BAB II
LANDASAN TEORI

2.1 Sistem informasi

Sistem informasi yaitu suatu sistem yang menyediakan informasi untuk manajemen dalam mengambil keputusan dan berfungsi untuk menjalankan operasional dalam sebuah perusahaan. Sistem tersebut merupakan kombinasi dari orang-orang, teknologi informasi dan prosedur-prosedur yang saling tergorganisasi (Binus University, 2016). Biasanya dalam suatu perusahan atau suatu badan usaha menyediakan semacam informasi yang nantinya berguna bagi manajemen.

a. Pengertian Sistem Informasi (SI) Menurut Para Ahli

Robert A. Leitch, Sistem informasi adalah suatu sistem yang berada di dalam suatu organisasi mempertemukan kebutuhan pengolahan transaksi harian, mendukung sistem operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi untuk menyediakan pihak luar secara tertentu dengan bentuk laporan-laporan yang diperlukan. Henry Lucas, Kegiatan dari prosedur-prosedur yang diorganisasikan, apabila dieksekusi akan menyediakan sekumpulan informasi untuk mendukung dalam pengambilan keputusan dan pengendalian di dalam. Sistem informasi adalah tipe khusus dari suatu sistem kerja yang dimana manusia dan mesin melakukan pekerjaan secara bersamaan dengan menggunakan sumber daya yang ada untuk memproduksi produk tertentu atau jasa bagi pelanggan (Binus University, 2016).

b. Fungsi Sistem Informasi

1. Meningkatkan aksesiblitas data yang ada secara efektif dan efisien kepada pengguna, tanpa dengan prantara dari sistem informasi.
2. Memperbaiki produktivitas aplikasi untuk pengembangan dan pemeliharaan sistem yang ada.
3. Menjamin tersedianya kualitas dan keterampilan dalam memanfaatkan sistem informasi secara kritis.

4. Mengidentifikasi kebutuhan dalam mengenai keterampilan pendukung dalam sistem informasi
5. Menetapkan investasi yang akan diarahkan pada sebuah sistem informasi
6. Mengembangkan proses perencanaan yang efektif

c. Komponen Sistem Informasi (SI)

Komponen-komponen dari sistem informasi adalah sebagai berikut :

1. Komponen *input*

   Adalah data yang masuk ke dalam sebuah sistem informasi
2. Komponen model

   Adalah kombinasi prosedur, logika dan model matematika yang nantinya digunakan untuk memproses data yang nantinya tersimpan di basis data dengan cara yang sudah di tentukan untuk menghasilkan keluaran yang diinginkan.
3. Komponen *output*

   Adalah hasil informasi yang berkualitas dan dokumentasi yang nantinya berguna untuk semua tingkatan dalam manajemen serta dalam semua pemakai sistem.
4. Komponen teknologi

   Adalah alat dalam sebuah sistem informasi, dimana teknologi digunakan dalam menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan *output* dan memantau dalam pengendalian sistem.
5. Komponen basis data

   Adalah kumpulan data yang saling terhubung satu sama lain yang tersimpan di dalam sebuah komputer dengan menggunakan *software database*.
6. Komponen kontrol

   Adalah komponen yang mengendalikan setiap gangguan terhadap sebuah sistem informasi.

2.2 *Item-Based Collaborative Filtering*

Metode *item-based collaborative filtering* digunakan dalam proses penentuan rekomendasi untuk lokasi pariwisata yang nanti sesuai dengan dengan *user*. Proses penilaian dilakukan mengunakan *rating* dari tiap lokasi yang nantinya akan di isi atau di nilai oleh *user* yang sudah berkunjung pada lokasi tersebut. Tahap awal dari metode *item-based collaborative filtering* adalah menghitung nilai kemiripan diantara tempat pariwisata yang telah dirating oleh *user*, yang dimana bentuk penilaian dari *user* sendiri biasanya berupa *rating* dalam skala tertentu (Sanjoyo, 2009). Untuk proses penghitungan nilai kemiripan antara dua tempat pariwisata, digunakan rumus *adjusted-cosine similarity* seperti dibawah ini :

$$sim(i,j) = \frac{\sum_{u\in U}(R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u\in U}(R_{u,i} - \bar{R}_u)^2}\sqrt{\sum_{u\in U}(R_{u,j} - \bar{R}_u)^2}} \quad \text{........... (2.1)}$$

Keterangan :

sim(i,j) = Nilai kemiripan antara tempat pariwisata i dan tempat pariwisata j.
u∈U = Himpunan *user* u yang merating tempat pariwisata i dan tempat pariwisata j.
R(u,i) = Rating *user* u pada tempat pariwisata i.
R(u,j) = Rating *user* u pada tempat pariwisata j.
Ru = Nilai rata-rata *rating user* u.

Tahapan selanjutnya yang paling penting dalam proses *collaborative filtering* adalah pembuatan prediksi. Setelah mendapatkan sekumpulan tempat pariwisata yang sangat mirip berdasarkan perhitungan kemiripan tersebut, dilakukan proses prediksi yang nantinya di proses untuk memperkirakan nilai *rating* dari *user* untuk suatu tempat pariwisata yang belum pernah dirating atau didatangi sebelumnya oleh *user* tersebut.

Teknik yang digunakan untuk mendapatkan nilai prediksi tersebut adalah dengan persamaan *weighted sum*, teknik ini digunakan untuk memprediksi tempat pariwisata semisal j untuk *user* u dengan menghitung jumlah dari nilai *rating* yang diberikan oleh *user* terhadap suatu tempat pariwisata yang berkorelasi dengan tempat pariwisata j. Setiap *rating* yang diberikan pada tempat pariwisata yang berkorelasi nantinya akan dikalikan dengan nilai kemiripannya (Alfian, 2009). Kemudian dibagi dengan jumlah nilai absolut kemiripan dengan seluruh tempat pariwisata yang berkorelasi. Persamaan *weighted sum* adalah sebagai berikut:

$$P_{(u,j)} = \frac{\sum_{i \in I}(R_{u.i} * S_{i.j})}{\sum_{i \in I}|S_{i.j}|} \qquad \text{....................(2.2)}$$

Keterangan :

P(u,j) = Prediksi untuk *user* u pada tempat pariwisata j.

i∈I = Himpunan tempat pariwisata yang mirip dengan tempat pariwisata j.

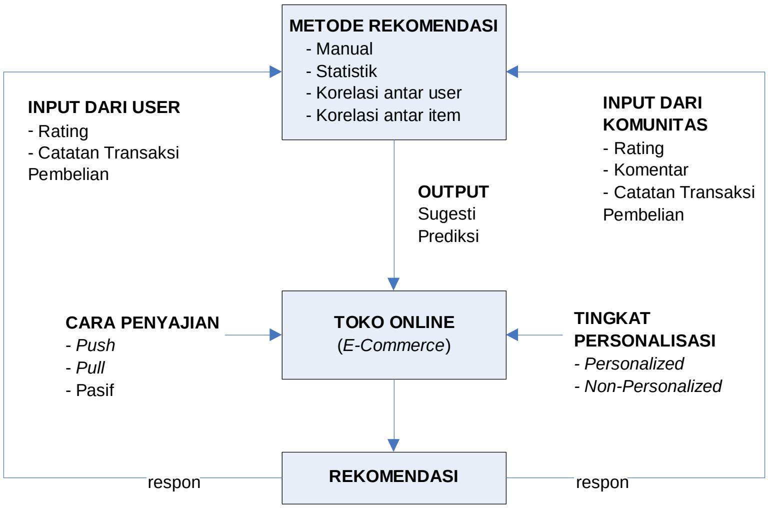
R(u,i) = Rating *user* u pada tempat pariwisata i.

S(i,j) = Nilai kemiripan antara tempat pariwisata i dan tempat pariwisata j.

Menghitung nilai kemiripan menggunakan nilai yang akan dihasilkan oleh persamaan *weighted sum* adalah berkisar antara +1.0 dengan -1.0. Tempat pariwisata diketahui jika nilai kemiripan sama dengan 0 berarti tempat pariwisata tidak berkorelasi (independen). Nilai kemiripan mendekati +1.0 berarti tempat pariwisata cenderung akan mirip antara satu dengan yang lainnya, apabila rating suatu tempat pariwisata telah diketahui maka rating tempat pariwisata yang lainnya dapat diketahui dan disimpulkan dengan probabilitas yang tinggi. Nilai kemiripan mendekati -1.0 berarti tempat pariwisata saling bertolak belakang dan dalam kasus ini juga rating suatu tempat pariwisata bisa ditentukan berdasarkan rating dari tempat pariwisata lainnya, tapi keadaannya sekarang apabila rating tempat pariwisata pertama meningkat maka rating tempat pariwisata kedua justru akan sebaliknya, yaitu menurun.

2.2.1 *Recommender System*

Recommender system (RSs) adalah salah satu dari sekian bentuk *personalized information system* yang banyak digunakan dalam pembuatan *website e-commerce* untuk menawarkan sejumlah *item* kepada *user* dan memberi informasi tambahan kepada *user* untuk dapat membantu *user* dalam memilih atau membeli *item* yang di tawarkan. Gambar 2.1 berikut ini menunjukkan taksonomi *recommender system* (Alfian, 2009).



Gambar 2.1 Taksonomi *Recommender System*

Gambar 2.1 terdapat tiga komponen utama dalam sebuah *recommender system*. Penjelasan dari Gambar 2.1 akan di jelasakan sebagai berikut:

1. Input / Output

Input yang dianalisa RSs didapatkan baik secara eksplisit maupun implisit dari *user* yang kemudian dikombinasikan dengan input dari *user-user* lain atau komunitas yang sudah dibuat. Input yang didapatkan secara eksplisit, misalnya dengan cara meminta *user* untuk memberikan rating terhadap suatu item terlebih dahulu. Implisit misalnya dari data transaksi pembelian item oleh *user* pada waktu lampau, atau bisa juga dengan memonitor item-item mana saja yang telah dilihat oleh *user* tersebut. Output yang dihasilkan oleh RSs dapat berupa sugesti atau rekomendasi (merekomendasikan sebuah *item* secara khusus) atau prediksi (bisa berupa prediksi per *item* atau beberapa *item* sekaligus dalam bentuk *list*/ daftar).

2. Metode Rekomendasi

Metode yang digunakan dalam memberikan rekomendasi dapat dilakukan dengan beberapa cara. Rekomendasi secara manual, melalui pendekatan data statistik, dengan berdasarkan korelasi antar *user* (*user-to-user*), atau dengan pendekatan berdasarkan korelasi antar *item* (*item-to-item*).

3. Desain Rekomendasi

Desain rekomendasi terkait pada 2 hal. Pertama yaitu bagaimana rekomendasi dapat disajikan dan bagaimana dengan sifat rekomendasi atau tingkat personalisasinya. Ada 3 cara untuk menyajikan rekomendasi kepada *user*, yaitu:

a. *Push* : bentuk aktif pemberian rekomendasi, seperti mengirimkan kepada *user* melalui email atau notifikasi.
b. *Pull* : rekomendasi tidak ditampilkan apabila *user* tidak meminta rekomendasi.
c. Pasif : menampilkan item lain yang berhubungan antara *item* yang sedang dilihat atau diakses *user* pada saat itu.

Tingkat personalisasi rekomendasi yang diberikan kepada *user* ada 2 macam, yaitu:

a. Personalized : rekomendasi yang diberikan kepada *user* tidak sama antara *user* yang satu dengan *user* yang lain, tergantung pada masing-masing profil *user* dari *user*.
b. Non-personalized : bentuk rekomendasi ini tidak melihat dari profil masing-masing *user*, dengan kata lain rekomendasi tersebut bersifat umum sehingga dapat diberikan atau dilihat semua pengunjung atau *visitor*.

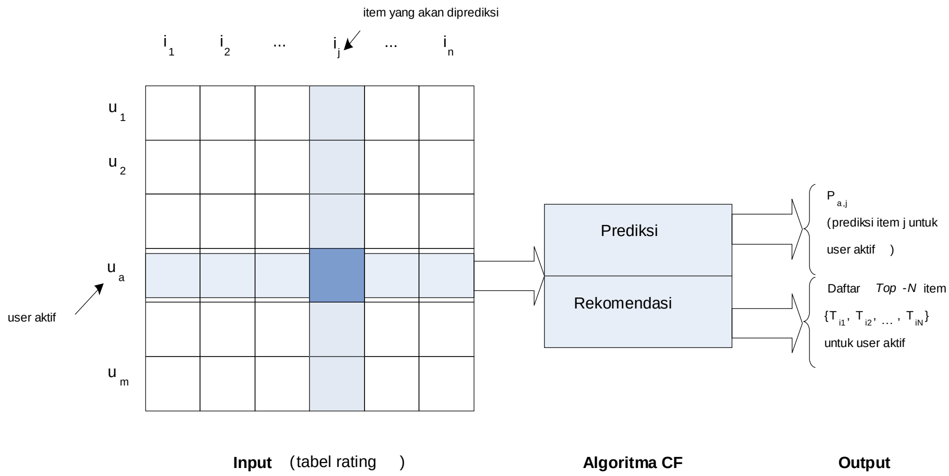
Teknik data *mining* seperti *association rule* dan *market basket analysis*, *nearest neighbor*, hingga *clustering* telah digunakan dalam membangun sebuah RSs. Namun, secara garis besar terdapat teknik yang digunakan dalam RSs ada 2 macam, yakni *content-based filtering* dan *collaborative filtering* . *Content-based filtering* bekerja dengan mencari *item* yang mempunyai korelasi dengan *item* yang disukai *user* berdasarkan *content* atau informasi dari tiap-tiap *item*. *Collaborative filtering* (CF) merekomendasikan *item* kepada seorang *user* berdasarkan *rating* yang diberikannya terhadap suatu *item*. CF lebih banyak digunakan karena dalam beberapa *web domain* seperti musik dan film, sulit dilakukan ekstraksi *content* tiap item yang merupakan langkah utama dari *content-based filtering* .

Penelitian tentang RSs berbasis CF telah banyak dilakukan penelitian. RSs berbasis CF yang sudah ada pertama kali menggunakan metode *nearest-neighbor*. Metode ini dikenal pula dengan nama *memory-based* atau *user-based* karena mencari *user* yang paling mirip dengan *user* target dalam hal merating *item,* lalu merekomendasikan *item-item* yang disukai *user* terdekat terhadap *user* target. Kelemahan metode *memory based* adalah masalah skalabilitas seiring dengan makin banyaknya jumlah *user* dan jumlah *item* akan mempengaruhi rekomendasi. Metode *model-based* atau *item-based* muncul sebagai solusi metode *user-based* CF. Metode *userbased* yang menghitung kemiripan antar *user* secara *online,* metode *item-based* melakukan pembuatan dari model korelasi antar item terlebih dahulu secara *offline* untuk kemudian digunakan pehitungan dalam membuat rekomendasi secara *online* sehingga rekomendasi dapat diberikan secara *real-time* secara langsung.

2.2.2 *Collaborative Filtering*

Collaborative filtering (CF) adalah suatu metode yang sering digunakan dalam pembuatan prediksi otomatis untuk memperkirakan ketertarikan atau selera seorang *user* terhadap suatu item dengan cara mengumpulkan informasi dari *user-user* yang lain yang nantinya direpresentasikan dalam bentuk nilai *rating*. Secara umum, ada 2 proses yang dilakukan dalam CF, yaitu:

1. Mencari *user* lain yang dimana *user* lain tersebut mempunyai kemiripan pola rating dengan *user* target (*user* yang akan diberikan prediksi).
2. Menggunakan nilai rating dari *user* lain yang didapat dari langkah 1 di atas untuk menghitung prediksi bagi *user* yang sedang aktif.



Gambar 2.2 Proses *collaborative filtering*

Gambar 2.2 adalah gambaran proses *collaborative filtering*. Algoritma CF menghasilkan 2 bentuk *output* yaitu (Alfian, 2009) :

a. Prediksi ($P_{a,j}$) adalah suatu nilai yang menyatakan sebuah prediksi dari besarnya dari *rating item* j yang memungkinkan didapat dari *user* aktif, dimana *item* j belum pernah dirating oleh *user* aktif tersebut.

b. Rekomendasi adalah sebuah daftar yang berisi N *item* yang mempunyai kemungkinan terbesar untuk disukai oleh *user* aktif, dimana *item-item* tersebut belum pernah dirating oleh *user* aktif. Bentuk ini disebut juga sebagai rekomendasi Top-N.

Perbedaan antara rekomendasi (dengan seorang *user*, disarankan item-item yang nantinya mungkin menarik oleh *user* tersebut) dan prediksi (dengan sebuah *item*, dilakukan prediksi berapa *rating* yang mungkin nantinya diberikan *user* pada *item* tersebut). Pada dasarnya untuk memberi rekomendasi kepada *user* tersebut diperlukan perhitungan prediksi pada tingkat ketertarikan *user* terhadap *item-item* tersebut. Algoritma CF yang ada saat ini dapat dikelompokkan menjadi 2 kategori (Schafer, dkk, 2007), yaitu:

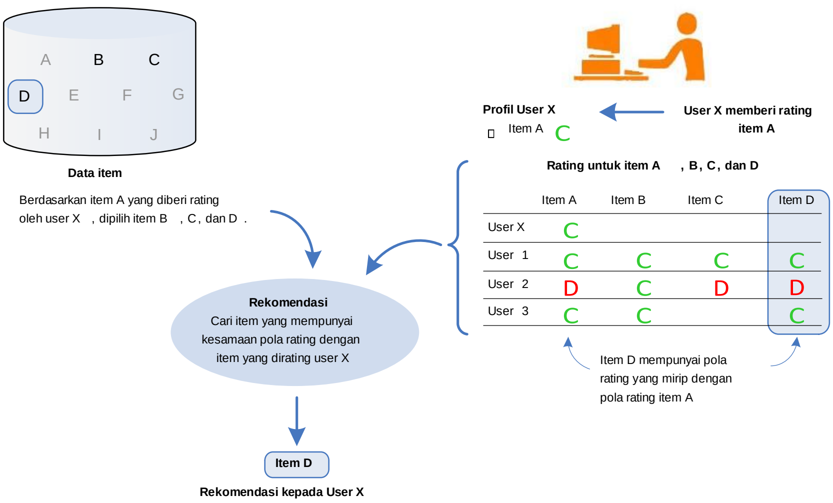
1. *Memory-based* CF

Algoritma *memory-based* mempergunakan seluruh data *rating* yang ada dalam membuat prediksi. Metode ini juga memakai teknik statistik dalam mencari sekumpulan *user*, disebut dengan *nearest-neighbor*, yang digunakan untuk mencari kemiripan dengan *user* target. Kemiripan antar *user* tersebut, dibuat prediksi dan rekomendasi *Top-N* untuk *user* target. Metode ini sering disebut juga dengan *nearest-neighbor* atau *userbased collaborative filtering*.

2. *Model-based CF*

Memory-based yang mencari hubungan antar *user*, *model-based* mencari hubungan antar *item* berdasarkan tabel *rating* untuk pembuatan rekomendasi. Sehingga metode ini sering disebut juga dengan *item-based*. Sebelum menghasilkan rekomendasi, algoritma *model-based* membuat sebuah model korelasi antar *item* terlebih dahulu untuk mengetahui hubungan antar *item* dengan menghitung nilai *rating* yang didapat.

Proses pembuatan model dapat dilakukan dengan cara menggunakan berbagai teknik, seperti *association rule*, *classification*, atau *clustering*. Pembuatan model korelasi antar *item* dapat dilakukan secara *offline*. Model yang sudah dibentuk, maka perhitungan prediksi atau pemberian rekomendasi dapat dilakukan secara *online* atau *real-time*. Skema *model-based* atau *item-based* dapat dilihat pada Gambar 2.3.



Gambar 2.3 Skema *model-based/ item-based* CF

2.3 *Database mysql*

MySQL adalah sistem manajemen *database* SQL yang bersifat *Open Source* dan paling digunakan dan populer saat ini. Sistem *Database* MySQL mendukung beberapa fitur seperti *multithreaded*, *multi-user*, dan SQL *database* manajemen sistem (DBMS). *Database* ini juga dapat dibuat untuk keperluan sistem *database* karena memiliki performa yang cepat, handal dan mudah digunakan dalam pengoperasianya (MySQL, 2017).

Ulf Micheal Widenius adalah penemu awal versi pertama MySQL yang kemudian pengembangannya dilakukan oleh perusahaan MySQL AB. MySQL AB yang merupakan sebuah perusahaan komersial yang didirikan oleh para pengembang MySQL. MySQL sudah digunakan lebih dari 11 millar instalasi sampai saat ini. Berikut adalah beberapa kelebihan MySQL sebagai database server yaitu:

1. Source MySQL dapat diperoleh dengan mudah dan gratis tanpa menggunakan lisensi.
2. Pengaksesan *database* dapat dilakukan dengan mudah dan cepat.
3. MySQL merupakan program yang *multithreaded*, sehingga dapat dipasang pada *server* yang memiliki *multiCPU* yang memiliki 2 atau lebih prosesor.
4. Didukung program umum seperti C, C++, *Java, Perl,* PHP, *Python*, dsb.
5. Bekerja pada berbagai *platform operating sistem*. (tersedia berbagai versi untuk berbagai sistem operasi).
6. Memiliki jenis kolom yang cukup banyak sehingga fleksibel dan memudahkan konfigurasi sistem penyimpanan *database*.
7. Memiliki sistem *security* yang cukup baik dengan verifikasi *host*.
8. Mendukung ODBC untuk aplikasi yang berjalan pada sistem operasi *windows*.
9. Mendukung *record* yang memiliki kolom dengan panjang tetap atau panjang bervariasi.

2.4 Pengertian *Android*

Android adalah sistem operasi untuk perangkat mobile berbasis *Linux* yang mencakup sistem operasi, *middleware* dan juga aplikasi. *Android* menyediakan platform terbuka bagi para pengembangnya untuk menciptakan aplikasi sesuai keinginan mereka sendiri untuk digunakan. *Android* umum digunakan di *smartphone* dan juga *tablet* PC.

2.4.1 Karakteristik *Android*

a. Terbuka

Android dibangun dari sebuah aplikasi yang dapat memanggil salah satu fungsi inti ponsel seperti membuat panggilan, mengirim pesan teks, menggunakan kamera, dan lain-lain. *Android* menggunakan sebuah mesin *virtual* yang dirancang khusus untuk mengoptimalkan sumber daya dari memori dan perangkat keras yang terdapat di dalam perangkat tersebut. *Android* juga merupakan sistem operasi *open source*, dapat secara bebas melakukan perkembangan untuk memasukkan teknologi baru yang lebih maju pada saat teknologi tersebut muncul. *Platform* ini akan terus berkembang untuk membangun aplikasi *mobile* yang inovatif seiring berjalannya waktu dimasa mendatang.

b. Semua aplikasi dibuat sama

Android tidak memberikan perbedaan terhadap aplikasi dari telepon dan aplikasi pihak ketiga *(third-party application)*. Semua aplikasi dapat dibangun atau dibuat untuk memiliki akses yang sama terhadap kemampuan dalam sebuah telepon yang menyediakan layanan dan aplikasi yang luas terhadap para pengguna telepon.

c. Memecahkan hambatan pada aplikasi

Android memecah hambatan untuk pembuatan aplikasi yang bersifat baru dan inovatif. Pengembang dapat menggabungkan informasi yang diperoleh dari *website* dengan data yang terdapat pada ponsel seseorang seperti kontak, kalender, atau lokasi geografis dari pengguna.

d. Pengembangan aplikasi yang cepat dan mudah

Android menyediakan akses yang sangat luas kepada pengguna untuk menggunakan *library* yang diperlukan serta *tools* - *tools* yang dapat digunakan untuk membangun aplikasi. *Android* memiliki sekumpulan *tools* yang dapat digunakan sehingga dapat membantu para pengembang dalam meningkatkan produktivitas saat membangun sebuah aplikasi (Android, 2017)

2.4.2 Sejarah *Android*

Google bekerjasama dengan perusahaan *Android Inc.,* perusahaan yang berada di Palo Alto, California Amerika Serikat. Para pendiri *Android Inc*. pekerja yang berada di *Google*, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Banyak yang menganggap fungsi *Android Inc* hanyalah sebagai sebuah perangkat lunak pada telepon seluler. Saat itu muncul rumor bahwa Perusahaan *Google* hendak memasuki pasar telepon seluler. Perusahaan *Google* sendiri membentuk tim yang dipimpin Rubin yang di beri tugas untuk mengembangkan program perangkat seluler yang didukung oleh *kernel linux*. Hal tersebut menunjukkan indikasi bahwa perusahaan *Google* sedang bersiap menghadapi persaingan dalam pasar telepon seluler atau telepon genggam.

Pada 9 Desember 2008, diumumkanlah anggota baru yang ikut bergabung dalam program kerja *Android* yaitu ARM *Holdings*, *Atheros Communications*, diproduksi oleh *Asustek Computer Inc*, *Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc.* Pembentukan *Open Handset Alliance*, OHA mengumumkan produk perdana mereka, *Android,* perangkat bergerak (*mobile*) yang merupakan modifikasi *kernel linux* 2.6. *Android* dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru.

Telepon pertama yang memakai sistem operasi *android* adalah HTC *Dream*, yang dirilis pada 22 Oktober 2008. Penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan *android* (Android, 2017).

2.4.3 Arsitektur *Android*

Arsitektur *android* terdiri atas *Applications* dan *Widgets*, *Applications Frameworks, Libraries, Android Run Time,* dan *Linux Kernel*. Bagian arsitektur *android* akan dijelaskan satu persatu sebagai berikut:

a. *Application*s dan *Widgets*

Application dan *widgets* merupakan *layer* dimana berhubungan dengan aplikasi saja yang kebanyakan *download* aplikasi dan kemudian dilakukan instalasi untuk menjalankan aplikasi tersebut. *Layer* terdapat aplikasi inti termasuk *client email*, program sms, kalender, peta, *browser*, kontak, dan lain-lain. Aplikasi ditulis atau dibuat menggunakan bahasa pemrograman *java.*

b. *Application*s *Framework*

*Application*s *Framework* merupakan *Open Development Platform* yang ditawarkan *android* untuk dapat dikembangkan dalam membangun aplikasi. Pengembang memiliki akses penuh menuju API *Framework* seperti yang dilakukan oleh aplikasi inti. Komponen-komponen yang termasuk dalam *Application*s *Framework* akan dijelasakan sebagai berikut :

1. *Views* yang kaya dan *extensible* yang dapat digunakan untuk membangun aplikasi seperti *list, grids*, kotak teks, tombol dan bahkan sebuah *embeddable web.*
2. *Content Provider* yang memungkinkan aplikasi untuk mengakses data dalam daftar kontak telepon atau membagi data tersebut.
3. *Resource Manager* yang menyediakan akses ke kode yang berada di luar sumbernya. seperti kata, gambar, dan tata letak *file* program.
4. *Notification Manager* yang memungkinan aplikasi menampilkan *alert* yang biasa dikustomisasi di dalam status bar atau notifikasi.
5. *Activity Manager* yang Pengelolaan siklus tampilan dari aplikasi dan menyediakan navigasi umum berupa *backstack.*

c. *Libraries*

Libraries merupakan layer *dimana* terdapat fitur-fitur *android*. *Android* menyertakan *libraries* C / C+ + yang nantinya digunakan oleh berbagai komponen dari sistem *android*. Kemampuan itu disediakan kepada *developer* aplikasi melalui *framework* dari aplikasi *android* yang telah dibuat sebelumnya.

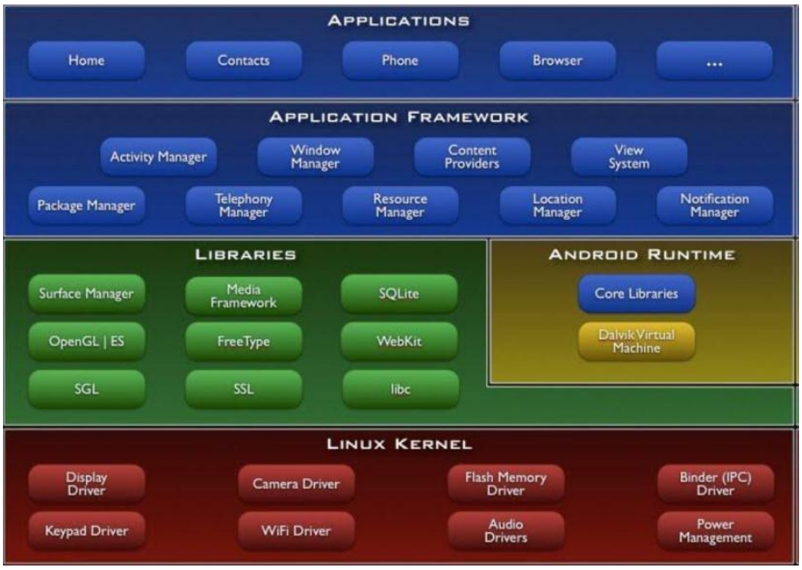
1. Sistem *C-Library* merupakan variasi dari implementasi BSD yang berasal dari pelaksanaan sistem standar *C-Library (libe)*, yang sesuai untuk dijalankan pada perangkat *embedded* berbasis *linux*.
2. Media *Library* merupakan *library* yang mendukung pemutaran rekaman seperti *file* yang berformat audio dan video, serta *file* gambar.
3. *Graphic Library* termasuk didalamnya SGL dan *OpenGL* untuk tampilan 2D dan 3D.
4. *Surface Manager* merupakan *library* untuk pengelolaan akses ke subsistem pada layar.
5. *LibWebcore* merupakan mesin *web* modern yang baik untuk *browser* yang di gunakan dalam *android.*
6. *FreeType* mendukung *bitmap* dan *vector font rendering* untuk tampilan huruf.
7. *SQLite* merupakan mesin *database* yang kuat dan ringan, dan dapat menjadi penghubung untuk semua aplikasi *android.*

d. *Android Run Time*

Android Run Time merupakan *layer* yang digunakan untuk membuat aplikasi *android* untuk dapat dijalankan, dimana dalam prosesnya menggunakan implementasi operating sistem *linux*. *Android* terdiri dari satu *set core libraries* yang menyediakan beberapa fungsi yang sama dengan terdapat pada *core* yang *libraries* yang mengunakan bahasa pemrograman *java*.

e. *Linux Kernel*

Linux Kernel merupakan *layer* inti dari sebuah sistem operasi *android*. Berisi *file-file* sistem yang mengatur sistem *processing*, *memory*, *resources*, *drivers*, dan sistem-sistem operasi *android* lainnya. *Android* bukanlah *linux*, akan tetapi *android* dibangun diatas *linux kernel* yaitu versi 2.6 sehingga kehandalannya dapat dipercaya. Inti sistem *service linux* yang digunakan seperti keamanan, manajemen memori, proses manajemen, *network* dan *driver* model. *Linux* menyediakan *driver* layar, kamera, *keypad*, *wifi*, *flash memory*, audio dan IPC (*Interprocess Communication*) untuk mengatur keamanan. *Kernel* juga bertindak sebagai lapisan abstrak antara *hardware* dan *software*, untuk lebih jelasnya lihat gambar arsitektur android berikut (Android, 2017).



Gambar 2.5 Arsitektur *Android*

2.4.4 Versi *Android*

Android memiliki sejumlah pembaharuan semenjak rilis aslinya. Pembaharuan ini dilakukan untuk memperbaiki *bug* dan menambah fitur-fitur yang baru. Berikut merupakan versi-versi yang dimiliki Android dari dulu sampai saat ini:

Nama	Versi	Peluncuran
Cupcake	1.5	27 April 2009
Donut	1.6	15 September 2009
Eclair	2.0 – 2.1	26 Oktober 2009
Froyo	2.2 – 2.2.3	20 Mei 2010
Gingerbread	2.3 – 2.3.7	6 Desember 2010
Honeycomb	3.0–3.2.6	22 Pebruari 2011
Ice Cream Sandwich	4.0 – 4.0.4	18 Oktober 2011
Jelly Bean	4.1 – 4.3.1	9 Juli 2012
KitKat	4.4 – 4.4.4	31 Oktobe 2013
Lollipop	5.0 – 5.1.1	12 November 2014
Marshmallow	6.0 – 6.0.1	5 Oktober 2015
Nougat	7.0	Agustus / September 2016

Gambar 2.6 Tabel Versi *Android*

2.5 *Android* Studio

Android Studio adalah sebuah IDE yang digunakan untuk *Android Development* yang diperkenalkan *google* pada acara *Google* I/O 2013. *Android* Studio merupakan pengembangkan dari *Eclipse* IDE, dan dibuat berdasarkan IDE *java* populer, yaitu IntelliJ IDEA. Android Studio juga merupakan IDE resmi untuk pengembangan aplikasi pada *android*.

Pengembang *Eclipse Android* Studio mempunyai banyak fitur-fitur baru dibandingkan dengan e*clipse* IDE. Hal ini berbeda dengan e*clipse* yang menggunakan *Ant*, *android* studio menggunakan g*radle* sebagai *build environment*. Fitur dari *android* studio adalah sebagai berikut:

a. Menggunakan *gradle-based build system* yang sangat fleksibel.
b. Bisa mem-*build multiple* APK secara bersamaan.
c. *Template support* untuk *google services* dalam berbagai macam tipe perangkat.
d. *Layout editor* yang lebih bagus dibandungkan *eclipse* IDE.
e. *Built-in support* untuk *google cloud platform*, sehingga mudah untuk integrasi dengan *google cloud messaging* dan a*pp engine*.
f. *Import library* langsung dari *maven repository*.