BAB II
TINJAUAN PUSTAKA

Bagian ini berisi tentang beberapa teori-teori yang mendukung dalam proses analisis dan implementasi berdasarkan masalah yang diangkat dalam pembangunan Aplikasi Safety Way Guide berbasis Android.

2.1 Sistem Operasi Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, middleware dan aplikasi. Android menyediakan sistem operasi yang terbuka bagi para pengembang untuk menciptakan aplikasi Android diluncurkan untuk umum pada tahun 2008. Android sangat berkembang pesat di industri karena dua aspek utama yaitu bersifat opensource dan model arsitekturya. Sebagai sebuah proyek yang bersifat opensource, memungkinkan Android untuk sepenuhnya dipahami dan dianalisis mengenai fitur, penyelesaian pada bug program hingga hardware [4].

Aplikasi Android terdiri dari komponen yang bebas digabungkan, diikat menggunakan suatu project manifest yang menggambarkan masing-masing komponen dan bagaimana komponen-komponen itu saling berhubungan. Ada enam komponen dalam Android untuk membangun suatu aplikasi yaitu [4]:

1. *Activities*

Activity merupakan bagian yang paling penting dalam sebuah aplikasi, karena *activity* menyajikan tampilan visual program yang sedang dijalankan oleh user. Setiap *activity* dideklarasikan dalam sebuah *class* yang bertugas untuk menampilkan antarmuka yang terdiri dari *views* dan *respons* terhadap *event*.

Setiap aplikasi mempunyai satu buah *activity* atau lebih. Biasanya pasti akan ada *activity* yang pertama kali tampil ketika aplikasi sedang dijalankan. Perpindahan antara *activity* dengan *activity* lainnya diatur melalui sistem, dengan memanfaatkan *activity stack*. Keadaan suatu *activity* ditentukan oleh posisinya pada *activity stack*. Bila suatu *activity* baru dimulai, *activity* yang sebelumnya digunakan maka akan

dipindahkan ke tumpukan paling atas. Ketika *activity* dipanggil dan disimpan dalam *activity stack* terdapat 4 kemungkinan kondisi transisi yang akan terjadi [4]:

a. *Active*

Setiap *activity* yang berada di tumpukan paling atas, maka *activity* tersebut akan terlihat dan menerima masukan dari *user*. Android akan membuat *activity* ini tetap hidup dengan menghentikan *activity* yang berada dibawah tumpukan jika diperlukan. Ketika *activity* sedang aktif, maka yang lainnya akan dihentikan sementara.

b. *Paused*

Beberapa *activity* akan terlihat tapi tidak terfokus pada kondisi ini disebut *paused*. Keadaan ini terjadi jika *activity* transparan dan tidak *fullscreen* pada layar. Ketika *activity* dalam keadaan *paused*, dia terlihat aktif namun tidak bisa diakses oleh *user*.

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. Namum akan di eksekusi oleh sistem jika sistem membutuhkan sumberdaya lebih.

d. *Inactive*

Kondisi ini ketika *activity* telah dihentikan atau sebelum dijalankan. *Inactive activity* telah ditiadakan dari *activity stack* sehinga perlu *restart* ulang agar dapat tampil dan digunakan kembali.

2. *Services*

Suatu *services* tidak memiliki tampilan antarmuka, melainkan berjalan di *background* untuk waktu yang tidak terbatas. Komponen *service* diproses tidak terlihat, memperbaharui sumber data dan menampilkan notifikasi.

3. *Content Providers*

Content providers digunakan untuk mengelola dan berbagi *database*. Data dapat disimpan dalam *file system*, dalam *database* SQLite, atau dengan cara lain yang pada prinsipnya sama. Dengan adanya *content provider* memungkinkan antar aplikasi untuk saling berbagi data.

4. *Intens*

Intens merupakan sebuah mekanisme untuk menggambarkan tindakan tertentu, seperti memilih foto, menampilkan halaman web, dan lain sebagainya. *Intens* tidak selalu dimulai dengan menjalankan aplikasi, namun juga digunakan oleh sistem untuk memberitahukan ke aplikasi bila terjadi sesuatu, semisal pesan masuk.

5. *Broadcast Receivers*

Broadcast receivers merupakan komponen yang sebenarnya tidak melakukan apa-apa kecuali menerima dan bereaksi menyampaikan pemberitahuan. Sebagian besar *broadcast* berasal dari sistem contohnya baterai sudah hampir habis, informasi zona waktu sudah berubah, atau user telah mengganti bahasa *default* pada perangkat Android. Sama halnya dengan *service*, *broadcast receivers* tidak menampilakan antarmuka *user*. Namun, *broadcast receivers* dapat menggunakan *notification* manager untuk memberitahukan sesuatu kepada *user*.

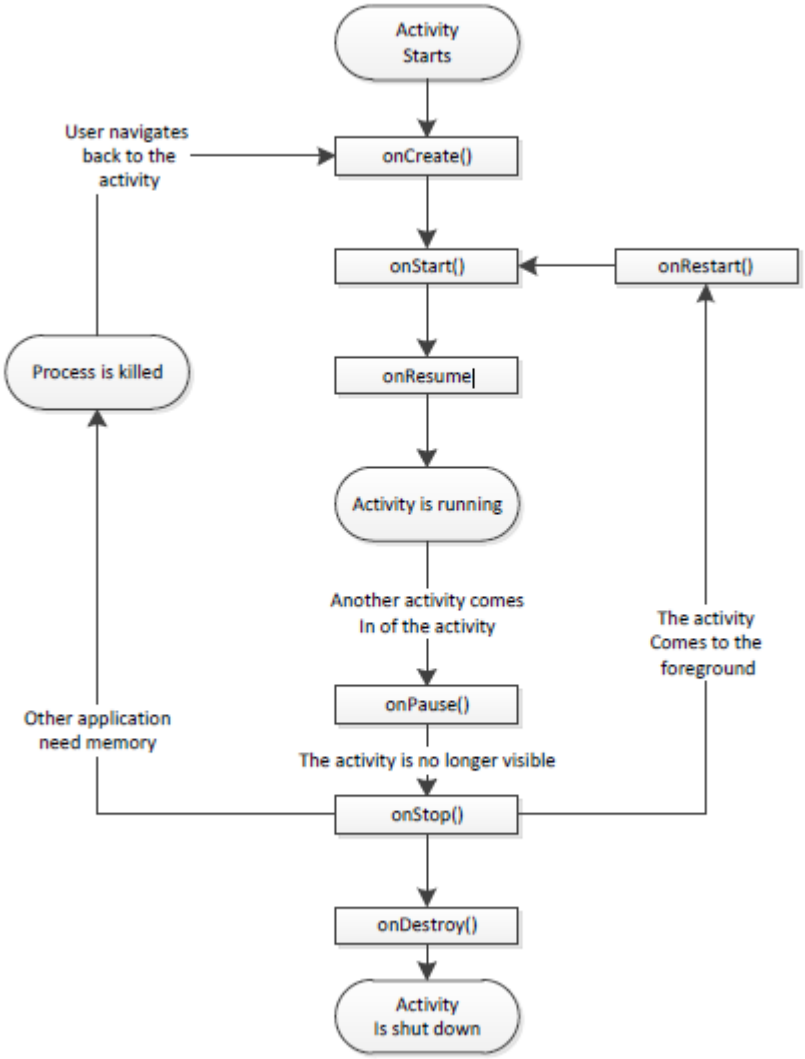
6. *Notifications*

Notifications merupakan kerangka pemberitahuan untuk *user*. *Notifications* memberikan isyarat atau peringatan tanpa mengganggu aktivitas yang sedang dilakukan oleh user.

2.1.1 Siklus Hidup Aplikasi Android

Siklus hidup aplikasi Android dikelola oleh sistem, berdasarkan kebutuhan user dan sumber daya yang tersedia. Sistem sangat berperan dalam menentukan apakah aplikasi dijalankan, dihentikan sementara, atau dihentikan sama sekali. Jika user sedang menjalankan sebuah activity, maka sistem akan memprioritaskan

aplikasi tersebut. Sebaliknya, jika suatu activity tidak terlihat dan sistem membutuhkan sumber daya yang lebih, maka activity yang prioritas rendah akan ditutup[5]. Berikut adalah siklus hidup activity Android pada gambar 2.1:



Gambar 2.1 Siklus Hidup *Activity* Android [5]

Pengguna mengakses method init pada activity class, dan Android akan memanggilnya menurut waktu yang tepat [5].

1. *onCreate*( )

   Dipanggil ketika *activity* pertama kali dibuat, disini user melakukan pengaturan statis secara normal, misal membuat *view*, memasukan data dan sebagainya. selalu diikuti dengan *onStart*( ).

2. *onStart*( )

   Dipanggil sebelum *activity* terlihat oleh pengguna. Selalu diikuti dengan *onResume*( ).

3. *onResume*( )

   Dipanggil sebelum *activity* mulai berinteraksi dengan *user*. Pada titik ini, *activity* berada pada *stack activity* yang paling atas. Selalu diikuti dengan *onPause*( ).

4. *onPause*( )

   Dipanggil ketika sistem akan memulai *activity* lain. *Method* ini biasa digunakan untuk melakukan perubahan yang tidak disimpan kedalam data yang tetap, menghentikan animasi, dan hal lain yang memakai *CPU*. Diikuti *onResume*( ) jika *activity* tetap berada dilayar atau *onStop*( ) jika *activity* menjadi tidak terlihat oleh user.

5. *onStop*( )

   Dipanggil ketika *activity* tidak lagi terlihat oleh *user*. Ini mungkin terjadi karena *activity* kemudian dihancurkan atau jika *activity* lain yang sudah ada atau yang baru telah dilanjutkan. Diikuti dengan *onRestart*( ) jika *activity* kembali berinteraksi dengan user atau dengan *onDestroy*( ) jika *activity* akan ditutup.

6. *onRestart*( )

   Dipanggil setelah *activity* dihentikan, dan merupakan prioritas untuk dijalankan kembali dan selalu diikuti *onStart*( ).

7. *onDestroy*( )

   Dipanggil sebelum *activity* dihancurkan, ini merupakan panggilan terakhir yang diterima *activity*. Bisa dikatakan *activity* diakhiri (*finish*( )), atau karena sistem secara *temporary* menghancurkan *instant activity* untuk menghemat ruang.

8. *onSaveInstanceState*( )

Android akan memanggil *method* ini untuk mengizinkan *activity* menyimpan setiap contoh keadaan, seperti posisi kursor dalam sebuah *textfield*.

9. *onRestoreInstanceState*( )

   Dipanggil ketika *activity* di inisialisasi ulang sejak keadaan sebelum disimpan dari *onSaveInstanceState*( ).

Android menjalankan setiap aplikasi dalam proses secara terpisah, yang masing-masing memiliki mesin *virtual* pengolah sendiri, dengan melindungi penggunaan memori pada aplikasi. Selain itu juga Android dapat mengontrol aplikasi mana yang layak menjadi prioritas utama. Karena itu Android menjadi sangat sensitif dengan siklus hidup aplikasi dan komponen-komponennya.

2.1.2 Karakteristik Android

Andorid merupakan subset perangkat lunak untuk perangkat mobile yang meliputi sistem operasi, middleware, dan aplikasi inti yang di-release oleh Google. SDK adalah suatu tools dan API yang diperlukan untuk mengembangkan aplikasi pada platform atau lingkungan Android. Pengembangan aplikasi Android menggunakan bahasa pemrograman Java. Seperti kita ketahui, SDK ini dikembangkan oleh OHA (Open Handsate Alliance). Organisasi OHA ini terdiri atas Google, Intel, Motorola, Qualcomm, T-Mobile, dan NVIDIA.[4]

2.1.3 Perkembangan Android

Dikutip dari http://id.wikipedia.org/wiki/Android_(sistem_operasi) Android telah banyak melakukan perkembangan, berikut adalah perkembangan dari android [5] :

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 (Donuts)

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

4. Android Versi 2.1 (Éclair)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), perubahan yang dilakukan adalah pengoptimalan hardware, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML5, daftar kontak yang baru, dukungan flash untuk kamera 3,2 MP, 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)

Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, intergrasi V8 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 Android versi ini antara lain peningkatan kemampuan permainan (gaming), peningkatan fungsi copy paste, layar antar muka (User Interface) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (reverb, equalization, headphone virtualization, dan bass boost), dukungan kemampuan Near Field Communication (NFC), dan dukungan jumlah kamera yang lebih dari satu.

7. Android Versi 3.0 (HoneyComb)

Android Honeycomb dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. User Interface pada Honeycomb juga berbeda karena sudah didesain untuk tablet. Honeycomb juga mendukung multi prosesor dan juga akselerasi perangkat keras (hardware) untuk grafis. Tablet pertama yang dibuat dengan menjalankan Honeycomb adalah Motorola Xoom. Perangkat tablet dengan platform Android 3.0 akan segera hadir di Indonesia.

Perangkat tersebut bernama Eee Pad Transformer produksi dari Asus. Rencana masuk pasar Indonesia pada Mei 2011.

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)

Diumumkan ada tanggal 27 Juni 2012 meruupakan pembaruan yang meningkatkan fungsi dan kinerja antarmuka pengguna (UI). Pembaruan ini diwujudkan dalam "Proyek Butter", perbaikan ini termasuk antisipasi sentuh, triple buffering, perpanjangan waktu vsync, dan peningkatan frame rate hingga 60 fps untuk menciptakan UI yang lebih halus.

10. Android Versi 4.4 (Kitkat)

Google mengumumkan Android 4.4 KitKat dengan pembaruan antarmuka bar status dan navigasi transparan pada layar depan, dengan optimasi kinerja perangkat dengan spesifikasi yang lebih rendah, perluasan fungsionalitas bagi layanan pendengar notifikasi dengan menambahkan dukungan Bluetooth Message Access Profil (MAP).

11. Android Versi 5.0 (Lollipop)

Android Lollipop adalah versi stabil terbaru dari sistem operasi Android yang dikembangkan oleh Google, yang pada saat ini mencakup versi antara 5.0 dan 5.1. Diresmikan pada 25 Juni 2014 saat Google I / O, dan tersedia secara resmi melalui over-the-air (OTA) update pada tanggal 12 November 2014, untuk memilih perangkat yang menjalankan distribusi Android dilayani oleh Google (seperti perangkat Nexus dan Google Play edition). Kode sumbernya dibuat tersedia pada 3 November 2014.

Salah satu perubahan yang paling menonjol dalam rilis Lollipop adalah user interface yang didesain ulang dan dibangun dengan yang dalam bahasa desain disebut sebagai "material design". Perubahan lain termasuk perbaikan pemberitahuan, yang dapat diakses dari lockscreen dan ditampilkan pada banner di bagian atas screen. Google juga membuat perubahan internal untuk platform, dengan Android Runtime (ART) secara resmi menggantikan Dalvik untuk meningkatkan kinerja aplikasi, dan dengan perubahan yang ditujukan untuk meningkatkan dan mengoptimalkan penggunaan baterai, yang dikenal secara internal sebagai Project Volta.

12. Android Versi 6.0 (Marshmallow)

Android Marshmallow memperkenalkan model izin yang didesain ulang: sekarang ada hanya delapan kategori izin, dan aplikasi yang tidak lagi secara otomatis diberikan semua hak akses mereka ditentukan pada waktu instalasi. Sebuah sistem opt-in sekarang digunakan, di mana pengguna akan diminta untuk memberikan atau menolak izin individu (seperti kemampuan untuk mengakses kamera atau mikrofon) untuk aplikasi ketika mereka dibutuhkan. Aplikasi mengingat hibah izin mereka, dan mereka dapat disesuaikan oleh pengguna setiap saat. Model izin baru akan digunakan hanya oleh aplikasi yang dikompilasi untuk Marshmallow menggunakan kit pengembangan perangkat lunak (SDK) tersebut, sementara semua aplikasi lainnya akan terus menggunakan model izin sebelumnya.

2.1.4 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 system 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. 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.
3. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga kerja system menjadi lebih stabil. Pengguna tak perlu kawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.
4. Dukungan grafis dan suarat terbaik, dengan adanya dukungan 2D grafis dan animasi yang diilhami oleh Flash menyatu dalam 3D menggunakan OpenGL memungkinkan membuat aplikasi maupun game yang berbeda.
5. 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 portabel antara ARM, X86, dan arsitektur lainnya. Sama halnya dengan dukungan masukan seperti penggunaan Keyboard, layar sentuh, trackball dan resolusi layar semua dapat disesuaikan dengan program.

2.1.5 Kekurangan Android

Tak bisa dipungkiri disamping kelebihan tentu saja ada kekurangan dari sistem operasi ini beberapa kekuranganya yaitu seperti :

1. Koneksi Internet yang terus menerus, kebanyakan ponsel berbasis system ini memerlukan koneksi internet yang simultan alias terus menerus aktif. Koneksi internet GPRS selalu aktif setiap waktu, itu artinya Anda harus siap berlangganan paket GPRS yang sesuai dengan kebutuhan.

2. Iklan – Aplikasi di Ponsel Android memang bisa didapatkan dengan mudah dan gratis, namun konsekuensinya di setiap Aplikasi tersebut, akan selalu Iklan yang terpampang, entah itu bagian atas atau bawah aplikasi.

2.1.6 Android SDK

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

1. Framework : aplikasi yang mendukung pengganti komponen dan reusable.
2. Dalvik Virtual Machine dioptimalkan untuk perangkat mobile
3. Integrated Browser verdasarkan engine open source WebKit.
4. Grafis yang dioptimalkan dan didukung oleh libraries grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional Ekselerasi hardware)
5. SQLite untuk penyimpanan data.
6. Media Support yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PING, GIF), GSM Telephony (tergantung Hardware)
7. Bluetooth, EDGE, 3G, dan WiFi (tergantung hardware)
8. Kamera, GPS, Kompas, dan Accelerometer (tergantung hardware)
9. Lingkungan Development yang lengkap dan termasuk pernagkat emulator, tools untuk debugging, profil dan kinerja memori, dan plugin untuk IDE Android Studio.

2.1.7 Android Studio

Android Studio adalah sebuah IDE untuk Android Development yang diperkenalkan google pada acara Google I/O 2013. Android Studio merupakan

pengembangkan dari Eclipse IDE, dan dibuat berdasarkan IDE Java populer, yaitu IntelliJ IDEA. Android Studio merupakan IDE resmi untuk pengembangan aplikasi Android.

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

1. Menggunakan Gradle-based build system yang fleksibel.
2. Bisa mem-build multiple APK .
3. Template support untuk Google Services dan berbagai macam tipe perangkat.
4. Layout editor yang lebih bagus.
5. Built-in support untuk Google Cloud Platform, sehingga mudah untuk integrasi dengan Google Cloud Messaging dan App Engine.
6. Import library langsung dari Maven repository

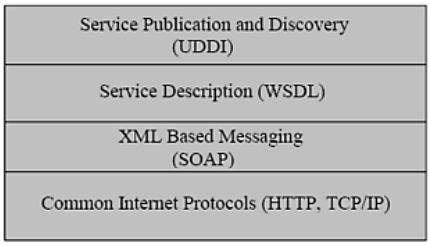
Satu hal tambahan lagi yang membuat Android Studio unggul adalah dukungan layout xml editor secara visual yang jauh lebih baik daripada Eclipse.

2.2 Web Services

Web service merupakan suatu komponen software yang merupakan selfcontaining, aplikasi modular self-describing yang dapat dipublikasikan, dialokasikan, dan dilaksanakan pada web. Web service adalah teknologi yang mengubah kemampuan internet dengan menambahkan kemampuan transactional web, yaitu kemampuan web untuk saling berkomunikasi dengan pola program-to-program (P2P). Fokus web selama ini didominasi oleh komunikasi program-to-user dengan interaksi business-to-consumer (B2C), sedangkan transactional web akan didominasi oleh program-to-program dengan interaksi business-to-business.

Gambar 2.2 merupakan blok bangunan web service yang mana menyediakan fasilitas komunikasi jarak jauh antara dua aplikasi yang merupakan layer arsitektur web service.

1. Layer 1 : Protokol internet standar yang digunakan sebagai sarana transportasi adalah HTTP dan TCP/IP.
2. Layer 2 : Simple Object Access Protocol (SOAP) berbasiskan XML dan digunakan untuk pertukaran informasi antar sekelompok layanan.
3. Layer 3 : Web service Definition Language (WSDL) digunakan untuk mendiskripsikan attribute layanan.
4. Layer 4 : Universal Description, Discovery and Integration, yang mana merupakan direktori pusat untuk deskripsi layanan.



Gambar 2.2 Blok Bangunan *Web Service*

2.3 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 pemprograman 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 [7].

JSON terbuat dari dua struktur [7] :

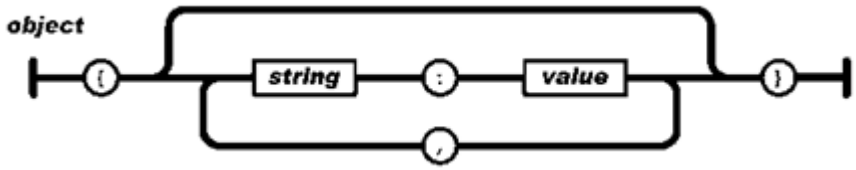
1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau associative array.

2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini [7].

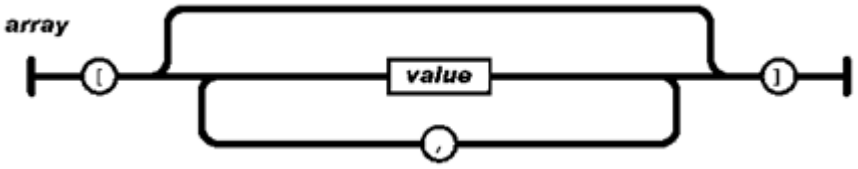
JSON menggunakan bentuk sebagai berikut [7] :

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).



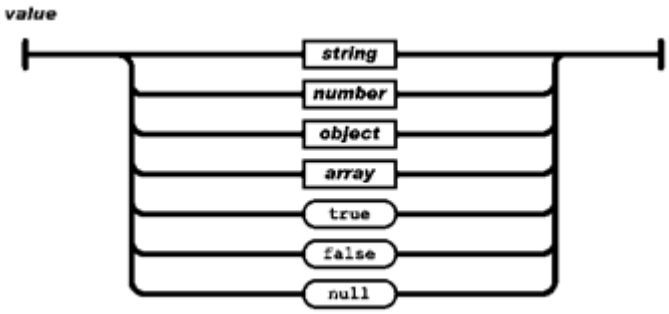
Gambar 2.3 Objek JSON[7]

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



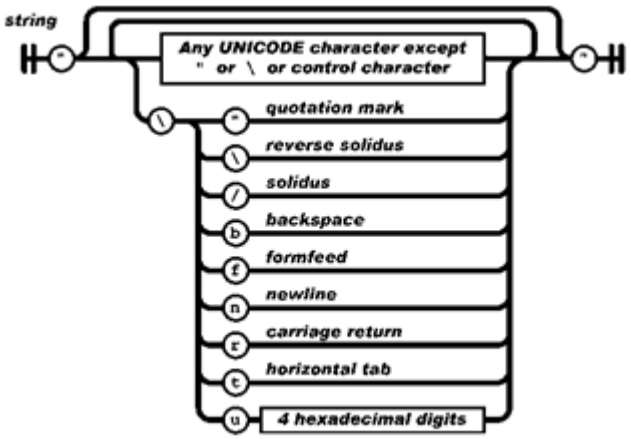
Gambar 2.4 Array JSON[7]

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.



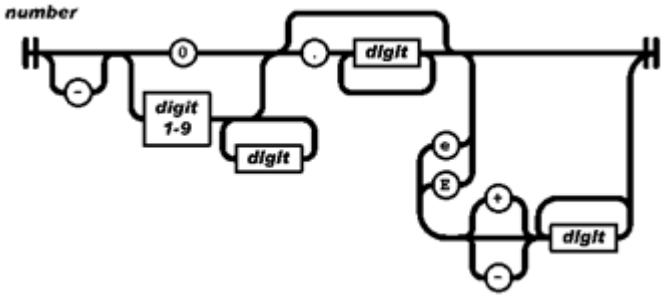
Gambar 2.5 Value JSON[7]

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 2.6 String JSON[7]

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



Gambar 2.7 Number JSON[7]

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

2.4 *Application Programming Interface* (API)

Application Programming Interface (API) adalah sekumpulan perintah, fungsi, komponen, dan protokol yang disediakan oleh sistem operasi ataupun bahasa pemograman tertentu yang dapat digunakan oleh programmer saat membangun perangkat lunak.

Dalam suatu pemograman dibutuhkan setidaknya ribuan *system call* per detik oleh karena itu banyak *programmer* yang menggunakan API. Didalam API terdapat fungsi-fungsi/perintah untuk menggantikan bahasa yang digunakan dalam *system calls* dengan bahasa yang lebih mudah dimengerti oleh *programmer*. Fungsi yang dibuat dengan menggunakan API tersebut kemudian akan memanggil *system call* sesuai dengan sistem operasinya. Tidak tertutup kemungkinan nama dari *system call* sama dengan nama API.

Kelebihan pemograman menggunakan API adalah [8]:

1. Portabilitas
   Programmer yang menggunakan API dapat menjalankan programnya dengan sistem operasi mana saja asalkan sudah terinstal API tersebut.
2. Mudah Dimengerti
   API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call*.

2.5 *Global Positioning System* (GPS)

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 [9].

2.6 *Location Based Services* (LBS)

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 garis lintang dan garis bujur untuk menentukan letak geografis, namun sebagai user kita membutuhkan alamat atau posisi *real time* kita bukan nilai garis lintang dan garis bujur. Android menyediakan *geocoder* yang mendukung *forward* dan *reverse geocoding*. Menggunakan *geocoder*, anda dapat mengkorversi nilai garis lintang dan garis bujur menjadi alamat dunia nyata atau sebaliknya [3].

2.7 GPS Photo Tagging (GeoTagging)

GPS Photo Tagging juga dikenal sebagai geotagging, 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 PhotoTagging 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 [17].

2.8 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 antara lain [17]:

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.

2.9 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].

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.

No.	Tagging	Keterangan
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.

2.10 *Geocoding*

Geocoding adalah proses konversi alamat (seperti "1600 Amphitheatre Parkway, Mountain View, CA") menjadi koordinat geografis (seperti garis lintang 37,423021 dan garis bujur -122,083739), yang bisa Anda gunakan untuk menempatkan marker pada peta, atau memosisikan peta.

Geocoding terbalik adalah proses mengonversi koordinat geografis menjadi alamat yang bisa dibaca orang. Layanan geocoding terbalik pada Google Maps Geocoding API juga memungkinkan Anda menemukan alamat untuk ID tempat yang diberikan.[16]

2.11 *Object Oriented Analysis and 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 memeriksa 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 [10].

2.11.1 *Unfied Modeling Language* (UML)

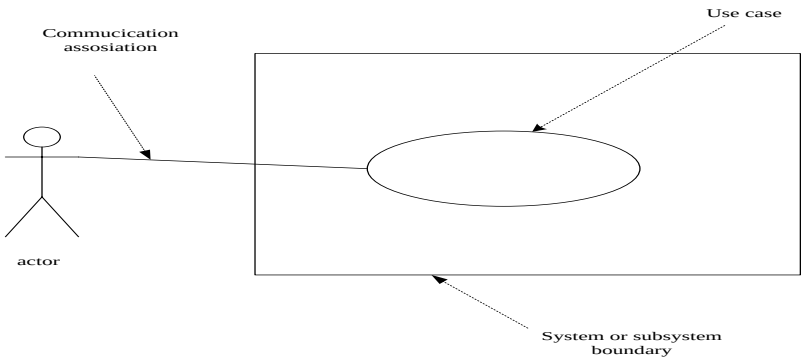
Unified Modeling Language (UML) adalah termasuk ke dalam rumpun jenis pemodelan notasi grafis yang didukung oleh meta-model tunggal. Pemodelan ini berguna untuk membantu dalam menjelaskan dan merancang perangkat lunak yang dibangun dengan object-oriented (OO). UML merupakan standar terbuka yang dikelola oleh Open Management Group (OMG) yang berada dibawah naungan perusahaan-perusahaan konsorsium terbuka[11]. UML merupakan bahasa pemodelan yang terdiri banyak model diantaranya adalah :

1. Use case Diagram

Menurut Booch (2005), suatu *use case diagram* menampilkan sekumpulan *use case* dan aktor (pelaku) dan hubungan diantara *use case* dan aktor tersebut. *Use case diagram* digunakan untuk penggambaran *use case* statik dari suatu sistem. *Use case diagram* penting dalam mengatur dan memodelkan kelakuan dari suatu sistem.

Use case menjelaskan apa yang dilakukan sistem (atau subsistem) tetapi tidak tidak menspesifikasikan cara kerjanya. *Flow of event* digunakan untuk menspesifikasikan kelakuan dari *use case*. *Flow of event* menjelaskan *use case* dalam bentuk tulisan dengan sejelas-jelasnya, diantaranya bagaimana, kapan *use*

case dimulai dan berakhir, ketika *use case* berinteraksi dengan aktor, obyek apa yang digunakan, alur dasar dan alur alternatif.[11]

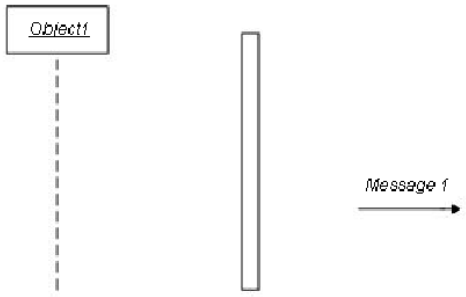


Gambar 2.8 Notasi *Use Case Diagram* [11]

2. Sequence Diagram

Menurut Booch (2005), suatu *sequence diagram* adalah suatu diagram interaksi yang menekankan pada pengaturan waktu dari pesan-pesan. Diagram ini menampilkan sekumpulan peran dan pesan-pesan yang dikirim dan diterima oleh instansi yang memegang peranan tersebut.

Sequence diagram menangkap objek dan *class* yang terlibat dalam skenario dan urut-urutan pesan yang ditukar antara objek diperlukan untuk melaksanakan fungsionalitas skenario. *Sequence diagram* berasosiasi dengan *use case* selama proses pengembangan. Dalam *Unified Model Language* (UML), objek dalam *sequence diagram* digambar dengan segiempat yang berisi nama objek yang diberi garis bawah. Objek dapat diberi nama dengan tiga cara : (nama objek), (nama objek dan *class*) atau (hanya nama *class* (*anonymous* object)). Berikut notasi *sequence diagram* seperti terlihat pada gambar dibawah ini [11]:



Gambar 2.9 Notasi *Sequence Diagram* [11]

3. Class Diagram

Menurut Booch (2005), *class diagram* menunjukan sekumpulan kelas, antarmuka, dan kerjasama serta hubungannya. *Class diagram* digunakan untuk memodelkan perancangan statik dari gambaran sistem. Biasanya meliputi permodelan *vocabulary* dari sistem, permodelan kerjasama, atau permodelan skema.

Class diagram dapat digunakan untuk membangun sistem yang dapat dieksekusi melalui teknik *forward and reverse*, selain untuk penggambaran, penspesifikasian, dan pendokumentasian struktur model.

Class Diagram terdiri dari:

a) Nama *Class*.

b) Atribut.

c) Operasi/Method.

Tabel 2.2 *Class Diagram*

Nama Class
Atribut
Operasi/method

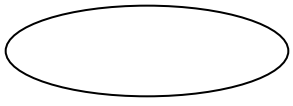
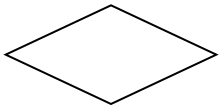
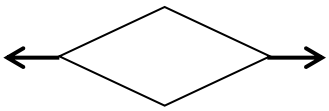
Atribut dan Operasi/*method* dapat memiliki tiga sifat berikut:

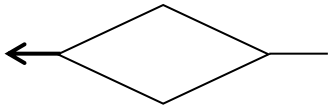
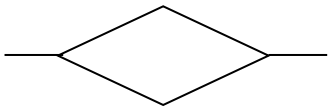
- *Public*, dapat dipanggil oleh *class* apa saja.
- *Protected*, hanya dapat dipanggil atau diakses oleh *class* yang bersangkutan dan *class* turunannya.
- *Private*, hanya dapat dipanggil oleh dirinya sendiri (tidak dapat diakses dari luar *class* yang bersangkutan).

2.12 Entity Relationship Diagram (ERD)

Entity Relationship Diagram adalah sekumpulan cara atau peralatan untuk mendeskripsikan data-data atau objek-objek yang dibuat berdasarkan dan berasal dari dunia nyata yang disebut entitas (entity) serta hubungan (relationship) antar entitas-entitas tersebut dengan menggunakan beberapa notasi [12]. Komponen-komponen pembentuk ERD dapat di lihat pada Tabel 2.3 Komponen-komponen ERD [12].

Tabel 2.3 Komponen-komponen ERD [12]

Notasi	Komponen	Keterangan
	Entitas	Individu yang mewakili suatu objek dan dapat dibedakan dengan objek yang lain
	Atribut	Properti yang dimiliki oleh suatu entitas, dimana dapat mendeskripsikan karakteristik darri entitas tersebut
	Relasi	Menunjukkan hubungan diantara sejumlah entitas yang berbeda
	Relasi 1 : 1	Relasi yang menunjukkan bahwa setiap entitas pada himpunan entitas pertama berhubungan dengan paling banyak satu entitas pada himpunan entitas kedua

Notasi	Komponen	Keterangan
	Relasi 1 : N	Relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak atau sebaliknya. Setiap entitas dapat berelasi dengan banyak entitas pada himpunan entitas yang lain
	Relasi N : N	Hubungan ini menunjukkan bahwa setiap entitas yang pertama dapat berhubungan dengan banyak entitas pada himpunan entitas yang kedua, demikian juga sebaliknya

2.13 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) [13].

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.

2.13.1 Kelebihan Java

Adapun beberapa kelebihan dari bahasa pemrograman Java adalah sebagai berikut [13] :

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. Terintrepentasi 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.

2.14 *Database Management System* (DBMS)

DBMS (*Database Management System*) adalah suatu system aplikasi yang digunakan untuk menyimpan, mengelola dan menampilkan data. suatu system aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut:

1. Menyediakan fasilitas untuk mengelola akses data
2. Mampu menangani integritas data
3. Mampu menangani akses data secara bersamaan
4. Mampu menangani backup data

DBMS sudah mulai berkembang sejak tahun 1960an. Kemudian sekitar tahun 1970an mulai berkembang teknologi Relational DBMS yaitu DBMS berbasis relasional model. Relasional model pertama kali dikembangkan oleh Edgar J. Codd pada tahun 1970. Secara sederhana relasional dapat dipahami sebagai suatu model yang memandang data sebagai sekumpulan tabel yang saling terkait. Hampir semua DBMS komersial dan open source saat ini berbasis Relastional DBMS. Pada tahun 1980an mulai berkembang Object Oriented DBMS (OODBMS). OODBMS berkembang seiring dengan perkembangan teknologi pemrograman berorientasi objek. Saat ini OODBMS juga cukup berkembang namun belum menggeser kepopuleran RDBMS.MySQL adalah perangkat lunak DBMS atau server basis data yang dapat melayani banyak pengguna, dan dapat melakukan proses dalam waktu yang bersamaan (multi threaded). MySQL juga sangat cepat dan fleksibel dan dapat menyimpan data yang berkapasitas besar. Dalam perancangannya, MySQL mengutamakan kecepatan proses, ketangguhan dan kemudahan bagi penggunanya.

Pada saat ini MySQL adalah server basis data yang paling banyak diminati oleh banyak perusahaan di dunia. MySQL menyediakan lingkungan yang mudah dan efisien untuk menyimpan data dan mengakses basis data berkapasitas besar lebih cepat dari pada server basis data lainnya. Keuntungan lainnya adalah bahwa MySQL server tidak membutuhkan spesifikasi hardware yang mahal. MySQL server memungkinkan membuat aplikasi client untuk mengakses dan memanipulasi data di server basis data melalui basis data provider. Keuntungan utama menggunakan MySQL server adalah mudah digunakan, berkecepatan tinggi dan membutuhkan sedikit memori serta tidak membutuhkan spesifikasi hardware yang mahal [12].

2.15 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.[14]

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:

Rumus Haversine

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

Keterangan:

x = Longitude (Lintang)

y= Lattitude (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 (longitude/latitude)^{o} → Radian

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

=12.96009233 km

2.16 Pengujian Black-Box

Menurut Pressman (2010, p495) *Black-Box testing* berfokus pada persyaratan fungsional perangkat lunak yang memungkinkan *engineers* untuk memperoleh set kondisi *input* yang sepenuhnya akan melaksanakan persyaratan fungsional untuk sebuah program. *Black-Box testing* berusaha untuk menemukan kesalahan dalam kategori berikut:

1. Fungsi yang tidak benar atau fungsi yang hilang
2. Kesalahan antarmuka
3. Kesalahan dalam struktur data atau akses *database* eksternal
4. Kesalahan perilaku (*behavior*) atau kesalahan kinerja
5. Inisialisasi dan pemutusan kesalahan

Macam-macam pengujian metode Black Box :

a. Equivalence Partitioning
b. Boundary Value Analysis
c. Cause-Effect Graphing Techniques
d. Comparison Testing

2.17 Skala Likert

Skala likert menurut Sugiyono (2010:93) adalah sebagai berikut : "Skala Likert digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang fenomena sosial." Untuk setiap pilihan jawaban diberi skor, maka responden harus menggambarkan, mendukung pernyataan. Untuk digunakan jawaban yang dipilih. Dengan skala Likert, maka variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak ukur menyusun item-item instrumen yang dapat berupa pertanyaan atau pernyataan.

Tabel 2.4 Skor Skala Likert

Skala Jawaban	Keterangan	Skor
SS	Sangat Setuju	5
S	Setuju	4
CS	Cukup Setuju	3
KS	Kurang Setuju	2
TS	Tidak Setuju	1

Data dari kuesioner akan diolah untuk mencari nilai rata-rata dari masing-masing jawaban dengan menggunakan rumus skala likert[18].

$$p = \frac{f}{\mathrm{n}}$$

Keterangan:

p = nilai rata-rata yang dicari

f = bobot jawaban * responden pemilih

n = jumlah responden