**BAB II**

**LANDASAN TEORI**

1. **Tinjauan Pustaka**

Dalam penelitian ini penulis menggunakan jurnal ilmiah sebagai kajian penelitian yang relevan. Peneliti melakukan penelusuran terhadap penelitian yang terdahulu dengan judul aplikasi mobile pembelajaran yang diangkat oleh penulis.

Berikut ini adalah hasil penelitian yang relevan yang dijadikan sebagai acuan bagi penulis untuk mendukung penulisan penelitian ini:

1. Kajian Penelitian Yang Relevan

Untuk menghindari duplikasi dari pihak lain, penulis melakukan penelusuran terhadap penelitian-penelitian terdahulu. Dari hasil penelusuran penelitian terdahulu, diperoleh beberapa masalah yang berkaitan dengan masalah yang akan diteliti, yaitu:

1. Penelitian yang dilakukan oleh Himawan Ariefianto, Mohammad Dani dan Asep Nugraha, pada tahun 2011 dengan judul Perancangan Dan Implementasi Aplikasi Pemesanan Makanan Dan Minuman Berbasis *Client Server* Dengan *Platform Android*. Penelitian ini membahas tentang perancangan dan implementasi pemesanan makanan dan minuman melalui *android* yang terhubung ke aplikasi *web* di komputer *server* melalui jaringan *wireless*. Hasil dari penelitian ini adalah aplikasi *android* yang mampu memesan makanan dan minuman, pesanan langsung diberitahukan ke bagian dapur dan kasir. Untuk lebih jelas lagi, berikut perbedaan dan persamaan penelitian pada Tabel 1:

Tabel 1. Penelitian Himawan Ariefianto

|  |  |  |
| --- | --- | --- |
| Perbedaan dan Persamaan | Himawan Ariefianto, dkk | Penelitian Penulis |
| 1 | 2 | 3 |
| Topik  Penelitian | Perancangan Dan Implementasi Aplikasi Pemesanan Makanan Dan Minuman Berbasis *Client Server* Dengan *Platform Android* | Aplikasi pengelolaan matakuliah teknik informatika berbasis *Client* *server* pada STMIK Palangka Raya |
| Metode Pengumpulan Data | - | *Observasi* dan kepustakaan |
| Bahasa  Pemrograman | *Java dan PHP* | *Ionic Framework, PHP* dan *HTML* |
| *Database* | *MySQL* | *MySQL* |
| Lokasi  Penelitian | Waroeng Steak And Shake | STMIK Palangka Raya |
| 1 | 2 | 3 |
| Pemodelan  Sistem | UML | UML |
| Metode Pengembangan | - | *Waterfall* |

1. Penelitian berikutnya dilakukan oleh Anggia Kususmawaty, Universitas Gunadarma, pada tahun 2012 dengan judul Aplikasi Pemesanan Makanan Pada Restoran Berbasis *Android* Dan *Php* Menggunakan Protokol *Json*. Penelitian ini membahas tentang pembuatan Aplikasi Pemesanan Makanan untuk mengatasi masalah yang timbul jika suatu restoran memiliki ruang yang besar atau pengunjung yang banyak. Oleh karena itu dibutuhkan sistem secara *online* untuk dapat mempercepat proses penyampaian daftar pesanan pengunjung. Hasil dari penelitian ini adalah Program aplikasi *mobile* Pemesanan Makanan pada Restoran untuk *handphone* dengan sistem operasi Android yang menggunakan bahasa pemrograman Java telah berhasil dibuat. Aplikasi Pemesanan Makanan pada Restoran ini dapat memudahkan pelayan dalam mencatat pesanan pelanggan pada setiap Restoran. Pelayan tidak perlu membawa selembar kertas dan alat tulis untuk mencatat pesanan pelanggan, tetapi menggunakan media  
   *handphone* untuk mencatat pesanan pelanggan. Untuk lebih jelas lagi, berikut perbedaan dan persamaan penelitian pada Tabel 2:

Tabel 2. Penelitian Anggia Kusumawaty

|  |  |  |
| --- | --- | --- |
| Perbedaan dan Persamaan | Anggia Kususmawaty | Penelitian Penulis |
| 1 | 2 | 3 |
| Topik  Penelitian | Aplikasi Pemesanan Makanan Pada Restoran Berbasis *Android* Dan *Php* Menggunakan Protokol *Json* | Aplikasi pengelolaan matakuliah teknik informatika berbasis *Client* *server* pada STMIK Palangka Raya |
| Metode Pengumpulan Data | - | *Observasi* dan kepustakaan |
| Bahasa  Pemrograman | *Java, CodeIgniter Framework, xml, json* dan *PHP* | *Ionic Framework, PHP* dan *HTML* |
| *Database* | *MySQL* | *MySQL* |
| Lokasi Penelitian | Restoran | STMIK Palangka Raya |
| Pemodelan  Sistem | UML | UML |
| Metode Pengembangan | - | *Waterfall* |

1. Penelitian berikutnya dilakukan oleh Albertus Radityo Hadi dan Maria Irmina Prasetiyowati, Universitas Multimedia Nusantara, pada tahun 2012 dengan judul Rancang Bangun Aplikasi Transport Booking Berbasis *Android* dengan Teknik Enkripsi *Advanced Encryption Standard*. Penelitian ini membahas tentang aplikasi pemesanan kendaraan berbasis *Android* untuk mempercepat proses pemesanan kendaraan pada PT Indodev Niaga Internet dan untuk menjaga keamanan data saat pengiriman digunakan teknik enkripsi AES. Hasil dari penelitian ini adalah aplikasi *android* yang digunakan oleh karyawan PT Indodev Niaga Internet. Aplikasi ini membantu karyawan untuk dapat memesan kendaraan kapan pun dan di mana pun selama terkoneksi dengan internet. Untuk lebih jelas lagi, berikut perbedaan dan persamaan penelitian pada Tabel 3:

Tabel 3. Penelitian Albertus dan Maria

|  |  |  |
| --- | --- | --- |
| Perbedaan dan Persamaan | Albertus Radityo Hadi dan Maria Irmina Prasetiyowati | Penelitian Penulis |
| 1 | 2 | 3 |
| Topik  Penelitian | Rancang Bangun Aplikasi Transport Booking Berbasis *Android* dengan Teknik Enkripsi Advanced Encryption Standard | Aplikasi pengelolaan matakuliah teknik informatika berbasis *Client* *server* pada STMIK Palangka Raya |
| Metode Pengumpulan Data | - | *Observasi* dan kepustakaan |
| Bahasa  Pemrograman | *Java dan Cold Fusion* | *Ionic Framework, PHP* dan *HTML* |
| *Database* | *SQL Server* | *MySQL* |
| Lokasi Penelitian | PT Indodev Niaga Internet | STMIK Palangka Raya |
| Pemodelan Sistem | DFD | UML |
| 1 | 2 | 3 |
| Metode Pengembangan | - | *Waterfall* |

1. Penelitian berikutnya dilakukan oleh Imam Chairul Arifin dan Sutariyani, STMIK AUB Surakarta, pada tahun 2014 dengan judul Aplikasi Pemesanan Menu Makanan Berbasis *Client Server* Smartphone *Android* Dan Komputer. Penelitian ini membahas tentang aplikasi yang memudahkan proses pemesanan. pelayanan restoran cepat saji yang mempunyai lahan cukup luas mengalami permasalan dalam hal mengantarkan daftar pesanan dari meja pelanggan menuju dapur untuk dimasak dan sebaliknya saat makanan sudah selesai dimasak dan diantarkan ke meja pelanggan. Bahkan ada pelanggan yang memesan makanan tambahan pada saat makanan sudah diantar dimeja dan tidak ada pelayan yang melayaninya. Sehingga banyak pelanggan yang keinginannya tidak terpenuhi, yang mengakibatkan banyak pelanggan yang tidak jadi memesan dan ini dapat mengurangi omset yang seharusnya dapat diperoleh pengelola restoran. Hasil dari penelitian ini adalah Dengan dibuatnya Aplikasi Pemesanan Menu Makanan Berbasis *Client Server* Pada Smartphone Android Dan Komputer Berbasis *Windows* maka dapat mempercepat proses pemesanan dan pembayaran. Pihak pengelolapun dapat mengurangi jumlah pengeluaran karena mengurangi jumlah *waiters* yang tersedia karena member dari *restaurant* dapat memesan menu yang disediakan dengan mandiri menggunakan *smartphone.* Untuk lebih jelas lagi, berikut perbedaan dan persamaan penelitian pada Tabel 4:

Tabel 4. Penelitian Imam Chairul Ariffin dan Sutariyani

|  |  |  |
| --- | --- | --- |
| Perbedaan dan Persamaan | Imam Chairul Arifin dan Sutariyani | Penelitian Penulis |
| 1 | 2 | 3 |
| Topik  Penelitian | Aplikasi Pemesanan Menu Makanan Berbasis Client Server Smartphone Android Dan Komputer | Aplikasi pengelolaan matakuliah Teknik Informatika berbasis *Client* *server* pada STMIK Palangka Raya |
| Metode Pengumpulan Data | - | *Observasi* dan kepustakaan |
| Bahasa  Pemrograman | *Java, Eclipse Juno,* dan *PHP* | *Ionic Framework, PHP* dan *HTML* |
| *Database* | *SQL Server* | *MySQL* |
| Lokasi  Penelitian | PT Indodev Niaga Internet | STMIK Palangka Raya |
| 1 | 2 | 3 |
| Pemodelan  Sistem | DFD | UML |
| Metode Pengembangan | - | *Waterfall* |

1. Penelitian berikutnya dilakukan oleh juniwati, STMIK Palangka raya, pada tahun 2015 dengan judul Implementasi Pengolah Data Berbasis *Client* *Server* Pada SMAN-3 Palangka Raya. Penelitian ini membahas tentang pengolahan data siswa, data guru maupun data nilai pada SMAN-3 Palangka Raya yang saling berhubungan dalam hal penginputan data kelas dimana seorang guru bertindak sebagai walikelas yang membimbing atau menilai seorang siswa. Untuk lebih jelas lagi, berikut perbedaan dan persamaan penelitian pada Tabel 5 :

Tabel 5. Penelitain Juniwati

|  |  |  |
| --- | --- | --- |
| Perbedaan dan Persamaan | Juniwati | Penelitian Penulis |
| 1 | 2 | 3 |
| Topik  Penelitian | Implementasi Pengolahan Data Berbasis *Client Server* Pada SMAN-3 Palangka Raya | Aplikasi pengelolaan matakuliah teknik informatika berbasis *Client* *server* pada STMIK Palangka Raya |
| Metode Pengumpulan Data | *Observasi* dan kepustakaan | *Observasi* dan kepustakaan |
| Bahasa  Pemrograman | *Microsoft Visual Basic 6.0* dan *Crytal Report 8.5* | *Ionic Framework, PHP* dan *HTML* |
| *Database* | *MySQL* | *MySQL* |
| Lokasi  Penelitian | SMAN-3 Palangka Raya | STMIK Palangka Raya |
| Pemodelan  Sistem | DFD | UML |
| 1 | 2 | 3 |
| Metode Pengembangan | *Waterfall* | *Waterfall* |

1. **Kajian Teori**
2. Definisi
3. *Framework*

Menurut pressman (2005:282) framework adalah kerangka kode yang dapat disempurnakan dengan *class*es yang spesifik atau dengan fungsi yang telah dirancang untuk mengatasi masalah yang dihadapi.

*Framework* adalah kumpulan kode-kode program yang disimpan pada *File-file* yang berbeda yang dapat menyederhanakan operasi yang berulang-ulang. Kelebihan dengan adanya *Framework* adalah dapat melakukan pengembagan aplikasi menjadi seragam, dimana pengembang dituntut untuk mengikuti alur kerja yang ditetapkan oleh *Framework*. Dalam artian setiap pengembang harus mempunyai metode yang sama dalam menyelesaikan aplikasi tersebut.

1. *Website*

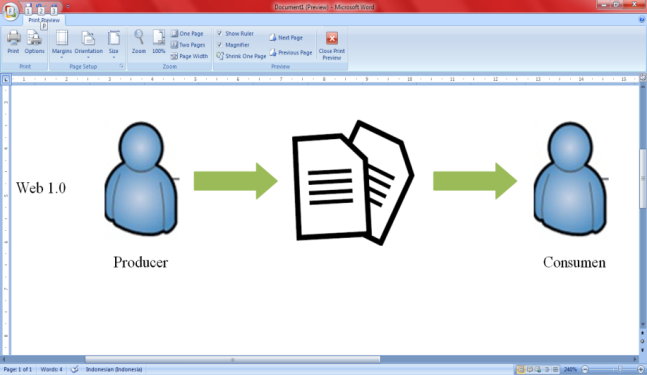
Menurut Rahmat, Hidayat (2010:2) *website* atau situs dapat diartikan sebagai kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman.

Hubungan antara satu halaman dengan halaman *web* yang lainnya disebut *Hyperlink*, sedangkan teks yang dijadikan media penghubung disebut *Hypertext*.

Seiring dengan perkembangan teknologi informasi yang begitu cepat, *website* juga mengalami perkembangan yang sangat berarti. Perkembangan *web* sampai saat ini paling baru yaitu:

1. *Web* 1.0

Versi yang pertama dari *web* menyajikan informasi secara statis atau dikenal sebagai “*read-only web*” sehingga seorang pembaca dimungkinkan mencari informasi yang mereka inginkan. Interaksi dengan pembaca masih sangat sedikit sekali, kalaupun ada maka pihak pembuat/programmer akan sering dicari untuk mengubah keseluruhan atau sebagiann konten *web*.

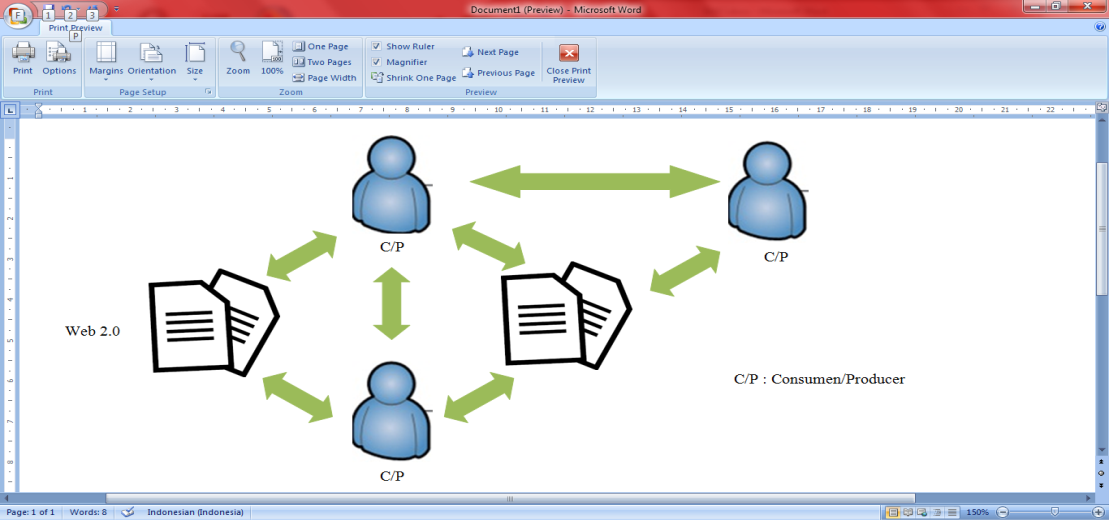


Gambar 1. web 1.0

(Sumber : Rahmat Hidayat., *Cara Praktis Membangun Website Gratis*)

1. *Web* 2.0

Versi yang kedua ini menandakan adanya hubungan interaktif antara pembaca dan pembuat *web* sehingga pengguna dapat memberikan kontribusi terhadap tampilan/konten *website*. Istilah *web* 2.0 ini pertama kali diperkenalkan oleh O’Reilly Media pada tahun 2004. Didukung dengan kemajuan inovasi antarmuka (*interface*) yang menggabungkan antara teknologi HTML, *JavaScript*, CSS, dan XML. Contoh web 2.0 ini antara lain *web* jejaring sosial seperti *facebook, twitter, youtube, wikipedia,* dan  *flickr* yang menekankan kolaborasi *online* dan berbagai antar pengguna.



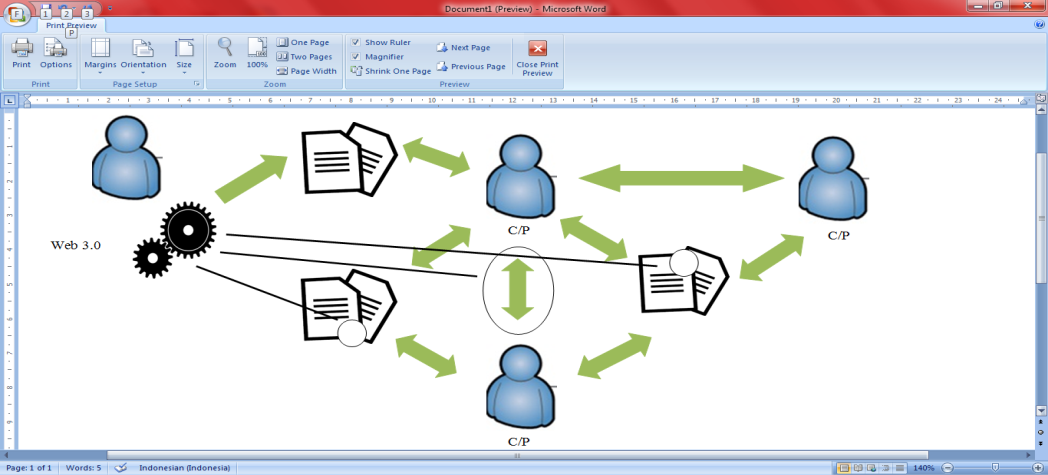
Gambar 2. web 2.0

(Sumber : Rahmat Hidayat., *Cara Praktis Membangun Website Gratis*)

1. *Web* 3.0

Versi yang ketiga ini dikenal sebagai *semantic-web* yang mempunyai beberapa ciri/karakteristik, antara lain :

1. Kecerdasan buatan (*artificial intelligence)* yang lebih maju sehingga *web* akan dapat memahami kata perkata yang diketikkan oleh pengguna.
2. *Abiquitous connectivity* yang memungkinkan *website* dapat diakses pada perangkat, bahkan di prediksi tidak hanya perangkat ponsel saja melainkan dapat diakses melalui jam tangan, televisi dan lain-lain.
3. Tampilan dari dua dimensi ketiga dimensi



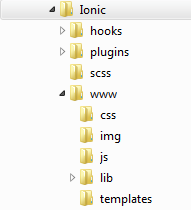
Gambar 3. web 3.0

(Sumber : Rahmat Hidayat)

1. *Ionic*

Menurut Jeremy wilken (2015 : 1) bahwa dengan *Ionic*, Anda akan mampu membangun aplikasi *mobile* *hybrid* yang terlihat dan terasa seperti aplikasi *mobile* asli. Sebuah aplikasi *hybrid* adalah jenis aplikasi *mobile* yang menggunakan jendela *browser* untuk menampilkan antarmuka.

*Ionic* adalah kombinasi dari alat dan utilitas yang memungkinkan pengembang untuk segera membangun aplikasi *mobile* *hybrid* menggunakan teknologi yang sama digunakan untuk membangun *website* dan aplikasi *web*, terutama *HTML*, *CSS* (*Cascading Style Sheets* ), dan *JavaScript*. *Ionic* bekerja dengan melekatkan sebuah aplikasi *web* dalam aplikasi asli dengan menggunakan *Cordova* . ia dirancang bersama *angular* agar dapat membangun sebuah aplikasi *web* untuk lingkangan *mobile*. dan termasuk dukungan untuk fitur ponsel seperti kontrol antarmuka pengguna dan menanggapi input sentuhan.



Gambar Struktur *folder Ionic*

Adapun cara instalasi *ionic* adalah sebagai berikut :

1. Instalasi *Tools Ionic*

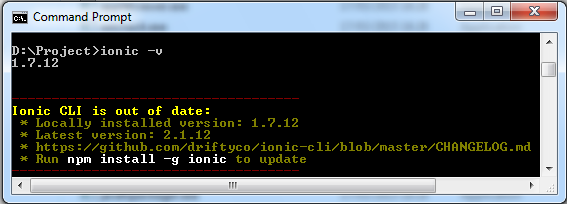
Untuk instalasi ionic sendiri berbeda beda menurut sistem operasi yang digunakan. Namun saya disini menggunakan sistem operasi *windows.* Berikut beberapa *tool* yang harus di persiapkan terlebih dahulu :

1. *install NodeJS* yang pertama harus kita lakukan adalah *install* *nodeJS*. *NodeJS* sediri berfungsi untuk mengeksekusi aplikasi *server*. salah satu fungsinya yaitu kita bisa mengakses *npm*, perintah yang akan digunakan untuk instalasi *ionic*. langkah-langkah instalasinya sama dengan proses *install* aplikasi lain di *windows*.
2. *Install* *Java JDK*, sama halnya dengan instalasi node js, fungsi java disini adalah kita memerlukannya ketika mengkompile aplikasi *ionic* yang kita buat
3. *apache ant*, setelah *apache ant* di download, ekstrak terlebih dahulu dan letakkan pada direktori yang diinginkan sebenarnya *apache ant* ini tidak di *install* seperti aplikasi lainnya, dia hanya perlu di ekstrak dan di *path*, berfungsi ketika kita mamanggil lewat *cmd* .
4. *Intall android SDK*, *android sdk* diperlukan untuk menjalankan aplikasi dan mem-*build* aplikasi android yang menggunakan ionic.
5. Mengatur *Path Environment*

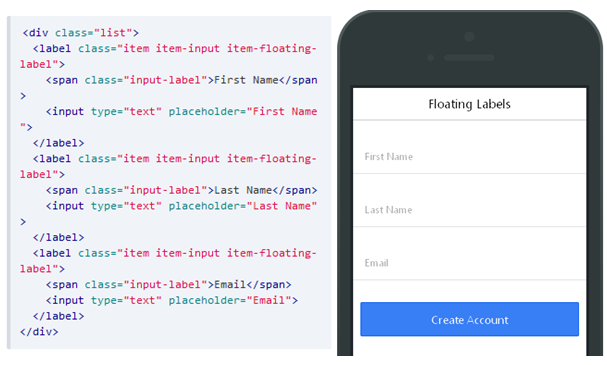
Sebelum ke tahap instalasi *ionic*, kita harus mengatur *path environment* untuk tiap – tiap *tool* yang sudah di *install*. Berikut langkah – langkah mengatur *path environment* untuk tiap – tiap *tool*:

1. *path NodeJS,* klik kanan pada *my computer* pilih *properties* pilih advanced System setting-environment variables. Klik pada variabel PATH pilih edit dan tambahkan C:\users\mhs\AppData\Roaming\npm
2. *path* *Java JDK*, klik kanan pada *my computer* pilih *properties* pilih advanced System setting-environment variables. Klik pada variabel PATH pilih edit dan tambahkan C:\Program Files\Java\jdk\bin
3. *path apache ant*, klik kanan pada *my computer* pilih *properties* pilih advanced System setting-environment variables. Klik pada variabel PATH pilih edit dan tambahkan C:\apache ant\bin.
4. *path android SDK*, klik kanan pada *my computer* pilih *properties* pilih advanced System setting-environment variables. Klik pada variabel PATH pilih edit dan tambahkan C:\Program Files\Android\android-sdk.
5. Instalasi *Cordova* dan *Ionic*

Inilah tahapan terakhir untuk *install* *ionic*. Setelah semua *tool* di atas sudah di siapkan, tahap terakhir adalah melakukan instalasi *Cordova* dan *Ionic*. Buka *command prompt atau Node.js command prompt ,*ketikkan perintah *npm install**–g**cordova ionic.* Ini akan membutuhkan waktu beberapa saat, waktu yang dibutuhkanpun tergantung pada kecepatan koneksi internet. Jika intalasi sudah selesai maka buka *command prompt* dan ketikkan perintah *ionic –v*, hasilnya akan seperti gambar di bawah ini :

Gambar . *Ionic* pada *command prompt*

*Ionic* sudah siap digunakan untuk membuat aplikasi *android*, berikut adalah gambar contoh sintak – sintak untuk membuat form pada *ionic*,



Gambar 6. contoh sintak - sintak *ionic*

1. *BlackBox*

Terdapat dua metode pengujian perangkat lunak yang umum digunakan, yaitu metode *Black Box* dan *White Box*. Pengujian dengan *Black Box* merupakan pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Sebuah perangkat lunak yang diuji menggunakan metode *Black Box* dikatakan berhasil jika fungsi-fungsi yang ada telah memenuhi spesifikasi kebutuhan yang telah dibuat sebelumnya. Sedangkan metode *White Box* adalah menguji perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi kebutuhan pada sistem sedangkan metode pengujian menggunakan *White Box* dilakukan dengan memeriksa logika dari kode-kode program.

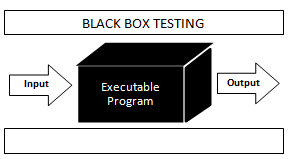
Di tahapan pengujian sistem ini penulis menggunakan metode pengujian *Black Box testing* karena hanya menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program.

Menurut Pressman (2010:597) pengujian kotak hitam(*Black Box Testing*) disebut juga pengujian perilaku, berfokus pada persyaratan fungsionalperangkat lunak. Artinya, teknik pengujian kotak hitam memungkinkan untuk membuat beberapa kumpulan kondisi masukan yang sepenuhnya akan melakukan semua kebutuhan fungsional untuk program.

Pengujian *Black Box* dapat dilakukan untuk memverifikasi apakah perilaku komponen memang seperti yang diharapkan tanpa harus mempertimbangkan bagaimana perilaku tersebut secara internal di implementasikan. Pengujian pada jenis ini melihat pada apakah komponen tertentu menghasilkan keluaran (*output*) seperti yang diharapkan saat komponen itu diberi asupan tertentu serta berjalan dalam *state* tertentu. Pengujian kotak hitam berupaya untuk menemukan kesalahan dalam kategori berikut:

* 1. Fungsi yang salah atau hilang
  2. Kesalahan antarmuka
  3. Kesalahan dalam struktur data atau akses basis data eksternal
  4. Kesalahan perilaku atau kinerja
  5. Kesalahan inisialisasi dan penghentian.

Metode pengujian *black box* memfokuskan pada keperluan fungsional dari *software*. Karena untuk membuat himpunan kondisi *input* yang akan melatih seluruh syarat-syarat fungsional suatu program. *Black box* dapat digambarkan sebagai berikut :



Gambar 7. Black box testing

Di sini terdapat adanya suatu “relasi transformasi” antara *input* dan *output*, yang seakan-akan dialihkan melalui elemen-elemen sistem yang tidak dikenal dan relasi-relasi mereka yang tidak dikenal. Jadi orang hanya memperhatikan *input* dan *output* sistem tersebut. Oleh sebab itu, orang menggunakan istilah “*Black Box*”.

1. Skala Likert

Menurut Sudaryono,dkk (2010:110), skala *likert* ialah skala digunakan untuk mengukur sikap, pendapat,dan persepsi seseorang atau sekelompok tentang kejadian atau gejala sosial. Dalam penelitian gejala sosial, peneliti telah menetapkan secara spesifik skalanya dan selanjutnya disebut variabel peneliti.

Dengan menggunakan skala likert, variabel yang akan diukur dijabarkan menjadi dimensi, lalu dimensi dijabarkan menjadi subvariabel dan subvariabel dijabarkan lagi menjadi indikator yang dapat diukur. Akhirnya indikator yang terukur dapat menjadi titik tolak untuk membuat item instrumen berupa pernyataan atau pertanyaan yang perlu dijawab oleh responden.

Setiap jawaban dihubungkan dengan bentuk pernyataan atau dukungan sikap yang diungkapkan dengan kata-kata sebagai berikut:

1. Sangat baik (SB) = 5
2. Baik (B) = 4
3. Cukup baik (CB) = 3
4. Kurang baik (KB) = 2
5. Tidak baik (TB) = 1
6. *Android*

Menurut Safaat (2012:1) *Android* adalah sebuah sistem operasi pada handphone yang bersifat terbuka dan berbasis pada sistem operasi *Linux.* *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi sendiri yang akan digunakan untuk bermacam piranti bergerak.

*Android* merupakan sistem operasi yang berbasis *Linux* untuk telepon seluler seperti telepon pintar dan komputer *tablet.* *Android* menyediakan *plaform* terbuka bagi para pengembang untuk menciptakan aplikasi sendiri untuk digunakan oleh bermacam piranti bergerak.



Gambar . Logo Android

Awalnya *Android* dikembangkan oleh *Android*, Inc., dengan dukungan finansial dari *Google*, yang kemudian membelinya pada tahun 2005 Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya *Open Handset Alliance*, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Ponsel *Android* pertama mulai dijual pada bulan Oktober 2008.

Pada Juli 2005, *Google* bekerjasama dengan *Android* Inc., perusahaan yang berada di Polo Alto, California Amerika Serikat. Para pendiri *Android* Inc. bekerja pada Google, diantaranya Andy Rubin, Rich Miner, Nick Sears dan Chris White. Saat itu banyak yang menganggap fungsi *Android* hanya sebagai perangkat lunak pada telepon *seluler.* September 2007 Google mengajukan hak paten aplikasi telepon *seluler.* Telepon pertama yang memakai sistem operasi *Android* adalah *HTC Dream*, yang dirilis pada 22 Oktober 2008.

1. Konsep Bahasa Pemrograman
2. *Cascading Style Sheet* (CSS)

*Cascading Style Sheet* atau CSS adalah salah satu bahasa desain *web* yang mengontrol format tampilan sebuah halaman *web* yang ditulis dengan menggunakan penanda (markup language). Biasanya CSS digunakan untuk mendesain sebuah halaman HTML dan XHTML.

1. *HyperText Markup Language* (HTML)

Menurut Arief (2011:23) HTML atau *HyperText Markup Language* merupakan salah satu format yang digunakan dalam pembuatan dokumen dan aplikasi yang berjalan di halaman *web.* Dokumen ini dikenal sebagai *web page.*Dokumen HTMLmerupakan dokumen yang disajikan pada *web browser*.

*HyperText Markup Language* adalah bahasa pengkodean untuk menghasilkan dokumen-dokumen *hypertext* untuk digunakan di *World Wide Web*. HTML terlihat seperti bahasa pengkodean lama, yang diawali dan diakhiri dengan kode-kode html dimana menunjukkan bagaimana keluarannya pada saat dijalankan pada sebuah *web browser*.

Tabel 6. Sintaks HTML5

<!DOCTYPE html>

<html>

<head>

<meta charset=”utf-8”>

<title>Untitled Document</title>

</head>

<body>

</body>

</html>

Sintak HTML selalu dibuka dengan kode *<html>* dan diakhiri dengan *</html>.*

1. PHP *Hypertext Preprocessor* (PHP)

Menurut Aditya N. A (2011:1) PHP atau *PHP Hypertext Preprocessor* adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk memprogram situs *Web* dinamis. PHP dapat digunakan untuk membangun sebuah CMS.

Menurut Sukarno (2006:10) PHP memiliki beberapa kelebihan dari bahasa pemrograman lainnya, yaitu :

1. PHP merupakan sebuah bahasa *Script* yang tidak melakukan sebuah kompilasi dalam penggunaannya. Tidak seperti halnya bahasa pemrograman aplikasi seperti *Visual Basic* dan sebagainya.
2. PHP dapat berjalan pada *Web Server* yang dirilis oleh *Microsoft*, seperti IIS atau PWS juga pada *Apache* yang bersifat *Open Source*.
3. Sifatnya *Open Source*, maka perubahan dan perkembangan *Interpreter* pada PHP lebih cepat dan mudah.
4. Dari segi pemahaman, PHP memiliki *Referensi* yang begitu banyak sehingga sangat mudah untuk dipahami.
5. PHP dapat berjalan pada tiga *Operating System,* yaitu: *Linux*, *Unix*, dan *Windows*, dan juga dapat dijalankan secara *Runtime* pada suatu *Console.*

Untuk mengenal sintak dalam PHP, tentu saja kita juga harus sudah mengenal algoritma dalam pemrograman, karena pada dasarnya semua bahasa pemrograman menggunakan algoritma yang sama. Mulai dari pengenalan *variable*, proses pengulangan (*looping*), dan menghasilkan keluaran/*output*. Sintak PHP selalu dimulai dengan *<?* Atau *<?php* dan diakhiri dengan *?>*. dan di dalam Sintak *php* juga dapat kita sisipkan kode *html* dengan menambahkan perintah *echo* “ ”:. Di dalam PHP, setiap *variable* diberi tanda *dollar* (*$*).

1. *JavaScript*

Menurut Wahana Komputer (2010:1) *JavaScript* merupakan bahasa yang terbentuk kumpulan skrip yang berfungsi untuk memberikan tampilan yang tampak lebih interaktif pada dokumen *web*.

*JavaScript* adalah bahasa pemrograman untuk memberikan kemampuan tambahan kedalam bahasa pemrograman HTML (*Hypertext Markup Languae*) dengan menginjinkan pengeksekusian perintah-perintah pada sisi *client*, dan bukan sisi *server* dokumen *web*.

Pada hakikatnya, bahasa pemrograman *JavaScript* berisi skrip yang pemasangannya terselip disebuah dokumen HTML. Sehingga bahasa *JavaScript* ini tidaklah memerlukan sebuah kompilator atau penerjemah khusus untuk mengeksekusinya. Hal tersebut juga bergantung pada *navigator* yang terdapat disetiap *browser*.

*JavaScript* dan *java* memamg hampir sama, tidak jarang orang salah konsepsi dalama pengertian kedua bahasa tersebut dan beranggapan bahsa pemrograman *java* merupakan bagian dari pemrograman *JavaScript*. Oleh karena itu, penulis akan menjelaskan perbedaaan kedua bahasa tersebut. Beberapa perbandingan mendasar antara *JavaScript* dan *Java* dapat dilihat dari tabel berikut ini :

Tabel 7. Perbedaan JavaScript dengan Java

|  |  |
| --- | --- |
| ***JavaScript*** | ***Java*** |
| 1 | 2 |
| Bahasa langsung diintepretasikan oleh *browser* (tidak ada kompilasi) | Bahasa dikompilasi terlebih dulu sebelum dijalankan *client* |
| 1 | 2 |
| Kode tidak terdiri dari kelas-kelas dan terintegrasi dengan HTML | Terdiri dari kelas-kelas yang terpisah dari dokumen HTML, dan diakses dari dokumen HTML |
| Kode program dapat langsung diakses | Kode program harus dideklarasikan terlebih dahulu |
| Hubungan dinamis | Hubungan statis (referensi objek harus dikompilasi) |

Untuk menerapkan program *JavaScript* dalam *web*, ada beberapa yang harus diketahui oleh seorang perancang *web*, yaitu :

1. Seorang perancang harus mengetahui cara menggunakan HTML dan mengedit dokumen HTML.
2. Seorang perancang harus menggunakan *browser* yang sudah mendukung pemrograman *JavaScript*, misalnya *Internet Explorer* versi 4 atau versi diatasnya, *Netscape* versi 3 atau versi diatasnya.
3. *Twitter Bootstrap*

Menurut Ridha (2003-2007:4) *Twitter Bootstrap* adalah sebuah alat bantu membuat tampilan halaman *website* yang dapat mempercepat pekerjaan seseorang pengembang *website* ataupun pendesain halaman *website*. Sesuai namanya, *website* yang dibuat dengan alat bantu ini memiliki tampilan halaman yang sama atau mirip dengan tampilan halaman *web* sesuai dengan kebutuhan.

*Bootstrap* awalnya dibangun oleh Mark Otto dan hanya ditunjukan untuk penggunaan internat di *Twitter*. Kemudian *Bootstrap* diluncurkan sebagai perangkat lunak bebas untuk umum. Dalam perkembangan awal *Bootstrap, Fitur Responsive* belum ada, kemudian ditambahkan dalam versi 2.0 sebagai respon terhadap meningkatnya permintaan untuk *Responsive web Design* (*Support* segala macam layar dan *Device*). *Twitter Bootstrap* dibangun dengan teknologi HTML, dan CSS yang dapat membuat *layout* halaman *website, table, tombol, form, navigasi*, dan komponen lainnya dalam sebuah *website* hanya dengan memangil fungsi CSS (*Class*) dalam berkas HTML yang telah didefinisikan. Keuntungan menggunakan *Twitter Bootstrap* yaitu:

1. Memudahkan dalam mendesain *website*
2. *Responsive* (*Support* segala macam layar dan *device*)
3. Dokumentasi lengkap
4. Elegan.
5. Konsep Arsitektur Sistem

Arsitektur sistem yang digunnakan dalam penelitian ini adalah Arsitektur Sistem *Client Server.*

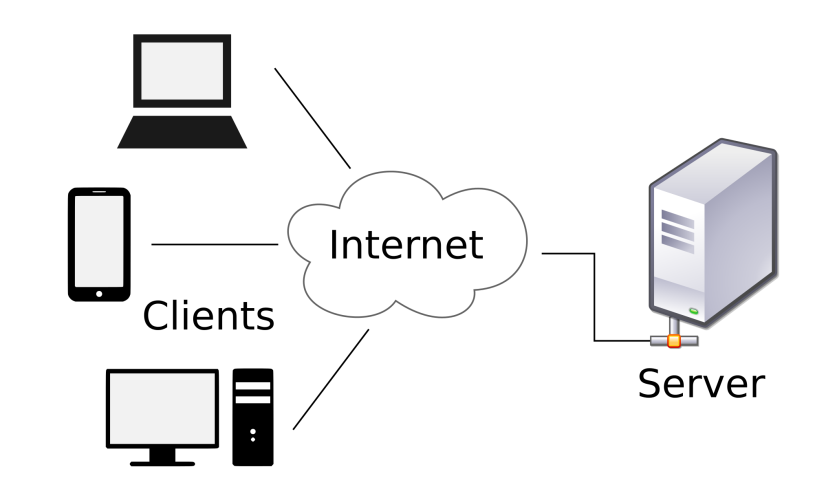
Menurut Handoko (2001:11) *Client server* adalah arsitektur jaringan yang memisahkan *client* (biasanya aplikasi yang menggunakan GUI) dengan *server*. Masisng-masing *client* dapat meminta data atau informasi dari *server.* Jadi, jaringan *Client Server* menghubungkan komputer *server* dengan komputer *client* atau *workstation*. Komputer *server* merupakan komputer yang menyediakan fasilitas bagi komputer-komputer *client* atau *workstation* yang terhubung dalam jaringan. Sedangkan komputer *client* merupakan komputer yang menggunakan fasilitas yang disediakan oleh komputer *server*.

Adapun Kelebihan arsitektur *Client Server* yaitu :

1. Lebih aman- Semua data dapat dibackup pada satu lokasi sentral
2. Kecepatan akses lebih tinggi karena penyediaan fasilitas jaringan dan pengelolaannya dilakukan secara khusus oleh satu komputer (*server*) yang tidak dibebani dengan tugas lain sebagai *workstation*

Sedangkan Kekurangan arsitektur *Client Server* yaitu :

1. Membutuhkan *administrator* yang handal
2. Jika *server* mati maka semua *client* akan mati juga



Gambar . Arsitektur *Client Server*

1. Konsep Pemodelan Sistem

Pada penelitian ini, pemodelan sistem yang akan penulis gunakan adalah *Unified Modelling Language* (UML).

Menurut Pressman (2010:987) *Unifield Modeling Language* (UML) adalah “bahasa standar untuk penulisan cetak biru perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menentukan, mengonstruksi, dan mendokumentasikan artifak-artifak suatu sistem *software-intensive*”.

*Unifield Modeling Language* atauUML ada beberapa diagram yaitu sebagai berikut :

1. Use Case Diagram

Menurut Nugroho (2016:6) Use Case Diagram Menjelaskan manfaat sistem menurut pandangan orang yang berada diluar sistem (actor). Diagram ini menunjukan fungsionalitas suatu sistem atau kelas dan bagaimana sistem itu berinteraksi dengan dunia luar.

*Diagram Use Case* menggambarkan apa saja aktifitas yang dilakukan olehsuatu sistem dari sudut pandang pengamatan luar. yang menjadi persoalan itu apa yangdilakukan bukan bagaimana melakukannya.

*Diagram Use Case* dekat kaitannya dengan kejadian-kejadian. Kejadian(scenario) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem. Simbol-simbol yang ada pada *use case diagram* ditunjukkan tabel 8 berikut :

Tabel 8. Simbol-simbol *use case diagram*

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Keterangan** |
| 1 | 2 | 3 |
|  | *Actor* | Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *Use Case*. |
|  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya  . |
| 1 | 2 | 3 |
|  | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi prilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
|  | *Include* | Menspesifikasikan bahwa *Use Case* sumber secara *eksplisit* |
|  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas prilaku dari *use case* sumber pada suatu titik yang diberikan. |
|  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
| 1 | 2 | 3 |
|  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
|  | *Use Case* | Deskripsi dari urutan aksi-aksi yang ditampikan sistem yang menghasilkan  suatu hasil yang terukur bagi suatu *actor*. |
|  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi) |
|  | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi |

1. *Class Diagram*

Menurut Nugroho (2016:6) *Class Diagram* membantu dalam visualisasi struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. *Class diagram* memperlihatkan hubungan antar kelas dan penjelasan secara detail tiap-tiap kelas didalam model desain dari suatu sistem.

*Diagram Class* memberikan pandangan secara luas dari suatu sistem dengan menunjukan kelas-kelasnya dan hubungan mereka. *Diagram Class* bersifat statis menggambarkan hubungan apa yang terjadi bukan apa yang terjadi jika mereka berhubungan, Simbol-simbol yang ada pada *diagram class* ditunjukkan pada tabel 9.

Tabel 9. Simbol-simbol *diagram class*

|  |  |  |
| --- | --- | --- |
| Simbol | Nama | Keterangan |
| 1 | 2 | 3 |
| |  | | --- | | Nama *Class* | | + atribut  + atribut  + atribut | | + *method*  + *method* | | *Class* | *Class* adalah blok -blok pembangun pada pemrograman berorientasi obyek. Sebuah *class* terbagi atas 3 bagian. Bagian atas adalah bagian nama dari *class*. Bagian tengah mendefinisikan property/atribut *class*. Bagian akhir mendefinisikan *method-method* dari sebuah *class*. |
|  | *Association* | Sebuah *asosiasi* merupakan sebuah *relationship* paling umum antara 2 *class* dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 *class*. Garis |
| 1 | 2 | 3 |
|  |  | ini bisa melambangkan tipe-tipe *relationship* dan juga dapat menampilkan hukum-hukum multiplisitas pada sebuah relationship. (Contoh: One-to-one, one-to-many, many-to-many). |
|  | *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. |
|  | *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. Sebuah *dependency* dilambangkan sebagai sebuah panah bertitik-titik. |
|  | *Aggregation* | *Aggregation* mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi |

1. *Activity Diagram*

Pada dasarnya *Activity Diagram* sering digunakan oleh *flowchart*. Diagram ini berhubungan dengan diagram Statechart. *Diagram Statechart* berfokus pada obyek yang dalam suatu proses (atau proses menjadi suatu obyek),  *Activity Diagram* berfokus pada aktifitas-aktifitas yang terjadi yang terkait dalam suatu proses tunggal. Jadi dengan kata lain, diagram ini menunjukkan bagaimana aktifitas-aktifitas tersebut bergantung satu sama lain.

Simbol-simbol yang ada pada *Activity Diagram*:

Tabel 10. Simbol-simbol *Activity Diagram*

|  |  |  |
| --- | --- | --- |
| Simbol | Nama | Keterangan |
| 1 | 2 | 3 |
|  | *Start Point* | Merepresentasikan dimulainya alur kerja suatu sistem dalam activity diagram dinotasikan dengan solid. |
|  | *End Point* | Merepresentasikan diakhirnya alur kerja suatu sistem dalam activity diagram. Dinotasikan dengan lingkaran solid dengan lingkaran di luarnya. |
|  | *Activitiy* | Merepresentasikan performa dari beberapa tingkah laku di dalam alur kerja, dinotasikan dengan segiempat |
| 1 | 2 | 3 |
|  | *Fork* (Percabangan) | *Fork*; Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel |
|  | *Join* (Penggabungan) | *Join*, digunakan untuk menggabungkan dua kegiatan paralel menjadi satu. |
|  | *Decision* | Menentukan kapan alur dalam aktivitas menjadi bercabang. Dinotasikan dengan belah ketupat. |
| Partition 1 | *Swimlame* | Sebuah cara untuk mengelompokan *activity* berdasarkan *actor*. *Actor* (Mengelompokan *activity* dalam sebuah urutan yang sama) |

1. Konsep Basis Data
2. Pengertian Konsep Basis Data

Menurut Edhy Sutanta (2011:9) basis data adalah suatu kumpulan data terhubung (*inter related* data) yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data kalaupun ada maka kerangkapan data tersebut harus seminimal mungkin dan terkontrol (*controller redudancy*), data disimpan dengan cara-cara tertentu sehingga mudah digunakan atau ditampilkan kembali, data dapat digunakan oleh satu atau lebih program-program aplikasi secara optimal, data disimpan tanpa mengalami ketergantungan dengan program yang akan menggunakannya data disimpan sedemikian rupa sehingga proses penambahan, pengambilan, dan modifikasi data dapat dilakukan dengan mudah dan terkontrol.

Untuk mendapatkan informasi yang berguna Dari kumpulan data maka diperlukan suatu perangkat lunak (*software*) untuk memanipulasi data sehingga mendapatkan informasi yang berguna.*Database Manajement System* (DBMS) merupakaan *software* yang digunakan untuk membangun sebuah sistem basis data yang berbasis komputerisasi.DBMS membantu dalam pemeliharaan dan pengolahan kumpulan data dalam jumlah besar, sehingga dengan menggunakan DBMStidak menimbulkan kekacauan dan dapat digunakan oleh pengguna sesuai dengan kebutuhan.DBMS merupakan perantara bagi pemakai dengan basis data.Untuk berinteraksi dengan DBMS (basis data) menggunakan bahasa basis data yang telah ditentukan oleh perusahaan DBMS.Bahasa basis data biasanya terdiri atas perintah-perintah yang diformulasikan sehingga biasanya ditentukan oleh *user*. Ada 2 (dua) bahasa basis data antara lain :

1. *Data Definition Language* (DDL)

Struktur/skema database yang menggambarkan desain database secara keseluruhan.Bahasa inilah dapat dibuat tabel baru, membuat indeks, mengubah tabel, menentukan struktur penyimpanan tabel, dan sebagainya.

1. *Data Manipulation Language* (DML)

Menurut Edhy Sutanta (2011:57) Bahasa manipulasi data (*data manipulation language*) merupakan bahasa yang digunakan untuk memanipulasi basis data. Manipulasi basis data dapat berupa menambah (*append*), menghapus (*delete*), mengganti (*update*), menyisip (*insert*), menarik informasi tertentu (*query*).

*Data Manipulation Language* (DML) merupakan bentuk bahasa database yang berguna untuk melakukan manipulasi dan pengambilan data pada suatu database. Manipulasi data dapat berupa penyisipan/penambahan data baru ke suatu basis data, penghapusan data, dan pengubahan data di suatu *database*. Ada dua tipe *Data Manipulation Language* (DML), yaitu :

1. *Procedural* DML, mengharuskan pengguna untuk menentukan data yang dibutuhkan dan bagaimana mendapatkannya.
2. *Nonprocedural* DML / *Declarative* DML, mengharuskan pengguna menentukan data yang dibutuhkan tanpa menentukan bagaimana mendapatkannya. *Nonprocedural DML / Declarative DML* lebih mudah untuk digunakan dari pada *procedural* DML. Dikarenakan pengguna tidak harus menentukan cara mendapatkan data dalam basis data, maka sistem basis data harus mencari alat yang efektif untuk mengakses data. Komponen DML dari bahasa SQL merupakan *nonprocedural*. *Query* adalah pernyataan yang digunakan pengguna untuk mengambil data.
3. Kelebihan dari Penggunaan Basis Data
4. Kecepatan dan Kemudahan (*Speed*)

Dengan menggunakan basis data pengambilan informasi dapat dilakukan dengan cepat dan mudah.Basis data memiliki kemampuan dalam pengelompokan, pengurutan bahkan perhitungan dengan matematika. Dengan perancangan yang benar, maka penyajian informasi akan dapat dilakukan dengan cepat dan mudah,

1. Kebersamaan Pemakai

Sebuah basis data dapat digunakan oleh banyak *user* dan banyak aplikasi.Untuk data-data yang diperlukan oleh banyak orang/bagian, tidak perlu dilakukan pencatatan dimasing-masing bagian, tetapi cukup dengan satu basis data untuk dipakai bersama. Misalnya data mahasiswa dalam suatu perguruan tinggi, dibutuhkan oleh banyak bagian, diantaranya : bagian akademik, bagian keuangan, bagian kemahasiswaan, dan perpustakaan. Tidak harus semua bagian ini memiliki catatan dan semua bagian dapat mengakses data tersebut sesuai dengan keperluannya.

1. Pemusatan Kontrol Data

Karena cukup dengan satu basis data untuk banyak keperluan, pengontrolan terhadap data juga cukup dilakukan di satu tempat saja.Jika ada perubahan data alamat mahasiswa misalnya, maka tidak perlu meng-update semua data dimasing-masing bagian tetapi cukup hanya disatu basis data.

1. Efisiensi Ruang Penyimpanan (*Space*)

Dengan pemakaian bersama, tidak diperlukan menyediakan tempat penyimpanan diberbagai tempat, tetapi cukup satu saja sehingga ini akan menghemat ruang penyimpanan data yang dimiliki oleh sebuah organisasi. Dengan teknik perancangan basis data yang benar, maka akan menyederhanakan penyimpanan sehingga tidak semua data harus disimpan.

1. Keakuratan (*accuracy*)

Penerapan secara ketat aturan tipe data, domain data, keunikan data, hubungan antara data, dan lain-lain, dapat menekan keakuratan dalam pemasukan dan penyimpanan data.

1. Ketersediaan (*Availability*)

Dengan basis data dapat dilakukan backup data, memilah-milah data mana yang masih diperlukan dan data mana yang perlu disimpan ke tempat lain. Hal ini mengingat pertumbuhan transaksi suatu organisasi dari waktu ke waktu membutuhkan media penyimpanan yang semakin besar.

1. Keamanan (*Security*)

Kebanyakan DBMS dilengkapi dengan fasilitas manajemen pengguna diberikan hak akses yang berbeda-beda sesuai dengan pengguna dan posisinya. Basis data bias diberikan *password* untuk membatasi orang yang mengaksesnya.

1. Kemudahan Dalam Pembuatan Program Aplikasi Baru

Pengguna basis data merupakan bagian dari perkembangan teknologi. Dengan adanya basis data pembuatan aplikasi dapat memanfaatkan kemampuan dari DBMS, sehingga pembuatan aplikasi tidak perlu mengurus penyimpanan data, tetapi cukup mengatur *interface* untuk pengguna.

1. Pemakaian Secara Langsung

Basis data memiliki fasilitas untuk melihat datanya secara langsung dengan *tool* yang disediakan oleh DBMS. Untuk melihat data, langsung ke tabel ataupun menggunakan *query*. Biasanya yang menggunakan fasilitas ini adalah *user* yang sudah ahli, atau *database administrator*.

1. Kebebasan Data (Data Independence)

Jika sebuah program telah selesai dibuat, dan ternyata ada perubahan isi atau struktur data, maka dengan basis data, perubahan ini hanya perlu dilakukan pada *level* DBMS tanpa harus membongkar kembali program aplikasinya.

1. *User View*

Basis data penyediaan pandangan yang berbeda-beda untuk setiap pengguna.Misalnya data yang dimiliki perusahaan yang bergerak dibidang *retail*. Data yang ada berupa data barang, penjualan, dan pembelian.Ada beberapa jenis pengguna yang memerlukan informasi terkait dengan data perusahaan tersebut.Mereka adalah pelanggan, kasir, bagian gudang, bagian akutansi dan manager. Tidak semua data boleh diakses oleh semua pengguna. Misalnya kasir hanya berhak melihat informasi nama barang dan harga jualnya. Sementara itu pemilik berhak untuk memasukan data penjualan.Berbeda dengan pelanggan yang hanya melihat data keberadaan barang dan harga jual tetapi tidak berhak memasukkan atau merubah data.Bagian akutansi berhak melihat keuntungan dari tiap-tiap barang untuk menganalisa data akutansinya.Basis data mampu memberikan layanan organisasi seperti ini.

1. *Entity Relationship Diagram* (ERD)

*Entity Relationship Diagram* atau ERD merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi.

*Entity Relationship Diagram* atau ERD untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannnya digunakan beberapa notasi atau symbol. Notasi atau symbol ERD yang digunakan oleh James Martin yaitu:

1. Entitas (*Entity*)

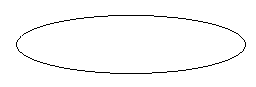
Entitas merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Bentuk dari entitas dapat digambarkan seperti pada gambar berikut :



Gambar . *Entity*

1. Atribut (Attribute)

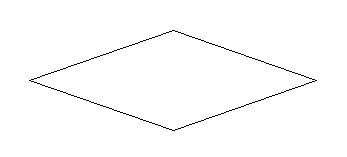
Setiap entitas pasti mempunyai elemen yang disebut atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasi isi elemen satu dengan yang lain. Bentuk atribut dapat digambarkan seperti gambar berikut :



Gambar . *Attribute*

1. Hubungan atau Relasi (*Relationship*)

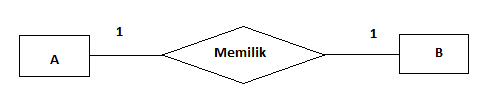
*Relationship* adalah hubungan antara suatu himpunan entitas dengan himpunan entitas lainnya.Symbol yang digunakan adalah bentuk belah ketupat. Bentuk dari hubungan atau relasi dapat digambarkan seperti gambar berikut :

Gambar . Relationship

Adapun jenis-jenis relasi yang terjadi diantara dua himpunan entitas (misalnya A dan B) dalam satu basis data yaitu:

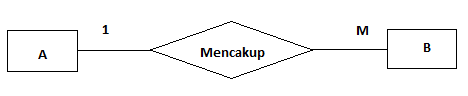
1. Satu ke Satu (*One to one*)

Hubungan relasi satu ke satu yaitu setiap entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas pada himpunan entitas B. Contoh gambar dari hubungan relasi satu ke satu (*One to one*) adalah seperti gambar berikut

Gambar . One to One Relationship

1. Satu ke Banyak (*One to Many*)

Setiap entitas pada himpunan A dapat berhubungan dengan banyak entitas pada himpunan entitas B, tetapi pada entitas B hanya dapat berhubungan dengan satu entitas pada himpunan entitas adalah seperti gambar berikut :



Gambar . One to Many Relationship

1. Banyak ke Banyak (*Many to Many*)

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan B dan sebaliknya entitas B dapat berhubungan banyak pada himpunan entitas A adalah seperti gambar berikut :

Gambar . Many to Many Relationship

1. *Structured Query Language (SQL)*

Menurut Abdul Kadir (2009:101) SQL merupakan bahasa *query*standar yang digunakan untuk mengakses basis data relasional. SQL dirancang untuk menggunakan relasi untuk mengubah *input* menjadi *output* yang diinginkan.

Sekumpulan perintah khusus yang digunakan unutk mengakses data dalam database relasional.SQL merupakan sebuah bahasa computer yang mengikuti standar ANSI (American Nasional Standard Institute) yang digunakan dalam manajemen database relational.

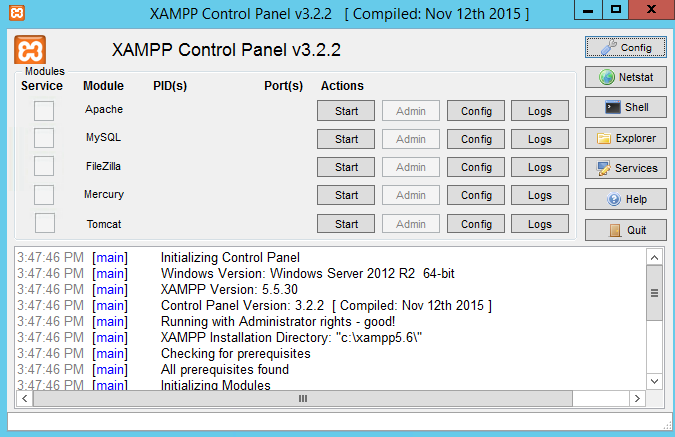
1. **Perangkat Lunak yang digunakan**

Perangkat lunak yang digunakan penulis untuk membangun sistem ini adalah sebagai berikut :

1. XAMPP

Menurut Riyanto (2013:1) XAMPP merupakan paket PHP dan MySQL berbasis *open source*, yang dapat digunakan sebagai *tool* pembantu pengembangan aplikasi berbasis PHP.

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 (empat sistem operasi apapun), *Apache, MySQL, PHP, dan Perl.* Program ini tersedia dalam GNU (*General Public License*) dan bebas, merupakan *web server* yang sudah digunakan yang dapat melayani tampilan halaman *web* yang dinamis.

Gambar . XAMPP

1. Notepad ++

Notepad ++ adalah sebuah penyunting teks dan penyunting kode sumber yang berjalan di sistem operasi *Windows*.Notepad ++ menggunakan komponen Scintila untuk dapat menampilkan dan menyunting teks dan berkas kode sumber berbagai bahasa pemrograman. Karena hanya mengandalkan model teks didalam mengembangkan *web*.



Gambar . Notepad ++

1. Android SDK

Android SDK adalah *tools* API (*Application Pro*gramming *Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java*. Android merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di-*release* oleh Google. Saat ini disediakan Android SDK *(Software Development Kit)* sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java*. Sebagai *platform* aplikasi-netral, Android memberi anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan *Smartphone/Handphone*. Beberpa fitur-fitur *Android* yang paling penting adalah :

1. *Framework* Aplikasi yang mendukung penggantian komponen dan *reusable*.
2. Mesin *Virtual Dalvik* dioptimalkan untuk perangkat *mobile.*
3. *Integrated browser* berdasarkan *engine* *open source Webkit*.
4. Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D
5. berdasarkan spesifikasi opengl ES1,0 (tergantung *hardware*)
6. SQLite untuk penyimpanan data
7. Media *Support* yang mendukung *audio, video*, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM *Telephony* (tergantung *hardware*)
8. Bluethooth, EDGE, 3G, dan WiFi (tergantung *hardware*)
9. Kamera, GPS, kompas, dan *accelerometer* (tegantung *hardware*)
10. Lingkungan *Development* yang lengkap dan kaya termasuk perangkat *emulator, tools* untuk *debugging, profil* dan kinerja memori, dan *plugin* untuk IDE Eclipse.



Gambar 18. *Android SDK*