BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Usaha pertokoan merupakan usaha yang banyak dilakukan di Indonesia. Toko sembako merupakan sebuah usaha yang cukup diminati. Bisnis ini bisa dimulai dengan modal kecil hingga bermodal besar. Meski begitu, persaingan bisnis ini cukup ketat. Tentunya, untuk memenangkan pasar diperlukan strategi-strategi yang matang.

Diantaranya ada toko sembako, toko alat tulis, toko pecah belah dan sebagainya. Setiap pertokoan melakukan transaksi penjualan, disetiap transaksi penjualan akan tersimpan data penjualan yang akan menjadi laporan harian penjualan seperti laporan penjualan. Dari sekian banyak toko di indonesia masih banyak yang memiliki masalah di dalam laporan harian toko.

Salah satu cara untuk memperbaiki kinerja penjualan pada bisnis pertokoaan adalah dengan menerapkan aplikasi pengadaan dan penjualan. Dengan adanya aplikasi pengadaan (pembelian) dan penjualan maka transaksi di toko akan mudah di dokumentasikan dengan baik, akurat, dan efisien.

Beberapa contoh aplikasi pertokoaan yang cukup sukses digunakan adalah:

1. Sistem Informasi Alfamart

Alfamart adalah sebuah brand minimarket penyediaan kebutuhan hidup sehari-hari oleh PT.sumber alfaria trijaya,tbk. Lajunya pertumbuhan gerai toko alfamart yang cepat dengan transaksi lebih dari empat puluh transaksi struk perbulan, dapat terlaksana karena didukung oleh system teritegrasi pada setiap *poin of sales* (POS) kasir disemua gerai yang mencakup system penjualan, persediaan, dan penerimaan barang. tehnologi di pos kasir dirancang sudah memenuhi kebutuhan perkembangan dan transaksi di masa depan. Untuk mempercepat pelayanan dan keyamanan belanja dikasir, alfamart menggunakan pemandaian scanner barcode, pembayaran pun kini memberikan kemudahan bagi

konsumen dengan menggunakan bca debit, mandiri debit dan berbagai macam bank yang tercantum.

Persediaan operasi toko adalah salah satu system yang sudah ditetapkan perusahaan agar kegiatan di toko berjalan dengan lancar atau sempurna. Adapun pengendalian toko tersebut adalah:

- pengendalian persediaan
- pengendalian penjualan
- pengendalian biaya
- pengendalian administrasi
- pendapat lain-lain
- pengendalian lingkungan

2. Sistem Informasi Hypermart

Josephine (2010) Hypermart menyediakan segala kebutuhan pokok sehari-hari. Untuk itu Hypermart membutuhkan banyak informasi demi kelangsungan kegiatan perdagangan dalam Hypermart tersebut. Sistem informasi yang dibutuhkan oleh Hypermart tersebut dalam beberapa hal, yaitu:

a. Informasi harga

Harga adalah elemen penting yang harus diketahui oleh Hypermart. Karena dengan mengetahui harga pasar maka Hypermart dapat menentukan tarif harga yang akan digunakan dalam kegiatan penjualan tersebut. Harga juga menjadi tolak ukur konsumen dalam menentukan apakah ia akan belanja di Hypermart atau di supermarket lain, karena dalam bisnis supermarket persaingan sangat ditentukan melalui harga.

b. Informasi Konsumen

Konsumen juga merupakan suatu yang penting dalam kegiatan penjualan di Hypermart. Dalam hal ini Hypermart harus tahu apakah konsumen di sekitar Hypermart adalah masyarakat dengan ekonomi menengah ke atas atau kebawah agar Hypermart dengan mudah dapat menentukan produk apa saja yang akan dijual di Hypermart tersebut.

c. Informasi barang

Produk yang akan dijual di dalam Hypermart bervariasi. untuk dapat menentukan barang yang bagaimana dengan merek apa yang akan dijual oleh Hypermart tersebut dibutuhkan beberapa informasi.

Contohnya informasi merk barang tersebut, karena kadang-kadang konsumen memilih suatu barang karena merk dari barang tersebut terkenal, ditambah lagi dengan harga yang lebih murah. Setelah informasi diatas dimiliki maka sebuah Hypermart dapat melakukan kegiatan transaksi jual belinya dengan secara normal.

Cara pendistribusian barang pada Hypermart. Barang-barang yang ada di dalam Hypermart tentunya sudah didistribusikan dari pusat nya dan dituju ke cabang-cabang Hypermart.

Josephine (2010) Dengan adanya perkembangan teknologi, Hypermart bekerja sama dengan Bank BCA dan Mandiri Yaitu konsumen dapat membayar harga barang yang dibelinya dengan cara : debit BCA dan Debit Mandiri, selain itu menyediakan tarik tunai dengan syarat berbelanja di Hypermart minimal Rp.25000, tarik tunai BCA maksimal Rp.500000 dan minimalnya Rp.50000. Manfaat Penggunaan Barcode Dan Mesin Kasir

Josephine (2010) Dalam sistem manajemen penjualan, dengan menggunakan barcode kita akan mendapatkan informasi yang sangat detail dari banyak aspek usaha yang kita tekuni,memungkinkan kita dalam proses pengambilan keputusan dilakukan dengan percaya diri dan tepat sebagai misal:

a. Proses penjualan yang cepat sehingga dapat mengidentifikasi secara cepat dan tepat serta melakukan pemesanan kembali (*re-order*) barang dari *supplier* dengan cepat dan mampu mengimbangi tingkat permintaan barang oleh konsumen.
b. Mampu mengetahui barang yang lakunya lambat (*slow moving*) sehingga mencegah pemesanan barang yang tidak bergerak dan menguntungkan bagi aliran dana (*cash flow*) perusahaan.

c. Pergerakan penjualan produk dapat di monitor dari kecepatan perputarannya serta tingkat profitabilitasnya dan memungkinkan untuk produk tersebut mendapat ruang yang bagus untuk di pajang.
d. Catatan data penjualan secara periodik dapat digunakan untuk memprediksi loncatan penjualan musiman.
e. Informasi mengenai item produk dapat diketahui di rak bila ada harga promo maupun kenaikan harga.

Josephine (2010) Alasan utama dari penggunaan sistem barcode adalah mempermudah sistem kerja dan mengurangi biaya karena mampu bekerja lebih efesien.

Mesin Kasir (*Cash Register*)

Mesin Kasir (*Cash register*)adalah suatu peralatan mekanik maupun elektronik untuk menghitung dan mencatat transaksi penjualan yang biasanya terintegrasi secara modul dengan laci (*cash drawer*) untuk menyimpan sejumlah mata uang. *Cash register* umumnya juga mengeluarkan hasil cetak (*print*) dari struk penjualan (*receipt*) untuk pelanggan.

Generasi Pertama Mesin Kasir

Pada umumnya laci (*drawer)* mesin kasir atau *cash register* akan terbuka secara otomatis setelah ada penjualan atau transaksi kalaupun tidak yang bisa membuka hanya pengawas atau pemilik. Ini bertujuan untuk mengurangi resiko dari kehilangan dan pencurian.

Hampir semua mesin kasir memiliki tombol NS (*No Sale*) yang bertujuan untuk membuka laci tanpa adanya transaksi dan tombol inipun bila digunakan akan terekam dalam laporan akhir oleh pengawas atau pemilik. Ada beberapa mesin kasir yang saat ini dilengkapi sandi/ password untuk melakukan transaksi seperti NS tadi.

Beberapa fungsi lainnya dari mesin kasir/ *cash register* juga digunakan untuk mencatatkan komponen pajak dalam penjualan. Saat ini beberapa mesin kasir elektronik (*Electronic Cash Register*) bisa disambungkan dengan perangkat bantu lainnya seperti timbangan digital, barcode scanner, juga pembaca kartu kredit atau kartu debit.dan perkembangannya saat ini menagarah pada penggunaan

mesin kasir yang berbasiskan komputer (PC *Based Cash Register / Point of Sale* POS).

Mesin kasir yang berbasiskan komputer biasanya juga dilengkapi dengan *software/* piranti lunak baik yang berbasiskan sistem operasi DOS,Windows, Linux maupun Unix dimana data tersimpan dalam database baik di mesin kasir tersebut maupun di server induknya. dan umumnya banyak Mesin kasir yang berbasiskan komputer ini memiliki konfigurasi jaringan lokal (LAN)

Keunggulan Mesin kasir dibandingkan dengan sekedar software penjualan biasa adalah di sistem keamanannya karena selain dari sistem perangkatnya pun dilengkapi dengan kunci pengaman. Beberapa merek mesin kasir meliputi CASIO, NCR, IBM, *Wincor-Nixdorf, Sharp, Uniwell Toshiba* TEC

3. Sistem Informasi Indomaret

Indomaret merupakan jaringan minimarket yang menyediakan kebutuhan pokok dan kebutuhan sehari-hari dengan luas penjualan kurang dari 200 M2. Dikelola oleh PT Indomarco Prismatama, cikal bakal pembukaan Indomaret di Kalimantan dan toko pertama dibuka di Ancol, Jakarta Utara. (Sitanggang, 2014)

Laju pertumbuhan gerai Indomaret yang pesat dengan jumlah transaksi 14,99 juta transaksi per bulan didukung oleh sistem teknologi yang handal. Sistem teknologi informasi Indomaret pada setiap point of sales di setiap gerai mencakup sistem penjualan, persediaan dan penerimaan barang. Sistem ini dirancang untuk memenuhi kebutuhan saat ini dengan memperhatikan perkembangan jumlah gerai dan jumlah transaksi di masa mendatang. Indomaret berupaya meningkatkan pelayanan dan kenyamanan belanja konsumen dengan menerapkan sistem check out yang menggunakan scanner di setiap kasir dan pemasangan fasilitas pembayaran Debit BCA.(Sitanggang, 2014)

Pada setiap pusat distribusi diterapkan *digital picking system* (DPS). Sistem teknologi informasi ini memungkinkan pelayanan permintaan dan suplai barang dari pusat distribusi ke toko-toko dengan tingkat kecepatan yang tinggi dan efisiensi yang optimal.(Sitanggang, 2014)

Tiga sistem dari Alfamart, Indomaret, dan Hypermart di atas, sukses diterapkan pada bisnis mereka, namun untuk penerapan pada skala bisnis yang

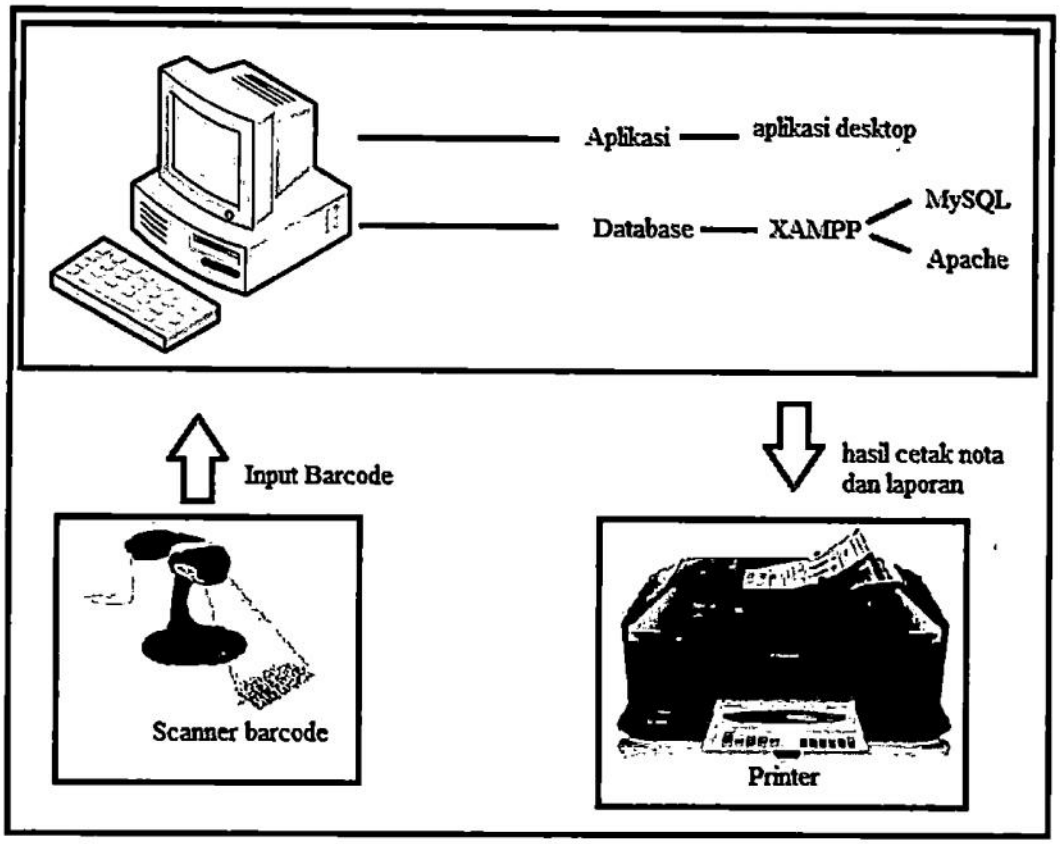
lebih kecil atau model toko yang lain membutuhkan system yang berbeda pula. Selain itu system mereka dirancang dengan tujuan yang sesuai dengan tujuan perusahaan, sehingga dalam penelitian ini dirancang model yang berbeda karena hanya digunakan untuk memenuhi kebutuh pengelolaan keuangan yang dipicu dari proses transaksi.

2.2 Landasan Teori

2.2.1 Arsitektur

Arsitektur aplikasi merupakan suatu desain aplikasi yang terdiri dari komponen-komponen yang saling berinteraksi satu sama lain. Biasanya juga disebut dengan infrastruktur aplikasi. Cara komunikasi komponen-komponen tersebut melalui network atau jaringan yang saling terhubung. Terdapat beberapa macam arsitektur aplikasi, di antaranya *Stand Alone, Client Server* (*Two Tier*), dan *Three Tier*. Selain ketiga arsitektur tersebut, *Clustering* dan DRC (*Disaster Recovery Center*) merupakan suatu metode tambahan pada arsitektur aplikasi yang lazim digunakan untuk menjaga *availability* suatu sistem.

Untuk membangun sebuah sistem, diperlukan rancangan penyusun sistem tersebut, karena itu arsitektur *software* ini berutujuan untuk menggambarkan bagaimana sistem ini akan dibagun dan dijalankan. Arsitektur *software* dapat dilihat pada Gambar 2.1.



Gambar 2.1 Arsitektur

2.2.2 Sistem Manajemen Workflow

Penggunaan sistem *manajemen workflow* memungkinkan perusahaan dapat mencapai efisiensi dan efektifitas *operational* perusahaan yang dapat meningkatkan produktifitas perusahaan.

Pengertian *workflow* adalah aliran kerja atau suatu informasi dari proses bisnis, baik secara keseluruhan maupun sebagian dimana dokumen atau informasi tugas tersebut diteruskan dari satu partisipan ke partisipan lain sesuai dengan prosedur atau ketentuan yang berlaku.

Menurut *WFMC (Worliflow Management Coalition),* yang merupakan perintis standarisasi pembentukan *Worliflow; Worliflow* didefinisikan sebagai berikut:

"77ze automation of business process, in a whole or part, during which document, information or tasks are passed from one participant to

another jiJr action, according to a set of procedural rules"

2.2.3 UML

Perancangan aplikasi toko menggunakan UML seabagai bahasa untuk menspesifikasi, memvisualisasi, membangun dan mendokumentasikan *artifacts* (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, *artifact* tersebut dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya. Selain itu UML adalah bahasa pemodelan yang menggunakan konsep orientasi object. UML dibuat oleh Grady Booch, James Rumbaugh, dan Ivar Jacobson di bawah bendera Rational Software Corp. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai perspektif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan.(Riyadi, 2004)

2.2.3.1 Konsep dasar UML

Menurut Noran (2000), UML dapat dianggap penerus dari *Object Oriented Analysis and Desain* (OOAD), metode yang berkembangan biak selama perang metode era 80-an dan 90-an yang merupakan awal UML. Hanya kompponen bahasa metode dan dilengkapi secara terpisah oleh *Rational Unified Process* (RUP).

*Unified Modelling Language (UML) m*enawarkan 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 *modelling* aplikasi prosedural dalam VB atau C.

Menurut *Object Management Group* (OMG), Unified Modelling Language (UML) didefinisikan sebagai bahasa grafis untuk visualisasi, menentukan, membangun, dan mendokumentasikan dalam sistem perangkat lunak-intensif.

2.2.3.2 Aturan UML

Menurut Noran (2000), UML adalah bahasa pemodelan. Oleh karena itu, UML mengandung satu set simbol (simbol) dan sekelompok aturan (semantik) yang mengelola bahasa. Aturan dapat diklasifikasikan menjadi:

- Sintaksis: menentukan aspek dan kombinasi aturan.
- Semantik: menentukan arti dari simbol-simbol, secara individu dalam konteks.
- Pragmatis: panduan tentang bagaimana menggunakan bahasa (maksud dari simbol-simbol).

2.2.3.3 Bagian-bagian Utama Diagram UML

Menurut Riyadi (2004), Bagian-bagian utama dari UML adalah *view, diagram, model element, dan general mechanism.*

1. View

View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. *View* bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah diagram. Beberapa jenis *view* dalam UML antara lain: *use case view, logical view, component view, concurrency view,* dan *deployment view. Use case view* Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan external actors. *Actor* yang berinteraksi dengan sistem dapat berupa *user* atau sistem lainnya. *View* ini digambarkan dalam *use case diagrams* dan kadang-kadang dengan *activity diagrams*. *View* ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang (*developer*), dan penguji sistem (*tester*).

Logical view mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class, object, dan relationship*) dan kolaborasi dinamis yang terjadi ketika object mengirim pesan ke object lain dalam suatu fungsi tertentu. *View* ini digambarkan dalam *class diagrams* untuk struktur statis dan dalam *state,*

sequence, collaboration, dan *activity* diagram untuk model dinamisnya. *View* ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

Component view mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari *code module* diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi administrative lainnya. *View* ini digambarkan dalam *component view* dan digunakan untuk pengembang (*developer*).

Concurrency view membagi sistem ke dalam proses dan prosesor. View ini digambarkan dalam diagram dinamis (*state, sequence, collaboration, dan activity diagrams*) dan diagram implementasi *(component dan deployment diagrams*) serta digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

Deployment view mendeskripsikan fisik dari sistem seperti komputer dan perangkat (*nodes*) dan bagaimana hubungannya dengan lainnya. *View* ini digambarkan dalam *deployment diagrams* dan digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

2. Diagram

Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu *view* tertentu dan ketika digambarkan biasanya dialokasikan untuk *view* tertentu.

2.2.3.4 *Use Case* Diagram

Menggambarkan sejumlah *external actors* dan hubungannya ke *use case* yang diberikan oleh sistem. *Use case* adalah deskripsi fungsi yang disediakan oleh sistem dalam bentuk teks sebagai dokumentasi dari *use case symbol* namun dapat juga dilakukan dalam *activity diagrams*. *Use case* digambarkan hanya yang dilihat dari luar oleh actor (keadaan lingkungan sistem yang dilihat *user*) dan bukan bagaimana fungsi yang ada di dalam sistem. Komponen-komponen yang terlibat dalam *use case* diagram:

1. Actor

Pada dasarnya *actor* bukanlah bagian dari *use case* diagram, namun untuk dapat terciptanya suatu *use case* diagram diperlukan beberapa *actor* dimana *actor* tersebut mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem. Sebuah *actor* mungkin hanya memberikan informasi inputan pada sistem, hanya menerima informasi dari sistem atau keduanya menerima dan memberi informasi pada sistem, actor hanya berinteraksi dengan *use case* tetapi tidak memiliki kontrol atas *use case*. *Actor* digambarkan dengan stick man. *Actor* dapat digambarkan secara secara umum atau spesifik, dimana untuk membedakannya dapat digunakan relationship.

2. Use Case

Use case adalah gambaran fungsionalitas dari suatu sistem, sehingga *customer* atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

3. Relasi dalam *Use Case*

Ada beberapa relasi yang terdapat pada *use case* diagram:

- *Association*, menghubungkan link antar element.
- *Generalization*, disebut juga *inheritance* (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.
- *Dependency*, sebuah element bergantung dalam beberapa cara ke element lainnya.
- *Aggregation*, bentuk assosiation dimana sebuah elemen berisi elemen lainnya.

Tipe relasi/stereotype yang mungkin terjadi pada *use case* diagram:

1. <<include>>, yaitu kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari *use case* lainnya.
2. <<extends>>, kelakuan yang hanya berjalan di bawah kondisi tertentu seperti menggerakkan alarm.
3. <<communicates>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah *communicates association*. Ini

merupakan pilihan selama *asociasi* hanya tipe ralationship yang dibolehkan antara *actor* dan *use case*.

2.2.3.5 Activity Diagram

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi.

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses pararel yang mungkin terjadi pada beberapa eksekusi (Dharwiyanti,2003).

Activity diagram merupakan state diagram khusus, dimana sebagian besar *state* adalah action dan sebagian besar transisi di-trigger oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem)secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari *level* atas secara umum (Dharwiyanti,2003).

Sebuah aktivitas dapat direalisasikan oleh satu *usecase* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *usecase* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas (Dharwiyanti,2003).

2.2.3.6 Class Diagram

Class diagram sebuah spesifikasi yang jika diinstansikan 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) menurut Dharwiyanti (2003).

2.2.4 Database (Basis Data)

Secara konsep basis data atau database adalah kumpulan dari data-data yang membentuk suatu berkas (*file*) yang saling berhubungan (*relation*) dengan

tata cara tetentu untuk membentuk data baru atau informasi. Kumpulan dari datayang saling berhubungan (*relasi*) antara satu dengan lainnya yang diorganisasikan berdasarkan skema atau struktur tertentu (Supriyanto, 2005).

Menurut Fabbri dan Schwab (Kadir, 1999), basis data adalah sistem berkas terpadu yang dirancang terutama untuk meminimalkan pengulangan data. Secara nyata, basis data merupakan kumpulan berkas-berkas yang memiliki kaitan antara satu dengan yang lain sehingga membentuk suatu bangunan data. *Database Management System* (DBMS) adalah perangkat lunak yang didisain untuk membantu dalam hal pengorganisasian dan pemeliharaan data.

Basis data dapat didefinisikan dalam beberapa sudut pandang antara lain:

- Himpunan kelompok data yang saling berhubungan yang terorganisir sehingga dapat dimanfaatkan.
- Kumpulan data yang berhubungan secara bersama tanpa pebgulangan yang tidak perlu untuk memenuhi kebutuhan.

2.2.4.1 Model *Database* (Basis Data)

Model basis data menyatakan hubungan antar rekaman yang tersimpan dalam basis data (Kadir, 1999). Berikut ini adalah macam-macam model data :

1. *Model hirarkis (hierarchical model)*

   Model hirarkis biasa disebut model pohon. Model ini menggunakan pola hubungan orangtua-anak (*parent - child*). Simpul yang terhubung ke simpul pada level dibawahnya disebut orangtua (*parent*). Setiap orangtua bisa memiliki lebih dari satu anak, tetapi anak hanya memiliki satu orangtua. Simpul yang dibawahi oleh simpul orangtua disebut anak (*child*). Simpul orangtua yang tidak memiliki orangtua disebut akar (*root*). Simpul yang tidak memiliki anak disebut daun. Adapun hubungan antara orangtua dan anak disebut cabang (Kadir, 1999).

2. Model jaringan (*network model*)

   Model ini menyerupai model hirarkis, dengan perbedaan suatu simpul anak bisa memiliki lebih dari satu orangtua.

3. Model keterhubungan entitas (*entity-relationship model*)

Model ini berisi komponen-komponen himpunan entitas dan relasi yang masing-masing dilengkapi dengan atribut-atribut, dan dapat digambarkan dengan menggunakan *Diagram Entity-Relationship* (Diagram E-R).

4. Model relasional

   Model ini menggunakan sekumpulan tabel berdimensi dua (yang disebut relasi atau tabel), dengan masing-masing relasi tersusun atas tupel atau baris menurut Kadir (1999).Model data relasional mengandung komponen inti sebagai berikut :

   - Struktur data, data-data diorganisasikan dalam bentuk tabel dengan baris baris dan kolom-kolom.
   - Manipulasi data, operasi yang sangat berdaya-guna (menggunakan *Structured Query Languange*) digunakan untuk memanipulasi data-data yang disimpan di relasi-relasi.
   - Integritas data, fasilitas-fasilitas untuk menspesifikasi aturan bisnis yang memelihara integritas data saat mereka dimanipulasi.

2.2.4.2 MySQL Sebagai Database Server

MySQL merupakan sebuah sistem manajemen basis data relasi (*relational database management system*) yang bersifat "terbuka" (*open source*). MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses basis datanya (Kadir, 2003).

Menurut Nugroho (2005), MySQL adalah sebuah program *database server* yang mampu menerima dan mengirimkan data dengan sangat cepat, *multi user* serta menggunakan perintah standar SQL.

Software database mulai bermunculan seiring dengan bertambahnya kebutuhan akan database server. Salah satu dari pendatang baru dalam dunia database ialah MySQL, sebuah server /klien database SQL yang berasal dari Skandinavia. MySQL terdiri atas server SQL *client* program untuk mengakse *server*, *tools* untuk administrasi, dan *interface* program untuk menulis program sendiri.

2.2.4.3 Mengapa Memilih MySQL

Dwi Apri Setyorini, S.Kom mengatakan Jika mencari system manajemen *database* yang murah atau bahkan gratis ada beberapa pilihan antara lain yQL, mSQL, PostgresSQL, atau salah satu dari produk vendor komersial yang gratis. Ketika dibandingkan antara MySQL dengan system database yang lain, maka perlu dipikirkan apa yang paling penting. Apakah performa, support, fitur-fitur SQL, kondisi keamanan dalam liensi, atau masalah harga. Dengan pertimbangan tersebut, MySQL memiliki banyak yang bisa ditawarkan antara lain:

- Kecepatan : banyak ahli berpendapat MySQL merupakan server tercepat.
- Kemudahan penggunaa : MySQL punya performa tinggi namun merupakan database yang simple sehingga mudah disetup dan dikonfigurasi.
- Harga : MySQL cenderung gratis untuk penggunaan tertentu.
- Mendukung query language : MysqL mengerti bahasa SQL (*Structur Query Language*) yang merupakan *system database* modern. Anda juga dapat mengakse MySQL lewat protocol ODBC (*Open Database Connectivity*) buatan Microsoft.
- Kapasitas dan sekuritas : *Databse* MySQL dapat diakses dari semua tempat di internet dengan hak akses tertentu.
- Pertabilitas : MySQL dapat berjalan dalam banyak varian UNIX dengan baik, sebaik seperti saat berjalan si system non-UNIX.
- Distribusi yang terbuka : MySQL mudah didapatkan dan memiliki *source code* yang boleh disebarluaskan sehingga dikembangkan lebih lanjut.

2.2.5 Bahasa Pemrograman C#

Visual C#.NET adalah sebuah bahasa pemrograman yang handal, cepat, mendukung penuh OOP (*Object Oriented Programing*), serta tersedia fasilitas GUI. Visual C#.NET ini memiliki banyak keunggulan dibanding dengan bahasa pemrograman yang terdahulu seperti Visual Basic.NET atau Java yaitu lebih kuat, stabil, dan produktif (Budiharto dan Sukmadi, 2004). Keunggulan dari Visual C#.NET lainnya adalah:

C# (baca : *See-Sharp*) adalah bahasa pemrograman baru yang diciptakan Microsoft yang digunakan oleh banyak *developer* .NET untuk mengembangkan applikas dengan *platform* .NET

Kelebihan C# antara lain yaitu: **(Anda, 2013)**

- ***Simple***. C# bersifat sederhana, karena bahasa ini didasarkan kepada bahasa C dan C++. Jika anda familiar dengan C dan C++ atau bahkan Java, anda akan menemukan aspek-aspek yang begitu familiar, seperti *statements*, *expression*, *operators*, dan beberapa fungsi yang diadopsi langsung dari C dan C++, tetapi dengan berbagai perbaikan yang membuat bahasanya menjadi lebih sederhana.
- ***Object Oriented Language***. C# memenuhi syarat-syarat sebagai sebuah bahasa pemrograman yang bersifat *Object Oriented* yaitu *encapsulation, inheritance* dan *polymorphism*.
- ***Powerfull dan Fleksibel***. C# bisa digunakan untuk membuat berbagai macam aplikasi, seperti aplikasi pengolah kata, grapik, *spreadsheets*, atau bahkan membuat kompiler untuk sebuah bahasa pemrograman.
- **Efisien**. C# tidak memiliki terlalu banyak *keyword*, sehingga dapat mengurangi kerumitan.
- **Modular**. Kode C# ditulis dengan pembagian masing *Class-class* (*classes*) yang terdiri dari beberapa routines yang disebut sebagai *member methods*. *Class-class* dan metode-metode ini dapat digunakan kembali oleh program atau aplikasi lain. Hanya dengan memberikan informasi yang dibutuhkan oleh Class dan metode yang dimaksud, maka kita akan dapat membuat suata kode yang dapat digunakan oleh satu atau beberapa aplikasi dan program (*reusable code*)

  Setelah kita membaca dan mengetahui apa itu bahasa C# dan mengapa di baca c sharp , sekarang kita harus mengetahui apa saja kelebihan dan kekurangan bahasa C#:

Kekurangan C# :

- Banyaknya operator serta fleksibilitas penulisan program kadang-kadang membingungkan pemakai

- Bagi pemula pada umumnya akan kesulitan menggunakan *pointer*.

2.2.6 Perangkat Lunak Sistem

Perangkat lunak yang digunakan untuk membuat aplikasi toko tersebut yaitu XAMPP, Visual Studio 2010 dan *crytal report*.

2.2.6.1 XAMPP

Damayanti (2014) XAMPP adalah perangkat lunak bebas yang merupakan paket instalasi untuk Apache, MySQL, PHP dan Perl. Dengan menggunakan XAMPP tidak perlu lagi menginstal ketiga *software* itu secara terpisah. XAMPP dikembangkan dari sebuah tim proyek bernama Apache *Friends,* yang terdiri dari Tim Inti (*Core Team*), Tim Pengembang (*Development Team*) dan Tim Dukungan (*Support Team*).

Bagian penting dari XAMPP yang biasa digunakan pada umumnya:htdoc adalah *folder* te mpat meletakkan berkas-berkas yang akan dijalankan, seperti berkasPHP, HTML dan skrip lain. phpMyAdmin merupakan bagian untuk mengelola basis data MySQL yang ada dikomputer. Kontrol panel yang berfungsi untuk mengelola layanan *service* XAMPP. Seperti menghentikan *stop* layanan, ataupun memulai *start*.

Software XAMPP versi ini terdiri atas:

- Apache versi 2.0.54
- MySQL versi 4.1.12
- PHP versi 5.0.4
- phpMyAdmin versi 2.6.2-p11 dan lain-lain

XAMPP digunakan sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri atas program Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl.

Kelebihan dari XAMPP:

- Dapat berperan sebagai server web Apache untuk simulasi pengembangan website. Tool pengembangan web ini mendukung teknologi web populer seperti PHP, MySQL, dan Perl.

- Melalui program ini, programmer web dapat menguji aplikasi web yang dikembangkan dan mempresentasikannya ke pihak lain secara langsung dari komputer, tanpa perlu terkoneksi ke internet.
- XAMPP juga dilengkapi fitur manajemen database PHPMyAdmin seperti pada server hosting sungguhan, sehingga pengembang web dapat mengembangkan aplikasi web berbasis database secara mudah.
- XAMPP dapat dijalankan di sistem operasi Windows 2000/XP/Vista/7 dan sistem operasi lain.

2.2.6.2 Visual Studio

Merupakan sebuah perangkat lunak lengkap (*suite*) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi *console*, aplikasi Windows, ataupun aplikasi Web. Visual Studio mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro, dan Visual SourceSafe.

2.2.6.3 Crytal Report

Crystal Report adalah program Penbuat Laporan dari *seagate. Corp* yang dibuat untuk membantu *user* untuk membuat laporan dengan mudah tanpa mengunakan *Data Environment* dan *Data Report*, dimana di *Crystal Report* tersebut bisa menggunakan fasilitas Expert untuk membantu mendesain laporan sesara mudah.

Pada *Crystal Report* dapat terdiri dari satu atau beberapa tabel, *query*, dan *report*. Sebuah *Report* tidak harus memiliki ketiga elemen yang disebutkan. Kita dapat menyebutkan kumpulan data kita sebuah database kendati hanya ada sebuah tabel didalamnya. Yang pasti, dalam sebuah *report* haruslah terdapat sebuah tabel karena tabel atau entiti dalam model relasional digunakan untuk mendukung antar muka komunikasi antara pemakai dengan para pengguna komputer. Dalam tabel

tersebut merupakan *source* atau sumber dari item-item data yang diorganisasikan dalam bentuk laporan.

Beberapa kelebihan dari *Crystal Report* ini adalah:

- Dari segi pembuatan laporan tidak terlalu rumit yang memungkinkan para programmer pemula sekalipun dapat membuat laporan yang sederhana tanpa melibatkan banyak kode program.
- Integrasi dengan bahasa-bahasa pemograman lain yang memungkinkan dapat digunakan oleh banyak programmer dengan masing-masing
- keahlian.
- Fasilitas impor hasil laporan yang mendukung format-format popular seperti *Microsoft Word, Excel, Acces, Adobe Acrobat Reader*, HTML, dan sebagainya.