BAB II

LANDASAN TEORI

Penyusunan tugas akhir ini berkaitan dengan pembuatan suatu aplikasi di dalam *smartphone* berbasis android untuk membantu masyarakat dalam mencapai tempat tujuan dengan menggunkan alat trasportasi umum yang ada di kota Pekanbaru dengan memanfaatkan GPS dari *Google Maps (API's Key)*. Sehingga pembahasan-pembahasan teori yang mengenai teknologi Android beserta *tools* yang mendukung dalam pembuatan aplikasi ini.

2.1 Transportasi

Menurut Sukarto (2006) Transportasi atau perangkutan adalah perpindahan dari suatu tempat ke tempat lain dengan menggunakan alat pengangkutan, baik yang digerakkan oleh tenaga manusia, hewan (kuda, sapi, kerbau), atau mesin. Konsep transportasi didasarkan pada adanya perjalanan (*trip*) antara asal (*origin*) dan tujuan (*desti-nation*). Perjalanan adalah pergerakan orang dan barang antara dua tempat kegiatan yang terpisah untuk melakukan kegiatan perorangan atau kelompok dalam masyarakat. Perjalanan dilakukan melalui suatu lintasan tertentu yang menghubungkan asal dantujuan, menggunakan alat angkut atau kendaraan dengan kecepatan tertentu. Jadi perjalanan adalah proses perpindahan dari satu tempat ke tempat yang lain.

1. Unsur-Unsur Dasar Transportasi

   Ada lima unsur pokok transportasi, yaitu:

   a. Manusia, yang membutuhkan transportasi
   b. Barang, yang diperlukan manusia
   c. Kendaraan, sebagai sarana transportasi
   d. Jalan, sebagai prasarana transportasi
   e. Organisasi, sebagai pengelola transportasi

Pada dasarnya, ke lima unsur di atas saling terkait untuk terlaksananya transportasi, yaitu terjaminnya penumpang atau barang yang diangkut akan sampai ke tempat tujuan dalam keadaan baik seperti pada saat awal diangkut. Dalam hal ini perlu diketahui terlebih dulu ciri penumpang dan barang, kondisi sarana dan konstruksi prasarana, serta pelaksanaan transportasi.

2.2 Android

2.2.1 Sejarah Android

Menurut safaat (2012, 1) Android adalah sistem operasi untuk telepon seluler berbasis Linux sebagai *kernel*nya. Saat ini Android bisa disebut raja dari *smartphone* . Android menyediakan *platform* terbuka (*Open source*) bagi para pengembang untuk menciptakan aplikasi mereka sendiri. Awalnya, perusahaan search *engine* terbesar saat ini, yaitu Google Inc. membeli Android Inc. pendatang baru yang membuat peranti lunak untuk ponsel. Android, Inc. didirikan oleh Andy Rubin, Rich Milner, Nick Sears dan Chris White pada tahun 2003. Pada Agustus 2005 Google membeli Android Inc. Kemudian untuk mengembangkan android dibentuklah *Open* Handset Alliance konsorsium dari 34 perusahaan *hardware, software* dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualqomm, T-*Mobile* dan Nividia.

Pada saat perilisan perdana android, 5 November 2007, android bersama *Open* Handset Alliance menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Dilain pihak, Google merilis kode–kode android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler.

Android memiliki dua distributor, yaitu *Google Mail Service (GMS*) dan *Open Handset Distributor (OHD).* GMS adalah distributor android yang mendapatkan dukungan penuh dari Google, sedangkan OHD adalah distibtor Android tanpa dukungan langsung dari Google.

Saat ini sudah banyak bermunculan vendor-vendor untuk *smartphone* , yaitu diantaranya : HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, Acer,

Philips, T-*Mobile*, Nexian, IMO, Asus dan lainnya vendor yang meproduksi *smartphone* android. Mengapa saat ini sudah banyak bermunculan vendor *smartphone* yang telah disebutkan tadi? Karena sistem operasi Android bersifat *opensource* sehingga saat ini bermunculan verndor *smartphone* sebanyak itu.

Android menjadi pesaing utama dari produk *smartphone* lainnya seperti Apple dan BlackBerry. Pesatnya pertumbuhan Android karena Android adalah *platform* yang sangatlah lengkap baik dari segi sistem operasinya, aplikasi dan *tools* pengembangannya, *market* yang menyimpan berbagai aplikasi serta ditambah dengan berbagai dukungan oleh komunitas *open source* di dunia, sehingga Android berkembang pesat hingga saat ini, baik dari segi teknologi maupun dari segi jumlah *device* di dunia.

2.2.2 *The Dalvik Virtual Machine* (DVM)

Android berjalan di dalam DVM bukan pada *Java Virtual Machine* (JVM) banyak kesamaan antara DVM dan JVM, namun DVM memiliki *feature* yang lebih baik dibandingkan dengan JVM untuk perangkat *mobile*. *Dalvik Virtual Machine* (DVM) adalah "*Register bases*" sementara *Java Vitual Machine* adalah "*Stack bases*" DVM didesain dan ditulis Dan Bornsten dan beberapa *engineers* Google lainnya. Dalam mengatasi fungsionalitas tingkat rendah DVM menggunakan *kernel* Linux untuk keamanan, *threading*, proses dan manajemen memori. Itu memungkinan kita menggunakan bahasa C / C++ dalam membuat aplikasi sama halnya dengan OS Linux kebanyakan. Oleh karena itu kita harus kita harus memahami arsitektur dan proses daru *kernel* Linux yang digunakan dalam Android tersebut.

Android memiliki *virtual machine* untuk eksekusi aplikasi. DVM mengeksekusi *executeable file*, artinya sebuah format yang dioptimalkan untuk memastikan memori yang digunakan sangatlah kecil, karena *executeable file* mengubah kelas bahasa *Java* dan dikompilasi dengan menggunakan *tools* yang sudah ada.

2.2.3 *Software Development Kit* (SDK)

Android SDK merupakan sebuah *tools* yang diperlukan untuk mengembangkan aplikasi berbasis Android menggunakan bahasa pemrograman *Java*. Pada saat ini Android SDK telah menjadi alat bantu dan *API* (*Application Programming Interface*) untuk mengembangkan aplikasi bebasis Android. Android SDK dapat Anda lihat dan unduh pada situs resminya, yaitu http://www.developer.android.com/. Android SDK bersifat gratis dan bebas Anda distribusikan karena Android bersifat *open source*.

2.2.4 Arsitektur Android

Secara garis besar Arsitektur Android dapat jelaskan sebagai berikut :

1. *Application dan Widgets*

   Application dan *Widgets* adalah *layer* dimana kita berhubungan dengan aplikasi saja. Di *layer* terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, *browse*r, kontak, dan lain-lain. Semua aplikasi di tulis dengan menggunakan bahasa pemograman *Java*.

2. *Application Frameworks*

   Android adalah *Open Development Platform* yaitu Android menawarkan kepada pengembang atau *member* kebebasan kepada pengembang untuk membangun aplikasi yang bagus. Pengembang memiliki akses penuh *API framework* seperti yang dilakukan oleh aplikasi yang kategori inti. Komponen-komponen yang termasuk didalam *Application Framework* adalah sebagai berikut :

   a. *View*s.
   b. *Conten provider*.
   c. *Resource manager*.
   d. *Notification manager*.
   e. *Activity manager*.

3. *Libraries*

   Libraries ini adalah *layer* dimana fitur-fitur Android berada, biasanya *libraries* ini untuk menjalankan aplikasi. Berjalan di atas *kernel*,*layer* ini meliputi berbagai *library* C/C++ inti seperti Libc dan SSL,serta :

   a. *Libraries* media untuk pemutaran media audio dan video.
   b. *Libraries* untuk menajemen tampilan.
   c. *Libraries Graphics* mencakup SGL dan *Open*Gl untuk grafis 2D dan 3D.
   d. *Libraries SQLite* untuk mendukung *database*.
   e. *Libraries* SSL dan Webkit terintefgrasi dengan web browser dan *security*.
   f. *Libraries LiveWebcore* mencakup modern web browser dengan *engine embedded web view*.
   g. *Libraries 3D* yang mencakup implementasi *OpenGL ES 1.0 API 's*.

4. *Android Run Time*

   Di dalam Android *Run Time* dibagi menjadi dua bagian yaitu sebagaimana berikut ini.

   a. *Core Libraries.*

      Aplikasi android dibangun dalam bahasa *Java*, sementara *Dalvik* sebagai *virtual* mesinnya bukan *virtual Machine Java*, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menerjemahkan bahasa *Java* yang ditangani oleh *core libraries*.

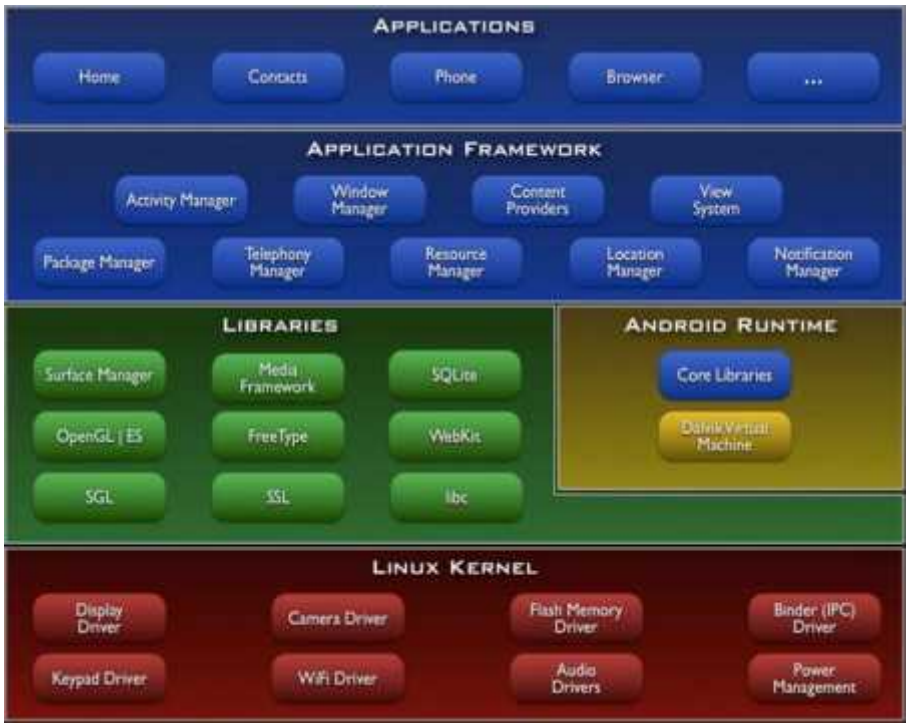
   b. Dalvik *Virtual Machine*

      virtual mesin berbasis *regirster* yang dioptimalkan untuk menjalankan fungsi-fungsi secara efesien, dimana merupakan pengembangan yang mampu membuat linux *kernel* untuk melakukan *threading* dan manajemen tingkat rendah.

5. Linux *Kernel*

   Linux *Kernel* adalah *layer* dimana inti dari *operating* sistem berada. Berisi *file-file system* yang mengatur sistem *processing, memory ,resource,*

drivers dan sistem-sistem oprerasi android lainnya. Linux *Kernel* yang digunakan android adalah linux *kernel realse* 2.6.



Gambar 2.1 *Linux kernel realse 2.6.*

(Sumber: http://elinux.org/Android_Architecture, 2012)

2.2.5 Versi Android

1. Android versi awal (2007 – 2008)

   Pada September 2007 Google mengajukan hak paten aplikasi telepon seluler. Google mengenalkan Nexus One, salah satu jenis telepon pintar GSM yang menggunakan Android pada sistem operasinya. Telepon seluler ini diproduksi oleh HTC *Corporation* dan tersedia di pasaran pada 5 Januari 2010.

   Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android *ARM Holdings, Atheros Communications,* diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc. Seiring pembentukan *Open* Handset Alliance, OHA mengumumkan produk perdana mereka, Android, perangkat bergerak

(*mobile*) yang merupakan modifikasi *kernel* Linux 2.6. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan *bug* dan penambahan fitur baru. *Smartphone* yang memakai sistem operasi Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android.

2. Android versi 1.1

   Pada 9 Maret 2009, Google merilis Android versi 1.1. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email.

3. Android versi 1.5 (*Cupcake*)

   Pada pertengahan Mei 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke *youtube* dan gambar ke *picasa* langsung dari telepon, dukungan *bluetooth* A2DP, kemampuan terhubung secara otomatis ke *headset bluetooth*, animasi layar, dan *keyboar*d pada layar yang dapat disesuaikan dengan sistem.

4. Android versi 1.6 (*Donut*)

   Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol *applet* VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus, kamera, *camcorder* dan galeri yang dintegrasikan; CDMA / EVDO, 802.1x, VPN, *Gestures*, *dan Text-to-speech engine*; kemampuan dial kontak; teknologi *text to change speech* (tidak tersedia pada semua ponsel; pengadaan resolusi VWGA.

5. Android versi 2.0 / 2.1 (*Éclair*)

   Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1(*Eclair*), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan *Google Maps* 3.1.2, perubahan UI dengan browser baru dan dukungan HTML5, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 MP, *digita*l *zoom*, dan *bluetooth* 2.1. Untuk bergerak cepat dalam persaingan perangkat generasi berikut, Google melakukan investasi dengan mengadakan kompetisi aplikasi *mobile* terbaik (*killer apps*-aplikasi unggulan). Kompetisi ini berhadiah $25,000 bagi setiap pengembang aplikasi terpilih. Kompetisi diadakan selama dua tahap yang tiap tahapnya dipilih 50 aplikasi terbaik.

   Dengan semakin berkembangnya dan semakin bertambahnya jumlah *handset* Android, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi Android. Aplikasi terkenal yang diubah ke dalam sistem operasi Android adalah *shazam*, *backgrounds*, dan *weatherBug*. Sistem operasi Android dalam situs Internet juga dianggap penting untuk menciptakan aplikasi Android asli, contohnya oleh *my space* dan *facebook*.

6. Android versi 2.2 (*Froyo : Frozen Yoghurt*)

   Pada 20 Mei 2010, Android versi 2.2 (*Froyo*) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan *Adobe Flash* 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, intergrasi V8 *Java script engine* yang dipakai Google Chrome yang mempercepat kemampuan *rendering* pada *browser*, pemasangan aplikasi dalam SD *card*, kemampuan *wifi hotspot portabel*, dan kemampuan *auto update* dalam aplikasi Android *market*.

7. Android versi 2.3 (*Gingerbread*)

   Pada 6 Desember 2010, Android versi 2.3 (*Gingerbread*) diluncurkan. Perubahan-perubahan umum yang didapat dari Android versi ini antara

lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (*reverb, equalization, headphone virtualization, dan bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

8. Android versi 3.0 / 3.1 (*Honeycomb*)

   Android *Honeycomb* dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada *Honeycomb* juga berbeda karena sudah didesain untuk tablet. *Honeycomb* juga mendukung multi prosesor dan juga akselerasi perangkat keras (*hardware*) untuk grafis. Tablet pertama yang dibuat dengan menjalankan *Honeycomb* adalah Motorola Xoom. Perangkat tablet dengan *platform* Android 3.0 akan segera hadir di Indonesia. Perangkat tersebut bernama *Eee Pad Transformer* produksi dari Asus. Rencana masuk pasar Indonesia pada Mei 2011.

9. Android versi 4.0 (*Ice Cream*)

   Diumumkan pada tanggal 19 Oktober 2011, membawa fitur *Honeycomb* untuk *smartphone* dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari *email* secara *offline*, dan berbagi informasi dengan menggunakan NFC. Ponsel pertama yang menggunakan sistem operasi ini adalah Samsung Galaxy Nexus.

10. Android versi 4.1 (*Jelly Bean*)

    Android *Jelly Bean* yaang diluncurkan pada acara Google I/O lalu membawa sejumlah keunggulan dan fitur baru. Penambahan baru diantaranya meningkatkan *input keyboard*, desain baru fitur pencarian, UI yang baru dan pencarian melalui *Voice Search* yang lebih cepat. Tak ketinggalan *Google Now* juga menjadi bagian yang diperbarui. *Google Now* memberikan informasi yang tepat pada waktu yang tepat

pula. Salah satu kemampuannya adalah dapat mengetahui informasi cuaca,lalu-lintas, ataupun hasil pertandingan olahraga. Sistem operasi Android *Jelly Bean* 4. muncul pertama kali dalam produk tablet Asus, yakni Google Nexus 7. Android versi 4.2 (*Jelly Bean*) fitur *photo sphere* untuk panaroma, *daydream* sebagai *screensaver*, *power control*, *lock screen widget*, menjalankan banyak *user* (dalam tablet saja), *widgetter* baru.

2.3 *Tools* Pengembang Sistem

Dalam melakukan pengembangan terhadap sebuah sistem diperlukan adanya aplikasi yang menjadi *tools* pendukung dalam pembuatan sistem. Dalam penelitian ini yang digunakan sebagai aplikasi pendukung adalah sebagai berikut.

2.3.1 *Eclipse*

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platform independent*). Berikut ini adalah sifat dari Eclipse.

1. *Multi-platform*: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. *Multi-languag*e: Eclipse dikembangkan dengan bahasa pemrograman *Java*, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.
3. *Multi-role*: Selain sebagai IDE untuk pengembangan aplikasi, eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya. Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya

populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in.*

2.3.2 *Java*

Beberapa definisi *Java* adalah sebagai berikut:

1. *Java* sebagai bahasa pemrograman

   Sebagai sebuah bahasa pemrograman, *Java* dapat membuat seluruh aplikasi, *desktop, web* dan lainnya. Sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain. *Java* adalah bahasa pemrograman berorientasi objek (OOP) dan dapat dijalankan pada berbagai *platform* sistem operasi. Perkembangan *Java* tidak hanya terfokus pada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat *open source*.

2. *Java* sebagai *Development Environment*

   Sebagai sebuah peralatan pembangun, teknologi *Java* menyediakan banyak *tools*: *compiler, interpreter*, penyusun dokumentasi, paket kelas dan sebagainya.

3. *Java* sebagai sebuah aplikasi

   Aplikasi dengan teknologi *Java* secara umum adalah aplikasi serba guna yang dapat dijalankan pada seluruh mesin yang memiliki *Java Runtime Environment* (JRE).

4. *Java* sebagai *Deployment Environment*

   Terdapat dua komponen utama dari *Deployment Environment*. Yang pertama adalah JRE, yang terdapat pada paket J2SDK, mengandung kelas-kelas untuk semua paket teknologi *Java* yang meliputi kelas dasar dari *Java*, komponen GUI dan sebagainya. Komponen yang lain terdapat pada Web Browser. Hampir seluruh web browser komersial menyediakan *interpreter* dan *runtime environment* dari teknologi *Java*.

Java memiliki karakteristik sebagai berikut :

1. Sederhana

   Bahasa pemrograman *Java* menggunakan sintaks mirip dengan C++ namun sintaks pada *Java* telah banyak diperbaiki terutama menghilangkan penggunaan pointer yang rumit dan *multiple inheritance*. *Java* juga menggunakan *automatic memory allocation* dan *memory garbage collection*.

2. Berorientasi objek

   Java menggunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Program berorientasi objek memodelkan dunia nyata kedalam objek dan melakukan interaksi antar objek-objek tersebut.

3. Dapat didistribusikan dengan mudah

   Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya *libraries* networking yang terintegrasi pada *Java*.

4. Interpreter

   Program *Java* dijalankan menggunakan interpreter yaitu *Java Virtual Machine* (JVM). Hal ini menyebabkan *source code Java* yang telah dikompilasi menjadi *Java bytecodes* dapat dijalankan pada *platform* yang berbeda-beda.

5. Robust

   Java mempunyai reliabilitas yang tinggi. Compiler pada *Java* mempunyai kemampuan mendeteksi *error* secara lebih teliti dibandingkan bahasa pemrograman lain. *Java* mempunyai *runtime exception handling* untuk membantu mengatasi *error* pada pemrograman.

6. Aman

   Sebagai bahasa pemrograman untuk aplikasi internet dan terdistribusi, *Java* memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut.

7. *Architecture Neutral*

   Program *Java* merupakan *platform independen.* Program cukup mempunyai satu buah versi yang dapat dijalankan pada *platform* yang berbeda dengan *Java Virtual Machine*.

8. Portabel

   Source code maupun program *Java* dapat dengan mudah dibawa ke *platform* yang berbeda-beda tanpa harus dikompilasi ulang.

9. *Performance*

   Performance pada *Java* sering dikatakan kurang tinggi. Namun *performance Java* dapat ditingkatkan menggunakan kompilasi *Java* lain seperti buatan Inprise, Microssoft ataupun Symantec yang mengunakan *Just In Time Compilers* (JIT)

10. *Multithreaded*

    Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa pekerjaan secara sekaligus dan simultan.

11. Dinamis

    Java didesain untuk dapat dijalankan pada lingkungan yang dinamis. Perubahan pada suatu *class* dengan menambahkan properties ataupun method dapat dilakukan tanpa mengganggu program yang menggunakan *class* tersebut.

2.3.3 XML (*Extensible Markup Language*)

XML adalah sebuah *meta-language* untuk mendeskripsikan data. XML merupakan sebuah cara mempersentasikan data tanpa tergantung kepada sistem. XML juga dapat digunakan sebagai *extension markup languages*. XML berbasis *text*, sehingga ia dapat dengan mudah dipindahkan dari satu sistem komputer ke sistem yang lain.

Secara sederhana XML merupakan bahasa berbasis penandaan (*tag*) untuk mendeskripsikan data atau informasi tanpa mempedulikan aplikasi yang kelak akan menggunakannya. Hal ini cukup kontras dibanding HTML yang mendefinisikan bagaimana data atau informasi ditampilkan dan sangat bergantung

pada aplikasi apa (misalnya *browser*) yang akan menggunakannya. Meski demikian aplikasi yang akan menggunakan XML harus tahu aturan yang berlaku dalam pembuatan berkas XML agar aplikasi tersebut mampu memanfaatkannya. XML memungkinkan kita untuk membuat struktur kita sendiri, tidak seperti HTML yang menggunakan struktur yang bersifat baku (Nugroho, 2008).

Android menyediakan model pembuatan *User interface* (UI) dengan file layout berbasis XML. Struktur umum dari sebuah file layout Android sangat sederhana. Dimana terdiri dari elemen XML disusun mirip hierarki pohon, dan setiap node adalah nama dari class View. Kita bisa menggunakan apa saja nama class yang menjadikan View sebagai elemen pada layout XML, termasuk class-class tampilan yang kita atur sendiri dalam kode. Dengan metode ini, kita bisa membuat layout UI dengan cepat, karena menggunakan struktur dan sintaks yang lebih sedehana daripada layout dengan metode programmatic. Model ini terinspirasi dari pengembangan web, dimana kita bisa memisahkan pengaturan tampilan (UI) dari logika aplikasi yang digunakan untuk mengisi, mengolah dan menampilkan data.

2.3.4 PHP (*Hypertext Preprcessor*)

Pada awalnya PHP merupakan singkatan dari *Personal Home Page tools*, yang gunanya untuk memonitor pengunjung suatu web. PHP mula-mula dikembangkan oleh Rasmus Lerdofr. Istilah PHP kemudian mengacu pada *Hypertext Preprcessor*. PHP kemudian lebih dikembangkan untuk membangun aplikasi web yang mendukung *database*. Biasanya dipasangkan dengan Mysql. PHP ini banyak digunaan untuk membuat aplikasi berbasis web.

Menurut Kadir (2009, 47) PHP merupakan bahasa berbentuk skrip yang ditempatkan dalam server dan diproses di server. Hasilnya akan dikirimkan ke client, tempat pemakai menggunakan browser. PHP dikenal sebagai sebuah bahasa scripting, yang menyatu dengan tag-tag HTML, dieksekusi di server, dan digunakan untuk membuat halaman web yang dinamis. Sedangkan menurut Sidik (2012, 76) PHP merupakan bahasa utama *script server-side* yang disisipkan pada

html yang dijalankan di server, juga bisa digunakan untuk membuat aplikasi *desktop*.

Metode kerja PHP diawali dengan permintaan suatu halaman web oleh *browser*, berdasarkan *Uniform Resource Locator* (URL) atau dikenal dengan sebutan alamat internet. *Browser* mendapatkan alamat dari *web server*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *web server*. Selanjutnya *web server* akan mencarikan barkas PHP yang diminta dan setelah didapatkan, isinya akan segera dikirimkan ke mesin PHP dan mesin inilah yang memeroses dan memberikan hasilnya berupa kode HTML ke web server. Lalu *web server* akan menyampaikan isi halaman web tersebut kepada klient melalui *browser*. Setiap *statement*/perintah dari PHP harus diakhiri dengan menggunakan tanda titik koma (;). Umumnya setiap *statement* dituliskan dalam satu baris. Penulisan skrip PHP dalam tag HTML dapat dilakukan dengan dua cara yaitu *Embedded Script* dan *non-Embeddded Script* Kadir (2009, 48).

2.3.5 MySQL

Menurut Raharjo (2011, 21) MySQL merupakan *software* RDMS (*server database*) yang dapat mengelola *database* dengan sangat cepat, dapat menampung data dengan jumlah besar, dapat diakses oleh banyak user (*multi-user*), dan dapat melakukan suatu proses secara sinkronisasi atau bersamaan (*multi-thearead*). Sedangkan menurut sidik (2012, 333) adalah software yang *database* yang memiliki performansi query dari *database*nya sangat cepat, dan jarang terjadi bermasalah

MySQL adalah software *database server* dapat mengelola dengan sangat cepat dan daapt menampung kapasitas data yang banyak selain itu, MySQL sifatnya yang *open source* maka menjadikan MySQL sangat popular dikalangan *programmer* web. Ada beberapa alasan mengapa memilih MySQL sebagai *server database* untuk aplikasi-aplikasi yang dikembangkan menurut Raharjo (2011, 22):

1. Fleksibel

   MySQL dapat digunakan untuk mengembangkan aplikasi *desktop* maupun aplikasi web dengan menggunakan teknologi yang bervariasi.

2. Peforma tinggi

   MySQL memiliki mesin query dengan performa tinggi, dengan demikian proses transaksi yang dilakukan dengan sangat cepat.

3. Lintas *platform*

   MySQL dapat digunakan pada *platform* atau lingkungan (dalam hal ini sistem operasi) yang beragam, bisa Microsoft Windows, Linux, atau Unix. Ini menyebabakan proses migrasi data antar sistem dapat dilakukan secara mudah.

4. Gratis

   MySQL digunakan secara gratis.

5. Proteksi data yang handal

   MySQL menyediakan fasilitas manajemen *user*, enkripsi data, dan lain sebagainya.

6. Komunitas luas

   Karena MySQL banyak penggunanya sehingga memiliki komunitas yang luas. Hal ini berguna jika kita menemui permasalah dalam proses pengolahan data menggunakan MySQL.

2.4 *Google Maps*

Google *Maps* merupakan sebuah layanan peta dunia *virtual* berbasis web yang disediakan oleh Google. Layanan ini gratis dan dapat ditemukan *dihttp://Maps.Google.com*. Google *Maps* menawarkan peta yang dapat digeser *(panned*), diperbesar (*zoom in*), diperkecil (*zoom out*), dapat diganti dalam beberapa mode (map, *satelit*, *hybrid*, dan lain-lain), fitur pencarian rute (*routing*), penunjuk arah dari satu objek peta ke objek yang lain (*direction*) dan juga pencarian tempat (*place*). Sampai saat ini, Google *Maps* masih berada dalam tahap peta, dan masih terus dikembangkan dengan data yang selalu diperbarui secara berkala.

Google *Maps* merupakan hak cipta Google secara *propritery*, sehingga dalam menggunakannya memerlukan adanya perjanjian, Google membuat mekanisme untuk dapat mengakses Google *Maps* dengan *coding* aplikasi dengan sebuah kunci yang dikenal dengan *API Key*.

Gambar 2.2 Google *Maps* Kota Pekanbaru

2.5 *Unified Modeling Language* (UML)

Berikut ini definisi *Unified Modeling Language* (UML) menurut para ahli. Menurut Chonoles, (dalam Widodo 2011, 6) *Unified Modeling Language* (UML) adalah bahasa yang telah menjadi standard untuk visualisasi, menetapkan, membangun dan mendokumentasikan artifak suatu sistem perangkat lunak.

Sedangkan menurut Martin (2005, 33) *Unified Modeling Language* (UML) adalah keluarga notasi garis yang mendukung oleh meta-model tunggal, yang membantu mendeskripsikan dan desain sistem perangkat lunak. Khususnya sistem yang dibangun menggunakan program berbasis objek.

2.5.1 *Use case Diagram*

Pooley, (dalam Widodo 2011, 16) mengatakan bahwa model *use case* dapat dijabarkan dalam diagram *use case*, tetapi diagram tidak identik dengan

model karena model lebih luas dari diagram. Komponen pembentuk diagram *use case*

1. Aktor, menggambarkan pihak yang berperan dalam sistem.
2. *Use case*, aktivitas/sarana yang disiapkan oleh bisnis/sistem.
3. Hubungan (*link*), menghubungkan aktor mana saja yang terlibat dalam *use case* ini.

Tabel 2.1 *Use case Diagram*

No	Notasi	Keterangan	Simbol
1	*Actor*	Pengguna sistem atau yang berikteraksi langsung dengan sistem,bisa manusia,aplikasi, atau pun objek lain.	
2	*Use case*	Digambarkan dengan lingkaran elips dengan nama *use case*nya tertulis di tengah lingkaran	
3	*Association*	Digambarkan dengan sebuah garis yang berfungsi menghubungkan actor dengan *use case*	

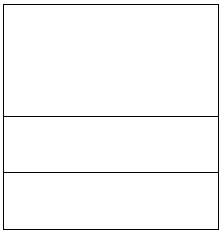
Sumber: http://www.pribadiraharja.com/neli/SKRIPSI/Lampiran/DAFTAR_SIMBOL.doc (2009)

2.5.2 *Class Diagram*

Kelas dinyatakan dalam kotak yang terbagi menjadi beberpa kompartemen. Kompartemen kelas yang berisi infromasi, kompartemen yang pertama Nama kelas *attribute*, *operation* dan *responsibility* dari *class* ada pada kotak tersebut. *Stereotypes* bisa dipergunakan untuk mengorganisasikan daftar *attribute* dan *operation*. Dalam beberapa kasus, kadangkala hanya perlu ditampilkan sebagian saja dari *attribute* dan *operation.* Tipe *attribute* dan nilai *default* bisa dimunculkan sebagaimana pada operation. Untuk mengurangi

ambiguitas pada pendeskripsian *class*, *constraint* bisa ditambahkan. Bahkan kalau perlu bisa ditambahkan *attached notes* ke dalam kotak tersebut, Widodo (2011, 38).

Tabel 2.2 *Class Diagram*

No	Notasi	Keterangan	Simbol
1	*Class*	Blok-blok pembangun program. Bagian atas *class* menunjukkan nama dari *clas*s,bagian tengah mengindikasikan *atribut* dari *clas*s,dan bagian bawah mengidentifikasikan *method* dari sebuah *class*	
2	*Association*	Menunjukkan relationship atau hubungan antar *class*	
3	*Dependency*	Menunnjukkan ketergantungan antara satu class dengan *class* yang lain	
4	*Generallization*	Menunjukkan *inheritance* dari satu class ke beberapa *class*	

Sumber: http://www.pribadiraharja.com/neli/SKRIPSI/Lampiran/DAFTAR_SIMBOL.doc (2009)

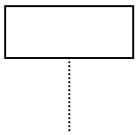
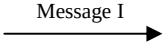
2.5.3 *Sequence Diagram*

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah contoh objek dan *message* yang diletakkan diantara objek-objek ini di dalam *use case*. *Sequence diagram* menambahkan dimensi waktu pada interaksi diantara objek. Pada diagram ini participant diletakkan di atas dan waktu ditunjukkan dari atas ke bawah. *Life lin*e *participant* diurutkan dari setiap *participant*. Kotak kecil pada *lifeline* menyatakan *activation*, yaitu menjalankan salah satu operation dari *participan*t. *State* bisa ditambahkan dengan menambahkannya sepanjang *lifeline Message* (sederhana, s*ynchronous* atau *asynchronous*) adalah

tanda panah yang menghubungkan suatu *lifeline* ke *lifeline* yang lain. Lokasi *lifeline* dalam dimensi vertikal mewakili urutan waktu dalam *sequence diagram*. *Message* yang pertama terjadi adalah yang paling dekat dengan bagian atas diagram dan yang terjadi belakangan adalah yang dekat dengan bagian bawah.

Pada beberapa sistem, operasi bisa dilakukan kepada dirinya sendiri. Hal ini disebut dengan *rekursif*. Untuk melukiskannya digunakan anak panah dari *activation* kembali ke dirinya sendiri, dan sebuah kotak kecil diletakkan pada bagian atas dari *activation.*

Tabel 2.3 *Sequence Diagram*

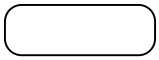
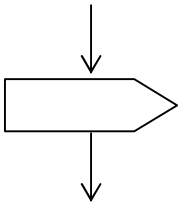
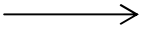
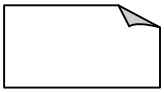
No	Notasi	Keterangan	Simbol
1	*Object*	*Object* adalah *instance* dari sebuah class yang dituliskan tersusun secara *horizontal* diikuti *life line*	
2	*Activation*	Indikasi dari sebuah objek yang melakukan suatu aksi	
3	*Lifeline*	Indikasi keberadaan sebuah *object* dalam basis waktu	
4	*Message*	Indikasi untuk komunikasi antar *object*	
5	S*elf-Massage*	Komunikasi kembali kedalam *object* itu sendiri	

Sumber: http://www.pribadiraharja.com/neli/SKRIPSI/Lampiran/DAFTAR_SIMBOL.doc (2009)

2.5.4 *Activity Diagram*

Diagram ini memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi dalam suatu sistem dan memberi tekanan pada aliran kendali antar objek.

Tabel 2.4 *Activity Diagram*

No	Notasi	Keterangan	Simbol
1	*Initial State*	Titik awal untuk memulai suatu aktivitas	
2	*Final State*	Titik akhir untuk mengakhiri suatu aktivitas	
3	*Activity*	Menandakan sebuah aktivitas	
4	*Decision*	Pilihan untuk pengambilan keputusan	
5	*Fork/Join*	Menunjukkan kegiatan menggabungkan dua panel activity menjadi dua	
6	*Send*	Tanda pengiriman	
7	*Receive*	Tanda penerimaan	
8	*Control Flow*	Arus aktivitas	
9	*Note*	Catatan khusus untuk sebuah aktivitas	

Sumber: http://www.pribadiraharja.com/neli/SKRIPSI/Lampiran/DAFTAR_SIMBOL.doc (2009)