BAB 2
LANDASAN TEORI

2.1 Kalori

Kalori merupakan satuan unit untuk menghitung kuantitas energi. Setiap makanan yang dikonsumsi mengandung sejumlah kalori yang diperlukan oleh tubuh untuk melakukan aktivitas. Kalori dapat dianalogikan sebagai bahan bakar dari sebuah mesin untuk bergerak. Kalori yang terkandung dalam makanan disediakan oleh protein, karbohidrat dan juga lemak. Diantara ketiga komponen penghasul energi tersebut, lemak mengandung kalori yang terbesar [8].

Jumlah kalori dalam makanan diperlukan untuk memperhitungkan keseimbangan energi. Apabila jumlah kalori yang dikonsumsi lebih sedikit dari kalori yang dikeluarkan, maka berat badan akan berkurang dikarenakan cadangan energi dari lemak akan digunakan. Sebaliknya, apabila jumlah kalori yang dikonsumsi lebih besar dari kalori yang digunakan untuk beraktivitas, maka kelebihan kalori tersebut akan disimpan sebagai lemak. Akibatnya berat badan akan meningkat. Penumpukan lemak yang berlebih dapat meningkatkan resiko terjadinya obesitas, hipertensi, stroke, penyakit jantung, dan diabetes. Karena itu, asupan kalori perlu dikontrol untuk menjaga berat badan dan mencegah terjadinya penyakit metabolik.

Berikut Tabel 2.1 Jumlah Perkiraan Kalori yang Digunakan Saat Beraktivitas.

Tabel 2.1 Jumlah Perkiraan Kalori yang Digunakan Saat Beraktivitas

Aktivitas	Durasi	Energi yang dibutuhkan
Tidur	1 Jam	65 kkal
Duduk	1 Jam	75 kkal
Berdiri	1 Jam	90 kkal
Berjalan (4 km/jam)	1 Jam	210 kkal
Jogging (7 km/jam)	1 Jam	400 kkal
Berlari (10 km/jam)	1 Jam	660 kkal

Berenang	1 Jam	580 kkal

2.2 Kebutuhan Energi

Energi dalam tubuh manusia dapat timbul dikarenakan adanya pembakaran karbohidrat, protein dan lemak, dengan demikian agar manusia selalu tercukupi energinya diperlukan zat-zat makanan yang cukup pula ke dalam tubuhnya. Manusia yang kurang makan akan lebih baik kekuatannya, fisiknya maupun daya ingatnya serta daya pemikirannya karena kurangnya zat-zat makanan yang diterima tubuhnya yang dapat menghasilkan energi [9].

Menurut Kartasapoetra dan Marsetyo [9], dalam pengertian makanan sebagai sumber energi ternyata energi makanan dalam proses-proses yang terjadi dalam tubuh hanya sebagian saja yang dapat diubah menjadi panas. Dalam keadaan hanya sedikit melakukan kerja fisik, sebagian besar energi diubah menjadi panas, dan dalam keadaan tidak melakukan pekerjaan fisik maka relatif seluruh energi diubah menjadi panas dan selanjutnya panas akan keluar dari tubuh.

Konsumsi energi berasal dari makanan yang diperlukan untuk menutupi pengeluaran energi sseorang bila ia mempunyai ukuran dan komposisi tubuh dengan tingkat aktivitas yang sesuai dengan kesehatan jangka panjang dan memungkinkan pemeliharaan aktivitas fisik yang dibutuhkan secara sosial dan ekonomi. Kebutuhan energi total orang dewasa diperlukan untuk :

a. Metabolisme Basal
b. Aktivitas Fisik
c. Efek makanan atau pengaruh dinamik khusus (*Specific Dynamic Action)*

Rumus Harris-Bennedict (1990) memperhitungkan berat badan, tinggi badan dan umur. Indeks paling berpengaruh kepada AMB adalah berat badan menurut umur dan rumus linier. Angka metabolisme basal berdasarkan usia dan jenis kelamin dapat dilihat pada tabel 2.2 Angka Metabolisme Basal Berdasarkan Usia dan Jenis Kelamin.

Tabel 2.2 Angka Metabolisme Basal Berdasarkan Usia dan Jenis Kelamin

Kelompok Umur (Tahun)	AMB (Kkal/hari)	
	Laki-Laki	Perempuan
0-3	60,9 B – 54	61,0 B – 51
3-10	22,7 B + 495	22,5 B + 499
10-15	17,5 B + 651	12,1 B + 746
18-30	15,3 B + 679	14,7 B + 496
30 – 60	11,6 B + 879	8,7 B + 829
>60	13,5 B + 487	10,5 b + 596

Menurut Cakrawati dan Mustika [8], aktivitas fisik memerlukan energi di luar kebutuhan untuk metabolisme basal. Pengertian aktivitas fisik adalah gerakan yang dilakukan otot tubuh dan sistem penunjangnya. Selama aktivitas fisik, otot memerlukan energi di luar metabolisme untuk bergerak. Jantung dan paru-paru memerlukan tambahan energi untuk mengantarkan zat-zat gizi dan oksigen ke seluruh tubuh dan mengeluarkan sisa-sisa dari tubuh. Kebutuhan energi untuk aktivitas fisik tergantung dari banyaknya otot yang bergerak, waktu dan beban pekerjaan yang dilakukan sehingga seseorang yang gemuk memerlukan energi lebih besar daripada seseorang yang kurus.

Kebutuhan energi untuk pengaruh termis makanan atau kegiatan dinamik khusus adalah energi tambahan yang diperlukan tubuh untuk pencernaan makanan, absorpsi dan metabolisme zat-zat gizi yang menghasilkan energi. SDA bergantung pada jumlah energi yang dikonsumsi yaitu ± 10% kebutuhan energi untuk metabolisme basal dan untuk aktivitas fisik .

Menurut Kartasapoetra dan Marsetyo [9], faktor-faktor yang mempengaruhi energi metabolisme dasar sebagai berikut :

a. Faktor jaringan aktif di dalam tubuh

Adanya kontraksi otot dan kelenjar yang aktif merupakan alat-alat gerak aktif yang menandakan adanya jaringan aktif. Mekanisme pergerakan tulang sendiri merupakan gerakan aktif yang memerlukan tonus dan kontraksi otot. Otot dan

kelenjar sebagai jaringan aktif tentunya akan lebih banyak memerlukan energi agar masing-masing dapat berfungsi dengan baik dibandingkan dengan tulang dan lemak yang merupakan jaringan tidak aktif.

b. Besar dan luas bidang permukaan tubuh

Seseorang yang bertubuh besar, bidang permukaan tubuhnya akan lebih luas daripada seseorang yang bertubuh lebih kecil. Tubuh yang besar dengan bidang permukaan luas juga akan mempunyai jaringan aktif yang lebih banyak dengan demikian energi metabolisme dasar orang yang bertubuh besar akan lebih besar daripada orang yang bertubuh lebih kecil dalam melakukan gerakan-gerakan fisik yang sama

c. Komposisi tubuh

Dua orang yang sama berat tubuhnya akan tetapi yang seorang bertubuh gemuk (banyak lemak) tampak tubuhnya tidak padat dan tidak kekar dan seorang lagi bertubuh olahragawan, padat, dan kekar menandakan banyak kegiatan / gerakan fisik yang dilakukannya dibandingkan yang bertubuh gemuk, maka energi minimal yang diperlukan oleh orang yang banyak melakukan gerakan/kegiatan fisiknya akan lebih besar (dibandingkan dengan orang yang gemuk yang kurang melakukan gerakan/kegiatan fisiknya).

d. Jenis kelamin

Seorang laki-laki dan seorang wanita dengan berat badan yang sama, biasanya dalam kesamaan berat ini, wanita lebih banyak mengandung lemak di dalam tubuhnya, yang berarti pula bahwa jaringan tidak aktif dalam tubuh wanita lebih banyak. Dengan demikian, energi metabolisme dasar pada tubuh wanita lebih rendah daripada energi metabolisme dasar pada tubuh laki-laki. Biasanya energi minimal yang diperlukan wanita sepuluh persen lebih rendah daripada yang diperlukan laki-laki.

e. Usia

Seorang pemuda mampu melakukan pekerjaan-pekerjaan berat, bergerak lincah. Giat berkegiatan, kesemuanya itu karena didorong oleh intensitas kerja organ-organ di dalam tubuhnya yang masih besar dan cepat. Lain halnya dengan orang yang telah berusia setengah abad ke atas, yang dikarenakan kehebatan kerja organ-organ dalam tubunya telah menurun maka pekerjaan berat biasanya tidak sanggup lagi dikerjakannya, gerakan-gerakan dan kegiatan-kegiatannya telah banyak menurun. Keadaan demikian juga berlaku untuk pemudi dan ibunya. Denyut jantung, pengembangan paru-paru, berlangsungnya proses oksidasi di dalam jaringan tubuh pemuda/pemudi masih berlangsung cepat jika dibandingkan dengan berfungsinya organ-organ tubuh tersebut pada orang tua (bapak/ibu).

Menurunnya intensitas kerja organ-organ dalam tubuh orang tua dikarenakan mengendornya tonus otot (jaringan aktif). Nilai energi dasar pada tubuh seseorang memang pada permulaannya akan selalu meningkat. Ketika masih bayi akan berlangsung peningkatan dan pada usia 1 sampai 2 tahun mencapai titik optimum, setelah itu mulai terjadi penurunan. Namun demikian nilai energi dasar tersebut sampai pada kurun waktu akil balig (periode puber) masih dapat dikatakan cukup tinggi dan selanjutnya penurunan-penurunan akan makin tampak dalam perjalanannya menuju hari tua. Sejak umur dewasa dengan bertambahnya umur 1 tahun, pada laki-laki akan terjadi penurunan energi minimal sekitar 7 sampai 15 kalori, dan demikian seterusnya, sedangkan pada perempuan dengan bertambahnya umur 1 tahun terjadi penurunan sekitar 2 sampai 3 kalori (Harris, Benedict).

f. Sekresi hormon

Di dalam tubuh terdapat kelenjar-kelenjar hormon, seperti kelenjar hipofise, epifise, tiroid (gondok), paratiroid, adrenalin (ginjal), lambung, usus, pancreas, kelenjar kelamin, dan sebagainya. Hormon merupakan zat kimiawi yang dihasilkan oleh kelenjar endokrin yang mengatur homeostatis, reproduksi, metabolisme, dan tingkah laku. Hormon tiroksin (*thyroxin*) yang dihasilkan oleh kelenjar tiroid (*thyroid*) yang fungsinya mengatur metabolisme karbohidrat,

mempengaruhi pertumbuhan, perkembangan, dan differensiasi jaringan tubuh, sekresi hormon ini yang berlebihan ditandai dengan meningkatnya metabolisme tubuh, denyut jantung, emosional, dan lain-lain tentunya mengakibatkan nilai energi dasar metabolisme meningkat. Peningkatan ini dapat berlangsung sampai 75%. Sebaliknya apabila sekresi hormon ini terlalu sedikit maka nilai energi dasar metabolisme menurun. Penurunan ini dapat berlangsung sampai 30%.

Selanjutnya perhatikan pula hormon adrenalin yang dihasilkan bagian medula kelenjar adrenalin (ginjal), dalam hal sekresinya yang berlebihan dapat menyebabkan peningkatan pemacuan aktivitas jantung, pengerutan otot polos pada arteri, peningkatan tekanan darah, pernafasan, pengubahan glikogen menjadi glukosa, yang tentunya sangat berpengaruh pada peningkatan pemakaian energi minimal.

g. *Tonus* pada waktu tidur

Keadaan *tonus* pada waktu seseorang dewasa tidur dan berbaring terdapat perbedaan, di mana waktu tidur keadaannya lebih rendah. Hal ini disebabkan atau dikaitkan dengan kerja-kerja internal dalam tubuh orang yang bersangkutan, dimana dalam keadaan tidur kerja-kerja organ internal dalam tubuh akan berlangsung lebih lambat dibandingkan dengan dalam keadaan berbaring. Berdasarkan penelitian para pakar, pada waktu orang dewasa tidur energi minim/metabolisme dasar yang diperlukan berada 10% lebih rendah dibandingkan dengan dalam keadaan orang itu berbaring.

h. *Tonus* otot

Otot akan bekerja terus secara teratur selama manusia itu masih hidup dan untuk gerakannya itu selalu diperlukan energi. Proses gerak otot berlangsung sebagai berikut :

1. Pertama – tama urat syaraf menyampaikan rangsang
2. Rangsang diterima oleh *asetilkolin* yang menyebabkan protein dalam otot (*aktin-miosin*) mengerut.

3. Pada proses pengerutan tersebut diperlukan energi yang diambil dari penguraian senyawa : Adenosin Trifosfat menjadi Adenosin Difosfat dan kemudian menjadi Adenosin Monofosfat yang terjadi secara anaerob.
4. Pembentukan *Adenosin Trifosfat* (ATP) dari *Adenosin Difosfat* (ADP) dan *Adenosin Difosfat* (ADP) menjadi *Adenosin Monofosfat* (AMP) diperlukan asam fosfor dan energi.
5. Energi tersebut diambil dari penguraian glikogen-glikogen (gula otot) yang dilarutkan terlebih dahulu menjadi laktasidogen (pembentukan asam susu/asam laktat)

Jumlah energi yang diperlukan tergantung dari tinggi rendahnya *tonus*, yang dalam hal ini tentunya jelas akan lebih banyak dibandingkan dengan yang diperlukan untuk mengerakkan otot jantung, otot-otot pernafasan, dan alat-alat tubuh lainnya, mengingat jumlah otot jauh lebih banyak daripada jaringan alatalat tubuh tadi.

i. Kondisi emosi dan mental

Keperluan terhadap energi minimal atau energi metabolisme dasar akan terpengaruh pula oleh kondisi emosi dan mental manusia. Pada waktu manusia berada dalam keadaan emosi akan berlangsung sekresi adrenalin sehingga terjadi pemacuan aktivitas jantung, peningkatan tekanan darah, dan lain-lain.

j. Gerakan tubuh yang berat

Pada waktu orang tersebut melakukan gerak fisik yang lebih berat maka proses oksidasi berlangsung lebih aktif, yang tentunya memerlukan tambahan/peningkatan sejumlah energi metabolisme dasar (energi minimal). Keadaan sebaliknya (penurunan keperluan energi metabolisme dasar) akan terjadi pada waktu orang tersebut bersemedi, mengurangi gerak fisiknya selama beberapa dari (dalam hal ini akan berlangsung penyesuaian gerakan dalam tubuh dengan keterbatasan energi yang dihasilkan sehubungan dengan pengurangan pemasukan makanan ke dalam tubuhnya).

k. Kehamilan

Energi metabolisme dasar yang dibutuhkan seorang ibu yang sedang hamil akan menjadi lebih tinggi daripada apa yang diperlukannya ketika tidak hamil. Menjadikannya keperluan ini lebih tinggi adalah sejalan dengan kenaikan berat tubuhnya, rata-rata biasanya sekitar 4%.

l. Kondisi tubuh yang tidak sehat

Kondisi tubuh yang tidak sehat menjadikan atau diikuti dengan kenaikan suhu di dalam tubuh banyak berpengaruh pula terhadap keperluan energi dasar/energi minimal di dalam tubuh. Menurut penelitian para pakar, setiap terjadi kenaikan suhu tubuh 1^0 C diperlukan peningkatan energi dasar sekitar 13%.

2.3 Android

Android adalah sebuah sistem operasi untuk telepon seluler yang berbasis linux. Android menyediakan platfotm yang terbuka bagi para *developer* untuk menciptakan aplikasi ciptaan mereka sendiri. Pada mulanya,, Google.Inc. membeli Android Inc. Pendatang baru yang membuat piranti lunak untuk handphone. Kemudian untuk mengembangkan Android, dibentuklah Handset Alliance, konsursium dari 34 perusahaan piranti keras, piranti lukan dan telekomunikasi [10].

Pada saat pertama kali Android dirilis pada 5 November 2007, Android bersama dengan Open Handset Alliance menyatakan untuk mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android dibawah lisensi Apache, sebuah lisensi perangkat software dan standar terbuka perangkat seluler. Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung dari Google atau dikenal sebagai Open Handset Distribution (OHD).

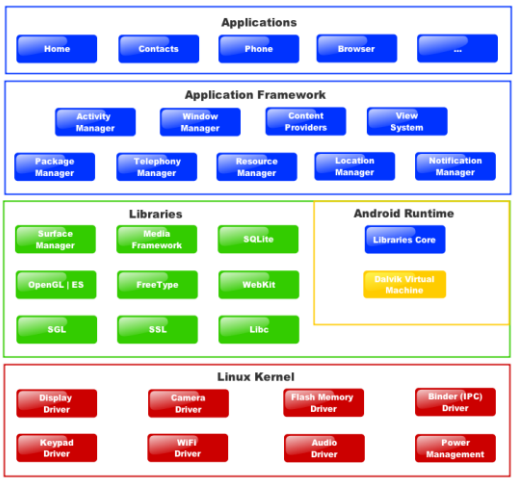
Android menawarkan sebuah lingkungan yang berbeda untuk pengembang. Setiap aplikasi memiliki tingkatan yang sama. Android tidak membedakan antara

aplikasi inti dengan aplikasi pihak ketiga. *Application Programming Interface* (API) yang disediakan menawarkan akses ke hardware, maupun data-data ponsel sekalipun, atau data sistem sendiri. Bahkan pengguna dapat menghapus aplikasi inti dan menggantikannya dengan aplikasi pihak ketiga.

Android merupakan sistem operasi yang berkembang dengan pesat, namun tidak menjadikannya sistem operasi yang sempurna ada beberapa kekurangan dari sistem operasi Android diantaranya Android terkesan rumit, karena mempunyai banyak sekali *widget* maupun aplikasi dengan banyak pengaturan sehingga pengguna harus banyak belajar mengenai Android, selain itu Android yang merupakan sistem operasi terbuka sehingga pengguna dapat memasang aplikasi di luar toko aplikasi yang ditawarkan oleh perangkat Android tersebut sehingga sangat rentan terkena ancaman *malware* atau *virus*. Tidak semua perangkat Android dapat langsung memperbarui sistem operasi terbaru, karena produsen smartphone lebih mementingkan produk baru untuk diberi sistem operasi yang terbaru, dibanding dengan memberi pemberitahuan tentang update sistem operasi terbaru sehingga membutuhkan waktu lama untuk memperbarui sistem operasi bagi beberapa perangkat.

2.3.1 Arsitektur Android

Arsitektur Android dapat digambarkan seperti pada gambar 2.1 Arsitektur Android. Secara garis besar Arsitektur Android dapat dijelaskan sebagai berikut :



Sumber gambar : http://developer.android.com

Gambar 2.1 Arsitektur Android

1. *Apllication and Widget*

Application and 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.

2. *Application Frameworks*

Applications Frameworks 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 notifikasi, 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* adalah sebagai berikut :

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

3. *Libraries*

Libraries ini adalah *layer* dimana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya. Berjalan di atas Kernel, *layer* ini meliputi berbagai *library* C/C++ inti seperti Libc SSL, 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 web view*.
7) Libraries 3D yang mencakup implementasi OpenGL ES1.0 API's.

4. *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* 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 fungsifungsi secara efisien, dimana merupakan pengembangan yang mampu membuat Linux Kernel untuk melakukan *threading* dan manajemen tingkat rendah.

5. *Linux Kernel*

Linux Kernel adalah layer dimana inti dari sistem operasi Android itu berada. Berisi *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. Keunikan dari nama sistem operasi (OS) Android adalah dengan menggunakan nama makanan hidangan penutup (*Dessert*). Selain itu juga nama-nama OS Android memiliki huruf awal berurutan sesuai abjad. Adapun beberapa nama dan versi Android yang sudah diluncurkan adalah sebagai berikut :

1) Android v 1.0 *Astro* (*Alpha*)

Sebenarnya sebelum mereka memberikan nama-nama kudapan sebagai nama untuk versi OS nya, Android sempat memiliki 2 versi awal dengan nama Android *Alpha* dan *Beta*. Nama untuk versi pertama ini sendiri sebenarnya adalah Android Astro, namun karena alasan hak cipta (*trademark*), nama ini tidak jadi digunakan. Di versi awal ini belum ada perangkat dengan sistem operasi Android yang dijual secara komersil.

2) Android v 1.1 *Bender* (*Beta*)

Versi ini dirilis pada tanggal 5 November 2007 yang merupakan versi lanjutan dari Android Astro (Alpha). Sama seperti versi awalnya, nama Bender juga juga tak jadi digunakan karena alasan hak cipta (*trademark*). Kemudian lahirlah telepon seluler pertama dengan sistem operasi Android yang dijual secara komersil yakni HTC Dream

3) Android v 1.5 *Cupcake*

Ini merupakan versi pertama yang menggunakan nama makanan manis sebagai kode nama untuk tiap versi Android yang kemudian tradisi untuk

menamai versi Android dengan nama makanan manis masih diteruskan hingga saat ini. Android *Cupcake* dirilis pada tanggal 30 April 2009.

4) Android v 1.6 *Donut*

Dirilis tidak sampai setahun setelah perilisan Android *Cupcake*, yakni pada tanggal 15 September 2009. Versi ini dihadirkan untuk menutupi *bug* pada versi sebelumnya, sekaligus untuk penambahan beberapa fitur seperti misalnya dukungan untuk perangkat dengan ukuran layar yang lebih besar.

5) Android v 2.0 – 2.1 *Éclair*

Sistem operasi ini juga dirilis tidak sampai setahun setelah perilisan dua versi sebelumnya yakni pada tanggal 26 Oktober 2009. Mereka masih berfokus untuk menutupi *bug* yang ada dan juga menambahkan beberapa fitur seperti *Bluetooth*, *flash* pada kamera, fitur digital zoom pada kamera, *multi-touch*, *live wallpaper*, dan lainnya. Hadirnya perangkat seri Nexus dari Google yang pertama kali muncul yakni HTC Nexus One juga menggunakan versi OS Android *Eclair*.

6) Android v 2.2 *Frozen Yoghurt* (Froyo)

Dirilis pada tanggal 20 Mei 2010. Perangkat dengan OS Android semakin banyak dan kehadirannya mulai dilirik oleh pasar meski masih jauh dibawah kepopuleran OS lain seperti Symbian dan Windows Mobile.

7) Android v 2.3 *Gingerbread*

Dirilis pada tanggal 6 Desember 2010 bersamaan dengan dihadirkannya Nexus S yang merupakan perangkat smartphone seri Nexus yang diproduksi oleh Samsung. Versi OS ini juga mengawali kesuksesan Android di jagad *smartphone* meski masih kalah populer dengan BlackBerry OS. Beberapa vendor mulai serius untuk menggarap perangkat dengan OS Android.Pada saat itu, Samsung dengan Galaxy series nya berperan besar dalam kesuksesan Android. Promosi yang luar biasa gencarnya membuat orang awam mulai mengenal sistem operasi Android.

Bahkan saat itu sebagian besar orang beranggapan bahwa OS Android adalah milik Samsung karena kuatnya branding yang dilakukan oleh Samsung. Ini juga menjadi awal mula kedigdayaan Samsung di jagad *smartphone*.

8) Android v 3.0 – 3.2 *Honeycomb*

Versi ini dirilis pada tanggal 10 Mei 2011 dan dirancang khusus untuk perangkat tablet, yang kala itu mulai populer di pasaran salah satunya berkat promosi Samsung dan juga kepopuleran Apple iPad.

9) Android v 4.0 Ice Cream Sandwich

Dirilis pada 16 Desember 2011. Bisa dibilang merupakan Android *Honeycomb* yang disempurnakan, dan dioptimalkan untuk penggunaan baik smartphone maupun tablet. Perubahan yang paling terlihat dari versi ini dibanding denganversi sebelumnya adalah dari segi *User interface* yang nampak lebih bersih danelegan. Versi ini juga lebih dioptimalkan untuk urusan multitasking. Bersamaan dengan diperkenalkannya Android ICS, Google juga memperkenalkan perangkat Galaxy Nexus yang merupakan seri smartphone Nexus yang diproduksi oleh Samsung. Setelah versi ini, Google kemudian secara rutin memperkenalkan perangkat seri Nexus pada tiap kali mereka memperkenalkan versi Android terbaru.

10) Android v 4.1 – 4.3 *Jelly Bean*

Dirilis pada 9 Juli 2012. Bersamaan dengan diperkenalkannya versi OS 4.1 pada 27 Juni 2012, Google juga memperkenalkan Nexus 7 yang diproduksi oleh ASUS. Nexus 7 (generasi 1) merupakan seri Nexus pertama yang merupakan perangkat tablet. Jelly Bean mengalami 3x update versi yakni 4.1, 4.2 hingga 4.3. Selanjutnya mereka memperkenalkan Android v4.2 bersamaan dengan dihadirkannya Nexus 4, smartphone yang diproduksi oleh LG plus Nexus 10, perangkat tablet yang diproduksi oleh Samsung.Pada saat versi 4.3 dirilis, Google juga merilis Nexus 7 generasi 2 yang masih diproduksi oleh ASUS yang mana ia memiliki beberapa peningkatan seperti misalnya penambahan kamera belakang serta dukungan untuk konektivitas internet.

11) Android v 4.4 *Kitkat*

Nama *Kitkat* diambil dari sebuah produk cemilan wafer berlapis coklat yang dimiliki oleh Nestle. Sebelumnya Android versi "K" ini disebut-sebut sebagai *Key Lime Pie*, namun atas beberapa pertimbangan akhirnya Google lebih memilih untuk memberi nama *Kitkat*. Ceritanya, *Kitkat* adalah salah satu cemilan yang tersedia di dapur kantor yang biasanya juga menemani para programer Google. Hingga seseorang berkata "Hey, kenapa kita tidak menamainya sebagai Kitkat?".Sesaat setelah ide itu muncul, Google segera menghubungi pihak Nestle sebagai pemilik merk dagang Kitkat dan mereka menyetujui pemberian nama Kitkat untuk versi Android K. Karyawan Google sendiri tidak mengetahui bahwa Android 4.4 akan diberi nama Kitkat karena yang mereka tau versi Android K adalah *Key Lime Pie*. Mereka baru mengetahuinya setelah patung maskot Android Kitkat diletakkan di kantor pusat Google.Versi ini diklaim lebih ramah terhadap perangkat dengan spesifikasi seadanya. Bahkan perangkat dengan RAM 512 MB masih bisa menjalankan OS versi ini dengan mulus. Berbeda dengan *Jelly Bean* yang minimal harus memiliki RAM diatas 756 MB agar dapat berjalan dengan mulus. Bersamaan dengan dirilisnya Android Kitkat pada tanggal 31 Oktober 2013, Google juga merilis Smartphone Nexus 5 yang diproduksi oleh LG.

12) Android v 5.0 *Lollipop*

Dirilis pada tanggal 15 Oktober 2014, versi OS ini mengusung perubahan besar dari segi UI yang nampak lebih *flat* dengan konsep *material design*. Versi Android ini sudah mendukung arsitektur 64-bit sehingga sudah memungkinkan untuk penggunaan RAM diatas 3 GB pada *hardware* perangkat. Penggunaan prosesor 64-bit pun makin banyak diadopsi oleh para *vendor*, mulai dari penerapan pada perangkat flagship hingga perangkat kelas menengah kebawah.

13) Android v 6.0 *Marshmallow*

Versi Android ini resmi dirilis pada bulan September tahun 2015. Bersamaan dengan dirilisnya versi ini, untuk pertama kalinya Google juga memperkenalkan 2 perangkat smartphone Nexus sekaligus yang diproduksi oleh 2 *vendor* yang berbeda. Nexus 5X adalah versi smartphone Nexus kelas menengah dengan ukuran layar 5.2 inch yang diproduksi oleh LG. Sedangkan yang satunya lagi memiliki bentang layar yang lebih lebar yakni 5.7 *inch* yang diberi nama Nexus 6P yang merupakan smartphone flagship hasil kerjasama Google dengan Huawei. Sejak Android 6.0 (Marshmallow), Google memberi perhatian khusus terhadap usaha penghematan konsumsi baterai. Pada Marshmallow, Google memperkenalkan dua fitur baru untuk keperluan ini: *Doze* dan *App Standby*. Bila Android mendeteksi bahwa peranti tidak digunakan dalam waktu lama (layar mati dan peranti tidak bergerak), sistem bisa masuk dalam mode *Doze*. Dalam mode ini, Android akan membatasi akses aplikasi terhadap berbagai perangkat keras seperti prosesor dan jaringan. *App Standby* pada dasarnya serupa, hanya saja di sini pembatasan dilakukan terhadap aplikasi tertentu, bukan terhadap sistem secara keseluruhan. Pada versi ini, Google juga memperkenalkan dukungan terhadap porta USB Type C, dan untuk pertama kalinya memberikan dukungan terhadap pembaca sidik jari.

14) Android v 7.0 *Nougat*

Resmi diperkenalkan pada akhir Juni 2016. Banyak netizen yang berspekulasi bahwa kemungkinan besar, pemberian nama untuk Android versi “N” ini adalah Nutella. Namun Google menepis kabar tersebut setelah resmi memperkenalkannya bersamaan dengan dipamerkannya patung icon Android yang berdiri diatas potongan *Nougat* (yang sepintas lebih mirip dengan tempe itu). Sebelumnya, Google telah mengundang para penggunanya untuk memberikan ide penamaan pada versi ini. Beberapa nama termasuk Nutella dan Nastar pun muncul, hingga akhirnya Google lebih memilih nama Nutella. Google memperkenalkan peningkatan terhadap kompilator aplikasi yang memungkinkan pemasangan yang lebih cepat (sampai 75%) dan ukuran aplikasi yang lebih kompak (sampai 50%), selain eksekusi yang lebih cepat. Pada *Nougat* juga,

Google menyertakan platform realitas virtual (*Daydream*), penghematan data (*Data Saver*), serta dukungan yang lebih baik untuk tampilan multijendela (*Multiwindow*). Yang terakhir ini diperlukan untuk multitugas lebih baik pada peranti dengan layar lebar, seperti phablet dan komputer tablet. Untuk pertama kalinya juga, Google memperkenalkan dukungan terhadap teknologi grafis baru Vulkan, yang diharapkan akan menggantikan OpenGL pada Android.

15) Android v 8.0 Oreo

Android Oreo pertama kali dirilis sebagai preview pengembang pada tanggal 21 Maret 2017. Android Oreo mengusung fitur baru yang membuat smartphone lebih cepat, pintar dan *powerful* dibandingkan dengan versi android yang sebelumnya. Beberapa fitur unggulan dari versi ini adalah *background limit*, *Google Play Protect*, Emoji baru dan *picture in picture*.

16) Android v 9.0 Pie

Android 9.0 (Pie) resmi dirilis pada Agustus 2018. Seperti sebelumnya, Google terus mencari cara untuk menghemat baterai. Fitur baru yang diperkenalkan pada Android 9.0 (Pie) untuk keperluan ini adalah *adaptive battery*, yang menggunakan pembelajaran mesin untuk meramalkan kapan suatu aplikasi tidak digunakan. Android akan "membekukan" aplikasi pada saat-saat tertentu. Pada Android Pie, Google secara resmi mengintegrasikan dukungan untuk fitur kamera ganda, yang sebelumnya sudah diperkenalkan pada beberapa model ponsel Android unggulan. Pembuat ponsel akan lebih mudah memberikan dukungan terhadap fitur tersebut dari sisi perangkat lunak.

2.3.2 Android *Life Cycle*

Aplikasi android terdiri dari beberapa fungsi dasar seperti mengedit catatan, memutar file musik, membunyikan alarm, atau membuka kontak telepon. Fungsi-fungsi tersebut dapat diklasifikasikan ke dalam empat komponen android yang berbeda seperti ditunjukkan pada, klasifikasi tersebut

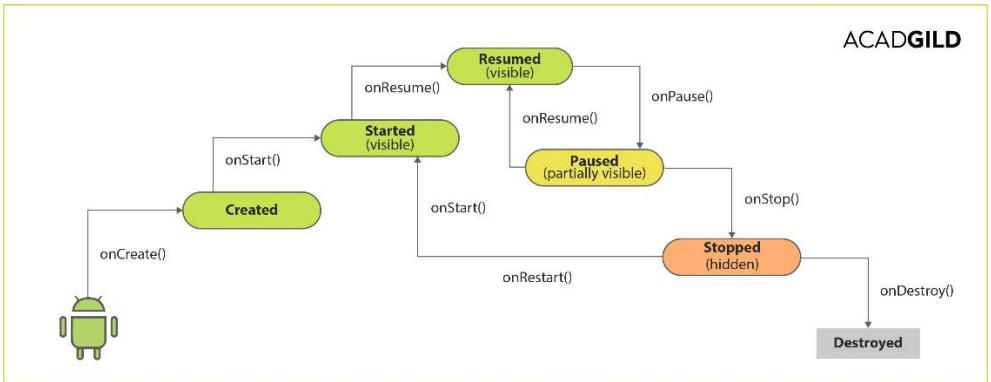
berdasarkan kelas-kelas dasar java yang digunakan. Tabel 2.3 adalah Komponen Aplikasi Mobile.

Tabel 2.3 Komponen Aplikasi Mobile

Functionality	Java Base Class	Examples
Focused thing a user can do	*Activity*	*Edit a note, play a game*
Background	*Service*	*Play music, update weather icon*
Receive messages	*Broadcast Receiver*	*Trigger alarm upon event*
Store and retrieve data	*Content Provider*	*Open a phone contact*

Setiap aplikasi pasti menggunakan minimal satu dari komponen tersebut, akan tetapi terdapat beberapa komponen yang mengharuskan mencantumkan specified permission sebelum digunakan seperti komponen *Service, Broadcast Receiver, Content Provider*.

Android memiliki paradigma pemrograman lain tidak seperti paradigma pemrograman biasa di mana aplikasi yang dijalankan pada fungsi main(), sistem android menjalankan kode dalam method *Activity* dengan menerapkan metode callback tertentu yang sesuai dengan tahap tertentu dari siklus hidup. Setiap aplikasi yang berjalan dalam sistem operasi Android memiliki siklus hidup yang berbeda dengan aplikasi *desktop* ataupun web. Hal ini dikarenakan aplikasi mobile memiliki tingkat interupsi proses yang cukup tinggi seperti ketika *handling* panggilan masuk aplikasi diharuskan menghentikan proses sementara. Penerapan siklus hidup juga berguna untuk memastikan aplikasi tidak menghabiskan sumber daya baterai pengguna.



Sumber gambar : https://medium.com/sketchware/activity-lifecycle-in-android-applications-1b48a7bb584c

Gambar 2.2 Siklus Hidup Android

diilustrasikan pada gambar 2.2 siklus hidup android akan tetapi hanya beberapa dari state tersebut yang menjadi statis diantaranya:

1. *Resumed*

   Resumed terjadi ketika aplikasi berjalan setelah state *paused* . State ini akan menjalankan perintah program yang ditulis pada *method* onResume().

2. *Paused*

   Dalam keadaan ini aktivitas yang terjadi dihentikan secara sementara tetapi masih terlihat oleh pengguna karena terdapat proses yang memiliki prioritas lebih tinggi seperti panggilan telepon. Aplikasi tidak dapat menjalankan perintah apapun ataupun menampilkan apapun dalam state ini .

3. *Stopped*

   Dalam keadaan ini, aplikasi benar-benar tidak ditampilkan dan tidak terlihat oleh pengguna tetapi masih meninggalkan service dibackground. State lain seperti *Created* dan *Started* bersifat sementara dan sistem dengan cepat menjalankan state berikutnya dengan memanggil metode *life cyclecallback* berikutnya. Artinya, setelah sistem OnCreate() dipanggil, dengan cepat sistem akan memanggil method OnStart(), kemudian diikuti oleh onResume()

2.3.3 Fitur Android

Android memiliki beberapa fitur utama yang sering digunakan dalam proses pembangunan aplikasi diantaranya adalah [10] :

1. *Multi-process* dan *App Widgets*

Sistem operasi android tidak melarang prosesor menjalankan lebih dari satu aplikasi dalam satu waktu. Sistem operasi android dapat mengatur aplikasi dan thread yang berjalan secara *multitasking*. Keuntungan yang didapat adalah ketika aplikasi berjalan dan berinteraksi dengan pengguna di layer depan sistem operasi, proses dari aplikasi lain dapat berjalan untuk melakukan pembaruan informasi. Sebagai contoh misalnya ketika pengguna memainkan game, proses lain dapat berjalan di belakang aplikasi seperti memeriksa harga saham dan memunculkan peringatan.

App Widgets adalah mini aplikasi yang dapat *embedded* dalam aplikasi seperti home screen. App widgets dapat menjalankan proses *request* seperti musik *streaming* atau mendeteksi suhu ruangan secara background.

Multi-proses dapat memberikan manfaat berupa *user experience* yang lebih banyak, namun penggunaan fitur tersebut dapat menghabiskan banyak energi baterai jika penggunaan tidak benar.

2. *Touch Gesture* dan *Multi-Touch*

Touchscreen adalah *user interface* intuitif yang digunakan banyak *smartphone* didunia. Dengan fitur ini interaksi dapat dibuat lebih mudah karena cukup dengan menggunakan jari tangan. *Multi-touch* adalah kemampuan yang dapat melakukan tracking lebih dari satu tangan dalam satu waktu. Fitur ini sering digunakan untuk interaksi memperbesar atau memutar objek. Selain itu, pengembang dapat membuat interaksi baru dengan memanfaatkan fitur tersebut.

3. *Hard* dan *Soft Keyboard*

Salah satu fitur pada perangkat smartphone adalah tombol fisik dan non fisik, tombol fisik digunakan untuk navigasi pendukung dalam pengoperasian android. Pengembang aplikasi tidak perlu secara manual untuk mengintegrasikan tombol

tersebut dalam aplikasi. Tombol non fisik adalah tombol yang dibuat oleh sistem operasi seperti keyboard *virtual*, dan tombol navigasi aplikasi.

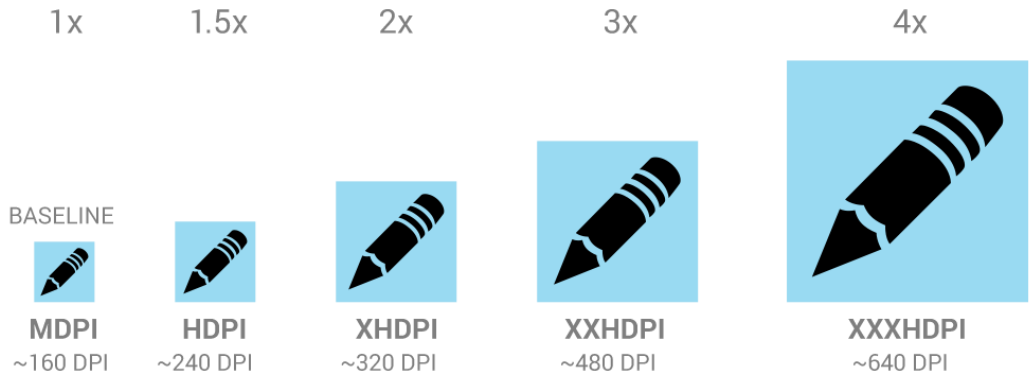
2.3.4 Prinsip Desain

Android memiliki beberapa prinsip desain yang dapat menjadi acuan dalam membuat desain aplikasi android diantaranya adalah [11] :

1. *Multiple Assets*

Android mendukung jutaan smartphone, tablet dan perangkat lain dalam berbagai ukuran layar dan ukuran, untuk itu *Multiple Assets* sangat disarankan digunakan untuk mengatasi fragmentasi pada android. Seperti ilustrasi pada gambar

5 *Mutiple Assets*, android menciptakan beberapa klasifikasi ukuran icon yaitu : MDPI, HDPI, XHDPI, XXHDPI dan XXXHDPI. MDPI dan HDPI dikhususkan untuk icon yang akan digunakan pada device berukuran smartphone sedangkan untuk XHDPI, XXHDPI dan XXXHDPI digunakan pada device berukuran tablet. Gambar 2.3 merupakan *Multiple Access*

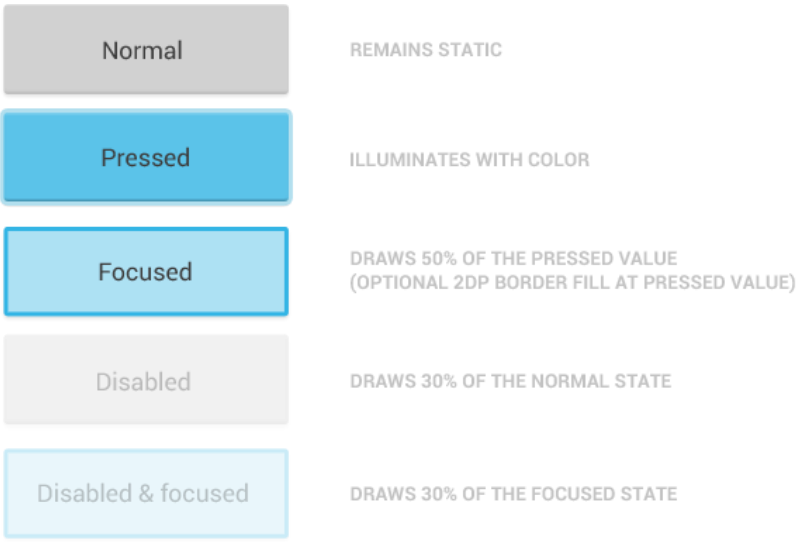


Sumber Gambar : https://stackoverflow.com/questions/25124056/image-sizes-for-multiple-android-screens

Gambar 2.3 Multiple Assets

2. *Touch Feedback*

Touch Feedback dalam android digunakan sebagai respon setiap objek yang ditekan pengguna. Hal ini bertujuan untuk memberi tahu pengguna objek mana yang berinteraksi dengan pengguna.Berikut gambar 2.4 *Touch Feedback.*



Sumber gambar :
https://stuff.mit.edu/afs/sipb/project/android/docs/design/style/touch-feedback.html

Gambar 2.4 Touch Feedback

2.3.5 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 member anda kesempatan untuk membuat aplikasi

yang kita butuhkan yang bukan merupakan aplikasi bawaan Handphone atau Smartphone. Beberapa fitur-fitur android yang paling penting adalah [10] :

1) *Framework* : Aplikasi yang mendukung pengganti komponen dan reusable.
2) *Dalvik Virtual Machine* dioptimalkan untuk perangkat mobile.
3) *Integrated Browser* berdasarkan 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 perangkat emulator, tools untuk debugging, profil dan kinerja memori, dan plugin.

2.4 Java

Inovasi bahasa komputer dimotivasi oleh dua faktor: perbaikan dalam seni pemrograman dan perubahan dalam lingkungan komputasi, tidak terkecuali Java. Dibangun di atas warisan yang kaya dari C dan C++, Java menambahkan perbaikan dan fitur yang mencerminkan keadaan seni dalam pemrograman saat ini. Menanggapi munculnya lingkungan online, Java menawarkan fitur yang merampingkan pemrograman untuk arsitektur yang sangat terdistribusi [12].

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java

umumnya dikompilasi ke dalam p-code (*bytecode*) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM) [12].

Versi awal Java ditahun 1996 sudah merupakan versi rilis sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya :

a) java.lang: Peruntukan kelas elemen-elemen dasar.
b) java.io: Peruntukan kelas input dan output, termasuk penggunaan berkas.
c) java.util: Peruntukan kelas pelengkap seperti kelas struktur data dan kelas kelas penanggalan.
d) java.net: Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
e) java.awt: Kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI).
f) java.applet: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

Seperti telah dibahas sebelumnya, banyak jenis komputer dan sistem operasi yang terhubung ke Internet. Untuk program-program untuk secara dinamis didownload ke semua berbagai jenis platform, beberapa sarana untuk menghasilkan kode dieksekusi portabel diperlukan. Mekanisme yang sama yang membantu menjamin keamanan juga membantu menciptakan portabilitas karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda. Saat ini java merupakan bahasa pemrograman yang populer digunakan dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

Java memliki beberapa kelebihan dibandingkan dengan bahasa pemograman lain,diantaranya :

a) Multiplatform
b) OOP (*Object Oriented Programming)*
c) *Library Class* yang lengkap
d) Mewarisi Kekayaan C/C++
e) Pengumpulan Sampah Otomatis

f) Mudah didekompilasi

2.5 *Web Service*

W3C mendefinisikan *web service* sebagai sebuah *software* aplikasi yang dapat teridentifikasi oleh URI dan memiliki *interface* yang didefinisikan, dideskripsikan, dan dimengerti oleh XML atau JSON dan juga mendukung interaksi langsung dengan *i* aplikasi yang lain dengan menggunakan *message* berbasis XML atau JSON melalui protokol internet. *Web service* adalah sebuah software aplikasi yang tidak terpengaruh oleh platform, menyediakan *method* yang dapat diakses oleh *network*. *Web Service* juga akan menggunakan XML untuk pertukaran data, khususnya pada dua entities bisnis yang berbeda.

Beberapa karakteristik dari *web service* adalah:

1. *Message-Based*
2. *Standards-Based*
3. *Programming Language Independent*
4. *Platform-Neutral*

Beberapa key standard didalam web service adalah: JSON, XML, SOAP, WSDL and UDDI.

2.6 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

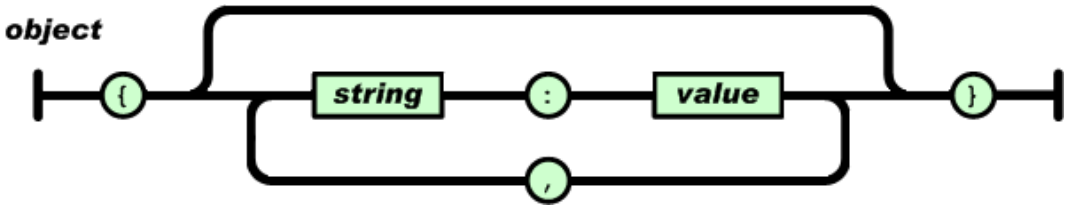
Pemrograman JavaScript, Standar ECMA-262 Edisi ke 3 – Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, JavaScript, Perl, Python dll. Oleh karena sifatsifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur [13] :

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), 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 pemrograman modern mendukung sruktur 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. JSON menggunakan bentuk sebagai berikut:

1. Objek

Objek adalah sepasang nama / nilai yang tidak terurutkan. Seperti yang diilustrasikan pada Gambar 7 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). Objek biasanya digunakan untuk menyimpan data tunggal dalam bentuk JSON. Gambar 2.5 merupakan gambar dari object JSON.



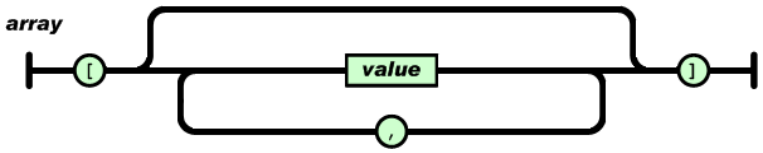
Sumber gambar : https://www.json.org/

Gambar 2.5 Objek JSON

2. Larik

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

, (koma) seperti yang diilustrasikan pada Gambar 8. Larik dalam JSON dapat digunakan sebagai value dari JSON object hal ini dapat berguna jika JSON menyimpan data bertingkat. Gambar 2.6 merupakan array json.



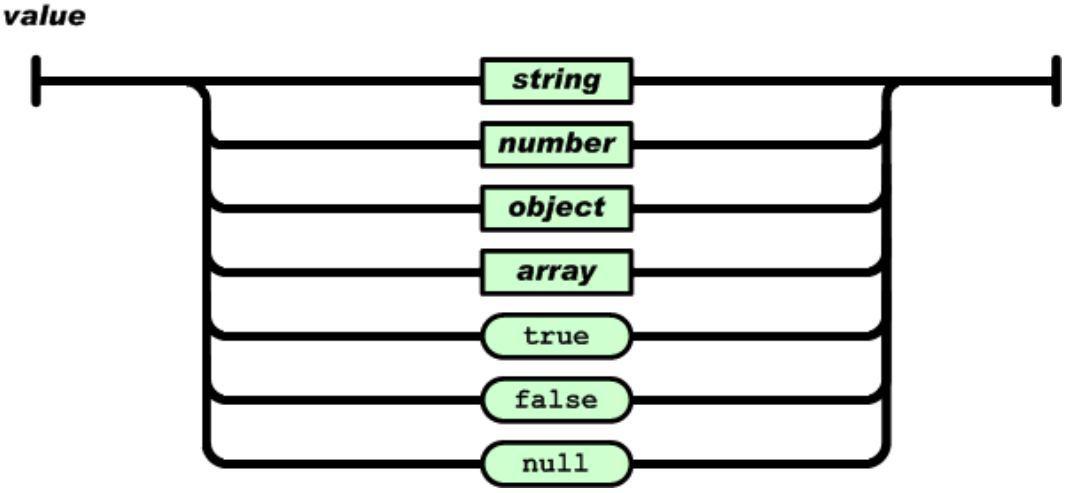
Sumber gambar : http://www.json.org/

Gambar 2.6 Array JSON

Bentuk data JSON objek dan larik dapat saling dikombinasikan untuk mendukung struktur data yang lebih kompleks. JSON mendukung beberapa tipe data untuk menjadi value seperti Angka, String, Bollean dan Nilai NULL.

3. Nilai

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 . Berikut gambar 2.7 *Value* JSON.



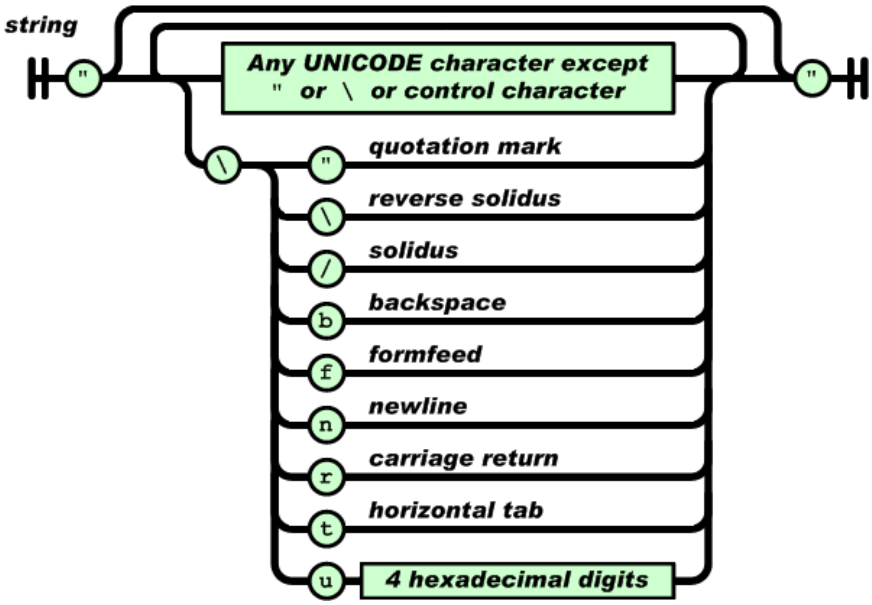
Sumber gambar : http://www.json.org/

Gambar 2.7 Value JSON

4. String

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 . Berikut gambar 2.8 String JSON.

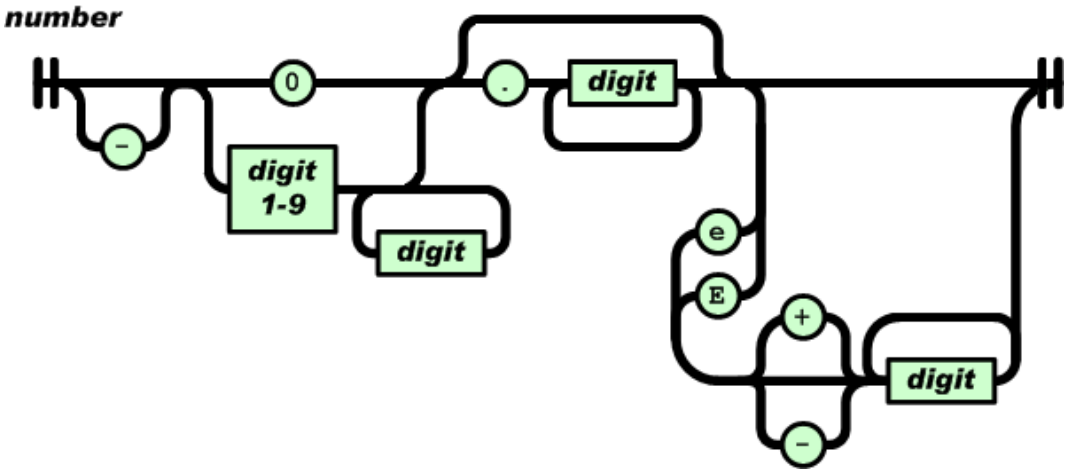


Sumber gambar : http://www.json.org/

Gambar 2.8 String JSON

5. Angka

   Angka, Format oktal dan heksadesimal tidak digunakan . Berikut gambar 2.9 Number JSON.



Sumber gambar : http://www.json.org/

Gambar 2.9 Number JSON

2.7 API

API adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan system operasi. API atau *Application Programming Interface* juga merupakan suatu dokumentasi yang terdiri dari antar muka, fungsi, kelas, struktur untuk membangun sebuah perangkat lunak.

Dengan adanya API, maka memudahkan seorang programmer untuk membongkar suatu *software* untuk kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya. Suatu rutin standar yang memungkinkan *developer* menggunakan *system function*. Proses ini dikelola melalui *operating system*. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya untuk saling berinteraksi .
Keuntungan dengan menggunakan API adalah sebagai berikut:

1. Portabilitas

Developer yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah terinstal API tersebut.

2. Lebih Mudah Dimengerti

API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa system call. Hal ini sangat penting dalam hal editing dan pengembangan. System call interface ini berfungsi sebagai penghubung antara API dan system call yang dimengerti oleh sistem operasi. System call interface ini akan menerjemahkan perintah dalam API dan kemudian akan memanggil system calls yang diperlukan. Untuk membuka suatu file tersebut user menggunakan program yang telah dibuat dengan menggunakan bantuan API, maka perintah dari user tersebut diterjemahkan dulu oleh program menjadi perintah open().

Perintah open() ini merupakan perintah dari API dan bukan perintah yang langsung dimengerti oleh kernel sistem operasi. Oleh karena itu, agar keinginan

pengguna dapat dimengerti oleh sistem operasi, maka perintah open() tadi diterjemahkan ke dalam bentuk *system call* oleh *system call interface*. Implementasi perintah open() tadi bisa bermacam-macam tergantung dari sistem operasi yang digunakan .

Cara menggunakan API :

1. Dilakukan dengan mengimpor package/kelas.
2. Ada beberapa kelas bernama sama dipackage yang berbeda, yaitu :

a) Import salah satu dan gunakan nama lengkap untuk yang lain.
b) Gunakan nama lengkap semua kelas

Kebanyakan Sistem Operasi seperti Windows, menyediakan fasilitas API sehingga programmer dapat melakukan aktivitas programming dengan lebih konsisten. Meskipun API didesain untuk programer, namun API juga baik untuk user karena setidaknya dapat menjamin bahwa program tersebut memiliki *interface*
yang sama, sehingga lebih mudah untuk dipelajari.

2.8 Clarifai API

Clarifai adalah alat pengenalan gambar dan video yang secara otomatis memberikan tag ke obyek dan kategori mengambil hanya piksel sebagai input, menggunakan library semantik dan visual untuk kecerdasan buatan. Sistem ini didasarkan pada jaringan saraf, teknik pembelajaran mesin scalable yang dapat menangani skala besar konten visual yang mengalir melalui API. Clarifai juga menggunakan kesamaan semantik dan visual untuk menyeberangi membandingkan
gambar yang diunggah dengan gambar lainnya di library mereka untuk menampilkan kesamaan [14]

2.8.1 Model Clarifai

Ketika gambar atau video dijalankan mereka ditandai menggunakan model. Sebuah model adalah classifier yang terlatih yang dapat mengenali apa yang ada di

dalam gambar atau video sesuai dengan apa yang diketahui . model yang berbeda digunakan untuk mengetahui hal-hal yang berbeda. Menjalankan gambar atau video
melalui model yang berbeda dapat menghasilkan hasil yang berbeda secara drastis.

1. *The* '*Food'* Model

Model Makanan menganalisa gambar dan video dan skor probabilitas pengembalian kemungkinan bahwa gambar berisi bahan makanan diakui dan hidangan. Model saat ini dirancang untuk mengidentifikasi item makanan tertentu dan bahan-bahan yang terlihat [14].

2. *General* Model

General Model berisi berbagai macam tag di berbagai topik yang berbeda. Dalam kebanyakan kasus, tag kembali dari model umum cukup akan mengenali apa yang ada di dalam gambar Anda.

3. *Travel* Model

Model Travel menganalisa gambar dan skor probabilitas pengembalian kemungkinan bahwa gambar berisi kategori perjalanan terkait diakui. Model saat ini dirancang untuk mengidentifikasi fitur khusus dari perumahan, hotel dan properti perjalanan terkait.

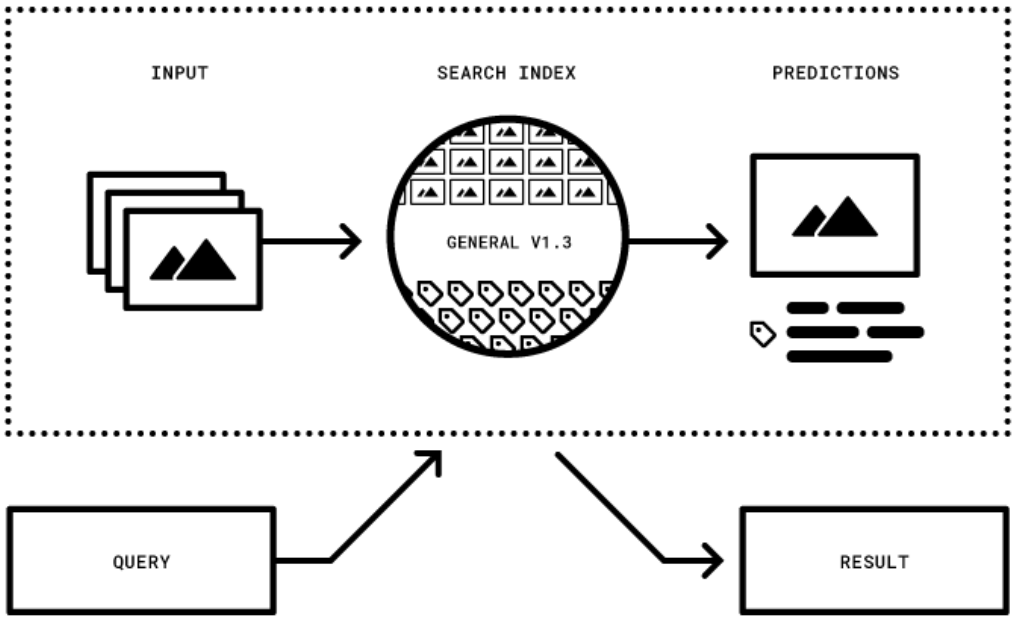
4. NSFW Model

NSFW (*Not Safe For Work*) Model analisis gambar dan video dan skor probabilitas pengembalian kemungkinan bahwa gambar berisi pornografi.

2.8.2 Cara Kerja Clarifai API

Teknologi Clarifai bergantung pada penggunaan Convolutional Neural Networks CNN untuk memproses gambar, dan kemudian menghasilkan daftar tag gambar. CNNs secara de didefinisikan sebagai model pembelajaran mesin

hirarkis, yang belajar gambar yang kompleks representasi dari volume data yang besar dijelaskan. Mereka menggunakan beberapa lapisan transformasi dasar yang akhirnya menghasilkan representasi yang sangat canggih. Clarifai bekerja melalui analisis gambar untuk menghasilkan daftar deskripsi tag dari gambar yang diberikan. Untuk setiap tag dalam daftar ini, sistem juga menyediakan nilai probabilitas. probabilitas ini merupakan kemungkinan menggambarkan gambar menggunakan tag spesifik. The Clarifai API dapat diakses sebagai layanan web jarak jauh. Skema kerja teknologi Clarifai ditampilkan pada Gambar 2.10 dalam skema ini, Clarifai menggunakan gambar ViDRILO sebagai masukan dan menggunakan CNNs untuk menganalisis dan menghasilkan daftar label dan probabilitas.



Sumber gambar : https://clarifai.com/developer/guide/

Gambar 2.10 Skema Proses Clarifai API

Clarifai API dapat diakses menggunakan pilihan trial atau digunakan di bawah

pembayaran mode. Dalam karya ini, mode percobaan telah digunakan untuk mendapatkan tag gambar. Mode ini memungkinkan untuk mendapatkan tag untuk

sejumlah terbatas gambar (10000), dan membatasi jumlah panggilan per jam untuk

1000. Menggunakan mode percobaan, kita hanya bisa memperoleh 20 tag per gambar dengan probabilitas yang terkait. Pada Gambar. 2.10 kita menunjukkan gambar dikirim ke API dan kembali label / probabilitas. dari ini label.

2.9 Google Fit API

Google Fit API adalah sebuah *open platform* yang memungkinkan pengguna untuk mengontrol kebugaran dan kesehatannya. Google fit merupakan bagian dari Google Play Services dan didukung mulai dari Android 2.3 (API level 9) atau lebih tinggi [15].

Beberapa fitur dari Google Fit API adalah :

a) Sensors

Sensor menyediakan akses ke aliran data sensor mentah dari sensor yang tersedia di perangkat android dan dari sensor yang tersedia di perangkat pendamping, seperti *wearable device*.

b) Recording API

Recording API menyediakan penyimpanan otomatis data kebugaran. Google Fit menyimpan data kebugaran dari jenis yang ditentukan di *background*.

c) History

History menyediakan akses ke riwayat kebugaran dan memungkinkan aplikasi melakukan operasi massal, seperti menyisipkan, menghapus, dan membaca data kebugaran. Aplikasi juga dapat mengimpor data kumpulan ke Google Fit.

d) Sessions

Sessions menyediakan fungsionalitas untuk menyimpan data kebugaran dengan metadata *sessions*. *Sessions* mewakili interval waktu di mana pengguna melakukan aktivitas kebugaran.

e) Goals

Goals menyediakan cara untuk menetapkan tujuan yang telah ditetapkan pengguna untuk kemajuan kesehatan dan kebugaran mereka.

Google Fit menyertakan dukungan untuk sensor pada perangkat seluler dan sensor *Bluetooth Low Energy* yang dipasangkan dengan perangkat. Google Fit memungkinkan pengembang menerapkan dukungan untuk sensor lain dan mengeksposnya sebagai sensor perangkat lunak di aplikasi Android. Sensor yang didukung oleh Google Fit tersedia untuk aplikasi Android sebagai objek sumber data.

2.10 Nutritionix API

Nutritionix adalah mesin pencari dan basis data untuk informasi gizi. Pengguna dapat menelusuri atau mencari berdasarkan restoran dan makanan untuk menemukan informasi nutrisi. Pengguna juga dapat membagikan data gizi mereka dengan Nutritionix. API Nutritionix memungkinkan pengembang untuk mengakses dan mengintegrasikan fungsionalitas dan data Nutritionix dengan aplikasi lain dan untuk membuat aplikasi baru. Saat ini sekitar 779.903 jenis makanan di seluruh dunia tersimpan di dalam *database* Nutritionix [16].

2.11 Sensor Accelerometer

Percepatan merupakan suatu keadaan berubahnya kecepatan terhadap waktu. Bertambahnya suatu kecepatan dalam suatu rentang waktu disebut juga percepatan (*acceleration*). Jika kecepatan semakin berkurang daripada kecepatan sebelumnya, disebut deceleration [3].

Bergantung pada arah/orientasi karena merupakan penurunan kecepatan yang merupakan besaran vektor. Berubahnya arah pergerakan suatu benda akan menimbulkan percepatan pula. Untuk memperoleh data jarak dari sensor accelerometer, diperlukan proses integral ganda terhadap keluaran sensor .

Pada *smartphone Accelerometer* merupakan sensor yang bisa membaca pergerakan sehingga dapat mengubah tampilan layar dari posisi *landscape* ke *portrait* atau sebaliknya dengan cukup memiringkan badan ponsel secara otomatis.

Accelerometer dapat diimplementasikan pada beberapa bidang Salah satu pengguaan *Accelerometer* yang sangat umum yaitu dalam sistem *airbag* yang terdapat pada kendaraan, khususnya mobil. *Accelerometer* ini digunakan untuk

mendeteksi penurunan percepatan yang sangat besar yang biasanya terjadi ketika terjadinya tabrakan antar kendaraan.

Sport Watch, berupa jam tangan olahraga yang juga dapat menghitung berapa banyak langkah yang telah lakukan, *Accelerometer* juga digunakan untuk menghitung kecepatan dan jarak dari si pelari yang menggunakannya. *Accelerometer* banyak digunakan untuk menghitung percepatan dan penurunan percepatan dari sebuah kendaraan. *Accelerometer* membantu untuk mengevaluasi performansi dari mesin dan sistem percepatan dan juga *breaking system* (sistem penurunan percepatan).

Kecepatan yang biasa ditampilkan pada kendaraan anda umumnya didapatkan dari penggunaan *Accelerometer*. Selain itu juga biasa digunakan untuk menghitung vibrasi pada kendaraan, mesin, bangunan, dan sistem keamanan pada kendaraan (*safety installation*). *Accelerometer* juga dapat mengkalkulasi percepatan yang diakibatkan oleh gravitasi bumi. *Accelerometer* yang menghitung gravitasi secara spesifik digunakan pada *gravimetry*, disebut sebagai *gravimeter*. *Notebook* atau *laptop* juga dilengkapi dengan *Accelerometer* untuk mengevaluasi goncangan yang dirasakan oleh *laptop* tersebut.

Accelerometer pada *laptop* biasanya digunakan pada sistem *Sudden Motion Sensor*, yang biasa digunakan untuk mendeteksi jatuhnya *laptop*. Jika kondisi pada saat jatuh terdeteksi, *hard diskdrive* yang ada akan diproteksi sehingga tidak terjadi *data loss*. Sekarang ini juga terdapat *notebook* yang menggunakan *Accelerometer* untuk secara otomatis mengubah arah layar (menjadi miring ataupun terbalik) sesuai dengan arah monitor tersebut ditegakkan (*portrait* atau *landscape*).

Terdapat juga sejumlah ponsel yang menggunakan *Accelerometer* untuk mengubah lagu yang dimainkan (*Track Switching*). *Camera recorder* menggunakan *Accelerometer* untuk menstabilkan gambar. Kamera digital menggunakan *Accelerometer* untuk menu pilihan anti blur ketika mengambil gambar. Baru-baru ini Apple.Inc memperkenalkan sebuah gebrakan dengan mengkombinasikan 2 sensor gerakan yaitu antara *Accelerometer*

dan *Gyroscope* pada sebuah perangkat ponsel. Ini akan menyempurnakan fitur dari ponsel yang hanya menggunakan *Accelerometer* dalam mendeteksi gerakan. Dengan kombinasi ini maka akan didapatkan 6 sumbu gerakan yaitu 3 sumbu linier (atas-bawah, kanan-kiri, depan-belakang) dan 3 sumbu rotasi (rotasi *roll,pitch and yaw* seperti pada gambar 2). 1 keunggulan lagi dari kombinasi ini adalah akan didapatkan *output* gambar yang tiap detil gerakannya lebih halus dari pada perangkat ponsel yang hanya menggunakan *Accelerometer.*

2.12 Unified Modeling Language (UML)

Unified Modeling Languange (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). Definisi ini merupakan definisi yang sederhana. Pada kenyataannya, pendapat orang-orang tentang UML berbeda satu sama lain. Hal ini dikarenakan oleh sejarahnya sendiri dan oleh perbedaan persepsi tentang apa yang membuat sebuah proses rancang-bangun perangkat lunak efektif [17].

Dalam kerangka spesifikasi, UML menyediakan model-model yang tepat,tidak mendua-arti (ambigu), serta lengkap. Secara khusus, UML menspesifikasi langka-langka penting dalam pengambilan keputusan analisis, perancangan, serta implementasi dalam sistem yang sangat bernuansa perangkat lunak (*software intensive system*). Dalam hal itu, UML bukanlah merupakan Bahasa pemrograman tetapi model-model yang tercipta berhubungan langsung dengan berbagai macam Bahasa pemrograman perorientasi objek, katakanlah Java, Borland Delphi, Visual BASIC, C++, dan lain-lain [17].

UML lahir dari penggabungan banyak Bahasa pemodelan grafis berorientasi objek yang berkembang pesat pada akhir 1980-an dan awal 1990-an. Sejak kehadirannya pada tahun 1997, UML menghancurkan Menara Babel tersebut menjadi sejarah. Pada intinya peran UML dalam pengembangan perangkat lunak, orang-orang memiliki cara-cara yang berbeda dalam penggunaanya, perbedaan-

perbedaan yang masih dibawa dari Bahasa-bahasa pemodelan grafis lain. Perbedaan-perbedaan ini mengakibatkan perselisihan yang panjang dan keras tentang bagaimana UML seharusnya digunakan.

Berdasarkan pemaparan mengenai *Unified Modeling Languange* (UML) dari beberapa sumber referensi, maka dapat disimpulkan UML merupakan alat bantu dalam melakukan pemodelan yang saling berhubungan secara langsung dalam pembangunan sebuah sistem agar lebih efektif.

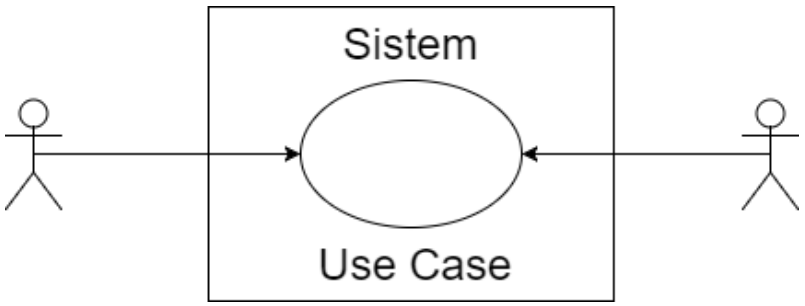
2.12.1 Use Case Diagram

Use Case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. Use Case bekerja dengan cara mendeskripsikan tipikal interaksi antara pengguna sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai [17].

Use Case diagram digunakan untuk menggambarkan analisis kebutuhan dari sistem dari level atas melalui fungsionalitas dari sistem dan interaksi diantara para aktor. Aktor adalah sesuatu yang berinteraksi dengan sistem.

Secara umum, tujuan dari use case diagram adalah sebagai berikut :

1. Digunakan untuk mengumpulkan kebutuhan dari sebuah sistem
2. Untuk mendapatkan pandangan dari luar sistem
3. Untuk mengidentifikasi factor yang mempengaruhi sistem baik internal maupun eksternal
4. Untuk menunjukkan interaksi dari para actor dari sistem Ilustrasi dari actor, use case dan boundary dapat dilihat pada gambar .



Sumber gambar :

https://www.uml-diagrams.org/use-case-diagrams-examples.html

Gambar 2.11 Use Case Diagram

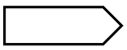
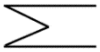
2.12.2 Activity Diagram

Activity Diagram adalah bagian penting dari UML yang menggambarkan aspek dinamis dari sistem. Logika procedural, proses bisnis, dan aliran kerja suatu bisnis bias dengan mudah dideskripsikan dalam activity diagram. Activity diagram memiliki peran seperti halnya flowchart, akan tetapi perbedaannya adalah activity diagram bisa mendukung perilaku pararel [17].

Tujuan dari activity diagram adalah untuk menangkap tingkah laku dinamis dari sistem dengan cara menunjukkan aliran pesan dari satu aktifitas ke aktifitas lainnya. Secara umum tujuan activity diagram adalah sebagai berikut :

1. Menggambarkan aliran aktivitas dari sistem
2. Menggambarkan urutan aktifitas dari satu aktifitas ke aktifitas lainnya
3. Menggambarkan pararelisme, percabangan dan aliran konkuren dari sistem

Tabel 2.4 Simbol-simbol yang dipakai di Activity Diagram

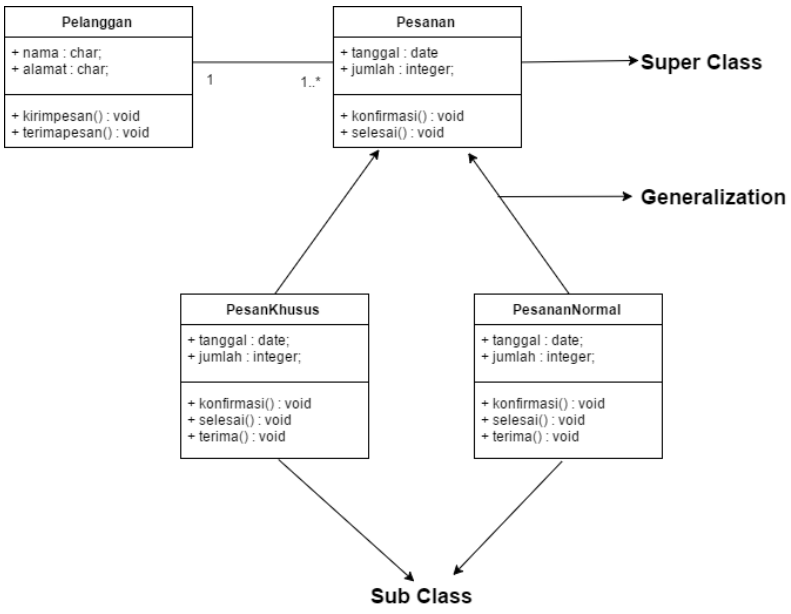
Simbol	Keterangan
	Titik Awal.
	Titik Akhir.
	Activity.
	Pilihan Untuk Pengambilan Keputusan (percabangan).
	Fork ; digunakan untuk menunjukan kegiatan yang dilakukan secara pararel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	Rake ; menunjukan adanya dekomposisi.
	Tanda Waktu.
	Tanda Pengiriman.
	Tanda Penerimaan.
	Aliran Akhir (*Flow Final*).

2.12.3 *Class Diagram*

Class Diagram adalah diagram statis yang mewakili pandangan statis dari suatu aplikasi. Class diagram tidak hanya digunakan untuk memvisualisasikan, menggambarkan, dan mendokumentasikan berbagai aspek sistem tetapi juga untuk membangun kode eksekusi (*executetable code*) dari aplikasi perangkat lunak. *Class diagram* menunjukan koleksi kelas, antarmuka, asosiasi, kolaborasi, dan constraint. *Class diagram* juga dikenal sebagai diagram struktural [17]

Tujuan dari *class diagram* adalah untuk memodelkan pandangan statis suatu aplikasi. Secara lebih rinci tujuannya adalah sebagai berikut :

1. Analisis dan desain pandangan statis aplikasi.
2. Menjelaskan tanggung jawab suatu sistem.
3. Basis untuk diagram komponen dan penyebaran (*deployment*)
4. *Forward and reverse engineering*

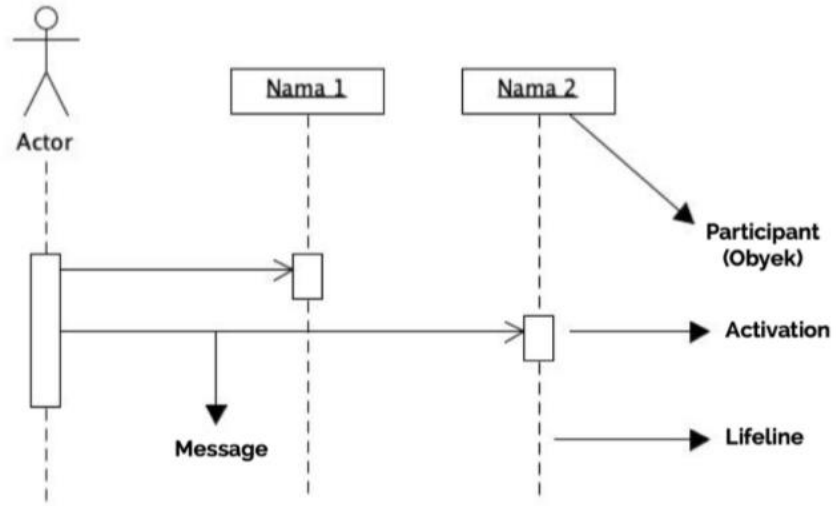


Gambar 2.12 Contoh Class Diagram Sistem Pemesanan

2.12.4 *Sequence Diagram*

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah scenario. Diagram ini menunjukan sejumlah contoh obyek dan *message* (pesan) yang diletakkan diantara obyek-obyek ini di dalam use case.

Komponen utama *sequence diagram* terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical*. Simbol – symbol yang ada pada *sequence diagram* dapat dilihat pada gambar 15.



Gambar 2.13 Simbol – simbol yang ada pada Sequence Diagram

2.12.5 Bangun Dasar UML

Metodologi UML menggunakan 3 bangunan dasar untuk mendeskripsikan sistem/perangkat lunak yang akan dikembangkan, yaitu :

1. Sesuatu (*Things*).
   Ada 4 macam '*things*' dalam UML, yaitu :
   a. *Structural Things.*
   b. *Behavioral Things.*
   c. *Grouping Things.*
   d. *Annotational Things.*
2. Relasi (*Relationship*).

Yang dimaksud *relationship* adalah hubungan-hubungan yang terjadi antarelemen dalam UML. Hubungan-hubungan ini penting sekali dalam UML. Dapat dikatakan, tidak mungkin membuat model-model UML tanpa *relationship* ini.

3. *Diagrams*.

Setiap sistem yang kompleks seharusnya bisa dipandang dari sudut yang berbeda-beda sehingga kita bisa mendapatkan pemahaman secara menyeluruh.
Secara umum UML diterapkan dalam pengembangan sistem/perangkat lunak berorientasi objek sebab metodologi UML ini umumnya memiliki keunggulan-keunggulan, yaitu :

1. ***Uniformity.*** Dengan metodologi UML (atau metodologi berorientasi objek pada umumnya), para pengembang cukup menggunakan 1metodologi dari tahap analisis hingga perancangan. UML juga memungkinkan kita merancang komponen antarmuka penguna (*User Interface*) secara terintegrasi bersama dengan perancangan perangkat lunak sekaligus dengan perancangan basis data.
2. ***Understandability.*** Dengan metodologi ini kode yang dihasilkan dapat diorganisasi kedalam kelas-kelas yang berhubungan dengan masalah sesungguhnya sehingga lebih mudah dipahami siapa pun juga.
3. ***Stability.*** Kode program yang dihasilkan relative stabil sepanjang waktu sebab sangat mendekati permasalahan sesungguhnya dilapangan.
4. ***Reusability.*** Dengan metodologi perorientasi objek, dimungkinkan penggunaan ulang kode, sehingga pada gilirannya akan sangat mempercepat waktu untuk pengembangan perangkat lunak (atau sistem informasi).

2.13 Pengujian Black Box

Black-Box Testing merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program [15].

Black Box Testing berfokus pada spesifikasi fungsional dari perangkat lunak. Tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program.

Black Box Testing bukanlah solusi alternatif dari *White Box Testing* tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing.*

Black Box Testing cenderung untuk menemukan hal-hal berikut [15]:

1. Fungsi yang tidak benar atau tidak ada.
2. Kesalahan antarmuka (*interface errors*).
3. Kesalahan pada struktur data dan akses basis data.
4. Kesalahan performansi (*performance errors*).
5. Kesalahan inisialisasi dan terminasi.