APLIKASI PENJUALAN

PULSA ELEKTRIK & VOUCHER

PROYEK AKHIR I

Oleh :

1. **NURDIANSYAH FAZRIKI 3311211012**
2. **ENI KURNIA 3311211024**
3. **YAYANK LESMANA 3311211028**
4. **WINARTI 3311211032**

PROGRAM STUDI TEKNIK INFORMATIKA

POLITEKNIK NEGERI BATAM

BATAM

2013

DAFTAR ISI

BAB I

PENDAHULUAN

1.1. Latar Belakang

Laporan penjualan berguna dalam proses pengambilan keputusan karenanya memerlukan pengendalian dalam proses pembuatan laporan penjualan tersebut. Program dibuat guna meningkatkan kelayakan penyajian laporan penjualan. Kurang optimalnya waktu pembuatan laporan dan masih banyaknya kesalahan yang terjadi dalam sebuah laporan penjualan membuat proses perekapan laporan yang telah berlangsung selama inipun kebanyakan masih menggunakan cara manual hanya dengan menggunakan MS. Office Ecxel. Dalam era perkembangan teknologi komputer saat ini masih banyak pengusaha yang belum memanfaatkan kemajuan teknologi komputer tersebut dengan baik. Kebanyakan dari mereka masih mencatat transaksi secara sistem manual. Dalam sistem manual pengusaha mencatat setiap transaksi yang terjadi sehingga hal ini bisa menimbulkan permasalahan yang terjadi pada saat proses pencatatan tersebut seperti kekeliruan pencatatan pemasukan dan pengeluaran barang, penghitungan laba rugi, dan pembuatan laporan. Semakin menjamurnya penyedia layanan pengisian pulsa baik pulsa elektrik maupun pulsa fisik, makin banyak persaingan yang terjadi. Oleh karena itu, untuk mempermudah dalam melakukan transaksi, penulis ingin membuat suatu aplikasi yang dapat mempermudah baik dalam melakukan transaksi penjualan maupun transaksi pembelian. Untuk itu, penulis membuat suatu aplikasi yang benar-benar dibutuhkan atau sesuai dengan keinginan pemilik toko. Pemilik toko akan sukar melakukan penghitungan laba dan pengecekan stok barang pun harus dilakukan dengan mengecek atau menghitung jumlah barang yang tersedia. Untuk dapat menutupi semua kelemahan ini maka penulis membuat suatu aplikasi penjualan ini.

1.2. Rumusan Masalah

Berdasarkan latar belakang di atas, maka penulis menyampaikan rumusan permasalahan, sebagai berikut:

1. Pemilik kesulitan memperoleh laporan hasil penjualan kartu telpon & pulsa terhadap costumer.
2. Bagaimana mengurangi kesalahan dalam transaksi penjualan.
3. Bagaimana mengelola stok barang.

1.3. Batasan Masalah

Adapun batasan masalah yang ada dalam pembuatan sistem ini adalah sebagai berikut :

1. Sistem hanya digunakan oleh karyawan / petugas.
2. Pembuatan sistem menggunakan aplikasi Java, Netbean & MYSQL.

1.4. Tujuan

Berdasarkan rumusan masalah di atas, maka tujuan dari pembuatan aplikasi ini, adalah:

1. Mempermudah pembuatan laporan penjualan kepada pemilik.
2. Mempermudah dalam mengurangi kesalahan transaksi penjualan.
3. Mempermudah pengelolaan stok barang.

1.5. Sistematika Penulisan

Sistematika penulisan penelitian ini sebagai berikut:

Bab I Pendahuluan

Bab ini berisi tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

Bab II Tinjauan Pustaka

Bab ini menjelaskan tentang teori-teori dari berbagai referensi yang berkaitan dengan judul yang penulis ambil untuk menunjang penelitian.

Bab III Analisis & Perancangan Bahasa Pemrograman JAVA, NETBEAN & MYSQL Penulis menguraikan gambaran lebih jelas tentang analisis & perancangan bahasa pemrograman JAVA, NETBEAN & MYSQL yg digunakan.

Bab IV Pembahasan

Bab ini menguraikan jawaban atas pertanaan yang sebelumnya telah dikemukakan pada rumusan masalah.

Bab V Penutup

Pada bab ini berisi kesimpulan jawaban dari pertanyaan pada rumusan masalah yang diperoleh dari Bab IV (Pembahasan) secara ringkas dan saran penulis terkait kesimpulan jawaban yang dikemukakan sebelumnya.

BAB II

TINJAUAN PUSTAKA

2.1 Pengenalan Sistem Secara Umum Pada Laporan Penjualan

Secara sederhana sistem dapat diartikan sebagai suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu.

Laporan penjualan adalah sekumpulan informasi penjualan dalan suatu periode tertentu yang disajikan dalam bentuk laporan sistematis yang mudah dibaca dan dipahami oleh semua pihak yang membutuhkan. Laporan penjualan dibuat oleh bagian manajemen dengan tujuan untuk mempertanggung jawabkan tugas-tugas yang dibebankan kepadanya oleh para pemilik perusahaan selama satu periode. Laporan penjualan harus menyajikan secara wajar posisi keuangan, kinerja keuangan. Disamping itu laporan keuangan dapat juga digunakan untuk memenuhi tujuan-tujuan lain yaitu sebagai laporan kepada pihak-pihak diluar perusahaan yang meliputi para kreditur, para investor dan pemerintah dimana perusahaan tersebut berdomisili, serta masyarakat sekitarnya.

2.2 Konsep Dasar Pemograman Aplikasi Java

2.2.1. Pengertian Java

Java adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer standalone ataupun pada lingkungan jaringan. Kita lebih menyukai menyebut Java sebagai sebuah teknologi dibanding hanya sebuah bahasa pemrograman, karena Java lebih lengkap dibanding sebuah bahasa pemrograman konvensional. Teknologi Java memiliki tiga komponen penting, yaitu:

1. Programming-language specification
2. Application-programming interface
3. Virtual-machine specification

Java dibuat dan diperkenalkan pertama sekali oleh sebuah tim *Sun Microsystem* yang dipimpin oleh Patrick Naughton dan James Gosling pada tahun 1991 dengan *code name* **Oak**. Tahun 1995 *Sun* mengubah nama *Oak* tersebut menjadi *Java*.

2.3 OOP Pada Java

OOP *(Object Oriented Programming)* yaitu proses pembuatan program / aplikasi dengan cara menyusunnya dari beberapa object (komponen yang lebih kecil). Object atau komponen itu sendiri juga bisa tersusun dari object yang lain. Pemrograman OOP memiliki berbagai kelebihan dibandingkan dengan konsep pemrograman lain. Kelebihan OOP antara lain kemampuan untuk mewariskan elemen program ke program lain (inheritance), dapat menyembunyikan kerumitan program dengan encapsulation, pembuatan beberapa fungsi dengan nama sama tapi memiliki parameter dan atribut yang berbeda dengan overloading, pembutan fungsi baru yang menimpa fungsi lain dengan overiding.

1. Atribut dan Method

Atribut merupakan aspek individual yang membedakan sebuah object dengan object lainnya. Misalkan sebuah object Motor, maka atribut yang dimiliki motor adalah warna, jumlah, roda, merek dan lain sebagainya. Dalam pembuatan object di java, atribut didefinisikan menggunakan *variable instance*. Tipe-tipe atribut didefinisikan dalam class, dan setiap instance yang dibuat dari class tersebut menyimpan atributnya masing-masing di dalam variable instance. Method merupakan kemampuan sebuah object untuk melaksanakan sesuatu. Dengan kata lain, method merupakan fungsi (subroutine) yang didefinisikan di dalam class dan beroperasi pada instance yang dibuat dari class tersebut.

2. Constructor

Constuctor merupakan method yang bernama sama dengan class-nya yang digunakan untuk inisialisasi object. Sebuah class dapat memiliki banyak constructor dengan overloading, yaitu membuat method yang bernama sama tetapi memiliki parameter dan implementasi yang berbeda. Pada saat instance dari class dibuat, maka object yang terjadi adalah hasil inisialisasi dari constructor. Apabila class memiliki lebih dari satu constructor maka constructor

yang bekerja adalah yang memiliki parameter yang sesuai dengan pembuatan instance.

3. Abstract dan Interface

Abstract Suatu method dari suatu class dapat dibuat tanpa implementasi kode apapun. Method seperti itu disebut method abstract dan untuk menggunakannya harus didefinisikan dahulu pada turunan class tersebut. Suatu class yang memiliki method abstract maka menjadi class abstract. Suatu class abstract tidak dapat digunakan untuk membuat objek. Namun demikian suatu class abstract tetap dapat diturunkan. *Interface* Adalah suatu abstraksi dari class. Interface hanya berisi daftar field dan method tanpa detail kode. Suatu class yang mengimplementasikan suatu interface maka class tersebut harus mendefinisikan method yang terdapat pada interface tersebut.

4. Penanganan Kesalahan Pada Java

Java menggunakan model *exception* dalam menangani kesalahan pada program. Oleh karena pemeriksaan pada tahap kompilasi program Java sangat ketat, suatu method yang berpotensi untuk membangkitkan kesalahan harus ditangani dengan struktur try…catch…finally atau dengan menyatakan bahwa pada method tersebut berpotensi menbangkitkan kesalahan. Struktur dari blok try….catch….finally adalah sebagai berikut:

```
try
{
….statement yang berpeluang untuk terjadinya kesalahan….
}
catch(classexception pengenal)
{
….statement yang akan dikerjakan apabila kesalahan terjadi….
}
finally
{
….statement yang akan dikerjakan baik apabila kesalahan terjadi maupun
tidak….
}
```

Jika dikehendaki untuk melakukan aksi yang berbeda untuk setiap tipe eksepsi yang muncul, maka blok try…catch tersebut dapat ditulis sebagai berikut:

```
try
{
….statement yang berpeluang untuk terjadinya kesalahan….
}
catch(classexception1 pengenal)
{
….statement yang akan dikerjakan apabila kesalahan 1 terjadi….
12
}
catch(classexception2 pengenal)
{
….statement yang akan dikerjakan baik apabila kesalahan 2 terjadi….
}
```

2.3.3. Keunggulan Bahasa Pemograman Java

Jika kita membahas keunggulan Java, yang terbayang pasti tentang bagaimana cara memprogramnya. Di sini terdapat beberapa keunggulan JAVA, dan bahasa java sangat lebih mudah di mengerti tidak seperti bahasa mesin lainnya lebih mendekati kepada bahasa manusia, contohnya: *if, else*, dan lebih banyak lagi. Maka dari itu baca dengan seksama beberapa keunggulan Java di bawah ini. Keunggulan bahasa pemrograman Java antara lain:

1. **Berorientasi Objek**

   Java adalah bahasa pemrograman yang berorientasi pada objek. Java membagi program menjadi objek-objek serta memodelkan sifat dan tingkah laku masing-masing dalam menyelesaikan suatu masalah.

2. **Java bersifat multiplatform**

   Java dirancang untuk mendukung aplikasi yang dapat beroperasi di lingkungan jaringan berbeda. Untuk mengakomodasi hal tersebut, Java compiler membangkitkan bytecodes (sebuah format yang tidak tergantung pada arsitektur tertentu yang didesain untuk mengirimkan kode ke banyak platform perangkat keras dan perangkat lunak secara efisien). Java dapat dijalankan oleh banyak *platform* seperti Linux, *Unix, Windows, Solari*, maupun *Mac.*

3. J**ava multithread**

   Multithreading adalah kemampuan suatu program komputer untuk mengerjakan beberapa proses dalam suatu waktu. *Thread* dalam Java memiliki kemampuan untuk memanfaatkan kelebihan multi *prosessor* apabila sistem operasi yang digunakan mendukung multi *prosessor*.

4. **Dapat didistribusi dengan mudah**

   Java memiliki *library* rutin yang lengkap untuk dirangkai pada *protocol* TCP/IP (seperti HTTP dan FTP) dengan mudah. Kemampuan networking Java lebih kuat dan lebih mudah digunakan. Java memudahkan tugas pemrograman jaringan yang sulit seperti membuka dan mengakses sebuah soket koneksi. Java juga mamudahkan pembuatan CGI (Common Gateway Interface).

5. **Bersifat dinamis**

   Java dirancang untuk beradaptasi dengan lingkungan yang sedang berkembang. Java bersifat dinamis dalam tahap linking. *Class* yang ada dapat di link sebatas yang diperlukan, apabila diperlukan modul kode yang baru dapat di link dari beberapa sumber, bahkan dari sumber dalam jaringan Internet.

2.4. Konsep Dasar MYSQL

2.4.1. Pengertian MYSQL

MySQL merupakan perangkat lunak siste manajemen basis data SQL atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional yang didistribusikan secara gratis dibawah lisensi GPL (*General Public Lecense*). MySQL adalah Relational Database Management System (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (General Public License). Dimana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu SQL (Structured Query Language). SQL adalah sebuah konsep pengoperasian database, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Keandalan suatu sistem database (DBMS) dapat diketahui dari cara kerja optimizer-nya dalam melakukan proses perintah-perintah SQL, yang dibuat oleh user maupun program-program aplikasinya. Sebagai database server, MySQL dapat dikatakan lebih unggul dibandingkan database server lainnya dalam query data. Hal ini terbukti untuk query yang dilakukan oleh single user, kecepatan query MySQL bisa sepuluh kali lebih cepat dari PostgreSQL dan lima kali lebih cepat dibandingkan Interbase.

2.4.2. Keistimewaan MYSQL

MySQL memiliki beberapa keistimewaan, antara lain :

- **Portabilitas**

  MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.

- **Open Source**

  MySQL didistribusikan secara *open source*, dibawah lisensi GPL sehingga dapat digunakan secara cuma-cuma.

- **Multiuser**

  MySQL dapat digunakan oleh beberapa user dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

- **Performance tuning**

  MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.

- **Jenis Kolom**

  MySQL memiliki tipe kolom yang sangat kompleks, seperti signed / unsigned integer, float, double, char, text, date, timestamp, dan lain-lain.

- **Perintah dan Fungsi**

  MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (*query*).

- **Keamanan**

  MySQL memiliki beberapa lapisan sekuritas seperti level subnetmask, nama host, dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenkripsi.

- **Skalabilitas dan Pembatasan**

  MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (records) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar

baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.

- **Konektivitas**

  MySQL dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).

- **Lokalisasi**

  MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.

- **Antar Muka**

  MySQL memiliki interface (antar muka) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (Application Programming Interface).

- **Klien dan Peralatan**

  MySQL dilengkapi dengan berbagai peralatan (tool)yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.

- **Struktur tabel**

  MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupun Oracle.

2.5. NetBean

NetBeans mengacu pada kedua platform kerangka untuk aplikasi desktop Java, dan sebuah lingkungan pengembangan terpadu (IDE) untuk pengembangan dengan Java , JavaScript , PHP , Python , Ruby , Groovy , C , C + + , Scala , Clojure , dan lain-lain. NetBeans IDE ditulis dalam Java dan berjalan di mana-mana mana JVM diinstal, termasuk Windows, Mac OS, Linux, dan Solaris.Sebuah JDK diperlukan untuk pengembangan fungsionalitas Jawa, tetapi tidak diperlukan untuk pembangunan di bahasa pemrograman lain. Platform NetBeans memungkinkan aplikasi untuk dikembangkan dari satu set modular komponen software yang disebut modul. Aplikasi berbasis platform NetBeans (termasuk IDE NetBeans) dapat diperpanjang oleh pengembang pihak ketiga .

2.5.1. Awal sejarah

NetBeans dimulai pada tahun 1996 sebagai Xelfi (kata bermain pada Delphi), Java IDE proyek mahasiswa di bawah bimbingan Fakultas Matematika dan Fisika di Charles University di Praha . Pada tahun 1997 Staněk Romawi membentuk perusahaan sekitar proyek tersebut dan menghasilkan versi komersial NetBeans IDE hingga kemudian dibeli oleh Sun Microsystems pada tahun 1999. Komunitas NetBeans sejak terus tumbuh, berkat individu dan perusahaan yang menggunakan dan berkontribusi dalam proyek ini. versi Lancar NetBeans IDE 6.0 memperkenalkan dukungan untuk mengembangkan modul IDE dan aplikasi klien kaya berdasarkan platform NetBeans, Java Swing GUI builder (sebelumnya dikenal sebagai "Proyek Matisse"), meningkatkan CVS dukungan, WebLogic 9 dan JBoss 4 dukungan, dan perangkat tambahan banyak editor. NetBeans 6 is available in official repositories of major Linux distributions. NetBeans 6 tersedia dalam repositori resmi dari distribusi Linux utama. Selain itu, NetBeans Enterprise Pack mendukung pengembangan aplikasi Java EE 5 perusahaan, termasuk SOA alat desain visual, skema XML tools, web orkestrasi layanan (untuk BPEL), dan UML modeling. The NetBeans IDE Bundle for C/C++ supports C/C++ development. The NetBeans IDE Bundle

untuk C / C + + mendukung C / C + + pembangunan. Hosting pengembang sumber terbuka proyek di kenai.com tambahan manfaat dari instant messaging dan pelacakan masalah integrasi dan navigasi kanan dalam IDE, dukungan untuk pengembangan aplikasi web dengan PHP 5.3 dan kerangka Symfony, dan kode selesai diperbaiki, layout, petunjuk dan navigasi dalam proyek JavaFX. dirilis pada bulan Juni 2010, menambahkan dukungan untuk OSGi , Spring Framework 3.0, Java EE injeksi ketergantungan (JSR-299), Zend Framework untuk PHP , dan navigasi kode lebih mudah (seperti "Apakah / ditimpa Penerapan" penjelasan), format , petunjuk, dan refactoring di beberapa bahasa.

2.6. Unified Modeling Language (UML)

Saat ini piranti lunak semakin luas dan besar lingkupnya, sehingga tidak bisa lagi dibuat asal-asalan.Piranti lunak saat ini seharusnya dirancang dengan memperhatikan hal-hal seperti *scalability*, *security*, dan eksekusi yang robust walaupun dalam kondisi yang sulit. Selain itu arsitekturnya harus didefinisikan dengan jelas, agar *bug* mudah ditemukan dan diperbaiki, bahkan oleh orang lain selain *programmer* aslinya. Keuntungan lain dari perencanaan arsitektur yang matang adalah dimungkinkannya penggunaan kembali modul atau komponen untuk aplikasi piranti lunak lain yang membutuhkan fungsionalitas yang sama. Pemodelan (*modeling*) adalah proses merancang piranti lunak sebelum melakukan pengkodean (*coding*). Model piranti lunak dapat dianalogikan seperti pembuatan blueprint pada pembangunan gedung. Membuat model dari sebuah sistem yang kompleks sangatlah penting karena kita tidak dapat memahami sistem semacam itu secara menyeluruh. Semakin komplek sebuah sistem, semakin penting pula penggunaan teknik pemodelan yang baik. Dengan menggunakan model, diharapkan pengembangan piranti lunak dapat memenuhi semua kebutuhan pengguna dengan lengkap dan tepat, termasuk faktor-faktor seperti *scalability, robustness, security*, dan sebagainya. Kesuksesan suatu pemodelan piranti lunak ditentukan oleh tiga unsur, yang kemudian terkenal dengan sebuan segitiga sukses (*the triangle for success*). Ketiga unsur tersebut adalah metode pemodelan (*notation*), proses (*process*) dan *tool* yang digunakan. Memahami notasi pemodelan tanpa mengetahui cara pemakaian yang

sebenarnya (proses) akan membuat proyek gagal. Dan pemahaman terhadap metode pemodelan dan proses disempurnakan dengan penggunaan tool yang tepat.

2.7. Apa itu UML

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa – bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C. Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. 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).

2.7.1. Sejarah UML

Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi booch [1], metodologi coad [2], metodologi OOSE [3], metodologi OMT [4], metodologi shlaer-mellor [5], metodologi wirfs-brock [6], dsb. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul

masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan. 18 Dimulai pada bulan Oktober 1994 Booch, Rumbaugh dan Jacobson, yang merupakan tiga tokoh yang boleh dikata metodologinya banyak digunakan mempelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 direlease draft pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh **Object Management Group** (OMG – http://www.omg.org). Tahun 1997 UML versi 1.1 muncul, dan saat ini versi terbaru adalah versi 1.5 yang dirilis bulan Maret 2003. Booch, Rumbaugh dan Jacobson menyusun tiga buku serial tentang UML pada tahun 1999 [7] [8] [9]. Sejak saat itulah UML telah menjelma menjadi standar bahasa pemodelan untuk aplikasi berorientasi objek.

2.8. Konsepsi Dasar UML

Dari berbagai penjelasan rumit yang terdapat di dokumen dan buku-buku UML. Abstraksi konsep dasar UML yang terdiri dari *structural classification*, *dynamic behavior*, dan *model management*, bisa kita pahami dengan mudah apabila kita melihat gambar diatas dari *Diagrams*. *Main concepts* bisa kita pandang sebagai term yang akan muncul pada saat kita membuat diagram. Dan view adalah kategori dari diagram tersebut. Untuk menguasai UML, sebenarnya cukup dua hal yang harus kita perhatikan:

1. Menguasai pembuatan diagram UML

2. Menguasai langkah-langkah dalam analisa dan pengembangan dengan UML

Adapun proses UML yang digunakan penulis dalam merancang proyek kali ini adalah sebagai berikut :

1. *Use Case Diagram*,
2. *Class diagram*, dan
3. *Sequnce Diagram.*

2.8.1. Pengertian *Use Case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain. Berikut adalah gambaran *use case* untuk mempresentasikan sistem yang penulis buat.

2.8.2. Pengertian Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok :

1. Nama (dan stereotype)
2. Atribut

3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
2. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.
3. *Public*, dapat dipanggil oleh siapa saja.

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*. Sesuai dengan perkembangan *class* model, *class* dapat dikelompokkan menjadi *package*. Kita juga dapat membuat diagram yang terdiri atas *package*.

Hubungan Antar *Class:*

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar *class*.
2. Agregasi, yaitu hubungan yang menyatakan bagian ("terdiri atas..").
3. Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
4. Hubungan dinamis, yaitu rangkaian pesan (*message*) yang di-*passing* dari satu*class* kepada *class* lain.

2.8.3. Pengertian Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Masing-masing objek, termasuk aktor, memiliki *lifeline*

vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah message. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan *icon* khusus untuk objek *boundary, controller* dan *persistent entity*.

2.9. Pengertian Provider

Provider ialah Badan Usaha yang menyediakan layanan koneksi. Provider mempunyai banyak jenis. Salah satunya adalah provider kartu untuk handphone.

2.10. Pengertian Pulsa

pulsa adalah satuan perhitungan biaya telepon yang kita bayar di depan (prabayar) untuk dapat menggunakan layanan dari suatu provider.

2.10.1. Jenis-Jenis Pulsa

1. **Pulsa Fisik / Voucher Fisik**

   Voucher fisik adalah voucher yang pertama kali diperkenalkan oleh para provider di Indonesia untuk para pelanggan prabayarnya agar dapat terus menggunakan layanan yang diberikan. Cara penggunaaan voucher fisik ialah dengan menggosok bagian timah pelindung nomor voucher, lalu memasukkan angka-angka yang tersembunyi di dalamnya sesuai dengan prosedur dan kebijakan masing-masing penyedia layanan

2. **Pulsa Elektrik / Voucher Elektrik**

   Disamping voucher fisik ada juga voucher elektrik, apa yang dimaksud dengan voucher elektrik? voucher elektrik adalah salah satu jenis voucher isi ulang yang dikeluarkan oleh provider yang hanya dapat di-top up oleh chip khusus keluaran provider yang telah diotorisasi penggunaannya untuk melakukan top up voucher isi ulang. jenis voucher ini tidak berbentuk karena saat digunakan / di top up maka akan langsung otomatis terisi sesuai dengan nominal yang diinginkan ke nomor handphone yang dituju.

BAB III

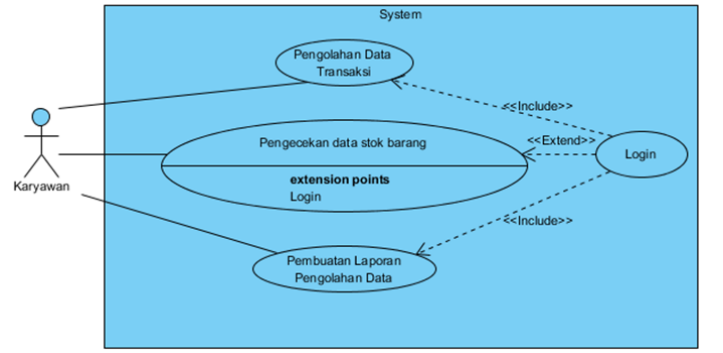
ANALISIS & PERANCANGAN

3.1. Deskripsi Umum Sistem

Dalam sistem ini dijelaskan bahwa sistem ini dirancang untuk memudahkan karyawan dalam melakukan transaksi penjualan. Aplikasi yang digunakanpun dapat dengan mudah dipakai. Sistem menerima masukan data dengan memasukkan terlebih dahulu *user name* & *password*, selanjutnya sistem akan mengecek *user name* & *password*, setelah itu sistem akan menampilkan menu utama, pada pilihan menu utama pengguna dapat memilih menu yang akan dipakai, contohnya menu transaksi dimana dalam menu tersebut pengguna bisa memasukkan data transaksi yang dilakukan. Dalam sistem ini juga dapat memberikan informasi tentang stok saldo pulsa untuk pulsa elektrik & stok untuk voucher yang tersedia pada sistem. Dalam hal pentotalan harga sistem pun akan langsung secara otomatis menjumlahkannya dan langsung mengurangkan stok saldo pulsa yang ada pada sistem.

3.2. Use Case Diagram

Pada diagram ini terdapat tiga aktor yang merupakan pengguna sistem, yaitu pelanggan, karyawan, dan aplikasi. pelanggan sebagai aktor yang melakukan transaksi pembelian, memiliki *use case* melakukan transaksi pembelian lalu memberikan nomor telepon selular dan nominal pulsa yang akan dibeli. Aktor karyawan memiliki *use case* memasukkan nomor telepon dan nominal pulsa dari customer ke aplikasi. Selanjutnya karyawan mengirimkan SMS ke provider penyedia pulsa elektrik dan memberikan status pengiriman pulsa elektrik atau pemberitahuan stok voucher jika yang dibeli adalah voucher kepada aktor karyawan. Setelah itu aktor karyawan memberikan status pengiriman pulsa elektrik kepada pelanggan atau memberikan voucher kepada pelanggan.



2.11. Gambar Use Case Diagram

2.12. Skenario Use Case Diagram

Adapun tahapan-tahapan sekenario use case Penjualan pulsa elektrik & voucher yang sedang berjalan adalah sebagai berikut :

3.3.1 Skenario Use case Login

Nama *Use Case* : Login

Aktor : Karyawan

Tujuan : Karyawan melakukan proses Login ke sistem untuk memasukan data transaksi.

3.3.1. Tabel skenario Login

Karyawan	Sistem
3. Memasukkan user name & pasword.	
	4. Memeriksa user name & pasword.

3.3.2 Skenario Use Case Pengecekan Data Stok Barang

Nama *Use Case* : Melakukan Pengecekan Stok Barang

Aktor : Karyawan

Tujuan : Karyawan mengecek stok barang pada data barang

3.3.2. Tabel Skenario Pengecekan Data Stok Barang

Karyawan	Sistem
1. Melakukan pengecekan.	
	2. Sistem akan menampilkan data stok barang.
3. Mengecek stok yang masih tersedia, & melakukan perubahan data stok barang.	
	4. Menyimpan data stok barang jika ada perubahan stok barang setelah pengecekan.

3.3.3 Skenario Melakukan Transaksi

Nama Use case : Melakukan Transaksi Penjualan

Aktor : Karyawan

Tujuan : Karyawan melakukan pengisian transaksi penjualan

3.3.3. Table skenario Melakukan Transaksi

Karyawan	Sistem
1. Melakukan transaksi, membuka pilihan menu transaksi (Transaksi penjualan pulsa elekrik atau transaksi penjualan voucher.	
	2. Sistem akan menampilkan form transaksi penjualan yang yang diinginkan.
5. Melakukan pengisian form transaksi.	
	6. Menyimpan data transaksi.

3.3.4 Skenario Laporan Pengolahan Data

Nama Use case : Pembuatan laporan Pengolahan Data

Aktor : Karyawan

Tujuan : Karyawan membuat Laporan Penjualan pulsa elektrik & Voucher.

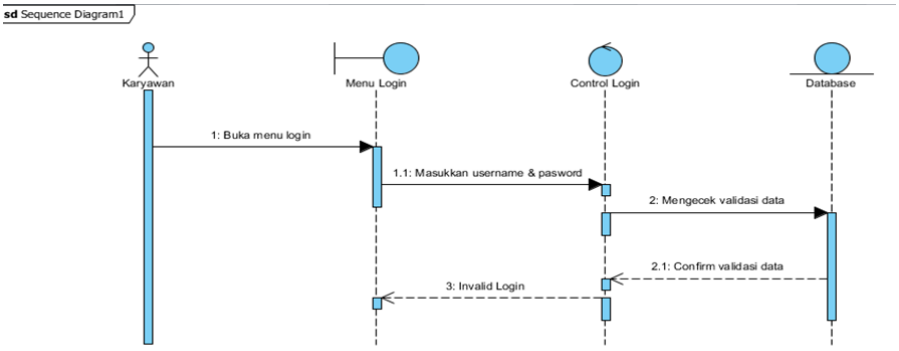
3.3.4. Tabel Skenario Laporan pengolahan data

Karyawan	Sistem
1. Melakukan pembuatan laporan pengolahan data.	
	2. Sistem akan menampilkan form semua hasil transaksi penjualan yang telah dimasukkan oleh karyawan.
7. Mencetak laporan.	

3.4. Sequence Diagram

Berikut adalah gambaran *sequence diagram* dalam aplikasi ini.

1. Sequence Diagram Login

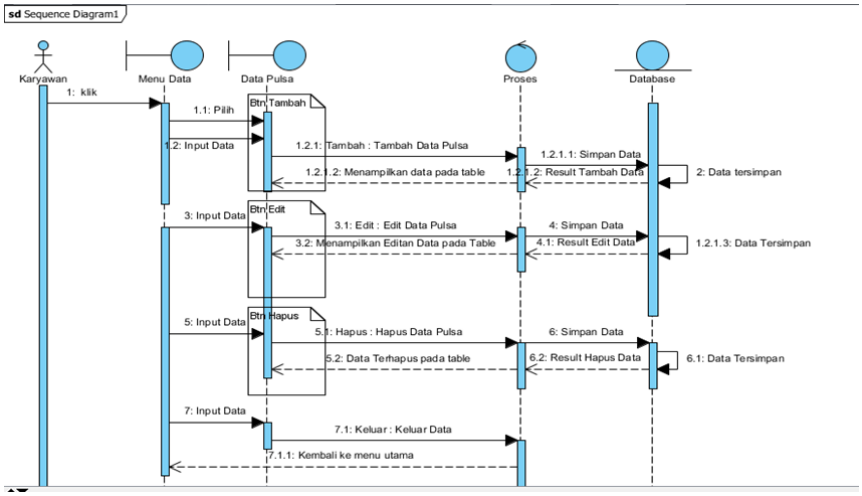


3.4.1. Gambar Sequence diagram Login

Pada gambar diatas adalah proses Login. Dimana pada proses ini pengguna aplikasi yaitu Karyawan memasukkan *user_name & password* ke

sistem, lalu sistem akan mengecek / mencocokkan data *user_name & password* tersebut. Setelah proses selesai lalu sistem akan kembali ke menu utama.

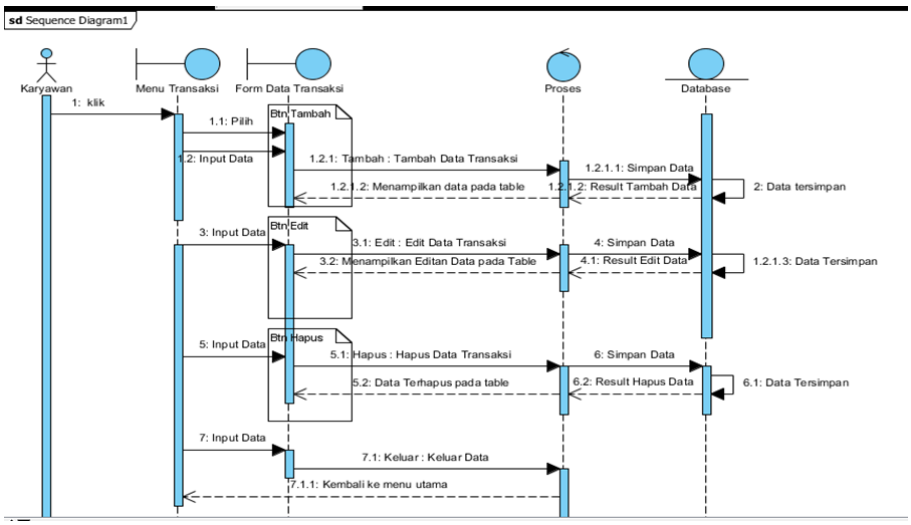
2. Sequence Diagram Pengecekan Data Stok Barang



3.4.2. Gambar Sequence Diagram Pengecekan Data Stok Barang

Pada gambar diatas menjelaskan bahwa untuk data pengecekan stok barang dilakukan di menu data lalu pengguna pilih data pulsa, pada data pulsa pengguna memasukkan data pulsa berupa Id pulsa, jenis pulsa, nama provider harga beli, harga jual dan stok saldo, untuk menambahkan data cukup dengan klik tombol tambah, untuk merubah data pulsa klik tombol edit, dan pengguna siap untuk mengedit data yang diinginkan, klik tombol hapus untuk menghapus data yang diinginkan.

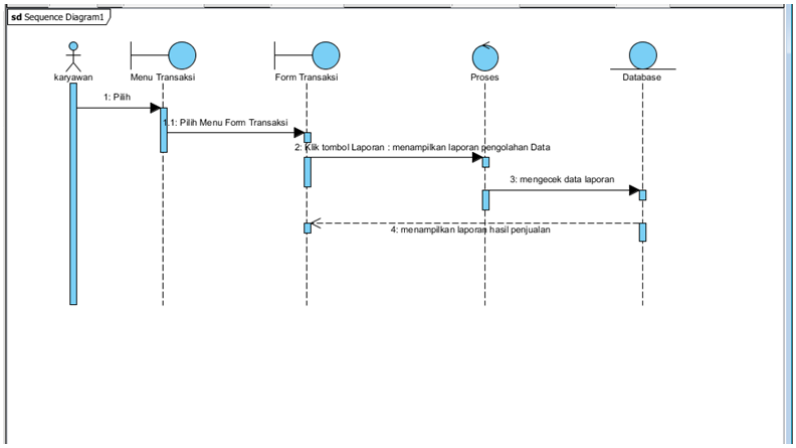
3. Siequence proses pengelolaan data transaksi



Gambar 3.4.2. Sequence Diagram Pengelolaan Data Transaksi

Untuk melakukan transaksi penjualan pulsa terdapat menu pada transaksi lalu pilih menu pulsa. Setelah itu akan muncul form transaksi pulsa, untuk transaksi penjualan, untuk melakukan pengisian transaksi penjualan pulsa elektrik pengguna memasukkan jenis pulsa yang akan ditransaksi dalam hal ini jenis pulsa elektrik & jenis pulsa voucher yang digunakan oleh pembeli, pulsa yang ingin dibeli dan nomor tujuan pengiriman pulsa elektrik setelah itu klik tombol Simpan. Sedangkan untuk melakukan transaksi pembelian voucher, pengguna hanya mengganti jenis pulsa saja untuk transaksi penjualan voucher

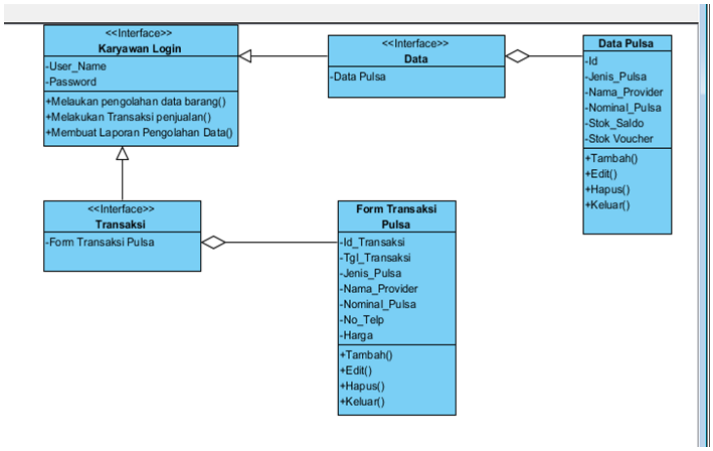
4. Proses pembuatan laporan pengelolaan data



Gambar 3.4.3. Sequence Diagram Laporan Pengolahan Data

3.5. Class Diagram

Class Diagram digunakan untuk menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class* adalah sebuah spesifikasi yang jika digambarkan akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek.



3.5.1. Gambar Class Diagram

BAB IV

PEMBAHASAN

4.1. Pengujian

Pengujian adalah proses menjalankan dan mengevaluasi sebuah perangkat lunak secara manual maupun otomatis untuk menguji apakah perangkat lunak sudah memenuhi persyaratan atau belum. Dan untuk menentukan perbedaan antara hasil yang diharapkan dengan hasil sebenarnya.

4.2. Rencana Pengujian

Pengujian yang telah dilakukan selama membuat aplikasi ini, antara lain :

1. Pengujian frame, yaitu pengujian ini difokuskan pada suatu frame dari program secara sendiri.
2. Pengujian Aplikasi Penjualan Pulsa menggunakan data uji berupa sebuah data masukan.

4.3. Implementasi

Tahapan implementasi adalah tahapan penerapan sistem untuk dapat dioperasikan. Pada tahapan ini dijelaskan mengenai sistem yang dirancang dan bagaimana cara penggunaannya.

4.3.1. Batasan Implementasi

Mengimplementasikan sistem aplikasi ini ada beberapa hal yangdijadikan batasan- batasan implementasi yaitu sebagai berikut :

1. Implementasi perangkat lunak dilakukan pada PC (personal computer)dengan sistem operasi microsoft windows 7.
2. Basis data yang digunakan dalam pengimplementasian ini adalah MySQL.
3. Pembuatan Aplikasi menggunakan bahasa pemrograman Java.

4.3.2. Implementasi Perangkat Lunak

Adapun perangkat lunak yang digunakan dalam pengembangan aplikasi ini adalah adalah :

1. NetBeans

   Penyusun menggunakan Netbeans untuk membuat Aplikasi Penjualan Pulsa ini.

2. Heidi SQL

   Penyusun menggunakan Heidi SQL untuk membuat basis data.

4.4. Implementasi Perangkat Keras

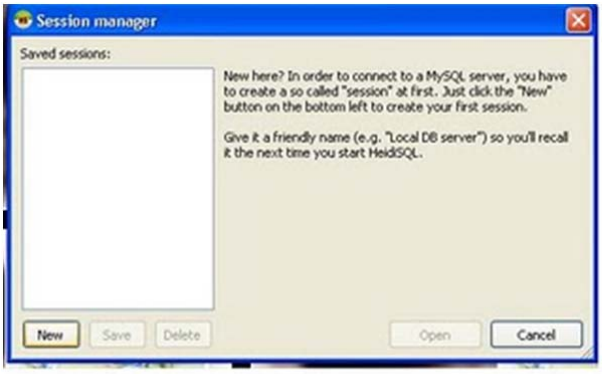
Perangkat keras yang digunakan berdasarkan kebutuhan minimalyang harus terpenuhi untuk menjalankan program tersebut antara lain adalah :

a. Menggunakan minimal prosessor intel duo core
b. RAM 1 G
c. Harddisk untuk menyimpan data minimal 160Ghz.
d. DVD RW
e. Monitor, mouse, keyboard, dan printer

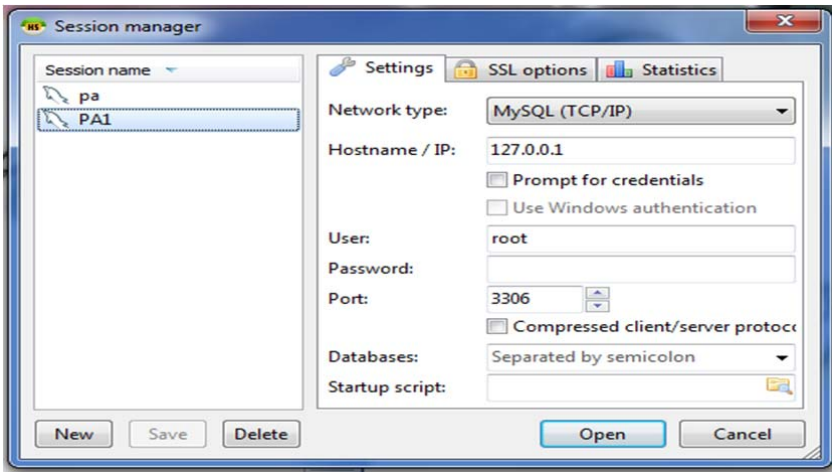
4.5. Implementasi Basis Data

Pembuatan basis data dilakukan dengan menggunakan Heidi SQL Berikut akan dijelaskan langkah-langkah dalam pembuatan basis data sistem informasi yang dibangun:

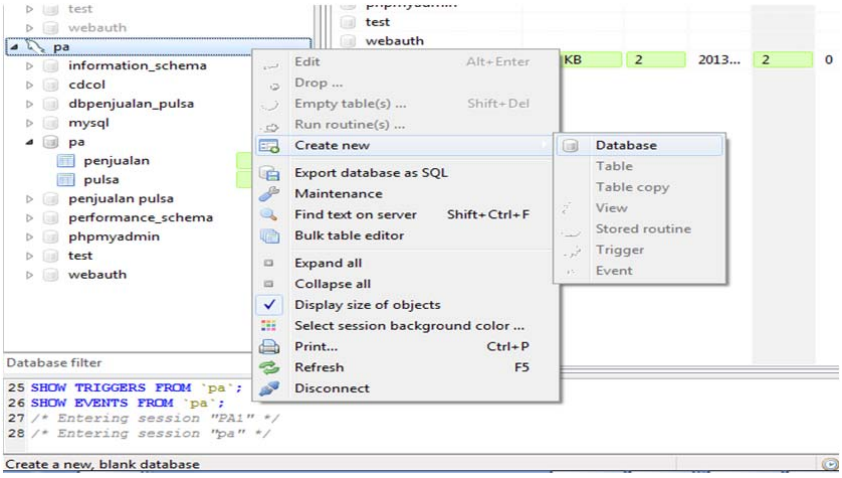
1. Buka HeidiSql, Klik New untuk New session



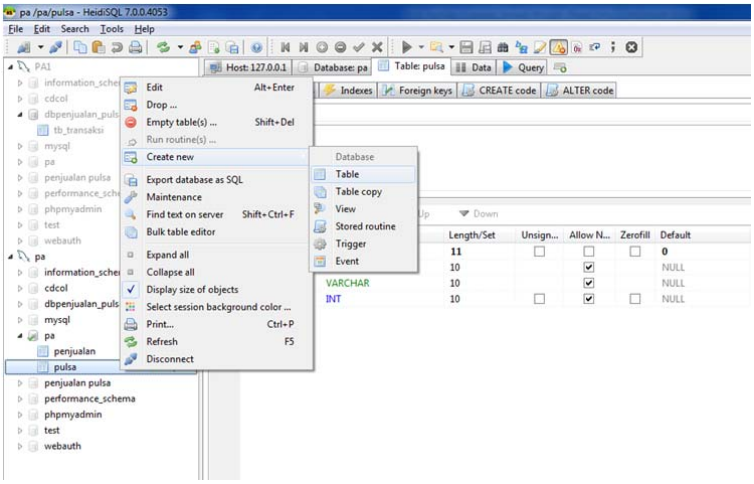
2. Contoh Beri Nama PA dan kosongkan password lalu klik open, yes



3. Buatlah database baru dengan klik kanan di PA Lalu klik Create New, Database.



4. Buatlah database bernama dbpenjualan
5. Di dbpenjualan, klik kanan lalu create new, table



6. Kemudian buat table baru beri nama: tb_pulsa, Klik Add di Columns lalu isi

   `Id_Pulsa` int(11) NOT NULL,

   ` Jenis_Pulsa'varchar (10)NOT NULL,

   ` Nama_Provider ` varchar(10) NOT NULL,

   `Stok` int(10)NOT NULL,

   Kemudian set Id_Pulsa sebagai primary key

7. Kemudian buat table baru beri nama: tb_penjualan, Klik Add di Columns lalu isi

   `Id_Transaksi` int(10) NOT NULL,

   `Tgl_Transaksi` date NOT NULL,

   `Jenis_Pulsa` varchar(10) NOT NULL,

   `Nama_Provider` varchar(10) NOT NULL,

   `Nominal_Pulsa` int(50) NOT NULL,

   `No_Telp` int(12) NOT NULL,

   `Harga` int(50) NOT NULL,

   Kemudian set NoPinjam sebagai primary key

4.6. Implementasi Antar Muka

Kegiatan ini bertujuan untuk menerangkan secara singkat penggunaanAplikasi Koperasi Simpan Pinjam. Cara penggunaanya adalah sebagai berikut :

4.6.1. Implementasi Frame Login

Perancangan tampilan login merupakan syarat untuk dapat memanfaatkan secara keseluruhan, di mana yang menggunakan aplikasi ini adalah petugas yang di beri wewenang untuk memasukan data ke dalam basisdata ataupun karyawan yang bertugas di masing-masing bagian. Pada menu login ini ada dua data yang harus di masukan , yaitu:

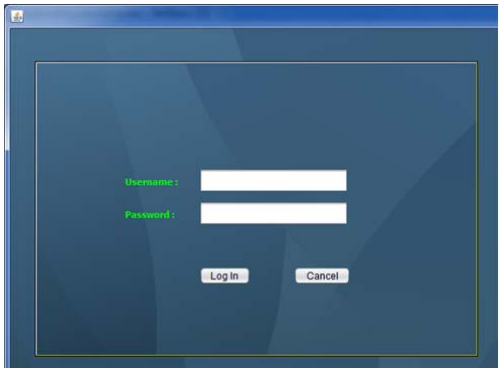
a. User name

User name merupakan pengguna yang sudah terdaftar dan diberikan wewenang untuk menggunakan aplikasi koperasi simpan pinjam.Pada kotak teks, yang dimasukan adalah karakter.

b. Password

Password merupakan syarat mutlak untuk bisa masuk ke dalam aplikasi koperasi simpan pinjam, yang harus di masukan dengan tepat, dimaksudkan agar orang yang tidak berhak tidak dapat masuk dan melakukan sesuatu pada aplikasi koperasi simpan pinjam.Demi keamanan kemudian diberikan validasi menampilkan karakter yang diketikan dalam bentuk karakter bintang (*****).

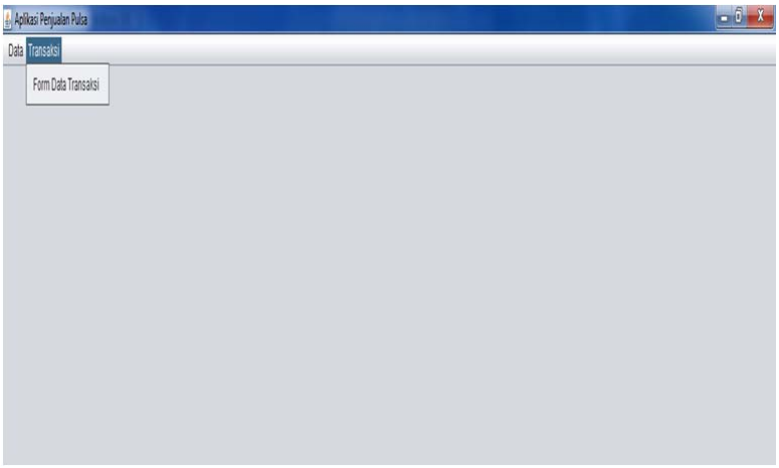
Berikut adalah gambaran tampilan *Login* :



Gambar 4.6.1. Tampilan Login

4.6.2. Implementasi Frame Menu Utama

Setelah Login berhasil Dilakukan maka akan muncul tampilan menu utama, berikut adalah tampilan menu utama.



Gambar 4.6.2. Tampilan Menu Utama

Untuk pengolahan data barang dilakukan di menu data, sedangkan untuk melakukan pengisian data transaksi dilakukan pada menu transaksi.

4.6.3. Implementasi Data Pulsa

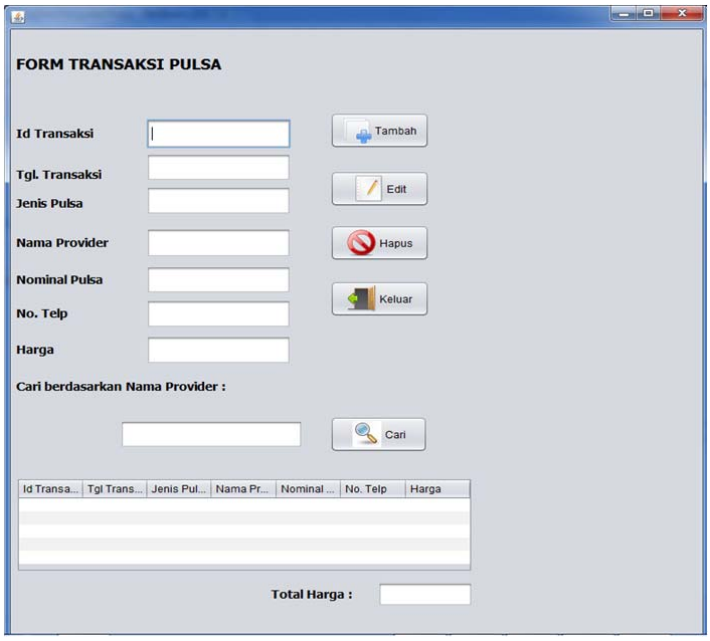
Untuk pengisian data pulsa, berikut tampilan menu data barang, dalam hal ini data pulsa.

Gambar 4.6.3 Tampilan Data Pulsa

Data pulsa digunakan untuk mengolah data pulsa elektrik & pulsa voucher yaitu berupa Id pulsa, jenis pulsa, nama provider, harga beli, harga jual, serta stok saldo.Untuk menambah data pulsa pertama kita masukkan data dengan lengkap setelah itu klik tombol Tambah untuk menyimpan datanya. Untuk merubah data pulsa klik tombol Edit lalu klik data yang ingin diedit pada tabel, setelah itu ubah datanya untuk menyelesaikan perubahan data. Untuk menghapus data klik tombol Hapus lalu pilih data yang ingin dihapus pada tabel, setelah itu datapun akan terhapus.

4.6.4. Implementasi Transaksi

Untuk melakukan transaksi penjualan pulsa, pengguna harus masuk ke menu transaksi lalu pilih form transaksi pulsa. Berikut adalah gambar tampilan form menu transaksi.



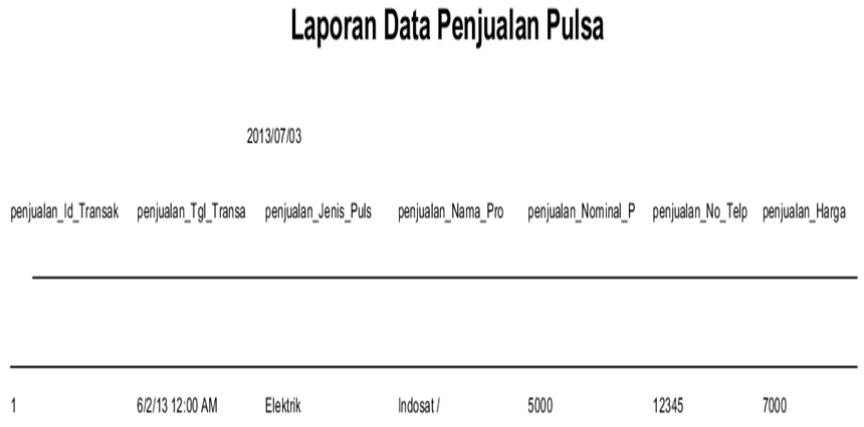
Ganbar 4.6.4. Tampilan Form Transaksi Pulsa

Dalam melakukan proses transaksi penjualan pulsa, maka pengguna mengisi id transaksi, tanggal transaksi, jenis pulsa, nama provider, nominal pulsa yang diinginkan, serta harga pulsa yang dibeli. Untuk menambah data transaksi klik tombol Tambah, lalu masukkan data dengan lengkap, untuk menyimpan datanya. Untuk merubah data transaksi klik tombol Edit lalu klik data yang ingin

diedit pada tabel, setelah itu ubah datanya untuk menyelesaikan perubahan data. Untuk menghapus data klik tombol Hapus lalu pilih data yang ingin dihapus pada tabel, setelah itu data pun terhapus.

4.6.5. Implementasi Laporan Pengolahan Data

Untuk laporan pada aplikasi ini akan menampilkan laporan pengolahan data dari keseluruhan penjualan pulsa baik itu pulsa elektrik maupun voucher. Laporan yang ditampilkan berupa ireport. Berikut tampilan laporan pengolahan data.



Gambar 4.6. Tampilan Laporan Pengolahan Data

4.7. Pembahasan Hasil Program

Program yang dibuat telah dapat menggantikan transaksi penjualan secara tradisional, dimana program dapat memberikan kemudahan untuk pengguna baik dalam hal laporan maupun pada saat transaksi penjualan, program juga dibuat untuk mudah dimengerti walaupun oleh pengguna yang baru menggunakannya. Program juga dapat meminimalisir kesalahan dari pengguna pada saat transaksi penjualan. Dalam hal laporan, pengguna

dipermudah untuk melihat total keuntungan yang diperoleh baik secara harian, bulanan atau tahunan.

BAB V
PENUTUP

5.1. Kesimpulan

Berdasarkan hasil ujicoba pada tahap sebelumnya, diperoleh bobot nilai yang baik untuk setiap point yang diuji seperti tampilan program,aantara lain

1. Penggunaan program yang mudah dimengerti, kegunaan program, fasilitas pada program dan ketepatan program. Hal ini membuktikan bahwa program yang dibuat sangat baik digunakan untuk menggantikan transaksi penjualan pulsa elektrik maupun voucher secara tradisional dan penggunaan buku untuk pencatatan laporan.
2. Penggunaan program juga tidak dapat dilakukan oleh sembarang orang karena dilengkapi dengan sistem *login*.
3. Transaksi penjualan pulsa elektrik pengguna hanya diminta untuk memilih operator customer dan besarnya nominal yang dibeli oleh customer, selanjutnya program yang akan mengirim kode SMS ke provider penyedia layanan pulsa elektrik.

5.2 Saran

Penulis menyadari aplikasi ini masih perlu pengembangan yang lebih lanjut agar aplikasi ini benar-benar menjadi aplikasi yang dapat mempermudah user dalam melakukan segala aktivitas yang berhubungan dengan pencatatan berbagai macam transaksi yang terjadi, antara lain :

1. Dalam pengembangannya diharapkan sistem ini tidak hanya mampu melakukan pencatatan transaksi tetapi juga mampu melakukan pengisian pulsa.
2. Diharapkan sistem ini mampu dikembangkan menjadi sistem yang dapat melakukan transaksi secara online, sehingga pelanggan yang enggan berjalan ke toko mampu mengisi ulang pulsa dengan ketentuan pelanggan harus terkoneksi dengan internet.

3. Penambahan gambar maupun animasi juga disarankan agar tampilan aplikasi lebih menarik dan tidak terlihat monoton.
4. Untuk data stok barang nantinya agar dapat secara otomatis berkurang pada saat transaksi dilakukan dan dapat menjadi rekomendasi kedepannya.

BAB VI

DAFTAR PUSTAKA

1. Didi indra Saputra.(13 oktober 2009).Pengertian-java.(online).(http://globalonlinebook.blogspot.com/2010/02/pengertian-java.html). 03 juni 2013
2. Nurjaya.(2 Januari 2010).Devinisi dan Sejarah MySQL.(online).(http://www.blogofnurjaya.com/2010/01/definisi-dan-sejarah-mysql.html#axzz2M5DMExnj). 10 mei 2013
3. Wartawarga.(24 April 2012).Keunggulan Java.(online).(http://wartawarga.gunadarma.ac.id/2012/04/keunggulan-java/).15 mei 2013
4. Scott Kendall, Rosenberg Doug. Applying Use Case Driven Object Modeling with UML: An Annotated e-Commerce Example.June 14, 2001.
5. Wahana Komputer(2005). Buku Latihan Membuat Aplikasi Profesional dengan Java + CD Wahana Komputer. Jakarta: Penerbit PT Elex Media Komputindo.