BAB II
TINJAUAN PUSTAKA

2.1. Profil Dinas Perhubungan Kota Bandung

Dinas Perhubungan Pemerintah Kota Bandung atau bisa disingkat Dishub Kota Bandung adalah dinas yang bergerak dalam bidang lalu lintas dan parkir, bidang angkutan dan terminal, bidang sarana, dan bidang operasional.

2.1.1. Sejarah Dinas Perhubungan Kota Bandung

Dinas Perhubungan Kota Bandung dibentuk berdasarkan Perda Kota Bandung Nomor 5 Tahun 2001 tentang Pembentukan dan Susunan Organisasi Dinas Daerah di Lingkungan Pemerintah Kota Bandung.

Sebelum berubah menjadi Dinas Perhubungan, nomenklatur Dinas Perhubungan adalah sebagai berikut:

1. Dinas Lalu Lintas dan Angkutan Jalan Provinsi DT 1 Jawa Barat Cabang Kotamadya DT II Bandung sampai dengan Tahun 1997.
2. Dari Tahun 1997 sampai dengan Tahun 2001 diubah menjadi Dinas Lalu Lintas dan Angkutan Jalan Kotamadya DT II Bandung berdasarkan Perda Kotamadya Bandung Nomor 21 Tahun 1997 tanggal 1 April 1997 tentang pembentukan Dinas Lalu Lintas dan Angkutan Jalan Kotamadya DT II Bandung.
3. Sejak Tahun 2001 sampai dengan sekarang menjadi Dinas Perhubungan Kota Bandung.

2.1.2. Dasar Hukum

Dasar hukum dari berdirinya Dinas Perhubungan Kota Bandung dapat dilihat pada rincian sebagai berikut:

1. Peraturan daerah (Perda) Nomor 13 Tahun 2007 tentang pembentukan dan susunan organisasi Dinas daerah Kota Bandung.
2. Peraturan daerah (Perda) Nomor 16 Tahun 2012 tentang penyelenggaraan perhubungan dan retribusi di bidang perhubungan.

3. Keputusan Walikota Bandung Nomor 475 Tahun 2008 tentang tugas pokok dan fungsi.
4. Keputusan Walikota Bandung Nomor 475 Tahun 2008 tentang uraian tugas jabatan struktural pada Dinas daerah Kota Bandung.
5. Keputusan Walikota Bandung Nomor 1230 Tahun 2001 tentang petunjuk teknis pelaksanaan pengujian kendaraan bermotor, penyelenggaraan bengkel umum, tenaga teknis penguji, penyelenggaraan pendidikan sekolah mengemudi dan kegiatan bongkar muat barang di Kota Bandung.
6. Keputusan Walikota Bandung Nomor 1714 Tahun 2001 tentang petunjuk penyelenggaraan perhubungan di Kota Bandung.
7. Keputusan Walikota Bandung Nomor 551/Kep.140-Dishub/2012 tentang penetapan lokasi dan posisi parkir di tepi jalan umum dan tempat khusus parkir di Kota Bandung.
8. Keputusan Walikota Bandung Nomor 551/Kep.737-Dishub/2012 tentang tim koordinasi kegiatan intensifikasi dan ekstensifikasi parkir dalam rangka penertiban parkir di tepi jalan umum dan tempat khusus Parkir di Kota Bandung.
9. Keputusan Walikota Bandung Nomor 551/Kep.764-Dishub/2012 tentang pengoperasian trans metro bandung pada koridor 2 Cicaheum-Cibereum di Kota Bandung.
10. Peraturan Walikota Bandung Nomor 163 Tahun 2012 tentang harga sewa parkir dan petunjuk teknis pengelolaan perpakiran di gedung dan pelataran parkir di Kota Bandung.

2.1.3. Visi dan Misi Dinas Perhubungan Kota Bandung

Visi dan Misi dari Dinas Perhubungan Pemerintah Kota Bandung adalah dapat dilihat sebagai berikut:

2.1.3.1. Visi

Visi dari Dinas Perhubungan Kota Bandung, yaitu mewujudkan sistem transportasi kota yang lebih baik untuk mendukung Kota Bandung sebagai kota yang unggul, nyaman, dan sejahtera.

2.1.3.2. Misi

Misi dari Dinas Perhubungan Pemerintah Kota Bandung tertuang dalam empat pandangan adalah sebagai berikut :

1. Menciptakan kondisi lalu lintas yang aman, nyaman, tertib, dan terkendali.
2. Menyediakan sarana angkutan umum yang aman dan nyaman.
3. Menyediakan prasarana dan fasilitas perhubungan yang memadai.
4. Meningkatkan profesionalisme aparat Dinas Perhubungan Kota Bandung.

2.1.4. Tugas Pokok dan Fungsi Dinas Perhubungan Kota Bandung

Tugas pokok dan fungsi dari Dinas Perhubungan Pemerintah Kota Bandung adalah sebagai berikut:

2.1.4.1. Tugas Pokok

Tugas pokok Dinas Perhubungan Kota Bandung yaitu melaksanakan sebagian urusan Pemerintah Daerah Kota Bandung di bidang perhubungan berdasarkan atas otonomi dan pembantuan.

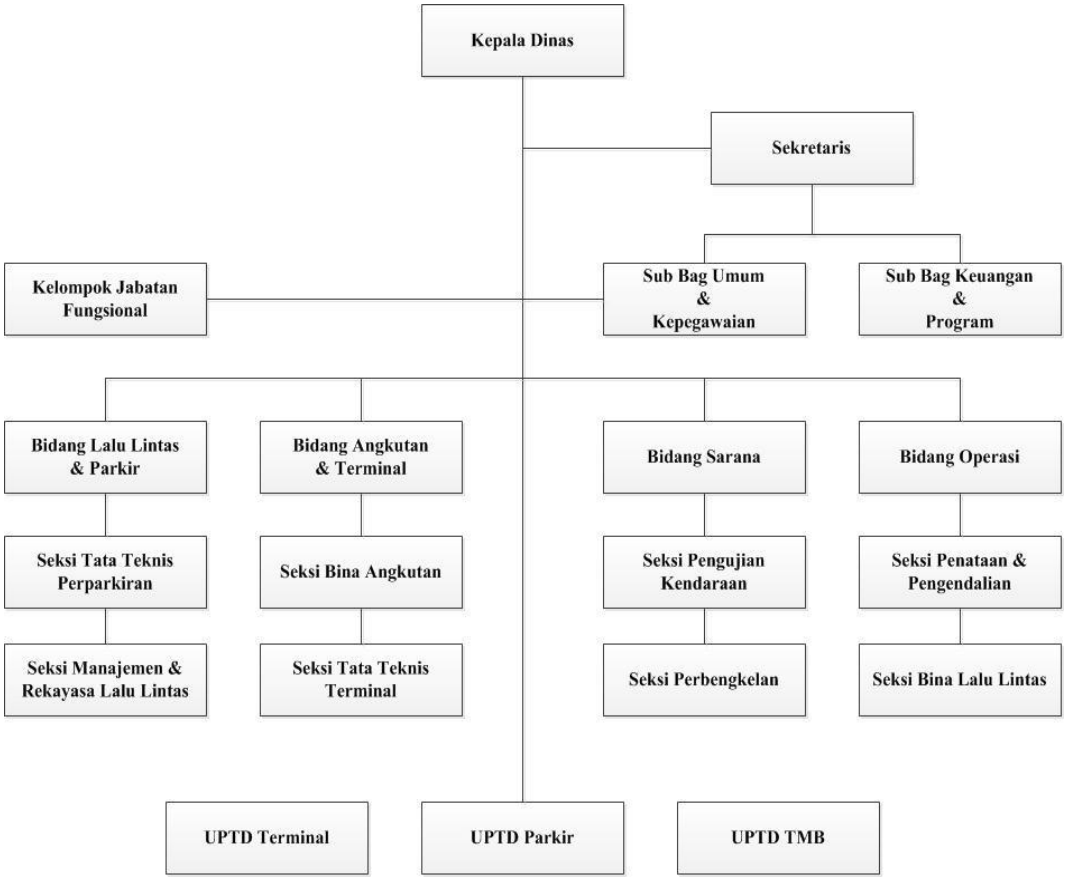
2.1.4.2. Fungsi

Fungsi dari Dinas Perhubungan Kota Bandung terdiri atas empat dasar utama yang dijelaskan sebagai berikut:

1. Perumusan kebijakan teknis di bidang perhubungan.
2. Penyelenggaraan sebagian urusan pemerintah dan pelayanan umum di bidang perhubungan.
3. Pembinaan dan pelaksanaan tugas operasional perhubungan yang meliputi lalu lintas parkir, angkutan, terminal, sarana dan operasional.
4. Pelaksanaan pelayanan teknis administratif dinas.

2.1.5. Struktur Organisasi

Sesuai dengan Peraturan Daerah Kota Bandung Nomor 13 tahun 2007, maka struktur organisasi Dinas Perhubungan Kota Bandung sebagai berikut:



Gambar 2.1 Struktur Organisasi Dinas Perhubungan Kota Bandung

Sumber : Data Dinas Perhubungan Kota Bandung, 2016.

2.2. Landasan Teori

Landasan teori membahas mengenai materi atau teori apa saja yang digunakan sebagai acuan dalam membuat tugas akhir ini. Landasan teori yang diuraikan merupakan hasil dari studi literatur yaitu pengumpulan data dengan cara mempelajari jurnal, browsing internet, referensi buku dan bacaan-bacaan lainnya yang berhubungan erat dengan penelitian yang dilakukan.

2.2.1. Aplikasi Mobile

Menurut Buyens [7] aplikasi mobile berasal dari kata *application* dan *mobile*. *Application* yang artinya penerapan, lamaran, dan penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran

yang dituju. Sedangkan *mobile* dapat diartikan sebagai perpindahan dari suatu tempat ke tempat yang lain.

Maka aplikasi *mobile* dapat diartikan sebuah program aplikasi yang dapat dijalankan atau digunakan walaupun pengguna berpindah-pindah dari satu tempat ke tempat yang lain serta mempunyai ukuran yang kecil. Aplikasi *mobile* ini dapat diakses melalui perangkat nirkabel, pager, PDA, telepon seluler, *smartphone*, dan perangkat sejenisnya.

2.2.1.1. Karakteristik Perangkat Mobile

Perangkat mobile memiliki banyak jenis dalam hal ukuran, desain dan layout, tetapi mereka memiliki karakteristik yang sangat berbeda dari sistem desktop.

Berikut karakteristik perangkat mobile, diantaranya:

a. Ukuran yang kecil

   Perangkat mobile memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.

b. Memory yang terbatas

   Perangkat mobile juga memiliki memory yang kecil, yaitu primary (RAM) dan secondary (disk). Pembatasan ini adalah salah satu faktor yang mempengaruhi penulisan program untuk berbagai jenis dari perangkat ini. Dengan pembatasan jumlah dari memory, pertimbangan-pertimbangan khusus harus diambil untuk memelihara pemakaian dari sumber daya yang mahal ini.

c. Daya proses yang terbatas

   Sistem mobile tidaklah setangguh rekan mereka yaitu desktop. Ukuran, teknologi dan biaya adalah beberapa faktor yang mempengaruhi status dari sumber daya ini. Seperti harddisk dan RAM, anda dapat menemukan mereka dalam ukuran yang pas dengan sebuah kemasan kecil.

d. Mengkonsumsi daya yang rendah

   Perangkat mobile menghabiskan sedikit daya dibandingkan dengan mesin desktop. Perangkat ini harus menghemat daya karena mereka berjalan pada keadaan dimana daya yang disediakan dibatasi oleh baterai-baterai.

e. Kuat dan dapat diandalkan

   Karena perangkat mobile selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapi benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.

f. Konektivitas yang terbatas

   Perangkat mobile memiliki bandwith rendah, beberapa dari mereka bahkan tidak tersambung. Kebanyakan dari mereka menggunakan koneksi wireless.

g. Masa hidup yang pendek

   Perangkat-perangkat konsumen ini menyala dalam hitungan detik, kebanyakan dari mereka selalu menyala. Coba ambil kasus sebuah handphone, mereka booting dalam hitungan detik dan kebanyakan orang tidak mematikan handphone mereka bahkan ketika malam hari. PDA akan menyala jika anda menekan tombol power mereka.

2.2.2. Android

Android merupakan sistem operasi mobile yang tumbuh di tengah sistem operasi lainnya yang berkembang dewasa ini. Sistem operasi lainnya seperti Windows Mobile, i-Phone OS, Symbian dan masih banyak lagi juga menawarkan kekayaan isi dan keoptimalan berjalan diatas perangkat hardware yang ada. Akan tetapi, sistem operasi yang ada ini berjalan dengan memprioritaskan aplikasi inti yang dibangun sendiri tanpa melihat potensi yang cukup besar dari aplikasi pihak ketig. Oleh karena itu, adanya keterbatasan distribusi aplikasi pihak ketiga untuk platform mereka.

Android menawarkan sebuah lingkungan yang berbeda untuk pengembang. Setiap aplikasi memiliki tingkatan yang sama. Android tidak membedakan antara aplikasi inti dengan aplikasi pihak ketiga. Application Programming Interface (API) yang disediakan menawarkan akses ke hardware,

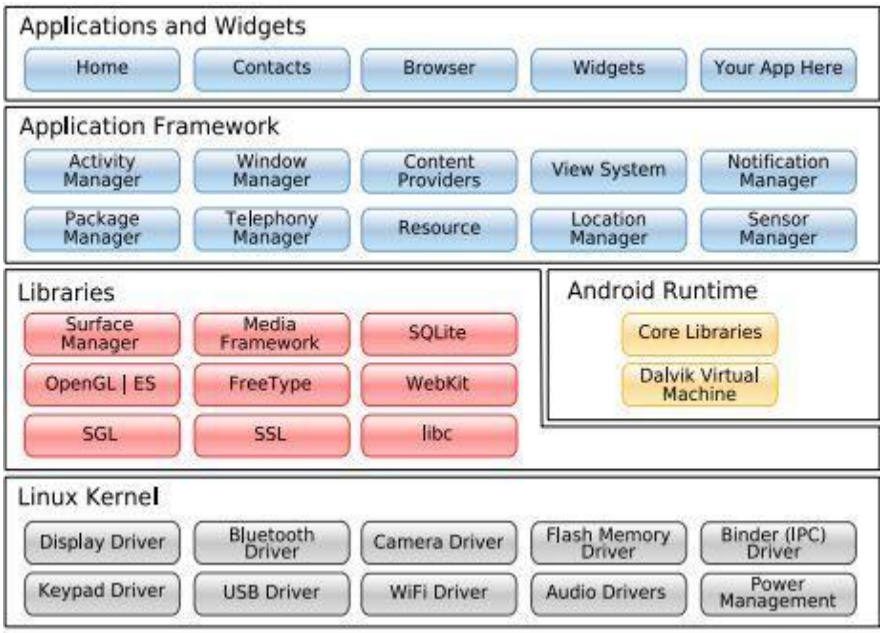
maupun data-data ponsel sekalipun, atau data sistem. Bahkan pengguna dapat menghapus aplikasi inti dan menggantikannya dengan aplikasi pihak ketiga [8].

2.2.2.1. Arsitektur Android

Arsitektur android dapat digambarkan seperti pada gambar 2.2 dan secara garis besar arsitektur android dapat dijelaskan sebagai berikut:

a. Application dan Widgets

Application dan Widgets ini adalah layer dimana kita berhubungan dengan aplikasi saja, dimana biasanya kita download aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut. Di layer terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Hampir semua aplikasi ditulis menggunakan bahasa pemrograman Java.



Gambar 2.2 Arsitektur Android [8]

b. Application Framework

Android adalah “Open Development Platform” yaitu android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi resource, menjalankan service

background, mengatur alarm, dan menambah status notifications, dan sebagainya. Pengembang memiliki akses penuh menuju API framework seperti yang dilakukan oleh aplikasi kategori inti. Arsitektur aplikasi dirancang supaya kita dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (reuse). Sehingga bisa kita simpulkan Application Frameworks ini adalah layer dimana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi android, karena pada layer inilah aplikasi dapat dirancang dan dibuat, seperti content providers yang berupa sms dan panggilan telepon.

Komponen-komponen yang termasuk di dalam Application Frameworks [8], adalah sebagai berikut:

1. Views
2. Content Provider
3. Resource Manager
4. Notification Manager
5. Activity Manager

c. Libraries

Libraries ini adalah layer dimana fitur-fitur android berada, biasanya para pembuat aplikasi mengakses libraries untuk menjalankan aplikasinya. Berjalan diatas kernel, layer ini meliputi berbagai library C/C++ inti seperti Libc SSL [8], serta:

1. Libraries media untuk pemutaran media audio dan video
2. Libraries untuk manajemen tampilan
3. Libraries graphics mencakup SGL dan OpenGL untuk grafis 2D dan 3D
4. Libraries SQLite untuk dukungan database
5. Libraries SSL dan WebKit terintegrasi dengan web browser dan security
6. Libraries LiveWebcore mencakup modern web browser dengan engine embedded webview
7. Libraries 3D yang mencakup implementasi OpenGL ES1.0 API’s

d. Android Run Time

Layer yang membuat aplikasi android dapat dijalankan dimana dalam prosesnya menggunakan Implementasi Linux. Dalvik Virtual Machine (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi android. Di dalam android Run Time [8] dibagi menjadi dua bagian yaitu:

1. Core Libraries : aplikasi android dibangun dalam bahasa Java, sementara Dalvik sebagai virtual mesinnya bukan Virtual Machine Java, sehingga diperlukan sebuah libraries yang berfungsi untuk menterjemahkan bahasa Java/C yang ditangani oleh Core Libraries.
2. Dalvik Virtual Machine : virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat Linux Kernel untuk melakukan threading dan manajemen tingkat rendah.

e. Linux Kernel

Linux Kernel adalah layer dimana inti dari sistem operasi android itu berada. Berisi file-file sistem yang mengatur sistem processing, memory, resource, drivers, dan sistem-sistem operasi android lainnya. Linux Kernel yang digunakan android adalah Linux Kernel release 2.6 [8].

2.2.2.2. Sejarah Android

Pada Juli 2000, *Google* bekerjasama dengan *Android Inc*., perusahaan yang berada di Palo 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 Inc*. hanyalah sebagai perangkat lunak pada telepon seluler. Sejak saat itu muncul rumor bahwa *Google* hendak memasuki pasar telepon seluler. Di perusahaan *Google*, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yang didukung oleh *Kernel Linux*. Hal ini menunjukan indikasi bahwa *Google* sedang bersiap menghadapi persaingan dalam pasara telepon seluler. Versi *android* terbaru yaitu versi 5.0. (*KitKat*). *Android* juga sudah bergabung dengan beberapa *smart mobile* seperti *LG, Samsung, Sony Ericsson, HTC, Motorola* dan lainnya.

Sekitar September 2007 sebuah studi melaporkan bahwa *Google* mengajukan hak paten aplikasi telepon seluler (akhirnya *Google* mengenalkan *Nexus One,* salah satu jenis telepon pintar GSM yang menggunakan *Android* pada sistem operasinya. Telepon seluler ini diproduksi oleh *HTC Corporotaion* dan tersedia di pasaran pada 5 Januari 2010).

Pada 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 penmbahan fitur baru [9].

2.2.2.3. Versi Android

Banyak *smartphone* dan PC Tablet menggunakan sistem operasi dengan versi yang berbeda. Semakin tinggi versi, fiturnya semakin canggih dan banyak. Telepon pertama yang memakai sistem operasi *Android* adalah HTC Dream yang dirilis pada tanggal 22 Oktober 2008. Beberapa uraian versi *android* seperti dibawah ini :

1. 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.

2. 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 *keyboard* pada layar yang dapat disesuaikan dengan sistem.

3. 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 pada kamera, *camcorder* dan galeri yang diintegrasikan pada CDMA / EVDO, 802.1x, VPN, *Gestures*, dan *Text-to-spech engine*. Kemampuan dial kontak teknologi *text to change speech* (tidak tersedia pada semua ponsel).

4. 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 mengoptimalkan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML 5, daftar kontak yang baru, dukungan *flash* untuk kamera 3.2 MP, *digital 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 *MySpace* dan *Facebook*.

5. 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, integrasi dengan V8 JavaScript *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.

6. Android Versi 2.3 (*Gingerbread*)

   Pada 6 Desember 2010, Android versi 2.3 (*Gingerbread*) diluncurkan. Perubahan-perubahan umum yang didapat dari versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antarmuka (*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.

7. Android Versi 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 peratam yang dibuat dengan menjalankan *Honeycomb* adalah Motorola Xoom.

8. Android Versi 4.0 (*Ice Cream Sandwich*)

   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.

9. Android Versi 4.1 (*Jelly Bean*)

   Pada tanggal 27 Juni 2012 Google mengumumkan *Android* versi 4.1 (*Jelly Bean*) dalam konferensi Google I/O berdasarkan kernel Linux 3.0.31. *Jelly Bean* adalah pembaruan penting yang bertujuan untuk meningkatkan fungsi

dan kinerja antarmuak pengguna (*User Interface*). Pembaruan ini diwujudkan dalam proyek *Butter*, perbaikan ini termasuk antisipasi sentuh, *triple buffering*, perpanjangan waktu vsync dan peningkatan *frame rate* hingga 60fps untuk menciptakan UI yang lebih halus. Perangkat pertama yang menggunakan sistem operasi ini adalah tablet Nexus 7, yang dirilis pada 13 Juli 2012.

10. Android Versi 4.4 (*KitKat*)

    Android versi 4.4 (KitKat) direncanakan akan dirilis pada bulan Oktober 2013. Setelah sebelumnya beredar rumor bahwa Android versi berikutnya setelah *Jelly Bean* diperkirakan akan diberi nomor 5.0 dan dinamai *Key Lime Pe* [9].

2.2.2.4. Komponen Aplikasi Android

Fitur penting Android adalah bahwa satu aplikasi dapat menggunakan elemen dari aplikasi lain (untuk aplikasi yang memungkinkan). Sebagai contoh, sebuah aplikasi memerlukan fitur *scroller* dan aplikasi lain telah mengembangkan fitur *scroller* yang baik dan memungkinkan aplikasi lain menggunakannya. Maka pengembang tidak perlu lagi mengembangkan hal serupa untuk aplikasinya, cukup menggunakan *scroller* yang telah ada. Agar fitur tersebut dapat bekerja, sistem harus dapat menjalankan aplikasi ketika setiap bagian aplikasi itu dibutuhkan, dan pemanggilan objek java untuk bagian itu. Oleh karenanya Android berbeda dari sistem-sistem lain, Android tidak memiliki satu tampilan utama program seperti fungsi main() pada aplikasi lain. Sebaliknya, aplikasi memiliki komponen penting yang memungkinkan sistem untuk memanggil dan menjalankan ketika dibutuhkan [9].

1. Activities

   Activity merupakan bagian yang paling penting dalam sebuah aplikasi, karena *Activity* menyajikan tampilan visual program yang sedang digunakan oleh pengguna. Setiap *Activity* dideklarasikan dalam sebuah kelas yang bertugas untuk menampilkan antarmuka pengguna yang terdiri dari *Views* dan respon

terhadap *Event*. Setiap aplikasi memiliki sebuah *activity* atau lebih. Biasanya pasti akan ada *activity* yang pertama kali tampil ketika aplikasi dijalankan.

Perpindahan antara *activity* dengan *activity* lainnya diatur melalui sistem, dengan memanfaatkan *activity stack*. Keadaan suatu *activity* ditentukan oleh posisinya dalam tumpukan *acitivity*, LIFO (*Last In First Out*) dari semua aplikasi yang sedang berjalan. Bila suatu *activity* baru dimulai, *activity* yang sebelumnya digunakan maka akan dipindahkan ketumpukan paling atas. Jika pengguna ingin menggunakan *activity* sebelumnya, cukup menekan tombol *Back*, atau menutup *activity* yang sedang digunakan, maka *activity* yang berada diatas akan aktif kembali. *Memory Manager* Android menggunakan tumpukkan ini untuk menentukan prioritas aplikasi berdasarkan *activity*, memutuskan untuk mengakhiri suatu aplikasi dan mengambil sumber daya dari aplikasi tersebut [9].

Ketika *activity* diambil dan disimpan dalam tumpukkan *activity* terdapat 4 kemungkinan kondisi transisi yang akan terjadi :

a. *Active*

   Setiap *activity* yang berada ditumpukan paling atas, maka dia akan terlihat, terfokus, dan menerima masukkan dari pengguna. Android akan berusaha untuk membuat *activity* aplikasi ini untuk untuk tetap hidup dengan segala cara, bahkan akan menghentikan *activity* yang berada dibawah tumpukkannya jika diperlukan. Ketika *activity* sedang aktif, maka yang lainnya akan dihentikan sementara [9].

b. *Paused*

   Dalam beberapa kasus *activity* akan terlihat tapi tidak terfokus pada kondisi inilah disebut *paused*. Keadaan ini terjadi jika *activity* transparan dan tidak *fullscreen* pada layar. Ketika *activity* dalam keadaan *paused*, dia terlihat *active* namun tidak dapat menerima masukkan dari pengguna. Dalam kasus ekstrim, Android akan menghentikan *activity* dalam keadaan *paused* ini, untuk menunjang sumber daya bagi *activity* yang sedang aktif [9].

c. *Stopped*

Ketika sebuah *activity* tidak terlihat, maka itulah yang disebut *stopped*. *Activity* akan tetap berada dalam memori dengan semua keadaan dan informasi yang ada. Namun akan menjadi kandidat utama untuk dieksekusi oleh sistem ketika membutuhkan sumberdaya lebih. Oleh karenanya ketika suatu *activity* dalam kondisi *stopped* maka perlu disimpan data dan kondisi antarmuka saat itu. Karena ketika *activity* telah keluar atau ditutup, maka dia akan menjadi *inactive* [9].

d. *Inactive*

Kondisi ketika *activity* telah dihentikan dan sebelum dijalankan. *Inactive activity* telah ditiadakan dari tumpukan *activity* sehingga perlu *restart* ulang agar dapat tampil dan digunakan kembali. Kondisi transisi ini sepenuhnya ditangani oleh manajer memori Android. Android akan memulai menutup aplikasi yang mengandung *activity inactive*, kemudian *stopped activity*, dan dalam kass luar biasa *paused activity* juga akan ditutup [9].

2. Services

Suatu *service* tidak memiliki tampilan antarmuka, melainkan berjalan di *background* untuk waktu yang tidak terbatas. Komponen *service* diproses tidak terlihat, memperbarui sumber data dan menampilkan notifikasi. *Service* digunakan untuk melakukan pengolahan data yang perlu terus diproses, bahkan ketika *Activity* tidak aktif atau tidak tampak [9].

3. Intents

Intents merupakan sebuah mekanisme untuk menggambarkan tindakan tertentu, seperti memilih foto, menampilkan halaman *web*, dan lain sebagainya. *Intents* tidak selalu dimulai dengan menjalankan aplikasi, namun juga digunakan oleh sistem untuk memberitahukan ke aplikasi bila terjadi suatu hal, misal pesan masuk. *Intents* dapat eksplisit atau implisit, contohnya jika suatu aplikasi ingin menampilkan *URL*, sistem akan menentukan komponen apa yang dibutuhkan oleh *Intents* tersebut [9].

4. Broadcast Receiver

Broadcast Receivers merupakan komponen yang sebenarnya tidak melakukan apa-apa kecuali menerima dan bereaksi menyampaikan pemberitahuan. Sebagian besar *Broadcast* berasal dari sistem misalnya, Batre sudah hampir habis, informasi zona waktu telah berubah, atau pengguna telah merubah bahasa *default* pada perangkat. Sama halnya dengan *service*, *Broadcast Receivers* tidak menampilkan antarmuka pengguna. Namun, *Broadcast Receivers* dapat menggunakan *Notification Manager* untuk memberitahukan sesuatu kepada pengguna [9].

5. Content Providers

   Content Providers digunakan untuk mengelola dan berbagi *database*. Data dapat disimpan dalam file sistem, dalam database *SQLite*, atau dengan cara lain yang pada prinsipnya sama. Dengan adanya *Content Provider* memungkinkan antar aplikasi untuk saling berbagi data. Komponen ini sangat berguna ketika sebuah aplikasi membutuhkan data dari aplikasi lain, sehingga mudah dalam penerapannya [9].

2.2.2.5. Tipe Aplikasi Android

Terdapat tiga kategori aplikasi pada android dapat dilihat pada penjelasan sebagai berikut :

1. *Foreground Activity*

   Aplikasi yang hanya dapat dijalankan jika tampil pada layar dan tetap efektif walaupun tidak terlihat. Aplikasi dengan tipe ini mempertimbangkan siklus hidup *activity*, sehingga perpindahan antar *activity* dapat berlangsung dengan lancar [9].

2. *Background Service*

   Aplikasi yang memiliki interaksi terbatas dengan user, selain dari pengaturan konfigurasi, semua dari prosesnya tidak tidak tampak pada layar. Contohnya aplikasi penyaringan panggilan atau sms *auto respon* [9].

3. *Intermittent Activity*

   Aplikasi yang masih membutuhkan beberapa masukkan dari pengguna, namun sebagian sangat efektif jika dijalankan di *background* dan jika

diperlukan akan memberi tahu pengguna tentang kondisi tertentu. Contohnya pemutar musik. Untuk aplikasi yang kompleks akan sulit untuk menentukan kategori aplikasi tersebut apalagi aplikasi memiliki ciri-ciri dari semua kategori. Oleh karenanya perlu pertimbangan bagaimana aplikasi tersebut digunakan dan menentukan kategori aplikasi yang sesuai [9].

2.2.2.6. Kelebihan Android

Sudah banyak *platform* untuk perangkat selular saat ini, termasuk didalamnya *Symbian*, *iPhone*, *Windows Mobile*, *BlackBerry*, *Java Mobile Edition*, *Linux Mobile* (LiM), dan banyak lagi. Namun ada beberapa hal yang menjadi kelebihan Android. Walaupun beberapa fitur-fitur yang ada telah muncul sebelumnya pada *platform* lain, Android adalah yang pertama menggabungkan hal seperti berikut :

1. Keterbukaan, bebas pengembangan tanpa dikenakan biaya terhadap sistem karena berbasiskan *Linux* dan *open source*. Pembuat perangkat menyukai hal ini karena dapat membangun *platform* yang sesuai yang diinginkan tanpa harus membayar *royality*. Sementara pengembang *software* menyukai karena Android dapat digunakan diperangkat manapun dan tanpa terikat oleh *vendor* manapun.
2. Arsitektur komponen dasar android terinspirasi dari teknologi *internet Mashup*. Bagian dalam sebuah aplikasi dapat digunakan oleh aplikasi lainnya, bahkan dapat diganti dengan komponen lain yang sesuai dengan aplikasi yang dikembangkan.
3. Banyak dukungan *service*, kemudahan dalam menggunakan berbagai macam layanan pada aplikasi seperti penggunaan layanan pencarian lokasi, *database* SQL, *browser* dan penggunaan peta. Semua itu sudah tertanam pada Android sehingga memudahkan dalam pengembangan aplikasi.
4. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga kerja sistem menjadi lebih stabil. Pengguna tak perlu khawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.

5. Dukungan grafis dan suara terbaik, dengan adanya dukungan 2D grafis dan animasi yang diilhami oleh Flash menyatu dalam 3D menggunakan *OpenGL* memungkinkan membuat aplikasi maupun *game* yang berbeda.

   Portabilitas aplikasi, aplikasi dapat digunakan pada perangkat yang ada saat ini maupun yang akan datang. Semua program ditulis dengan menggunakan bahas pemrograman Java dan dieksekusi oleh mesin *virtual Dalvik*, sehingga kode program *portable* antara ARM, X86, dan arsitektur lainnya. Sama halnya dengan dukungan masukan seperti penggunaan *Keyboard*, layar sentuh, *trackball* dan resolusi lyr semua dapat disesuaikan dengan program [9].

2.2.3. Pemrograman Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek [10].

Pada saat ini, metode berorientasi objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentransformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan pada saat ini beragam dengan platform yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi ke semua jenis aplikasi tersebut. Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut :

1. Meningkatkan produktivitas

Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (reusable).

2. Kecepatan pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengodean.

3. Kemudahan pemeliharaan

   Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.

4. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

5. Meningkatkan kualitas perangkat lunak

   Karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

2.2.3.1. Konsep Dasar Berorientasi Objek

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi, atau sistem lainnya). Pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata.

Ada banyak cara untuk mengabstraksikan dan memodelkan objek-objek tersebut, mulai dan abstraksi objek, kelas, hubungan antar kelas sampai abstraksi sistem. Saat mengabstraksikan dan memodelkan objek, data dan proses-proses

yang dipunyai oleh objek akan dienkapsulasi (dihubungkan) menjadi suatu kesatuan.

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancangan, pemrograman, dan pengujian perangkat lunak. Ada berbagai teknik yang dapat digunakan pada masing-masing tahap tersebut, dengan aturan dan alat bantu pemodelan tertentu [10].

Sistem berorientasi objek merupakan sebuah sistem yang dibangun dengan berdasarkan metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus (dienkapsulasi) menjadi kelompok data dan fungsi. Setiap komponen dalam sistem tersebut dapat mewarisi atribut dan sifat dan komponen lainnya, dan dapat berinteraksi satu sama lain.

Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek:

1. Kelas (*Class*)

   Kelas adalah sekumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statis dan himpunan objek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (model/operasi), hubungan (relationship) dan arti. Suatu kelas dapat diturunkan dan kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru. Secara teknis kelas adalah sebuah struktur dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek.

2. Objek (*Object*)

   Objek adalah abstraksi dari sesuatu yang mewakili dunia nyata benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan. Secara teknis, sebuah kelas saat program dieksekusi makan akan dibuat sebuah objek. Objek dilihat dari segi teknis adalah elemen pada saat runtime yang akan diciptakan,

dimanipulasi, dan dihancurkan saat eksekusi sehinga sebuah objek hanya ada saat sebuah program dieksekusi. Jika masih dalam bentuk kode, disebut sebagai kelas jadi pada saat runtime (saat sebuah program dieksekusi), yang kita punya adalah objek, di dalam teks program yang kita lihat hanyalah kelas.

3. Metode (*Method*)

   Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek. Metode atau operasi dapat berasal dari event, aktifitas atau aksi keadaan, fungsi, atau kelakuan dunia nyata. Contoh metode atau operasi misalnya *Read*, *Write*, *Move*, *Copy*, dan sebagainya.

4. Atribut (*Attribute*)

   Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya.

5. Abstraksi (*Abstraction*)

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

6. Enkapsulasi (*Encapsulation*)

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerja-nya.

7. Pewarisan (*Inheritance*)

   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.

8. Antarmuka (*Interface*)

   Antarmuka sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah interface dapat diimplementasikan oleh kelas lain.

9. *Reusability*

   Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

10. Generalisasi dan Spesialisasi

    Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus. Misalnya kelas yang lebih umum (generalisasi) adalah kendaraan darat dan kelas khususnya (spesialisasi) adalah mobil, motor, dan kereta.

11. Komunikasi Antarobjek

    Komunikasi antarobjek dilakukan lewat pesan (message) yang dikirim dari satu objek ke objek lainnya.

12. Polimorfisme (*Polymorphism*)

    Kemampuan suatu objek digunakan di banyak tujuan yang berbeda dengan nama yang sehingga menghemat baris program.

13. *Package*

    Package adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam package yang berbeda.

2.2.4. GPS (Global Positioning System)

Global Positioning System adalah sistem navigasi berbasis satelit terdiri dari jaringan 24 satelit ditempatkan ke orbit oleh Departemen Pertahanan Amerika Serikat yang pertama kali diperkenalkan mulai tahun 1978. Layanan GPS dahulu hanya dipergunakan untuk keperluan militer namun mulai terbuka untuk publik. 24 satelit GPS tersebut berada sekitar 12.000 mil di atas bumi bergerak

mengelilingi bumi 12 jam dengan kecepatan 7.000 mil per jam. Satelit GPS berkekuatan energi sinar matahari, memiliki baterai cadangan untuk menjaga agar tetap berjalan pada saat gerhana matahari atau pada saat tidak ada energi matahari dan memiliki roket penguat kecil pada masing-masing satelit agar dapat mengorbit tepat pada tempatnya.

Satelit-satelit GPS harus selalu berada pada posisi orbit yang tepat untuk menjaga akurasi data yang dikirim ke GPS *receiver*, sehingga harus selalu dipelihara agar posisinya tepat. Posisi satelit-satelit tersebut selalu dipantau oleh stasiun pengendali. Stasiun-stasiun pengendali di bumi ada di Hawaii, Ascension Islan, Diego Garcia, Kwajalein dan Colorado Spring. Untuk dapat mengetahui posisi seseorang maka diperlukan alat yang diberi nama GPS *receiver* yang berfungsi untuk menerima sinyal yang dikirim dari satelit GPS. GPS *receiver* mengambil informasi tersebut dan melakukan perhitungan triangulation untuk menentukan lokasi pengguna dengan tepat. GPS receiver membandingkan waktu sinyal dikirim dengan waktu sinyal tersebut diterima untuk mengetahui jarak satelit. Dengan mengetahui jarak tersebut, GPS receiver dapat melakukan perhitungan dan menentukan posisi pengguna dan menampilkan dalam peta elektronik. Setelah menentukan posisi pengguna, selanjutnya GPS dapat menghitung informasi lain, seperti kecepatan, arah yang dituju, jalur, tujuan perjalanan, jarak tujuan, matahari terbit dan matahari terbenam dan masih banyak lagi.

Satelit GPS sangat presisi dalam mengirim informasi karena satelit tersebut menggunakan jam atom. Jam atom yang ada pada satelit merupakan partikel atom yang diisolasi, sehingga dapat menghasilkan jam yang akurat dibandingkan dengan jam biasa. Keistimewaan GPS adalah mampu bekerja dalam berbagai kondisi cuaca, siang atau malam. Keakuratan sebuah perangkat GPS bisa mencapai 15 meter, bahkan model terbaru yang dilengkapi teknologi Wide Area Augmentation System (WAAS) keakuratannya sampai 3 meter. Karena GPS bekerja mengandalkan satelit, maka penggunaannya disarankan di tempat terbuka. Penggunaan di dalam ruangan, atau di tempat yang menghalangi arah satelit (di angkasa), maka GPS tidak akan bekerja secara akurat dan maksimal. Perhitungan

waktu yang akurat sangat menentukan akurasi perhitungan untuk menentukan informasi lokasi. Semakin banyak sinyal satelit yang dapat diterima maka akan semakin presisi data posisi yang dihasilkan. Selain itu, ketinggian juga mempengaruhi proses kerja GPS, karena semakin tinggi maka semakin bersih atmosfer, sehingga gangguan semakin sedikit [4].

Sinyal yang diterima oleh alat penerima (receiver) di permukaan, dimana GPS receiver ini akan mengumpulkan informasi dari satelit GPS [11], seperti:

a. Waktu, GPS receiver menerima informasi waktu dari jam atom yang mempunyai keakurasian sangat tinggi.
b. Lokasi, GPS memberikan informasi lokasi dalam tiga dimensi:
    1. Latitude
    2. Longitude
    3. Elevasi
c. Kecepatan, ketika berpindah tempat, GPS dapat menunjukkan informasi kecepatan berpindah tersebut.
d. Arah perjalanan, GPS dapat menunjukkan arah tujuan.
e. Simpan lokasi, tempat-tempat yang sudah pernah atau ingin dikunjungi bisa disimpan oleh GPS receiver.
f. Komulasi data, GPS receiver dapat menyimpan informasi track, seperti total perjalanan yang sudah pernah dilakukan, kecepatan rata-rata, kecepatan paling tinggi, kecepatan paling rendah, waktu/jam sampai tujuan, dan sebagainya.

2.2.4.1. Akurasi GPS

Posisi yang ditunjukkan oleh suatu GPS mempunyai faktor kesalahan atau juga disebut tingkat akurasi. Sebagai contoh suatu alat GPS menunjukkan titik koordinat dengan tingkat akurasi 5 meter, itu berarti posisi pengguna bisa berada dalam range radius 5 meter dari titik yang ditunjukkan tersebut. Mengapa tingkat akurasi yang terlihat bisa berubah-ubah? Kadang terlihat 10 meter, 15 meter, atau 5 meter. Ada beberapa hal yang mempengaruhi tingkat akurasi tersebut [11], antara lain:

a. Kesalahan Ephemeris, terjadi jika satellite tidak dapat mentransmisikan posisinya di orbit dengan tepat.
b. Keadaan Ionosphere, ionosphere berada pada jarak sekitar 43-50 mil di atas permukaan bumi. Satelite yang melewati ionosphere akan menjadi lambat dikarenakan adanya plasma (gas dengan tingkat kepadatan rendah). Walaupun GPS receiver berusaha untuk mengkoreksi/memperbaiki faktor keterlambatan yang terjadi tetap saja aktivitas tertentu dari plasma bisa menyebabkan kesalahan perhitungan.
c. Keadaan Troposphere, troposphere adalah bagian terendah dari atmosfer sampai dengan ketinggian sekitar 11 mil dari permukaan tanah. Variasi pada temperature, tekanan, dan kelembaban bisa menyebabkan perbedaan kecepatan penerimaan gelombang radio.
d. Kesalahan Waktu, karena penempatan jam atom pada setiap GPS receiver tidak berjalan sebagaimana mestinya. Kesalahan waktu dari GPS receiver yang tidak presisi dapat menimbulkan ketidakakurasian.
e. Kesalahan Multipath, terjadi karena sinyal satelite membentur permukaan keras (seperti bangunan atau tebing) sebelum mencapai GPS receiver. Hal tersebut bisa menyebabkan terjadinya delay sehingga perhitungan jarak menjadi tidak akurat.
f. Buruknya Sinyal Satelit, keadaan langit yang terhalang akan menyebabkan GPS sulit menerima data satelit. Sebuah sinyal satelit yang pada hari tertentu diterima dengan sangat bagus belum tentu pada hari lain bisa diterima dengan kualitas yang sama walaupun user berdiri pada tempat yang sama. Hal tersebut dikarenakan posisi dari satelit yang terus bergerak atau bisa juga disebabkan faktor penghalang lain seperti pohon, gedung bertingkat, dan sebagainya.

2.2.4.2. Android dan GPS

Dewasa ini, teknologi berkembang dengan pesat. Dulu ponsel hanya sekedar digunakan untuk menelpon dan SMS saja. Sekarang ponsel sudah menjelma menjadi kotak kecil ajaib yang serba bisa. Salah satunya adalah ponsel

dengan sistem operasi Android. Dengan standarisasi fitur dan hardware yang dimiliki, menjadikan ponsel android ponsel canggih yang disukai banyak orang. Tidak lagi canggih karena adanya fitur MMS, radio, atau internet berkecepatan tinggi tapi juga karena ditanamkannya fitur teknologi satelit di dalamnya. Ya, perangkat GPS receiver yang dulu besar dan eksklusif, sekarang sudah bisa dimiliki dengan "hanya" membeli sebuah ponsel.

Dengan ponsel berteknologi satelit (GPS), banyak hal yang bisa dilakukan. Ingin melihat di mana posisi user sekarang dalam sebuah peta? Mengambil foto/video yang sudah dilengkapi dengan data koordinat? Ingin tahu jalur olahraga bersepeda yang sudah pernah user lalui? Atau ingin tahu dimana posisi ponsel yang hilang dicuri orang? Semua itu bisa dilakukan dengan ponsel android yang memiliki fitur GPS. Tidak hanya itu, user juga dapat pergi ke tempat wisata tertentu dengan dipandu gambar dan suara dari sebuah ponsel! Bahkan lebih jauh lagi, GPS dapat digunakan untuk membantu memberikan peringatan dini terhadap terjadinya bencana alam.

Sekarang ini banyak sekali pengembang-pengembang aplikasi untuk sistem operasi android termasuk aplikasi-aplikasi GPS. Yang menyenangkan aplikasi-aplikasi tersebut jenisnya beragam dan jumlahnya pun banyak [11].

2.2.4.3. Latitude dan Longitude

Latitude dan longitude adalah sebuah koordinat yang ada di bumi dan digunakan untuk menentukan kedudukan kita di atas bumi [12]. Berikut penjelasan latitude dan longitude:

1. Latitude

   Latitude adalah garis yang melintang dari kutub utara dan kutub selatan. Titik 0 adalah sudut ekuator, tanda + menunjukan arah ke atas menuju kutub utara, sedangkan tanda minus di koordinat Latitude menuju ke kutub selatan. Titik yang dipakai dari 0 ke 90 derajat arah kutub utara, dan 0 ke -90 derajat ke kutub selatan.

2. Longitude

Longitude adalah garis lintang. Angka dari sudut bundar bumi horisontal. Titik diawali dari 0 ke 180 derajat, dan - ke arah sebaliknya. Titik 0 dimulai dari garis negara Inggris. Mengarah ke Indonesia akan menjadi angka positif. Kebalikannya koordinat Longitude minus adalah arah kebalikan.

2.2.4.4. LBS (Location Based-Service)

Salah satu ciri mobile phone atau tablet pc adalah portabilitas, sehingga tidak mengherankan bahwa beberapa fitur Android sangat menarik seperti layanan yang memungkinkan anda menemukan, mencari serta menvisualisasikan posisi kita ke dalam peta lokasi fisik seperti GoogleMaps. Kita dapat membuat peta berbasis GoogleMaps dan menjadikannya sebagai element dalam layout User Interface (UI) yang kita rancang. Kita dapat melakukan akses penuh ke peta GoogleMaps, dan memungkinkan kita untuk mengontrol pengaturan tampilan, mengubah tampilan zoom lokasi, dan memindahkan lokasi tampilan [11].

Location Based Service adalah service yang berfungsi untuk mencari dengan teknologi Global Positioning Service (GPS) dan Google's cell-based location. Map dan layanan berbasis lokasi menggunakan lintang dan bujur untuk menentukan lokasi geografis, namun sebagai user kita membutuhkan alamat atau posisi realtime kita bukan nilai lintang dan bujur. Android menyediakan geocoder yang mendukung forward dan reverse geocoding. Menggunakan geocoder, anda dapat mengkonversi nilai lintang bujur menjadi alamat dunia nyata atau sebaliknya.

Location based service atau layanan berbasis lokasi adalah istilah umum yang digunakan untuk menggambarkan teknologi yang digunakan untuk menemukan lokasi perangkat yang kita gunakan. Dua unsur utama LBS [11] adalah:

a. Location Manager (API Maps)

   Menyediakan tools/resource untuk LBS, Application Programming Interface (API) Maps menyediakan fasilitas untuk menampilkan, memanipulasi maps/peta beserta feature-feature lainnya seperti tampilan satelit, street (jalan), maupun gabungannya. Paket ini berada pada com.google.android.maps.

b. Location Providers (API Location)

Menyediakan teknologi pencarian lokasi yang digunakan oleh device/perangkat. API Location berhubungan dengan data GPS dan data lokasi realtime. API Location berada pada paket android yaitu dalam paket android.location. Dengan Location Manager, kita dapat menentukan lokasi kita saat ini, Track gerakan/perpindahan, serta kedekatan dengan lokasi tertentu dengan mendeteksi perpindahan.

2.2.4.5. Haversine

Metode Haversine digunakan untuk menghitung jarak antara titik di permukaan bumi menggunakan garis lintang (longitude) dan garis bujur (lattitude) sebagai variabel inputan. Haversine formula adalah persamaan penting pada navigasi, memberikan jarak lingkaran besar antara dua titik pada permukaan bola (bumi) berdasarkan bujur dan lintang.

Dengan mengasumsikan bahwa bumi berbentuk bulat sempurna dengan jari-jari R 6.367,45 km, dan lokasi dari 2 titik di koordinant bola (lintang dan bujur) masing-masing adalah lon1, lat1, dan lon2, lat2, maka rumus Haversine dapat ditulis dengan persamaan sebagai berikut [13] :

```
x = (lon2-lon1) * cos ((lat1+lat2)/2);
y= (lat2-lat1);
d= sqrt(x*x+y*y)*R
```

Gambar 2.3 Rumus Haversine

Keterangan:

x	= Longitude (Lintang)
y	= Latitude (Bujur)
d	= Jarak
r	= Radius Bumi = 6371 km
1 derajat	= 0.0174532925 radian

Contoh perhitungan haversine formula

Lokasi 1: lon1= 119.800801, lat1= -0.790175	
Lokasi 2: lon2= 119.8428 , lat2= -0.8989	
lat1	= -0.790175 * 0.0174532925 radian = -0.013791155 radian
lon	= 119.800801 * 0.0174532925 radian = 2.090918422 radian
lat2	= -0.8989 * 0.0174532925 radian = -0.01569 radian
lon2	= 119.8428 * 0.0174532925 radian= 2.091651 radian

```
x       = (lon2-lon1) * cos ((lat1+lat2)/2)
        = (2.091651-2.090918422) * cos ((-0.013791155 + -0.01569)/2)
        = 0.0007329412
y       = (lat2-lat1)
        = (-0.01569- (-0.013791155))
        = -0.001897609
d       = sqrt (x*x + y*y) * R
        = sqrt((0.0007329412 * 0.0007329412) + ( -0.001897609 * -
          0.001897609)) * 6371
        = sqrt (0.0000041381) * 6371
        =12.96012927km
```

2.2.5. Google Maps

GoogleMaps adalah peta online atau membuka peta secara online, dapat dilakukan secara mudah melalui layanan gratis dari Google. Bahkan layanan ini menyediakan API (Application Programming Interface) yang memungkinkan developer lain untuk memanfaatkan aplikasi ini di aplikasi buatannya. Tampilan GoogleMaps pun dapat dipilih, berdasarkan foto asli atau peta gambar rute saja.

GoogleMaps adalah layanan gratis yang diberikan oleh Google dan sangat popular. GoogleMaps adalah suatu peta dunia yang dapat kita gunakan untuk melihat suatu daerah seperti pada gambar 2.4. Dengan kata lain, GoogleMaps merupakan suatu peta yang dapat dilihat dengan menggunakan suatu browser. Kita dapat menambahkan fitur GoogleMaps dalam web yang telah kita buat atau pada blog kita yang berbayar maupun gratis sekalipun dengan GoogleMaps API. GoogleMaps API adalah suatu library yang berbentuk JavaScript.

Cara membuat GoogleMaps untuk ditampilkan pada suatu web atau blog sangat mudah hanya dengan membutuhkan pengetahuan mengenai HTML serta JavaScript, serta koneksi Internet yang sangat stabil. Dengan menggunakan GoogleMaps API, kita dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga kita dapat fokus hanya pada data-data yang akan ditampilkan. Dengan kata lain, kita hanya membuat suatu data sedangkan peta yang akan ditampilkan adalah milik Google sehingga kita tidak dipusingkan dengan membuat peta suatu lokasi, bahkan dunia.



Gambar 2.4 Google Maps

Pada GoogleMaps API terdapat 4 jenis pilihan model peta yang disediakan oleh Google, diantaranya adalah:

a. ROADMAP, untuk menampilkan peta biasa 2 dimensi.
b. SATELITE, untuk menampilkan foto satelit.
c. TERRAIN, untuk menunjukkan relief fisik permukaan bumi dan menunjukkan seberapa tingginya suatu lokasi, contohnya akan menunjukkan gunung dan sungai.
d. HYBRID, akan menunjukkan foto satelit yang diatasnya tergambar pula apa yang tampil pada ROADMAP (jalan dan nama kota).

2.2.6. Geofencing

Geofence adalah parameter virtual untuk wilayah geografis di dunia nyata. Parameter virtual yang ditetapkan ini bisa menjadi statis ataupun dinamis. Salah satu contoh khas dari geofence dinamis adalah radius sekitar toko atau titik lokasi. geofence statis dapat di atur dari batas yang telah ditetapkan, seperti zona kehadiran di sekolah atau batas lingkungan. Kostum digital geofencing juga dapat digunakan, jika perangkat yang menyediakan layanan berbasis lokasi (LBS) pengguna masuk atau keluar geofencing, perangkat akan menerima pesan notifikasi. Notifikasi ini mungkin berisi informasi tentang lokasi dari perangkat. Notifikasi Geo-fence mungkin dikirimkan ke ponsel sebagai contoh meningkatkan setiap bentuk lain dari tindakan.

Kita bisa menggambarkan geo fencing sebagai LBS proaktif. Dalam banyak kasus sistem proaktif jauh lebih nyaman dari yang reaktif, di mana pengguna harus secara eksplisit meminta data untuk lokasi. Ada berbagai jenis

lokasi peristiwa untuk memperbaiki posisi GPS yang dapat diuji. Misalnya, apakah pengguna di dekat dengan tempat tujuan (POI) atau ke pengguna lain.

Geofence sekarang ini menjadi salah satu wajah baru di Layanan Berbasis Lokasi. Sekarang pengembangan asli untuk LBS sendiri lebih atau kurang selesai. Kita bisa melihat beberapa standar umum dan aplikasi (berpikir tentang berbagai implementasi layanan Tempat-tempat seperti Foursquare, Twitter places, Google place, Facebook place, dll) Semua dari mereka memudahkan anda dalam berbagi lokasi info atau mendapatkan beberapa hal yang berada di dekatnya. Dan geo-fence membuka pintu untuk beberapa personalisasi. Anda bisa mendapatkan beberapa kustom disesuaikan dengan data ditempat yang benar, terutama ketika kedua aliran data dan tempat (area) untuk menerima itu dihasilkan secara dinamis. Pada kedua, geo-fence akan terintegrasi dalam waktu terdekat dengan sensor dan aplikasi M2M [14].

2.2.7. Geotagging

Geotagging atau juga dikenal sebagai GPS *photo tagging*, merupakan proses penambahan informasi posisi data GPS (*Latitude*, *Longitude*, *Altitude*) dalam sebuah foto digital. Ponsel-ponsel berkamera yang memiliki GPS *receiver* internal umumnya memiliki fitur ini. Mekanisme GPS *photo tagging* adalah pada saat sebuah foto diambil menggunakan kamera (digital atau ponsel) yang memiliki fitur *geotagging*, kamera atau ponsel tersebut mencatat lebih banyak informasi/data dibandingkan dengan sebuah foto yang diambil dengan kamera biasa. Informasi tersebut termasuk waktu dan data ketika sebuah foto diambil, orientasi dari kamera (*portrait* atau *landscape*), apakah menggunakan lampu flash dan detil kamera lainnya yang digunakan seperti Apertur, Exposure, dan Local Length. Semua data ini disimpan pada suatu tempat yang disebut EXIF Headers.

EXIF (*Exchangeable Image File Format*) headers berisi petunjuk foto dengan data yang dapat dibaca oleh aplikasi manajemen foto atau sebuah situs foto tertentu. Selain itu, EXIF *Headers* juga menyediakan sebuah ruang untuk mengisi koordinat *Longitude*, *Latitude*, dan *Altitude* [4].

2.2.7.1. EXIF Headers

Data EXIF (*Exchangeable Image File Format*) adalah data yang disimpan oleh kamera digital pada sebuah image yang berisi informasi mengenai kondisi dan setting kamera digital pada waktu dilakukan pemotretan. EXIF dikembangkan oleh Japanese Electronics Industry Development Association (JEIDA) sebagai upaya untuk mempermudah dan membuat standar pertukaran data antara perangkat lunak pengolah gambar/citra digital dan perangkat keras seperti kamera, serta didukung oleh hampir seluruh kamera digital.

Informasi umum yang bisa didapatkan dari EXIF sebuah foto digital adalah sebagai berikut [4]:

1. Tanggal dan jam berapa sebuah foto dibuat.
2. Merk, tipe kamera, dan jenis lensa yang dipakai untuk memotret foto.
3. Resolusi dari kamera yang digunakan untuk mengambil gambar, digunakan atau tidaknya fitur flash pada kamera.
4. Lokasi foto tersebut diambil, jika kamera yang digunakan mendukung GPS.

2.2.7.2. EXIF Interface Android Class

Kelas Exif Interface adalah salah satu kelas dalam pemrograman android yang dapat digunakan untuk membaca dan menulis informasi tag Exif pada file yang berformat JPEG. Berikut adalah tabel 2.1 Tag EXIF informasi Exif yang dapat diambil [15].

Tabel 2.1 Tag EXIF [15]

No	Tagging	Keterangan
1	TAG_APERTURE	Type is String
2	TAG_DATETIME	Type is String
3	TAG_DATETIME_DIGITIZED	Type is String
4	TAG_EXPOSURE_TIME	Type is String
5	TAG_FLASH	Type is int
6	TAG_FOCAL_LENGTH	Type is rational
7	TAG_GPS_ALTITUDE	The altitude (in meters) based on the reference in TAG_GPS_ALTITUDE_REF
8	TAG_GPS_ALTITUDE_REF	0 if the altitude is above sea level
9	TAG_GPS_DATESTAMP	Type is String
10	TAG_GPS_LATITUDE	String
11	TAG_GPS_LATITUDE_REF	Type is String
12	TAG_GPS_LONGITUDE	String
13	TAG_GPS_LONGITUDE_REF	Type is String
14	TAG_GPS_PROCESSING_METHOD	Type is String

15	TAG_GPS_TIMESTAMP	Type is String
16	TAG_IMAGE_LENGTH	Type is int
17	TAG_IMAGE_WIDTH	Type is int
18	TAG_ISO	Type is String
19	TAG_MAKE	Type is String
20	TAG_MODEL	Type is String
21	TAG_ORIENTATION	Type is int
22	TAG_SUBSEC_TIME	Type is int
23	TAG_SUBSEC_TIME_DIG	Type is int
24	TAG_SUBSEC_TIME_ORIG	Type is int
25	TAG_WHITE_BALANCE	Type is int

2.2.7.3. Google Geocoding

Google Maps sudah memiliki fitur yang cukup lengkap, termasuk mencari lokasi kota pada peta (*geocoding*). *Geocoding* adalah proses menemukan koordinat geografis, yaitu bujur dan lintang, dari data geografis lain, misalnya alamat. Fitur *geocoding* dapat dimanfaatkan untuk menambahkan informasi *latitude* dan *longitude* pada foto yang tidak memiliki data *geotagging* [4].

2.2.8. Metode Pengujian Sistem

Metode pengujian sistem terdiri dari Pengujian *black box* dan *beta* yang dilakukan untuk mengetahui efektifitas dari perangkat lunak (*software*) yang digunakan.

2.2.8.1. Pengujian *Black Box*

Pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Pengujian ini dianalogikan seperti melihat suatu kotak hitam yang hanya bisa dilihat penampilan luarnya saja, tanpa tahu ada apa dibalik bungkus hitamnya. *Black box* testing melakukan evaluasi untuk menemukan kesalahan dalam kategori berikut [16] :

a. Fungsi tidak benar atau hilang
b. Kesalahan *interface* atau antarmuka
c. Kesalahan dalam struktur data atau akses *database eksternal*
d. Kesalahan kinerja atau prilaku dan kesalahan inisialisasi dan terminasi

2.2.8.2. Pengujian *Beta*

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif dimana diuji secara langsung ke lapangan, dengan menggunakan kuesioner mengenai tanggapan pengguna terhadap aplikasi yang telah dibangun. Adapun metode penilaian pengujian yang digunakan yaitu metode kuantitatif.

2.2.8.3. Skala Likert

Skala likert adalah suatu skala psikometrik yang digunakan dalam kuesioner dan merupakan salah satu teknik yang dapat digunakan dalam evaluasi suatu program atau kebijakan perencanaan. Rensis Likert telah mengembangkan sebuah skala untuk mengukur sikap masyarakat di tahun 1932 yang sekarang terkenal dengan nama skala Likert. Skala Likert ini merupakan skala yang dapat dipergunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang mengenai suatu gejala atau fenomena [17].

Dalam skala Likert terdapat dua bentuk pernyataan yaitu pernyataan positif yang berfungsi untuk mengukur sikap positif, dan pernyataan negative yang berfungsi untuk mengukur sikap negative objek. Skor pernyataan positif dimulai dari 1 untuk sangat tidak setuju (STS), 2 untuk tidak setuju (TS), 3 untuk ragu-ragu (RR), 4 untuk setuju (S), dan 5 untuk sangat setuju (SS). Skor pernyataan negative dimulai dari 1 untuk sangat setuju (SS), 2 untuk setuju (S), 3 untuk ragu-ragu (RR), 4 untuk tidak setuju (TS), dan 5 untuk sangat tidak setuju (STS). Beberapa menghilangkan option "Ragu-ragu" dalam instrument untuk memudahkan dalam melihat angket yang responden isikan. Skala Likert digunakan untuk mengukur kesetujuan dan ketidaksetujuan seseorang terhadap sesuatu rencana program, pelaksanaan program ataupun tingkat keberhasilan suatu program.

Keuntungan Skala Likert :

1. Karena ukuran yang digunakan adalah ukuran ordinal, skala likert hanya dapat mengurutkan individu dalam skala, tetapi tidak dapat membandingkan berapa kali satu individu lebih baik dari individu yang lain.

2. Kadangkala total skor dari individu tidak memberikan arti yang jelas, karena banyak pola respons terhadap beberapa item akan memberikan skor yang sama. Adanya kelemahan diatas sebenarnya dapat dipikirkan sebagai error dari respons yang terjadi.

Kelebihan Skala Likert :

1. Mudah dibuat dan diterapkan
2. Skala likert lebih mudah membuatnya dibanding lain seperti skala Thurstone
3. Terdapat kebebasan dalam memasukkan pertanyaan – pertanyaan, asalkan sesuai dengan konteks permasalahan yang diteliti.
4. Jawaban suatu item dapat berupa alternative, sehingga informasi mengenai item tersebut diperjelas.
5. Reliabilitas pengukuran bisa diperoleh dengan jumlah item tersebut diperjelas
6. Karena jangka responsi yang lebih besar membuat skala likert dapat memberikan keterangan yang lebih jelas dan nyata tentang pendapatan atau sikap responden tentang isu yang dipertanyakan.

2.2.8.4. Kuesioner

Menurut Depdikbud kuesioner atau angket adalah suatu alat pengumpul data yang berupa serangkaian pertanyaan yang diajukan pada *responden* untuk mendapat jawaban. Angket merupakan teknik pengumpulan data yang dilakukan dengan mengadakan komunikasi dengan sumber data. Menurut KBBI (Kamus Besar Bahasa Indonesia) kuesioner merupakan alat riset atau survei yang terdiri atas serangkaian pertanyaan tertulis, bertujuan mendapatkan tanggapan dari kelompok orang terpilih melalui wawancara pribadi atau melalui pos daftar pertanyaan.

Kuesioner merupakan teknik pengumpulan data yang dilakukan dengan cara memberi seperangkat pertanyaan atau pernyataan tertulis kepada responden untuk dijawabnya. Selain itu, kuesioner juga cocok digunakan bila jumlah responden cukup besar dan tersebar di wilayah yang luas. kuesioner dapat berupa pertanyaan/pernyataan tertutup atau terbuka, dapat diberikan kepada responden secara langsung atau dikirim melalui pos, atau internet. Bila penelitian dilakukan

pada lingkup yang tidak terlalu luas, sehingga kuesioner dapat diantarkan langsung dalam waktu yang tidak terlalu lama, maka pengiriman kuesioner kepada responden tidak perlu melalui pos [18].

2.3. Alat-alat Pemodelan Sistem

Alat-alat pemodelan sistem membahas mengenai *tools* apa saja yang digunakan dalam membuat tugas akhir ini.

2.3.1. Structure Analysis and Design

Structure analysis and design adalah pendekatan sistem analisis dan desain sistem informasi secara hierarki. SADM diproduksi untuk CCTA, kantor pemerintah Inggris yang berkaitan dengan penggunaan teknologi di pemerintahan dari tahun 1980 dan seterusnya [19].

2.3.1.1. Diagram Konteks

Diagram Context, adalah data flow diagram tingkat atas (DFD Top Level), yaitu diagram yang paling tidak detail, dari sebuah sistem informasi yang menggambarkan aliran-aliran data ke dalam dan ke luar sistem dan ke dalam dan ke luar entitas-entitas eksternal. (CD menggambarkan sistem dalam satu lingkaran dan hubungan dengan entitas luar. Lingkaran tersebut menggambarkan keseluruhan proses dalam sistem) [20].

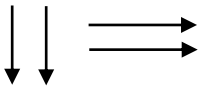
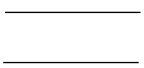
2.3.1.2. Data Flow Diagram

Data Flow Diagram (DFD) adalah representasi grafik dari sebuah sistem. DFD menggambarkan komponen-komponen sebuah sistem, aliran-aliran data di mana komponen-komponen tersebut, dan asal, tujuan, dan penyimpanan dari data tersebut. DFD terdiri dari empat komponen utama diantaranya adalah:

1. Entitas
2. Proses
3. Data Store
4. Data Flow

DFD dapat dikatakan merupakan breakdown dari diagram konteks. Analisis mengenai sistem yang lebih detail dapat dilakukan dengan melakukan breakdown pada DFD level 1, 2 dan seterusnya. Berikut adalah Tabel 2.2 Simbol Diagram Konteks dan DFD [20]

Tabel 2.2 Simbol Diagram Konteks dan DFD [20]

Simbol	Nama Simbol	Fungsi
	Entitas Luar	Menggambarkan entitas eksternal yang berhubungan dengan sistem
	Sistem (Konteks) / Proses (DFD)	Menggambarkan proses yang ada dalam suatu sistem
	Aliran Data/Informasi	Menggambarkan aliran data antar proses, data store dan entitas luar
	Data Storage	Mengambarkan tempat penyimpanan data di dalam sistem

2.3.1.3. Spesifikasi Proses

Spesifikasi Proses adalah tabel yang berisi keterangan atau deskripsi dari semua proses yang terdapat pada DFD. Logika proses yang ditulis pada spesifikasi proses harus dituliskan secara jelas baik menggunakan bahasa deskriptif atau *pseudecode* tetapi tidak boleh mengkombinasikan kedua bahasa tersebut. Berikut adalah Tabel 2.3 Contoh Spesifikasi Proses [20].

Tabel 2.3 Contoh Spesifikasi Proses [20]

No	Proses	Keterangan
1.	No Proses	Menyatakn no proses
	Nama Proses	Menyatakan nama proses
	Source (Sumber)	(Menyatakan sumber data input menuju proses)
	Input	(menyatakan isi data yang masuk ke proses)
	Output	(menyatakan informasi yang keluar dari proses)
	Destination	(menatakan tujuan informasi output dari proses)
	Logika Proses	(menyatakan algoritma dari proses)

2.3.1.4. Kamus Data

Kamus data adalah tabel yang berisi deskripsi dari data yang mengalir pada DFD, Penjelasan struktur data dapat berupa field tiap data harus sama dengan yang sudah dimodelkan di ERD. Berikut adalah Tabel 2.4 Contoh Kamus Data [20].

Tabel 2.4 Contoh Kamus Data [20]

Nama aliran data	Data Siswa Nama dari data yang digunakan
Where used/How used	Daftar pada poin proses apa data digunakan
Keterangan	Uraian singkat dari data yang digunakan
Struktur Data	Daftar komponen data yang ada pada data yang digunakan
Deskripsi	Jenis data dalam representasi komputer untuk masing-masing data.

2.3.2. Object Oriented Analysis Design

Konsep OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang memberikan requirement (syarat/keperluan) yang harus dipenuhi sebuah sistem) dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup sistem. Sedangkan OOD adalah metode untuk mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek sistem atau subsistem [21].

2.3.2.1. Unified Modeling Language (UML)

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO).

UML merupakan standar yang relatif terbuka yang dikontrol oleh Object Management Group (OMG), sebuah konsorsium terbuka yang terdiri dari banyak perusahaan. OMG dibentuk untuk membuat standar-standar yang mendukung interoperabilitas, khususnya interoperabilitas sistem berorientasi objek. OMG

mungkin lebih dikenal dengan standar-standar CORBA (Common Object Request Broker Architecture) [22].

UML terdiri dari 13 jenis diagram resmi seperti tertulis dalam Tabel 2.5 jenis diagram resmi UML [22].

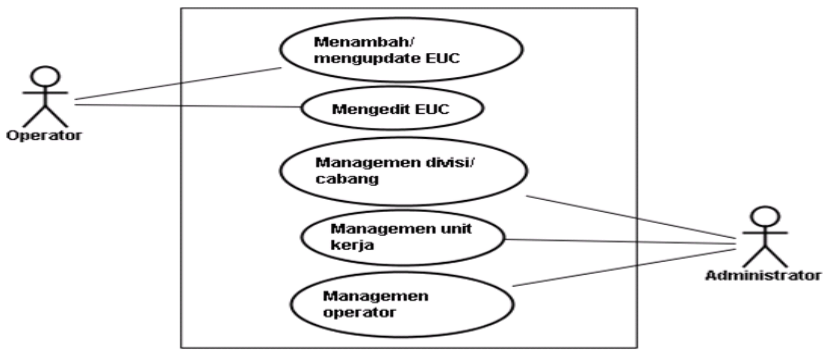
Tabel 2.5 Jenis Diagram Resmi UML [22]

Diagram	Kegunaan
Activity	Behavior prosedural dan paralel
Class	Class, fitur, dan hubungan-hubungan
Communication	Interaksi antar objek; penekanan pada jalur
Component	Struktur dan koneksi komponen
Composite structure	Dekomposisis runtime sebuah class
Deployment	Pemindahan artifak ke node
Interaction overview	Campuran sequence dan activity diagram
Object	Contoh konfigurasi dari contoh-contoh
Package	Struktur hirarki compile-time
Sequence	Interaksi antar objek; penekanan pada sequence
State machine	Bagaimana event mengubah objek selama aktif
Timing	Interaksi antar objek; penekanan pada timing
Use case	Bagaimana pengguna berinteraksi dengan sebuah sistem

Adapun penjelasan dari Use Case Diagram, Activity Diagram Class Diagram, dan Sequence Diagram adalah sebagai berikut [22]:

1) Use Case Diagram

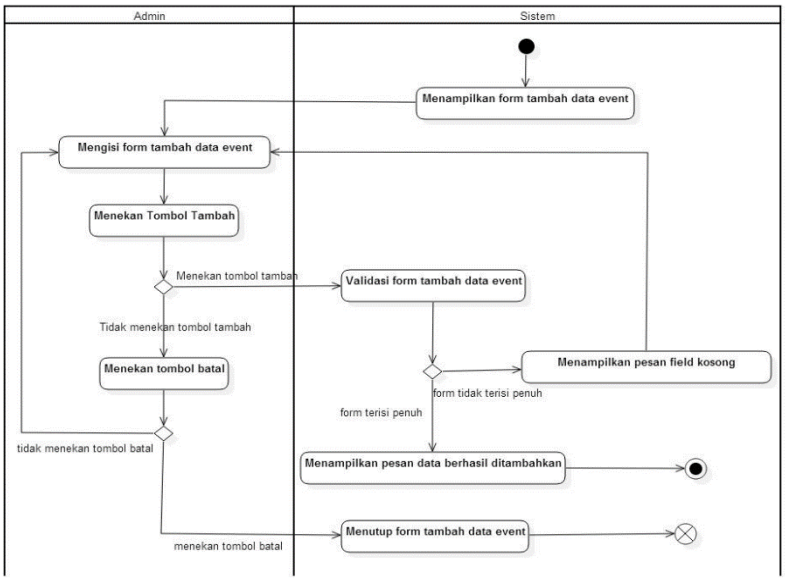
   Use case diagram menggambarkan sejumlah eksternal aktor dan hubungannya ke use case yang diberikan oleh sistem.



Gambar 2.5 Contoh Use Case Diagram [22]

2) Activity Diagram

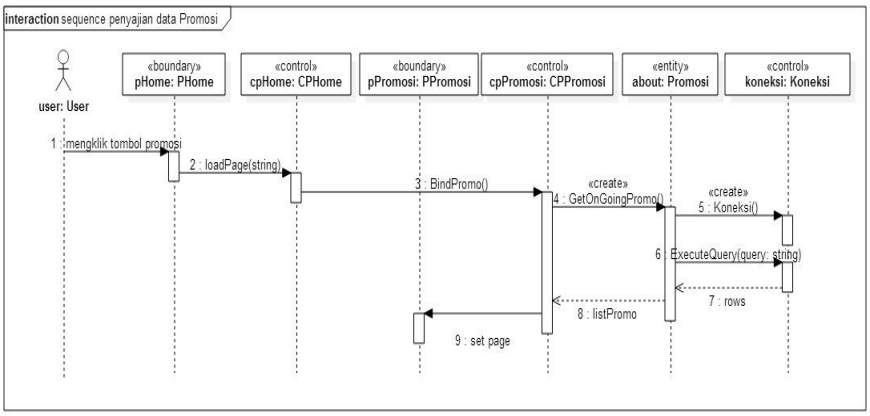
Activity diagram menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktivitas lainya seperti use case.



Gambar 2.6 Contoh Activity Diagram [22]

3) Sequence Diagram

Sequence Diagram menggambarkan kolaborasi dinamis antara sejumlah objek dan untuk menunjukan rangkaian pesan yang dikirim antara objek juga interaksi. Sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.



Gambar 2.7 Contoh Sequence Diagram [22]

4) Class Diagram

Class diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antaranya. Class diagram juga

menunjukan properti dan operasi sebuah class dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. Kotak-kotak yang terdapat di dalam diagram merupakan class, yang dibagi menjadi tiga bagian : nama class (cetak tebal), atributnya dan operasinya.

a. Properti

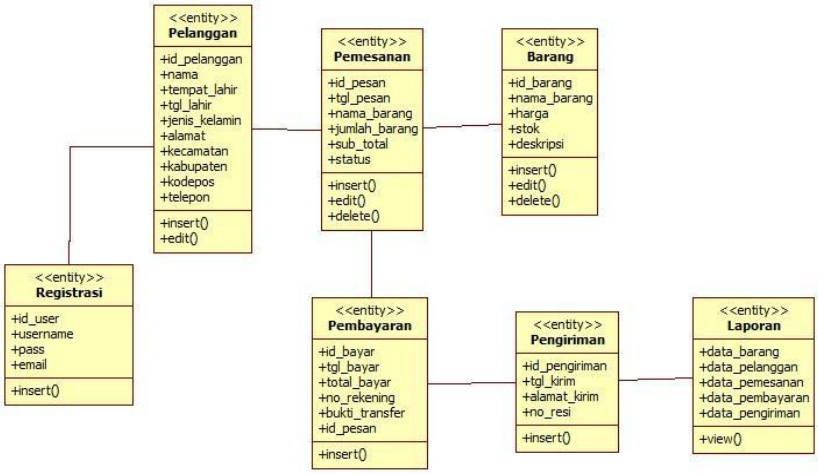
   Properti mewakili fitur-fitur struktural dari sebuah class. Properti merupakan sebuah konsep tunggal, tetapi tampak seperti dua notasi yang sedikit berbeda: atribut dan asosiasi.

b. Atribut

   Notasi atribut mendeskripsikan properti dengan sebaris teks di dalam kotak class tersebut.

c. Operasi

   Operasi merupakan suatu kegiatan yang dimengerti sebuah class untuk dilakukan. Operasi secara jelas berhubungan dengan metode dalam sebuah class.



Gambar 2.8 Contoh Class Diagram [22]

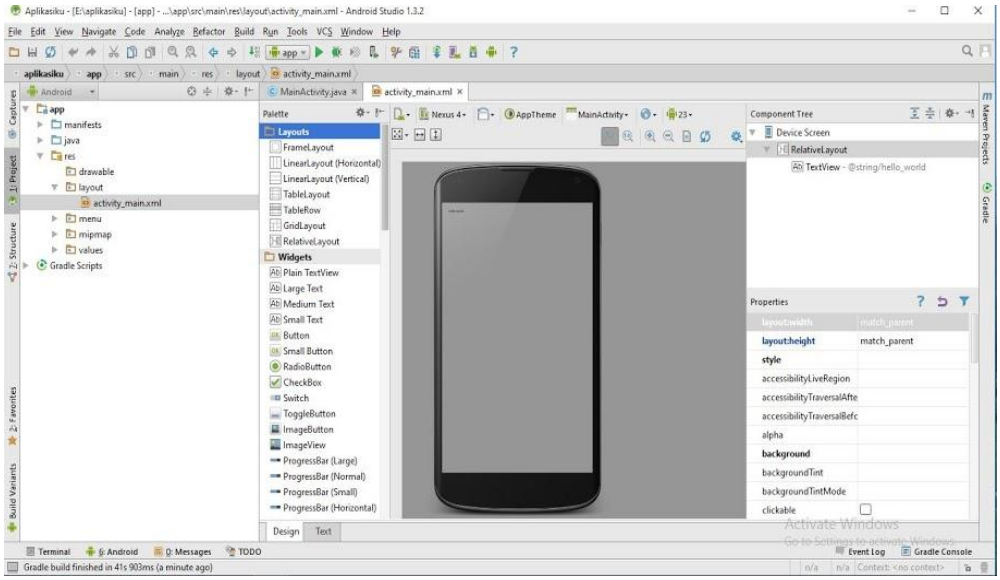
2.4. Alat-alat Pembangunan Perangkat Lunak

Alat-alat pembangunan perangkat lunak membahas mengenai alat atau tools apa saja yang digunakan dalam pembangunan perangkat lunak dalam penelitian ini.

2.4.1. Android Studio

Android studio adalah lingkungan pengembangan terpadu (*IDE)* resmi untuk pengembangan platform android, hal itu disampaikan pada tanggal 16 mei 2013 Google I/O,android studio sudah tersedia secara bebas dibawah lisensi Apache 2.0.

Android studio pada awalnya tahap *preview* versi 0.1 yang dipakai pada tanggal 1 mei 2013 dan memasuki tahap beta pada bulan juni 2014 dan mulai stabil dirilis pada Desember 2014 dengan versi 1.0 ,berdasarkan *jetBrains* 'IDEA *Intellij Software*, android studio dirancang khusus untuk pengembangan android yang tersedia untuk Windows,Mac OS X dan Linux sebagai pengganti *Eclipse* [23].



Gambar 2.9 Layout Android Studio [23]

2.4.1.1. Struktur Android Studio

Berikut adalah rincian dari struktur android studio dapat dijelaskan sebagai berikut :

1. App : berfungsi untuk menyimpan file file pendukung aplikasi yang akan diolah.
   a. *Manifest*

      Tempat dimana konfigurasi konfigurasi pendukung project serta izin (permision) untuk mengakses hardware.

b. *Java*

Tempat berdiamnya *code/scripts* java untuk menjalankan aplikasi yang akan dibuat.

c. *Res*

- *Drawable* : tempat untuk menyimpan gambar atau *icon* pendukung aplikasi.
- *Layout* : tempat berkumpulnya design tampilan yang akan tampak pada tampilan aplikasi anda.
- *Menu* : berisi *code* berupa *file* xml yang didalamnya berisi *array* tentang menu yang akan tampil dalam *project* anda, misalkan *file*, *new*,*open*,*close*,dsb.
- *Mipmap* : tempat berkumpulnya *icon launcher* (icon utama aplikasi) yang ukurannya menyesuaikan *device* yang digunakan.
- *Values* : untuk menyimpan file .xml yang menggambarkan sumber dari berbagai jenis. Sebagai contoh folder berisi *file strings*.xml dengan definisi sumber *string*. Seperti yang akan anda lihat dari menambahkan bagian *color* , *folder* tata letak juga dapat berisi, misalnya, deskripsi warna. Di dalam file dimens.xml, terdapat *string-string* untuk konfigurasi dimensi aplikasi. Tidak jauh berbeda dengan dimens.xml maupun *strings*.xml, pada file styles.xml terdapat deklarasi variable untuk theme.

2. *Gradle Scripts* : berfungsi sebagai *configurasi* aplikasi anda dengan *device*, serta menghubungkan *library* pendukung aplikasi yang anda buat. Penggunaan android studio ini untuk membuat aplikasi rumah pintar di *smartphone* android yang sebagai perangkat bantuan untuk mengontrol rumah pintar [23].

2.4.2. Java

Java menurut definisi dari Sun adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer standalone ataupun pada lingkungan jaringan. Java dikembangkan pada bulan Agustus 1991,

dengan nama semula Oak. Pada Januari 1995, karena nama Oak dianggap kurang komersial, maka diganti menjadi Java. Pada Desember 1998, Sun memperkenalkan nama "Java 2" (J2) sebagai generasi kedua dari java platform. Konvensi nama baru ini diterapkan untuk semua edisi Java yaitu Standard Edition (J2SE), Enterprise Edition (J2EE), dan Micro Edition (J2ME) [24].

Ada tiga platform Java yang masing-masing diarahkan untuk tujuan tertentu dan untuk lingkungan komputasi yang berbeda-beda:

1. Standard Edition (J2SE) : J2SE merupakan inti dari bahasa pemrograman Java. J2SE didesain untuk jalan pada komputer desktop dan komputer workstations.
2. Enterprise Edition (J2EE) : Dengan built-in mendukung untuk servlets. JSP, dan XML, edisi ini ditujukan untuk aplikasi berbasis server.
3. Micro Edition (J2ME) : Didesain untuk piranti dengan memori terbatas, layar display terbatas dan power pemrosesan yang juga terbatas.

2.4.2.1. Kelebihan Java

Adapun beberapa kelebihan dari bahasa pemrograman Java adalah sebagai berikut [24] :

1. Sederhana dan Ampuh

   Java dirancang untuk mudah dipelajari, terutama bagi programmer yang telah mengenal C/C++ akan mudah sekali untuk berpindah ke Java. Pemakai dapat belajar membuat program dengan Java secara cepat jika telah memahami konsep dasar pemrograman berorientasi objek. Java tidak memiliki hal-hal yang mengejutkan dan aneh. Java memberi anda kemampuan untuk menuangkan semua ide, karena bahasa pemrograman ini bukan merupakan scripting language (bahasa naskah) yang menghilangkan kemampuan kita untuk berinovasi, tetapi dengan cara berorientasi objek yang mudah dan jelas.
2. Aman

   Java dirancang sebagai bahasa pemrograman yang handal dan aman. Aplikasi-aplikasi yang dibangun dengan bahasa Java sangat handal dengan manajemen memori yang bagus. Aplikasi Java juga dikenal sangat secure,

yaitu kasus-kasus seperti buffer everflow yang umumnya menjadi lubang keamanan aplikasi-aplikasi berbasis C/C++ tidak terjadi di Java, karena pengaturan securitynya yang bagus.

3. Berorientasi Objek

   Paradigma pemrograman berorientasi objek merupakan paradigma pemrograman masa depan. Java merupakan bahasa pemrograman berorientasi objek. Java bukan turunan langsung dari bahasa pemrograman manapun, juga sama sekali tidak kompetibel dengan semuanya. Java memiliki keseimbangan, Java secara cepat jika telah memahami konsep dasar pemrograman berorientasi objek. Java tidak memiliki hal-hal yang mengejutkan dan aneh. Java memberi anda kemampuan untuk menuangkan semua ide, karena bahasa pemrograman menyediakan mekanisme peng-class-an sederhana, dengan model antar muka dinamik yang intuitif hanya jika diperlukan.

4. Kokoh

   Java membatasi anda dari beberapa hal kunci supaya anda dapat menemukan kesalahan lebih cepat saat mengembangkan program. Java langsung memeriksa program saat anda menuliskannya, dan sekali lagi ketika program di jalankan. Karena Java adalah bahasa yang sangat ketat dalam hal tipe data dan deklarasi, banyak kesalahan umum terjadi saat kompilasi. Hal ini akan lebih menghemat waktu jika dibandingkan dengan keharusan menjalankan program terlebih dahulu dan memeriksa semua bagian program untuk melihat ketidakcocokan dinamis selama program berjalan. Ini adalah contoh di mana Java lebih luwes dan kokoh dari beberapa bahasa lain, tetapi dengan imbalan yang layak untuk kelebihan itu.

5. Interaktif

   Java memiliki beberapa kemampuan yang memungkinkan program melakukan beberapa hal pada saat bersamaan, tanpa harus kesulitan menangani proses yang akan terjadi selanjutnya. Jalinan program-program Java yang mudah digunakan memungkinkan kita untuk memikirkan

pembuatan perilaku khusus, tanpa harus mengintegrasikan perilaku tersebut dengan model pemrograman global yang mengatur perulangan kejadian.

6. Netral terhadap Berbagai Arsitektur

   Java telah mengambil beberapa keputusan yang sulit dalam pembuatan bahasa Java dan bagaimana program dijalankan, jadi anda dapat sepenuhnya percaya "tulis sekali, jalan di mana saja, kapan saja, selamanya".

7. Terinterpretasi dan Berkinerja Tinggi

   Java dilengkapi keajaiban lintas platform yang luar biasa dengan kompilasi ke dalam representasi langsung yang disebut kode-byte Java (Java byte-code), yang dapat diterjemahkan oleh sistem manapun yang memilki program Java didalamnya. Java, bagaimanapun dirancang untuk tetap berkinerja baik pada CPU yang tidak terlalu kuat. Walaupun Java merupakan bahasa terinterpretasi, kodekode Java telah dirancang dengan hati-hati sehingga mudah diterjemahkan ke dalam bahasa asli suatu mesin untuk menghasilkan kinerja yang tinggi.

   Sistem program Java yang melakukan optimasi tepat waktu tersebut tidak kehilangan keuntungan dari program yang netral terhadap platform. "lintas platform berkinerja tinggi" bukan sekedar omong-kosong. Dalam aplikasi Java (*.class) merupakan Java bytecode yang berjalan di atas jvm (Java Virtual Machine), yang kemudian jvm-lah yang akan menginterpresentasikan kode-kode tersebut ke kode native atau kode mesin dari arsitektur yang bersangkutan. Hal sangat menarik karena urusan arsitektur mesin bukan jadi masalah bagi programmer tapi menjadi urusan kompiler pada bahasa pemrograman Java.

2.4.3. JavaScript Object Notation (JSON)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa

pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data [25].

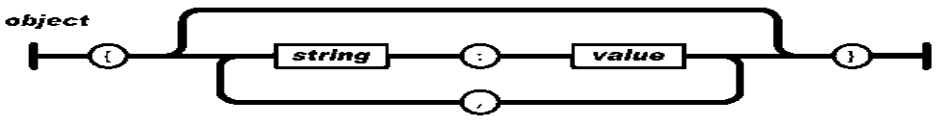
JSON terbuat dari dua struktur [25] :

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau associative array.
2. Daftar nilai terurutkan (*an ordered list of value*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai lirik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemrograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini [25].

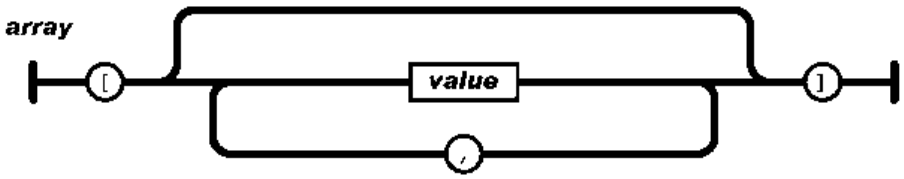
JSON menggunakan bentuk sebagai berikut [25] :

Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



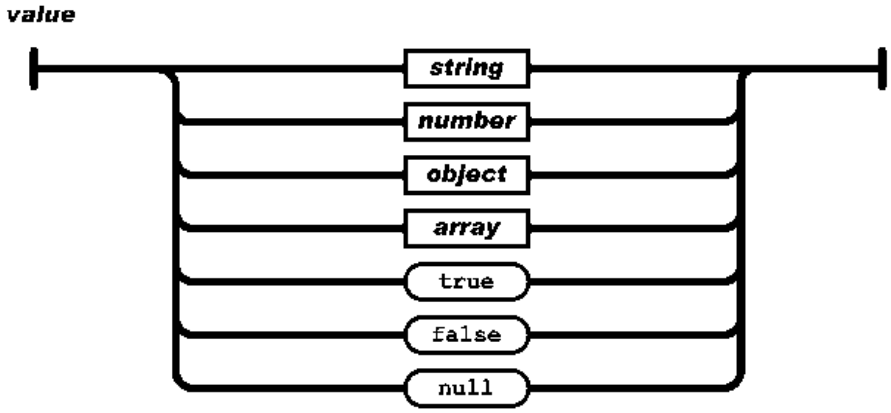
Gambar 2.10 Objek JSON [25]

Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).



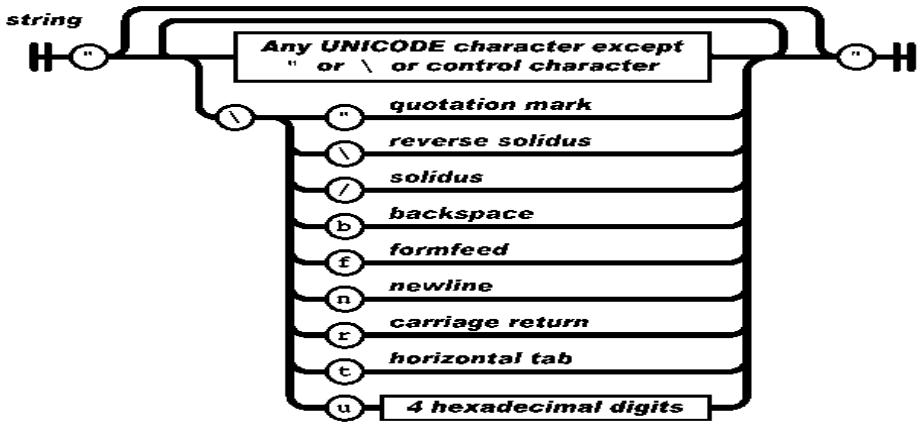
Gambar 2.11 Array JSON [25]

Nilai (value) dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.



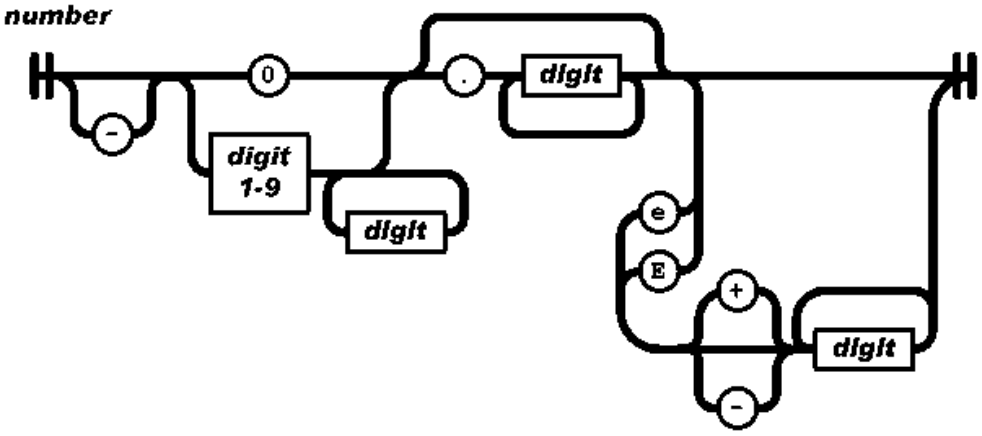
Gambar 2.12 Value JSON [25]

String adalah kumpulan dari nol atau lebih karakter Unicode, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan backslash escapes "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java.



Gambar 2.13 String JSON [25]

Angka adalah sangat mirip dengan angka di C atau Java, kecuali format oktal dan heksadesimal tidak digunakan.



Gambar 2.14 Number JSON [25]

Spasi kosong (whitespace) dapat disisipkan di antara pasangan tanda-tanda tersebut, kecuali beberapa detil encoding yang secara lengkap dipaparkan oleh bahasa pemrograman yang bersangkutan.

2.4.4. Processor Hypertext Protocol (PHP)

PHP adalah suatu bahasa *server-side* yang didesain khusus untuk aplikasi web. PHP dapat disisipkan diantara HTML, dan karena bahasa *server-side*, maka bahasa PHP akan dieksekusi di server, yang dikirimkan ke browser adalah "hasil jadi" bentuk HTML, dan kode PHP tidak akan terlihat. PHP dahulu merupakan proyek pribadi dari Rasmus Lerdorf tahun 1994 (dengan dikeluarkannya PHP versi 1 yang digunakan untuk membuat home page pribadi yang dipublikasikan 8 juni 1995). Versi pertama ini berupa kumpulan script PERL. Untuk versi kedua Ramus menulis ulang script-script PERL tersebut menggunakan bahasa C, kemudian menambahkan fasilitas untuk form HTML dan koneksi MYSQL. Adapun PHP didapat dari singkatan *PHP Hypertext Preprocessor*.

Setelah mengalami perkembangan, Zeev Suraski dan Andi Gutmans (termasuk Rasmus) yang tergabung dalam suatu kelompok open source, maka mereka memulai versi 3 PHP nya, PHP telah menampakkan keunggulan sebagai salah satu bahasa *server scripting* yang handal. Melalui perkembangan yang pesat ini banyak fasilitas yang ditambahkan oleh kelompok ini PHP disebut sebagai

"PHP: Hypertext Preprocessor". Pada bulan Mei 2000 dirilis PHP versi 4 yang dikeluarkan oleh Zend Engine yaitu sebuah produk yang diciptakan oleh Suraski dan Gutmans. Kehandalan PHP diperlengkap dengan dirilisnya PHP versi 5 pada 13 Juli 2004 yang mendukung OOP serta berbagai manajemen database yang luas dan penanganan error yang mudah. Sintak yang digunakan berasal dari bahasa C, Java maupun PERL. Sampai tulisan ini dibuat versi PHP yang terbaru adalah versi 5.2.0 yang dirilis pada November 2006.

PHP merupakan bahasa script yang digunakan untuk membuat halaman web yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru. Semua script PHP dieksekusi pada server dimana script tersebut dijalankan. Oleh karena itu, spesifikasi server lebih berpengaruh pada eksekusi dari script php daripada spesifikasi client. Namun tetap diperhatikan bahwa web yang dihasilkan tentunya harus dapat dibuka oleh browser pada client. Dalam hal ini versi dari html yang digunakan harus didukung oleh browser client. PHP termasuk dalam *Open Source Product*. Jadi dapat menambah *source code* dan dapat mendistribusikannya secara bebas. PHP juga diedarkan secara gratis. Kita bisa mendapatkannya secara gratis. PHP juga dapat berjalan diberbagai web server semisal IIS, Apache, PWS.

Dengan demikian kode program yang ditulis dalam PHP tidak akan terlihat oleh user sehingga keamanan halaman web lebih terjamin. PHP dirancang untuk membuat halaman web yang dinamis, yaitu halaman web yang dapat membentuk suatu tampilan berdasarkan permintaan terkini, seperti menampilkan isi basis data ke halaman web [26].

```
<?php
        header('Access-Control-Allow-Origin: *');
        require_once("../class/about.php");
        $obj = new About();
        echo json_encode($obj);
?>
```

Gambar 2.15 Contoh Skript PHP [26]

2.4.5. Database MySQL

Database adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. MySQL adalah sebuah perangkat lunak system manajemen basis data SQL (DBMS) yang multithread, dan multi-user. MySQL adalah implementasi dari system manajemen basis data relasional (RDBMS). MySQL dibuah oleh TcX dan telah dipercaya mengelola system dengan 40 buah database berisi 10.000 tabel dan 500 di antaranya memiliki 7 juta baris. MySQL AB merupakan perusahaan komersial Swedia yang mensponsori dan yang memiliki MySQL.

MySQL AB adalah dua orang Swedia yang bernama David Axmark, Allan Larsson dan satu orang Finlandia bernama Michael "Monty". Setiap pengguna MySQL dapat menggunakannya secara bebas yang didistribusikan gratis dibawah lisensi GPL (General Public License) namun tidak boleh menjadikan produk turunan yang bersifat komersial.

MySQL adalah salah satu jenis database server yang sangat terkenal dan banyak digunakan untuk membangun aplikasi web yang menggunakan database sebagai sumber dan pengolahan datanya. MySQL dikembangkan oleh perusahaan swedia bernama MySQL AB yang pada saat ini bernama Tcx DataKonsult AB sekitar tahun 1994-1995, namun cikal bakal kodenya sudah ada sejak tahun 1979. Awalnya Tcx merupakan perusahaan pengembang software dan konsultan database, dan saat ini MySQL sudah diambil alih oleh Oracle Corp.

Kepopuleran MySQL antara lain karena MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses databasenya sehingga mudah untuk digunakan, kinerja query cepat, dan mencukupi untuk kebutuhan database perusahaan-perusahaan yang berskala kecil sampai menengah, MySQL juga bersifat open source (tidak berbayar). MySQL merupakan database yang pertama kali didukung oleh bahasa pemrograman script untuk internet (PHP dan Perl). MySQL dan PHP dianggap sebagai pasangan software pembangun aplikasi web yang ideal. MySQL lebih sering digunakan untuk membangun aplikasi berbasis

web, umumnya pengembangan aplikasinya menggunakan bahasa pemrograman script PHP [26].

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. SQL dapat digunakan secara berdiri sendiri maupun di lekatkan pada bahasa pemrograman seperti C, dan Delphi.

Elemen dari SQL yang paling dasar antara lain pernyataan, nama, tipe data, ekspresi, konstanta dan fungsi bawaan. Pernyataan Perintah dari SQL yang digunakan untuk meminta sebuah tindakan kepada DBMS.

Sistem database menyediakan bahasa untuk mendefinisikan database Data Definition Language (DDL) serta bahasa untuk memanipulasi database Data Manipulation Language (DML) untuk melakukan operasi-operasi tertentu pada database dan juga Data Control Language (DCL).

a. Data Definition Language (DDL)

   Kumpulan perintah SQL yang digunakan untuk membangun database, bisa juga memiliki arti lain perintah untuk membuat struktur dasar database dan tabel. Perintah-perintah yang termasuk DDL adalah statment connect, create, show, describe, drop, rename, dan alter.

b. Data Manipulation Language (DML)

   Kumpulan statement yang digunakan sebagai perintah untuk mengelola data dalam sebuah tabel. Perintah DML digunakan setelah menggunakan DDL antara lain seperti statement insert, update, delete, dan select.

c. Data Control Language (DCL)

Perintah yang digunakan untuk keperluan keamanan (security) database dengan membuat hak akses tertentu bagi setiap user. Statment pada DCL antara lain grant, revoke, set, dan lock table.

Dalam MySQL ada beberapa operasi dasar yang lebih dikenal dengan operasi CRUD sebagai berikut:

1. *Create*

   Create adalah operasi penambahan data baru kedalam table. Terdapat 2 Query untuk menambah data, yang pertama adalah penambahan yang tidak menspesifikasikan nama kolom yang akan ditambahkan dan hanya memberikan isi dari tabelnya. Query nya adalah sebagai berikut:

   INSERT INTO table_name VALUES (value1,value2,value3,...);

   Untuk yang kedua adalah menginputkan data dengan mencantumkan nama kolom yang akan diisikan dengan isi di dalamnya. Query nya adalah sebagai berikut:

   INSERT INTO table_name (column1,column2,column3,...)
   VALUES (value1,value2,value3,...);

2. *Read*

   Read adalah operasi untuk menampilkan semua atau sebagian data yang berada di dalam database. Query nya adalah sebagai berikut:

   SELECT column_name,column_name
   FROM table_name;

3. *Update*

   Update adalah operasi untuk mengubah data yang ada di dalam database. Query nya adalah sebagai berikut:

   UPDATE table_name SET column1=value1,column2=value2,...
   WHERE some_column=some_value;

4. *Delete*

   Delete adalah operasi untuk menghapus data yang ada di dalam table. Query nya adalah sebagai berikut:

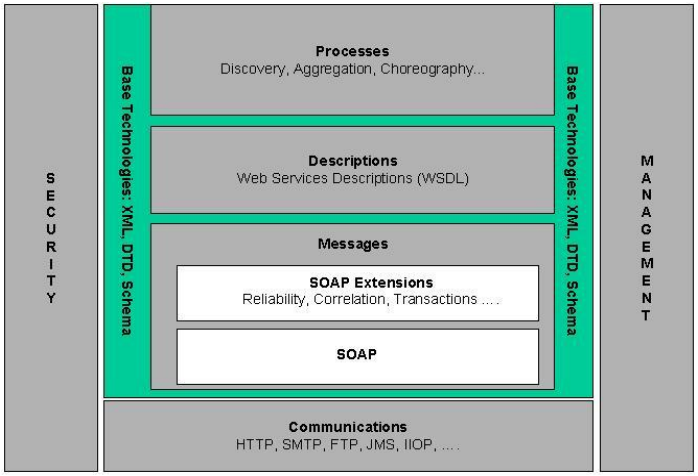
   DELETE FROM table_name

WHERE some_column=some_value;

2.4.6. Web Service

Web service adalah sebuah entitas komputasi yang dapat diakses melalui jaringan internet maupun intranet dengan standar protokol tertentu dalam *platform* dan antarmuka bahasa pemograman yang independen. Tujuan pengembangannya adalah untuk menjembatani komunikasi antar program, sehingga aplikasi yang satu dan aplikasi yang lain yang terdapat pada suatu jaringan yang sama atau pada jaringan berbeda dapat saling berkomunikasi asalkan menggunakan standart protokol yang ditetapkan oleh *web service*. Hal ini bisa terjadi, karena standart protokol itu tidaklah terikat pada suatu *platform* atau bahasa pemograman. Protokol itu sendiri dibangun oleh *Extensible Markup Language* (XML) yang memang kenyataannya telah didukung oleh banyak *paltform*, bahasa pemograman, dan oleh *developer* di seluruh dunia.

Arsitektur *web service* dibangun oleh beberapa layer dan teknologi yang saling berhubungan. Banyak cara untuk memvisualisaskan *service*, sama banyaknya dengan cara untuk membangun dan menggunakan *web service* gambar berikut hanyalah salah satu cara untuk menggambarkan arsitekturnya.



Gambar 2.16 Arsitektur Web Service [27]

Web service disusun oleh tiga komponen standart, yaitu :

1. *Simple Object Access Protocol* (SOAP), yaitu protokol yang bertanggung jawab dalam pertukuran informasi dalam lingkungan jaringan terdistribusi.

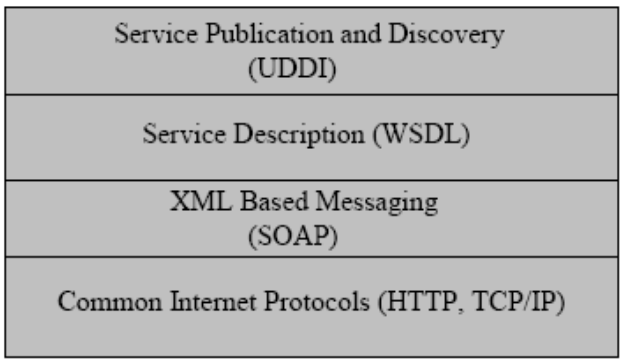
2. *Web Service Definition Language* (WSDL), dokumen standart yang dituliskan pada format XML, dan mendefinisikan kehadiran web service dalam suatu jaringan.
3. *Universal Description, Discovery and Integration* (UDDI), yaitu suatu lokasi direktori yang beriskan *service* (layanan) dan bersifat bebas *platform* (*platform independent*), dituliskan berbasis XML dan dapat diakses oleh entittas yang berada di dalam dan luar jaringan.

Adanya standart tersebut membuat web *service* mudah di akses melalui berbagai antarmuka dan juga memberi peluang dimungkinkannya berbagai sistem yang dibangun pada platform yang berbeda dan bahasa yang berbeda untuk berkolaborasi pada dalam suatu pekerjaan [27].

2.4.6.1. Komponen – Komponen Web Service

Web Service secara keseluruhan memiliki empat layer komponen yaitu sebagai berikut :

1. Layer 1 : Protokol internet standar seperti HTTP, TCP/IP
2. Layer 2 : *Simple Object Access Protocol* (SOAP), merupakan protokol akses objek berbasis XML yang digunakan untuk proses pertukaran data/informasi antar layanan.
3. Layer 3 : *Web Service Definition Language* (WSDL), merupakan suatu standar bahasa dalam format XML yang berfungsi untuk mendeskripsikan seluruh layanan yang tersedia.



Gambar 2.17 Komponen – Komponen Web Service [27]