BAB II
TINJAUAN PUSTAKA

Bagian ini berisi tentang beberapa teori-teori yang mendukung dalam proses analisis dan implementasi berdasarkan masalah yang diangkat dalam pembangunan Aplikasi Android *Smart Indoor Navigation* Di Gedung Perbelanjaan Untuk Petunjuk Rute Dan Informasi Promo Menggunakan Wifi *Tri-lateration* Dan *Twitter* API.

2.1 Sejarah Istana Plaza

Istana Plaza Bandung adalah sebuah pusat perbelanjaan yang berada di Bandung. Istana Plaza Bandung didirikan pada tahun 2001, dengan lokasi yang strategis berada di Jalan Pasirkaliki No.121-123, Bandung dekat dengan bandara Husein Sastranegara dan Stasiun KA Bandung. Istana Plaza terdiri dari 5 lantai dengan penyewa/*tenant* yang sudah terkenal sebagai perusahaan besar baik skala nasional maupun internasional, Pada tahun 2001, dibangun sebuah arena ice skating di lantai 3, namun pada Februari 2009 arena ice skating ditutup dan dialihfungsikan menjadi tempat makan. Istana Plaza merupakan family mall yang berkonsep untuk menyediakan seluruh kebutuhan keluarga dalam satu tempat.

2.1.1 Peta

Istana Plaza memiliki 5 lantai yang terdiri dari lantai bawah, lantai dasar, lantai 1, lantai 2 dan lantai 3, untuk peta Istana Plaza terlampir di Lampiran 1.

2.1.2 Tenant

Tenant dalam Bahasa Indonesia adalah penyewa. Penyewa disini merupakan perusahaan yang menyewa ruangan untuk dijadikan sebuah toko didalam gedung Istana Plaza, *tenant* dibedakan menjadi 4 kategori sebagai berikut:

1. *Store*
2. F&B (*Food and Beverage*)
3. *Entertainment*
4. *Fashion*

2.1.3 Aminities

Aminities dalam Bahasa Indonesia adalaha fasilitas. Istana Plaza menyediakan berbagai fasilitas yang dapat digunakan oleh pengunjung secara gratis seperti :

1. *Disable Facility*
2. *Mother's care*
3. Musola
4. *Free wifi*

2.2 Wireless Fidelity (WiFi)

Wireless Fidelity (WiFi) adalah sebuah teknologi jaringan nirkabel dengan parangkat berstandar IEEE 802.11 menghubungkan semua perangkat yang berada dalam jangkauannya ke dalam sebuah jaringan lokal sampai jaringan internet. Perbedaan dari standar – standar tersebut dijelaskan pada Tabel 2.1[1].

Tabel 2. 1 Perbedaan standar WiFi

Standar	Frekuensi	Kecepatan	Jangkauan
802.11	2.4GHz	1-2 Mb/s	20m
802.11a	5GHz	6-54 Mb/s	35m
802.11b	2.4GHz	1-11 Mb/s	35m
802.11g	2.4GHz	6-54 Mb/s	38m
802.11n	2.4GHz	7.2-72.2 Mb/s	70m
802.11n	5GHz	15-150 Mb/s	70m

2.3 Wireless Access Point (WAP)

Wireless access point (WAP) adalah sebuah perangkat keras jaringan yang berisi sebuah *transceiver* dan antenna yang memungkinkan perangkat *wireless* lain (seperti laptop, ponsel) untuk terhubung ke jaringan kabel menggunakan WiFi, bluetooth atau perangkat standar lainnya[2].

2.4 Received Signal Strength Indicator (RSSI)

Received Signal Strength Indicator (RSSI) adalah pengukuran kekuatan sinyal radio yang di terima. Setiap WAP memancarkan sinyal di daerah cakupannya. Perangkat nirkabel mempunyai kemampuan untuk mengukur kekuatan dari sinyal tersebut. Kekuatan sinyal ini dikonversi ke dalam sebuah nilai yang disebut dengan Received Signal Strength Indicator (RSSI). Perangkat nirkabel dapat mendeteksi RSSI dan identitas dari beberapa WAP, *range* nilai RSSI berkisaran antara -40dBm sampai -113dBm bergantung pada jarak perangkat nirkabel meneriam sinyal dari WAP[3].

2.5 WiFi Positioning System (WPS)

WiFi Positioning System adalah sistem penentuan posisi berdasarkan sinyal yang diterima oleh perangkat nirkabel. Selain WiFi, terdapat teknologi lain yang dapat digunakan untuk sistem penentuan posisi. Beberapa teknologi tersebut memiliki kelebihan dan kekurangan masing – masing yang dijelaskan dalam Tabel 2.2[3].

Tabel 2. 2 Kelebihan dan Kekurangan Teknologi WPS

No	Teknologi	Kelebihan	Kekurangan
1	*Global Positioning System* (GPS)	• Akurasi menengah hingga tinggi jika di luar ruangan. • Ketersediaan tinggi.	• Akurasi rendah hingga minimal jika di dalam ruangan.
2	A-GPS	• Akurasi menengah jika di luar ruangan.	• Akurasi minimal jika di dalam ruangan.
3	Pseudolite GPS	• Akurasi tinggi untuk di dalam dan luar ruangan.	• Perangkat sangat mahal.
4	*Cell Tower*	• Cakupan luas.	• Sangat tidak akurat untuk di dalam maupun di luar ruangan.

5	WiFi	• Sinyal mudah dibaca walaupun antar. ruangan • Cakupan cukup luas. • Implementasi dengan biaya minimal.	• Daya sinyal dapat berubah.
6	*Bluetooth*	• Daya listrik rendah. • Biaya perawatan rendah.	• Cakupan cukup hingga rendah • Biaya implementasi tinggi.
7	*Infrared*	• Akurasi cukup tinggi	• Biaya implementasi tinggi • Cahaya matahari dapat mempengaruhi hasil • Cakupan rendah
8	*Ultra Wide Band* (UWB)	• Akurasi tinggi • Bandwidth lebar	• Biaya implementasi tinggi • Tidak umum digunakan

2.5.1 WiFi Fingerprinting

Teknik ini menggunakan RSSI sebagai sinyal yang diterima. Teknik ini menerima beberapa RSSI dalam sebuah koordinat lalu menyimpan sebagai *Reference Point* (RP). Tiap RP memiliki karakteristik RSSI yang unik yang hanya dimiliki oleh masing-masing RP. Hal ini bias dianalogikan seperti sidik jari (*fingerprint*) manusia yang memiliki keunikan tersendiri yang membuatnya berbeda dengan manusia lainnya[4]. Untuk mengimplementasikan teknik ini terdapat 2 fase yaitu :

1. Fase *offline*

   Fase ini adalah fase pembentukan RP. RP dibentuk dengan menyimpan data RSSI di koordinat-koordinat yang telah ditentukan sebelumnya. Setelah semua RP terbentuk, semua RP tersebut disimpan dalam sebuah database.

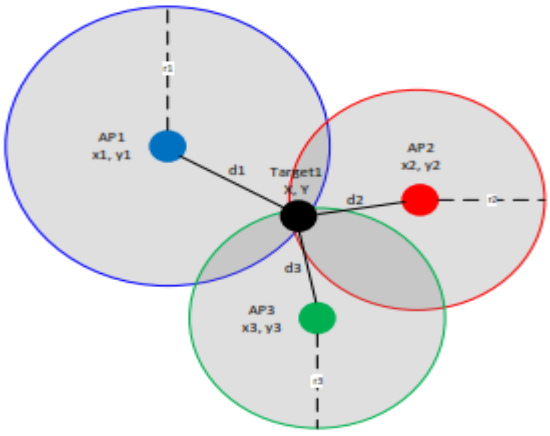
2. Fase *online*

Fase ini adalah fase pembacaan posisi pengguna saat itu lalu dibandingkan dengan RP. Jika posisi pengguna berada dekat dengan salah satu RP, dengan menggunakan algoritma pendekatan, maka posisi pengguna tersebut ditampilkan menjadi posisi RP.

Teknik ini mempunyai akurasi tinggi namun butuh waktu yang lama untuk diimpementasikan dan dalam kurun waktu tertentu database RP harus di kaji ulang agar akurasi tidak menurun. Dalam kondisi real time, teknik ini tidak dapat menampilkan posisi pengguna secara real time karena teknik ini menggunakan RP untuk menentukan posisi. Sedangkan RP hanya ada dalam koordinat tertentu, sehingga dalam koordinat lain posisi pengguna dijadikan koordinat RP tersebut, karena kondisi tersebut maka teknik ini kurang baik jika diimplementasikan dalam kondisi real time.

2.5.2 WiFi Trilateration

WiFi Trilateration adalah teknik penentuan posisi dari 3 lokasi AP sebagai referensi. *Trilateration* menggunakan jarak antar lokasi AP untuk menentukan posisi dari perangkat yang menerima sinyal AP. Dalam teknik ini, jarak antar lokasi AP dengan posisi yang ingin diketahui dapat dianalogikan sebagai radius dari beberapa lingkaran dengan pusat lingkarannya adalah lokasi dari AP. Posisi yang ingin diketahui adalah perpotongan dari 3 radius lingkaran tersebut seperti dalam gambar 2.1 .



Gambar 2. 1 *Trilateration Model*

Dalam (2.1), AP1, AP2, dan AP3 telah diketahui koordinatnya. Target1 dapat diketahui koordinatnya menggunakan koordinat referensi (AP1, AP2, dan AP3) dan jarak (d1, d2, dan d3) antara target dengan koordinat referensi. Dengan menggunakan teorema pythagoras pada Persamaan (2.1)[5] :

$$\left.\begin{aligned} {d_1}^2 &= (x_1 - x)^2 + (y_1 - y)^2 \quad (a) \\ {d_2}^2 &= (x_2 - x)^2 + (y_2 - y)^2 \quad (b) \\ {d_3}^2 &= (x_3 - x)^2 + (y_3 - y)^2 \quad (c) \end{aligned}\right\} \qquad \textbf{(2.1)}$$

Dari Persamaan (2.1) , dapat dibuat persamaan baru untuk x dan y target. Persamaan untuk x dan y target ditulis dalam Persamaan (2.2) [5].

$$\left.\begin{aligned} x &= \frac{AY_{32} + BY_{13} + CY_{21}}{2(x_1Y_{32} + x_2Y_{13} + x_3Y_{21})} \quad (a) \\ y &= \frac{AX_{32} + BX_{13} + CX_{21}}{2(y_1X_{32} + y_2X_{13} + y_3X_{21})} \quad (b) \end{aligned}\right\} \qquad \textbf{(2.2)}$$

Dimana

$$\left.\begin{aligned} A &= {x_1}^2 + {y_1}^2 - {d_1}^2 \quad (a) \\ B &= {x_2}^2 + {y_2}^2 - {d_2}^2 \quad (b) \\ C &= {x_3}^2 + {y_3}^2 - {d_3}^2 \quad (c) \end{aligned}\right\} \qquad \textbf{(2.3)}$$

Dan

$$\left.\begin{aligned} X_{32} &= (x_3 - x_2) \quad (a) \\ X_{13} &= (x_1 - x_3) \quad (b) \\ X_{21} &= (x_2 - x_1) \quad (c) \\ Y_{32} &= (y_3 - y_2) \quad (d) \\ Y_{13} &= (y_1 - y_3) \quad (e) \\ Y_{21} &= (y_2 - y_1) \quad (f) \end{aligned}\right\} \qquad \textbf{(2.4)}$$

Maka telah didapat koordinat (x, y) untuk target. Koordinat target selalu diperbarui selama target berada dalam jangkauan 3 AP. Maka dari itu teknik ini dapat diimplementasikan dalam keadaan real time. Untuk penentuan jarak antara target dengan koordinat referensi dapat dilakukan dengan 3 metode penerimaan sinyal yaitu TOA, TDOA, dan RSSI. TOA dan TDOA menggunakan waktu propagasi sebagai parameter jarak. Metode TOA mengukur waktu kedatangan

sinyal dari *transmitter* ke *receiver*. Dari parameter waktu propagasi dapat dihasilkan jarak antara *transmitter* dan *receiver*. Semua transmitter harus bersinkronisasi untuk mendapatkan hasil jarak yang baik[5]. Metode TDOA adalah versi yang telah ditingkatkan dari TOA untuk menghindari kesulitan sinkronisasi. Untuk mengimplementasikan TDOA, sebuah *transmitter* mengirim 2 sinyal dengan perbedaan kecepatan propagasi. Saat kedua sinyal diterima di *receiver,* perbedaan waktu kedatangan dari kedua sinyal dapat dihitung untuk menentukan jarak antara *transmitter* dan *receiver*[5]. RSSI menggunakan daya sinyal sebagai parameter jarak. Metode ini didasari oleh model sinyal propagasi untuk mengukur jarak antara *transmitter* dengan *receiver* yang dapat secara akurat mengukur daya sinyal. RSSI dipilih sebagai metode yang digunakan karena metodenya yang lebih sederhana dari metode lain dan sisi receiver yang berperan lebih besar dalam tingkat akurasi karena dalam penelitian ini sisi transmitter tidak dapat diubah pengaturannya.

2.6 Indoor Propagation Models

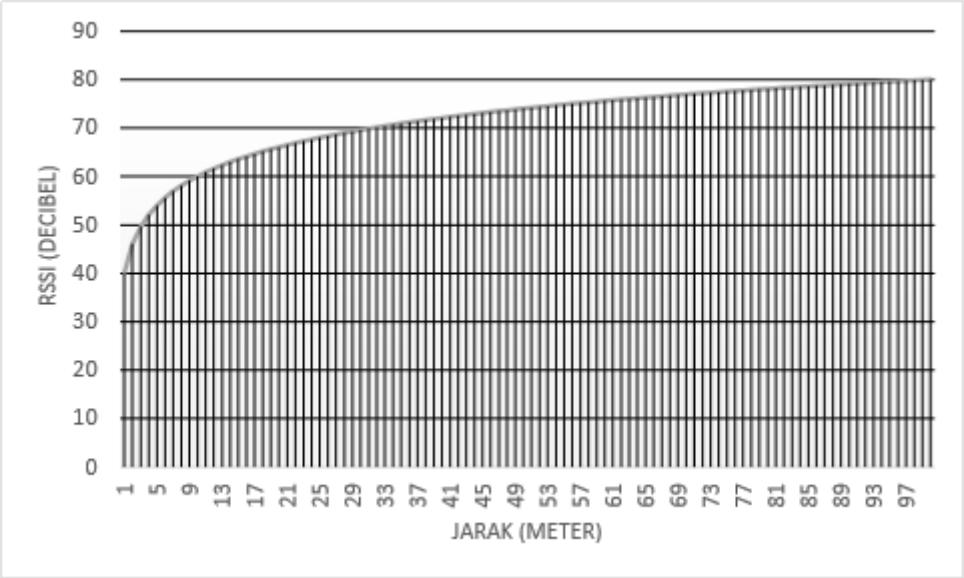
Indoor propagation model dibutuhkan untuk memodelkan kondisi saat pengujian agar dapat mengukur jarak transmitter dengan receiver secara akurat. Model propagasi membutuhkan nilai daya sinyal (RSSI) agar bisa dikonversi ke jarak. RSSI dalam ruangan dipengaruhi oleh dinding, manusia, perabotan dan lainnya seperti fenomena multipath. Oleh karena itu, dibutuhkan model propagasi untuk mengukur jarak secara akurat sesuai dengan lingkungan yang ada.

1. *Free Space Path Loss Model*

Dalam model ini, daya sinyal terima berbanding terbalik dengan jarak kuadrat dari transmitter ke receiver. Nilai dari path loss direpresentasikan dalam decibel (dB). Hubungan daya terima $P\,r$ dan jarak r digambarkan dalam Persamaan 2.5 [x].

$$P_r \propto \frac{1}{r^2} \qquad (2.5)$$

Gambar *Free Space Path Loss Model* seperti pada gambar 2.2.



Gambar 2. 2 *Free Space Path Loss Model*

Model ini adalah model path loss yang sederhana dengan komputasi yang sederhana [x]. Model ini dirumuskan dalam persamaan 2.6.

$$FSPL_{dB} = 20log_{10}(d) + 20log_{10}(f) - 147.55 \quad (2.6)$$

Persamaan 2.6 merepresentasikan jarak dalam kilometer dan frekuensi dalam gigahertz, untuk merepresentasikan jarak dalam meter dan frekuensi dalam megahertz, persamaan 2.6 ditulis kembali pada persamaan 2.7.

$$FSPL_{dB} = 20log_{10}(d) + 20log_{10}(f) - 27.55 \quad (2.7)$$

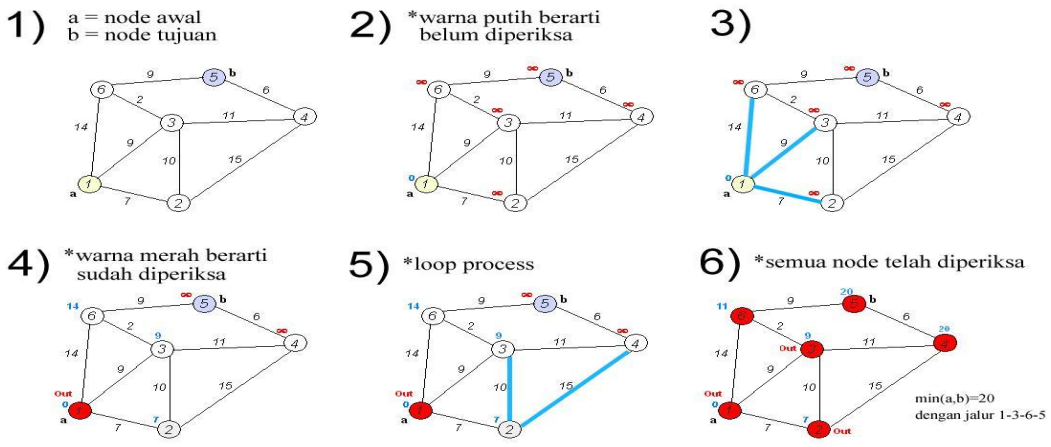
Karena yang dicari dalam topik ini adalah jarak, maka persamaan 2.7 diubah menjadi persamaan 2.8.

$$d = 10^{\frac{(FSPL_{dB} - 20log_{10}(f) + 27.55)}{20}} \quad (2.8)$$

2.7 Algoritma Djikstra

Ada berbagai macam algoritma untuk menentukan rute terpendek. Salah satu algoritma yang paling populer adalah algoritma Djikstra yang banyak digunakan dalam routing paket data pada jaringan komputer. Salah satu keunggulan algoritma ini adalah kebutuhan waktu dan ruangnya yang relatif kecil sehingga dapat diimplementasikan pada perangkat yang memiliki spesifikasi hardware yang tidak terlalu tinggi.

Proses pengerjaan algoritma Djikstra untuk menentukan rute terpendek adalah sebagai berikut gambar 2.3



Gambar 2. 3 Ilustrasi algoritma Djikstra

1. Inisiasi nilai seluruh node, nilai nol untuk node awal dan nilai tak hingga untuk node lainnya.
2. Tandai seluruh node sebagai "*unvisited node*" atau "belum dikunjungi" dan tandai node awal sebagai node saat ini (current node).
3. Perhitungkan nilai sementara seluruh node tetangga (node yang terhubung langsung dengan current node melalui sebuah edge) yang berupa "*unvisited node*". Nilai sementara adalah nilai current node ditambah dengan nilai edge yang menghubungkan node tersebut dengan current node. Jika nilai sementara lebih kecil daripada nilai node yang sedang diperhitungkan, maka nilai node tersebut akan digantikan dengan nilai sementaranya.

4. Setelah seluruh node tetangga selesai diperhitungkan, tandai node saat ini sebagai "*visited node*" atau "sudah dikunjungi". Sebuah *visited node* tidak akan pernah diperhitungkan lagi.
5. Pilih satu un*visited node* dengan nilai terkecil dan tetapkan node tersebut sebagai current node. Ulangi proses dari langkah 3 hingga seluruh node telah memiliki nilai dan telah ditandai sebagai "*visited node*".
6. Proses selanjutnya adalah menentukan rute mana yang dilewati. Proses akan berjalan terbalik dari node tujuan hingga sampai di node awal. Dimulai dengan menetapkan node tujuan sebagai current node. Kemudian kurangi nilai node saat ini dengan node tetangga ditambah dengan nilai edge-nya untuk mendapatkan nilai kelayakan jalur. Jalur yang memiliki nilai lebih kecil daripada nol berarti tidak layak digunakan, sementara jalur dengan nilai lebih besar daripada nol namun paling mendekati nol adalah jalur yang layak digunakan. Proses selesai ketika node awal telah tercapai dan rute yang menghubungkan node awal dan node tujuan telah terbentuk.

2.7.1 Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup system operasi, *middleware* dan aplikasi. Android menyediakan system operasi yang terbuka bagi para pengembang untuk menciptakan aplikasi. Android diluncurkan untuk umum pada tahun 2008. Android sangat berkembang pesat di industry karena dua asopek utama yaitu bersifat *opensource* dan model arsitekturnya. Sebagai sebuah proyek yang bersifat *opensource*, memungkinkan android untuk sepenuhnya dipahami dan dianalisis mengenai fitur, penyelesaian pada bug program hingga hardware [6].

Aplikasi Android terdiri dari komponen yang bebas digabungkan, diikat menggunakan suatu *project manifest* yang meng kan masing-masing komponen dan bagaimana komponen-komponen itu saling berhubungan. Ada enam komponen dalam Android untuk membangun suatu aplikasi yaitu [5] :

1. *Activities*

Activity merupakan bagian yang paling penting dalam sebuah aplikasi,karena *activity* menyajikan tampilan visual program yang sedang dijalankan oleh *user*. Setiap *activity* dideklarasikan dalam sebuah *class* yang bertugas untuk menampilkan antarmuka yang terdiri dari *views* dan *respons* terhadap *event*. Setiap aplikasi mempunyai satu buah *activity* atau lebih. Biasanya pasti aka nada *activity* yang pertama kali tampil ketika aplikasi sedang dijalankan. Perpindahan antara *activity* dengan *activity* lainnya ditentukan oleh posisinya oada *activity stack*. Bila suatu *activity* baru dimulai *activity* yang sebelumnya digunakan maka akan dipindahkan ke tumoukan paling atas. Ketika *activity* dipanggil dan disimpan dalam *activity stack* terdapat 4 kemungkinan kondisi transisi yang akan terjadi [5] :

a. *Active*

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

b. *Paused*

   Beberapa *activity* akan terlihat tapi tidak terfokus pada kondisi ini disebut *paused*, dia terlihat aktif namun tidak bias diakses oleh *user*.

c. *Stopped*

   Ketika sebuah *activity* tidak terlihat maka itulah yang disebut *stopped*. *Activity* akan tetap berada dalam memori dengan semua keadaan dan informasi yang ada. Namun akan di eksekusi oleh system jika system membutuhkan sumberdaya lebih.

d. *Inactive*

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

2. *Content Providers*

Content providers digunakan untuk mengelola dan berbagi database. Data dapat disimpan dalam file system, dalam database SQLite, atau dengan

cara lain yang pada prinsipnya sama. Dengan adanya content provider memungkinkan antar aplikasi untuk saling berbagi data.

3. *Services*

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

4. *Intens*

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

5. *Broadcast receivers*

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

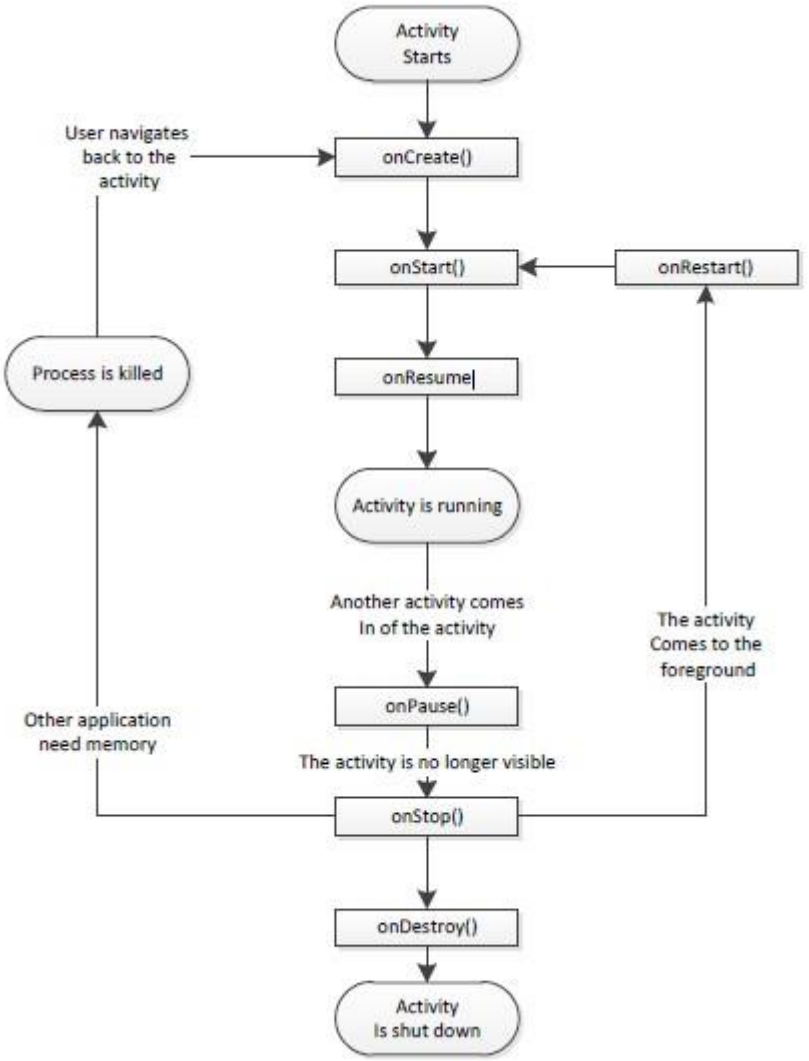
6. *Notifications*

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

2.7.2 Siklus Hidup Aplikasi Android

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

membutuhkan sumber daya yang lebih, maka activity yang prioritas rendah akan ditutup. Berikut adalah siklus hidup activity Android pada gambar 2.4[6] :



Gambar 2. 4 Siklus Hidup *Activity* Android

Pengguna mengakses method init pada activity class, dan Android akan memanggilnya menurut waktu yang tepat[6].

1 . *onCreate( )*

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

2 . *onStart*( )

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

3. *onResume( )*

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

4. *onPause*( )

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

5. onStop( )

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

6. *onRestart*( )

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

7. onDestroy( )

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

8. *onSaveInstanceState( )*

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

9. *onRestoreInstanceState*( )

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

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

2.7.3 Karakteristik Android

Andorid merupakan subset perangkat lunak untuk perangkat mobile yang meliputi sistem operasi, middleware, dan aplikasi inti yang di-release oleh Google. SDK adalah suatu tools dan API yang diperlukan untuk mengembangkan aplikasi pada platform atau lingkungan Android. Pengembangan aplikasi Android menggunakan bahasa pemrograman Java. Seperti kita ketahui, SDK ini dikembangkan oleh OHA (*Open Handsate Alliance*). Organisasi OHA ini terdiri atas Google, Intel, Motorola, Qualcomm, T-Mobile, dan NVIDIA[6].

2.7.4 Perkembangan Android

Android telah banyak melakukan perkembangan, berikut adalah perkembangan dari android[6] :

1. Android Versi 1.1

   Pada 9 Maret 2009, Google merilis Android versi 1.1. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, voice search (pencarian suara), pengiriman pesan dengan Gmail dan pemberitahuan email.

2. Android Versi 1.5 (Cupcake)

   Pada pertengahan Mei 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (Software Development Kit) dengan versi 1.5 (Cupcake). Terdapat beberapa pembaruan termasuk juga penambahan beberapa

fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke Youtube dan ke Picasa langsung dari telepon, dukungan Bluetooth A2DP, kemampuan terhubung secara otomatis ke headset Bluetooth, animasi layar, dan keyboard pada layar yang dapat disesuaikan dengan system.

3. Android Versi 1.6 (Donuts)

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

4. Android Versi 2.1 (Éclair)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (*Eclair*), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan flash untuk kamera 3.2MP, digital zoom, dan Bluetooth 2.1.

5. Android Versi 2.2 (Froyo)

Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, intergrasi V8 JavaScript engine yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam SD Card, kemampuan WiFi Hotspot portabel, dan kemampuan auto update dalam aplikasi Android Market.

6. Android Versi 2.3 (Gingerbread)

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

layar antar muka (User Interface) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (reverb, equalization, headphone virtualization, dan bass boost), dukungan kemampuan Near Field Communication (NFC), dan dukungan jumlah kamera yang lebih dari satu.

7. Android Versi 3.0 (HoneyComb)

Android Honeycomb dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. User Interface pada Honeycomb juga berbeda karena sudah didesain untuk tablet. Honeycomb juga mendukung mult i prosesor dan juga akselerasi perangkat keras (hardware) untuk grafis. Tablet pertama yang dibuat dengan menjalankan Honeycomb adalah Motorola Xoom. Perangkat tablet dengan platform Android 3.0 akan segera hadir di Indonesia. Perangkat tersebut bernama Eee Pad Transformer produksi dari Asus. Rencana masuk pasar Indonesia pada Mei 2011.

8. Android Versi 4.0 (Ice Cream Sandwich)

Diumumkan pada tanggal 19 Oktober 2011, membawa fitur Honeycomb untuk smartphone dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara offline, dan berbagi informasi dengan menggunakan NFC.

9. Android Versi 4.1 (Jelly Bean)

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

10. Android Versi 4.4 (Kitkat)

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

11. Android Versi 5.0 (Lollipop)

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

12. Android Versi 6.0 (Marshmallow)

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

13. 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.

2.7.5 Kelebihan Android

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

1. Keterbukaan, Bebas pengembangan tanpa dikenakan biaya terhadap system karena berbasiskan Linux dan open source. Pembuat perangkat menyukai hal ini karena dapat membangun platform yang sesuai yang diinginka n tanpa harus membayar royality. Sementara pengembang software menyukai karena android dapat digunakan diperangkat manapun dan tanpa terikat oleh vendor manapun.
2. Banyak dukungan service, kemudahan dalam menggunakan berbagai macam layanan pada aplikasi seperti penggunaan layanan pencarian lokasi, database SQL, browser dan penggunaan peta. Semua itu sudah tertanam pada android sehingga memudahkan dalam pengembangan aplikasi.
3. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga kerja system menjadi lebih stabil. Pengguna tak perlu kawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.
4. Dukungan grafis dan suarat terbaik, dengan adanya dukungan 2D grafis dan animasi yang diilhami oleh Flash menyatu dalam 3D menggunakan OpenGL memungkinkan membuat aplikasi maupun game yang berbeda.
5. Portabilitas aplikasi, aplikasi dapat digunakan pada perangkat yang ada saat ini maupun yang akan datang. Semua program ditulis dengan menggunakan bahas pemrograman Java dan dieksekusi oleh mesin virtual Dalvik, sehingga kode program portabel antara ARM, X86, dan arsitektur lainnya. Sama halnya dengan dukungan masukan seperti penggunaan Keyboard, layar sentuh, trackball dan resolusi layar semua dapat disesuaikan dengan program.

2.7.6 Kekurangan Android

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

1. Koneksi Internet yang terus menerus, kebanyakan ponsel berbasis system ini memerlukan koneksi internet yang simultan alias terus menerus aktif. Koneksi internet GPRS selalu aktif setiap waktu, itu artinya Anda harus siap berlangganan paket GPRS yang sesuai dengan kebutuhan.
2. Iklan – Aplikasi di Ponsel Android memang bisa didapatkan dengan mudah dan gratis, namun konsekuensinya di setiap Aplikasi tersebut, akan selalu Iklan yang terpampang, entah itu bagian atas atau bawah aplikasi.

2.7.7 Android SDK

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

1. Framework : aplikasi yang mendukung pengganti komponen dan reusable.
2. Dalvik Virtual Machine dioptimalkan untuk perangkat mobile.
3. Integrated Browser verdasarkan engine open source WebKit.
4. Grafis yang dioptimalkan dan didukung oleh libraries grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional Ekselerasi hardware)
5. SQLite untuk penyimpanan data.
6. Media Support yang mendukung audio, video, dan (MPEG4, H.264, MP3, AAC, AMR, JPG, PING, GIF), GSM Telephony (tergantung Hardware).

7. Bluetooth, EDGE, 3G, dan WiFi (tergantung hardware).
8. Kamera, GPS, Kompas, dan Accelerometer (tergantung hardware).
9. Lingkungan Development yang lengkap dan termasuk pernagkat emulator, tools untuk debugging, profil dan kinerja memori, dan plugin untuk IDE Android Studio.

2.7.8 Android Studio

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

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

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

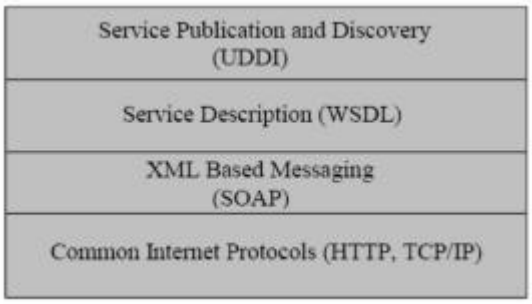
Satu hal tambahan lagi yang membuat Android Studio unggul adalah dukungan layout xml editor secara visual yang jauh lebih baik daripada Eclipse.

2.8 Web Services

Web service merupakan suatu komponen software yang merupakan selfcontaining, aplikasi modular self-describing yang dapat dipublikasikan,

dialokasikan, dan dilaksanakan pada web. Web service adalah teknologi yang mengubah kemampuan internet dengan menambahkan kemampuan transactional web, yaitu kemampuan web untuk saling berkomunikasi dengan pola program-toprogram (P2P). Fokus web selama ini didominasi oleh komunikasi program-to-user dengan interaksi business-to-consumer (B2C), sedangkan transactional web akan didominasi oleh program-to-program dengan interaksi business-to-business. Gambar 2.5 merupakan blok bangunan web service yang mana menyediakan fasilitas komunikasi jarak jauh antara dua aplikasi yang merupakan layer arsitektur web service.

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



Gambar 2. 5 Blok Bangunan *Web Service*

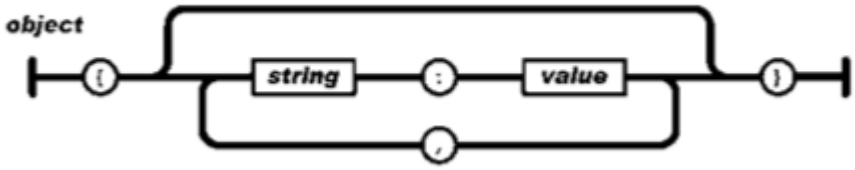
2.9 JavaScript Object Notation (JSON)

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

karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data.
JSON terbuat dari dua struktur :

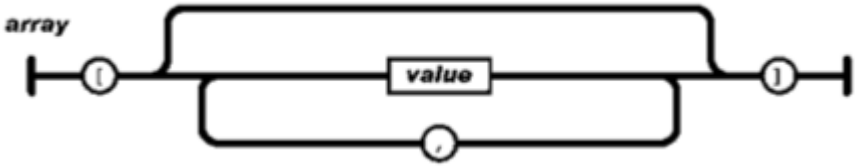
1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (object), rekaman (record), struktur (struct), kamus (dictionary), tabel hash (hash table), daftar berkunci (keyed list), atau associative array.
2. Daftar nilai terurutkan (an ordered list of values). Pada kebanyakan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

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini[7].
JSON menggunakan bentuk sebagai berikut[7]:
Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



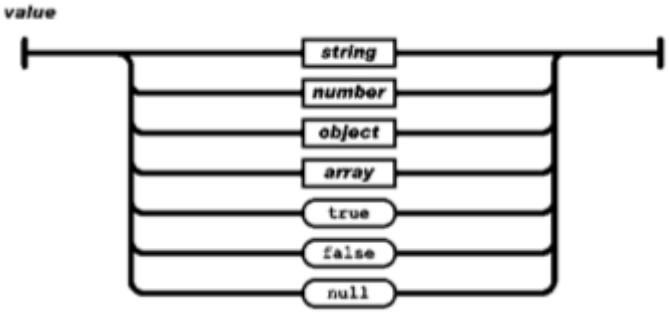
Gambar 2. 6 Objek JSON

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



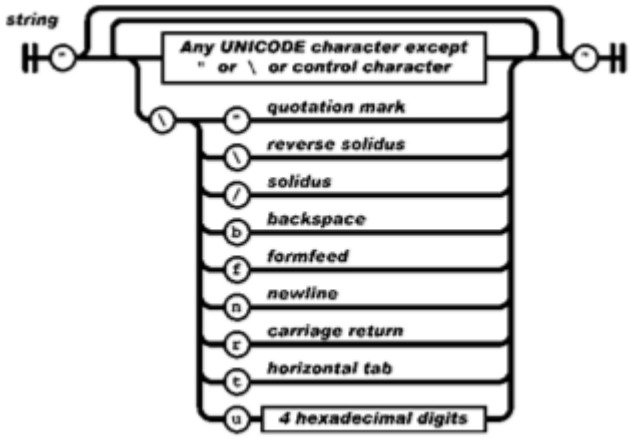
Gambar 2. 7 Array JSON

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.



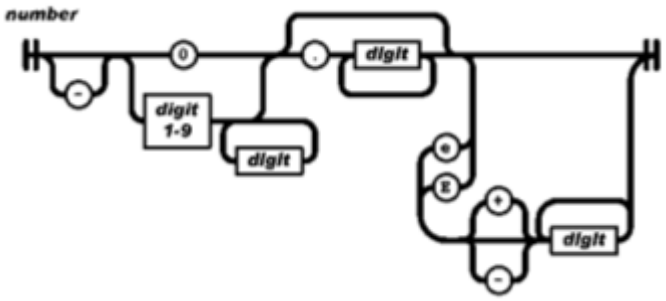
Gambar 2. 8 Value JSON

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. 9 Sting JSON

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



Gambar 2. 10 Number JSON

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

2.10 Application Programming Interface (API)

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

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

Kelebihan pemograman menggunakan API adalah[8] :

1. Portabilitas

   Programmer yang menggunakan API dapat menjalankan programnya dengan sistem operasi mana saja asalkan sudah terinstal API tersebut.

2. Mudah Dimengerti

2.10.1 Twitter API

Pada Awalnya perusahan Summize yang menyediakan fasilitas mencari data di Twitter. Kemudian perusahaan Summize ini diakuisisi dan diganti menjadi merek menjadi Twitter Search API terpisah sebagai entitas sendiri.

API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa system call.

API Twitter terdiri dari 3(tiga) bagaian yaitu :

1. ***Search* API**. *Search* API dirancang untuk memudahkan user dalam mengelola query search dikonten Twitter. User dapat menggunakan untuk mencari tweet berdasarkan keyword khusus atau mencari tweet lebih spesifik berdasarkan username Twitter. Search API juga menyediakan akses pada data Trending Topic.
2. ***Representastional State Transfer* (REST) API**. REST API memperbolehkan developer untuk mengakses inti dari Twitter seperti timeline, status update dan nformasi user. REST API digunakan dalam hal membangun sebuah aplikasi Twitter yag kompleks yang memerlukan inti dari Twitter.
3. ***Steaming* API**. Streaming API digunakan developer untuk kebutuhan yang lebih intensif seperti melakukan penelitian dan analistik data. Streaming API dapat menghasilkan aplikasi yang dapat mengetahui statistik status update, follwer dan lain sebagainya. Dalam penelitian ini, bagian Twitter API yang di gunakan adalah REST API.

2.11 Object Oriented Analysis and Desaing (OOAD)

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 requireme nt (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[9].

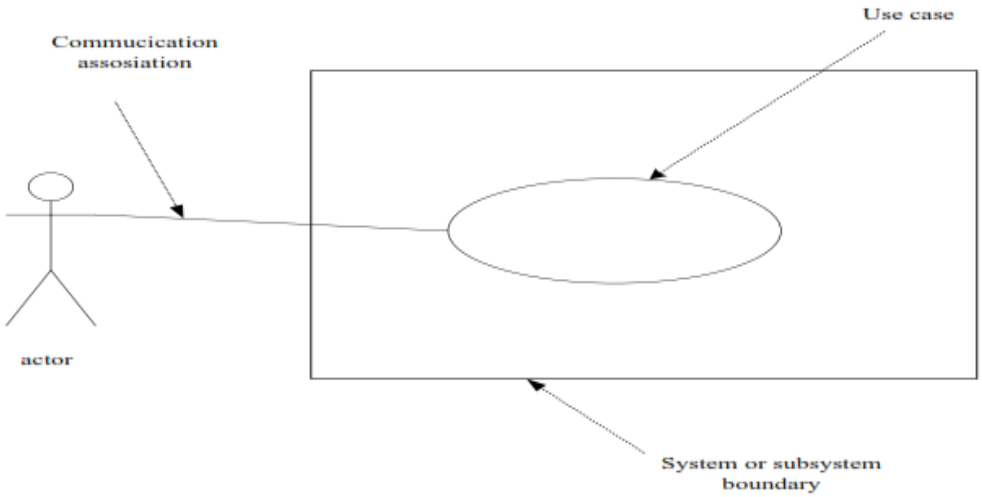
2.11.1 Unfied Modeling Language (UML)

Unified Modeling Language (UML) adalah termasuk ke dalam rumpun jenis pemodelan notasi grafis yang didukung oleh meta-model tunggal. Pemodelan ini berguna untuk membantu dalam menjelaskan dan merancang perangkat lunak yang dibangun dengan object-oriented (OO). UML merupakan standar terbuka yang dikelola oleh Open Management Group (OMG) yang berada dibawah naungan perusahaan-perusahaan konsorsium terbuka[10]. UML merupakan bahasa pemodelan yang terdiri banyak model diantaranya adalah :

1. *Usecase Diagram*

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

   Usecase menjelaskan apa yang dilakukan system (atau subsistem) tetapi tidak menspesifikasikan cara kerjanya. *Flow of event* digunakan untuk menspesifikan kelakuan dari *usecase. Flow of event* menjelaskan *usecase* dalam bentuk tulisan dengan sejelas-jelasnya, diantaranya bagaimana, kapan *usecase* dimulai dan berakhir, ketika use case berinteraksi dengan aktor, obyek apa yang digunakan, alur dasar dan alur alternative[10].

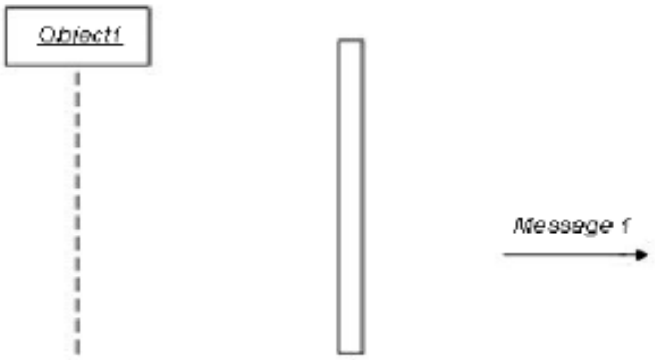


Gambar 2. 11 Notasi Usecase Diagram

2. *Sequence Diagram*

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

Sequence diagram menangkap objek dan class yang terlibat dalam scenario dan urut-urutan pesan yang ditukar antara objek diperlukan untuk melaksanakan fungsionalitas skenario. Sequence diagram berasosiasi dengan use case selama proses pengembangan. Dalam Unified Model Language (UML), objek dalam sequence diagram di dengan segiempat yang berisi nama objek yang diberi garis bawah. Objek dapat diberi nama dengan tiga cara : (nama objek), (nama objek dan class) atau (hanya nama class (anonymous object)). Berikut notasi sequence diagram seperti terlihat pada dibawah ini[10] :



Gambar 2. 12 Notasi Sequence Diagram

3. *Class Diagram*

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

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

1. Nama *class.*
2. Atribut.
3. Operasi/*Method.*

Tabel 2. 3 Class Diagram

Nama class
Atribut
Operasi/method

Atribut dan Operasi/method dapat memiliki tiga sifat berikut:

1. Public, dapat dipanggil oleh class apa saja.

2. *Protected*, hanya dapat dipanggil atau diakses oleh class yang bersangkutan dan *class* turunannya.
3. *Private*, hanya dapat dipanggil oleh dirinya sendiri (tidak dapat diakses dari luar class yang bersangkutan).

4. *Actifity Diagram*

Activity diagram menggambarkan berbagai alir aktifitas dalam sistem yang sedang diarancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *Action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity* diagram tidak menggambarkan *behavior* internal sebuah sistem.

Dalam penelitian ini, *activity diagram* menjelaskan tata cara pengunjung mall Istana Plaza untuk menggunakan aplikasi dengan sebuah penggambaran proses yang terjadi di suatu aplikasi.

2.12 Pengujuan Black-Box

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

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

Macam-macam pengujian metode Black Box :

a. *Equivalence Partitioning*.

b. *Boundary Value Analysis*.
c. *Cause-Effect Graphing Techniques*.
d. *Comparison Testing*.

2.13 Skala Likert

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

Tabel 2. 4 Skor Skala Likert

Skala Jawaban	Keterangan	Skor
SS	Sangat Setuju	5
S	Setuju	4
RG	Ragu	3
TS	Tidak Setuju	2
STS	Sangat Tidak Setuju	1

Data dari kuesioner akan diolah untuk mencari nilai rata-rata dari masingmasing jawaban dengan menggunakan rumus skala likert.

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

Keterangan :
p = nilai rata – rata yang dicari.
f = bobot jawaban * responden pemilih.
n = jumlah responden