**BAB II**

**LANDASAN TEORI**

* 1. **Aplikasi**

Aplikasi adalah objek tertentu yang dapat dijalankan seperti kode sumber, kode objek atau sebuah program lengkap yang di tambahkan dengan semua item dan pelayanan pendukung yang secara keseluruhan sehingga dapat memenuhi kebutuhan pemakai. Aplikasi memiliki banyak bagian yang meliputi manual, refrensi, tutorial, intruksi instalasi, data sampel, pelayanan pendidikan, pelayanan pendukung teknis dan sebagainya. Semua yang dihasilkan oleh proyek aplikasi adalah produk kerja (Gerald, 2000).

Produk kerja meliputi :

1. Dokumen *enginering* yang dipakai untuk menentukan, mengontrol, dan memantau usaha kerja.
2. Objek yang dijalankan seperti *prototype*, kendali tes (*test harness*), dan piranti.
3. Pengembangan tujuan khusus.
4. Data yang digunakan untuk testing, melacak proyek dan sebagainya.

Selanjutnya aplikasi tersebut dibagi ke dalam 4 kelompok, yakni sebagai berikut :

1. Sistem operasi, adalah aplikasi yang digunakan untuk mengendalikan resource selama proses berlangsung.
2. Bahasa pemrograman, merupakan aplikasi yang digunakan sebagai alat untuk pengembangan program aplikasi.
3. Bahasa Query, merupakan aplikasi bahasa tingkat sangat tinggi yang dapat digunakan untuk menampilkan informasi-informasi yang diiginkan hanya dengan menuliskan sedikit perintah saja.
4. Aplikasi merupakan aplikasi yang dikembangkan untuk digunakan pada aplikasi tertentu.
   1. **Sistem**

Sistem adalah suatu jaringankerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu. Sebuah sistem haruslah teridir dari sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan (Gerald, 2000).

Ada beberapa komponen yang dapat membentuk sebuah sistem (Kadir, 2003), yaitu sebagai berikut :

1. Tujuan

Setiap sistem memiliki tujuan (goal) yang dapat mengarahkan sistem. Tanpa tujuan, sistem menjadi tidak terarah dan tidak terkendali.

1. Masukan

Masukan (*input*) sistem adalah segala sesuatu yang masuk ke dalam sistem dan menjadi bahan untuk diproses selanjutnya.

1. Proses

Merupakan bagian yang melakkan perubahan atau transformasi dari masukan menjadi keluaran yang diterima oleh pengguna.

1. Keluaran

Keluaran (*output*) sistem adalah hasil dari pemrosesan.

1. Mekanisme pengendalian dan umpan balik

Mekanisme pengendalian diwujudkan dengan menggunakan umpan balik (*feedback*) yang dapat digunakan untuk mengendalikan baik masukan maupunm proses yang bertujuan untuk mengatur sistem agar berjalan sesuai dengan tujuan awal yang diharapkan.

1. Batas

Batas sitem merupakan pemisah antara sistem dengan daerah di luar sistem yang dapat menentukan konfigurasi, ruang lingkup, atau kemampuan sistem.

1. Lingkungan

Lingkungan adalah segalah sesuatu yang berada di luar sistem yang dapat berpengaruh terhadap sistem, baik itu merugikan ataupun menguntungkan.

* 1. **Client Server**

*Client* *server* merupakan sebuah sistem jaringan yang mana terdapat sebuah komputer yang berfunsi sebagai *server* sementara komputer-komputer lain sebagai *client.* Komputer *server* bertugas untuk melayani komputer-komputer *client* yang terdapat pada jaringan tersebut (Arifin, 2005).

Keuntungan dari sistem *client-server*, diantaranya adalah menyediakan *user account*, sistem keamanan dan *acces control* secara terpusat, *user* hanya perlu mengingat satu *password* untuk mengakses beberpa *resource* yang tersedia (Arifin, 2005).

Dalam teknologi informasi, *client*-*server* merujuk kepada cara mendistribusikan aplikasi ke pihak *client* dan pihak *server*. Dalam model *client*-*server*, sebuah aplikasi dibagi menjadi dua bagian yang terpisah (tetapi masih dalam sebuah kesatuan) yakni komponen *client* dan komponen *server*.  
Komponen *client* dijalankan pada sebuah workstation. Pemakai workstation memasukkan data dengan menggunakan teknologi pemrosesan tertentu, kemudian mengirimkannya ke komponen *server*, umumnya berupa permintaan layanan tertentu yang dimiliki oleh *server*.

Komponen *server* akan menerima permintaan layanan tersebut dan langsung memprosesnya serta mengembalikan hasil pemrosesan kepada *client*. *Client* pun menerima informasi hasil pemrosesan data tadi dan menampilkannya kepada pemakai dengan menggunakan aplikasi yang digunakan oleh pemakai.  
Sebuah contoh dari aplikasi *client*-*server* sederhana adalah aplikasi *web* yang didesain dengan menggunakan Active *Server* Pages (ASP). Skrip ASP akan dijalankan di dalam *web* *server* (Apache atau Internet Information Services), sementara skrip yang berjalan di pihak *client* akan dijalankan oleh *web* browser pada komputer *client* (workstation).

*Client*-*server* merupakan penyelesaian masalah pada software yang menggunakan database sehingga setiap komputer tidak perlu diinstall database. Dengan metode *client*-*server* database dapat diinstal pada komputer *server* dan aplikasinya diinstal pada *client*.

* 1. **Bahasa Pemrograman Java**

Java merupakan bahasa berorientasi objek dan serbaguna. Kode Java dikompilasi dalam format yang disebut *bytecode*. *Bytecode* ini dapat dijalankan di semua komputer yang telah dilengkapi dengan program *Java Interpreter* dan *Java Virtual Machine*.

Java menyediakan sarana untuk membuat program (*applet*) yang berjalan pada *web* *browser*. Bahasa ini juga mendukung koneksi ke database, menyediakan sarana untuk membuat aplikasi berbasis Windows dan juga dapat dipakai untuk pemrograman jaringan (Gosling, 1994).

* 1. **JSP (Java Server Page)**

JSP adalah suatu teknologi *web* berbasis bahasa pemrograman Java dan berjalan di Platform Java, serta merupakan bagian teknologi J2EE (*Java 2 Enterprise Edition*). JSP sangat sesuai dan tangguh untuk menangani presentasi di *web*. Sedangkan J2EE merupakan platform Java untuk pengembangan sistem aplikasi *enterprise* dengan dukungan API (*Application Programming Interface*) yang lengkap dan portabilitas serta memberikan sarana untuk membuat suatu aplikasi yang memisahkan antara *business logic* (sistem), presentasi, dan data. Untuk dapat berjalan, JSP memerlukan JVM (*Java Virtual Machine*). Selain JVM, JSP juga memerlukan *server* yang disebut *Web Container* (Satriyanto dkk, 2010).

Teknologi JSP menyediakan cara yang lebih mudah dan cepat untuk membuat halaman *web* yang menampilkan isi secara dinamik. Teknologi JSP didesain untuk membuat lebih mudah dan cepat dalam membuat aplikasi berbasis *web* yang bekerja dengan berbagai macam *web* *server*, *application* *server*, browser, dan development tool. JSP merupakan bahasan scripting untuk *web* *programming* yang bersifat *server* *side* seperti halnya PHP dan ASP. JSP juga dapat berupa gabungan antara baris HTML dan fungsi dari JSP itu sendiri.

Seperti halnya skrip-skrip *server side* yang lain, JSP pun memerlukan *web* *server*. Skrip ASP memerlukan IIS sebagai *web server*, PHP memerlukan IIS atau Apache, sedangkan JSP bisa menggunakan Apache Tomcat sebagai salah satu *web* *server* yang mendukungnya agar bisa menjalankan file-file JSP yang berbasis Java, atau setidaknya JSP eng*i*ne yang dapat terintegrasi dengan *web* *server*.

* 1. **Android**

Android adalah [sistem operasi](https://id.wikipedia.org/wiki/Sistem_operasi) berbasis [Linux](https://id.wikipedia.org/wiki/Linux) yang dirancang untuk perangkat bergerak [layar sentuh](https://id.wikipedia.org/wiki/Layar_sentuh) seperti [telepon pintar](https://id.wikipedia.org/wiki/Telepon_pintar) dan [komputer tablet](https://id.wikipedia.org/wiki/Komputer_tablet). Android awalnya dikembangkan oleh Android, Inc., dengan dukungan finansial dari [Google](https://id.wikipedia.org/wiki/Google), yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya [Open Handset Alliance](https://id.wikipedia.org/wiki/Open_Handset_Alliance), konsorsium dari perusahaan-perusahaan [perangkat keras](https://id.wikipedia.org/wiki/Perangkat_keras_komputer), aplikasi, dan telekomunikasi yang bertujuan untuk memajukan [standar terbuka](https://id.wikipedia.org/wiki/Standar_terbuka) perangkat seluler. [Ponsel Android pertama](https://id.wikipedia.org/wiki/HTC_Dream) mulai dijual pada bulan Oktober 2008.

Sistem operasi android atau [os android](https://www.riaume.com/daftar-nama-os-android-menurut-versi.html) terdiri dari beberapa versi ,setiap versi android terbaru memiliki nama-nama unik tersendiri dan memiliki beberapa jenis kelebihan mulai dari tampilan hingga optomasi keamana,berikut daftar nama [“os android”](https://www.riaume.com/daftar-nama-os-android-menurut-versi.html) menurut versi .mulai dari nama os android pertama kali di keluarkan sampai dengan os android versi terbaru yang baru di keluarkan di tahun 2016 ini :

* 1. OS android pertama versi 1.0 di rilis pada tanggal 23 bulan September tahun 2008 dan android versi 1.1 Dirilis pada tanggal 9 bulan Februari tahun 2009 di namakan dengan Android.
  2. OS android versi 1.5 Dirilis pertama pada tanggal 30 bulan April tahun 2009 di namakan dengan Android Cupcake.
  3. OS android versi 1.6 Dirilis pada tanggal 15 bulan September tahun 2009 di namakan dengan Android Donut.
  4. OS android versi 2.0 Dirilis pada tanggal 26 bulan Oktober tahun 2009 dan beberapa versi lainya seperti android v2.0.1 dan android V2.1 di namakan dengan Android Eclair
  5. OS android versi 2.2 dan versi v2.2.3 Dirilis pada tanggal 10 bulan mei tahun 2010 di namakan dengan Android Froyo
  6. OS android versi 2.3 sampai dengan versi v2.3.7 Dirilis pertama pada tanggal 6 bulan Desember tahun 2010 di namakan dengan Android Gingerbread . android Gingerbread menjadi salah satu os android tersukses di tahun 2010 , hingga saat ini masih banyak sekali ponsel android yang menggunakan versi ini.
  7. OS android versi v3.0 sampai dengan versi v3.2 Dirilis pertama pada tanggal 22 bulan Februari tahun 2011 di namakan dengan Android Honeycomb ,os android Honeycomb merupakan os android paling sempurna di bandingkan dengan os android versi sebelumnya,os ini terkenal akan kinerja os yang sangat baik dan memiliki tampilan mewah.jadi tidak heran apabila di tahun 2011 ponsel yang menggunakan android Honeycomb yang paling banyak di cari.
  8. OS android versi v4.0 sampai dengan v4.0.2 Dirilis pertama pada tanggal 19 bulan Oktober tahun 2011 di namakan dengan Android Ice Cream Sandwich . ini merupakan versi Os Android tersukses dan paling banyak cari,tampilan pada Android Ice Cream Sandwich sudah tidak terlalu jauh dengan android jelly bean tersebut,dan sampai saat ini android Ice Cream Sandwich masih banyak di gunakan pada ponsel kelas menengah kebawah dan beberapa samsung galaxy tab terdahulu.kelebihan utama dari android ice crean sandwich ini adalah sudah mendukung Flash player sehingga bisa memudahkan kita browsing di internat dll
  9. OS android versi v4.1 sampai dengan v4.3 Dirilis pertama pada tanggal 9 bulan juli tahun 2012 di namakan dengan Android Jelly Bean.
  10. Android Jelly Bean merupakan salah satu os yang di perbaharui dari versi ice cream sandwich,dari segi tampilan os android jelly bean paling sempurna di banding beberapa os terdahulu,tidak hanya memiliki tampilan yang lebih dinamis ,android jelly bean juga telah di optimasi pada bagian kinerja os super cepat.di tambah dengan beberapa peningkatan kinerja pada sisi kamera pada smartphone dan tablet pc,penambahan navigasi Gesture dan banyak lagi,sampai di tahun 2014 ini android jelly bean masih menjadi salah satu os yang paling banyak di gunakan dan masih banyak sekali ponsel terbaru android yang masih menggunakan os android jelly bean satu ini,
  11. OS android versi v4.4 sampai dengan v4.4.4 Dirilis pertama pada tanggal 31 bulan Oktober tahun 2013 di namakan dengan Android kitkat .os android kitkat memiliki tampilan 100% lebih dinamis dan berbeda total dengan android jelly bean,android kitkat di optimasi pada sisi konsumsi baterai dan kinerja os lebih cepat ketika di jalankan pada perangkat memiliki spesifikasi lebih rendah,seperti kita tahu jika andorid jelly bean memiliki kelebihan pada sisi konsumsi baterai yang lebih tinggi dan ketika di jalankan di perangkat yang memiliki versi rendah os ini tidak maksimal.
  12. OS android versi v5.0 dirilis pertama pada tahun 2014 dikenal dengan nama “Android Lollipop”, android lollipop memiliki peningkatan sistem keamanan dan tampilan lebih dinamis ,os versi terbaru dari android ini sepertinya bisa di gunakan berbagai perangkat elektronik seperti tv dll.
  13. OS Android versi 6.0 bernama Marshmallow di kenal juga dengan nama Android M ,os android Marshmallow terbaru ini merupakan pemuktakhiran untuk os android lollipop v5.0 sebelum nya.dari segi tampilan os android versi 6.0 ini tidak terlalu jauh berbeda di banding dengan os android lollipop sebelumnya, hanya saja ada terdapat pembaharuan keamanan atau tingkat keamanan os android 6.0 ini di tingkatkan dan di tambah beberapa fitur unggulan lainnya.
  14. OS Android versi 7.0 bernama Nougat. Pada versi ini Google telah menyempurnakan OS ini menjadi sangat nyaman dan mudah digunakan. Diantara fitur yang ditambahkan dari versi sebelumnya adalah : *direct reply notification, multi window support, quick setting menu,* dan *doze mode* 2.0*.*
  15. **Java Development Kit (JDK)**

Java Development Kit (JDK) merupakan aplikasi yang digunakan untuk menajemen dan membangun berbagai aplikasi Java. JDK merupakan superset dari JRE, berisikan segala sesuatu yang ada di JRE ditambahkan compiler dan debugger yang diperlukan untuk mengembangkan applet dan aplikasi.

Dalam satu paket JDK, terdapat dua komponen penting yang membuat sebuah aplikasi jAVA dapat berjalan. komponen penting itu adalah JRE dan JVM.

1. Java Runtime Environment(JRE)

Java Runtime Environment (JRE), beberapa praktisi cukup menyebutkannya dengan Java Runtime, merupakan bagian dari JDK, dan merupakan kumpulan dari ”tools” programming, class-class inti, dan banyak file-file pendukung aplikasi. JRE menyediakan kebutuhan minimum untuk mengeksekusi aplikasi-aplikasi Java. salah satu bagian dari JRE adalah JVM (Java Virtual Machine).

1. Java Virtual Machine (JVM)

Java Virtual Machine (JVM) adalah sebuah spesifikasi untuk sebuah komputer abstrak. JVM terdiri dari kelas pemanggil dan sebuah interpreter Java yang mengeksekusi kode arsitektur netral. Kelas pemanggil memanggil file .class dari kedua program Java dan Java API untuk dieksekusi oleh interpreter Java. Intrepreter Java mungkin sebuah aplikasi interpreter yang menterjemahkan satu kode byte pada satu waktu, atau mungkin sebuah just-intime (JIT) kompiler yang menurunkan bytecode arsitektur netral kedalam bahasa mesin untuk host komputer.

* 1. **Software Development Kit (SDK)**

Android-SDK merupakan tools bagi para programmer yang ingin mengembangkan aplikasi berbasis google android. Android SDK mencakup seperangkat alat pengembangan yang komprehensif. Android SDK terdiri dari debugger, libraries, handset emulator,dokumentasi, contoh kode, dan tutorial. Saat ini Android sudah mendukung arsitektur x86 pada Linux (distribusi Linux apapun untuk desktop modern), Mac OS X 10.4.8 atau lebih, Windows XP atau Vista. Persyaratan mencakup JDK, Apache Ant dan Python 2.2 atau yang lebih baru. IDE yang didukung secara resmi adalah Eclipse 3.2 atau lebih dengan menggunakan pluginAndroid Development Tools (ADT), dengan ini pengembang dapat menggunakan teks editor untuk mengedit file Java dan XML serta menggunakan peralatan command line untuk menciptakan, membangun, melakukan debug aplikasi Android dan pengendalian perangkat Android (misalnya, reboot, menginstal paket aplikasi dengan jarak jauh).

Android SDK telah dirilis pada tanggal 12 November 2007. Dan pada tanggal 15 Juli 2008 tim Android Developer Challenge sengaja mengirimkan email ke semua pendatang di Android Developer Challenge untuk mengumumkan bahwa rilis SDK terbaru telah tersedia pada halaman download pribadi. Email tersebut juga ditujukan kepada pemenang Android Developer Challenge putaran pertama. Sebuah penyataan bahwa Google telah menyediakan rilis SDK terbaru untuk beberapa pengembang dan bukan untuk orang lain.

Pada tanggal 18 Agustus 2008, Android SDK 0.9 beta dirilis. Rilis ini menyediakan API yang diperbarui dan diperluas, perbaikan pada alat-alat pengembangan dan desain terbaru untuk layar awal. Petunjuk untuk meng-upgrade SDK sudah tersedia pada rilis sebelumnya. Pada tanggal 23 September 2008, Android 1.0 SDK telah dirilis. Pada tanggal 9 Maret 2009, Google merilis versi 1.1 untuk telepon seluler Android. Rilis terbaru tersebut termasuk dukungan untuk pencarian dengan suara, harga aplikasi, perbaikan jam alarm, perbaikan pengiriman gmail, perbaikan surat pemberitahuan dan peta.

Pada pertengahan Mei 2009, Google merilis versi 1.5 (Cupcake) pada sistem operasi Android dan SDK. Pembaruan ini termasuk banyak fitur baru seperti perekaman video, dukungan untuk bluetooth, sistem keyboard pada layar dan pengenalan suara. Rilis ini juga membuka AppWidget framework kepada para pengembang yang memungkinkan orang untuk membuat widget sendiri pada halaman home. Pada September 2009 versi 1.6 (Donut) dirilis yang menampilkan hasil pencarian yang lebih baik dan penggunaan indikator baterai.

Aplikasi Android dipaketkan ke dalam format .apk dan disimpan pada folder /data/app. Pengguna dapat menjalankan perintah adb root untuk mengakses folder tersebut karena root memiliki izin untuk mengakses folder tersebut.

* 1. **Android Studio**

Android Studio adalah sebuah IDE untuk Android Development yang diperkenalkan google pada acara Google I/O 2013. Android Studio merupakan pengembangkan dari Eclipse IDE, dan dibuat berdasarkan IDE Java populer, yaitu IntelliJ IDEA. Android Studio merupakan IDE resmi untuk pengembangan aplikasi Android.

Sebagai pengembangan dari Eclipse, Android Studio mempunyai banyak fitur-fitur baru dibandingkan dengan Eclipse IDE. Berbeda dengan Eclipse yang menggunakan Ant, Android Studio menggunakan Gradle sebagai build environment. Fitur-fitur lainnya adalah sebagai berikut :

1. Menggunakan Gradle-based build system yang fleksibel.
2. Bisa mem-build multiple APK .
3. Template support untuk Google Services dan berbagai macam tipe perangkat.
4. Layout editor yang lebih bagus.
5. Built-in support untuk [Google Cloud Platform](https://developers.google.com/cloud/mobile), sehingga mudah untuk integrasi dengan Google Cloud Messaging dan App Engine.
6. Import library langsung dari Maven repository dan masih banyak lagi lainnya.
   1. **MySql**

MySQL merupakan sebuah aplikasi atau software sistem manajemen basis data SQL atau  DBMS Multithread dan multi user. MySQl sebenarnya merupakan turunan dari salah satu konsep utama dalam database untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengoperasian data dikerjakan secara mudah dan otomatis. MySQL diciptakan oleh Michael "Monty" Widenius pada tahun 1979, seorang programmer komputer asal Swedia yang mengembangkan sebuah sistem database sederhana yang dinamakan UNIREG yang menggunakan koneksi low-level ISAM database engine dengan indexing.

Pada saat ini MySQL merupakan database *server* yang sangat terkenal di dunia, semua itu tak lain karena bahasa dasar yang digunakan untuk mengakses database yaitu SQL. SQL (*Structured Query Language*) pertama kali diterapkan pada sebuah proyek riset pada laboratorium riset San Jose, IBM yang bernama system R. Kemudian SQL juga dikembangan oleh Oracle, Informix dan Sybase. Dengan menggunakan SQL, proses pengaksesan database lebih user-friendly dibandingan dengan yang lain, misalnya dBase atau Clipper karena mereka masih menggunakan perintah-perintah pemrograman murni.

* 1. **Hyper Text Markup Language (HTML)**

HTML (Hyper Text Markup Language) adalah sebuah *bahasa markup* yang digunakan untuk membuat sebuah halaman *web* dan menampilkan berbagai informasi di dalam sebuah browser Internet. Bermula dari sebuah bahasa yang sebelumnya banyak digunakan di dunia penerbitan dan percetakan yang disebut dengan SGML *(Standard Generalized Markup Language),* HTML adalah sebuah standar yang digunakan secara luas untuk menampilkan halaman *web*. HTML saat ini merupakan standar Internet yang didefinisikan dan dikendalikan penggunaannya oleh World Wide *Web* Consortium (W3C).

HTML berupa kode-kode tag yang menginstruksikan browser untuk menghasilkan tampilan sesuai dengan yang diinginkan. Sebuah file yang merupakan file HTML dapat dibuka dengan menggunakan browser *web* seperti Mozilla Firefox atau Microsoft Internet Explorer. HTML juga dapat dikenali oleh aplikasi pembuka email ataupun dari PDA dan program lain yang memiliki kemampuan browser.

HTML dokumen tersebut mirip dengan dokumen teks biasa, hanya dalam dokumen ini sebuah teks bisa memuat instruksi yang ditandai dengan kode atau lebih dikenal dengan TAG tertentu. Sebagai contoh jika ingin membuat teks ditampilkan menjadi tebal seperti: **TAMPIL TEBAL**, maka penulisannya dilakukan dengan cara: <b>TAMPIL TEBAL</b>. Tanda <b> digunakan untuk mengaktifkan instruksi cetak tebal, diikuti oleh teks yang ingin ditebalkan, dan diakhiri dengan tanda </b> untuk menonaktifkan cetak tebal tersebut.

Secara garis besar, terdapat 4 jenis elemen dari HTML:

1. *structural*. tanda yang menentukan level atau tingkatan dari sebuah teks (contoh, <h1>Golf</h1> akan memerintahkan browser untuk menampilkan “Golf” sebagai teks tebal besar yang menunjukkan sebagai Heading 1
2. *presentational*. tanda yang menentukan tampilan dari sebuah teks tidak peduli dengan level dari teks tersebut (contoh, <b>boldface</b> akan menampilkan **bold**. Tanda presentational saat ini sudah mulai digantikan oleh CSS dan tidak direkomendasikan untuk mengatur tampilan teks,
3. *hypertext*. tanda yang menunjukkan pranala ke bagian dari dokumen tersebut atau pranala ke dokumen lain (contoh, <a href="http://www.ilmukita.com/">IlmuKita</a> akan menampilkan [IlmuKita](http://ilmukita.com/) sebagai sebuah hyperlink ke URL tertentu),
4. Elemen *widget* yang membuat objek-objek lain seperti tombol (<button>), list (<li>), dan garis horizontal (<hr>).

Selain markup *presentational* , markup yang lin tidak menentukan bagaimana tampilan dari sebuah teks. Namun untuk saat ini, penggunaan tag HTML untuk menentukan tampilan telah dianjurkan untuk mulai ditinggalkan dan sebagai gantinya digunakan Cascading Style Sheets.

* 1. **Web Server**

Pengertian *Web* *server* adalah sebuah software yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari HTTP atau HTTPS pada klien yang dikenal dan biasanya kita kenal dengan nama *web* browser dan untuk mengirimkan kembali yang hasilnya dalam bentuk beberapa halaman *web* dan pada umumnya akan berbentuk dokumen HTML. itulah pengertian *web* *server* sebenarnya. dalam bentuk sederhana *web* *server* akan mengirim data HTML kepada permintaan *web* Browser sehingga akan terlihat seperti pada umumnya yaitu sebuah tampilan *web*site.

Fungsi utama *Web* *server* adalah untuk melakukan atau akan tranfer berkas permintaan pengguna melalui protokol komunikasi yang telah ditentukan sedemikian rupa. halaman *web* yang diminta terdiri dari berkas teks, video, gambar, file dan banyak lagi. pemanfaatan *web* *server* berfungsi untuk mentransfer seluruh aspek pemberkasan dalam sebuah halaman *web* termasuk yang di dalam berupa teks, video, gambar atau banyak lagi.

* 1. **Ayam Bakar Wong Solo Cabang Tuban**

Rumah makan Ayam Bakar Wong Solo adalah bisnis rumah makan yang bergerak dalam bisnis jasa, berdiri sejak 18 April 1991 di bawah kendali Bapak Puspo Wardoyo dan berkantor pusat di kota Medan dengan cabang hampir di seluruh propinsi di Indonesia. Dari seorang mantan guru SMA Perguruan Wahidin Bagan Si Api-api, beliu beralih profesi menjadi wirausahawan dengan membuka ayam bakar kaki lima di jalan SMA 2 Padang Golf Polonia Medan

Rumah Makan Ayam Bakar Wong Solo salah satu rumah makan tradisional besar yang bermoto "**HALLALAN TAYYIBAN"**yang berarti halal dari segi makan dan baik dari segi aspek pengolahan dan pelayanan. Walaupun menggunakan ayan bakar dan nama wong solo bukan berarti menu yang ada adalah ayam bakar saja tetapi juga tersedia menu-menu yang lain seperti ikan dan sayuran. Menggunakan nama wong solo karena pemilik rumah makan ini adalah orang dari kota Solo. Namun demikian menu yang ada adalah menu tradisional nusantara atau di ambil dari berbagai daerah di Indonesia.

Ayam Bakar Wong Solo Tuban adalah salah satu cabang dari sekitar 156 cabang di seluruh Indonesia. Cabang wilayah Tuban Jawa Timur terbilang belum terlalu lama, terhitung baru 2 tahun berjalan. Namun demikian pengunjung yang datang selalu ramai, terlebih saat bulan Ramadhan yang sering dijadikan pilihan tempat untuk buka bersama.

Ayam Bakar Wong Solo cabang Tuban bertempat di Jalan Sunan Kalijaga No. 19 Kel. Latsari, Tuban Jawa Timur.

* 1. **Flowchart**

Flowchart dapat diartikan sebagai gambaran dalam bentuk diagram alir dari algoritma-algoritma dalam suatu program yang menyatakan arah alur program tersebut (Pahlevy, 2010).

Berikut adalah daftar simbol flowchart yang ditunjukkan pada Tabel 2.1 :

Tabel 2.1 Simbol Flowchart (Pahlevy, 2010)

|  |  |  |  |
| --- | --- | --- | --- |
| **Simbol** | **Nama dan Keterangan** | **Simbol** | **Nama dan Keterangan** |
|  | **Flow Direction Symbol**  Digunakan untuk menghubungkan simbol satu dengan simbol yang lain. |  | **Symbol Manual Input**  Untuk Memasukkan Datan Secara Manual. |
|  | **Terminator Symbol**  Untuk permulaan (Start) atau akhiran (Finish) dari suatu kegiatan. |  | **Symbol Preparation**  Untuk mempersiapkan penyimpanan yang akan digunakan sebagai storage. |
| **Simbol** | **Nama dan Keterangan** | **Simbol** | **Nama dan Keterangan** |
|  | **Connector Symbol**  Untuk keluar masuk atau penyambungan proses pada lembar / halaman yang berbeda. |  | **Symbol Display**  Untuk menyatakan peralatan output. |
|  | **Processing Symbol**  Untuk melakukan suatu pengolahan oleh komputer. |  | **Symbol Disk**  Untuk menyatakan input dari suatu diks atau menyimpan ke disk. |
|  | **Symbol Manual Operation**  Untuk melakukan suatu pengolahan bukan komputer. |  | **Symbol Magnetic Tape Unit**  Untuk menyatakan input dari suatu pita magnetik atau menyimpan ke pita magnetik. |
|  | **Symbol Decision**  Untuk proses memilih kondisi yang ada. |  | **Symbol Punch Crad**  Untuk menyatakan input dari kartu atau menyimpan ke kartu. |
|  | **Symbol Input-Output**  Untuk melakukan suatu proses input atau output. |  | **Symbol Document**  Untuk menyatakan input dari suatu dokumen atau menyimpan ke dokumen. |

1. **Kegunaan Flowchart**

Pada awalnya, flowchart digunakan oleh para insinyur industri sebagai sebuah struktur proses kerja, seperti manufaktur perakitan. Saat ini, flowchart telah digunakan untuk berbagai keperluan di bidang manufaktur, arsitektur, teknik, bisnis, teknologi, pendidikan, ilmu pengetahuan, kedokteran, pemerintahan, administrasi dan disiplin ilmu lainnya.

1. Perencanaan Proyek Baru

Salah satu penggunaan umum dari flowchart adalah untuk memetakan proyek baru, terutama yang bersifat teknis atau membutuhkan beberapa desain struktur lain. Insinyur dan desainer aplikasi sering menggunakan diagram alur untuk tujuan ini. Berpikir tentang bagaimana proses akan bekerja menggunakan flowchart dapat menghemat banyak waktu dalam desain.

2. Desain Sistem atau Program

Menggunakan flowchart memungkinkan desainer untuk memikirkan banyak masalah yang kompleks di muka. Karena banyak sistem atau program memerlukan langkah-demi-langkah prosedur, sehingga menggunakan diagram alur adalah cara yang logis untuk memikirkan langkah-langkah tersebut. Jenis proses desain dapat jelas dipikirkan dan disampaikan kepada orang lain melalui penggunaan flowchart.

3. Dokumen Proses untuk Alasan Eksternal

Bisnis sering memiliki persyaratan eksternal untuk dokumentasi proses dan flowchart dapat berguna untuk tujuan ini. Dari standar pelaporan untuk mematuhi peraturan pemerintah, menggunakan flowchart bisa jauh lebih efisien daripada menghasilkan narasi tertulis. Standar pelaporan untuk mematuhi peraturan kerja, dapat menggunakan flowchart yang bisa jauh lebih efisien daripada menghasilkan narasi tertulis.

4. Dokumen Proses untuk Alasan internal

Penggunaan diagram alur, untuk lebih memahami bagaimana proses bisnis kerja menawarkan banyak keuntungan untuk organisasi:

* 1. Fleksibilitas – karyawan baru ataupun yang sudah ada dapat segera mempelajari cara yang tepat untuk melakukan pekerjaan apa pun.
  2. Kualitas – Jika semua orang di tim melakukan pekerjaan dengan cara yang sama setiap kali, hasilnya dapat diprediksi dan konsisten.
  3. Visibilitas – Manajemen dapat melihat persis bagaimana setiap pekerjaan yang dimaksudkan untuk dilkerjakan
  4. Perbaikan proses: Kita dapat memperbaiki cara organisasi, jika kita tahu bagaimana cara tersebut dilakukan sekarang.

5. Pemecahan Masalah

Salah satu alasan, mengapa sering digunakan flowchart dalam pembuatan program atau desain jaringan komputer adalah bahwa flowchart dapat menawarkan sumber yang baik untuk pemecahan masalah. Hal ini juga sebagai cara yang bagus, jika digunakan untuk pelanggan, untuk menunjukkan cara pemecahan masalah yang bersifat umum. karena pendekatan visual akan leboh mudah diikuti, ketika disajikan dengan benar.

1. **Jenis–Jenis Flowchart**

1. Flowchart Sistem

Flowchart Sistem merupakan bagan yang menunjukkan alur kerja atau apa yang sedang dikerjakan di dalam sistem secara keseluruhan dan menjelaskan urutan dari prosedur-prosedur yang ada di dalam sistem. Dengan kata lain, flowchart ini merupakan deskripsi secara grafik dari urutan prosedur-prosedur yang terkombinasi yang membentuk suatu sistem.

Flowchart Sistem terdiri dari data yang mengalir melalui sistem dan proses yang mentransformasikan data itu. Data dan proses dalam flowchart sistem dapat digambarkan secara online (dihubungkan langsung dengan komputer) atau offline (tidak dihubungkan langsung dengan komputer, misalnya mesin tik, cash register atau kalkulator).

2.   Flowchart Paperwork / Flowchart Dokumen  
Flowchart Paperwork menelusuri alur dari data yang ditulis melalui sistem. Flowchart Paperwork sering disebut juga dengan Flowchart Dokumen. Kegunaan utamanya adalah untuk menelusuri alur form dan laporan sistem dari satu bagian ke bagian lain baik bagaimana alur form dan laporan diproses, dicatat dan disimpan.

3. Flowchart Skematik  
Flowchart Skematik mirip dengan Flowchart Sistem yang menggambarkan suatu sistem atau prosedur. Flowchart Skematik ini bukan hanya menggunakan simbol-simbol flowchart standar, tetapi juga menggunakan gambar-gambar komputer, peripheral, form-form atau peralatan lain yang digunakan dalam sistem. Flowchart Skematik digunakan sebagai alat komunikasi antara analis sistem dengan seseorang yang tidak familiar dengan simbol-simbol flowchart yang konvensional. Pemakaian gambar sebagai ganti dari simbol-simbol flowchart akan menghemat waktu yang dibutuhkan oleh seseorang untuk mempelajari simbol abstrak sebelum dapat mengerti flowchart. Gambar-gambar ini mengurangi kemungkinan salah pengertian tentang sistem, hal ini disebabkan oleh ketidak-mengertian tentang simbol-simbol yang digunakan. Gambar-gambar juga memudahkan pengamat untuk mengerti segala sesuatu yang dimaksudkan oleh analis, sehingga hasilnya lebih menyenangkan dan tanpa ada salah pengertian.

4.Flowchart Program  
Flowchart Program dihasilkan dari Flowchart Sistem. Flowchart Program merupakan keterangan yang lebih rinci tentang bagaimana setiap langkah program atau prosedur sesungguhnya dilaksanakan. Flowchart ini menunjukkan setiap langkah program atau prosedur dalam urutan yang tepat saat terjadi. Programmer menggunakan flowchart program untuk menggambarkan urutan instruksi dari program komputer. Analis Sistem menggunakan flowchart program untuk menggambarkan urutan tugas-tugas pekerjaan dalam suatu prosedur atau operasi.

5. Flowchart Proses  
Flowchart Proses merupakan teknik penggambaran rekayasa industrial yang memecah dan menganalisis langkah-langkah selanjutnya dalam suatu prosedur atau sistem.

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

ERD (*Entity Relationship Diagram*) berfungsi untuk menggambarkan relasi dari dua file atau dua tabel yang dapat digolongkan dalam tiga macam bentuk relasi, yaitu satu-satu, satu-banyak, dan banyak-banyak. ERD menggunakan sejumlah notasi dan simbol untuk menggambarkan struktur dan hubungan antar data (Oetomo, 2002).

Pada dasarnya ada tiga macam simbol yang digunakan yaitu sebagai berikut :

* 1. Entiti

Entiti adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai, suatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat.

* 1. Atribut

Atribut berfungsi untuk mendeskripsikan karakter entiti.

* 1. Hubungan.

Entiti dapat berhubungan satu sama lain. Hubungan ini dinamakan *relationships* (relasi).

Sedangkan pendapat lain mengemukakan bahwa ERD digunakan untuk memodelkan struktur data dan hubungan antar data, karena hal ini relatif kompleks. Dengan ini ERD dapat menguji model dengan mengabaikan proses yang harus dilakukan (Fathansyah, 2002).

Berdasarkan berbagai pengertian di atas dapat disimpulkan bahwa ERD adalah salah satu metode pemodelan basis data yang digunakan untuk menghasilkan skema konseptual untuk jenis model data semantik sistem. Dimana sistem seringkali memiliki basis data relasional, dan ketentuannya bersifat *top-down.*

Tabel 2.2 Simbol-Simbol ERD (*Entity Relationship Diagram*) (Fathansyah, 2002)

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Notasi** | **Nama** | **Arti** |
| 1 |  | *Entity* | Objek yang dapat dibedakan dalam dunia nyata |
| 2 |  | *Week Entity* | Suatu *entity* dimana keberadaan dari *entity* tersebut tergantung dari keberadaan *entity* yang lain |
| 3 |  | *Relationship* | Hubungan yang terjadi antara satu atau lebih *entity* |
| 4 |  | *Identifying*  *Relationship* | Hubungan yang terjadi antara satu atau lebih *weak entity* |
| 5 |  | *Atribut Simple* | *Atribut* yang bernilai tunggal atau atribut *atomic* yang tidak dapat dipilah-pilah lagi |
| 6 |  | *Atribut Primary Key* | Satu atau gabungan dari beberapa *atribut* yang membedakan semua baris data *(row)* dalam *table* secara unik |
| 7 |  | *Atribut Composite* | *Atribut* yang masih dapat diuraikan lagi menjadi sub-sub *atribut* yang masing-masing memiliki makna |
| 8 |  | *Atribut Multivalue* | Suatu *atribut* yang memiliki sekelompok nilai untuk setiap *instant entity* |

1. **UML (Unified Modelling Language)**

UML adalah bahasa grafis untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak. UML berorientasi objek menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak tergantung pada bahasa dan teknologi, pemaduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak. Standar UML dikelola oleh OMG (Object Management Group) (Nugroho, 2010).

UML adalah bahasa pemodelan untuk menspesifikasikan, memvisualisasikan, membangun dan mendokumentasikan artifak- artifak dari sistem.

Diagram-diagram yang terdapat pada pemodelan UML sebagai berikut:

1. *Usecase Diagram*

Usecase diagram merupakan salah satu diagram untuk memodelkan aspek perilaku sistem. Masing-masing diagram usecase menunjukan sekumpulan usecase, aktor dan hubungannya. Diagram usecase digunakan untuk memvisualisasikan, menspesifikasikan dan mendokumentasikan kebutuhan perilaku sistem. Diagram-diagram usecase merupakan pusat permodelan perilaku sistem, subsistem dan kelas.

1. *Class Diagram*

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

1. Nama (dan stereotype)
2. Atribut
3. Metoda
4. *Statechart Diagram*

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu state ke state lainnya) suatu objek pada system sebagai akibat dari stimuli yang diterima. Pada umumnya *statechart diagram menggambarkan class tertentu (satu class* dapat memiliki lebih dari satu statechart diagram).

1. *Activity Diagram*

Activity diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa 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 perilaku internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

1. *Sequence Diagram*

*Sequence diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa message yang digambarkan terhadap waktu. Sequence diagram terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

*Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang menjadi trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan.

1. *Collaboration Diagram*

Collaboration diagram juga menggambarkan interaksi antar objek seperti sequence diagram, tetapi lebih menekankan pada peran masing masing objek dan bukan pada waktu penyampaian *message*. Setiap message memiliki sequence number, di mana message dari level tertinggi memiliki nomor 1. Messages dari level yang sama memiliki prefiks yang sama.

1. *Component Diagram*

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (dependency) di antaranya. Komponen piranti lunak adalah modul berisi code, baik berisi source code maupun binary code, baik *library* maupun *executable,* baik yang muncul pada *compile time,* link time, maupun run time. Umumnya komponen terbentuk daribeberapa class dan, atau package, tapi dapat juga dari komponen-komponen yang lebih kecil.

1. *Deployment Diagram*

*Deployment* atau *physical diagram* menggambarkan detail bagaimana komponen di-deploy dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisik. Sebuah node adalah server, workstation, atau piranti keras lain yang digunakan untuk men-deploy komponen dalam lingkungan sebenarnya. Hubungan antar node (misalnya TCP/IP) dan requirement dapat juga didefinisikan dalam diagram ini.

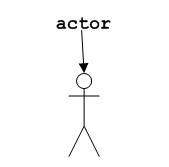
1. **Notasi UML**

Notasi UML diturunkan dari tiga notasi yang sudah ada sebelumnya yaitu Grady Booch OOD *(Object Oriented Design)*, Jim Rumbough OMT *(Object Modelling Technique)*, dan Ivar Jacobson *OOSE (Object Oriented Sofware Engineering).* Notasi UML yang digunakan sekarang merupakan penggabungan dari tiga notasi tersebut, yang terdiri dari :

1. Aktor **(***actor****)***

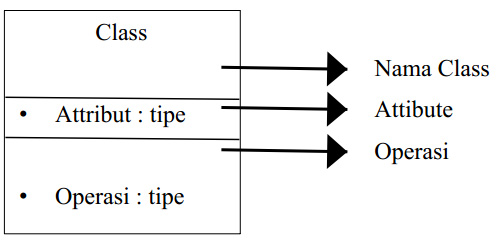
Dalam pemodelan sistem dengan UML, aktor adalah seseorang atau sesuatu yang berinteraksi dengan sistem yang sedang kita kembangkan. Aktor berada diluar lingkup sistem atau perangkat lunak yang sedang kita kembangkan; bersifat eksternal (Nugroho, 2010).

Secara prinsip dapat kita kenali 3 jenis aktor untuk hamper semua sistem atau perangkat lunak yang kita kembangkan: para pengguna sistem, perangkat lunak lain yang berinteraksi dengan sistem, dan perangkat lunak yang kita kembangkan serta waktu. Jadi aktor ini bisa berupa orang, perangkat keras atau mungkin juga objek lain dalam sistem yang sama. Biasanya yang dilakukan oleh aktor adalah memberikan informasi pada sistem dan atau memerintahkan sistem untuk melakukan sesuatu.



Gambar 2.1 Notasi Aktor Sumber: (Nugroho, 2010)

Kelas seperti juga objek, adalah sesuatu yang membungkus (*encapsulate*) informasi dan perilaku dalam dirinya (Nugroho, 2006). Dalam pengembangan sistem tradisional, kita mengadakan pendekatan dengan cara memisahkan informasi-informasi pada sisi basis data dan perilaku yang mengaksesnya di sisi aplikasi pemasup atau pengakses. Pendekatan berorientasi objek berbeda, yaitu menggabungkan potongan-potongan informasi dengan perilaku yang akan mengaksesnya dalam apa yang dinamakan kelas.

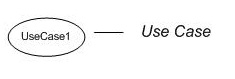


Gambar 2.2 Notasi Kelas Sumber : (Nugroho, 2010)

1. *Usecase*

*Usecase* adalah peringkat tertinggi dari fungsional yang dimiliki sistem (Nugroho, 2006). Dengan kata lain, usecase menggambarkan bagaimana seseorang akan menggunakan sistem. Usecase menjelaskan suatu urutan kegiatan yang dilakukan oleh aktor dan sistem untuk mencapai tujuan tertentu walaupun menjelaskan kegiatan namun usecase hanya menjelaskan apa yang dilakukan oleh aktor dan sistem, bukan bagaimana aktor dan sistem melakukan kegiatan tersebut.

Keunggulan dari cara memandang sistem sebagai kumpulan usecase adalah kemampuannya untuk memisahkan implementasi sistem dari alasan mengapa sistem harus ada. Ia akan membantu kita untuk berfokus pada apa yang paling penting, yaitu menentukan apa yang dibutuhkan serta apa harapan pengguna terhadap sistem atau perangkat lunak yang sedang dikembangkan.



Gambar 2.3 Notasi Usecase Sumber: (Nugroho, 2010)

1. *Interaction*

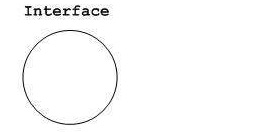
Menurut Interaction digunakan untuk menunjukan baik aliran pesan atau informasi antar objek maupun hubungan antar objek. Biasanya interaction ini dilengkapi juga dengan teks bernama *operation signature* yang tersusun dari nama operasi, parameter yang dikirim dan tipe parameter yang dikembalikan (Nugroho, 2004).



Gambar 2.4 Notasi Interaction Sumber : ( Nugroho, 2010)

1. *Interface*

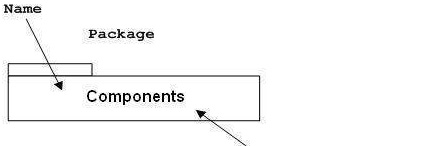
*Interface* merupakan kumpulan operasi tanpa implementasi dari suatu class. Implementasi operasi dalam interface dijabarkan dalam operasi dalam class. Oleh karena itu keberadaan interface selalu disertai oleh class yang mengimplementasikan operasinya (Nugroho, 2007). Interface ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam objek.



Gambar 2.5 Notasi Interface Sumber : (Nugroho, 2010)

1. *Package*

Package adalah container atau wadah konseptual yang digunakan untuk mengelompokkan elemen-elemen dari suatu sistem yang sedang dibangun, sehingga bisa dibuat model yang lebih sederhana. Tujuannya adalah untuk mempermudah penglihatan (*visibility*) dari suatu model yang sedang dibangun.



Gambar 2.6 Notasi Package Sumber: (Nugroho, 2010)

1. *Note*

*Note* dibangun untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. Note ini bisa ditempelkan ke semua elemen notasi yang lain.

Gambar 2.7 Notasi Note Sumber: (Nugroho, 2010)

1. *Dependency*

Merupakan relasi yang menunjukan bahwa perubahan pada suatu elemen memberikan pengaruh pada elemen yang lain. Elemen yang ada di bagian tanda panah adalah elemen yang tergantung pada elemen yang ada di bagian tanpa ada tanda panah.

Terdapat dua *stereotype* dari *dependency,* yaitu *include* dan *extend*. *Include* menunjukan bahwa suatu bagian dari elemen (yang ada di garis tanpa panah) memicu eksekusi bagian dari elemen yang lain (yang ada di garis dengan panah), misalnya untuk notasi A → B operasi yang ada di class A memicu dieksekusinya operasi yang berada di class B.

Extend menunjukan bahwa suatu bagian dari elemen di garis tanpa panah bisa disiapkan ke dalam elemen yang ada di gari dengan panah, misalnya untuk notasi A→B suatu fungsi dari usecase A bisa disisipkan ke dalam usecase B atau dengan kata lain A optional untuk B. Ke dua *stereotype* ini di representasikan denganmenambahkan text include atau extend di notasi dependency.



Gambar 2.8 Notasi Dependency (Nugroho, 2010)

1. *Association*

*Association* menggambarkan navigasi antar class (*navigation*), berapa banyak objek lain yang bisa berhubungan dengan satu objek (multiplicity antar class) dan apakah suatu class menjadi bagian dari class lainnya (aggregation).

*Navigation* di lambangkan dengan penambahan tanda panah di akhir garis*. Bidirectional navigation* menunjukan bahwa dengan mengetahui salah satu class bisa didapatkan dari informasi lainnya. Sementara dengan unidirectional navigation hanya dengan mengetahui class di ujung garis association tanpa panah kita bisa mendapatkan informasi dari class di ujung dengan panah, tetapi tidak sebaliknya.

Gambar 2.9 Notasi Association Sumber: (Nugroho, 2010)

Pada penelitian kali ini penulis menggunakan Visual Paradigm untuk merancang aplikasi. Visual Paradigm adalah perangkat lunak yang digunakan untuk perancangan aplikasi dengan tools UML.