BAB II
LANDASAN TEORI

2.1. Tinjauan Studi (*Relate Research* / Penelitian Sebelumnya)

Tinjauan studi dilakukan dengan membandingkan dokumen - dokumen penelitian sejenis seperti jurnal, skripsi, thesis dan tulisan - tulisan ilmiah tentang sistem informasi inventori sebelumnya dengan mempelajarinya untuk mendapatkan kelebihan dan kekurangan dalam penelitian tersebut. Dengan cara seperti ini, penelitian sebelumnya dapat dijadikan referensi dalam penggunaan metode yang akan diteliti. Sumber tinjauan studi penelitian sejenis yang dipergunakan didalam proposal skripsi ini adalah tinjauan studi hasil dari penelitian atau hasil penulisan karya ilmiah khususnya berkaitan dengan sistem informasi inventori. Terdapat 3 jurnal yang dijadikan bahan pembelajaran bagi peneliti dalam menunjukkan kekurangannya. Penelitan tersebut sebagai berikut :

1. Sistem Informasi Penjualan Dan Pengendalian Persediaan Barang Menggunakan Metode *Economic Order Quantity* (Eoq) Menggunakan Bahasa Pemrograman Java Dan Database Mysql Pada Toko Kansa Elpiji, yang disusun oleh Ir. Zefriyenni, MM dan Budi Santoso. Menggunakan metode SDLC (*System Development Life Cycle*) sebagai metode pengembangan sistem. Data yang digunakan adalah data penjualan pada Toko Kansa Elpiji. Dengan hasil sebagai berikut:
   a. Dengan menggunakan metode EOQ dapat membantu meminimalkan terjadinya pemesanan yang berlebihan atau kekurangan.
   b. Dengan sistem informasi penjuaalan dan pengendalian stock barang menggunakan metode EOQ dapat memberikankemudahan terhadap bagian penjualan, pimpinan dan bagian gudang dalam mencetak laporan yang dibutuhkan oleh pihak toko Kansa Elpiji.
   c. Dengan menggunakan program java dapat memudahkan dalam proses transaksi penjualan maupun pembelian. Karena data disimpan ke dalam database yang mudah dicari apabila diperlukan.

2. Rancang Bangun Sistem Informasi Inventory Barang Berbasis Web Studi Kasus di PT. Infinetworks Global Jakarta, yang disusun oleh Agus Heryanto, Hilmi Fuad, dan Dani Dananggi. Metode yang digunakan adalah metode berorientasi objek. Data yang digunakan adalah data inentori pada PT. Infinetworks Global Jakarta. Dengan hasil sebagai berikut:
    a. Pekerjaan dalam mencatat dan mengolah data seluruh barang dapat dilakukan dengan semakin mudah.
    b. Semakin mudah dalam melakukan pencarian pengguna barang.
    c. Resiko kehilangan data semakin kecil.
    d. Semakin mudah dan cepat dalam penyusunan laporan.
3. Perancangan Sistem Informasi *Inventory Sparepart* Mesin *Fotocopy* Dengan Menggunakan Visual Delphi 7 (Studi Kasus Di UD. Eka Taruna Madiun, yang disusun oleh Fatim Nugrahanti. Data yang digunakan adalah data sparepart pada UD. Eka Taruna Madiun. Dengan hasil sebagai berikut:
    a. Dengan sistem Informasi *Inventory Sparepart* mesin *Fotocopy* yang dirancang menggunakan program Delphi dengan database Ms. Acces, dapat memonitoring keluar masuk barang.
    b. Sistem Informasi *Inventory Sparepart* mesin *fotocopy* ini sangat bermanfaat dalam pencarian data *sparepart* dan membantu dalam pencarian data *supplier* maupun konsumen UD. Eka Taruna Madiun.
    c. Sistem informasi *Inventory Sparepart* mesin *fotocopy* ini memudahkan karyawan bagian administrasi dalam mengolah data,
    d. Serta sistem ini mampu meningkatkan efisiensi waktu bagi kinerja perusahaan dan meminimalisir semua kemungkinan dalam manipulasi data dankesalahan pencatatan.

Berdasarkan hasil dari tinjauan studi penelitian sebelumnya maka penelitian ini mengusulkan Pengembangan Sistem Informasi Inventori Berbasis Java NetBeans Pada CV. Bagaskara Galih Perkasa Jepara. Perbedaan penelitian ini dari penelitian sebelumnya adalah adanya fitur notifikasi list barang - barang

yang persediaannya sudah menipis atau habis; input data barang dilengkapi dengan link untuk melihat gambar barang; data dapat diakses dari komputer manapun, selama ada aplikasi inventori ini dan didukung dengan koneksi internet.

2.2. Tinjauan Pustaka

2.2.1. Konsep Dasar Sistem Informasi

2.2.1.1. Pengertian Sistem

Menurut Sutabri (2004), sistem adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama – sama untuk mencapai tujuan tertentu.

Sedangkan menurut Jogiyanto (2005), Sistem adalah kumpulan dari komponen atau elemen yang saling berhubungan satu dengan lainnya membentuk satu kesatuan untuk mencapai tujuan tertentu.

Berdasarkan pengertian diatas dapat disimpulkan, sistem adalah kumpulan dari beberapa unsur komponen yang saling berhubungan menjadi satu kesatuan untuk mencapai tujuan tertentu.

2.2.1.1.1. Karakteristik Sistem

Suatu sistem memiliki karakteristik atau sifat-sifat tertentu, yaitu (Sutabri, 2004) :

1. Komponen – komponen (*Components*)

   Suatu sistem terdiri dari sejumlah komponen yang sering disebut dengan subsistem yang saling berintraksi, yang artinya saling bekerjasama membentuk satu kesatuan. Komponen – komponen sistem dapat berupa suatu subsistem atau bagian – bagian dari sistem. Setiap subsistem memiliki sifat – sifat dari sistem untuk menjalankan suatu fungsi tertentu dan menpengaruhi proses sistem secara keseluruhan.

2. Batas Sistem (*Boundary*)

   Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang

sebagai satu kesatuan. Batas suatu sistem menunjukan ruang lingkup (*scope*) sistem itu sendiri.

3. Lingkungan Luar Sistem (*Environments*)

   Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat merugikan sistem tersebut.

4. Penghubung Sistem (*Interface*)

   Penghubung merupakan media penghubung antara subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber – sumber daya mengalir dari satu subsistem ke subsistem yang lainnya.

5. Masukkan Sistem (*Input*)

   Masukkan yaituenergi yang dimasukkan ke dalam sistem, dimana dapat berupa masukkan perawatan (*maintenance input*) dan masukkan signal (*signal input*). Masukkan perawatan adalah energi yang diinputkan supaya sistem tersebut dapat beroperasi, sedangkan masukkan signal adalah energi yang diproses untuk didapatkan keluaran.

6. Keluaran Sistem (*Output*)

   Keluaran yaitu hasil energi yang diolah dan diklasifikasikan menjadi yang berguna dan sisa pembuangan.

7. Pengolah Sistem

   Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah input menjadi output.

8. Sasaran Sistem (*Objective*)

   Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (*objective*). Apabila suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya.

2.2.1.1.2. Klasifikasi Sistem

Sistem dapa diklasifikasikan dari berbagai sudut pandang, diantaranya adalah sebagai berikut (Sutabri, 2004) :

1. Sistem abstrak dan sistem fisik

   Sistem abstrak adalah suatu sistem yang berupa ide – ide pemikiran atau ide – ide yang tidak tampak secara fisik, misalnya teologi yaitu sistem yang berupa pemikiran – pemikiran hubungan antara manusia dengan tuhan.

   Sedangkan sistem fisik adalah suatu sistem yang berupa pemikiran atau ide – ide yang nyata atau yang ada secara fisik. Misalnya sistem komputer, sistem akuntansi, sistem produksi dan lain sebagainya.

2. Sistem alamiah dan sistem buatan manusia

   Sistem alamiah aadalah sistem yang terjadi melalui proses alam, tidak dibuat manusia. Misalnya: sistem perputaran bumi. Sistem buatan manusia yang melibatkan interaksi antara manusia dengan mesin disebut dengan *human – machine system* atau ada yang menyebut dengan *man – machine system*.

3. Sistem tertentu dan sistem tak tentu

   Sistem tertentu beroperasi dengan tingkah laku yang sudah dapat diprediksi. Contohnya: sistem komputer. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengan dung unsur probabilitas.

4. Sistem tertutup dan sistem terbuka

   Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak berpengaruh dengan lingkungan luarnya. Sedangkan sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan.

2.2.1.2. Pengertian Informasi

Menurut Jogiyanto (2005), Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang. Sedangkan menurut McLeod informasi sebagai data yang telah diolah menjadi bentuk yang lebih berarti bagi penerimanya.

Berdasarkan pengertian diatas, informasi adalah suatu data yang telah diolah sehingga dapat digunakan sebagai dasar pengambilan suatu keputusan.

2.2.1.2.1. Kualitas Informasi

Informasi yang baik adalah informasi yang berkualitas, informasi yang berkualitas ditentukan oleh beberapa hal, yaitu (Sutabri, 2004) :

1. Akurat (*accurate*)

   Informasi harus bebas dari kesalahan – kesalahan dan tidak menyesatkan, informasi harus jelas mencerminkan maksudnya.

2. Tepat waktu (*time lines*)

   Informasi yang dihasilkan atau dibutuhkan tidak boleh terlambat, karena nantinya tidak mempunyai nilai yang baik, sehingga apabila dijadikan dasar dalam pengambilan keputusan akan berakibat fatal atau kesalahan pengambilan keputusan dan tindakan.

3. Relevan (*relevance*)

   Informasi harus memberikan manfaat yang baik untuk pemakai informasi tersebut.

2.2.1.2.2. Nilai Informasi

Nilai informasi didasarkan atas sepuluh sifat, yaitu (Sutabri, 2004) :

1. Mudah diperoleh

   Sifat ini menunjukkan kemudahan dan kecepatan untuk memperoleh informasi.

2. Luas dan lengkap

   Sifat ini menunjukkan kelengkapan isi informasi.

3. Ketelitian

   Sifat ini berhubungan dengan tingkat kebebasan dari kesalahan keseluruhan informasi.

4. Kecocokan

   Sifat ini menunjukkan seberapa baik keluaran informasi dalam hubungannya dengan permintaan para pemakai.

5. Ketepatan waktu

   Sifat ini berhubungan dengan waktu yang dilalui, yang lebih pendek dari siklus untuk mendapatkan informasi.

6. Kejelasan

   Sifat ini menunjukkan tingkat kejelasan informasi.

7. Keluwesan

   Sifat ini berhubungan dengan apakah informasi tersebut dapat digunakan untuk membuat lebih dari satu keputusan, tetapi apakah juga dapat digunakan untuk lebih dari seorang pengambil keputusan.

8. Dapat dibuktikan

   Sifat ini menunjukkan sejauh mana informasi itu dapat diuji oleh beberapa pemakai hingga sampai didapat kesimpulan yang sama.

9. Tidak ada prasangka

10. Dapat diukur

    Sifat ini menunjukkan hakikat informasi yang dihasilkan oleh sistem informasi formal.

2.2.1.3. Sistem Informasi

Sistem informasi bukan merupakan hal yang baru. Yang baru adalah komputerisasinya. Sebelum ada komputer, teknik penyaluran informasi yang memungkinkan manajer merencanakan serta mengendalikan operasi yang telah ada. Komputer menambahkan satu atau dua dimensi, seperti kecepatan, ketelitian dan penyedian data dengan volume yang lebih besar yang memberikan bahan pertimbangan yang lebih banyak untuk mengambil keputusan. Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan

kepada pihak luar tertentu dengan laporan – laporan yang diperlukan (Sutabri, 2004).

2.2.2. Konsep Dasar Inventori

2.2.2.1. Pengertian Inventori

Menurut Ristono (2009), Inventori adalah bahan-bahan, atau bagian yang disediakan, dan bahan-bahan dalam proses yang terdapat dalam perusahaan untuk proses produksi, serta barang-barang jadi atau produk yang disediakan untuk memenuhi permintaan dari konsumen atau pelanggan setiap waktu.Persediaan atau inventori dapat memiliki berbagai fungsi penting yang menambah fleksibilitas dari operasi suatu organisasi. Dengan adanya inventori dapat mempermudah dan melancarkan jalannya suatu proses bisnis. Sebuah Sistem inventori adalah kumpulan dari kebijakan-kebijakan dan kontrol dari tiap-tiap tingkatan persediaan barang dan menentukan tingkatan mana yang harus dipelihara dan berapa besar jumlah barang yang harus disimpan.

Menurut Bahagia (2006), Inventori adalah suatu sumber daya menganggur (idle resource) yang keberadaannya menunggu proses lebih lanjut. Yang dimaksud dengan proses lebih lanjut disini dapat berupa kegiatan produksi seperti dijumpai pada sistem manufaktur, kegiatan pemasaran seperti dijumpai pada sistem distribusi, ataupun kegiatan konsumsi seperti yang dijumpai pada sistem rumah tangga, perkantoran, dan sebagainya.

Dan menurut Heryanto, dkk. (2014), Inventory adalah sejumlah sumber daya baik berbentuk bahan mentah ataupun barang jadi yang disediakan perusahaan untuk memenuhi permintaan dari konsumen. Sedangkan pengertian inventory dalam definisi lainnya adalah suatu teknik untuk manajemen material yang berkaitan dengan persediaan.

Jadi dapat disimpulkan bahwa inventori adalah suatu persediaan sumber daya yang berupa bahan-bahan mentah serta bahan-bahan jadi yang disiapkan oleh perusahaan / organisasi untuk memenuhi kebutuhan proses produksi atau kegiatan pemasaran agar tidak terjadi kekurangan

ataupun kelebihan persediaan bahan yang dapat menimbulkan kerugian pada perusahaan / organisasi.

2.2.2.1.1. Fungsi Inventori

Menurut Bahagia (2006), Inventori dibutuhkan karena beberapa alasan, menurut Buchan dan Koenigsberg menjabarkan tiga jenis motif, yaitu motif transaksi (*transaction motive*), motif berjaga-jaga (*precautionary motive*), serta motif berspekulasi (*Ispeculative motive*), berikut penjelasannya :

1. Motif Transaksi (*Transaction Motive*)

   Motif transaksi merupakan motif utama mengapa keberadaan inventori dibutuhkan, yaitu motif untuk menjamin pemenuhan permintaan barang. Oleh karena itu ada atau tidaknya barang merupakan indikator utama dari dipenuhi atau tidaknya motif ini. Besar Minimum inventori yang diperlukan untuk menjamin kelancaran proses pemenuhan permintaan pemakai disebut inventori/stok operasi. Besarnya stok operasi ini pada prinsipnya tergantung pada besarnya waktu ancang-ancang dan banyaknya kebutuhan barang per satuan waktu. Dengan kata lain, besar stok operasi ini minimal sebesar kebutuhan barang selama waktu ancang-ancangnya.

2. Motif Berjaga - Jaga (*Precautionary Motive*)

   Motif berjaga-jaga timbul jika terjadi adanya ketidakpastian baik ketidakpastian dari sisi pemakai barang (*user*). Besarnya inventori yang ditujukan untuk untuk meredam ketidakpastian ini disebut sebagai inventori pengaman. Ada dua jenis inventori pengaman. cadangan pengaman (*safety stock*), bila ketidakpastian tersebut datangnya dari pemakai, dan cadangan penyangga (*buffer stock*), bila ketidakpastian tersebut berasal dari pemasok. Dengan demikian, semakin besar ketidapastian maka akan semakin besar pula inventori pengaman.

3. Motif Berspekulasi (*Ispeculative Motive*)

Berlainan dengan kedua motif diatas, pada motif ini keberadaan inventori timbul karena adanya keinginan untuk melakukan spekulasi dengan tujuan mendapatkan keuntungan yang berlipat ganda dari kenaikan harga barang di masa mendatang. Faktor spekulasi yang biasa terjadi pada barang-barang yang langka dipasaran atau barang-barang yang dipasarkan dengan sistem monopolitik.

2.2.2.2. Sistem Informasi Inventori

Menurut Bahagia (2006), Sistem informasi inventori adalah sistem informasi yang mengelola data transaksi dan persediaan dalam gudang. Perusahaan yang bergerak dibidang produksi umumnya memerlukan sistem inventori. Sistem ini harus dapat memberikan informasi inventori secara cepat dan akurat, selain itu sistem diharapkan dapat mempermudah kerja *user.*

Sedangkan menurut Susanto (2010), Sistem informasi inventori adalah suatu sistem untuk mengumpulkan dan memelihara data yang menjelaskan mengenai persediaan barang, mengubah data tersebut menjadi informasi, dan melaporkan informasi kepada pemakai.

Jadi dapat disimpulkan bahwa sistem informasi inventori adalah sistem yang mengelola data transaksi dan persediaan barang menjadi sebuah informasi yang bermanfaat bagi penggunanya.

2.2.3. Aplikasi Dalam Perancangan Sistem Informasi

2.2.3.1. Java

Java adalah suatu teknologi di dunia software komputer, yang merupakan suatu bahasa pemrograman, dan sekaligus suatu *platform*. Sebagai bahasa pemrograman, Java dikenal sebagai bahasa pemrograman tingkat tinggi. Java mudah dipelajari, terutama bagi programmer yang telah mengenal C/C++. Java merupakan bahasa pemrograman berorientasi objek yang merupakan paradigma pemrograman masa depan. Sebagai bahasa pemrograman Java dirancang menjadi handal dan aman. Java juga

dirancang agar dapat dijalankan di semua *platform*. Dan juga dirancang untuk menghasilkan aplikasi – aplikasi dengan performansi yang terbaik, seperti aplikasi database Oracle 8i/9i yang *core*-nya dibangun dengan bahasa pemrograman Java. Sedangkan Java bersifat *neutral architecture*, karena *Java Compiler* yang digunakan untuk mengkompilasi kode program Java dirancang untuk menghasilkan kode yang netral terhadap semua arsitektur perangkat keras yang disebut sebagai *Java Bytecode*. Sebagai sebuah *platform*, Java terdiri atas dua bagian utama (Suyanto, 2014), yaitu :

1. *Java Virtual Machine* (JVM).
2. *Java Application Programing Interface* (Java API).

2.2.3.1.1. Arsitektur Java

a. *Enterprise Java* (J2EE) untuk aplikasi berbasis web, aplikasi sistem tersebar dengan beraneka ragam klien dengan kompleksitas yang tinggi. Merupakan superset dari Standar Java.
b. *Standart Java (J2SE)*, yang biasa dikenal dengan sebagai bahasa Java.
c. *Micro Java (J2ME)*, merupakan subset dari J2SE dan dan salah satu aplikasinya yang banyak dipakai adalah untuk *wireless device / mobile device.*

2.2.3.1.2. Fitur – Fitur Java

a. *Applet* adalah program Java yang dapat berjalan di atas browser, yang dapat membuat halaman HTML lebih dinamis dan menarik.
b. *Java Networking* adalah Sekumpulan API (*Application Programming Interface*) yang menyediakan fungsi-fungsi dari aplikasi-aplikasi jaringan, seperti penyediaan akses untuk TCP, UDP, IP Adress dan URL. Tetapi *Java Networking* tidak menyediakan akses untuk ICMP dikarenakan alasan keamanan

dan pada kondisi umum hanya administrator (*root*) yang bisa memanfaatkan protokol ICMP.

c. *Java Database Connectivity* (JDBC) menyediakan sekumpulan API yang dapat digunakan untuk mengakses database seperti Oracle, MySQL, PotgreSQL, Microsoft SQL Server.
d. *Java Security* menyediakan sekumpulan API untuk mengatur *security* dari aplikasi Java baik secara *high level* atau *low level*, seperti *public/private key management* dan *certificates*.
e. *Java Swing* menyediakan sekumpulan API untuk membangun aplikasi-aplikasi GUI (*Graphical User Interface*) dan model GUI yang diinginkan bisa bermacam-macam, bisa model Java, model Motif / CDE atau model yang *dependent* terhadap *platform* yang digunakan.
f. Java RMI menyediakan sekumpulan API untuk membangun Aplikasi-aplikasi Java yang mirip dengan model RPC (*Remote Procedure Call*) jadi objek-objek Java bisa di *call* secara *remote* pada jaringan komputer.
g. Java 2D / 3D menyediakan sekumpulan API untuk membangun grafik-grafik 2D/3D yang menarik dan juga akses printer.
h. *Java Server Pages,* Berkembang dari *Java Serlet* yang digunakan untuk menggantikan aplikasi-aplikasi CGI, JSP(*Java Server Pages*) yang mirip ASP dan PHP merupakan alternatif terbaik untuk solusi internet.
i. JNI (*Java Native Interface*) menyediakan sekumpulan API yang digunakan untuk mengakses fungsi-fungsi pada library (*.dll atau *.so) yang dibuat dengan bahasa pemrograman lain seperti C, C++, dan Basic.
j. *Java Sound* Menyediakan sekumpulan API untuk memanipulasi *sound*.
k. Java IDL + CORBA Java IDL (*Interface Definition Language*) menyediakan dukungan Java untuk implementasi CORBA (*Common Object Request Broker*) yang merupakan model

distributed-object untuk solusi aplikasi besar di dunia *networking*.

l. *Java Card*, utamanya digunakan untuk aplikasi-aplikasi pada smart card, yang sederhana wujudnya seperti *SIM card* pada *handphone*.

m. JTAPI (*Java Telephony API*) menyediakan sekumpulan API untuk memanfaatkan *device-device telephony*, sehingga akan cocok untuk aplikasi-aplikasi CTI (*Computer Telephony Integration*) yang dibutuhkan seperti ACD (*Automatic Call Distribution*), PC-PBX dan lainnya.

2.2.3.2. NetBeans IDE (Integrated Development Environment)

NetBeans IDE ini digunakan oleh pengembang aplikasi untuk melakukan pemrograman, kompilasi, mencari kesalahan, dan menjalankan aplikasi yang telah dibuat. NetBeans IDE sendiri dibuat dengan menggunakan bahasa pemrograman Java, namun untuk membuat aplikasi yang dapat digunakan dalam sebuah perngkat komputer, maupun *mobile*, IDE ini mampu menukung bahasa pemrograman lain.

Beberapa bahasa pemrograman yang didukung oleh NetBeans IDE ini terdiri dari Java, C/C++, PHP, XML,HTML, Javadoc, Java Script, JSP, dan masih banyak lagi. Selain itu, pengembang dapat memasang plugin, maupun modul yang bisa didapatkan di komunitas untuk mendukung bahasa pemrograman lain agar dapat dijalankan pada NetBeans IDE.

Tampilan antarmuka pada NetBeans IDE bisa dibilang cukup memudahkan pengembang aplikasi dalam melakukan interaksi secara grafikal yang tidak hanya berupa kode saja sehingga dapat dilihat hasil kodenya secara langsung dalam bentuk grafis.

Selain itu, *library* yang disediakan NetBeans bisa dibilang cukup lengkap untuk beberapa bahasa pemrograman sehingga pengembang aplikasi pemula pun dapat mempelajari langsung *library* yang dibutuhkan dalam membuat aplikasi. NetBeans IDE bisa berjalan di sistem operasi Windows, MAC OS, dan Linux 32/64 bit (Putra, 2014).

2.2.3.3. MySQL

MySQL merupakan software yang tergolong sebagai DBMS (*Database Management System*) yang bersifat *Open Source*. *Open Source* menyatakan bahwa software ini dilengkapi dengan *source code* (kode yang dipakai untuk membuat MySQL), selain tentu saja bentuk *executable*-nya atau kode yang dapat dijalankan secara langsung dalam sistem operasi, dan bisa diperoleh dengan cara men-*download* (mengunduh) di internet secara gratis.

MySQL awalnya dibuat oleh perusahaan konsultan bernama TcX yang berlokasi di Swedia. Saat ini pengembangan MySQL berada dibawah naungan perusahaan MySQL AB.

Sebagai software DBMS, MySQL memiliki sejumlah fitur sebagai berikut (Kadir, 2008) :

1. Multiplatform

   MySQL tersedia pada beberapa platform (Windows, Linux, Unix, dan lain-lain).

2. Andal, cepat, dan mudah digunakan

   MySQL termasuk sebagai database server (server yang melayani permintaan terhadap database) yang andal, dapat menangani database yang besar dengan kecepatan yang tinggi, mendukung banyak sekali fungsi untuk mengakses database, dan sekaligus mudah untuk digunakan. Berbagai tools pendukung juga tersedia (walaupun dibuat oleh pihak lain). Perlu diketahui, MySQL dapat menangani sebuah tabel yang berukuran dalam terabyte (1 terabyte = 1024 gigabyte). Namun, ukuran yang sesungguhnya sangat bergantung pada batasan sistem operasi. Sebagai contoh, pada sistem Solaris 9/10, batasan ukuran file sebesar 16 terabyte.

3. Jaminan keamanan akses

   MySQL mendukung pengamanan database dengan berbagai kriteria pengaksesan. Sebagai gambaran, dimungkinkan untuk mengatur user tertentu agar bisa mengakses data yang bersifat

rahasia (misalnya gaji pegawai), sedangkan user lain tidak boleh. MySQL juga mendukung konektivitas ke berbagai software. sebagai contoh dengan menggunakan ODBC (*Open Database Conectivity*), database yang ditangani MySQL dapat diakses melalui program yang dibuat dengan Visual Basic. MySQL juga mendukung program klien yang berbasis Java untuk berkomunikasi dengan database MySQL melalui JDBC (*Java Database Conectivity*). MySQL juga bisa diakes melalui aplikasi berbasis Web, misalkan menggunakan PHP.

4. Dukungan SQL

   Seperti tersirat dalam namanya, MySQL mendukung perintah SQL (*Structured Query Language*). Sebagaimana diketahui, SQL merupakan standar dalam pengaksesan database relasional (database yang tersusun atas sejumlah tabel). Pengetahuan akan SQL akan memudahkan siapapun untuk menggunakan MySQL.

2.2.3.4. MySQL Workbench

Pada saat MySQL dibuat sebagai versi *open source* dari SQL sebagai database RDBMS, MySQL tidak memiliki GUI sehingga perintahuntuk mendisain database didapat dari *command line* (perintah baris) yang harus dimasukkan satu per satu. Namun seiring dengan penggunaannya yang semakin luas MySQL banyak didukung dengan GUI dari pihak ketiga. salah satunya phpMyAdmin, yang terkenal karena kemudahannya. Saat ini MySQL merupakan produk milik Oracle, dan saat ini MySQL memiliki MyQL Workbench sebagai mesin untuk mendisain database dengan *interface* visual (Pratama, 2013).

2.2.3.5. Pengertian *Unified Modelling Language* (UML)

Unified Modelling Language (UML) adalah satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek (Whitten and Bentley, 2007).

Unified Modelling Language (UML) adalah sebuah “bahasa” yang telah menjadi Standar dalam industri untuk visualisasi, merancang dan

mendokumentasikan sistem piranti lunak (Dharwiyanti dan Wahono, 2003). UML memiliki beberapa diagram yaitu :

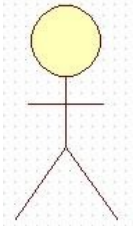
1) ***Use Case Diagram***

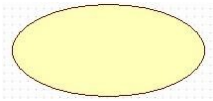
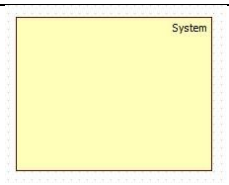
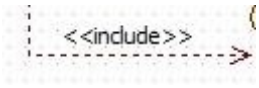
Diagram ini menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use Case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang / sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan tertentu.

Use case diagram dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan dengan klien, dan merancang *test case* untuk semua fitur 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 *behavior*-nya sendiri. Sementara hubungan generalisasi antara *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Tabel 2.1 Simbol UseCase Diagram

No.	Gambar	Nama	Keterangan
1		Actor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat.

2		UseCase	Deskripsi dari urutan aktifitas yang ditampilkan sistem dan menghasilkan suatu hasil yang terstruktur bagi aktor.
3		System	Menggambarkan paket yang menampilkan sistem.
4		Include	Menggambarkan usecase yang ditambahkan dan memerlukan usecase ini untuk menjalankan fungsinya
5		Association	Menghubungkan antara objek satu dengan objek lainnya.

2) *Activity Diagram*

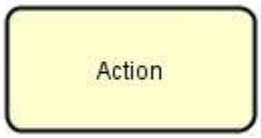
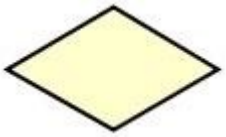
Diagram ini menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing – masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses pararel yang mungkin terjadi pada eksekusi.

Activity diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak 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.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebh. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

Tabel 2.2 Simbol Activity Diagram

No.	Gambar	Nama	Keterangan
1		Initial Node	Status awal ketika memulai diagram.
2		Action	Menggambarkan aktifitas dalam kegiatan sistem
3		Decision Node	Digunakan untuk menunjukkan suatu kegiatan yang memenuhi suatu kondisi.
4		Activity Final	Status akhir ketika mengakhiri diagram
5		Fork	Digunakan ketika berbagai aktifitas terjadi bersamaan
6		Control Flow	Digunakan untuk menghubungkan aliran aktifitas sistem.

3) *Squence Diagram*

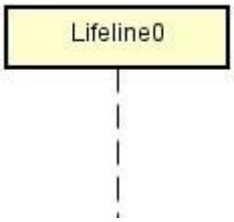
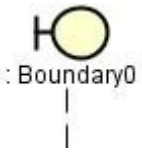
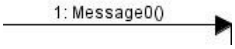
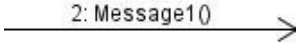
Diagram ini menggambarkan interaksi antara objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek – objek yang terkait).

Sequence diagram biasanya 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* aktifitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan.

Masing – masing objek, termasuk aktor, memiliki *lifeline* verrtikal. Message digambarkan sebagai garis berpanah, dari satu objek ke objek lainnya. Pada fase disain berikutnya, *message* akan dipetakan menjadi operasi / metoda dari *class*. *Activition bar* menunjukan 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*.

Tabel 2.3 Simbol Sequence Diagram

No.	Gambar	Nama	Keterangan
		Actor Lifeline	Actor menggambarkan entitas yang berada diluar sistem. Bisa berupa orang, perangkat keras atau sistem lain.

		General Lifeline	Merepresentasikan entitas tunggal dalam sequence diagram, digambarkan dengan kotak. Entitas ini memiliki nama, stereotype atau berupa instance (menggunakan instance:class)
		Boundary	Boundary biasanya berupa tepi dari system, seperti user interface, atau suatu alat yang berinteraksi dengan system lain.
		Synchronous Message	Relasi ini digunakan untuk memaggil operasi atau ethod yang dimiliki oleh suatu objek. Synchronous mengharuskan kita menyelesaikan 1 proses baru kemudian memanggil proses berikutnya.
		Asynchronous Message	Relasi ini digunakan untuk memanggil operasi atau method yang dimiliki oleh

			suatu objek. Asynchronous memberikan kita fasilitas untuk menjalankan proses lain ketika proses sebelumnya belum selesai.
		Reply Message	yang menggambarkan hasil dari pengiriman message

2.2.3.6. Pengertian Entity Relationship Diagram (ERD)

ERD merupakan gambaran grafis dari suatu model data yang menyertakan deskripsi detail dari seluruh entitas (*entity*), hubungan (*relationship*), dan batasan (*constraint*) untuk memenuhi kebutuhan sistem analis dalam menyelesaikan pengembangan sebuah sistem (Fernandes, 2017).

Sutanta (2011), menjabarkan komponen *Entity Relationship Diagram* (ERD) adalah sebagai berikut :

1. Entitas, entitas merupakan suatu objek yang dapat dibedakan dari lainnya yang dapat diwujudkan dalam basis data. Objek dasar dapat berupa orang, benda, atau hal yang keterangannya perlu disimpan didalam basis data. Untuk menggambarkan sebuah entitas digunakan aturan sebagai berikut :
    - Entitas dinyatakan dengan simbol persegi panjang.
    - Nama entitas dituliskan didalam simbol persegi panjang.
    - Nama entitas berupa kata benda, tunggal.
    - Nama entitas sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya dengan jelas.
2. Atribut, atribut merupakan keterangan-keterangan yang terkait pada sebuah entitas yang perlu disimpan dalam basis data. Atribut berfungsi

sebagai penjelas pada sebuah entitas. Untuk menggambarkan atribut digunakan aturan sebagai berikut:

- Atribut digambarkan dengan simbol ellips.
- Nama atribut dituliskan didalam simbol ellips.
- Nama atribut merupakan kata benda, tunggal.
- Nama atribut sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya dengan jelas.

3. Relasi, relasi merupakan hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda. Aturan penggambaran relasi adalah sebagai berikut :
   - Relasi dinyatakan dengan simbol belah ketupat.
   - Nama relasi dituliskan didalam simbol belah ketupat.
   - Nama relasi berupa kata kerja aktif.
   - Nama relasi sedapat mungkin menggunakan nama yang mudah dipahami dan dapat menyatakan maknanya dengan jelas.
4. *Line Conector*, Penghubung antara himpunan relasi
5. dengan himpunan entitas dan himpunan entitas dengan atribut dinyatakan dalam bentuk garis.
6. Kardinalitas, Krardinalitas relasi menunjukan jumlah maksimum tupel yang dapat berelasi dengan entitas pada entitas yang lain. Terdapat 3 macam kardinalitas relasi yang terdapat pada ERD , yaitu (Ayoe, https://www.scribd.com/doc/214997588/Pengertian-ERD) :
   1) Satu ke Satu (*One to One*)

      Hubungan relasi satu ke satu yaitu setiap entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas pada himpunan entitas B.
   2) Satu ke Banyak (*One to Many*)

      Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B, tetapi setiap entitas pada entitas B dapat berhubungan dengan satu entitas pada himpunan entitas A
   3) Banyak ke Banyak (*Many to Many*)

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B.

Tabel 2.4 Simbol ERD

No.	Gambar	Nama
1		Entity
2		Relationship
3		Attribute
4		Line Connector

2.2.4. Pengujian

Sebelum sistem informasi ini digunakan oleh *user* sepenuhnya, maka sistem informasi ini harus bebas dari kesalahan dan *error*. Sistem informasi ini harus diuji terlebih dahulu untuk mencari kesalahan yang mungkin dapat terjadi, seperti kesalahan dalam urutan proses, logika sistem, dan lain sebagainya.

2.2.4.1. Jenis Pengujian Perangkat Lunak

2.2.4.1.1. White-box Testing

White-Box Testing adalah cara untuk menguji suatu aplikasi atau *software* dengan cara melihat modul untuk dapat meneliti dan menganalisa kode dari program yang dibuat ada yang salah atau tidak . Jika modul yang sudah dihasilkan berupa *output* yang tidak sesuai dengan yang diharapkan

maka akan dikompilasi ulang dan dicek kembali kode – kode tersebut hingga sesuai dengan yang diharapkan (Nidhra and Dondetti, 2012).

2.2.4.1.2. Black-box Testing

Black-Box Testing adalah cara untuk menguji yang berfokus pada spesifikasi fungsional dari perangkat lunak. *Black-Box Testing* di desain untuk mengungkap kesalahan pada persyaratan fungsional tanpa mengabaikan kerja internal dari suatu program (Pressman, 2011).

Terdapat beberapa klasifikasi *black-box testing* menurut Simarmata (2010), Beberapa klasifikasi *black box testing* yang digunakan dalam penelitian ini adalah sebagai berikut :

1. Pengujian fungsional (*functional testing*), merupakan pengujian fitur dan perilaku operasional produk untuk memastikan apakah mereka sesuai dengan spesifikasinya. Pengujian ini mengabaikan mekanisme internal sistem atau komponen dan hanya berfokus pada keluaran yang dihasilkan sebagai respon terhadap *input* yang dipilih dan kondisi eksekusi.
2. Pengujian alfa (*alfa testing*), merupakan simulasi pengujian operasional atau aktual oleh pengguna potensial / pelanggan atau tim uji independen di situs pengembang. Pengujian alfa sering digunakan sebelum perangkat lunak dilanjutkan ke pengujian beta.
3. Pengujian beta (*beta testing*), dilakukan setelah pengujian alfa dan dapat dianggap sebagai bentuk eksternal pengujian penerimaan pengguna. Pengujian dari rilisnya suatu produk perangkat lunak yang dilakukan oleh pelanggan dan di luar tim pemrograman.

2.2.4.1.3. Perbandingan White Box Testing dan Black Box Testing

Perbandingan *white box testing* dan *black box testing* dapat dilihat dalam tabel 2.5 sebagai berikut (http://casilis.blogspot.co.id) :

Tabel 2.5 Perbandingan White box dan Black box Testing

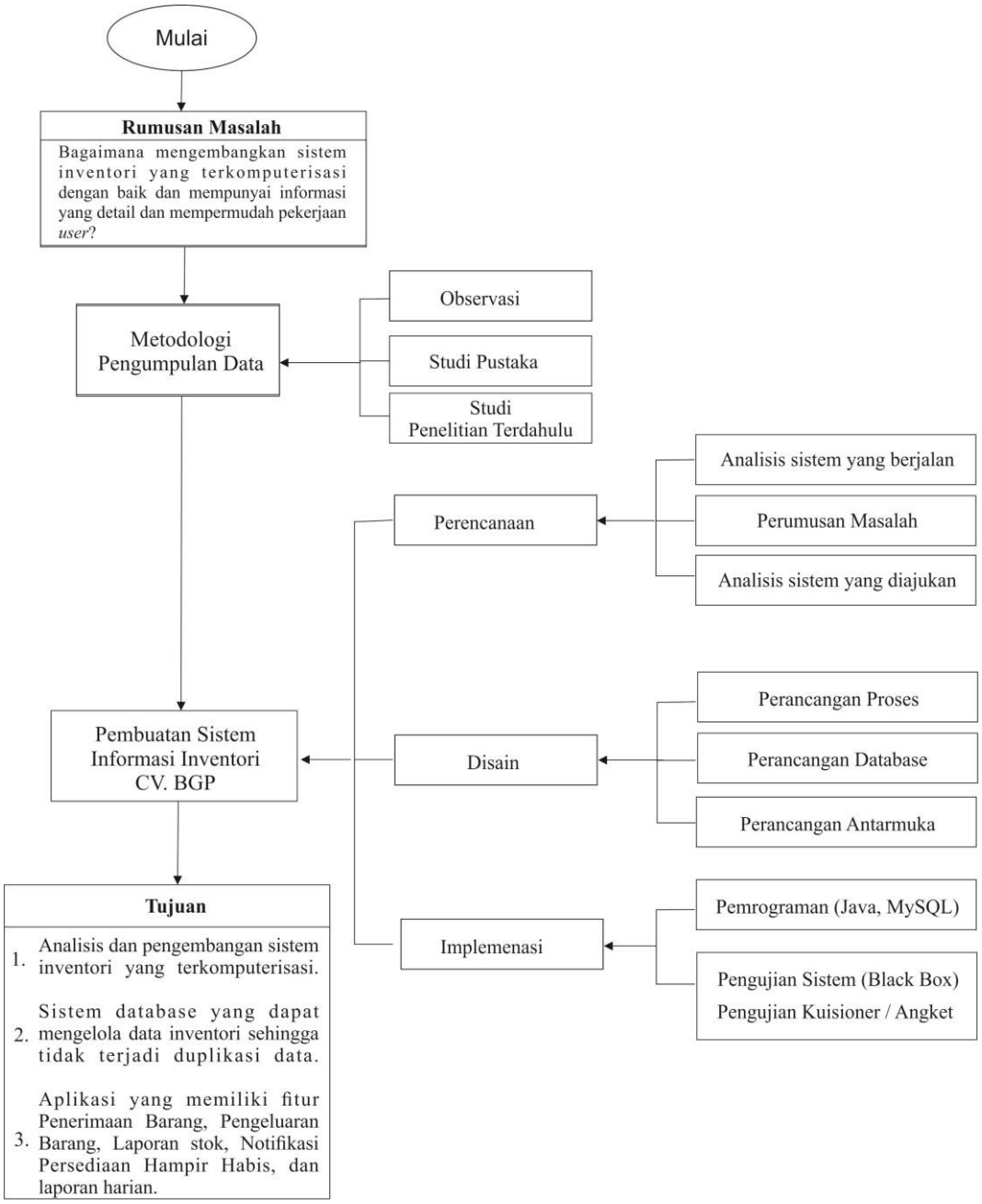
	Kelebihan	**Kekurangan**
White Box Testing	1. Sebagai *software engineer* yang memiliki akses ke *source code*, hal ini menjadi sangat mudah untuk melakukan skenario pengujian secara efektif. 2. Membantu *software engineer* untuk mengoptimalkan *source code.* 3. Baris kode yang tidak efisien dapat dihilangkan agar mencegah *bugs* pada program.	1. Karena dibutuhkan *software engineer* yang berpengalaman dalam *white box testing* sehingga mengeluarkan biaya tambahan. 2. Terkadang sangat sulit melihat setiap baris kode untuk mencari *bugs* pada program yang akan diuji.
Black Box Testing	1. Cocok untuk *source code* dengan skala besar. 2. Menguji dari sudut pandang *user*. 3. *Software tester* dalam jumlah yang banyak, dapat menguji program tersebut tanpa	1. *Software tester* hanya menjalankan beberapa skenario pengujian yang dipilih. 2. Pengujian yang tidak efisien karena *software tester* memiliki pengetahuan yang terbatas tentang program.

	harus memiliki pengetahuan tentang *programming.*	3. Pengujian yang tidak spesifik karena *software tester* tidak memiliki akses ke *source code*.

2.3. KERANGKA PEMIKIRAN

Kerangka Pemikiran berisi suatu bagan alur yang menghubungkan masalah dan pendekatan penelitian yang dihasilkan dari teori / konsep / model atau yang ada pada landasan teori yang akan dijadikan acuan dalam menyusun metodologi penelitian dan bisa juga digunakan untuk menguji logika penelitian. Kerangka pemikiran menjelaskan pola pikir dan konsep dalam melakukan penelitian. Penyusunan tentang sistem informasi inventori ini disusun dengan beberapa tahapan dalam sebuah kerang pemikiran.

Kerangka pemikiran ini dibuat sebagai acuan penyusunan dan pengembangan sistem yang dilakukan. Adapun kerangka pemikiran yang dibuat dapat dilihat pada gambar 2.1.



Gambar 2.1 Kerangka Pemikiran