# BAB II

# LANDASAN TEORI

## 2.1. Tinjauan Pustaka

### A. Konsep Dasar Sistem Informasi

#### 1. Pengertian Sistem

Kadir (2014:61), sistem adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Sebagai gambaran jika dalam sebuah sistem terdapat sebuah elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama maka elemen tersebut dapat dipastikan bukanlah bagian dari sistem. Ada 3 elemen yang membentuk sebuah sistem yaitu :

a. *Input*
   Segala sesuatu yang masuk ke dalam sistem dan selanjutnya menjadi bahan untuk di proses.
b. Proses
   Bagian yang melakukan perubahan dari *input* menjadi *output* yang berguna, misalnya berupa informasi dan produk, tetapi juga bisa berupa hal-hal yang tidak berguna, misalnya sisa pembuangan atau limbah.
c. *Output*
   Hasil dari pemrosesan, misalnya berupa suatu informasi, saran, dan cetakan laporan.

#### 2. Pengertian Informasi

Mulyanto (2009:12) menyatakan, informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya,

sedangkan data merupakan sumber informasi yang menggambarkan suatu kejadian yang nyata.

### 3. Pengertian Sistem Informasi

Sistem informasi merupakan bagian penting dari sebuah kegiatan bisnis, karena sistem informasi merupakan komponen penting yang mampu meningkatkan efisiensi dan efektifitas proses bisnis yang berjalan mulai dari manajerial, pengambilan keputusan, dan *workgroup colaboration*.

O'Brien dan Marakas (2010:4) berpendapat bahwa, sistem informasi merupakan kombinasi yang terorganisir antara pengguna, perangkat keras, perangkat lunak, jaringan komunikasi, sumber daya data kebijakan prosedur yang menyimpan, mengambil, mengubah, menyebarkan informasi dalam sebuah organisasi.

## B. Metodologi Penelitian

### 1. Metodologi Pengumpulan Data

a. Studi Pustaka

Studi kepustakaan berkaitan dengan kajian teoritis dan referensi lain yang berkaitan dengan nilai, budaya dan norma yang berkembang pada situasi sosial yang diteliti, selain itu studi kepustakaan sangat penting dalam melakukan penelitian, hal ini dikarenakan penelitian tidak akan lepas dari literatur-literatur Ilmiah (Sugiyono, 2012:291).

b. Observasi

Observasi merupakan teknik pengumpulan data yang mempunyai ciri yang spesifik bila dibandingkan dengan teknik yang lain yaitu wawancara dan kuesioner. Karena observasi tidak selalu dengan objek manusia tetapi juga objek-objek alam yang lain. Sugiyono (2012:145)

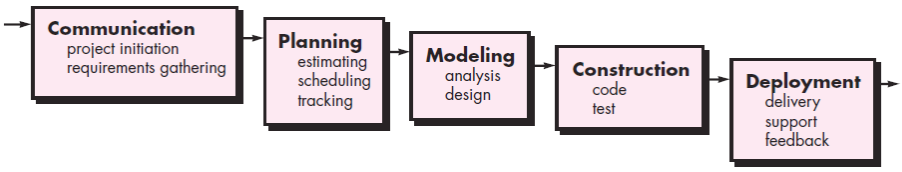
mengemukakan bahwa, observasi merupakan suatu proses yang kompleks, suatu proses yang tersusun dari berbagai proses biologis dan psikologis. Dua diantara yang terpenting adalah proses-proses pengamatan dan ingatan.

c. Wawancara

Sugiyono (2012:194) berpendapat, wawancara digunakan sebagai teknik pengumpulan data, apabila peneliti ingin melakukan studi pendahuluan untuk menemukan permasalahan yang harus diteliti, dan juga apabila peneliti ingin mengetahui hal-hal dari responden yang lebih mendalam dan jumlah respondennya sedikit. Teknik pengumpulan data ini mendasarkan diri pada laporan tentang diri sendiri atau *self-report*, atau setidak-tidaknya pada pengetahuan atau keyakinan pribadi. Wawancara tersebut dapat dilakukan secara terstruktur maupun tidak terstruktur, dan dapat dilakukan melalui tatap muka maupun dengan menggunakan telepon.

## 2. Metodologi Pengembangan Sistem

Pressman (2010:39) menjelaskan bahwa, *Waterfall model* disebut juga siklus hidup klasik, adalah paradigma tertua untuk rekayasa perangkat lunak yang menyarankan pendekatan sistematis, sekuensial untuk pengembangan perangkat lunak yang diawali dengan persyaratan spesifikasi pelanggan dan berkembang melalui perencanaan, pemodelan, konstruksi, dan penyebaran,yang berpuncak pada dukungan yang berkelanjutan dari perangkat lunak yang telah selesai.




Sumber : Pressman (2010:39)

**Gambar II.1.**
**Contoh *Waterfall* Pressman**

a. *Communication* (*Project Initiation* dan *Requirements Gathering*)
Sebelum memulai pekerjaan yang bersifat teknis, sangat diperlukan adanya komunikasi dengan *customer* demi memahami dan mencapai tujuan yang ingin dicapai. Hasil dari komunikasi tersebut adalah inisialisasi proyek, seperti menganalisis permasalahan yang dihadapi dan mengumpulkan data-data yang diperlukan, serta membantu mendefinisikan fitur dan fungsi *software*. Pengumpulan data-data tambahan bisa juga diambil dari jurnal, artikel, dan internet.

b. *Planning* (*Estimating, Scheduling, Tracking*)
Tahap berikutnya adalah tahapan perencanaan yang menjelaskan tentang estimasi tugas-tugas teknis yang akan dilakukan, resiko yang dapat terjadi, sumber daya yang diperlukan dalam membuat sistem, produk kerja yang ingin dihasilkan, penjadwalan kerja yang akan dilaksanakan, dan *tracking* proses pengerjaan sistem.

c. *Modeling* (*Analysis* dan *Design*)
Tahapan ini adalah tahap perancangan dan permodelan arsitektur sistem yang berfokus pada perancangan struktur data, arsitektur *software*, tampilan *interface*, dan algoritma program. Tujuannya untuk lebih memahami gambaran besar dari apa yang akan dikerjakan.

d. *Construction* (*Code* dan *Test*)
Tahapan *Construction* ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk bahasa yang dapat dibaca oleh mesin. Setelah

pengkodean selesai, dilakukan pengujian terhadap sistem dan juga kode yang sudah dibuat. Tujuannya untuk menemukan kesalahan yang mungkin terjadi untuk nantinya diperbaiki.

e. *Deployment* (*Delivery, Support, Feedback*)

Tahapan *Deployment* merupakan tahapan implementasi *software* ke *customer*, pemeliharaan *software* secara berkala, perbaikan *software*, evaluasi *software*, dan pengembangan *software* berdasarkan umpan balik yang diberikan agar sistem dapat tetap berjalan dan berkembang sesuai dengan fungsinya.

## *C. Unified Modelling Language*

*Unified Modeling Language* (UML) adalah bahasa standar untuk menulis denah perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menentukan, membangun, dan mendokumentasikan artefak dari sistem perangkat lunak. Dengan kata lain, seperti arsitek bangunan membuat denah yang akan digunakan oleh sebuah perusahaan konstruksi, arsitek *software* membuat diagram UML untuk membantu pengembang perangkat lunak membangun perangkat lunak. Jika anda memahami kosakata UML, anda dapat lebih mudah memahami dan menentukan sistem dan menjelaskan desain sistem kepada orang lain (Pressman, 2010:841).

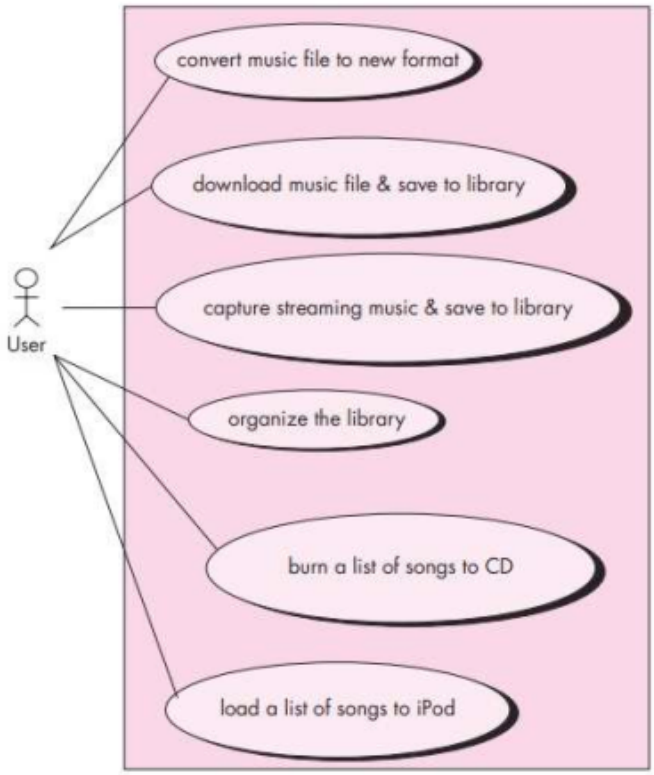
### *1. Use Case Diagram*

Sebuah UML *use case diagram* adalah gambaran dari semua kasus penggunaan dan bagaimana mereka berhubungan. Ini memberikan gambaran besar dari fungsi sistem. Sebagai contoh, aplikasi mesin penjual otomatis

mungkin memiliki tiga aktor yang mewakili pelanggan, teknisi perbaikan, dan vendor yang mengisi ulang mesin. Dalam diagram *use case*, kasus penggunaan ditampilkan sebagai oval.

Perhatikan bahwa tidak ada rincian kasus penggunaan termasuk di dalam diagram dan sebagai gantinya perlu disimpan secara terpisah. Perhatikan juga bahwa kasus penggunaan ditempatkan dalam persegi panjang tapi aktor tidak. Persegi panjang ini adalah pengingat visual dari batas-batas sistem dan bahwa aktor berada di luar sistem. Beberapa kasus penggunaan dalam sistem yang mungkin berkaitan satu sama lain. Misalnya, ada beberapa langkah yang sama dalam "membakar" daftar lagu ke CD dan memuat daftar lagu ke *iPod*.

Dalam kedua kasus, pengguna pertama menciptakan daftar kosong dan kemudian menambahkan lagu dari perpustakaan ke dalam daftar. Untuk menghindari duplikasi dalam kasus penggunaan, biasanya lebih baik untuk membuat kasus penggunaan baru yang mewakili kegiatan yang digandakan, dan kemudian membiarkan kegunaan lain kasus ini termasuk kasus penggunaan baru sebagai salah satu langkah mereka (Pressman, 2010:847 848).




Sumber : Pressman (2010:848)

**Gambar II.2.**
**Contoh *Use Case Diagram***

**2. *Activity Diagram***

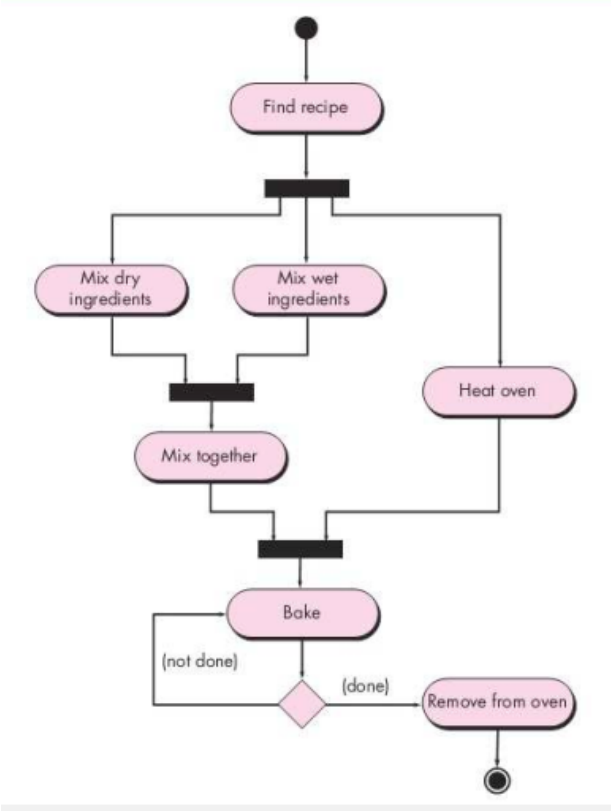
Sebuah diagram aktivitas UML menggambarkan perilaku dinamis dari suatu sistem atau bagian dari sistem melalui aliran kontrol antara aksi yang dilakukan sistem. Sebuah diagram aktivitas UML menggambarkan

perilaku dinamis dari suatu sistem atau bagian dari sistem melalui aliran kontrol antara aksi bahwa sistem melakukan.

Komponen utama dari suatu diagram aktivitas adalah *node* aksi, diwakili oleh persegi panjang bulat, yang sesuai dengan tugas yang dilakukan oleh sistem perangkat lunak. Panah dari satu *node* aksi lainnya menunjukkan aliran kontrol. Artinya, panah antara dua *node* aksi berarti bahwa setelah aksi pertama selesai aksi kedua dimulai. Sebuah titik hitam pekat membentuk *node* awal yang menunjukkan titik awal kegiatan. Sebuah titik hitam yang dikelilingi oleh lingkaran hitam adalah *node* akhir yang menunjukkan akhir kegiatan.

*Fork* merupakan pemisahan kegiatan menjadi dua atau lebih kegiatan bersamaan. Hal ini digambarkan sebagai persegi panjang hitam horizontal dengan satu panah menunjuk kepadanya dan dua atau lebih anak panah menunjuk darinya. Setiap panah keluar mewakili aliran kontrol yang dapat dijalankan bersamaan dengan arus yang sesuai dengan panah keluar lainnya. Kegiatan bersamaan ini dapat dilakukan pada komputer dengan menggunakan benang yang berbeda atau bahkan menggunakan komputer yang berbeda.

Sumber : Pressman (2010:850)

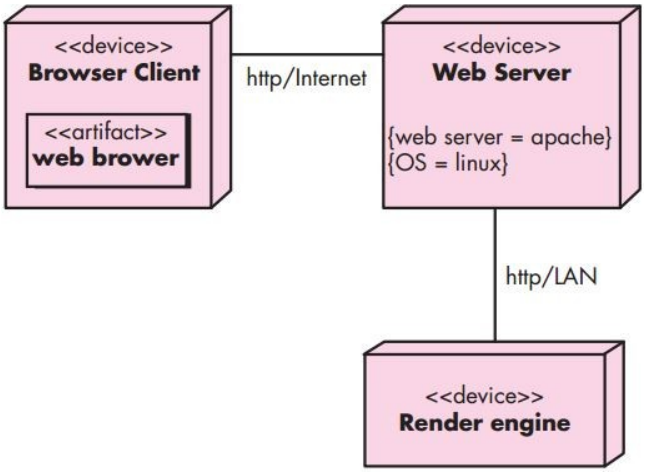



**Gambar II.3.**
**Contoh *Activity Diagram***

Sebuah *node* keputusan berkaitan dengan suatu cabang di aliran kontrol berdasarkan kondisi. Seperti *node* ditampilkan sebagai segitiga putih dengan panah masuk dan dua atau lebih anak panah keluar. Setiap panah keluar diberi label dengan penjaga (suatu kondisi di dalam tanda kurung siku). Aliran kontrol mengikuti panah keluar penjaga yang benar. Dianjurkan untuk memastikan bahwa kondisi mencakup semua kemungkinan sehingga tepat satu dari mereka benar setiap kali *node* keputusan tercapai (Pressman, 2010:853-854).

### *3. Deployment Diagram*

Pressman (2010:846) menyatakan bahwa, *deployment diagram* adalah sebuah diagram UML yang berfokus pada struktur sistem perangkat lunak dan berguna untuk menjukkan fisik sistem perangkat lunak tersebut berada.

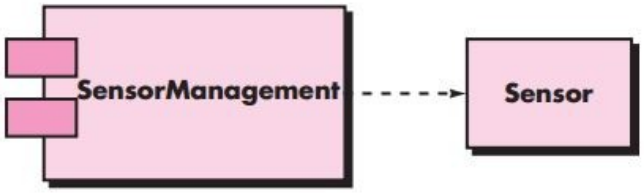



Sumber : Pressman (2010:846)

**Gambar II.4.**
**Contoh *Deployment Diagram***

**4. *Component Diagram***

Pressman (2010:237) menjelaskan, *component diagram* adalah diagram yang menunjukkan secara fisik komponen perangkat lunak pada sistem dan hubungannya antar mereka. *Component Diagram* merupakan bagian dari sistem yang diuraikan menjadi subsistem atau modul yang lebih kecil.




Sumber : Pressman (2010:237)

**Gambar II.5.**
**Contoh *Component Diagram***

Untuk mendukung dalam pembuatan sistem informasi *inventory* berbasis *web* pada PT. Livaza Tekonologi Indonesia, penulis menggunakan berbagai teknologi dan *tools*, yaitu sebagai berikut :

**1. MySQL**

MySQL adalah salah satu jenis *database* yang banyak digunakan untuk membuat aplikasi berbasis *web* yang dinamis. MySQL termasuk jenis RDBMS (*Relational Database Management Sistem*). MySQL ini mendukung Bahasa pemrograman PHP. MySQL juga mempunyai *query* atau bahasa SQL(*Structured Query Language*) yang *simple* dan menggunakan *escape character* yang sama dengan PHP (Kurniawan, 2010:16).

MySQL adalah sebuah program *open source*. *Open source* berarti bahwa memungkinkan bagi siapa saja untuk menggunakan dan memodifikasi *software* tersebut. *Source code* pada MySQL dapat dipelajari dan diubah sesuai dengan kebutuhan pemakainya. Perangkat lunak MySQL menggunakan GPL (*General Public License*) untuk menentukan apa yang boleh dan tidak boleh dilakukan dengan perangkat lunak dalam situasi yang berbeda.

**2. *Entity Relationship Diagram* (ERD)**

Connolly dan Begg (2015:405) menyatakan bahwa, *entity relationship* adalah model yang dapat digunakan untuk memberikan pengertian mengenai data yang akan digunakan oleh suatu perusahaan. Dalam perancangan basis data, *entity relationship* adalah pendekatan *top-down* dimana perancangan dimulai dengan mengidentifikasi data penting yang disebut entitas dan hubungan antara data yang harus dipresentasikan ke dalam model.

### *3. Database*

*Database* adalah sebuah tempat penyimpanan yang besar dimana terdapat kumpulan data yang tidak hanya berisi data operasional tetapi juga deskripsi data. Seperti yang disampaikan oleh Connolly dan Begg (2015:63), bahwa *database* adalah kumpulan data yang saling terhubung secara logis dan deskripsi dari data tersebut, dirancang untuk menemukan informasi yang dibutuhkan oleh sebuah organisasi. Dalam merancang *database*, salah satu hal yang perlu diperhatikan adalah efisiensi.

Banyaknya data yang redudansi dapat mengurangai efisiensi pada *database* sehingga perlu dilakukan normalisasi. *Database* ini digunakan tidak hanya oleh satu orang maupun satu departemen, *database* dapat digunakan oleh seluruh departemen dalam perusahaan. *Database* ini akan menjadi sumber data yang digunakan secara bersama dalam perusahaan.

### *4. Web Server*

Sibero (2011:11) menjelaskan, *web server* adalah sebuah komputer yang terdiri dari perangkat keras dan perangkat lunak. Secara bentuk fisik dan cara kerjanya, perangkat keras *web server* dengan PC dibedakan oleh kapasitas dan kapabilitasnya. Perangkat lunak dalam *web server* memiliki karakteristik dan teknologi yang digunakan untuk mengatur kerja sistemnya. Contoh-contoh perangkat lunak *web server* antara lain, *Apache*, *Nginx*, dan *Internet Information Services*.

**5. HTTP *(HyperText Transfer Protocol)***

Connolly dan Begg (2015:1053) menjelasakan bahwa, HTTP mendefinisikan bagaimana klien dan *server* berkomunikasi. HTTP berorientasi objek, dan merupakan *stateless protocol* untuk melakukan pertukaran informasi antara klien dan *server*.

HTTP berbasis pada paradigma *request-response*. Transaksi HTTP terdiri dari beberapa langkah berikut :

1. *Connection* : *client* menciptakan koneksi ke *web server*
2. *Request* : *client* mengirimkan permintaan pesan ke *web server*
3. *Response* : *web server* mengirimkan respon
4. *Close* : koneksi ditutup oleh *web server*

**6. HTML *(Hypertext Markup Language)***

HTML (*Hyper Text Mark Up Language*) merupakan bahasa yang digunakan untuk mendeskripsikan struktur sebuah halaman *web*. HTML berfungsi untuk mempublikasi dokumen *online*. *Statement* dasar dari HTML disebut *tags*. Sebuah *tag* dinyatakan dalam sebuah kurung siku (<>). *Tags* yang ditujukan untuk sebuah dokumen atau bagian dari suatu dokumen haruslah dibuat berupa pasangan. Terdiri dari *tag* pembuka dan tag penutup. Dimana *tag* penutup menggunakan tambahan tanda garis miring (/) di awal nama *tag*.

Contohnya <html> merupakan *tag* pembuka dan </html> merupakan *tag* penutupnya. Selain tag dasar terdapat juga tag a ,atau *anchor tag*. Seperti beberapa *tag* HTML lainnya, *tag* A digunakan bersamaan dengan atribut yang dapat menjelaskan lebih spesifik mengenai apa yang akan dikerjakan. *Tag* a biasanya digunakan bersamaan dengan atribut <href> atau *Hypertext Reference* yang berfungsi menghubungkan satu dokumen dengan dokumen lainnya. *Tag*

inilah yang membuat pengguna dapat berpindah-pindah halaman dengan cara memilih *button* tertentu (Henderson, 2009:232).

### 7. CSS

CSS kepanjangan dari *Cascading Style Sheet* adalah bahasa-bahasa yang merepresentasikan halaman *web*. Seperti warna, *layout*, dan *font*. Dengan menggunakan CSS, seorang *web developer* dapat membuat halaman *web* yang dapat beradaptasi dengan berbagai macam ukuran layar. Pembuatan CSS biasanya terpisah dengan halaman HTML. Meskipun CSS dapat disisipkan di dalam halaman HTML. Hal ini ditujukan untuk memudahkan pengaturan halaman HTML yang memiliki rancangan yang sama. Pada saat sebuah *style* diasosiasikan terhadap sebuah elemen, maka format yang terdapat pada *style* tersebut akan secara otomatis diaplikasikan ke setiap elemen yang terasosiasikan tersebut.

Sebagai contoh jika kita ingin merubah format *font* pada *heading* menjadi *italic,* maka kita hanya perlu mengaturnya pada *style* yang terasosiasikan pada elemen *heading*. Dengan begitu setiap elemen heading yang ada akan memiliki format font yang *italic*. Agar dapat berjalan, halaman CSS yang terpisah dengan HTML haruslah diasosiasikan dengan dokumen HTML yang dituju.

Pada saat *browser* melakukan *load* pada halaman HTML, maka secara otomatis *browser* juga akan melakukan load terhadap halaman CSS dan menggunakannya untuk menentukan *display* dari sebuah halaman *web*. *Style* pada CSS diaplikasikan dalam sebuah urutan yang bergantung pada hubungan elemen-elemen yang terasosiasi. Contohnya pada saat kita menentukan format *style* pada

sebuah *div*, maka setiap elemen yang berada dalam div tersebut akan memiliki format *style* yang sama (Henderson, 2009:72).

### 8. PHP

PHP adalah bahasa pemrograman untuk dijalankan melalui halaman *web*, umumnya digunakan untuk mengolah informasi di *internet*. Sedangkan dalam pengertian lain PHP adalah singkatan dari PHP adalah singkatan dari PHP *Hypertext Preprocessor* yaitu bahasa pemrograman *web server-side* yang bersifat *open source* atau gratis. PHP merupakan *script* yang menyatu dengan HTML dan berada pada *server* (Kurniawan, 2010:2).

PHP adalah bahasa pemrograman *script* yang paling banyak dipakai saat ini. PHP merupakan *script* yang digunakan untuk membuat halaman *web* yang dinamis yang berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh *client*. Mekanisme ini menyebabkan informasi yang diterima *client* selalu yang terbaru atau *up to date*. Semua *script* PHP dieksekusi pada *server* dimana *script* tersebut dijalankan.

### *9. JavaScript*

*JavaScript* adalah bahasa pemrograman *web* yang bersifat *Client Side Programming Language. Client Side Programming Language* adalah tipe bahasa pemrograman yang pemrosesannya dilakukan oleh *client*. Aplikasi *client* yang dimaksud merujuk kepada *web browser* seperti Google Chrome, Mozilla Firefox, Opera Mini dan sebagainya.

*JavaScript* pertama kali dikembangkan pada pertengahan dekade 90'an. Meskipun memiliki nama yang hampir serupa, *JavaScript* berbeda dengan bahasa pemrograman Java. Untuk penulisannya, *JavaScript* dapat disisipkan di dalam dokumen HTML ataupun dijadikan dokumen tersendiri yang kemudian diasosiasikan dengan dokumen lain yang dituju. *JavaScript* mengimplementasikan fitur yang dirancang untuk mengendalikan bagaimana sebuah halaman *web* berinteraksi dengan penggunanya.

Seperti tampilan pada *window* atau kendali pada menu dan *button. JavaScript* juga dapat digunakan untuk memvalidasi sebuah *web form* pada *browser* sebelum informasi pada *form* tersebut dikirim ke *server* (Henderson, 2009:256).

**10. PhpMyAdmin**

Sibero (2011:376) menyatakan bahwa, phpMyAdmin adalah aplikasi web yang dibuat oleh *phpMyAdmin.net*. phpMyAdmin digunakan untuk administrasi *database* MySQL. Program ini digunakan untuk mengakses *database* MySQL. Perintah untuk membuat tabel dapat menggunakan *form* yang sudah tersedia pada *phpMyAdmin* atau dapat langsung menuliskan *script* pada menu SQL.

**11. *Bootsrap***

Spurlock (2013:1) menyatakan bahwa *Bootsrap* adalah sebuah *framework* untuk CSS dan berupa produk open source yang dibuat oleh Mark Otto dan Jacob Thornton. Pada awalnya *Bootsrap* ini dibuat untuk membuat standarisasi front end untuk semua programmer di perusahaannya. *Bootsrap* telah berubah dari yang

sebelumnya adalah *CSS-Driven* proyek ke sebuah *host* dari *JavaScript plugins* dan ikon yang dapat dengan mudah digunakan untuk formulir dan tombol.

Dasar dari *Bootsrap* yaitu dapat digunakan untuk desain *web* yang *responsive* dan menampilkan dengan 12 kolom, 940 *pixel-wide grid*. *Bootsrap* dapat dimodifikasi sesuai dengan kebutuhan dengan memilih antara fitur CSS dan *JavaScript* yang dapat dimasukkan ke dalam *website* yang akan dibuat.

**12. *JQuery***

Beighley (2010:8) menjelaskan bahwa, JQuery adalah *open source add-on* pustaka *JavaScript* yang menekankan pada interaksi antara *JavaScript* dan HTML. *JQuery* merupakan kode *JavaScript* yang telah ditulis dan tinggal menambahkan satu atau dua baris kode untuk memanggil *JQuery*.

**13. *Sublime Text***

Bos (2014:12) menjelaskan bahwa, *Sublime Text* merupakan salah satu *text* editor yang sangat *powerful* yang dapat meningkatkan produktivitas dan mengembangkan kualitas kode yang tinggi.

**14. *Black* Box *Testing***

Black (2009 :3) menjelaskan bahwa, tester menggunakan *behavioral test* (disebut juga *Black-Box Tests*), sering digunakan untuk menemukan *bug* dalam *high level operations*, pada tingkatan fitur, profil operasional dan skenario *customer*. Tester dapat membuat pengujian fungsional *black box* berdasarkan pada apa yang harus sistem lakukan. *Behavioral* testing melibatkan pemahaman rinci mengenai domain aplikasi, masalah bisnis yang dipecahkan oleh sistem dan misi yang dilakukan sistem.

*Behavioral test* paling baik dilakukan oleh penguji yang memahami desain sistem, setidaknya pada tingkat yang tinggi sehingga mereka dapat secara efektif

menemukan bug umum untuk jenis desain. Menurut Nidhra dan Dondeti (2012:1), *black box testing* juga disebut *functional testing*, sebuah teknik pengujian fungsional yang merancang *test case* berdasarkan informasi dari spesifikasi.

## 2.2. Penelitian Terkait

Agusvianto (2017:1) berpendapat:
Manusia sebagai pengguna teknologi yang harus mampu memanfaatkan teknologi yang ada saat saat ini, maupun perkembangan teknologi tersebut selanjutnya. Adaptasi manusia dengan teknologi baru yang telah berkembang wajib untuk dilakukan melalui pendidikan. Hal ini dilakukan agar generasi penerus tidak tertinggal dalam hal teknologi baru. Dengan, teknologi dan pendidikan mampu berkembang bersama seiring dengan adanya generasi baru sebagai penerus generasi yang lama. Beberapa cara adaptasi tersebut dapat diwujudkan dalam bentuk sistem informasi *inventory*.

Sikumbang (2016:1) mengemukakan dalam jurnalnya:
Sistem persediaan barang yang dilakukan pada saat ini masih menggunakan cara manual, dimana pencatatan data masih menggunakan kertas sehingga kurang terjamin keakuratan data, kemungkinan terjadi kesalahan pencatatan dan perhitungan atas transaksi yang terjadi, kesulitan dalam mencari data yang dibutuhkan dan mengkontrol stok barang. Perancangan sistem informasi persediaan barang berbasis *web* ini menggunakan bahasa pemrograman PHP didukung dengan *database* MySQL. Model pengembangan sistem yang digunakan adalah model waterfall, analisis dan desain menggunakan diagram yang terkandung di dalam UML. Dengan sistem informasi persediaan barang berbasis *web* ini dapat mengurangi resiko kesalahan informasi dalam pencatatan persediaan barang, mempercepat pembuatan laporan dan membantu dalam menghasilkan keputusan-keputusan yang akurat dan cepat sehingga pelayanan terhadap pelanggan dapat meningkat dan membaik.