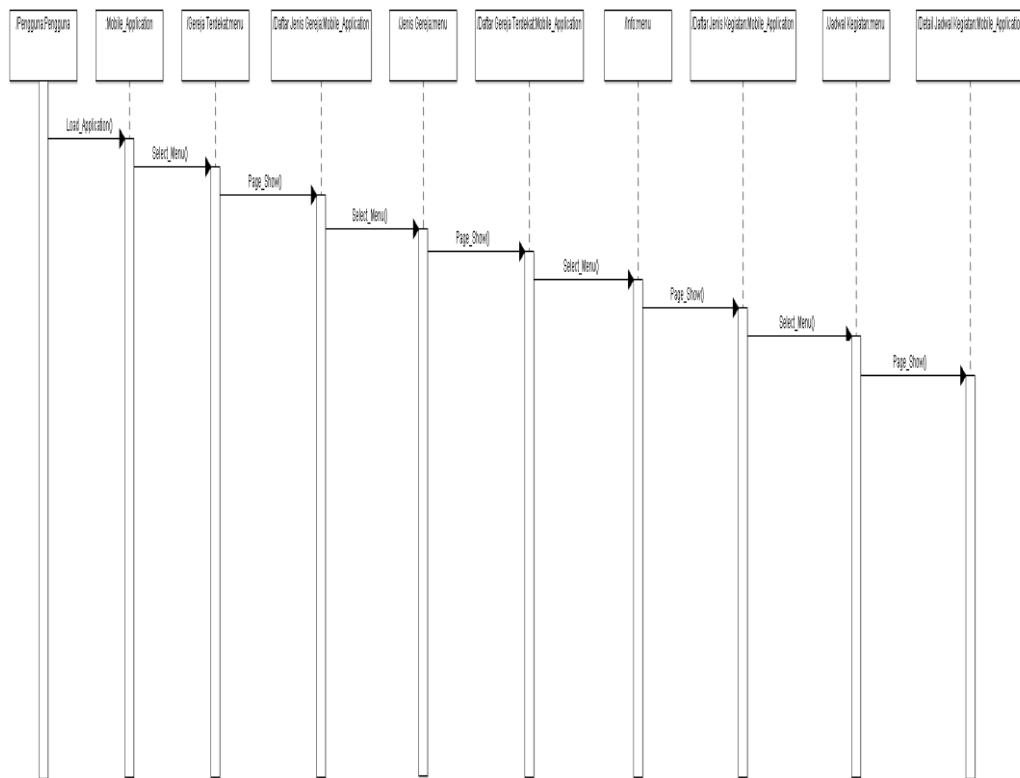
10.4.1 Memahami Fungsi yang digunakan

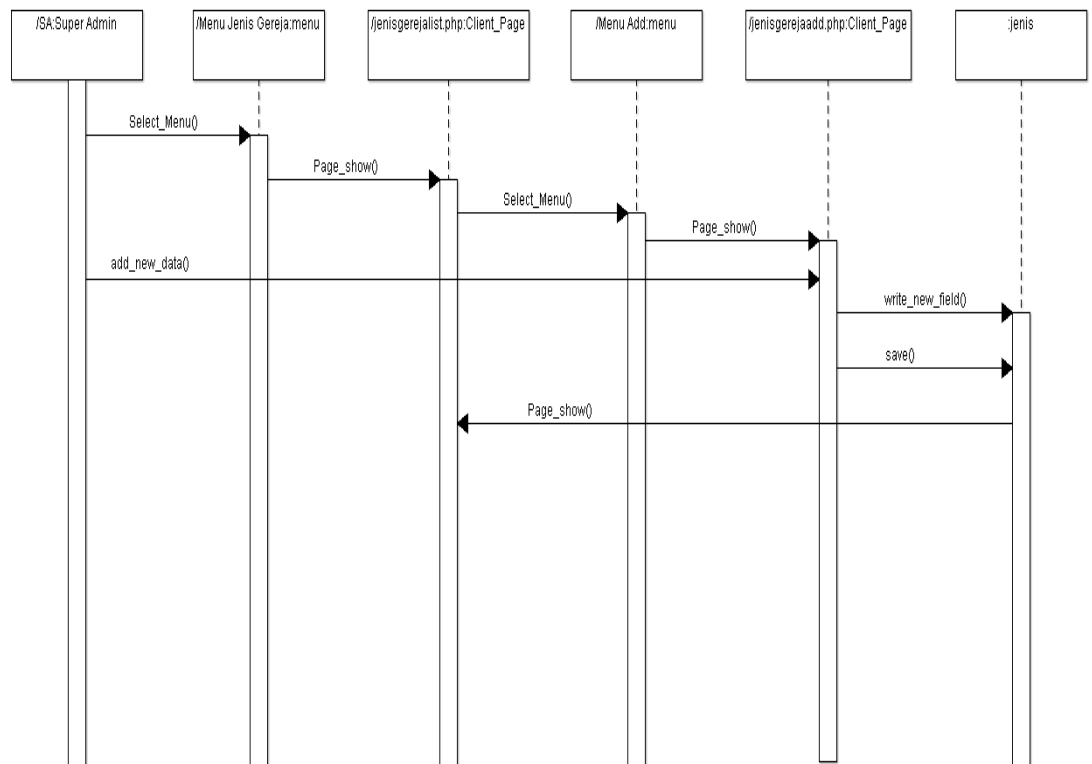
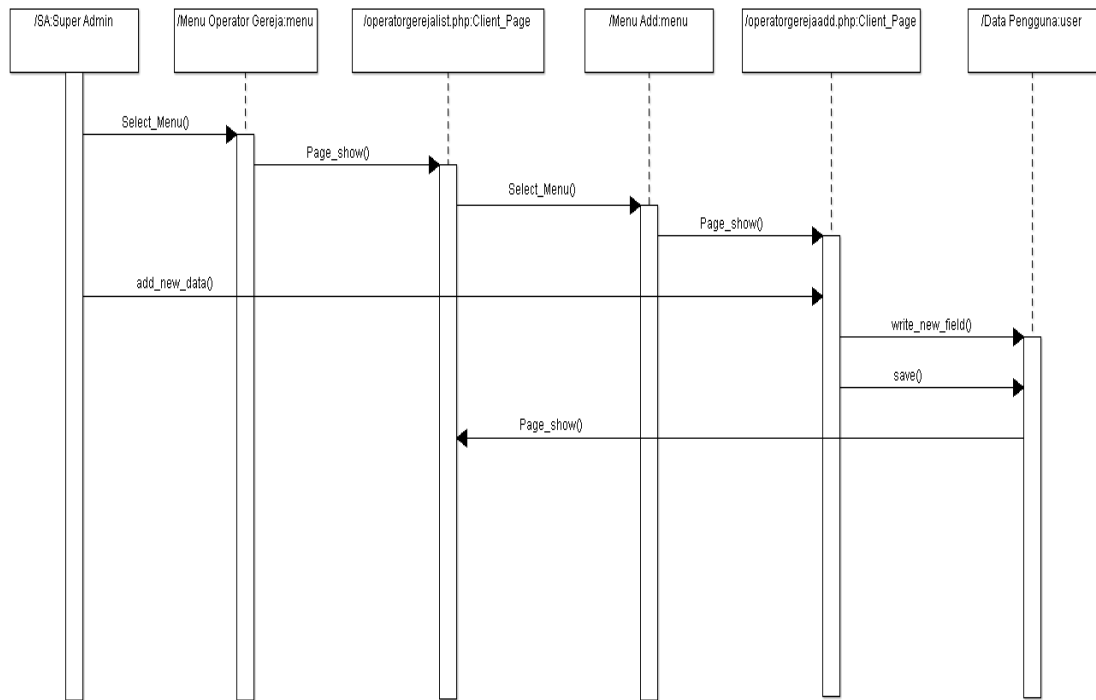


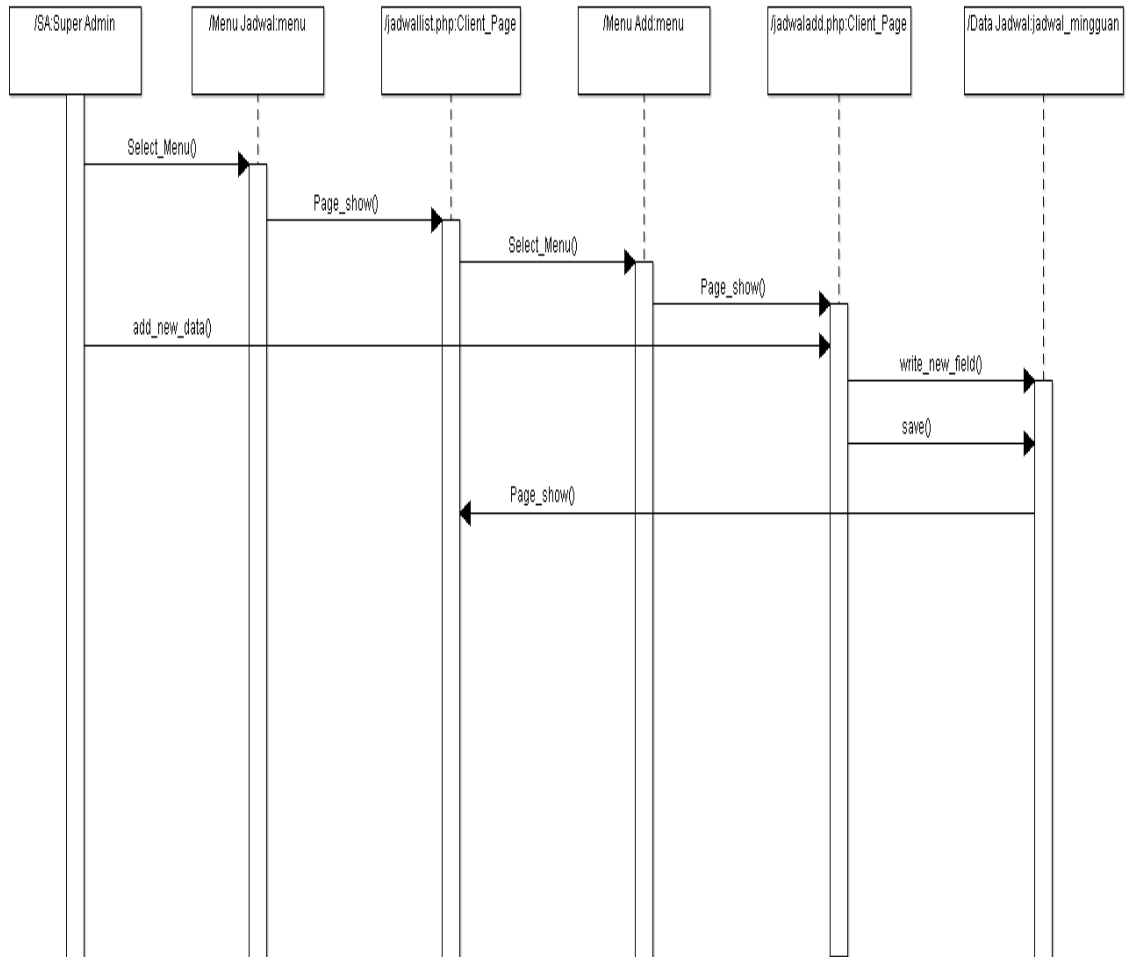
10.4.2 Menetapkan Time Line Masing-masing Fungsi











Penggunaan Star UML Untuk Pembuatan Diagram Class

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Star UML untuk menggambar Diagram Class.

Pertemuan ini akan menjelaskan secara singkat tentang menetapkan fungsionalitas, symbol-simbol yang digunakan, menyusun keterkaitan object dan menggambar diagram class.

11.1 Pengertian Diagram Kelas

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

- Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas.
- Atribut mendeskripsikan property dengan sebaris teks di dalam kotak kelas tersebut.
- Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Diagram kelas mendeskripsikan jenis-jenis objek dalam system dan berbagai hubungan statis yang terdapat di antara mereka. Diagram kelas juga menunjukkan property dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.

Diagram kelas menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain.

Kelas memiliki tiga area pokok:

1. Nama
2. Atribut
3. Operasi

Contoh kelas Manusia:

- Atribut: nama, usia, tanggal lahir
- Method/operasi: berjalan, makan, minum

1. Abstraksi Kelas

Abstraksi adalah menemukan hal-hal mendasar pada suatu objek dan mengabaikan hal-hal yang sifatnya incidental. Objek adalah instansiasi (contoh) dari sebuah kelas. Abstraksi bertujuan untuk menyaring properties dan operasi pada suatu objek, sehingga hanya tinggal yang dibutuhkan saja. Seringkali masalah yang berbeda membutuhkan sejumlah informasi yang berbeda pula pada area yang sama. Sebagai contoh, ketika kita akan membuat program untuk mengatur suatu pada objek TV dan perubahan channel, mungkin atribut no-seri TV harus dibuang karena tidak berguna. Tetapi ketika akan menelusuri transaksi penjualan TV, maka kita butuh nomor seri dari TV yang terjual.

2. Atribut

Atribut adalah karakteristik data yang dimiliki suatu objek dalam kelas.

Notasi dari atribut:

Visibility name: type multiplicity = default (property-string)

Contoh:

- Name: String[1] = "Untitled" {readOnly}

+ berarti public, - berarti private, # berarti protected

"Untitled" adalah nilai yang diberikan secara default jika tidak ditentukan saat objek dibuat.

{readOnly} adalah property tambahan dari atribut, dimana disini berarti tidak bisa dimodifikasi.

3. Operasi

Operasi adalah fungsi atau transformasi yang mungkin dapat diaplikasikan ke/oleh suatu objek dalam kelas. Misalnya, suatu objek dalam kelas manusia mungkin memiliki fungsi-fungsi tersenyum, marah, makan, minum, menerima perlakuan tertentu, dan sebagainya.

Notasi dari operations

Visibility name (parameter-list) : retur-type {property-string}

Dimana

Parameter pada parameter-list dinotasikan seperti pada atribut

Direction name: type = default value

Direction bias berupa: in, out, atau inout

Contoh:

+ balanceOn (date: Date) : Money

4. Multiplisitas/Multiplicity

Multiplisitas menunjukkan jumlah suatu objek yang bias berhubungan dengan objek yang lain. Umumnya ditunjukkan dengan berapa banyak objek yang bias mengisi property "satu" atau "banyak", tetapi secara khusus dapat ditunjukkan pula dengan bilangan integer lebih besar atau sama dengan nol.

- 1 (pasti satu)

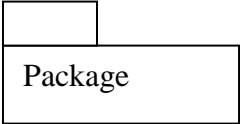
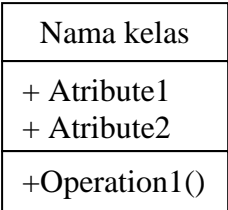
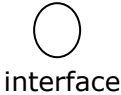


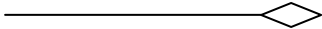

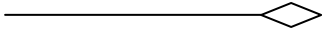
- 0..1 (0 atau 1)

- (Tidak ada batasan, nisa 0, 1, ..., n)

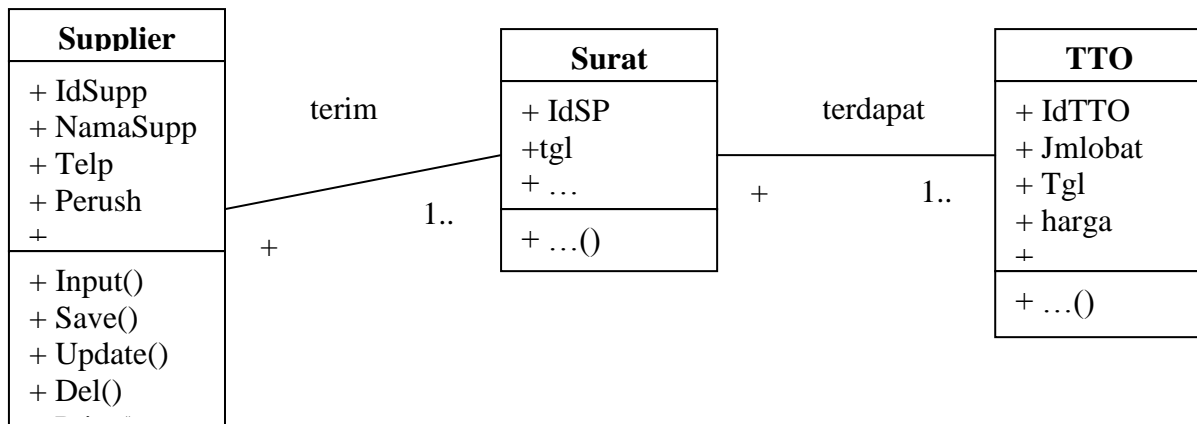
- Biasanya didefinisikan batas bawah dan atas, kecuali untuk yang pasti bernilai 1)

11.2 Simbol-simbol yang Digunakan

Simbol-simbol yang digunakan pada diagram kelas adalah sebagai berikut:

Simbol	Deskripsi
 <p>Package</p>	Package merupakan sebuah bungkusan dari satu atau lebih kelas
	Kelas pada struktur sistem
	Sama dengan konsep interface dalam pemrograman berorientasi objek
	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity
	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus)
	Relasi antar kelas dengan makna kebergantungan antar kelas
	Relasi antar kelas dengan makna semua-sebagian (whole-part)

Contoh diagram kelas:



Gambar 11.1 Contoh 1 Clas Diagram

11.3 Pendefinisian Kelas pada Diagram Kelas

Kelas/class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Kelas menggambarkan keadaan (atribut/property) suatu system, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (layanan/metoda/fungsi). Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan system. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

- Kelas main
Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan
- Kelas yang menangani tampilan sstem
Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
- Kelas yang diambil dari pendefinisian use case
Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian use case.
- Kelas yang diambil dari pendefinisian data
Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Jenis-jenis kelas di atas juga dapat digabungkan satu sama lain sesuai dengan pertimbangan yang dianggap baik asalkan fungsi-fungsi yang sebaiknya ada pada struktur kelas tetap ada. Susunan kelas juga dapat ditambahkan kelas utilitas seperti koneksi ke basis data, membaca file teks, dan lain sebagainya sesuai kebutuhan.

Dalam mendefinisikan metode yang ada di dalam kelas perlu memperhatikan apa yang disebut dengan cohesion dan coupling. Cohesion adalah ukuran seberapa dekat keterkaitan instruksi di dalam sebuah metode terkait satu sama lain, sedangkan coupling adalah ukuran seberapa dekat keterkaitan instruksi antara metode yang satu dengan metode yang lain dalam sebuah kelas. Sebagai aturan secara umum maka sebuah metode yang dibuat harus memiliki kadar cohesion yang kuat dan kadar coupling yang lemah.

11.4 Relasi Antar Kelas

Relasi antar kelas adalah keterkaitan hubungan antar kelas secara konseptual. UML menyediakan beberapa relasi antar kelas yaitu asosiasi, agregasi, generalisasi dan dependensy.

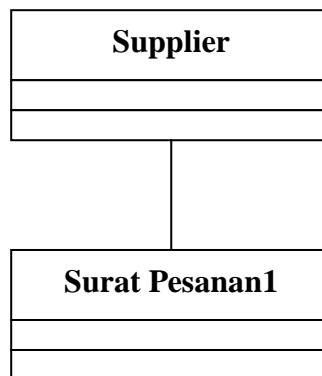
1. Asosiasi

Asosiasi yaitu hubungan statis antar kelas. Umumnya menggambarkan Class yang memiliki atribut berupa kelas lain, atau class yang harus mengetahui eksistensi class lain. Panah navigability menunjukkan arah query antar kelas.

Menggambarkan hubungan antar kelas:

- Ditandai dengan anak panah
- Seringkali ditambahkan label dan multiplicity untuk memperjelas hubungan

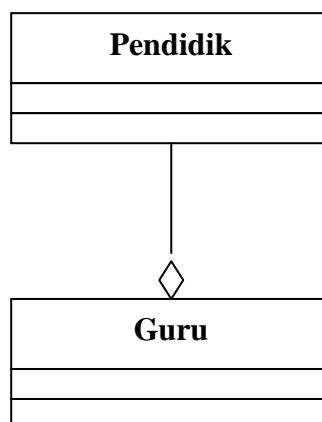
Contoh Asosiasi:



Gambar 11.2 Contoh 2 Class Diagram

2. Agregasi

Agregasi adalah hubungan bagian dari atau bagian ke keseluruhan. Suatu kelas/objek mungkin memiliki/bisa dibagi menjadi kelas/objek tertentu dimana objek/kelas yang disebut kemudian merupakan bagian dari kelas/objek yang terdahulu.

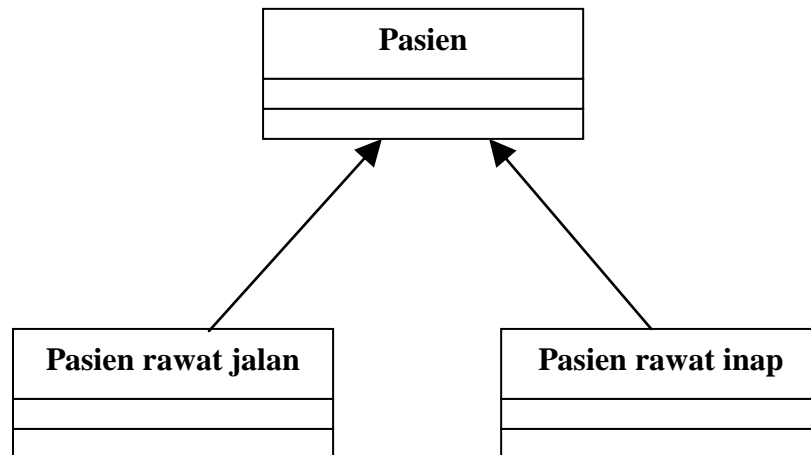


Gambar 11.3 Contoh Agregasi Class Diagram

3. Generalisasi

Generalisasi adalah relasi ke atas beberapa subkelas kepada super kelas di atasnya (ditunjukkan dengan notasi segitiga). Sub kelas mewarisi fitur dari super kelasnya. Sub kelas mampu overriding metode super kelasnya.

Contoh:



Gambar 11.4 Contoh Generalisasi Class Diagram

4. Depedensy

Dependency adalah hubungan dimana perubahan pada suatu kelas akan mempengaruhi kelas yang lain dimana kelas yang terakhir ini bergantung pada kelas yang sebelumnya. Dalam dependency antar 2 elemen jika terjadi perubahan pada salah satu elemen maka akan mengakibatkan perubahan pada elemen yang lain.

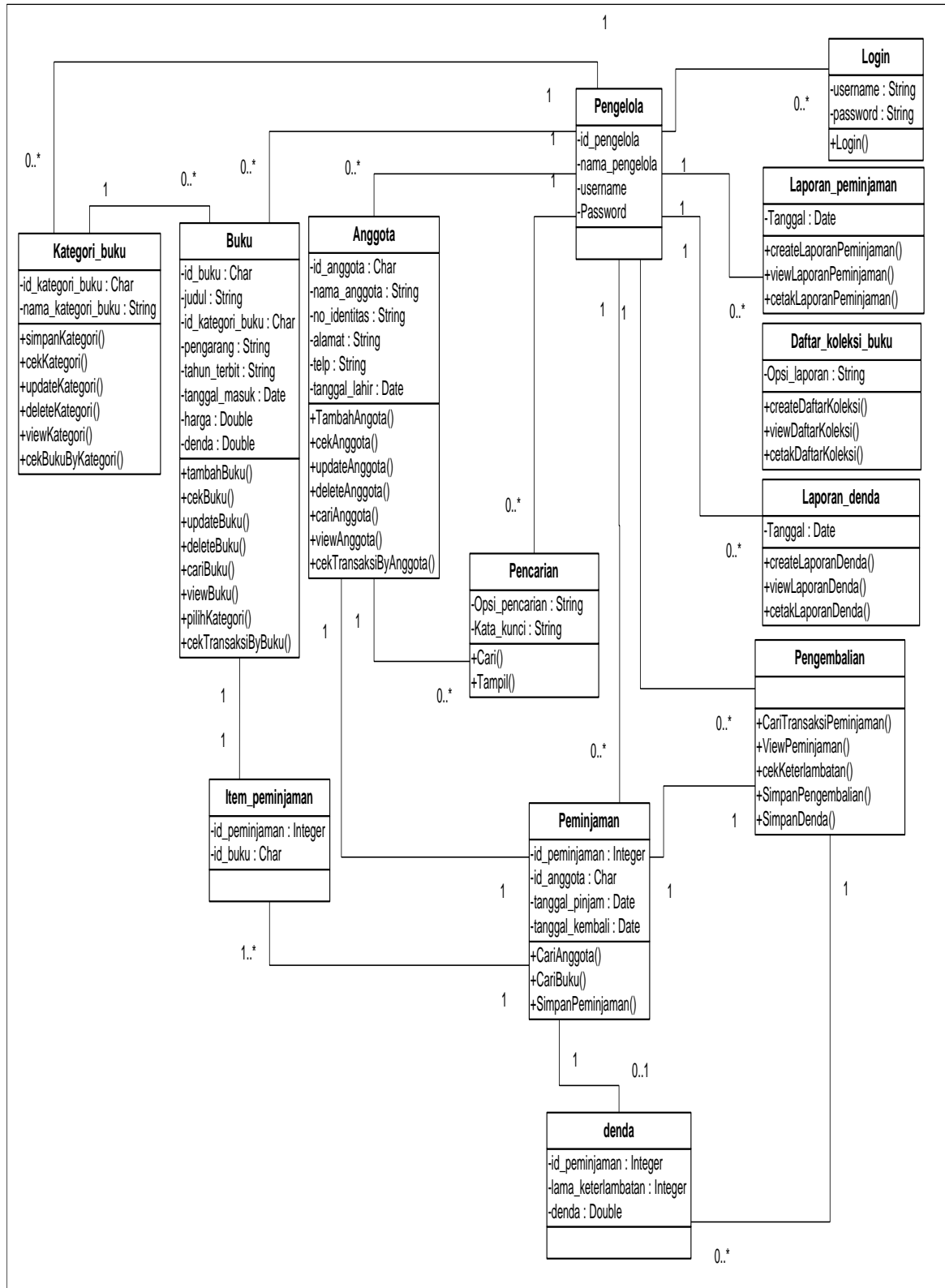
Semakin kompleks system, maka dependency menjadi sesuatu yang harus dipertimbangkan.

Dependency hanya berlaku satu arah.

Bisa diperjelas dengan penggunaan keyword, seperti <<parameter>>, <<use>>, <<call>> Notas anak panah dan garis putus-putus.

11.5 Praktek

11.5.1 Menggambar Diagram Class Proyek1



11.5.2 Menggambar Diagram Class Proyek2

Tujuan Instruksional Khusus:

Setelah mempelajari bab ini, mahasiswa diharapkan dapat menggunakan perangkat lunak Star UML untuk menyelesaikan suatu studi kasus.

Pertemuan ini akan menjelaskan secara singkat tentang identifikasi masalah, analisa sistem, menggambar use case, menggambar diagram activity, menggambar diagram sequence dan menggambar diagram class.

12.1 Identifikasi Masalah

Permasalahan yang dihadapi adalah bahwa Komunitas ResiBisma mendapatkan sumbangan buku dalam jumlah yang sangat banyak. Buku-buku ini kemudian akan dijadikan sebagai koleksi yang bisa dibaca dan dipinjam oleh siapa saja, sehingga sesuai dengan tujuan Komunitas ResiBisma untuk memajukan pendidikan khususnya memberikan pendidikan gratis.

Dengan jumlah koleksi buku yang begitu banyak, dikhawatirkan akan menimbulkan permasalahan terkait dengan aktivitas membaca dan meminjam koleksi buku. Permasalahan tersebut antara lain:

- a) Sulitnya mendata koleksi buku yang ada.
- b) Sulitnya mendata siapa saja yang berkunjung untuk membaca dan meminjam koleksi buku.
- c) Permasalahan pada poin a) dan poin b) kemudian dapat menimbulkan masalah yang lain, yaitu berkurangnya jumlah koleksi buku karena hilang.

12.2 Analisa Sistem

Pada saat ini belum terdapat sistem informasi perpustakaan di Komunitas ResiBisma. Sehingga tahapan analisa sistem yang sudah berjalan tidak perlu dilakukan. Berdasarkan pada tahapan identifikasi masalah, kemudian diputuskan untuk membuat sebuah sistem informasi perpustakaan. Fitur-fitur yang akan dikembangkan agar permasalahan-permasalahan di atas dapat teratasi adalah sebagai berikut:

- a) Fitur pengelolaan koleksi buku, meliputi proses klasifikasi buku ke dalam kategori-kategori tertentu serta operasi untuk menambah, mengedit dan menghapus data buku.
- b) Fitur pengelolaan data anggota perpustakaan. Yang dapat membaca dan meminjam koleksi buku adalah yang sudah terdaftar sebagai anggota perpustakaan.

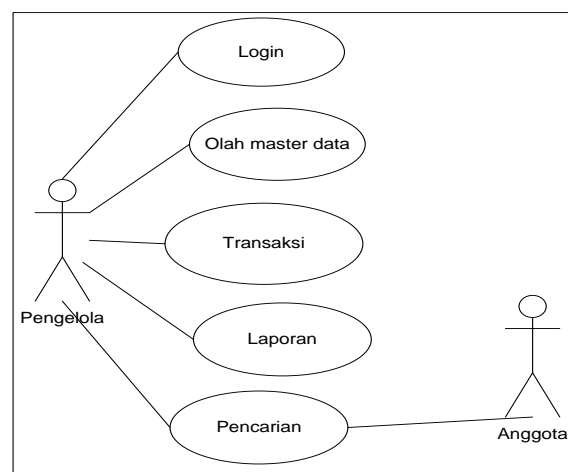
- c) Fitur pencatatan transaksi peminjaman dan pengembalian buku oleh anggota perpustakaan serta pencatatan denda jika pengembalian buku yang dipinjam melebihi batas waktu yang ditentukan.
- d) Fitur untuk menampilkan serta mencetak laporan transaksi peminjaman dan denda, serta daftar koleksi buku.

Mengingat pada Komunitas ResiBisma terdapat divisi IT, maka proses pengembangan sistem informasi perpustakaan ini akan dilakukan oleh divisi IT.

12.3 Praktek

12.3.1 Menggambar Use Case

Use case diagram akan memperlihatkan bagaimana peranan setiap *actor* dalam interaksi dengan sistem. *Use case diagram* untuk sistem yang akan dikembangkan dapat dilihat pada gambar 12.1 di bawah ini:



Gambar 12.1 *Use Case Diagram*

Dari seluruh *use case* yang ada pada gambar 2, *use case* olah master data, entri transaksi serta tampilkan dan cetak laporan dapat dirinci lagi ke dalam *use case diagram* yang lebih terperinci. Sedangkan deskripsi untuk *use case* login dan pencarian koleksi buku dapat dilihat pada tabel 1 dan tabel 2.

Tabel 1 Deskripsi *use case* login

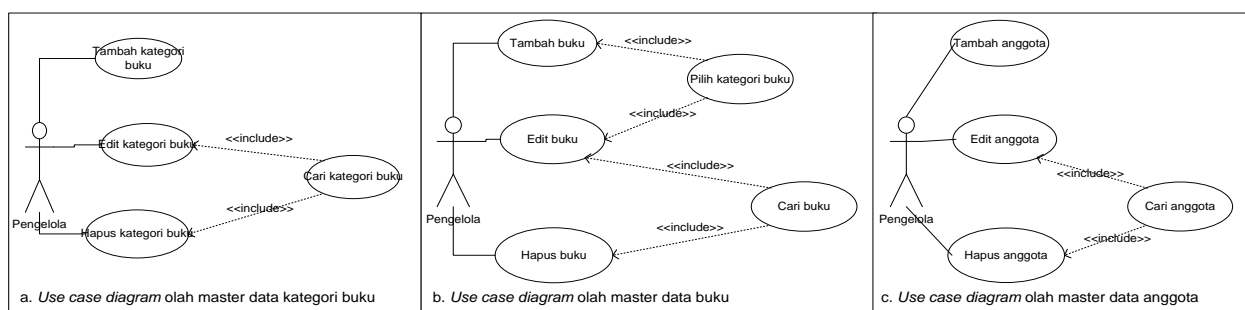
Use case name	Login
Scenario	Login ke sistem informasi perpustakaan
Brief description	Pengelola akan memasukkan username dan password pada form login. Sistem akan melakukan pengecekan username dan password yang dimasukkan dengan data yang tersimpan pada database.
Actors	Pengelola
Related use case	-
Stakeholder	Pengelola

Precondition	Data pengelola sudah harus tersimpan dalam database.	
Postcondition	Sistem informasi perpustakaan akan terbuka	
Flow of events	<i>Actors</i>	<i>System</i>
	Pengelola memasukkan username dan passwords	Sistem akan melakukan pengecekan username dan password yang dimasukkan dengan data yang tersimpan pada database
Exception condition	Jika pengelola belum terdaftar: • Gunakan login default.	

Tabel 2 Deskripsi *use case* pencarian koleksi buku

Use case name	Pencarian koleksi buku	
Scenario	Melakukan pencarian koleksi buku berdasarkan opsi pencarian tertentu	
Brief description	Pengelola/ Anggota akan memilih opsi pencarian tertentu dan memasukkan kata kunci, kemudian klik tombol cari.	
Actors	Pengelola & Anggota	
Related use case	-	
Stakeholder	Pengelola & Anggota	
Precondition	Data buku yang dicari sudah harus tersimpan dalam database	
Postcondition	Akan ditampilkan data buku sesuai dengan opsi pencarian dan kata kunci yang dipergunakan	
Flow of events	<i>Actors</i>	<i>System</i>
	Pengelola/ anggota memilih opsi pencarian memasukkan kata kunci	Sistem akan mencari data buku yang sesuai dengan opsi pencarian dan kata kunci yang dipergunakan
Exception condition	Jika tidak diperoleh hasil pencarian: • Tampilkan pesan	

Use case diagram yang lebih rinci untuk olah master data dapat dirinci ke dalam tiga *use case*, yaitu *use case* olah master data kategori buku, *use case* olah master data buku dan *use case* olah master data anggota. Gambar 12.2a menunjukkan *use case diagram* olah master data kategori buku, gambar 12.2b menunjukkan *use case diagram* olah master data buku dan gambar 12.2c menunjukkan *use case diagram* olah master data anggota.



Gambar 12.2 *Use case diagram* olah master data

Deskripsi untuk masing-masing *use case* adalah sebagai berikut:

- 1) *Use case* olah master data kategori buku

Dalam pengolahan master data kategori buku, terdapat tiga *use case* utama, yaitu tambah kategori buku, edit kategori buku dan hapus kategori buku. Deskripsi masing-masing *use case* dapat dilihat pada tabel 3, tabel 4 dan tabel 5.

Tabel 3 Deskripsi *use case* tambah kategori buku

Use case name	Tambah kategori buku	
Scenario	Menambah kategori buku.	
Brief description	Pengelola memasukkan id kategori buku dan nama kategori buku. Sistem akan mengecek apakah data tersebut sudah ada atau belum. Jika belum ada, sistem akan menyimpan data tersebut ke dalam database.	
Actors	Pengelola	
Related use case	Login, cari kategori buku	
Stakeholder	Pengelola	
Precondition	-	
Postcondition	Data kategori buku akan disimpan ke dalam sistem.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data kategori buku. 2) Pengelola memasukkan data kategori buku.	2a) Sistem akan mengecek apakah kategori buku sudah ada atau belum. Jika belum sistem akan menyimpan ke dalam database.
Exception condition	Jika id kategori buku sudah ada: • Tampilkan pesan.	

Tabel 4 Deskripsi *use case* edit kategori buku

Use case name	Edit kategori buku	
Scenario	Mengedit kategori buku.	
Brief description	Pengelola akan mencari data kategori buku tertentu yang akan diedit, kemudian data kategori buku diedit dan disimpan kembali ke dalam sistem.	
Actors	Pengelola	
Related use case	Login, tambah kategori buku, cari kategori buku	
Stakeholder	Pengelola	
Precondition	Data kategori buku sudah harus tersimpan dalam database.	
Postcondition	Data kategori buku akan disimpan ke dalam database.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data buku 2) Pengelola akan mencari dan memilih data kategori buku yang akan diedit 3) Pengelola mengedit sesuai kebutuhan kemudian klik tombol simpan	3a) Sistem akan menyimpan data kategori buku yang sudah diedit.
Exception condition		

Tabel 5 Deskripsi *use case* hapus kategori buku

Use case name	Hapus kategori buku	
Scenario	Menghapus kategori buku.	
Brief description	Pengelola akan mencari data kategori buku tertentu yang akan dihapus, sistem akan mengecek, apakah kategori buku tersebut terelasi dengan data buku tertentu. Jika tidak sistem akan menghapus data kategori buku tersebut dari database.	
Actors	Pengelola	
Related use case	Login, tambah kategori buku, cari kategori buku	
Stakeholder	Pengelola	
Precondition	Data kategori buku sudah harus tersimpan dalam database.	
Postcondition	Data kategori buku akan dihapus dari database.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data kategori buku 2) Pengelola akan mencari dan memilih data kategori buku 3) Pengelola mengklik tombol hapus untuk menghapus.	3a) Sistem akan mengecek apakah kategori buku tersebut terelasi dengan buku tertentu atau tidak. Jika tidak, maka kategori buku akan dihapus.
Exception condition	Jika kategori buku terelasi dengan buku tertentu: <ul style="list-style-type: none"> • Tampilkan pesan. 	

2) *Use case* olah master data buku

Use case olah master data buku dapat dirinci ke dalam tiga *use case* utama, yaitu: tambah buku, edit buku dan hapus buku. Deskripsi untuk masing-masing *use case* dapat dilihat pada tabel 6, tabel 7 dan tabel 8.

Tabel 6 Deskripsi *use case* tambah buku

Use case name	Tambah buku	
Scenario	Menambah buku	
Brief description	Pengelola memasukkan data buku dan sistem akan mengecek apakah data sudah ada dalam sistem atau belum, jika belum data buku akan disimpan.	
Actors	Pengelola	
Related use case	Login, olah master data kategori buku, cari kategori buku	
Stakeholder	Pengelola	
Precondition	Data kategori buku sudah harus tersimpan dalam database.	
Postcondition	Data buku akan disimpan ke dalam sistem.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data buku. 2) Pengelola memasukkan data buku dan klik tombol simpan.	2a) Sistem akan mengecek data yang dimasukkan sudah ada atau belum. Jika belum ada, maka sistem akan menyimpan ke dalam database
Exception condition	Jika buku sudah ada: <ul style="list-style-type: none"> • Tampilkan pesan. 	

Tabel 7 Deskripsi *use case* edit buku

Use case name	Edit buku	
Scenario	Mengedit kategori buku.	
Brief description	Pengelola akan mencari data buku tertentu yang akan diedit, kemudian data buku diedit dan disimpan kembali ke dalam sistem.	
Actors	Pengelola	
Related use case	Login, olah master data buku, cari kategori buku, tambah buku	
Stakeholder	Pengelola	
Precondition	Data kategori buku dan data buku sudah harus tersimpan dalam database	
Postcondition	Data buku akan disimpan ke dalam database.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data buku. 2) Pengelola akan mencari dan memilih data buku yang akan diedit dan mengedit sesuai kebutuhan. 3) Pengelola mengklik tombol simpan untuk menyimpan data buku.	3a) Sistem akan menyimpan data buku yang sudah diedit ke dalam database.
Exception condition		

Tabel 8 Deskripsi *use case* hapus buku

Use case name	Hapus buku	
Scenario	Menghapus buku.	
Brief description	Pengelola akan mencari data buku tertentu yang akan dihapus, sistem akan mengecek, apakah buku tersebut terelasi dengan transaksi peminjaman tertentu. Jika tidak sistem akan menghapus data buku tersebut dari database.	
Actors	Pengelola	
Related use case	Login, olah master data kategori buku, tambah buku	
Stakeholder	Pengelola	
Precondition	Data kategori buku dan buku sudah harus tersimpan dalam database.	
Postcondition	Data buku akan dihapus dari database.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data buku 2) Pengelola akan mencari dan memilih data buku yang akan dihapus 3) Pengelola mengklik tombol hapus untuk menghapus.	3a) Sistem akan mengecek apakah buku tersebut terelasi dengan transaksi peminjaman atau tidak. Jika tidak, maka buku akan dihapus dari database
Exception condition	Jika buku terelasi dengan transaksi peminjaman: <ul style="list-style-type: none"> • Tampilkan pesan. 	

1) *Use case* olah master data anggota

Use case olah master data anggota dapat dirinci ke dalam tiga *use case* utama, yaitu: tambah anggota, edit anggota dan hapus anggota. Deskripsi untuk masing-masing *use case* dapat dilihat pada tabel 9, tabel 10 dan tabel 11.

Tabel 9 Deskripsi *use case* tambah anggota

Use case name	Tambah anggota	
Scenario	Menambah anggota	
Brief description	Pengelola memasukkan data anggota dan sistem akan mengecek apakah data sudah ada dalam sistem atau belum, jika belum data anggota akan disimpan.	
Actors	Pengelola	
Related use case	Login	
Stakeholder	Pengelola	
Precondition		
Postcondition	Data anggota akan disimpan ke dalam sistem.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data anggota. 2) Pengelola memasukkan data anggota dan klik tombol simpan.	2a) Sistem akan mengecek data yang dimasukkan sudah ada atau belum. Jika belum ada, maka sistem akan menyimpan ke dalam database
Exception condition	Jika anggota sudah ada: <ul style="list-style-type: none"> • Tampilkan pesan. 	

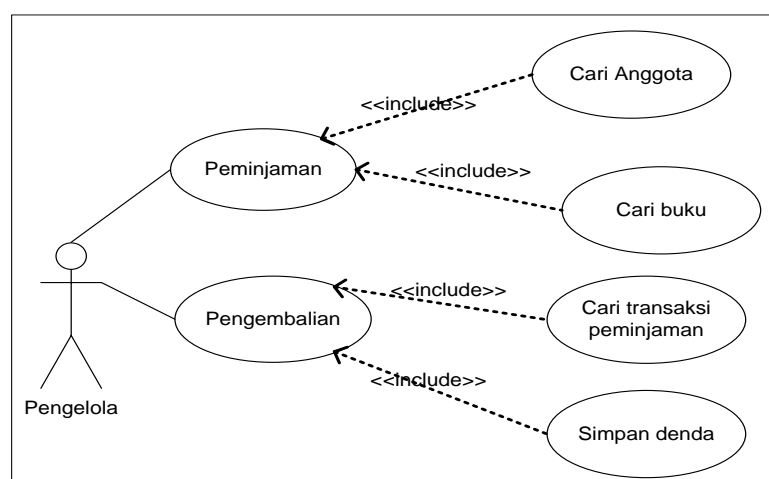
Tabel 10 Deskripsi *use case* edit anggota

Use case name	Edit anggota	
Scenario	Mengedit anggota	
Brief description	Pengelola akan mencari data anggota tertentu yang akan diedit, kemudian data anggota diedit dan disimpan kembali ke dalam sistem.	
Actors	Pengelola	
Related use case	Login, tambah anggota	
Stakeholder	Pengelola	
Precondition	Data anggota sudah harus tersimpan dalam database.	
Postcondition	Data anggota akan disimpan ke dalam database.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data anggota. 2) Pengelola akan mencari dan memilih data buku yang akan diedit dan mengedit data anggota 3) Pengelola mengklik tombol simpan untuk menyimpan	3a) Sistem akan menyimpan data anggota.
Exception condition		

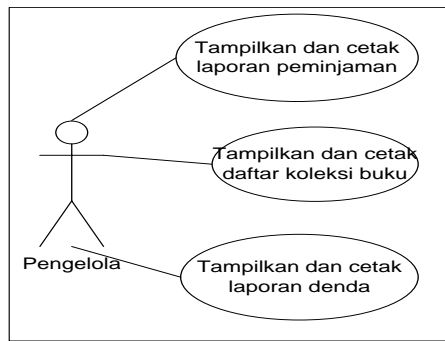
Tabel 11 Deskripsi *use case* hapus anggota

Use case name	Hapus anggota	
Scenario	Menghapus anggota	
Brief description	Pengelola akan mencari data anggota tertentu yang akan dihapus, sistem akan mengecek, apakah buku tersebut terelasi dengan transaksi peminjaman tertentu. Jika tidak sistem akan menghapus data anggota tersebut dari database.	
Actors	Pengelola	
Related use case	Login, olah master data kategori buku, tambah buku	
Stakeholder	Pengelola	
Precondition	Data anggota sudah harus tersimpan dalam database.	
Postcondition	Data anggota akan dihapus dari database.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form master data anggota 2) Pengelola akan mencari dan memilih data kategori anggota yang akan dihapus 3) Pengelola mengklik tombol hapus untuk menghapus.	3a) Sistem akan mengecek apakah anggota tersebut terelasi dengan transaksi peminjaman atau tidak. Jika tidak, maka anggota akan dihapus dari database
Exception condition	Jika anggota terelasi dengan transaksi peminjaman: <ul style="list-style-type: none"> • Tampilkan pesan. 	

Use case diagram untuk transaksi dapat dilihat pada gambar 12.3. Sedangkan *use case diagram* untuk laporan dapat dilihat pada gambar 12.4.



Gambar 12.3 *Use case diagram* transaksi



Gambar 12.4 *Use case diagram* laporan

Pada tabel-tabel di bawah ini disajikan deskripsi untuk *use case* pada transaksi dan laporan.

Tabel 12 Deskripsi *use case* transaksi peminjaman

Use case name	Transaksi peminjaman	
Scenario	Transaksi peminjaman buku oleh anggota	
Brief description	Pengelola memasukkan data transaksi peminjaman buku oleh anggota.	
Actors	Pengelola	
Related use case	Login, olah master data buku, olah master data anggota, cari anggota, cari buku.	
Stakeholder	Pengelola	
Precondition	Data buku dan anggota sudah harus tersimpan dalam database.	
Postcondition	Data transaksi peminjaman akan disimpan dalam database	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form transaksi peminjaman 2) Pengelola memasukkan data anggota dan data buku 3) Pengelola mengklik tombol simpan.	3a) Sistem akan menyimpan data transaksi peminjaman ke dalam database.
Exception condition	1. Jika id anggota salah Tampilkan pesan 2. Jika id buku salah Tampilkan pesan	

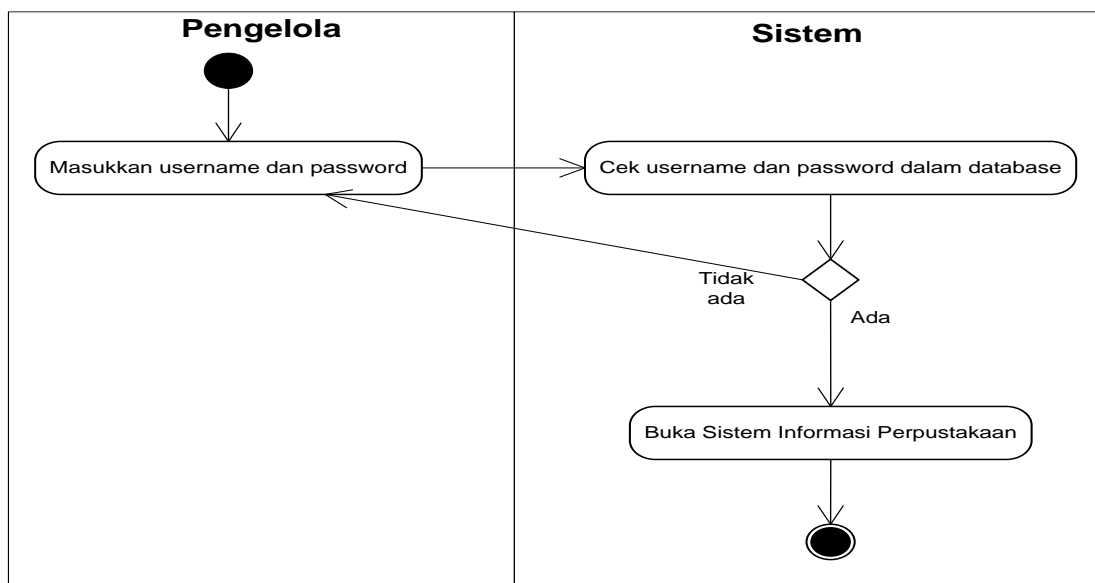
Tabel 13 Deskripsi *use case* transaksi pengembalian

Use case name	Transaksi pengembalian
Scenario	Transaksi pengembalian buku yang dipinjam oleh anggota
Brief description	Pengelola akan memasukkan data pengembalian buku berdasarkan transaksi peminjaman tertentu. Sistem akan menampilkan detail transaksi peminjaman. Sistem akan mengecek apakah terjadi keterlambatan atau tidak. Jika terjadi keterlambatan sistem akan menampilkan denda dan menyimpan transaksi pengembalian dan denda.
Actors	Pengelola
Related use case	Login, transaksi peminjaman, cari transaksi peminjaman, simpan denda
Stakeholder	Pengelola

Precondition	Data transaksi peminjaman sudah harus tersimpan dalam sistem.	
Postcondition	Data transaksi pengembalian akan disimpan dalam sistem. Jika terjadi keterlambatan, data denda akan disimpan dalam sistem.	
Flow of events	<i>Actors</i>	<i>System</i>
	1) Pengelola membuka form transaksi pengembalian 2) Pengelola akan mengklik tombol simpan	1a) Sistem akan menampilkan detail transaksi peminjaman. 1b) Jika terjadi keterlambatan, sistem akan menampilkan informasi denda. 2a) Sistem akan menyimpan transaksi pengembalian 2b) Jika terjadi keterlambatan, sistem akan menyimpan data denda.
Exception condition	Jika transaksi peminjaman tidak ditemukan: <ul style="list-style-type: none"> • Tampilkan pesan. 	

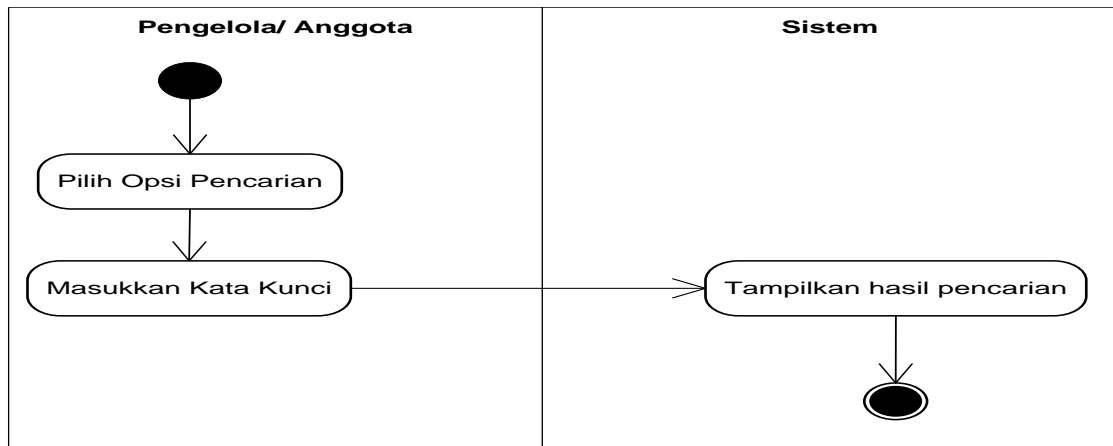
12.3.2 Menggambar Diagram Activity

Langkah selanjutnya setelah *use case diagram* selesai dibuat adalah membuat *activity diagram* untuk setiap *use case*. Gambar 6 menunjukkan *activity diagram* login. Dari gambar dapat dilihat bahwa *actor* pengelola pada saat login memasukkan username dan password. Kemudian sistem akan mengecek apakah username dan password yang dimasukkan terdaftar pada database atau tidak. Jika tidak, *actor* pengelola dapat memasukkan kembali. Jika terdaftar, maka sistem informasi perpustakaan akan terbuka.



Gambar 12.5 Activity diagram login

Activity diagram untuk *use case* pencarian dapat dilihat pada gambar 7. Pada aktivitas pencarian, *actor* pengelola/ anggota harus memilih dulu opsi pencarian kemudian memasukkan kata kunci pencarian. Berdasarkan opsi pencarian dan kata kunci yang dimasukkan, sistem melakukan kueri dan menampilkan hasil pencarian.

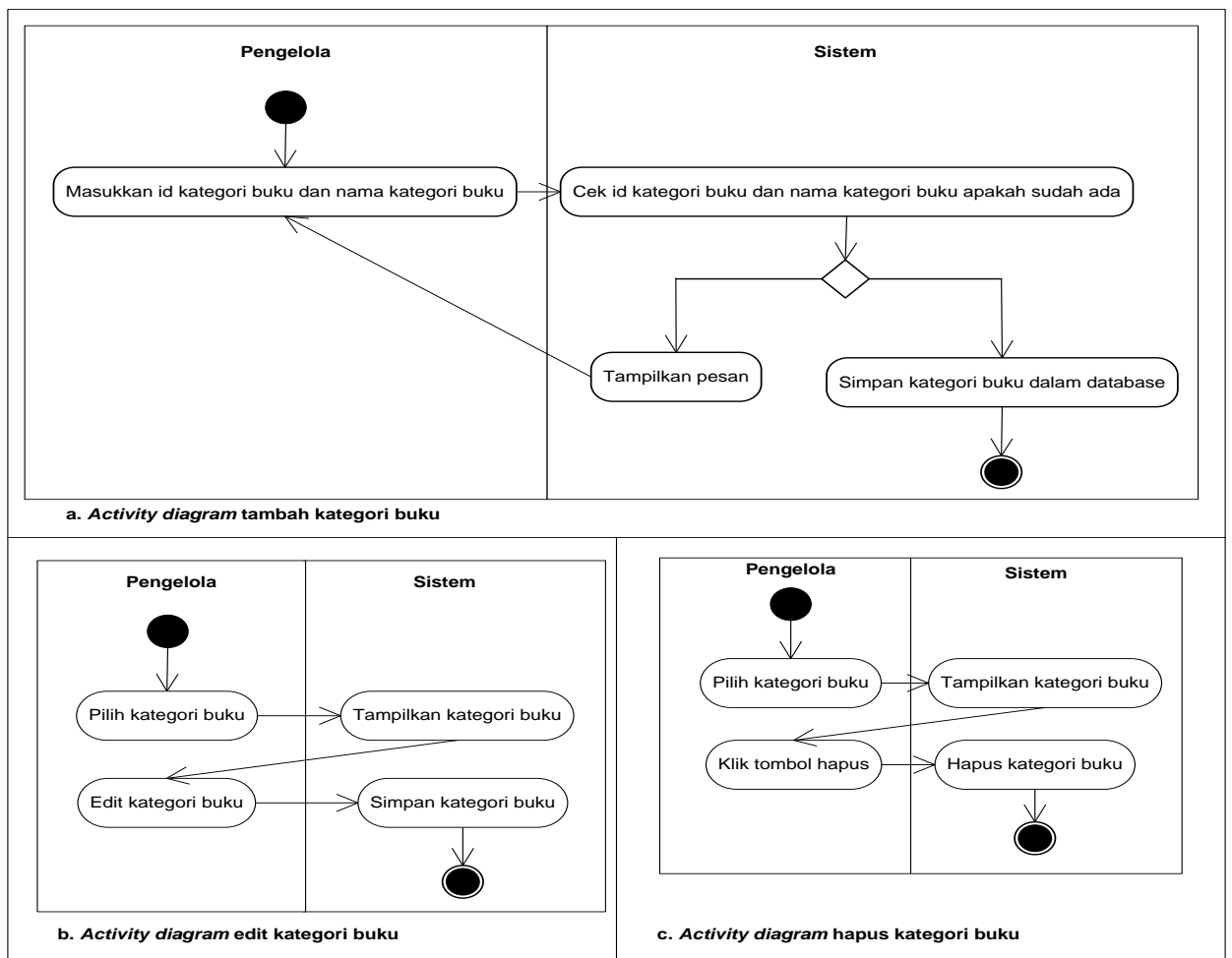


Gambar 12.6 *Activity diagram* pencarian

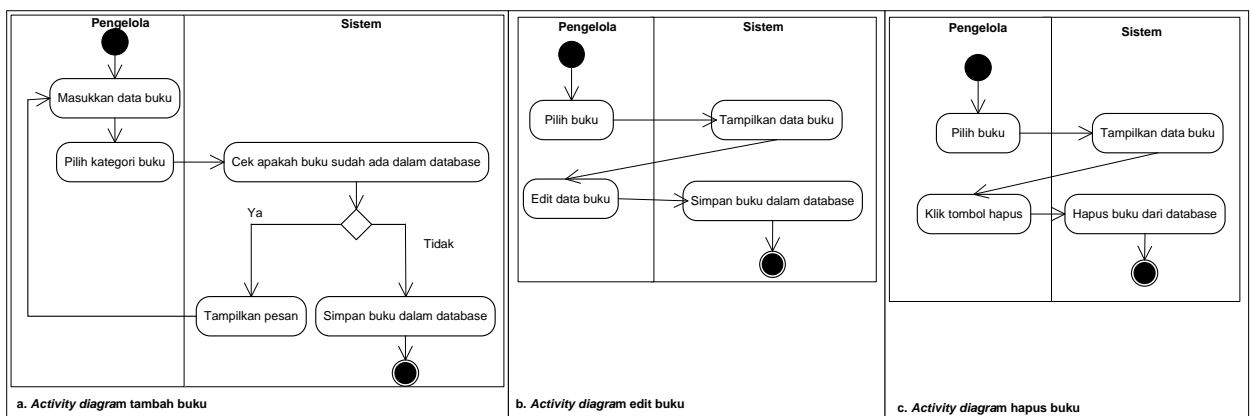
Untuk *activity diagram* olah master data kategori buku dapat dilihat pada gambar 8. Gambar 8a menunjukkan tambah kategori buku, gambar 8b menunjukkan edit kategori buku dan gambar 8c menunjukkan hapus kategori buku. Pada aktivitas menambah kategori buku, setelah pengelola menginputkan id kategori buku dan nama kategori buku, sistem akan mengecek apakah data sudah ada dalam database atau belum. Jika belum, maka sistem akan menyimpan kategori buku dalam database. Sedangkan aktivitas mengedit dan menghapus kategori buku didahului dengan memilih kategori buku tertentu, sistem menampilkan kategori buku yang dipilih kemudian dilakukan proses selanjutnya.

Sama seperti *activity diagram* olah master data kategori buku, *activity diagram* olah master data buku pun terbagi ke dalam tiga bagian, yaitu gambar 9a *activity diagram* tambah buku, gambar 9b *activity diagram* edit buku dan gambar 9c *activity diagram* hapus buku. Aktivitas menambah buku diawali dengan memasukkan data buku dan memilih kategori buku untuk buku yang akan ditambahkan. Hal ini sebagai wujud dari adanya relasi antara buku dan kategori buku.

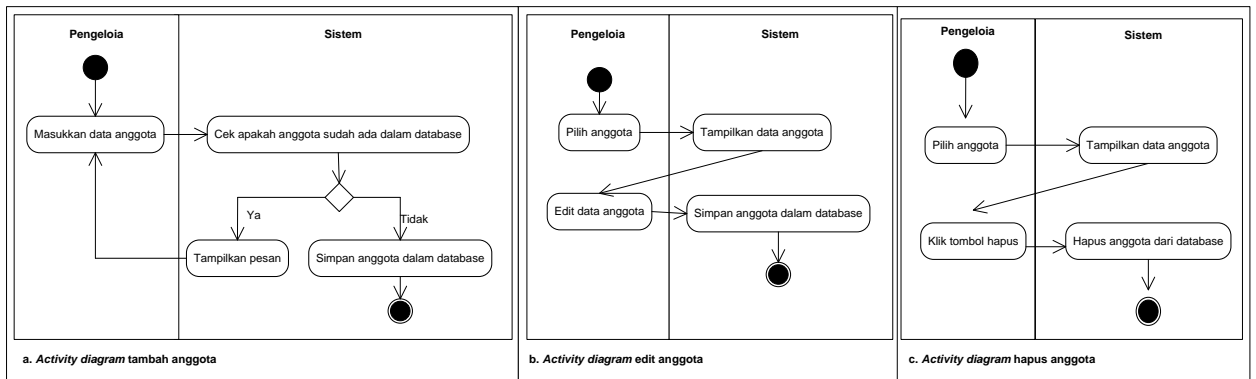
Sedangkan *activity diagram* olah master data anggota dapat dilihat pada gambar 10. Pada gambar 10a dapat dilihat *activity diagram* tambah anggota, gambar 10b *activity diagram* edit anggota dan gambar 10c *activity diagram* hapus anggota. Untuk *activity diagram* transaksi dan laporan dapat dilihat pada gambar 12.7 dan gambar 12.8.



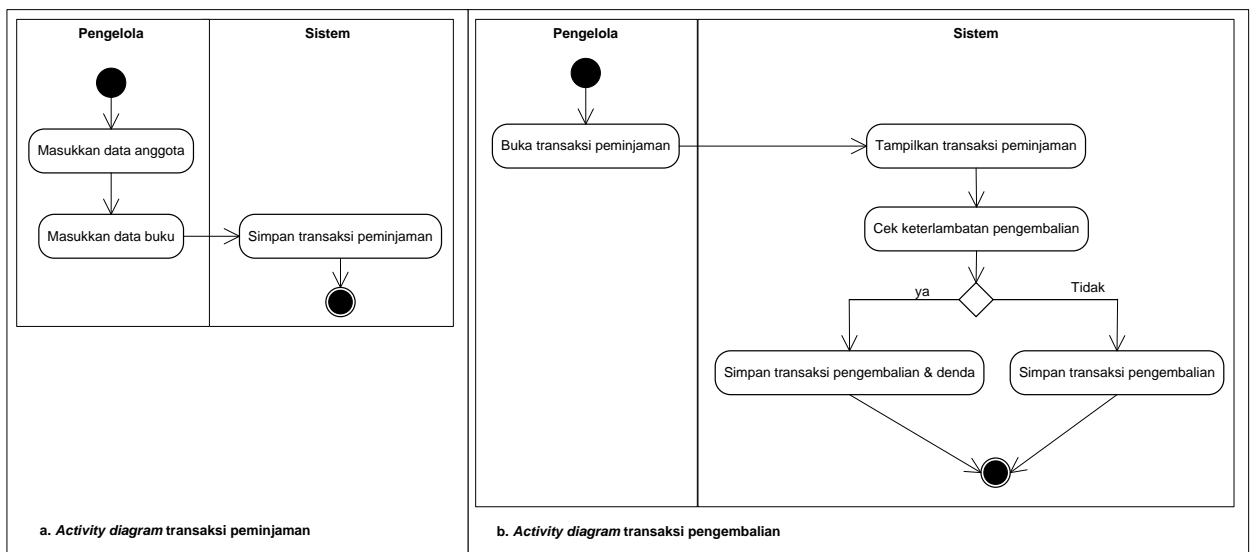
Gambar 12.7 Activity diagram olah master data kategori buku



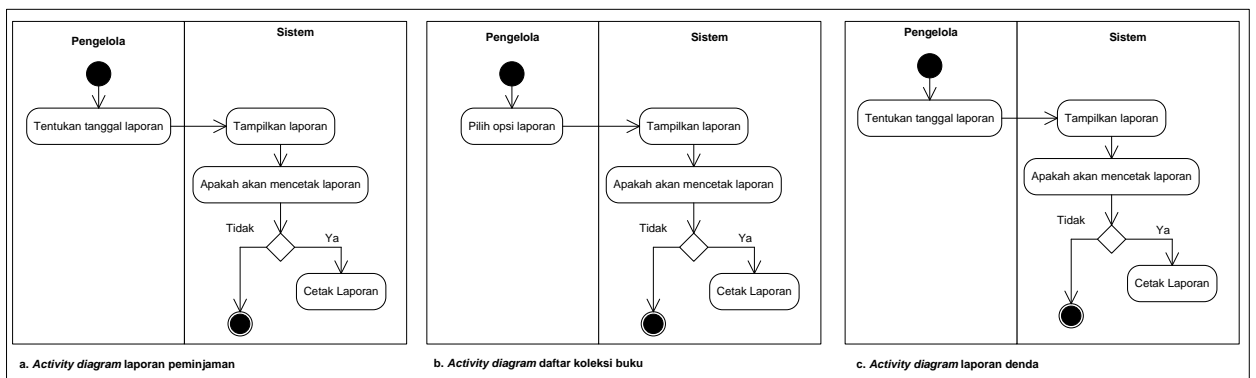
Gambar 12.8 Activity diagram olah master data buku



Gambar 12.9 Activity diagram olah master data anggota



Gambar 12.10 Activity diagram transaksi

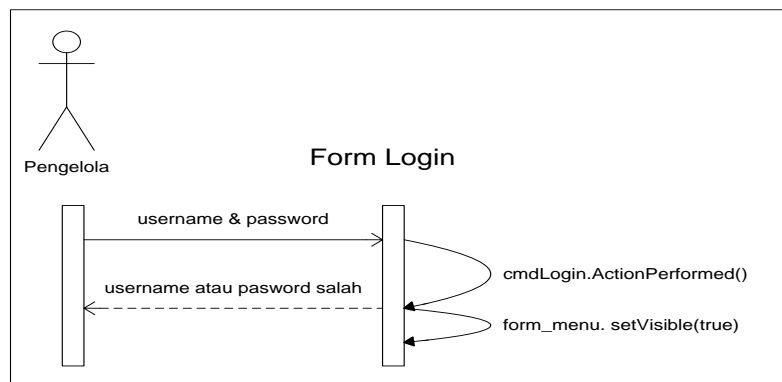


Gambar 12.11 Activity diagram laporan

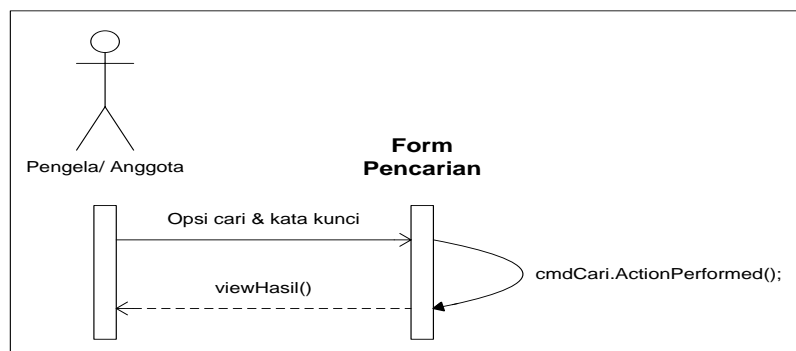
12.3.3 Menggambar Diagram Sequence

Sequence diagram digunakan untuk menjabarkan aktivitas yang ada pada *use case* kepada level yang lebih detail. Pada *sequence diagram* digambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display* dan sebagainya) berupa *message* yang digambarkan terhadap waktu.

Pada gambar 12.12 diperlihatkan *sequence diagram* login. Setelah pengelola memasukkan username dan password kemudian klik tombol login. Sistem akan mengecek apakah username dan password terdaftar. Jika tidak, sistem akan memberikan pesan bahwa username atau password salah. Jika terdaftar maka sistem informasi perpustakaan akan dibuka. Pesan *form_menu.setVisible(true)* merupakan perintah untuk mengaktifkan form menu atau sistem informasi perpustakaan.

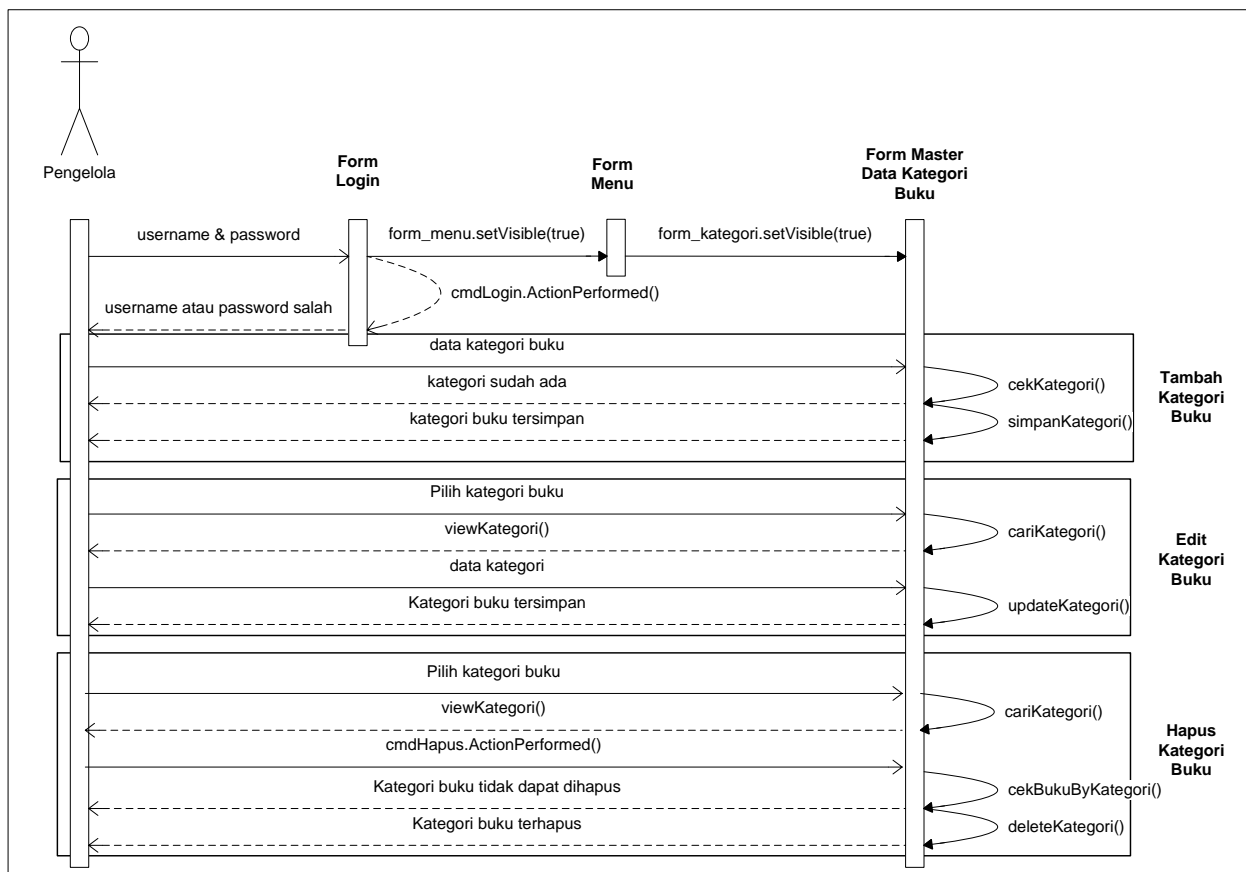


Gambar 12.12 *Sequence diagram* login



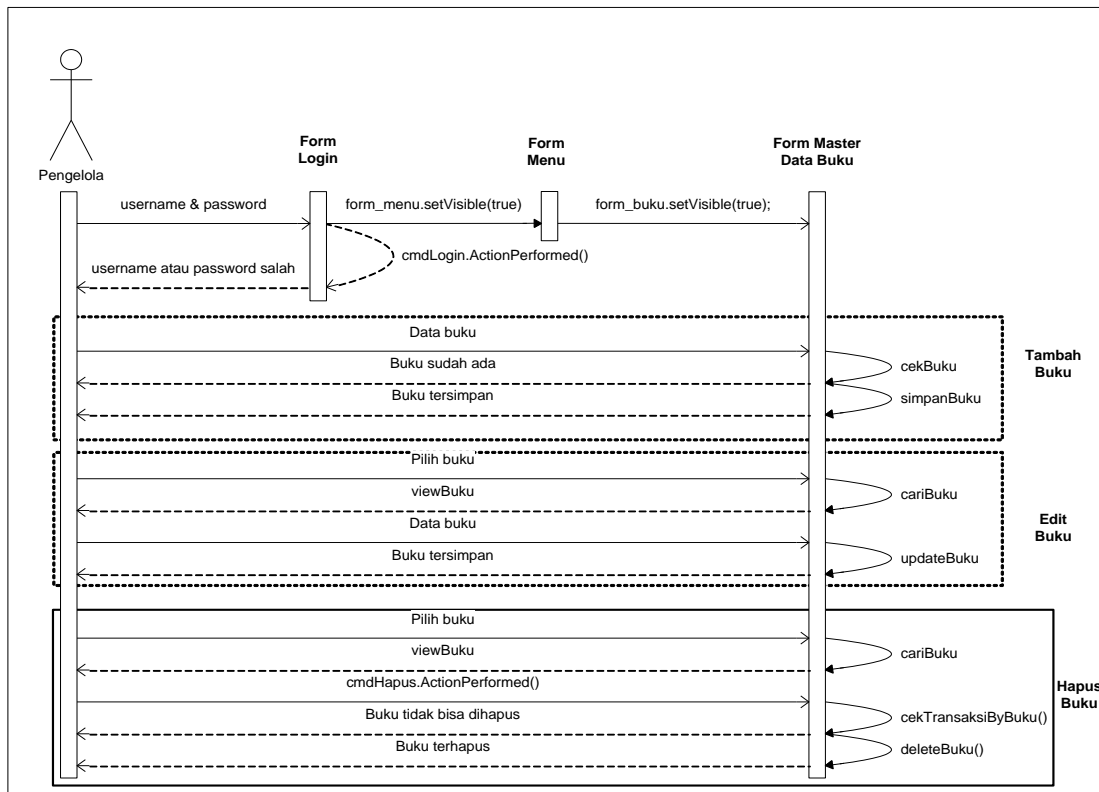
Gambar 12.13 *Sequence diagram* pencarian

Sequence diagram pencarian dapat dilihat pada gambar 12.13. Setelah opsi pencarian dan kata kunci dimasukkan, pengelola/ anggota akan mengklik tombol cari. Sistem akan menjalankan perintah *cmdCari.ActionPerformed()* dan kemudian menampilkan hasil pencarian. Gambar 12.14 menunjukkan *sequence diagram* olah master data kategori buku.

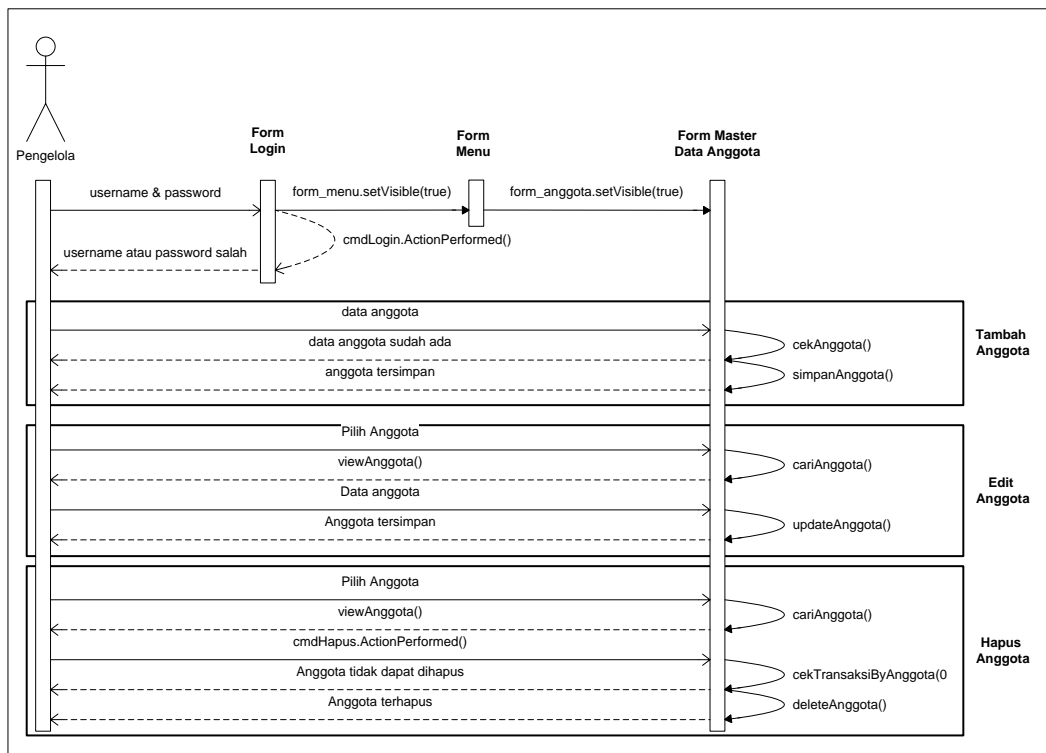


Gambar 12.14 *Sequence diagram* olah master data kategori buku

Pada gambar 12.15 ditunjukkan *sequence diagram* olah master data buku. Pada proses menghapus data data buku, akan dicek dulu apakah buku tersebut pernah digunakan untuk transaksi atau tidak menggunakan *method cekTransaksibyBuku()*. Jika buku pernah dipergunakan dalam transaksi maka akan ditampilkan pesan bahwa buku tidak dapat dihapus. Hal ini dilakukan untuk mencegah terjadinya inkonsistensi data pada database. Pada *sequence diagram* olah master data anggota pada gambar 16, proses penghapusan pun harus dilakukan pengecekan terlebih dahulu apakah anggota tersebut pernah meminjam buku atau tidak menggunakan *method cekTransaksiByAnggota()*.



Gambar 12.15 *Sequence diagram* olah master data buku

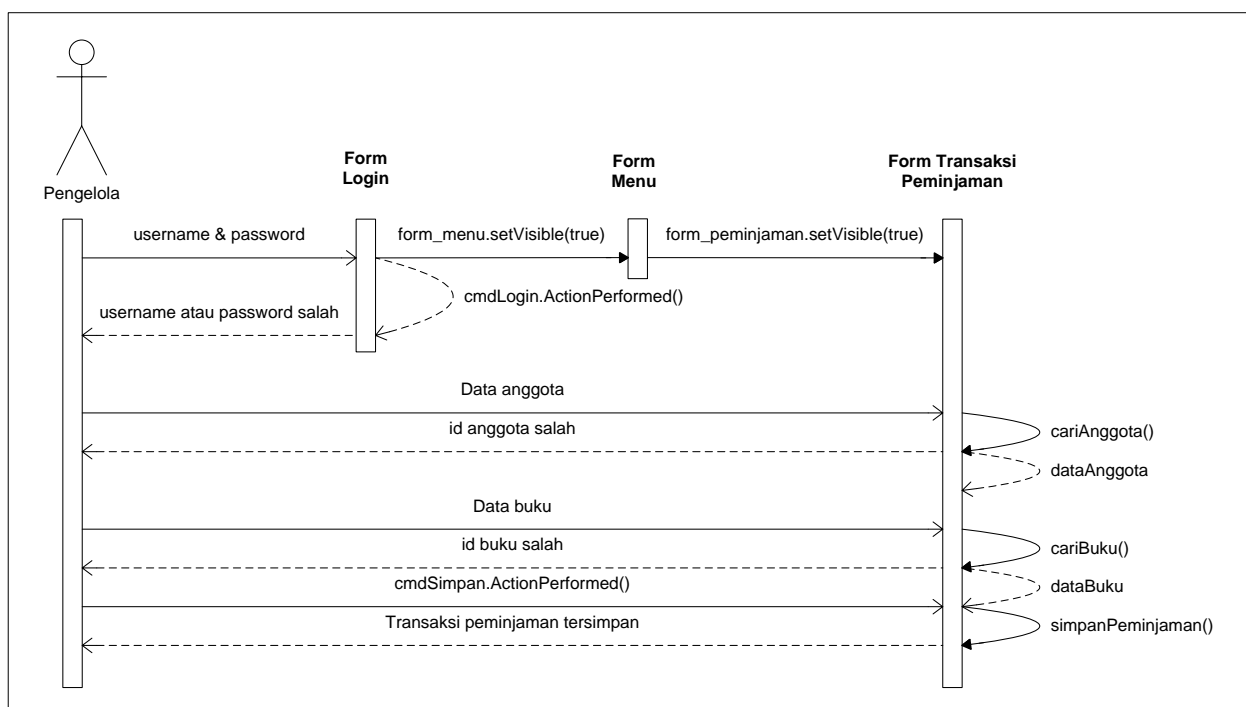


Gambar 12.16 *Sequence diagram* olah master data anggota

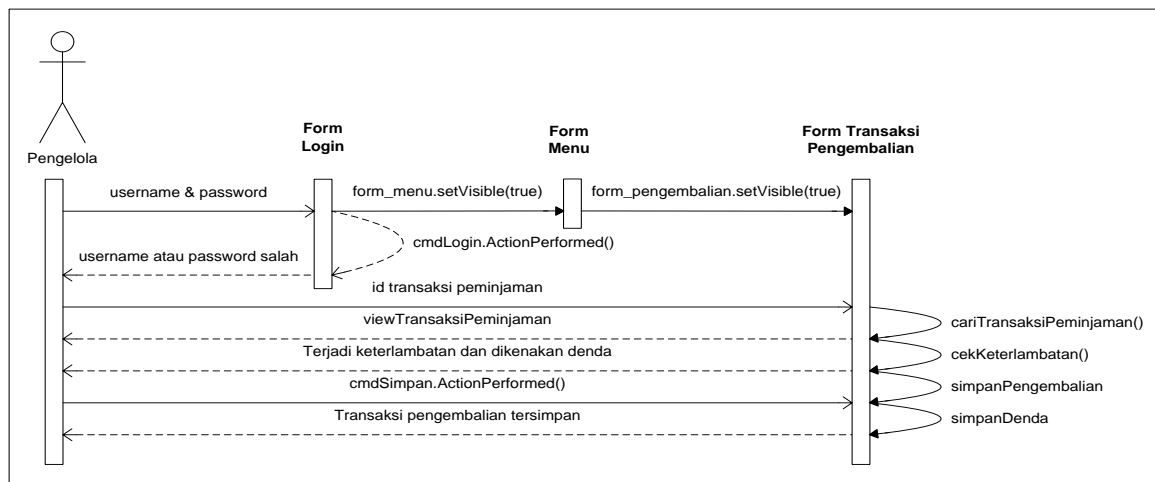
Gambar 12.17 menunjukkan *sequence diagram* transaksi peminjaman. Transaksi peminjaman dilakukan dengan memasukkan data anggota dan data buku. Pada saat proses memasukkan data anggota berupa id anggota. Sistem akan mengecek apakah id anggota terdaftar. Jika tidak terdaftar sistem akan memberikan pesan. Jika terdaftar selanjutnya dimasukkan data buku yang dipinjam menggunakan id buku. Di sinipun sistem akan melakukan pengecekan, apakah id buku terdaftar atau tidak. Jika tidak terdaftar maka sistem akan memberikan pesan.

Sequence diagram transaksi pengembalian ditunjukkan oleh gambar 12.18. Langkah awal dari transaksi pengembalian adalah memasukkan id transaksi peminjaman. Sistem akan mengecek apakah id peminjaman benar atau tidak. Jika benar sistem akan menampilkan detail transaksi peminjaman. Kemudian sistem akan mengecek keterlambatan. Jika terjadi keterlambatan akan ditampilkan pesan. Kemudian transaksi pengembalian dan denda akan disimpan.

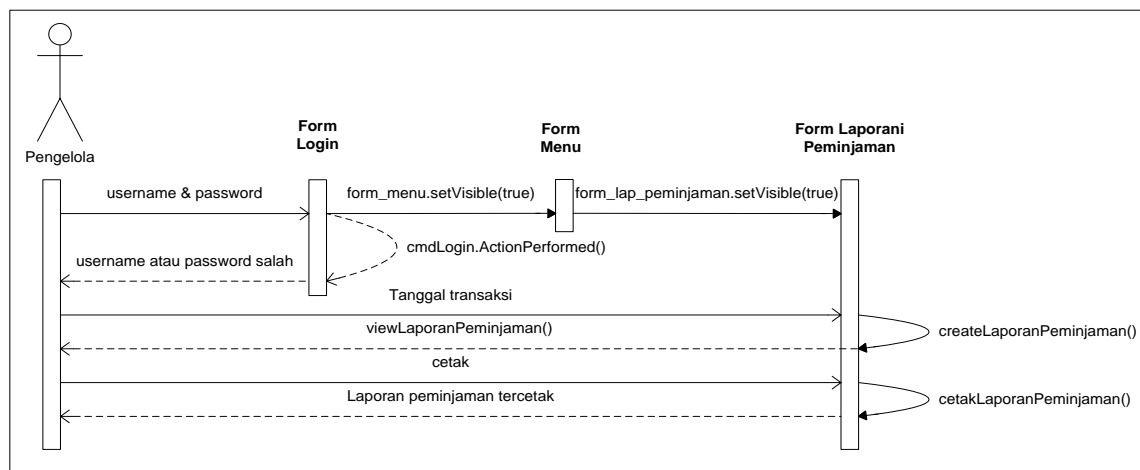
Sequence diagram laporan peminjaman dapat dilihat pada gambar 12.19, untuk daftar koleksi buku dapat dilihat pada gambar 12.20 dan laporan denda dapat dilihat pada gambar 12.21.



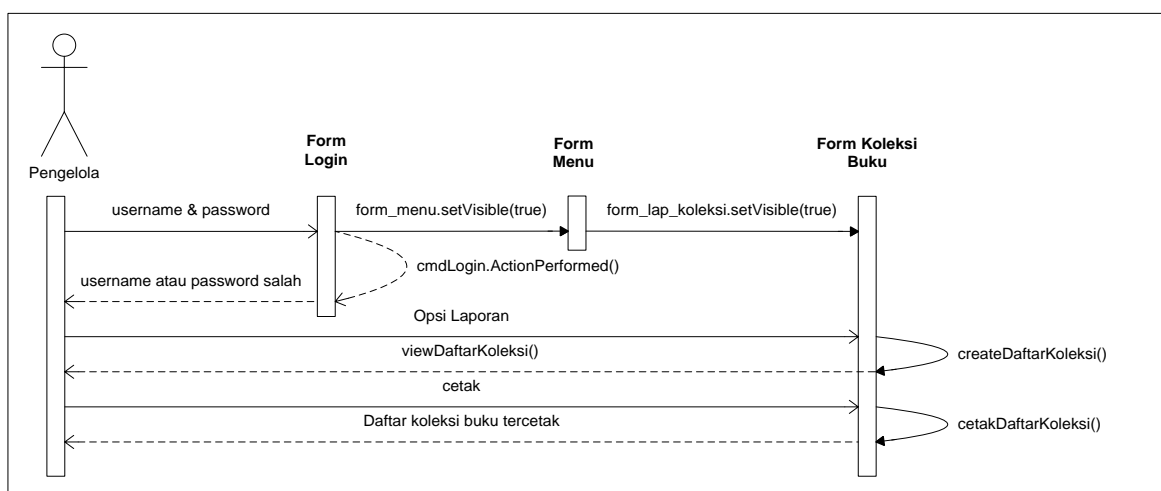
Gambar 12.17 *Sequence diagram* transaksi peminjaman



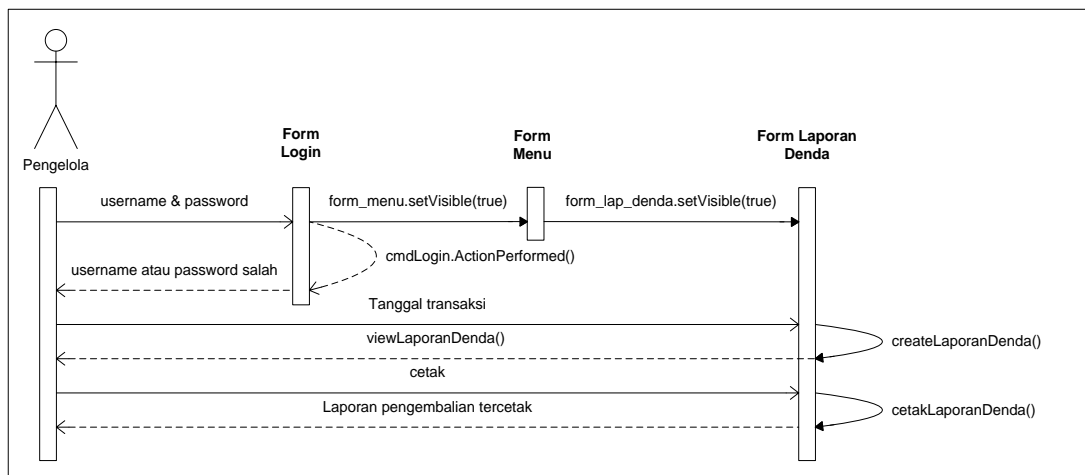
Gambar 12.18 *Sequence diagram* transaksi pengembalian



Gambar 12.19 *Sequence diagram* laporan peminjaman

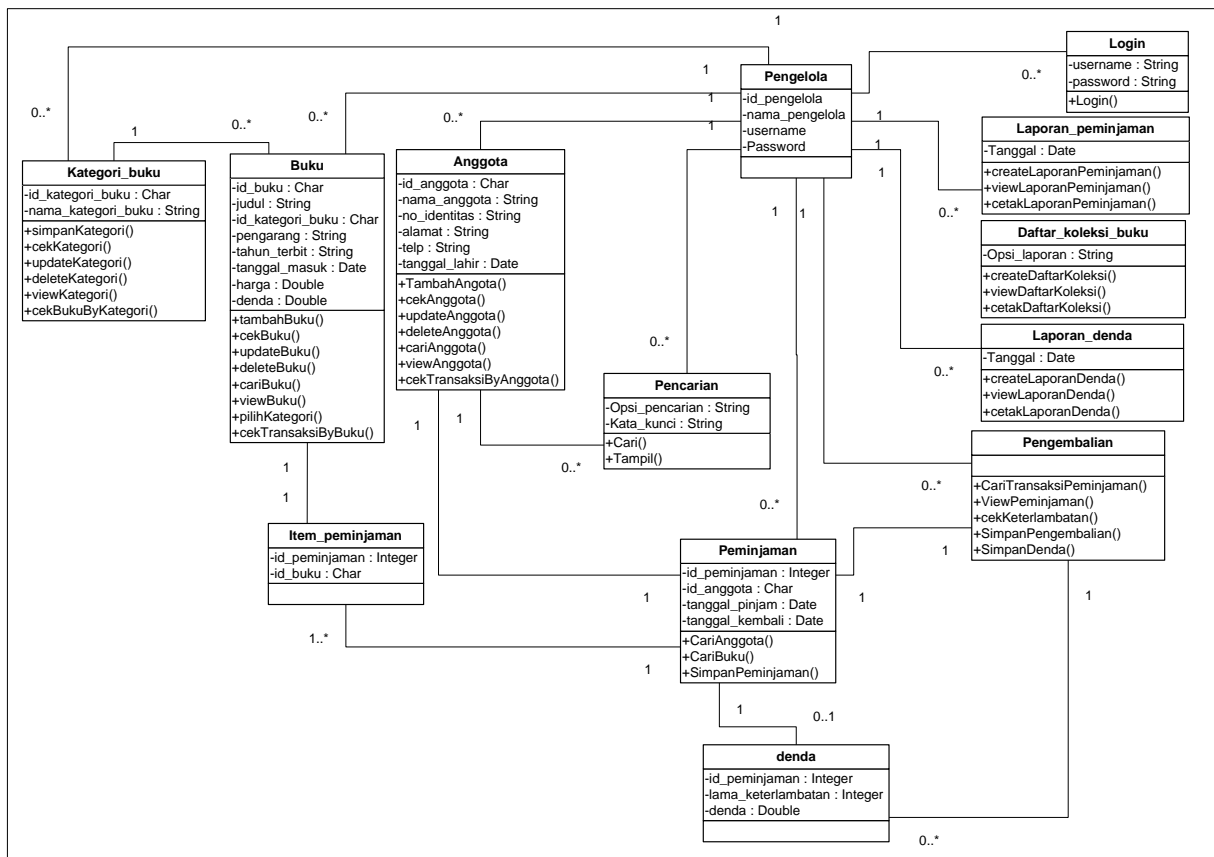


Gambar 12.20 *Sequence diagram* daftar koleksi buku



Gambar 12.21 *Sequence diagram* laporan denda

12.3.4 Menggambar Diagram Class



Gambar 12.22 Diagram Class

DAFTAR PUSTAKA

1. Al-Bahra Bin Ladjamudin, Analisis dan Desain Sistem Informasi, Graha Ilmu, Yogyakarta, 2005
2. Jogiyanto H.M, *Analisis & Desain Sistem informasi*, Penerbit Andi Offset. Yogyakarta, 2008
3. Munawar, Pemodel Visual Dengan UML, Graha Ilmu, Yogyakarta, 2005
4. Rosa A.S, Modul Pemrograman Berorientasi Objek, Modula, Bandung, 2010
5. Yuni Sugiarti, Analisis dan Perancangan UML, Graha Ilmu, Yogyakarta, 2013