BAB 2
TINJAUAN PUSTAKA

2.1. Profil Tempat Kerja Praktek

2.1.1. Sejarah Instansi

PT. Ihsan Solusi Informatika (ISI) berdiri pada tahun 2002 dan berlokasi di Jakarta dan Bandung. ISI berkomitmen untuk memberikan solusi terbaik kepada para *client* dengan menggunakan teknologi informasi terkini, dimana tidak hanya bekerja sesuai kebutuhan tetapi juga dengan metode yang *innovative*.

ISI hadir dalam era teknologi informasi ini sebagai partner yang akan membimbing dalam implementasi teknologi informasi, khusunya dalam dunia *software* (perangkat lunak).

ISI didirikan dalam semangat melayani dan memberikan kualitas pengembangan teknologi *software* terbaik. Pengalaman para pendiri dan staf ISI dalam mengembangkan *software* selama ini diharapkan dapat menjadi aset yang bernilai untuk membantu dalam menarikmanfaat maksimal aplikasi teknologi informasi.

Sebagai perusahaan berebasis *knowleedge* dan teknologi, diharuskan merumuskan kompetensi utama yang sesuai dengan visi misi perusahaan, yaitu:

1. Disiplin pengembangan *software (software engeieneering)* khususnya pada aspek pemodelan sistem dan manajemen pengembangan sistem.
2. Arsitektur sistem perangkat lunak dan perangkat-perangkat teknologi intinya.

3. Pengetahuan tentang *business knowledge* khususnya dibidang jasa finansial.

2.1.2. Logo Instansi

Logo merupakan simbol, tanda gambar, merek dagang (*trademark*) yang berfungsi sebagai lambang identitas diri dari suatu badan usaha dan tanda pengenal yang merupakan ciri khas perusahaan. Gambar dibawah merupakan logo dari PT. Ihsan Solusi Informatika.



Gambar 2.1 Logo Perusahaan

2.1.3. Visi dan Misi Perusahaan

2.1.3.1. Visi Perusahaan

Menyediakan layanan solusi perangkat lunak yang ekonomis dan tepat guna sebagai bagian integeral dari solusi teknologi informasi, yang ditunjang secara internal oleh kualitas SDM selalu meningkat serta dukungan riset dan pengembangan teknologi.

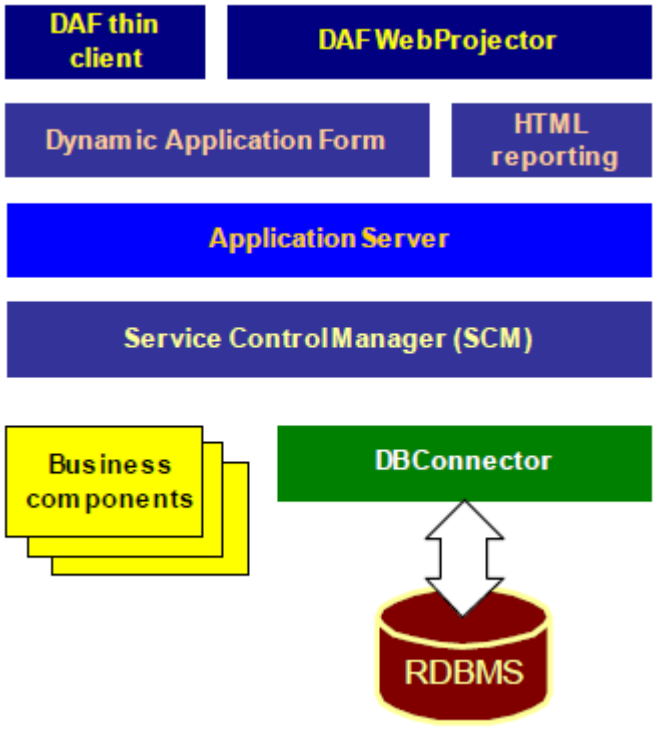
2.1.3.2.Misi Perusahaan

1. Mengembangkan berbagai solusi baik berupa solusi siap pakai maupun *customized development* khususnya dibidang jasa finansial.
2. Menciptakan arsitektur dan teknologi aplikasi yang handal, baik dari segi *development* maupun *deployment* sebagai pondasi utama bagi solusi yang akan dikembangkan untuk *customer.*
3. Meningkatkan kualitas sumber daya manusia sebagai lini depan pelayanan kepada *customer.*

2.1.4. Portofolio Teknologi

Sebagai perusahaan yang kompetensi intinya adalah pengembangan aplikasi, ISI telah mengembangkan suatu kerangka kerja (*framework*) pengembangan aplikasi yang diberi nama DAF (*Database Application Framework).*

DAF adalah suatu *framework* pengembangan aplikasi internet berbasis *database.* DAF menyediakan berbagai fasilitas agar pengembangan aplikasi bersifat modular dan mudah dikelola.



Gambar 2.2 Portofolio Teknologi

Berikut adalah ringkasan dari fitur – fitur yang tersedia dalam DAF untuk setiap level pengembangan aplikasi:

1. Level akses basisdata *(database access)*: DAF menyediakan suatu arsitektur yang bernama DBConnector. DBConnector adalah protokol akses basisdata yang bersifat ringan *(lightweight)* dan *product-neutral*, dengan memanfaatkan DBConnector ini maka aplikasi dapat diimplementasikan dengan mudah diberbagai RDBMS tanpa mengalami perubahan kode.
2. Level model data: DAF meneyediakan arsitektur berbasis objek, dengan teknik ini aplikasi yang besar/rumit dapat di*breakdown* ke dalam objek-objek secara modular. DAF meneyediakan pula akses data secara objek tanpa melalui SQL dan *metadata-driven query.*

3. Level antar-muka *(user-interface)*: DAF menyediakan deskripsi *user-interface* yang familiar dengan berbagai design pattern dalam aplikasi berbasis basisdata.
4. Level penyebaran dan implementasi (*deployment and implementation*): DAF menyediakan *application server* baik untuk kebutuhan arsitektur *thin-client* maupun arsitektur *webbased*.

2.1.5. Portofolio Project

Kualitas layanan dan produk ditunjang oleh pengalaman yang sarat selama bertahun-tahun, termasuk pengalaman para pendiri perusahaan sebelum perusahaan ini dibentuk. Berbagai pengalaman telah memberikan wawasan yang luas baik dari sisi pemahaman kepentingan bisnis di sisi *client* maupun dari sisi pengembangan teknologi dan proses pengembangan.

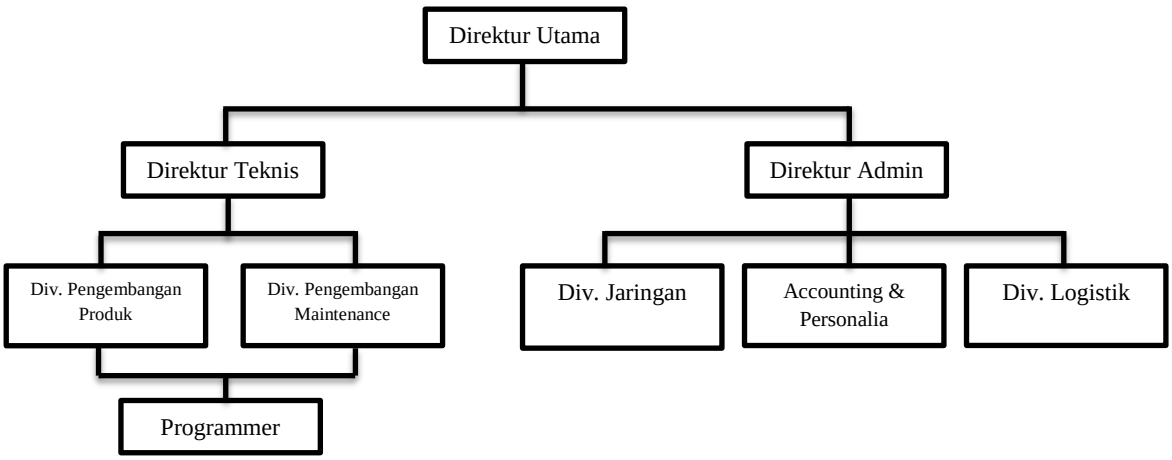
Berikut ini beberapa portofolio project yang telah kami tangani:

1. Pengembangan dan maintenance sistem aplikasi back office DPLK PT Bank Muamalat Indonesia (2000 – 2003).
2. Upgrading sistem informasi back office DPLK PT Bank Muamalat Indonesia dari clientserver ke arsitektur 3-tier dan koneksi dengan messaging ke sistem KIBLAT (2003-2004).
3. Pengembangan dan implementasi aplikasi manajemen fixed asset, BDD dan inventory di PT. Bank Muamalat Indonesia (2004 – 2005).
4. Pengembangan dan implementasi aplikasi pelaporan dan enterprise worksheet system di PT. Bank Muamalat Indonesia (2005).

5. Maintenance and support aplikasi pendukung di lingkungan PT Bank Muamalat Indonesia (2005 – 2008).
6. Pengembangan dan implementasi aplikasi Three Tier KIBLAT Reengineering (T2KR) di PT. Bank Muamalat Indonesia (2008 – 2009).
7. Maintenance and support core banking pada PT Bank Muamalat Indonesia (2010 – 2011).
8. Pengembangan aplikasi pembiayaan Bank Muamalat Indonesia (2010- 2011).
9. Pengembangan sistem informasi dan manajemen K3 pada PT Komatsu Indonesia (2011).

2.1.6. Struktur Organisasi dan Job Description

2.1.6.1.Struktur Organisasi



Gambar 2.3 Stuktur Organisassi

2.1.6.2.Job Description

Deskripsi Kerja CV Ihsan Solusi Informatika berdasarkan susunan organisasi.

1. Direktur Admin

   Direktur Admin bertugas sebagai orang yang mengesahkan pengeluaran keuangan di PT. Ihsan Solusi Informatika.

2. Direktur Teknis

   Direktur teknis bertugas untuk menganalisis dan mengimplementasikan berbagai system *software* akutansi dan keuangan sekaligus menguji program di PT. Ihsan Solusi Informatika.

3. Divisi Pengembangan Produk

   Divisi ini khusus membuat rancangan produk yang sudah dibuat sesuai dengan keinginan pasar atau keinginan konsumen.

4. Divisi Pengembangan Maintenance

   Divisi ini bertugas untuk meningkatkan kualitas perawatan pada produk yang sudah dibuat oleh programmer.

5. Divisi Jaringan

   Divisi ini bertugas untuk mengembangkan dan menciptakan produk yang berhubungan dengan jaringan dan memasarkan semua produk yang ada di PT. Ihsan Solusi Informatika.

6. Accounting & Personalia

   Mengembangkan *software, accounting, financial management system*, dan memprakarsai pengembangan DAF sebagai framework teknologi.

7. Divisi Logistik

   Divisi ini bertugas untuk menyiapkan segala kebutuhan logistik karyawan yang dibutuhkan untuk membuat suatu produk yang dihasilkan oleh PT. Ihsan Solusi.

8. Programer

   Core Developer bertugas untuk mendesain aplikasi databasebase.

2.2. Landasan Teori

2.2.1. Data

2.2.1.1.Pengertian Data

Data adalah sumber informasi yang bentuknya masih mentah. Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata dan diperoleh dalam bentuk simbol-simbol karakter huruf, angka, gambar, suara, sinyal, dan lain sebagainya[2].

2.2.1.2.Jenis Data

Adapun jenis data dapat dibedakan menjadi beberapa kategori

1. Berdasarkan sumber data
   a. Data primer

      Data primer adalah secara langsung diambil dari objek / obyek penelitian oleh peneliti perorangan maupun organisasi.

   b. Data sekunder

      Data sekunder adalah data yang didapat tidak secara langsung dari objek penelitian.

2. Berdasarkan tampilan data

   a. Data Kuantitatif

      Data kuantitatif adalah data yang dipaparkan dalam bentuk angka-angka.

   b. Data Kualitatif

      Data kualitatif adalah data yang disajikan dalam bentuk kata-kata yang mengandung makna.

2.2.2. Sistem Informasi

2.2.2.1. Penegertian Sistem Informasi

Ruang lingkup sistem informasi sebenarnya tertuang pada dua kata pembentuknya, yaitu “sistem” dan “informasi”.

1. Sistem merupakan kumpulan elemen yang saling berhubungan satu sama lain yang membentuk satu kesatuan dalam usaha mencapai satu tujuan. Di dalam perusahaan, yang dimaksud sistem adalah departemen internal, seperti persediaan barang mentah, produksi, persediaan barang jadi, promosi, penjualan, keuangan, personalia, serta pihak internal dan eksternal seperti supplier dan konsumen yang saling terkait satu sama lain dan membentuk kesatuan usaha.
2. Informasi adalah hasil pemrosesan data yang diperoleh dari elemen sistem tersebut jadi bentuk yang mudah dipahami dan pengetahuan yang relevan yang dibutuhkan untuk menambah pemahaman mengenai fakta-fakta yang ada. Informasi bagi setiap elemen akan berbeda satu sama lain sesuai kebutuhan masing-masing.

2.2.2.2. Fungsi Sistem Informasi

Adapun beberapa kemampuan sistem informasi adalah sebagai berikut:

1. Komputasi numerik, bervolume besar, dengan kecepatan tinggi.
2. Menyediakan komunikasi dalam organisasi atau antar organisasi yang murah akurat dan cepat.
3. Menyimpan informasi dalam jumlah yang sangat besar dalam ruang yang kecil tetapi mudah diakses
4. Memungkinkan pengaksesan informasi yang sangat banyak di seluruh dunia dengan cepat dan murah
5. Menyajikan informasi dengan jelas yang menggugah pikiran manusia
6. Mengotomasikan proses-proses bisnis yang semiotomatis dan tugas-tugas yang dikerjakan secara manual.

2.2.3. DBMS (Database Management System)

Database (basis data) dapat diartikan sebagai kumpulan data. Secara manual dapat berupa lemari arsip, sedangkan secara komputerisasi, basis data merupakan *file archive* [4].

Secara umum basis data berkaitan dengan pengarsipan data secara komputerisasi sehingga memudahkan dalam pengaksesan data pada suatu waktu dengan cepat.

Pengelolaan basis data secara fisik tidak dilakukan oleh pemakai secara langsung, tetapi ditangani oleh sebuah perangkat lunak yang khusus. Perangkat lunak inilah disebut DBMS yang akan menentukan bagaimana data diorganisasi,

disimpan, diubah dan diambil kembali. DBMS juga menerapkan mekanisme pengamanan data, pemakaian data secara bersama, pemaksaan keakuratan atau konsistensi data, dan sebagainya.

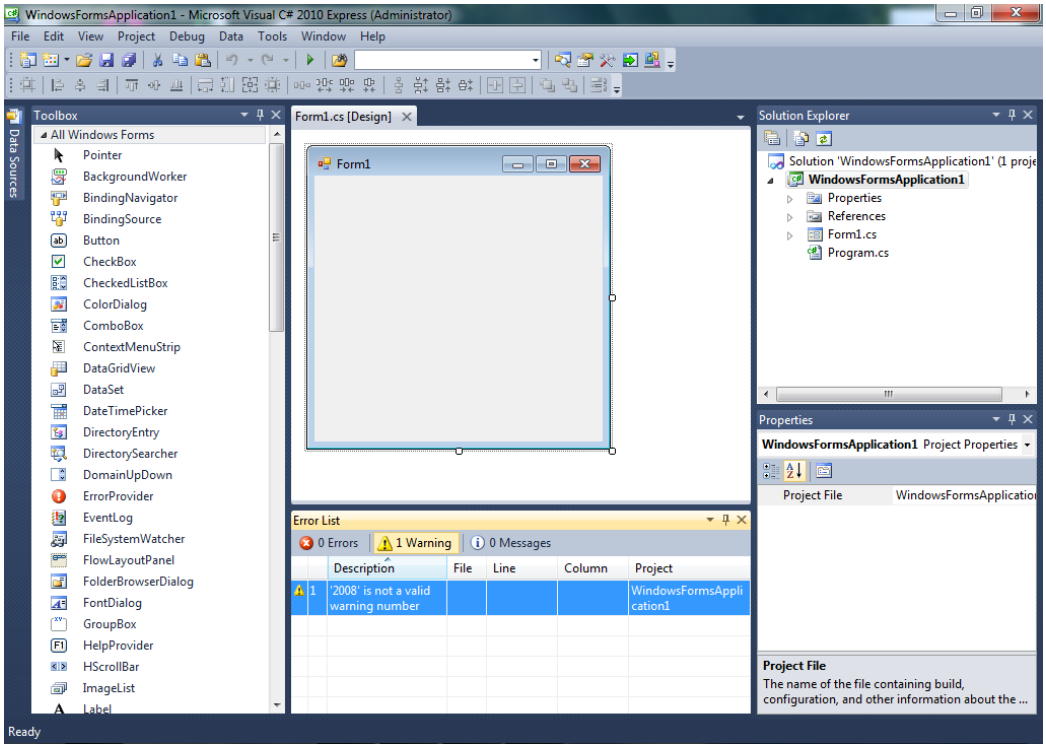
Perangkat lunak yang termasuk DBMS seperti dBase III+, dBase IV, FoxBase, Rbase, MS-Access dan Borland-Paradox (untuk kelas sederhana) atau Borland-Interbase, MS-SQLServer, CA-Open Ingres, Oracle, Informix dan Sybase (untuk kelas kompleks/berat).

2.2.4. Microsoft Visual C# 2010

Microsoft Visual C# 2010 atau *Visual C#(C Sharp)* adalah bahasa pemrograman yang bekerja dalam sistem operasi Windows yang merupakan bahasa pemrograman yang mudah dipahami karena menggunakan struktur pemrograman bahasa C. *Visual C#* juga menyediakan berbagai *tools* yang dapat mempermudah dalam membuat suatu aplikasi, diantanya fasilitas *web browser*, yang digunakan untuk membuat *browser*, sehingga akan mempermudah dalam pembuatan *web browser*.

Visual C# mendukung pemrograman berbasis objek atau biasa disebut OOP(*Object Oriented Programing*) dan memudahkan *programmer* untuk membuat aplikasi berbasis objek serta mendukung aplikasi berbasis *.NET*. Selain itu, *Visual C#* juga menyediakan fasiltas koneksi ke *database MySQL* dengan menggunakan konektor tambahan sebagai penghubungnya dan mendukung perintah – perintah *MySQL*.

Berikut ini gambar lingkungan kerja pada *Visual C# 2010*:



Gambar 2.4 Tampilan Lingkungan Kerja VIsual C# 2010

2.2.5. MySQL

MySQL adalah aplikasi *SQL database server* yang multi user. Oleh karena itu, *MySQL* digunakan dalam pembuatan aplikasi ini yang digunakan sebagai *database server* untuk menyimpan data pegawai pada SIAP.

MySQL adalah aplikasi *database* yang mudah digunakan karena telah banyak digunakan di berbagai aplikasi dan *websites*. *MySQL* juga mudah untuk dipadukan dengan aplikasi *Visual C#* karena *MySQL* menyediakan konektor untuk aplikasi pemrograman *.NET* untuk mendukung *MySQL* berjalan dengan baik di *Visual C#*.

2.3. Analisa dan perancangan berorientasi obyek

Analisa dan desain berorientasi obyek berarti merumuskan dan menyelesaikan masalah serta menghasilkan suatu hipotesa dan diagnosa (solusi), memodelkannya dengan pendekatan/paradigma obyek (obyek adalah riil punya atribut/data dan perilaku)[3].

Dalam melakukan analisa dan perancangan sistem berorientasi obyek penulis menggunakan UML (*Unified Modelling Language*) untuk memodelkannya. Sedangkan alat (*tool*) visual *modelling* yang digunakan untuk menggambarkan model analisa dan perancangan adalah Microsoft Visio 2007. Implementasi perangkat lunak menggunakan bahasa pemrograman PHP.

2.3.1. Unified Modelling Language (UML)

Pada Oktober 1994 Dr. James Rumbaugh yang mengembangkan *Object Modelling Technique* (OMT) bergabung dengan perusahaan Rational *Software*. Sebelumnya juga bergabung Grady Booch yang mengembangkan *Object Modelling Design* (OOD). Duet mereka pada Oktober 1995 menghasilkan *Unified Method* versi 0.8, yang menjadi cikal bakal dari UML sebagai bahasa pemodelan standar untuk aplikasi *object oriented*[3].

Pada tahun 2002 lahir UML versi 2.0 dengan penambahan dan penggantian diagram menjadi 13 buah diagram. Diagram-diagram ini terbagi menjadi 3 kategori :

a. *Structural diagrams* : menggambarkan elemen dari spesifikasi yang mengabaikan waktu. Terdiri dari : *Class Diagram, Object Diagram, Component Diagram, Deployment Diagram, Composite Structure Diagram* dan *Package Diagram.*

b. *Behavior diagram* : menggambarkan ciri-ciri *behavior/method/function* dari sebuah *system* atau *business process*. Terdiri dari : *Use case Diagram, Activity Diagram* dan *State Machine Diagram*.

c. *Interaction diagram* : bagian dari *behavior* diagram yang menggambarkan *object interactions*. Terdiri dari : Communication Diagram, *Interaction Overview Diagram, Sequence Diagram* dan *Timing Diagram*.

Karena UML sangat fleksibel, ada juga cara melihat diagram UML berdasar kategori berikut :

a. *Static Diagram* : menunjukkan segi *static* dari *system*. Kategori ini sama dengan *structural* diagram.

b. *Dynamic Diagram* : menunjukkan bagaimana *system* berkembang setiap waktu. Meliputi *state-machine* diagram dan *timing* diagram.

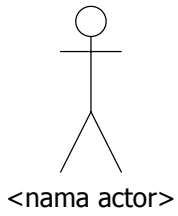
Functional Diagram : menunjukkan detail dari perilaku (*behavior*) dan algoritma bagaimana system memenuhi perilaku yang diinginkannya. Kategori ini termasuk *use case*, *interaction* dan *activity diagram*.

2.3.1.1. *Use case Diagram*

Use case diagram menggambarkan kebutuhan sistem dari sudut pandang *user*. Digunakan untuk menggambarkan hubungan antara internal sistem dan eksternal sistem atau hubungan antara *use case* dan aktor.

1. *Actor*

Actor adalah sesuatu (entitas) yang berhubungan dengan sistem dan berpartisipasi dalam *use case*. *Actor* menggambarkan orang, sistem atau entitas eksternal yang secara khusus membangkitkan sistem dengan *input* atau masukan kejadian-kejadian, atau menerima sesuatu dari sistem. *Actor* dilukiskan dengan peran yang mereka mainkan dalam *use case*, seperti Staff, Kurir dan lain-lain.



Gambar 2.5 Bentuk *Actor* dalam UML

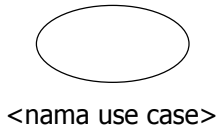
Dalam *use case* diagram terdapat satu aktor pemulai atau *initiator* actor yang membangkitkan rangsangan awal terhadap sistem, dan mungkin sejumlah aktor lain yang berpartisipasi atau *participating* actor. Akan sangat berguna untuk mengetahui siapa aktor pemulai tersebut.

2. *Use case*

Use case yang dibuat berdasar keperluan aktor merupakan gambaran dari "apa" yang dikerjakan oleh sistem, bukan "bagaimana" sistem mengerjakannya.

Use case diberi nama yang menyatakan apa hal yang dicapai dari interaksinya dengan aktor.

Dalam UML *use case* dinotasikan dengan gambar :

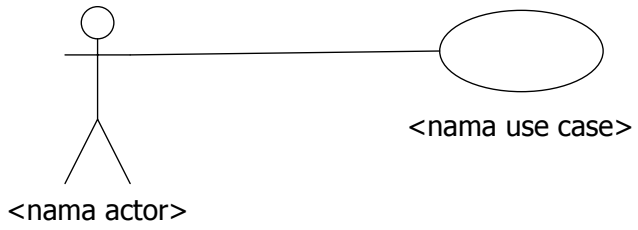


Gambar 2.6 Bentuk Use case dalam UML

3. *Relationship*

Relasi (*relationship*) digambarkan sebagai bentuk garis antara dua simbol dalam *use case* diagram. Relasi antara *actor* dan *use case* disebut juga dengan asosiasi (*association*). Asosiasi ini digunakan untuk menggambarkan bagaimana hubungan antara keduanya.

Relasi-relasi yang terjadi pada *use case* diagram bisa antara *actor* dengan *use case* atau *use case* dengan *use case*.



Gambar 2.7 Bentuk *Relationship* dalam UML

Relasi antara *use case* dengan *use case* :

a. *Include*, pemanggilan *use case* oleh *use case* lain atau untuk menggambarkan suatu *use case* termasuk di dalam *use case* lain (diharuskan). Contohnya adalah pemanggilan sebuah fungsi program. Digambarkan dengan garis lurus berpanah dengan tulisan <<*include*>>.

b. *Extend*, digunakan ketika hendak menggambarkan variasi pada kondisi perilaku normal dan menggunakan lebih banyak kontrol *form* dan mendeklarasikan *ekstension* pada *use case* utama. Atau dengan kata lain adalah perluasan dari *use case* lain jika syarat atau kondisi terpenuhi. Digambarkan dengan garis berpanah dengan tulisan <<*extend*>>.

c. *Generalization/Inheritance*, dibuat ketika ada sebuah kejadian yang lain sendiri atau perlakuan khusus dan merupakan pola berhubungan *base-parent use case*. Digambarkan dengan garis berpanah tertutup dari *base use case* ke parent *use case*.

2.3.1.2. Activity Diagram

Diagram aktivitas menggambarkan proses bisnis dan urutan aktivitas-aktivitas yang mendukung penggambaran tindakan sistem baik yang bersifat kondisional maupun paralel. Tindakan kondisional dilukiskan dengan cabang *(branch)* dan penyatuan *(merge)*.

Sebuah *branch* memiliki sebuah *transition* masuk atau yang disebut dengan *incoming transition* dan beberapa *transition* keluar atau yang disebut dengan *outgoing transition* dari *branch* yang berupa keputusan-keputusan. Hanya satu dari *outgoing transition* yang dapat diambil, maka keputusan-keputusan

tersebut harus bersifat *mutually exclusive*. [*else*] digunakan sebagai keterangan singkat yang menunjukkan bahwa *transition* "*else*" tersebut harus digunakan jika semua keputusan yang ada pada *branch* salah.

Sebuah *merge* memiliki banyak *input transition* dan sebuah *output*. *Merge* menandakan akhir dari suatu kondisi yang diawali dengan sebuah *branch*. Selain *branch* dan *merge*, di dalam diagram aktivitas terdapat pula *fork* dan *join*. *Fork* memiliki *satu incoming transition* dan beberapa *outgoing transition*. Sedangkan pada join, *outgoing transition* diambil atau digunakan hanya ketika semua *state* pada *incoming transition* telah menyelesaikan aktivitasnya.

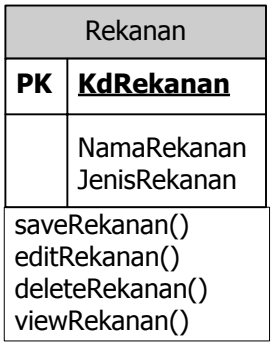
2.3.1.3. *Sequence Diagram*

Diagram yang menggambarkan bagaimana obyek berinteraksi dengan obyek lainnya melalui pesan *(message)* yang disampaikan, disusun dalam urutan kejadian atau waktu dan secara khusus berasosiasi dengan *use case*.

2.3.1.4. Class Diagram

Class diagram merupakan bagian yang paling penting dalam analisa dan perancangan berorientasi obyek. Dalam UML diagram kelas digunakan untuk memodelkan *static structure* dari sistem informasi.

Kelas merupakan himpunan dari obyek yang sejenis yang mempunyai atribut dan perilaku *(behaviors/method)* yang sama. Atribut adalah sebuah nilai data karakteristik yang dimiliki oleh obyek sebuah kelas sedangkan *method* adalah perilaku atau operasi yang dikenakan oleh suatu kelas. Pada gambar kelas terdapat tiga bagiannya.



Gambar 2.8 Bentuk Class dalam UML

Diagram kelas menggambarkan struktur obyek sistem, dimana diperlihatkan hubungan amtar mereka. Diagram kelas merupakan fondasi untuk *component* diagram dan *deployment* diagram.

Secara garis besar terdapat 3 jenis *class*. Ketiga jenis *class* tersebut dikelompokkan berdasarkan fungsi dan karakternya masing-masing, yaitu :

a. Entity Class Diagram

Merupakan paket utama dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk sistem dan menjadi landasan untuk menyusun basis data pada model data konseptual.



Gambar 2.9 Bentuk *Entity Class* dalam UML

b. Control Class Diagram

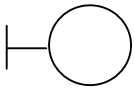
Berisi kumpulan kelas yang menjadi kontrol program termasuk koneksi dengan basis data dan merupakan kelas perantara atau penghubung antara *entity class* dengan kelas antar muka pemakai (*interface*).



Gambar 2.10 Bentuk *Control Clas*s dalam UML

c. Boundary Class Diagram

Berisi kumpulan kelas yang menjadi *interface* antara pemakai (*user*) dengan sistem, seperti tampilan *form* untuk pencetakan.



Gambar 2.11 Bentuk *Boundary Class* dalam UML