BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Hasil Penelitian terdahulu dalam penelitian ini digunakan sebagai salah satu dasar untuk mendapatkan gambaran dalam penyusunan kerangka penelitain, dan menjadi kajian yang dapat mengembangkan penelitian yang akan dilakukan dikemudian hari.

Berikut adalah penelitian terdahulu yang dijadikan panduan atau acuan dalam melakukan penelitian :

1. Penelitian yang dilakukan oleh Rahmawati (2017).

" Sistem Informasi *Inventory* Stok Barang Pada CV. Artha Palembang". CV. Artha merupakan perusahaan yang bergerak di bidang distributor bahan bagunan sejak tahun 2011 dan memiliki perkembangan cukup pesat dan mampu menjual barang mencapai ribuan barang yang terjual. [18]

Tabel 2. 1 Perbedaan dan persamaan dengan penelitian terdahulu

No	Perbedaan	Persamaan
1.	1. Sistem yang dibuat oleh Rahmawati berfokus pada *Inventory* pemesanan dan penjualan barang. 2. sistem yang dibuat oleh Rahmawati mengguanakan terstruktur.	1. Berbasis *Web*. 2. melakukan pengelolaan barang masuk dan barang keluar. 3. Menggunakan bahasa pemrograman php.

2. Penelitian yang digunakan oleh Dahlan Abdullah dan Cut Ita Erliana (2014).

Penyimpanan barang pada perusahaan merupakan salah satu tugas manajemen yang mendukung sistem kinerja suatu perusahaan dan menjadi salah satu kebutuhan yang sangat penting bagi semua perusahaan salah satunya CV. Iltizam Coorperation yang umumnya masih dilakukan secara manual.

Tabel 2. 2 Perbedaan dan persamaan dengan penelitian terdahulu

No.	Perbedaan	Persamaan
1.	Sistem yang dibuat oleh Dahlan Abdullah dan Cut Ita Erliana berfokus pada *Inventory* pemesanan dan penjualan barang.	1. Berbasis *Web*. 2. Melakukan pengelolaan barang masuk dan barang keluar. 3. Menggunakan bahasa pemrograman php.

	sistem yang dibuat oleh Dahlan Abdullah dan Cut Ita Erliana mengguanakan terstruktur.	
2.	Sistem dibuat oleh Amsal Bengris adalah sistem yang menggunakan terstruktur. Sedangkan sistem yang dirancang oloh penulis adalah menggunakan OOP.	

2.2 Sistem Informasi

Sistem menurut L. James Havery adalah prosedur logis dan rasional guna melakukan atau merancang suatu rangkaian komponen yang berhubungan satu sama lain. Menurut Henry Prat Fairchild dan Eric Kohle sistem adalah sebuah rangkaian yang saling terkait antara beberapa bagian dari yang terkecil, jika suatu bagian atau sub bagian terganggu, maka bagian yang lainnya ikut merasakan ketergangguan tersebut. Menurut penulis sistem adalah suatu komponen atau bagian-bagian yang saling berhubungan satu sama lain untuk mencapai suatu tujuan tertentu.

Informasi menurut Jogiyanto HM, adalah hasil dari pengolahan data dalam suatu bentuk yang lebih berguna dan lebih berarti bagi penerimanya yang menggambarkan suatu kejadian yang nyata dan digunakan untuk pengambilan keputusan. Pengertian informasi menurut Abdul Kadir, McFadden dkk data yang

telah diproses sedemikian rupa sehingga meningkatkan pengetahuan seseorang yang menggunakan data tersebut. Sedangkan menurut penulis sendiri informasi adalah sebuah data yang telah memiliki nilai yang berguna bagi penerimanya.

Sistem informasi menurut Tata Sutabri, S.Kom., MM, 2005:36 merupakan suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi organisasi yang bersifat manajerial dalam kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan – laporan yang diperlukan. Menurut Kertahadi (2007) sistem informasi adalah alat untuk menyajikan informasi sedemikian rupa sehingga bermanfaat bagi penerimanya. Tujuannya adalah untuk memberikan informasi dalam perencanaan, memulai, pengorganisasian, operasional sebuah perusahaan yang melayani sinergi organisasi dalam proses mengendalikan pengambilan keputusan. Menurut penulis, sistem informasi adalah kumpulan data atau elemen yang memiliki nilai dan saling berhubungan antara satu sama lain yang membentuk satu kesatuan untuk mengintegrasikan data, memproses dan menyimpan, yang bertujuan untuk mencapai suatu tujuan tertentu.

2.2.1 Sistem Informasi *Inventory*

Inventory atau persediaan menurut Alexandri (2009) merupakan aktiva yang meliputi barang-barang milik perusahaan dengan maksud untuk dijual dalam suatu periode usaha tertentu atau persediaan barang-barang yang masih dalam pengerjaan atau proses produksi maupun persediaan bahan baku yang menunggu penggunaannya dalam proses produksi. Menurut Ristono (2009) *Inventory* adalah

barang-barang yang disimpan untuk digunakan atau dijual pada masa atau periode yang akan datang. Jadi bisa dikatakan sistem informasi *Inventory* adalah sistem informasi yang mengelola data transaksi atau persediaan barang yang tersedia di gudang, biasanya terdiri dari sistem barang keluar, sistem pembelian barang, dan sistem gudang guna memberikan informasi yang akurat serta membantu kinerja user.

2.2.2 Komponen Sistem Informasi

Menurut Yakub, sistem informasi merupakan sebuah susunan yang terdiri dari beberapa komponen atau elemen. [6] Komponen-komponen dari sistem informasi ini dapat digambarkan sebagai berikut ini :

1. Komponen Masukan (Input Block)

   Input merupakan data dan perintah yang dimasukan ke dalam sistem untuk menghasilkan sebuah nilai.

2. Komponen Model (Model Block)

   Kombinasi prosedur, logika, dan model matemetik yang akan memanipulasi data input dan data yang tersimpan di basis data yang sudah ditentukan untuk menghasilkan output yang diingiinkan.

3. Komponen Keluaran (Output Block)

   Hasil dari blok keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

4. Komponen Teknologi (Technology Block)

Teknologi merupakan kotak alat (tool box) dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan, dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara menyeluruh.

5. Blok Basis Data (Database Block)

   Merupakan kumpulan dari data yang saling berhubungan satu sama lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.

2.2.3 Analisis Kebutuhan Sistem

Pembangunan sistem informasi memerlukan penyelidikan dan analisis mengenai alasan timbulnya ide atau gagasan untuk membangun dan mengembangkan sistem informasi. Analisis dilakukan untuk melihat berbagai komponen yang dipakai sistem yang sedang berjalan meliputi *hardware, software,* jaringan dan sumber daya manusia. Analisis juga mendokumentasikan aktivitas sistem informasi meliputi *input,* pemrosesan, *output,* penyimpanan dan pengendalian.

Selanjutnya melakukan studi kelayakan *(feasibility study)* untuk merumuskan informasi yang dibutuhkan pemakai akhir, kebutuhan sumber daya, biaya, manfaat dan kelayakan proyek yang diusulkan.

Analisis kebutuhan sistem sebagai bagian dari studi awal bertujuan mengidentifikasi masalah dan kebutuhan spesifik sistem. Kebutuhan spesifik sistem adalah spesifikasi mengenai hal-hal yang akan dilakukan sistem ketika diimplementasikan. [7]

Analisis kebutuhan sistem harus mendefinisikan kebutuhan sistem yang spesifik antara lain :

1) Masukan yang diperlukan sistem *(input)*
2) Keluaran yang dihasilkan *(output)*
3) Operasi-operasi yang dilakukan (proses)
4) Sumber data yang ditangani
5) Pengendalian (kontrol)

Tahap analisis kebutuhan sistem memerlukan evaluasi untuk mengetahui kemampuan sistem dengan mendefinisikan apa yang seharusnya dapat dilakukan oleh sistem tersebut kemudian menentukan kriteria yang harus dipenuhi sistem. Beberapa kriteria yang harus dipenuhi adalah pencapaian tujuan, kecepatan, biaya, kualitas informasi yang dihasilkan, efisiensi dan produktivitas, ketelitian dan validitas dan kehandalan atau reliabilitas.

2.2.4 Desain Sistem

Menurut Burch dan Grundnitski, desain sistem dapat didefinisikan sebagai penggmbaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh dan berfungsi.

Desain sistem menentukan bagaimana suatu sistem akan menyelesaikan tahap ini menyangkut konfigurasi dari komponen-komponen perangkat lunak dan perangkat keras dari suatu sistem sehingga setelah instalasi dari sistem akan benar-benar memuaskan rancang bangun yang telah ditetapkan pada akhir tahap analisis sistem. [8]

2.2.5 Implementasi Sistem

Setelah prototipe diterima maka pada tahap ini merupakan implementasi sistem yang siap dioperasikan dan selanjutnya terjadi proses pembelajaran terhadap sistem baru dan membandingkannya dengan sistem lama, evaluasi secara teknis dan operasional serta interaksi pengguna, sistem dan teknologi informasi.

2.2.6 Pengujian : Black Box Testing

Menurut Black, tester menggunakan behavioral test (disebut juga Black Box Tests), sering digunakan untuk menemukan bug dalam high level operations, pada tingkatan fitur, profil operasional dan scenario *customer*. Tester dapat membuat pengujian fungsional black box berdasrkan pada apa yang harus sistem lakukan. [9].

Menurut Nidhra dan Dondeti, black box testing juga disebut functional testing, sebuah teknik pengujian fungsional yang merancang test case berdasarkan informasi dari spesifikasi. [10]

Pengujian *black box* adalah pengujian aspek fundamental sistem tanpa memperhatikan struktur logika internal perangkat lunak . Pengujian *black-box* merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan. Pengujian *black-box* berusaha menemukan kesalahan dalam kategori sebagai berikut :

a. Fungsi-fungsi yang tidak benar atau hilang.
b. Kesalahan *interface*.
c. Kesalahan dalam struktur data atau akses *database* eksternal.
d. Kesalahan kinerja.

e. Inisialisasi dan kesalahan terminasi

Menurut Pressman, pengujian *black box* berfokus kepada persyaratan fungsional perangkat lunak. Pengujian *black box*memungkinkan perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program.[11] Pengujian *black box*bukan merupakan alternatif dari teknik *whitebox*, tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kesalahan-kesalahan pada metode *white box.*

2.3 *Pemodelan* : Object Oriented Programming

Object Oriented Programming (OOP) adalah suatu metode pemrograman yang berorientasi kepada objek. Tujuan dari OOP diciptakan adalah untuk mempermudah pengembangan program dengan cara mengikuti model yang telah ada di kehidupan sehari-hari. Jadi, setiap bagian dari suatu permasalahan adalah objek, objek itu sendiri merupakan gabungan dari beberapa objek yang lebih kecil lagi.Konsep dari OOP adalah lebih dari sekedar sebuahkonsep pemrograman, OOP adalah cara berpikir tentang aplikasiyang mempelajari untuk berpikir bahwa aplikasi bukan sekedar prosedur melainkan sebagai objek dan *real entity*.[12]

2.4 Unified Modelling Language

UML merupakan singkatan dari "*Unified Modelling Language*" yaitu suatu metode permodelan secara visual untuk sarana perancangan sistem berorientasi objek, atau definisi UML yaitu sebagai suatu bahasa yang sudah menjadi standar

pada visualisasi, perancangan dan juga pendokumentasian sistem software. Saat ini UML sudah menjadi bahasa standar dalam penulisan *blue print software*. [13]

UML Diagram bisa diibaratkan seperti cetakan biru untuk membangun sebuah rumah. Satu set cetakan biru biasanya membantu pembangunnya dengan gambaran yang jelas untuk saluran air, listrik, pemanas, dan sejenisnya, setiap UML Diagram membantu tim developer program dengan gambaran yang jelas untuk sistem tersebut.

1) *Use Case Diagram*

Use case adalah kegiatan atau urutan interaksi yang saling berkaitan antara sistem dan aktor. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara user sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. *Use case* juga digunakan untuk membentuk perilaku (*behaviour*) sistem yang akan dibuat. Sebuah *use case* menggambarkan sebuah interkasi antara pengguna (aktor) dengan sistem yang ada.

2.4.1 Komponen Use Case Diagram

a. Aktor

Aktor merupakan hal terpenting dari *use case diagram*, akan tetapi aktor bukanlah bagian dari *use case diagram*, namun untuk dapat terciptanya suatu *use case diagram* diperlukan beberapa aktor. Itulah pengertian Actor pada komponen *use case diagram* secara singkat .

b. *Use Case*

Use case merupakan sebuah gambaran fungsionalitas dari suatu sistem, sehingga pengguna sistem tersebut paham secara jelas mengenai kegunaan sistem atau aplikasi yang akan dibangun. Itulah pengertian *use case* pada komponen *use case diagram* secara singkat.

c. Relasi

Use case diagram memiliki relasi sebagai berikut:

1) *Association* : Menghubungkan link antar element.
2) *Generalization* : Sebuah elemen dapat merupakan spesialis dari elemen lainnya bisa disebut dengan pewarisan sifat (*inheritance*).
3) *Dependency* : Sebuah elemen bergantung dalam beberapa cara ke element lain.
4) *Aggregation* : Bentuk association dimana sebuah elemen berisi elemen lainnya.

Tipe Relasi/ Streotype yang mungkin terjadi pada *use case* diagram :

1) <<*include*>>, kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi.
2) <<*extends*>>, kelakuan yang hanya berjalan pada kondisi tertentu.
3) <<*communicates*>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah *communicates association*.

d. *System Boundary*

Digambarkan dengan kotak disekitar *use case,* untuk menggambarkan jangkauan sistem anda (*scope of of your system*). Biasanya digunakan apabila memberikan beberapa alternative system yang dapat dijadikan pilihan. *System boundary boxes* dalam penggunaannya optional.

2) Skenario *Use Case*

Skenario *use case* digunakan untuk memudahkan dalam menganalisa skenario yang akan kita gunakan pada fase-fase selanjutnya dengan melakukan penilaian terhadap skenario tersebut.

3) *Activity Diagram*

Activity Diagram adalah diagram yang menggambarkan worlflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Komponen-komponen yang ada pada activity diagram adalah sebagai berikut:

a. Nodes (initial dan final) adalah simbol untuk memulai (initial) dan mengakhiri (final) suatu activity diagram.
b. Activity (aktifitas) adalah proses komputasi yang bisa berupa kata kerja atau ekspresi dan bersifat atomiak atau todak dapat didekomposisi
c. Flow adalah awal dari proses yang paralel dan mampu menggambarkan aktivitas yang mungkin terjadi secara concurrent.
d. Join adalah akhir dari suatu proses paralel.

e. Decision adalah pilihan untuk mengambil keputusan.

f. Partition digunakan untuk menjelaskan siapa yang melakukan aktivitas dalam activity diagram. Untuk melakukan partisi dapat dilakukan dengan menggunakan *Swim Lane*.

g. Signal adalah tanda untuk memulai sebuah aktivitas.

4) *Sequence Diagram*

Sequence diagram adalah diagram yang menggambarkan kelakuan objek pada usecase dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Banyaknya diagram sekuen yang harus digambar sebanyak usecase yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak usecase yang ada maka diagram sekuen yang dibuat semakin banyak.

Komponen *Sequence Diagram :*

a. Aktor :Menggambarkan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem.

b. *Boundary :*Mengambarkan interaksi antara satu atau lebih aktor dengan sistem, memodelkan bagian dari sistem yang bergantung pada pihak lain disekitarnya dan merupakan pembatas sistem dengan dunia luar.

c. *Control :*Menggambarkan,mengkoordinasikan perilaku sistem dan dinamika dari suatu sistem, menangani tugas utama dan mengontrol alur kerja suatu

sistem.

d. *Entity :*Menggambarkan informasi yang harus disimpan oleh sistem (struktur data dari sebuah sistem).

e. *Object Message :*Menggambarkan pesan/hubungan antar obyek yang menunjukkan urutan kejadian yang terjadi.

f. *Message to Self :*Mengambarkan pesan/hubungan obyek itu sendiri, yang menunjukkan urutan kejadian yang terjadi.

g. *Return Message :*Menggambarkan pesan/hubungan antar obyek, yang menunjukan urutan kejadian yang terjadi.

h. *Lifeline :*Eksekusi obyek selama *sequence* (*message* dikirim atau diterima dan aktifasinya).

5) *Class Diagram*

Class Diagram adalah diagram yang menunjukan class-class yang ada dari sebuah sistem dan hubungannya secara logika. *Class diagram* menggambarkan struktur statis dari sebuah sistem. Karena itu class diagram merupakan tulang punggung atau kekuatan dasar dari hampir setiap metode berorientasi objek termasuk UML. [14]

Menurut Rosa A.S dan M. Shalahuddin,*class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. [15]

Komponen*Class Diagram :*

a. *Class* :

Blok - blok pembangun pada pemrograman berorientasi obyek. Sebuah *class* digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama dari *class*. Bagian tengah mendefinisikan *property*/atribut *class*. Bagian akhir mendefinisikan methodmethod dari sebuah *class*.

b. Association :

Sebuah asosiasi merupakan sebuah relationship paling umum antara 2 class dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 class. Garis ini bisa melambangkan tipe-tipe relationship dan juga dapat menampilkan hokum - hukum multiplisitas pada sebuah *relationship*.(Contoh:One-to-one, one-to-many, many-to-many).

c. Composition :

Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka class tersebut memiliki relasi *composition* terhadap *class* tempat dia bergantung tersebut. Sebuah *relationship composition* digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.

d. Dependency :

Kadangkala sebuah *class* menggunakan class yang lain. Hal ini disebut *dependency*. Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu *class* yang menggunakan *class* yang lain.

e. Aggregation :

Mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi.

6) *Object Diagram*

Diagram Objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. dalam diagram objek harus di pastikan bahwa semua kelas yang sudah didefinisikan pada diagram kelas harus dipakai objeknya, karena jika tidak pendefinisian kelas itu tidak dapat dipertanggung jawabkan.

Sebuah diagram objek merupakan gambaran objek-objek pada sebuah sistem pada satu titik waktu. karena lebih menonjolkan printah-printah dari pada kelas daigram objek sering disebut juga sebagai diagram perintah. elemen-elemen sebuah diagram objek adalah spesifikasi perintah.

Elemen-elemen diagram objek terdiri dari:

a. Objek merupakan Objek dari *class* yang berjalan saat sistem dijalankan
*b. Link*merupakan relasi antar objek

Terdapat dua *stereotype* terhadap *dependency* di antara objek, yaitu:

a. *Become,* menspesifikasikan target adalah objek yang sama dengan sumber, tapi pada titik waktu berikutnya dan dapat mempunyai nilai, *state* dan peran yang berbeda.
b. *Call,* menspesifikasikan operasi sumber memanggil operasi target.
c. *Copy*, menspesifikasikan objek target adalah nyata, tapi independen, mengkopi sumber.

7) *Deployment Diagram*

Deployment diagram merupakan gambaran proses-proses berbeda pada suatu sistem yang berjalan dan bagaimana relasi di dalamnya. Hal inilah yang mempermudah user dalam pemakaian sistem yang telah dibuat dan diagram tersebut merupakan diagram yang statis. Misalnya untuk mendeskripsikan sebuah situs *web*, *deployment diagram* menunjukkan komponen perangkat keras ("node") apa yang digunakan (misalnya, *web* server, server aplikasi, dan *database* server), komponen perangkat lunak ("artefak") apa yang berjalan pada setiap node (misalnya, aplikasi *web*, *database*), dan bagaimana bagian-bagian yang berbeda terhubung (misalnya JDBC, REST, RMI).

Node digambarkan sebagai kotak, dan artefak yang dialokasikan ke setiap node digambarkan sebagai persegi panjang di dalam kotak. Node mungkin memiliki subnodes, yang digambarkan sebagai kotak *nested*. Sebuah node tunggal secara konseptual dapat mewakili banyak node fisik, seperti sekelompok database server.

Komponen *Deployment Diagram :*

a. *Component :*Pada *deployment diagram*, komponen-komponen yang ada diletakkan didalam node untuk memastikan keberadaan posisi mereka.

b. *Node :Node* menggambarkan bagian-bagian*hardware* dalam sebuah sistem. Notasi untuk node digambarkan sebagai sebuah kubus 3 dimensi.

c. *Association :*Sebuah *association* digambarkan sebagai sebuah garis yang menghhubungkan dua node yang mengindikasikan jalur komunikasi antara komponen - komponen *hardware*.

d. *Dependency :*Merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain.

e. *Generalization :*Menunjukan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization*, *class* yang lebih spesifik (*subclass*) akan menurunkan atribut dan operasi dari *class* yang lebih umum (*superclass*).

f. *Note :*Menunjukan catatan untuk komentar dari suatu pesan antar elemen, sehingga bisa langsung terlampir dalam model.

g. *Class :Interface* merupakan kumpulan operasi tanpa implementasi dari suatu *class*. Implementasi operasi dalam *interface* dijabarkan oleh operasi didalam *class*.

8) *Component Diagram*

C*omponent diagram* dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada didalam sistem. diagram komponen juga dapat digunakan untuk memodelkan hal-hal berikut :

a. *Source code* program perangkat lunak.

b. Komponen *executable* yang di lepas ke *user.*

c. Basis data secara fisik.

d. Sistem yang harus beradaptasi dengan sistem lain.

e. *Framework* sistem, *framework* pada perangkat lunak merupakan kerangka kerja yang dibuat untuk memudahkan pengembangan dan pemeliharaan

aplikasi, contohnya seperti struts dari apache yang menggunakan prinsip desain *Model-View-Controller* (MVC) dimana *soucre code* program dikelompokkean berdasarkan fungsinya. dimana *controller* berisi *source code* yang menangani request dan validasi, model berisi *source code* yang menangani manipulasi data dan *bussiness logic*, dan *view* berisi *source code* yang menangani tampilan.

Komponen dasar yang biasanya ada dalam suatu sistem adalah sebagai berikut :

a. Komponen *user interface* yang menangani tampilan.
b. Komponen *bussiness process* yang menangani fungsi-fungsi proses bisnis.
c. Komponen data yang menangani manipulasi data.
d. Komponen *security* yang menangani keamanan sistem.

2.4.2 Tujuan Penggunaan *UML*

Tujuan atau fungsi dari penggunaan UML antara lain :

1. Dapat memberikan bahasa permodelan visual kepada pengguna dari berbagai macam pemerograman maupun proses rekayasa.
2. Dapat menyatukan praktek-praktek terbaik yang ada dalam permodelan.
3. Dapat memberikan model yang siap untuk digunakan, merupakan bahasa permodelan visual yang ekspresif untuk mengembangkan sistem dan untuk saling menukar model secara mudah.
4. Dapat berguna sebagai *blue print*, sebab sangat lengkap dan detail dalam perancangannya yang nantinya akan diketahui informasi yang detail mengenai koding suatu program.

5. Dapat memodelkan sistem yang berkonsep berorientasi objek, jadi tidak hanya digunakan untuk memodelkan perangkat lunak (*software*) saja.
6. Dapat menciptakan suatu bahasa permodelan yang nantinya dapat dipergunakan oleh manusia maupun oleh mesin.

2.5 Perangkat Lunak Pendukung

Penelitian ini menggunakan beberapa perangkat lunak untuk membangun Sistem Informasi *Inventory* dan Pemesanan Buku pada Toko Buku Waroeng Pendidikan. Berikut adalah perangkat lunak yang digunakan seperti :

2.5.1 Perl Hypertext Preprocessor (PHP)

Menurut Arief, PHP (*Perl Hypertext Preprocessor*) adalah bahasa *server-side-scripting* yang menyatu dengan HTML untuk membuat halaman *web* yang dinamis. [16] Dengan menggunakan program PHP, sebuah *web*site akan lebih interaktif dan dinamis.

Adapun kelebihan-kelebihan dari PHP yaitu:

1. PHP merupakan sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaannya. Tidak seperti halnya bahasa pemrograman aplikasi yang lainnya.
2. PHP dapat berjalan pada *web* server yang dirilis oleh Microsoft, seperti IIS atau PWS juga pada apache yang bersifat open source.
3. Karena sifatnya yang open source, maka perubahan dan perkembangan interpreter pada PHP lebih cepat dan mudah, karena banyak milis-milis dan developer yang siap membantu pengembanganya.

4. Jika dilihat dari segi pemahaman, PHP memiliki referensi yang begitu banyak sehingga sangat mudah untuk dipahami.
5. PHP dapat berjalan pada 3 operating sistem, yaitu: Linux, unux, dan windows, dan juga dapat dijalankan secara runtime pada suatu console.

2.5.2 Bootstrap

Bootstrap adalah sebuah framework CSS yang menyediakan kumpulan komponen-komponen antarmuka dasar pada *web* yang telah dirancang sedemikian rupa untuk digunakan bersama-sama. Selain komponen antarmuka, Bootstrap juga menyediakan sarana untukmembangunlayout halaman dengan mudah dan rapi, serta modifikasi pada tampilan dasar HTML untuk membuat seluruh halaman *web* yang dikembangkan senada dengan komponen-komponen lainnya.Bootstrap dibuat untuk memberikan sekumpulan perangkat yang dapat digunakan untuk membangun *web*site sederhana dengan mudah.

2.5.3 Cascading Style Sheet (CSS)

CSS merupakan salah satu kode pemrograman yang bertujuan untuk menghias dan mengatur gaya tampilan/layout halaman *web* supaya lebih elegan dan menarik.CSS adalah sebuah teknologi *internet* yang direkomendasikan oleh World Wide *Web* Consortium atau W3C pada tahun 1996. Awalnya, CSS dikembangkan di SGML pada tahun 1970, dan terus dikembangkan hingga saat ini. CSS telah mendukung banyak bahasa markup seperti HTML, XHTML, XML, SVG (Scalable Vector Graphics) dan Mozilla XUL (XML User Interface Language).

Pada desember 1996, W3C memperkenalkan Level 1 spesifikasi CSS atau juga dikenal CSS1 yang mendukung format, warna font teks, dan lain-lain. Kemudian, Mei 1998, W3C menerbitkan CSS2 yang di dalamnya diatur fungsi peletakan elemen. Dan sekarang, W3C telah memperbaiki dan meningkatkan Kemampuan CSS2 ke CSS3.

CSS digunakan oleh *web* programmer dan juga blogger untuk menentukan warna, tata letak font, dan semua aspek lain dari presentasi dokumen di situs mereka. Saat ini, hampir tidak ada situs *web* yang dibangun tanpa kode css.

2.5.4 XAMPP

Menurut Bunafit Nugroho (2008, h. 2), XAMPP merupakan paket php berbasis open source yang dikembangkan oleh sebuah komunitas *open source*. Dengan menggunakan XAMPP, tidak usah lagi bingung untuk melakukan penginstalan program lain, karena semua kebutuhan telah disediakan oleh XAMPP.

XAMPP adalah perangkat lunak bebas mendukung banyak sistem operasi yang merupakan kompilasi dari beberapa perangkat lunak. XAMPP dikembangkan oleh sebuah tim proyek bernama *Apache Friends*.

Fungsinya adalah 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. Nama XAMPP merupakan singkatan dari X berarti mendukung 4 sistem operasi, Apache, MySQL, PHP dan Perl.

2.5.5 Database Management System (DBMS)

DBMS adalah singkatan dari "*Database Management System*" yaitu sistem pengorganisasian dan sistem pengolahan *database* pada komputer. DBMS atau *database management system* ini merupakan perangkat lunak (*software*) yang dipakai untuk membangun basis data yang berbasis komputerisasi.

DBMS (*Database Management system*) ini juga dapat membantu dalam memelihara serta pengolahan data dalam jumlah yang besar, dengan menggunakan DBMS bertujuan agar tidak dapat menimbulkan kekacauan dan dapat dipakai oleh *user* sesuai dengan kebutuhan.

DBMS ialah perantara untuk user dengan basis data, untuk dapat berinteraksi dengan DBMS dapat memakai bahasa basis data yang sudah di tentukan oleh perusahaan DBMS. Bahasa basis data umumnya terdiri dari berbagai macam instruksi yang diformulasikan sehingga instruksi tersebut dapat di proses oleh DBMS.

Perintah atau instruksi tersebut umumnya ditentukan oleh *user*, adapun bahasa yang digunakan dibagi kedalam 2 (dua) macam diantaranya sebagaimana di bawah ini:

1) DDL (*Data Definition Language*)

Yang pertama adalah bahasa DDL atau kepanjangannya *Data Definition Languange*, yaitu dipakai untuk menggambarkan desain dari basis data secara menyeluruh. DDL (*Data Definition Language*) dapat dipakai untuk membuat tabel baru, memuat indeks, maupun mengubah tabel. Hasil dari kompilasi DDL akan disimpan di kamus data. Itulah definisi dari DDL.

2) DML (*Data Manipulation Language*)

DML atau kepanjangannya *Data Manipulation Language*, yaitu dipakai untuk memanipulasi daan pengambilan data pada suatu basis data, misalnya seperti penambahan data yang baru ke dalam suatu basis data, menghapus data pada suatu basis data dan mengubah data pada suatu basis data. Itulah definisi dar DML.

2.5.6 MySQL

MySQL merupakan DBMS yang bersifat *open source* dimana pengguna dapat melakukan modifikasi pada kodenya yang berorientasi pada performa dan merupakan DBMS yang mendukung beberapa tipe *table* (*storage engine*). Tipe *table* yang digunakan secara *default* adalah *MyISAM* yang memiliki kelebihan akses data yang cepat tetapi tidak mendukung fitur *foreign key*. Jika pengguna ingin ada dukungan foreign key pada *table*, pengguna dapat menggunakan tipe *table* InnoDB.

Kelebihannya:

1) *Free*/gratis.
2) Selalu stabil dan cukup tangguh.
3) Keamanan yang cukup baik.
4) Sangat mendukung transaksi, dan dukungan dari banyak komunitas.
5) Sangat fleksisbel dengan barbagai macam program.
6) Perkembangan yang cepat.

Kekurangannya:

1) Kurang mendukung koneksi bahasa pemerograman misalnya seperti Visual Basic (VB), Foxpro, Delphi sebab koneksi ini dapat menyebabkan field

yang dibaca harus sesuai dengan koneksi bari bahasa pemerograman visual tersebut.

2) Data yang dapat ditangani belum besar dan belum mendukung widowing Function.

2.6 Arsitektur Aplikasi

Arsitektur aplikasi berisi penjelasan tentang jaringan komputer, jenis-jenis jaringan komputer, topologi jaringan komputer, dan manfaat jaringan Komputer.

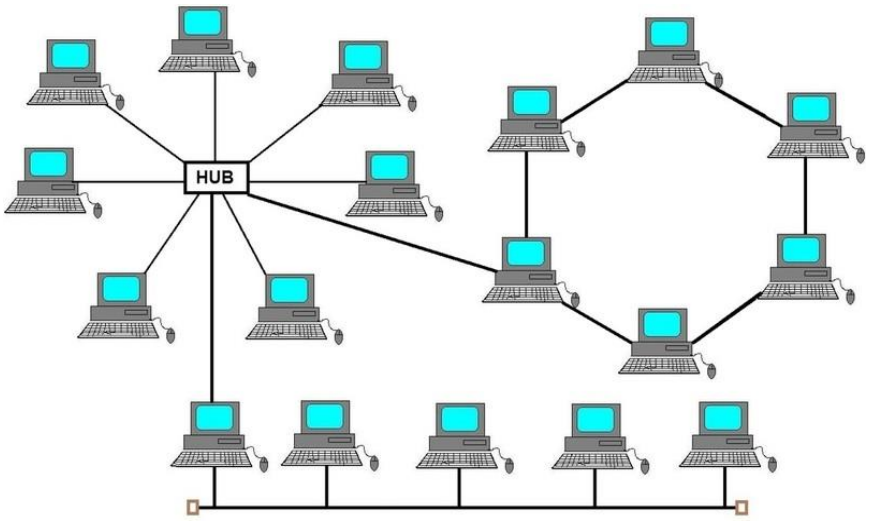
2.6.1 Jaringan Komputer

Jaringan komputer adalah suatu sistem yang didalamnya terdiri dari dua atau lebih perangkat komputer serta perangkat - perangkat lainnya yang dibuat atau dirancang untuk dapat berkerja sama dengan tujuan agar dapat berkomunikasi, mengakses informasi, meminta serta memberikan layanan atau *service* antara komputer satu dengan yang lainnya.

Menurut **Jafar Noor Yudianto,** jaringan komputer adalah sebuah sistem yang terdiri atas komputer-komputer yang didesain untuk dapat berbagi sumber daya (*printer*, CPU), berkomunikasi (surel, pesan instan), dan dapat mengakses informasi (peramban *web*). Tujuan dari jaringan komputer adalah agar dapat mencapai tujuannya, setiap bagian dari jaringan komputer dapat meminta dan memberikan layanan (*service*). Pihak yang meminta/menerima layanan disebut klien (*client*) dan yang memberikan/mengirim layanan disebut peladen (*server*). Desain ini disebut dengan sistem *client-server*, dan digunakan pada hampir seluruh aplikasi jaringan computer. [22]

Topologi jaringan adalah susunan fisik bagaimana node-node saling dihubungkan. Ada empat topologi yang biasa digunakan dalam membangun sebuah jaringan, yaitu topologi *bus*, topologi *ring*, topologi s*tar* atau *hub*, dan topologi *hybrid*. Adapun topologi yang digunakan oleh penulis dalam perancangan sistem informasi *Inventory* dan pemesnan buku pada topo buku Waroeng Pendidikan adalah topologi *hybird*. Topologi *Hybrid* adalah salah satu bentuk topologi jaringan fisik yang sudah sangat umum digunakan disamping jenis komputer lainnya seperti jaringan *point-to-point*, topologi *bus*, topologi *ring*, dll. Klasifikasi topologi dibuat berdasarkan hubungan antara node berbeda dalam jaringan. Pilihan untuk menggunakan topologi tertentu tergantung pada berbagai faktor. Masing-masing topologi ini memiliki kelebihan dan kekurangan.

Topologi *Hybrid* merupakan salah satu topologi jaringan komputer yang populer karena kelebihan Topologi *Hybrid* memiliki nilai plus dibanding topologi lainnya dan sudah terbukti menjadi salah satu bentuk terbaik untuk kebutuhan jaringan komputer skala besar. Namunpun demikian tidak berarti topologi *hybrid* tidak memiliki kekurangan.



Gambar 2. 1 Topologi Jaringan Hybird