

BAB 2

LANDASAN TEORI

2.1 Pengertian Makanan

Makanan adalah segala sesuatu yang dapat dimakan dan setelah dicerna serta diserap tubuh akan berguna bagi kesehatan dan kelangsungan hidup. Menurut sumber, makanan adalah bahan, biasanya berasal dari hewan atau tumbuhan, yang dimakan oleh makhluk hidup mendapatkan tenaga dan nutrisi. [1]

2.1.2 Jenis Makanan

1. Jenis Makanan adalah macam – macam makanan yang bisa untuk dimakan , Contohnya : nasi dengan berbagai cara memasaknya seperti nasi uduk, nasi goreng, nasi kuning, dan sebagainya.
2. Sayur dengan berbagai jenisnya seperti: sayur kangkung, bayam, pare, kacang panjang, dan jenis sayur lainnya.
3. Lauk pauk dengan berbagai jenisnya seperti: daging, ikan, tempe, tahu, dan lain-lain
4. Buah dengan berbagai macamnya seperti: buah mangga, jeruk, apel, nanas, dan sebagainya dan.
5. Berbagai jenis makanan lainnya.

2.1.3 Bahan Makanan

Bahan makanan/pangan adalah segala sesuatu yang dapat dimasak/diolah kemudian disajikan sebagai hidangan. Contohnya: beras, jagung, singkong, telur, dan sebagainya.

Bahan makanan dikelompokkan menjadi 4 kelompok sebagai berikut:

1. Bahan makanan pokok.
2. Bahan makanan lauk pauk.
3. Bahan makanan sayur mayur.
4. Bahan makanan buah.

2.2 Resep Masakan

Resep Masakan adalah suatu susunan intruksi atau algoritma yang menunjukkan cara membuat suatu masakan.

Dalam sebuah resep masakan harus terdapat :

1. Nama Masakan
2. Komposisi atau bahan dengan kuantitasnya
3. Alat-alat yang dibutuhkan
4. Cara Memasak
5. Lama Waktu memasak
6. Jumlah Sajian
7. Perkiraan Jumlah Kalori
8. Ketahanan makanan dan penyimpanan

2.3 Kuliner

Pengertian kuliner adalah hasil olahan yang berupa masakan berupa lauk-pauk, panganan maupun minuman. Kuliner tidak terlepas dari kegiatan masak-memasak yang erat kaitannya dengan konsumsi makanan sehari-hari. Kata kuliner merupakan unsur serapan bahasa Inggris yaitu *culinary* yang berarti berhubungan dengan memasak. Sedangkan orang yang bekerja di bidang kuliner disebut koki atau *chef*. Saat ini istilah kuliner sering didengar, dibaca lewat media cetak maupun audio visual.

2.4 Istilah Kuliner

Dalam perkembangannya, penggunaan istilah kuliner digunakan untuk berbagai macam kegiatan, seperti Seni kuliner yaitu seni persiapan, memasak dan penyajian makanan, biasanya dalam bentuk makanan.

1.5 Android

Android adalah system operasi untuk handphone yang berbasis Linux. Android menyediakan platform terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya,

Google Inc. membeli Android Inc. pendatang baru yang membuat peranti lunak untuk handphone. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi. [4]

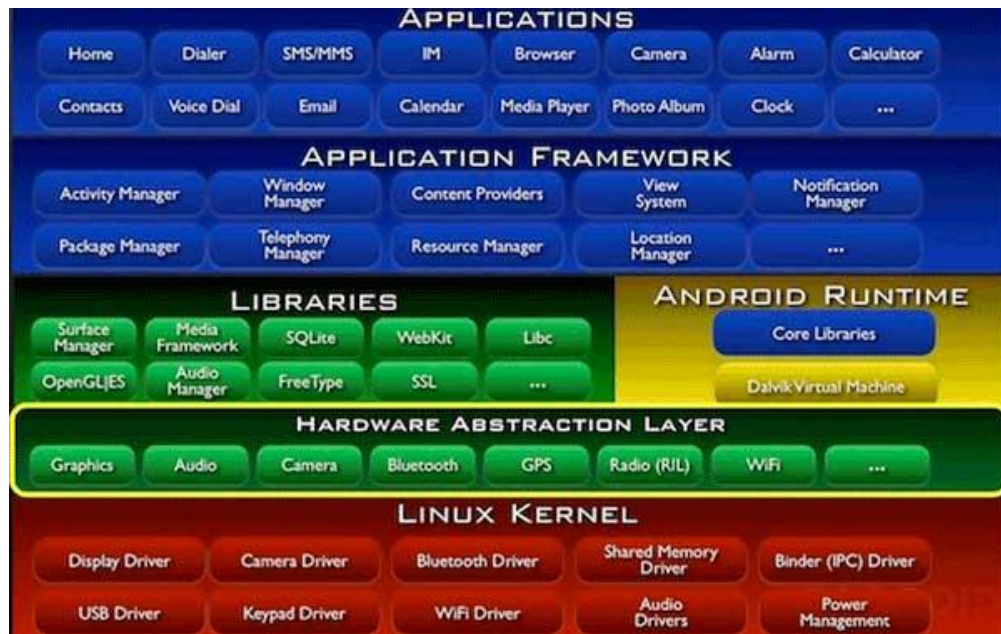
Pada saat perilisan perdana Android, 5 November 2007, Android bersama Open Handset Alliance menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android di bawah lisensi Apache, sebuah lisensi perangkat software dan standar terbuka perangkat 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 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.

2.5.1 Arsitektur Android

Arsitektur Android dapat digambarkan seperti pada gambar 2.1 Arsitektur Android secara garis besar Arsitektur Android dapat dijelaskan sebagai berikut [4]:



Gambar 2.1 Arsitektur Android

1. *Application dan Widget*

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

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 fungsi-fungsi 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-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.10 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 v1.1 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 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 v1.6 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 v2.0 – 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 v2.2 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 v2.3 Gingerbread

Dirilis pada tanggal 6 Desember 2010 bersamaan dengan dihidirkannya 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 v3.0 – 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 v4.0 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 dengan versi sebelumnya adalah dari segi User interface yang nampak lebih bersih dan elegan. 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 v4.1 – 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 v4.4 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 programmer 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 v5.0 – 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 v6.0 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.

14. Android v7.0 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.

2.5.2 Android Life Cycle

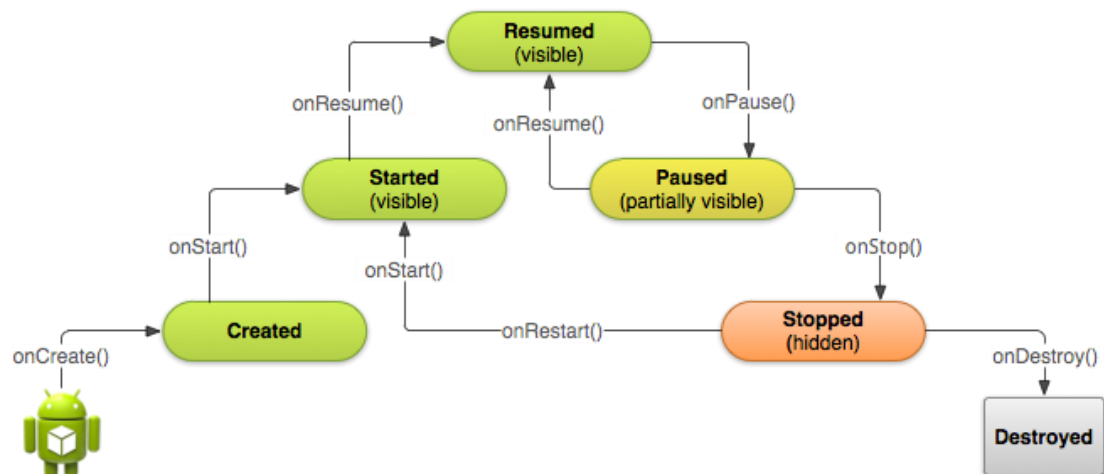
Aplikasi android terdiri dari beberapa fungsi dasar seperti mengedit catatan, memutar file musik, membunyikan alarm, atau membuka kontak telepon. Fungsi-fungsi tersebut dapat diklasifikasikan ke dalam empat komponen android yang berbeda seperti ditunjukkan pada, klasifikasi tersebut berdasarkan kelas-kelas dasar java yang digunakan [4].

Tabel 2.1 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*, *BroadcastReceiver*, *ContentProvider*.

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.



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()`.

2.5.3 Fitur Android

Android memiliki beberapa fitur utama yang sering digunakan dalam proses pembangunan aplikasi diantaranya adalah [4]:

1. *Multi-proses* 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 pengguna tidak benar.

2. *Touch Gestures* dan *Multi-touch*

Touchscreen adalah *user interface* intuitif yang digunakan banyak *smartphone* di dunia. 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.

2.5.4 Prinsip Desain

Android memiliki beberapa prinsip desain yang dapat menjadi acuan dalam membuat desain aplikasi android diantaranya adalah [4]

1. *Multiple Assets*

Android mendukung jutaan smartphone, tablet dan perangkat lain dalam berbagai ukuran layar dan ukuran, untuk itu Multiple Assetssangat disarankan digunakan untuk mengatasi fragmentasi pada android. Seperti ilustrasi pada gambar 2.3 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 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.



Gambar 2.4 Touch Feedback

2.5.5 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 [5] :

- a. *Framework* : Aplikasi yang mendukung pengganti komponen dan reusable.
- b. *Dalvik Virtual Machine* dioptimalkan untuk perangkat mobile.
- c. *Integrated Browser* berdasarkan engine open source WebKit.
- d. Grafis yang dioptimalkan dan didukung oleh libraries grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional Ekselerasi hardware)
- e. SQLite untuk penyimpanan data.
- f. Media Support yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PING, GIF), GSM Telephony (tergantung hardware)
- g. Bluetooth, EDGE, 3G, dan WiFi (tergantung hardware)
- h. Kamera, GPS, Kompas, dan Accelerometer (tergantung hardware)

- i. Lingkungan *Development* yang lengkap dan termasuk perangkat emulator, tools untuk debugging, profil dan kinerja memori, dan plugin.

2.6 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.

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).

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 dasa
- 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 memiliki beberapa kelebihan dibandingkan dengan bahasa pemrograman lain, diantaranya:

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

- a. Mudah didekompilasi

Dekompilasi adalah proses membalikkan hasil kompilasi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan bytecode yang menyimpan banyak atribut bahasa tingkat tinggi. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/direverse-engineer.

- b. Pengguna memori yang banyak

Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal.

2.7 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 software 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-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.

2.8 JSON

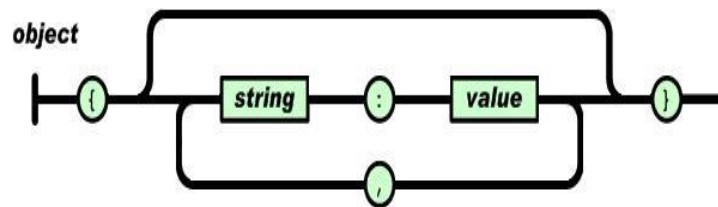
JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemrograman JavaScript, Standar ECMA-262 Edisi ke 3 – Desember 1999. 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 sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur [6]:

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 struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut:

1. Objek

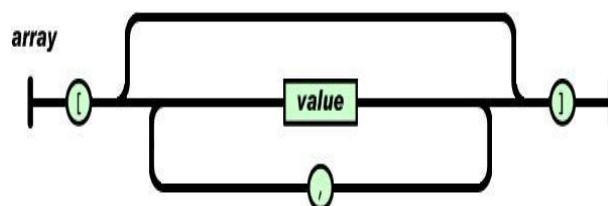
Objek adalah sepasang nama / nilai yang tidak terurutkan. Seperti yang diilustrasikan pada Gambar 2.5 objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma). Objek biasanya digunakan untuk menyimpan data tunggal dalam bentuk JSON [6].



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 2.6. Larik dalam JSON dapat digunakan sebagai value dari JSON object hal ini dapat berguna jika JSON menyimpan data bertingkat.

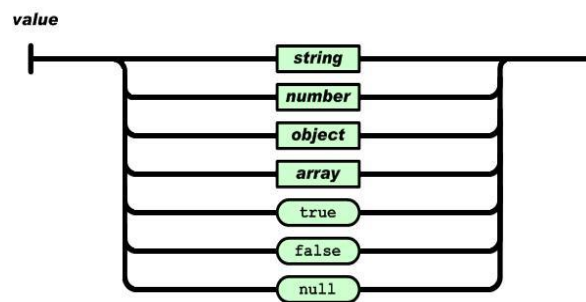


Gambar 2.6 Array JSON

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, Boolean dan Nilai NULL .

3. Nilai

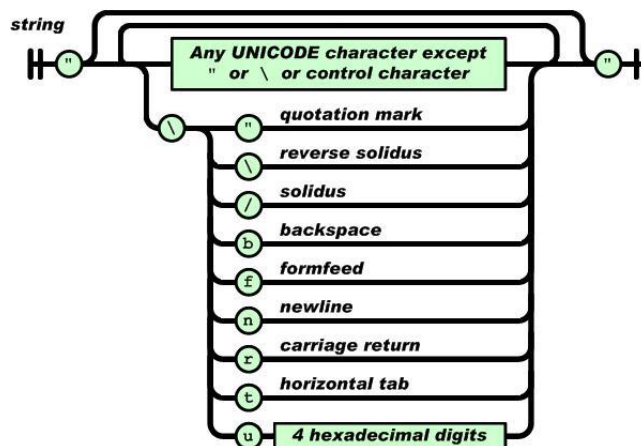
Nilai (value) dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat .



Gambar 2.7 Value JSON

4. String

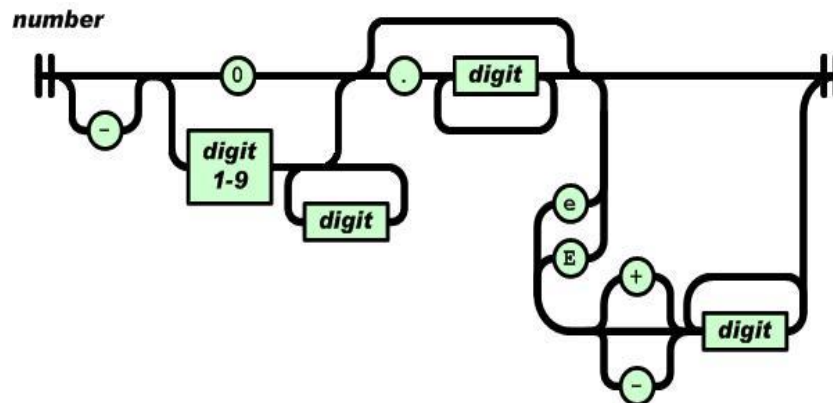
String adalah kumpulan dari nol atau lebih karakter Unicode, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan backslash *escapes* "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java .



Gambar 2.8 String JSON

5. Angka

Angka, Format oktal dan heksadesimal tidak digunakan .



Gambar 2.9 Number JSON

2.9 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 sistem 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 :
 - Import salah satu dan gunakan nama lengkap untuk yang lain
 - 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.

2.10 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

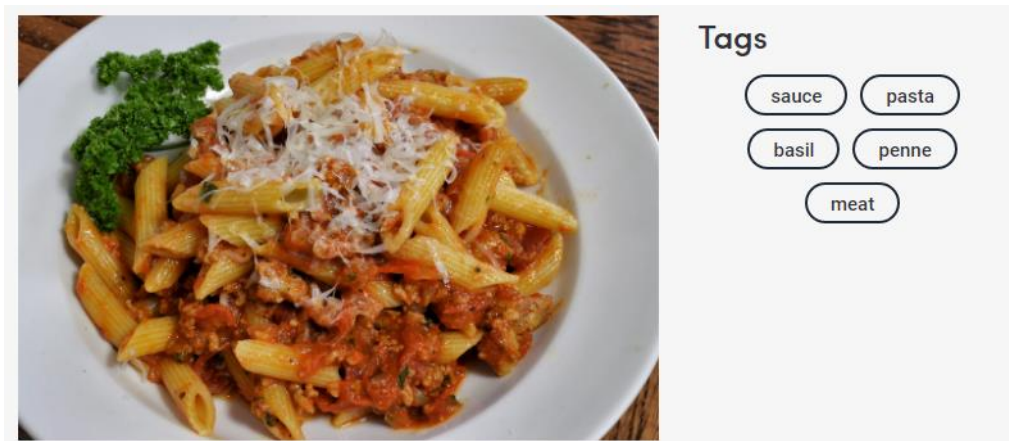
menangani skala besar konten visual yang mengalir melalui API. Clarifai juga menggunakan kesamaan semantik dan visual untuk menyeberangi membandingkan gambar yang diunggah dengan gambar lainnya di library mereka untuk menampilkan kesamaan [7].

2.10.1 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

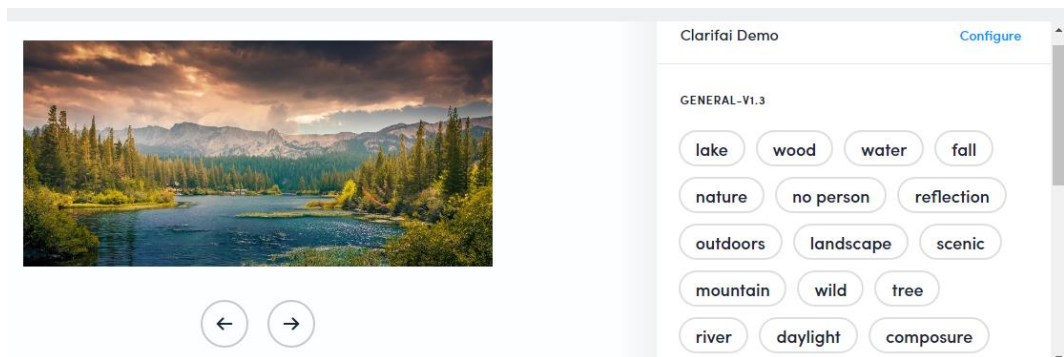
Model Makanan menganalisa gambar dan video dan skor probabilitas pengembalian kemungkinan bahwa gambar berisi bahan makanan diakui dan hidangan. Model saat ini dirancang untuk mengidentifikasi item makanan tertentu dan bahan-bahan yang terlihat [7].



Gambar 2.10 Model Food Items-v1.0

2. General Model

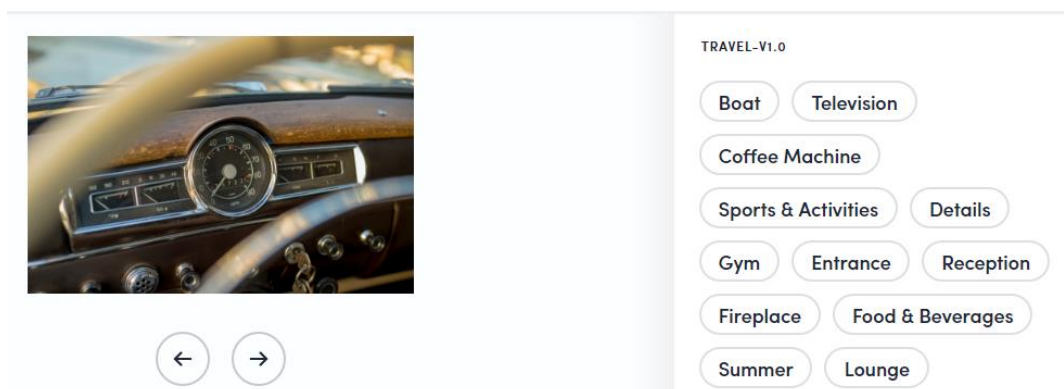
General Model berisi berbagai macam tag di berbagai topik yang berbeda. Dalam kebanyakan kasus, tag kembali dari model umum cukup akan mengenali apa yang ada di dalam gambar Anda.



Gambar 2.11 General Model v.1.3

3. Travel Model

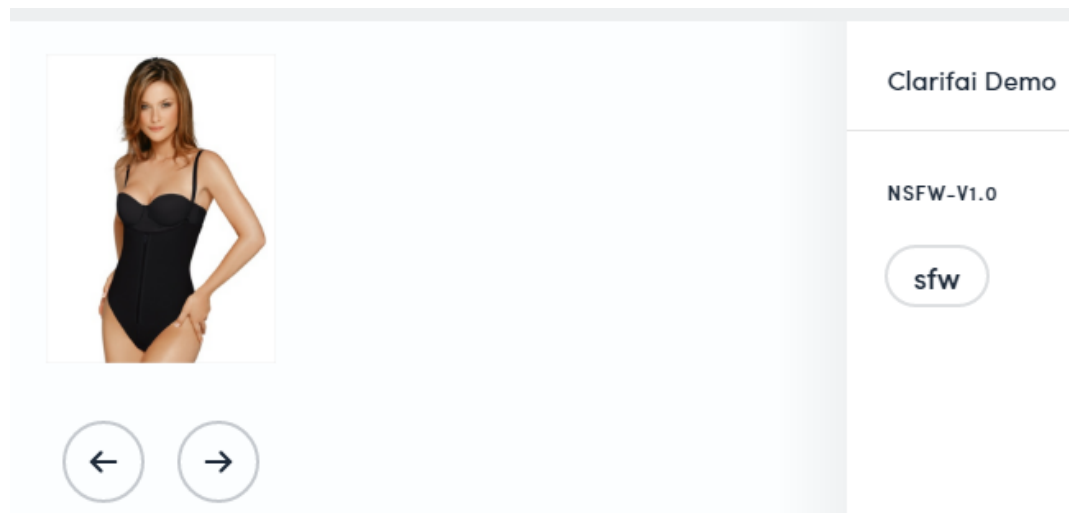
Model Travel menganalisa gambar dan skor probabilitas pengembalian kemungkinan bahwa gambar berisi kategori perjalanan terkait diakui. Model saat ini dirancang untuk mengidentifikasi fitur khusus dari perumahan, hotel dan properti perjalanan terkait.



Gambar 3.12 Travel Model

4. NSFW Model

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

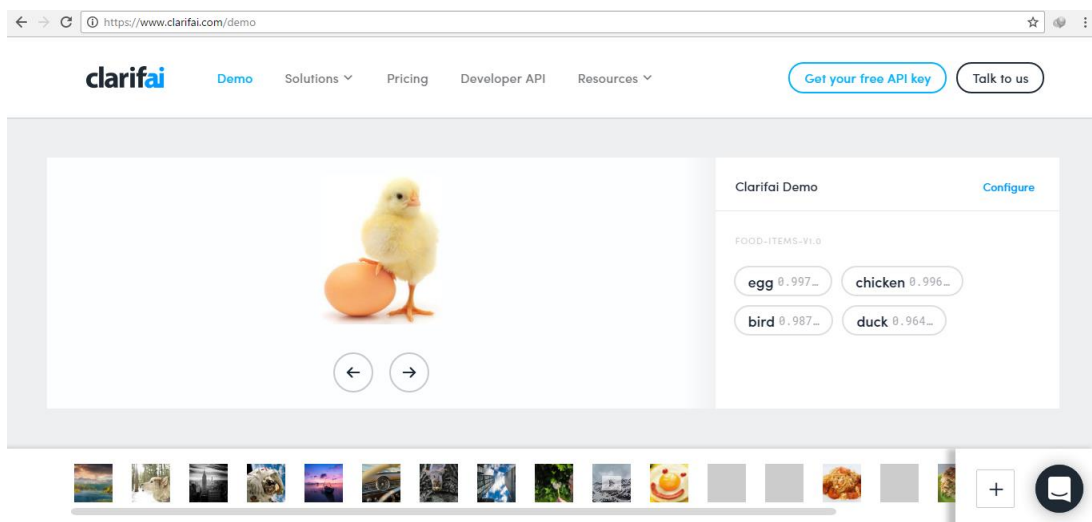


Gambar 3.13 NSFW Model

2.10.2 Cara Kerja Clarifai API

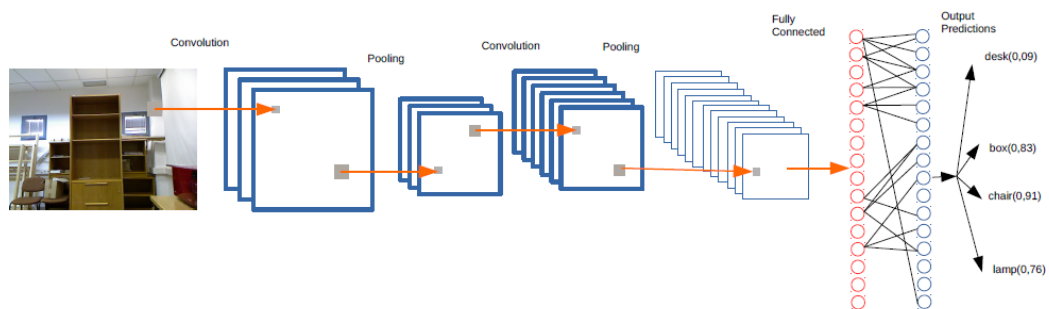
Teknologi Clarifai bergantung pada penggunaan Convolutional Neural Networks CNN

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 [7]



Gambar 2.14 Clarifai Demo

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. 2.15. Dalam skema ini, Clarifai menggunakan gambar ViDRILO sebagai masukan dan menggunakan CNNs untuk menganalisis dan menghasilkan daftar label dan probabilitas.




Gambar 2.15 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. 2.11 kita menunjukkan gambar dikirim ke API dan kembali label / probabilitas. dari ini label, Gambar. 3 menunjukkan beberapa contoh gambar yang dihasilkan oleh API. Gambar-gambar ini dihasilkan dengan menggunakan penjelasan dari ImageNet. [7]

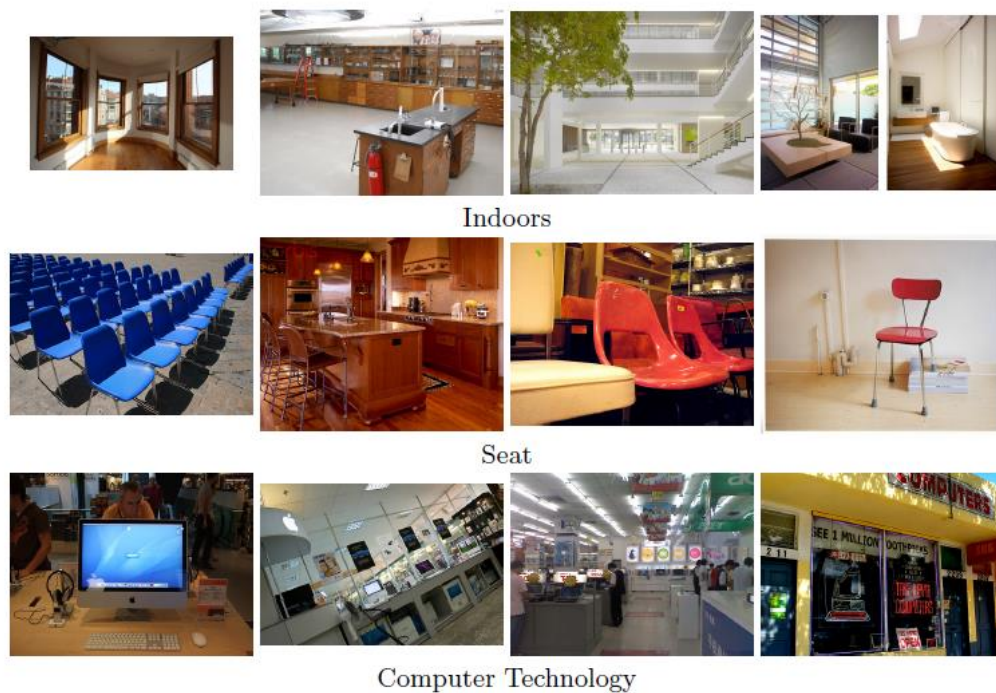
2.10.3 Gambar Deskriptor Anotasi Clarifai

Proses untuk menghasilkan Clarifai deskriptor dimulai dengan langkah awal di mana kita menemukan semua label termasuk dalam domain masalah kita. Hal ini dilakukan dengan memproses semua gambar yang tersedia dengan API Clarifai,

dan kemudian menghapus nilai-nilai label digandakan. Langkah ini dapat dilihat sebagai codebook atau kamus Proses generasi. Setiap gambar input kemudian dikodekan sebagai daftar probabilitas nilai-nilai yang panjangnya ditentukan oleh ukuran kamus. Nilai-I di deskripsi ini akan sesuai dengan probabilitas dan dikembalikan oleh Clarifai untuk menandakan label. Proses ini ditunjukkan pada Gambar 2.16 .

Image	Label	Probability
	Indoors	0.9936
	Seat	0.9892
	Contemporary	0.9787
	Chair	0.9779
	Furniture	0.9744
	Room	0.9634
	Interior design	0.9627
	Window	0.9505
	Table	0.9428
	Computer Technology	0.9417

Gambar 2.17 Label dan probabilitas API Clarifai untuk gambar ViDRILO.



Gambar 2.16 Representasi gambar untuk label diekstraksi

2.11 Object Oriented Analysis Design

Konsep OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang 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. [8]

2.11.1 Unified Modeling Language

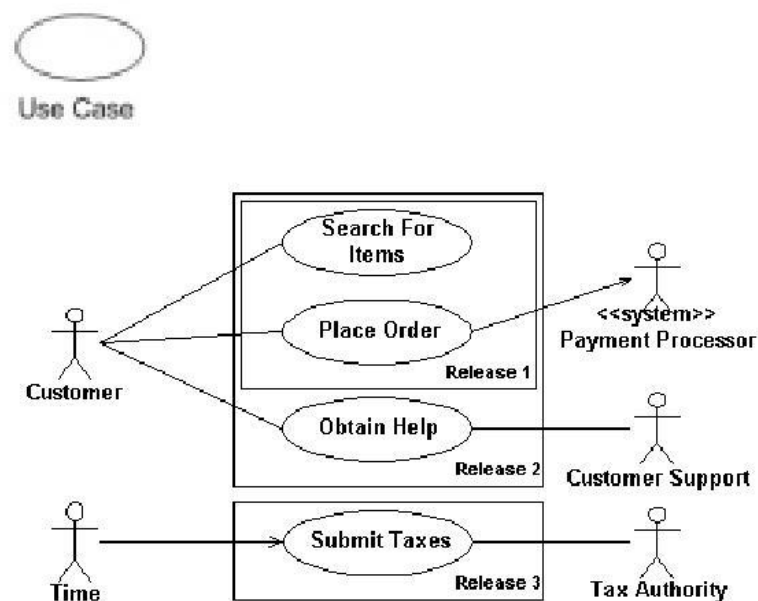
Unified Modelling Language (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual (Braun, et. al. 2001). Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek (Whitten, et. al. 2004). [8]

1. Use Case Diagram

Diagram yang menggambarkan *actor*, *use case* dan relasinya sebagai suatu urutan tindakan yang memberikan nilai terukur untuk aktor. Sebuah *use case* digambarkan sebagai elips horizontal dalam suatu diagram UML *use case*.

Use Case memiliki dua istilah

1. *System use case*; interaksi dengan sistem.
2. *Business use case*; interaksi bisnis dengan konsumen atau kejadian nyata



Gambar 2.18 Use Case Diagram

1) Actor

Aktor adalah segala hal diluar sistem yang akan menggunakan sistem tersebut untuk melakukan sesuatu. Bisa merupakan manusia, sistem, atau *device* yang memiliki peranan dalam keberhasilan operasi dari sistem.



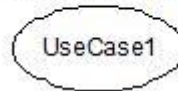
Gambar 2.19 Actor

2) Use Case

Use case merupakan gambaran umum dari fungsi atau proses utama yang menggambarkan tentang salah satu perilaku sistem. Perilaku sistem ini terdefinisi dari proses bisnis sistem yang akan dimodelkan. Tidak semua proses bisnis digambarkan secara fungsional pada *use case*, tetapi yang digambarkan hanya fungsionalitas utama yang berkaitan dengan

sistem. *Use case* menitik beratkan bagaimana suatu sistem dapat berinteraksi baik antar sistem maupun diluar sistem.

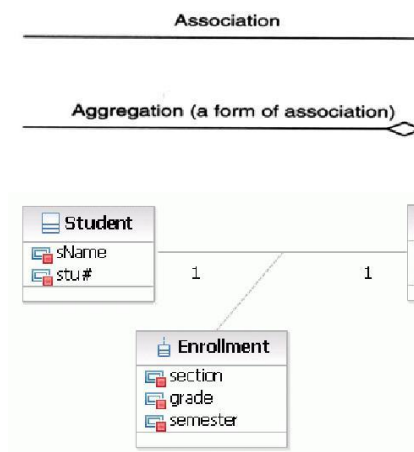
Use-case symbol



Gambar 2.20 Use Case

3) Association

Mengidentifikasi interaksi antara setiap *actor* tertentu dengan setiap *use case* tertentu. Digambarkan sebagai garis antara *actor* terhadap *use case* yang bersangkutan. Asosiasi bisa berarah (garis dengan anak panah) jika komunikasi satu arah, namun umumnya terjadi kedua arah (tanpa anak panah) karena selalu diperlukan demikian.



Gambar 2.21 Association

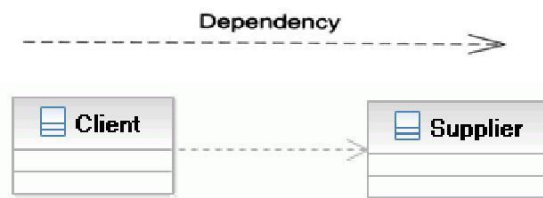
5) Dependency

Dependensi <<include>>

1. Mengidentifikasi hubungan antar dua *use case* dimana yang satu memanggil yang lain.
2. Jika pada beberapa *use case* terdapat bagian yang memiliki aktivitas yang sama maka bagian aktivitas tersebut biasanya dijadikan *use case* tersendiri dengan relasi dependensi setiap

use case semula ke *use case* yang baru ini sehingga memudahkan pemeliharaan.

3. Digambarkan dengan garis putus-putus bermata panah dengan notasi <<include>> pada garis.
4. Arah mata panah sesuai dengan arah pemanggilan
Dependensi <<extend>>
1. Jika pemanggilan memerlukan adanya kondisi tertentu maka berlaku dependensi <<extend>>.
2. Digambarkan serupa dengan dependensi <<include>> kecuali arah panah berlawanan.
3. Note: konsep “*extend*” ini berbeda dengan “*extend*” dalam Java.



Gambar 2.22 Dependency

6) *Generalization*

Mendefinisikan relasi antara dua *actor* atau dua *use case* yang mana salah satunya meng-*inherit* dan menambahkan atau *override* sifat dari yang lainnya. Penggambaran menggunakan garis bermata panah kosong dari yang meng-*inherit* mengarah ke yang di-*inherit*.

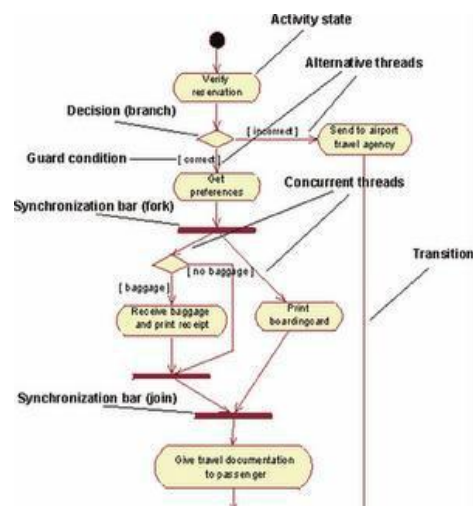


Gambar 2.23 Generalization

2. *Activity Diagram*

Activity Diagram adalah sebuah tahapan yang lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dimana biasanya dipakai pada business modeling untuk

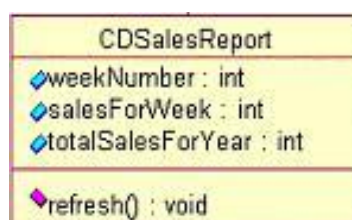
memperlihatkan urutan aktifitas proses bisnis. *Activity Diagram* ini sendiri memiliki struktur diagram yang mirip *Flowchart* atau *data flow diagram* pada perancangan terstruktur. *Activity Diagram* dibuat berdasarkan sebuah atau beberapa *use case* pada *Use Case Diagram*.



Gambar 2.24 Activity Diagram

3. Class Diagram

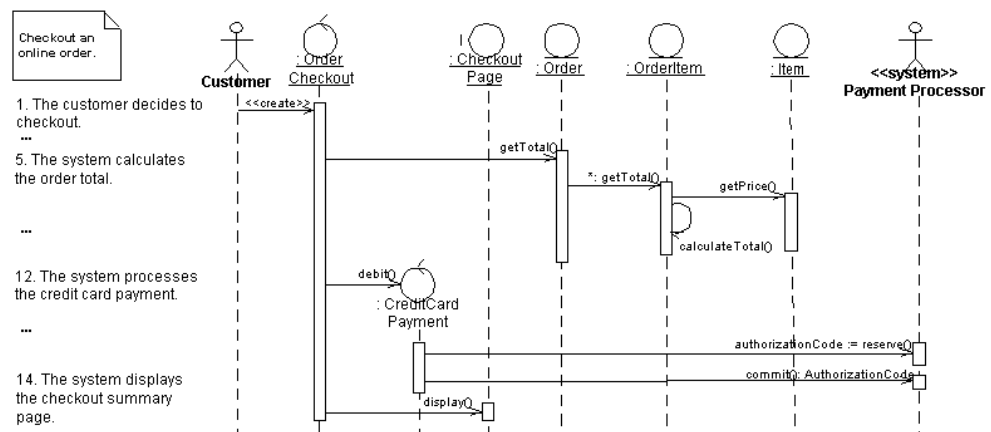
Class diagram merupakan diagram yang selalu ada di pemodelan system berorientasi objek. Class diagram menunjukkan hubungan antar class dalam system yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai satu tujuan. Kelas pada kelas diagram terdiri dari 3 bagian utama yaitu nama kelas, isi property dari kelas beserta metode yang ada pada kelas tersebut. Kelas juga memiliki jenis-jenis hubungan seperti asosiatif, dependensi, agregasi, komposisi, spesifikasi dan generalisasi. Hubungan ini digunakan untuk menggambarkan bagaimana hubungan dan interaksi yang terjadi antar kelas. Masing-masing komponen penyusun kelas memiliki hak akses seperti public, private dan protected.



Gambar 2.25 Class Diagram

4. Sequence Diagram

Sequence Diagram adalah suatu diagram yang digunakan untuk menggambarkan dan menjelaskan secara detail urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari use case. Adapun urutan proses yang dijelaskan yaitu interaksi yang terjadi antar class, operasi yang terlibat, urutan antar operasi, dan informs yang diperlukan oleh masing-masing operasi. Komponen utamanya adalah objek yang digambarkan dengan kotak segi empat atau bulat, message yang digambarkan dengan garis penuh, dan waktu yang ditunjukkan dengan progress *vertical*.



Gambar 2.26 Sequence Diagram

