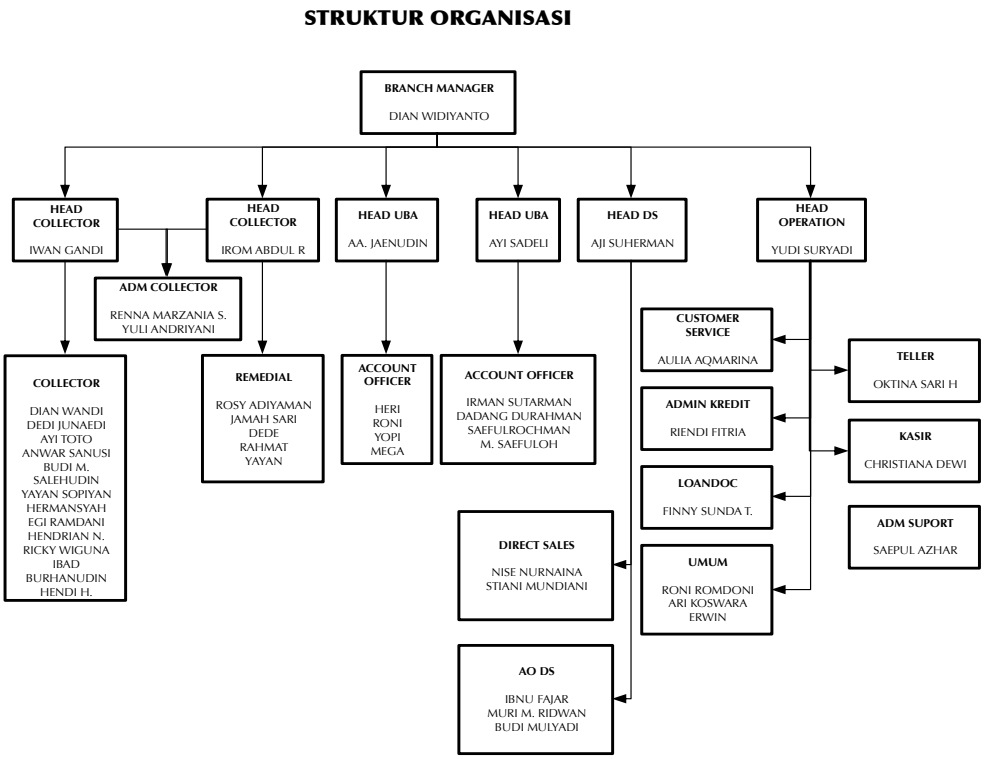
BAB II
LANDASAN TEORI

II.1 Tentang Bess Finance

Bess Finance merupakan badan usaha diluar bank atau lembaga keuangan bukan bank yang memberikan fasilitas pinjaman kepada nasabahnya. Awal mulanya perusahaan ini bernama PT. Adrindo Executive Finance yang memiliki cabang beralamat di Jl. Raya Cibiru No. 589, Cileunyi, Bandung, Jawa Barat, Indonesia.

II.2 Struktur Organisasi Perusahaan

Berikut ini adalah adalah struktur organisasi Bess Finace yang dapat dilihat pada Gambar 2.1 Struktur Organisasi Perusahaan.



Gambar II.1 Struktur Organisasi Perusahaan

II.3 Visi dan Misi

Berikut adalah visi dan misi dari Bess Finance:

1. Visi

   Menjadi perusahaan pembiayaan pilihan.
2. Misi
   a. Memberikan kemudahan dalam pelayanan kepada konsumen.
   b. Membangun komunikasi dan hubungan yang baik dengan pemangku kepentingan.
   c. Menumbuhkan sikap profesionalisme bagi seluruh jajaran organisasi.

II.4 Sistem Informasi

Sistem Informasi adalah kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi itu untuk mendukung operasi dan manajemen. Dalam arti yang sangat luas, istilah sistem informasi yang sering digunakan merujuk kepada interaksi antara orang, proses algoritmik, data, dan teknologi. Berikut adalah penjelasan sistem informasi menurut para ahli:

II.4.1 Pengertian Sistem

Menurut Jogiyanto (2003:34), Sistem dapat didefinisikan dengan pendekatan prosedur dan pendekatan komponen. Dengan pendekatan prosedur, sistem dapat didefinisikan sebagai kumpulan dari prosedur-prosedur yang mempunyai tujuan tertentu. Contoh sistem yang didefinisikan dengan pendekatan prosedur ini adalah sistem akuntansi. Sistem ini didefinisikan sebagai kumpulan dari prosedur-prosedur penerimaan kas, pengeluaran kas, penjualan, pembelian dan buku besar. Sedangkan dengan pendekatan komponen, sistem dapat didefinisikan sebagai kumpulan dari komponen yang saling berhubungan satu dengan yang lainnya membentuk satu kesatuan untuk mencapai tujuan tertentu. Contoh sistem yang didefinisikan dengan pendekatan ini misalnya adalah sistem komputer yang didefinisikan sebagai kumpulan dari perangkat keras dan perangkat lunak[2].

II.4.2 Pengertian Sistem Informasi

Menurut Ladjamudin (2005:13) Sistem Informasi dapat didefinisikan sebagai berikut:

a. Suatu sistem yang dibuat oleh manusia yang organisasi untuk mencapai suatu tujuan yaitu menyajikan informasi.
b. Sekumpulan prosedur organisasi yang pada saat dilakasanakan akan memberikan informasi bagi pengambilan keputusan dan/atau untuk mengendalikan organisasi.
c. Suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi, mendukung operasi, bersifat manajerial, dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan.

Menurut Sutabri (2004:36) “Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial[2].”

Definisi diatas dapat disimpulkan bahwa sistem informasi adalah sekumpulan komponen yang saling berhubungan, mengumpulkan, memproses, menyimpan, dan mendistribusikan informasi untuk menunjang pengambilan keputusan dan pengawasan dalam suatu organisasi[2].

II.5 Monitoring

Monitoring adalah proses pengumpulan analisis informasi (berdasarkan indikator yang ditetapkan) secara sistematis dan kontinu tentang kegiatan program / proyek sehingga dapat dilakukan tindakan koreksi untuk penyempurnaan program / proyek itu selajutnya[3].

II.5.1 Pengertian Monitoring

Adapun pengertian Monitoring menurut para ahli adalah sebagai berikut[3]:

1. Cassely dan Kumar 1987

Monitoring merupakan program yang terintegrasi, bagian penting dipraktek manajemen yang baik dan arena itu merupakan bagian integral di manajemen sehari-hari.

2. Calyton dan Petry 1983

Monitoring sebagai suatu proses mengukur, mencatat, mengumpulkan, memproses dan mengkomunikasikan informasi untuk membantu pengambilan keputusan manajemen program/proyek.

3. Oxfam 1995

Monitoring adalah mekanisme yang sudah menyatu untuk memeriksa yang sudah untuk memeriksan bahwa semua berjalan untuk direncanakan dan memberi kesempatan agar penyesuaian dapat dilakukan secara metodologis.

4. SCF 1995

Monitoring adalah penilaian yang skematis dan terus menerus terhadap kemauan suatu pekerjaan.

5. WHO

Monitoring adalah suatu proses pengumpulan dan menganalisis informasi dari penerapan suatu program termasuk mengecek secara reguler untuk melihat apakah kegiatan/program itu berjalan sesuai rencana sehingga masalah yang dilihat /ditemui dapat diatasi.

6. *Webster's New Collegiate Dictionary* (1981)

Monitoring menurut *Webster's New Collegiate Dictionary* (1981) adalah: "*a device for observing or giving admonition or warning*". Sementara itu menurut *Webstern's New World Dictionary*, maka pengertian "*Monitoring* adalah *something that reminds or warns' or any of various devices for checking or regular the performance*".

II.5.2 Tujuan Monitoring

Tujuan dari Monitoring adalah sebagai berikut[3]:

1. Mengkaji apakah kegiatan-kegiatan yang dilaksanakan telah sesuai dengan rencana.
2. Mengidentifikasi masalah yang timbul agar langsung dapat diatasi.
3. Melakukan penilaian apakah pola kerja dan manajemen yang digunakan sudah tepat untuk mencapai proyek.
4. Mengetahui kaitan antara kegiatan dengan tujuan untuk memperoleh ukuran kemajuan.
5. Menyesuaikan kegiatan dengan lingkungan yang berubah, tanpa menyimpang dari tujuan.

II.5.3 Manfaat Monitoring

Monitoring juga memiliki beberapa manfaat. Berikut adalah manfaat *monitoring*[3]:

1. Bagi pihak penanggung jawab program
   a. Salah satu fungsi manajemen yaitu pengendalian atau supervisi.
   b. Sebagai bentuk pertanggungjawaban (akuntabilitas) kinerja.
   c. Untuk meyakinkan pihak-pihak yang berkepentingan.
   d. Membantu penentuan langkah- langkah yang berkaitan dengan kegiatan proyek selanjutnya.
   e. Sebagai dasar untuk melakukan Monitoring dan evaluasi selanjutnya.
2. Bagi pihak pengelola proyek
   a. Membantu untuk mempersiapkan laporan dalam waktu singkat.
   b. Mengetahui kekurangan-kekurangan yang perlu diperbaiki dan menjaga kinerja yang sudah baik.
   c. Sebagai dasar (informasi) yang penting untuk melakukan evaluasi proyek.

II.6 Android

Android merupakan subset perangkat lunak untuk perangkat mobile yang meliputi sistem operasi, middleware¸ dan aplikasi inti yang dirilis oleh Google.

Android adalah sistem operasi bergerak (*mobile operating system*) yang mengadopsi sistem operasi linux, namun telah dimodifikasi. Android diambil alih oleh Google pada tahun 2005 dari Android,Inc sebagai bagian strategi untuk mengisi pasar sistem operasi bergerak. Google mengambil alih seluruh hasil kerja Android termasuk tim yang mengembangkan Android[4].

II.6.1 Sejarah Android

Perjalanan Android dimulai sejak Oktober 2003 ketika 4 orang pakar IT, Andi Rubin, Rich Minner, Nick Sears dan Chris White mendirikan Android.Inc, di California US. Visi Android untuk mewujudkan *mobile device* yang lebih peka dan mengerti pemiliknya, kemudian menarik raksasa dunia maya *Google*. *Google* kemudian mengakuisisi Android pada Agustus 2005. OS Android dibangun berbasis *platform* Linux yang bersifat *open source*,senada dengan Linux, Android juga bersifat *Open Source.* Dengan nama besar *Google* dan konsep *open source* pada OS Android, tidak membutuhkan waktu lama bagi android untuk bersaing dan menyisihkan *Mobile OS* lainnya seperti *Symbian, Windos Mobile, Blackberry* dan iOS. Kini siapa yang tak kenal Android yang telah menjelma menjadi penguasa *Operating System* bagi *Smartphone*[4].

II.6.2 Versi Android

Android memiliki beberapa versi sejak awal kemunculannya. Berikut adalah beberapa versi android[4]:

1. *Android 1.5 Cupcake*

   Cupcake dirilis 30 April 2009. *Cupcake* menjadi versi android pertama yang menggunakan nama makanan. Konon katanya versi ini seharusnya versi 1.2, namun Google memutuskan untuk membuat revisi besar dan membuatnya menjadi versi 1.5 *Cupcake* adalah kue kecil yang dipanggang dalam cetakan berbentuk cup.
2. *Android 1.6 Donut*

   Android V1.6, *codename Donut,* dirilis pada 15 September 2009. Pada versi ini diperbaiki beberapa kesalahan reboot, perubahan fitur foto dan video

dan integrasi pencarian yang lebih baik. Donat merupakan panganan berbentuk cincin. Bulat bolong ditengah. Adonan donat dimasak dengan cara digoreng dan biasanya disajikan dengan toping diatasnya.

3. *Android 2.0/2.1 Eclair*

Android 2.0/2.1 *Eclair* Dirilis 26 Oktober 2009. *Eclair* adalah makanan penutup yakni kue yang biasanya berbentuk persegi panjang yang dibuat dengan krim di tengah dan lapisan cokelat di atasnya.

4. *Android 2.2 Froyo*

Dirilis 20 Mei 2010. Menggunakan *codename Froyo*, yang merupakan makan penutup yang nama merek sebuah produk yang terbuat dari *Yoghurt*. *Froyo* singkatan dari *Frozen Yoghurt*, *Froyo* adalah *yoghurt* yang telah mengalami proses pendinginan, Sehingga secara terlihat sama seperti es krim.

5. *Android 2.3 Gingerbread*

Android versi 2.3 *Gingerbread* dirilis resmi tanggal 6 Desember 2010. *Gingerbread* merupakan jenis kue kering yang dengan rasa jahe. Kue jahe biasanya dibuat pada perayaan hari libur akhir tahun di Amerika. Biasanya cemilan kering ini dicetak berbentuk tubuh manusia.

6. *Android 3.0 Honeycomb*

Dirilis tanggal 22 February 2011. *Honeycomb* adalah sereal sarapan manis yang sudah dibuat tahun 1965 oleh Posting Sereal. Seperti namanya, *Honeycomb* atau sarang lebah, sereal ini terbuat dari potongan jagung berbentuk sarang lebah dengan rasa madu.

7. *Android 4.0 Ice Cream Sandwich*

Android 4.0-4.0.2 API Level 14 dan 4.0.3-4.0.4 API Level 15 pertama dirilis 19 Oktober 2001. Dinamai *Ice Cream Sandwich*. *Ice Cream Sandwich* adalah lapisan es krim, biasanya rasa vanilla yang terjepit di antara dua kue coklat, dan biasanya berbentuk persegi panjang.

8. *Android 4.1 Jelly bean*

Android *Jelly Bean* diluncurkan pertama kali pada Juli 2012, dengan berbasis *Linux Kernel* 3.0.31. Terdiri dari Android 4.1 API Level 16, Android 4.2 API Level 17, Android 4.3 API Level 18. Penamaan *Jelly Bean*

mengadaptasi nama sejenis permen dalam beraneka macam rasa buah. Ukurannya sebesar kacang merah. Permen ini keras di luar tapi lunak di dalam serta lengket bila di gigit.

9. *Android 4.4 KitKat*

Android 4.4 *Kitkat* API level 19. Google mengumumkan Android *KitKat* (dinamai dengan izin dari *Nestle* dan *Hershey*) pada 3 september 2013. Dengan tanggal rilis 31 Oktober 2013. *KitKat* merupakan merk sebuah coklat yang dikeluarkan oleh *Nestle*. Rilis berikutnya setelah nama KitKat diperkirakan banyak pengamat akan diberi nomor 5.0 dan dinamai '*Key Lime Pie'*.

10. *Android 5.0 Lollipop*

Android 5.0 *Lollipop* API level 21 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

11. *Android 6.0 Marshmallow*

Android 6.0 Marshmallow API level 23 merupakan pemutakhiran yang akan datang untuk sistem operasi telepon genggam Android, kemungkinan besar akan dirilis pada Q3 2015 ("sementara dijadwalkan untuk September"), dengan pratayang ketiga dan terakhir dirilis pada tanggal 17 Agustus 2015. Pertama diperkenalkan di Google I/O pada tanggal 28 Mei 2015, Marshmallow terutama akan berfokus pada perbaikan inkremental dan penambahan fitur lainnya.

II.6.3 Android Life Cycle

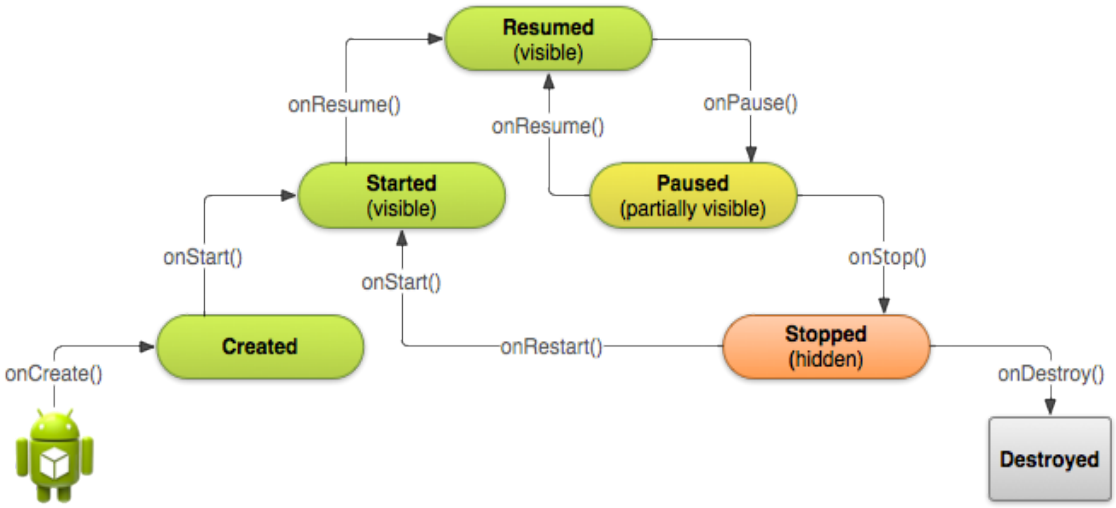
Aplikasi android terdiri dari beberapa fungsi dasar seperti mengedit catatan, memutar file musik, membunyikan alarm, atau membuka kontak telepon. Fungsi-fungsi tersebut dapat diklasifikasikan ke dalam empat komponen android yang berbeda. klasifikasi tersebut berdasarkan kelas-kelas dasar java yang digunakan[6]:

Table II.1 Komponen Aplikasi Android

Functionality	Java Base Class	Examples
Focused thing a user can do	Activity	Edit a note, play a game
Background process	Service	Play music, update weather icon
Receive messages	BroadcastReceiver	Trigger alarm upon event
Store and retrieve data	ContentProvider	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[6].

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 atau web. Hal ini dikarenakan aplikasi mobile memiliki tingkat interupsi proses yang lumayan 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[5].



Gambar II.2 Siklus Hidup Android [8]

1. Resumed

Resumed terjadi ketika aplikasi berjalan setelah state *paused* . State ini akan menjalankan perintah program yang ditulis pada method onResume() (Google Inc, 2014).

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 (Google Inc, 2014).

3. **Stopped**

Dalam keadaan ini, aplikasi benar-benar tidak ditampilkan dan tidak terlihat oleh pengguna tetapi masih meninggalkan service dibackground (Google Inc, 2014).

State lain seperti Created dan Started bersifat sementara dan sistem dengan cepat menjalankan state berikutnya dengan memanggil metode *life cycle callback* berikutnya. Artinya, setelah sistem OnCreate() dipanggil, dengan cepat sistem akan memanggil method OnStart(), krmudian diikuti oleh onResume() (Schwarz, Dutson, Stelee, & To, 2013).

II.6.4 Fitur Pada Android

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

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 [5].

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 [5].

Multi-proses dapat memberikan manfaat berupa user experience yang lebih banyak, namun penggunaan fitur tersebut dapat menghabiskan banyak energi baterai jika penggunaan tidak benar [6].

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 [6].

3. Hard dan Soft Keyboard

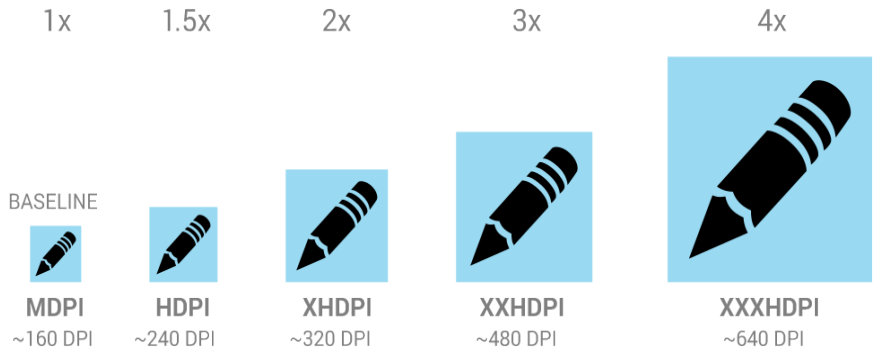
Salah satu fitur pada perangkat smartphone adalah tombol fisik dan non fisik, tombol fisik digunakan untuk navigasi pendukung dalam pengoperasian android.. Pengembang aplikasi tidak perlu secara manual untuk mengintegrasikan tombol tersebut dalam aplikasi. Tombol non fisik adalah tombol yang dibuat oleh sistem operasi seperti keyboard virtual, dan tombol navigasi aplikasi [6].

II.6.5 Prinsip Desain

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

1. Multiple Assets

Android mendukung jutaan *smartphone*, tablet dan perangkat lain dalam berbagai ukuran layar dan ukuran, untuk itu *Multiple Assets* sangat disarankan digunakan untuk mengatasi fragmentasi pada android. Seperti ilustrasi pada Gambar dibawah. Klasifikasi Ukuran Ikon, 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 [7].



Gambar II.3 Klasifikasi Ukuran Ikon

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 [7].



Gambar II.4 *Touch Feedback*

3. *Pattern Gesture*

Touch gesture berguna untuk meningkatkan *experience* pengguna dalam menggunakan aplikasi. Terdapat beberapa gesture yang didukung oleh android diataranya adalah [7]:

1. Touch

   Touch adalah gesture default sebagai event untuk item tertentu yang paling sering digunakan dalam perangkat android, hampir seluruh kontrol utama aplikasi ini menggunakan touch.

2. Long Press

   Biasanya digunakan untuk seleksi data , dengan gesture ini dimungkinkan untuk memilih satu atau lebih item dalam sebuah tampilan dan menjalankan suatu fungsionalitas tertentu [7].

3. Swipe Or Drag

   Swipe adalah menyentuh sebuah titik pada layar dan menggerakkan jari yang tetap tersentuh pada layar ke titik lain pada layar. Swipe biasanya dilakukan untuk kebutuhan berpindah layar atau tergantung dari aplikasi. Swipe dapat dilakukan dari dan ke arah mana saja [7].

4. Double Touch

Pada Smartphone dan tablet android, melakukan dua kali tapping secara berturut-turut pada satu obyek, fungsinya berbeda dengan double klik mouse komputer. Pada Android, teknik ini biasanya dipakai untuk melakukan zoom in dan zoom out atau memperbesar / memperkecil sebuah obyek gambar [7].

5. Pinch Open

Teknik lain yang biasa digunakan adalah dengan menggunakan dua jari, di mana kedua jari tersebut menyentuh dua titik pada layar yang terpisah di mana ujung dari dua jari tersebut tidak bersentuhan, kemudian kedua jari tersebut sambil tetap menyentuh layar bergerak saling mendekati. Teknik ini digunakan untuk membuka aplikasi tertentu [7].

6. Pinch Close

Teknik Pinch Close adalah kebalikan dari Pinch Open di mana spread dilakukan dengan berawal dari dua jari yang bersentuhan pada ujungnya dan ditempelkan sebuah titik yang sama pada layar, kemudian kedua jari tersebut bergerak memisahkan atau menjauhi satu sama lain Gerakan ini untuk menutup layar [7].

II.7 Web Services

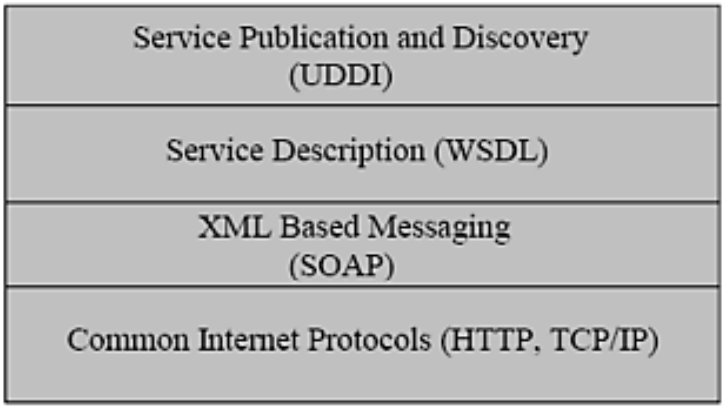
Web service adalah layanan yang diidentikasi dengan URI (*Uniform Resource Identier*) yang mengeksposturnya melalui internet menggunakan protokol dan bahasa standar internet serta dapat diimplementasikan menggunakan standar internet seperti XML (*Extensible Markup Language*)

Sampai dengan saat ini teknologi *web service* terus berkembang. Salah satu teknologi yang populer saat ini adalah REST (*Representational State Transfer*) atau terkadang disebut dengan RESTful. Beberapa contoh RESTful *web service* adalah *Amazon's Simple Storage Service* (S3), *Atom Publishing Protocol*, dan *Google Maps*. Pada prinsipnya *request* ke suatu RESTful *web service* sebenarnya adalah suatu HTTP *Request*.

Ada beberapa standar yang digunakan dalam *web service*. Beberapa di antaranya adalah SOAP (*Simple Object Access Protocol*), BPEL (*Business Process Execution Language*), UDDI (*Universal Description, Discovery, and Integration Infrastructure*), WSDL (*Web Service Description Language*)[8].

Gambar 2.6 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 II.5 Blok Bangunan *Web Service*

II.8 JSON (*JavaScript Object Notation)*

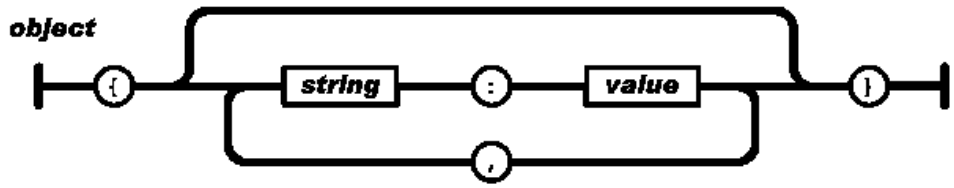
JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca, dan ditulis oleh manusia, serta mudah diterjemahkan dan

dibuat oleh komputer. Format ini dibuat berdasarkan bagian dari bahasa pemrograman *JavaScript*. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum seperti C, C++, C#, *Java*, *JavaScript*, *Perl*, dan *Python*. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur yaitu kumpulan pasangan nama atau nilai dan daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*)[9].

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 [9].

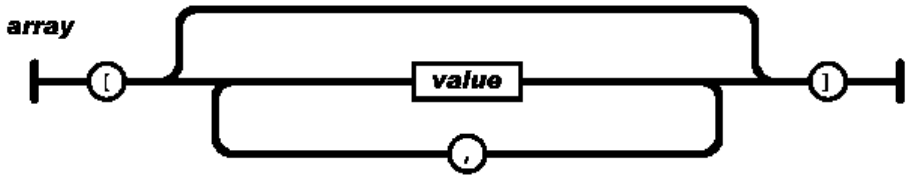
JSON menggunakan bentuk sebagai berikut [9] :

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).



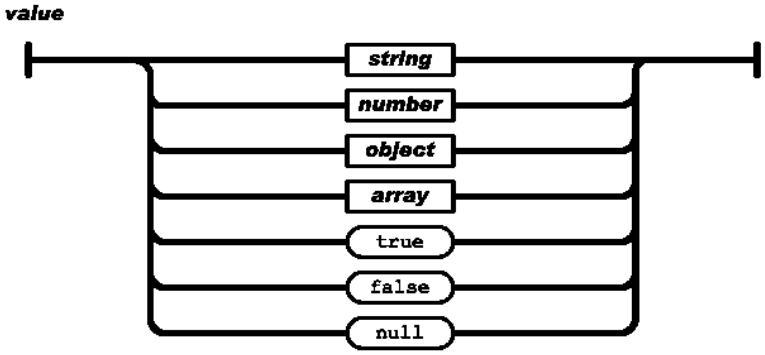
Gambar II.6 Objek JSON

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



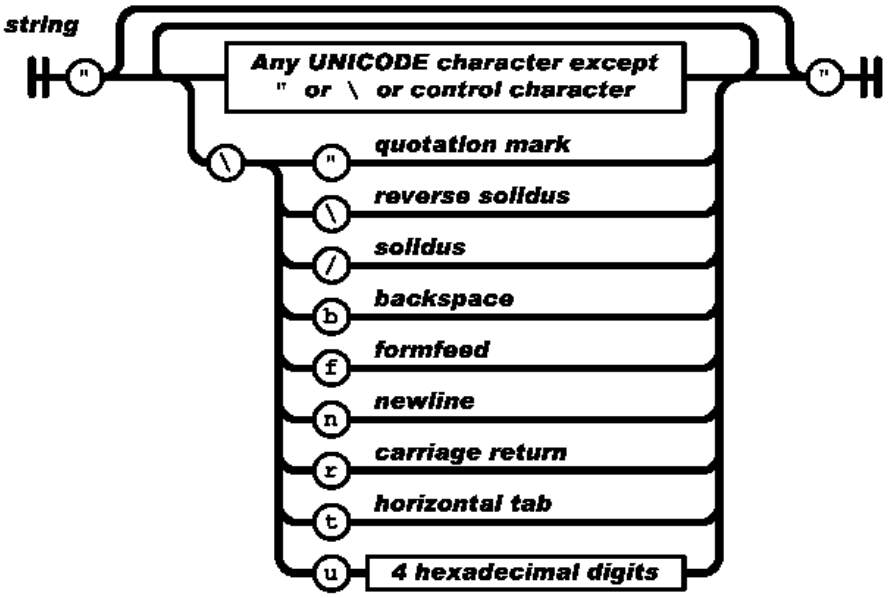
Gambar II.7 Array JSON

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.



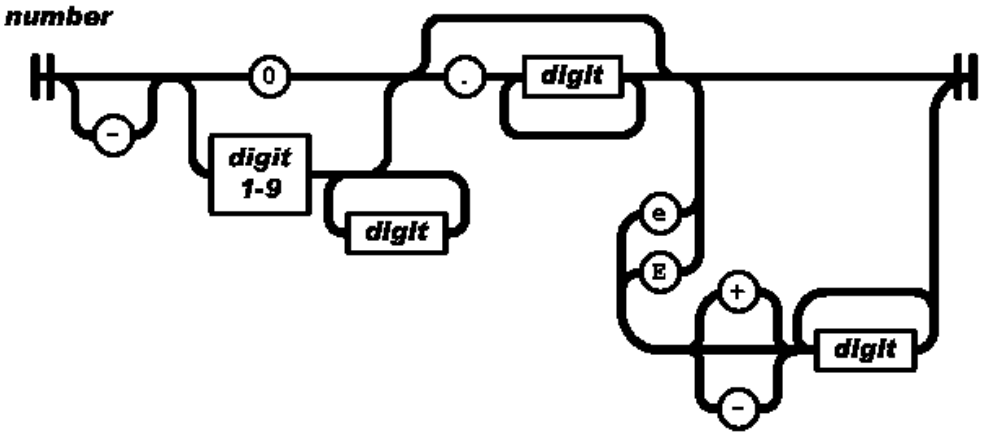
Gambar II.8 Value JSON

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 II.9 String JSON

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



Gambar II.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.

II.9 Global Positioning System (GPS)

GPS (*Global Positioning System*) adalah sistem radio navigasi dan penentuan posisi menggunakan satelit. Nama formalnya biasa disebut NAVSTAR GPS, kepanjangannya adalah *Navigation Satellite Timing and Ranging Global Positioning System*. *Global Positioning System* adalah sistem navigasi berbasis satelit yang terdiri dari jaringan 24 satelit yang mengorbit oleh Departemen Pertahanan Amerika Serikat. Pada awalnya GPS digunakan untuk aplikasi militer, namun pada 1980-an, Pemerintah membuat sistem yang tersedia untuk penggunaan sipil. GPS bekerja di semua kondisi cuaca, di mana saja di dunia selama 24 jam sehari[10].

Ekadinata menyatakan bahwa elemen satelit terdiri dari 24 satelit di luar angkasa (21 aktif, 3 cadangan) yang beredar di ketinggian 19,300 km di atas permukaan Bumi. Jaringan satelit ini disebut konstelasi GPS. Ketinggian tersebut dibutuhkan agar satelit dapat mencakup areal yang cukup luas. Posisi satelit tersebut di luar angkasa diatur sedemikian rupa sehingga pengguna di Bumi dapat menangkap paling tidak 4 sinyal satelit setiap waktu[10].

Satelit-satelit GPS harus selalu berada pada posisi orbit yang tepat untuk menjaga akurasi data yang dikirim ke GPS *receiver*, sehingga harus selalu dipelihara agar posisinya tepat. Posisi satelit-satelit tersebut selalu dipantau oleh stasiun pengendali. Stasiun-stasiun pengendali di bumi ada di Hawaii, *Ascension Islan, Diego Garcia, Kwajalein* dan *Colorado Spring*. Untuk dapat mengetahui posisi seseorang maka diperlukan alat yang diberi nama GPS *receiver* yang berfungsi untuk menerima sinyal yang dikirim dari satelit GPS. GPS *receiver* mengambil informasi tersebut dan melakukan perhitungan triangulation untuk menentukan lokasi pengguna dengan tepat. GPS *receiver* membandingkan waktu sinyal dikirim dengan waktu sinyal tersebut diterima untuk mengetahui jarak satelit. Dengan mengetahui jarak tersebut, GPS *receiver* dapat melakukan perhitungan dan menentukan posisi pengguna dan menampilkan dalam peta elektronik. Setelah menentukan posisi pengguna, selanjutnya GPS dapat menghitung informasi lain, seperti kecepatan, arah yang dituju, jalur, tujuan

perjalanan, jarak tujuan, matahari terbit dan matahari terbenam dan masih banyak lagi[10].

Gambaran sistem kerja atau cara kerja Satelit GPS yaitu melingkari Bumi dua kali sehari dalam orbit yang sangat tepat dan mengirimkan sinyal informasi ke Bumi. Orbit satelit berinklinasi 55 derajat terhadap bidang ekuator dengan ketinggian rata-rata dari permukaan Bumi sekitar 20.200 km. Satelit GPS bergerak dalam orbitnya dengan kecepatan kira-kira 3,87 km/ detik dan mempunyai periode 11 jam 58 menit (sekitar 12 jam). Sehingga 4 sampai 10 satelit GPS akan selalu dapat diamati pada setiap waktu dari manapun di permukaan Bumi. Penerima GPS menerima informasi ini dan triangulasi yang digunakan untuk menghitung lokasi pasti pengguna[10].

II.10 GPS Photo Tagging (*Geotagging*)

GPS *Photo Tagging* juga dikenal sebagai *geotagging*, merupakan proses penambahan informasi posisi data GPS (*Latitude*, *Longitude*, *Altitude*) dalam sebuah foto digital. Ponsel-ponsel berkamera yang memiliki GPS *receiver* internal umumnya memiliki fitur ini. Mekanisme GPS *PhotoTagging* adalah pada saat sebuah foto diambil menggunakan kamera (digital atau ponsel) yang memiliki fitur *geotagging*, kamera atau ponsel tersebut mencatat lebih banyak informasi/data dibandingkan dengan sebuah foto yang diambil dengan kamera biasa. Informasi tersebut termasuk waktu dan data ketika sebuah foto diambil, orientasi dari kamera (*portrait* atau *landscape*), apakah menggunakan lampu flash dan detil kamera lainnya yang digunakan seperti Apertur, Exposure, dan Local Length. Semua data ini disimpan pada suatu tempat yang disebut EXIF Headers.

EXIF (*Exchangeable Image File Format*) headers berisi petunjuk foto dengan data yang dapat dibaca oleh aplikasi manajemen foto atau sebuah situs foto tertentu. Selain itu, EXIF *Headers* juga menyediakan sebuah ruang untuk mengisi koordinat *Longitude*, *Latitude*, dan *Altitude* [11].

II.11 EXIF Headers

Data EXIF (*Exchangeable Image File Format*) adalah data yang disimpan oleh kamera digital pada sebuah image yang berisi informasi mengenai kondisi dan setting kamera digital pada waktu dilakukan pemotretan. EXIF dikembangkan oleh Japanese Electronics Industry Development Association (JEIDA) sebagai upaya untuk mempermudah dan membuat standar pertukaran data antara perangkat lunak pengolah gambar/citra digital dan perangkat keras seperti kamera, serta didukung oleh hampir seluruh kamera digital[11].

Informasi umum yang bisa didapatkan dari EXIF sebuah foto digital adalah antara lain [11]:

1. Tanggal dan jam berapa sebuah foto dibuat.
2. Merk, tipe kamera, dan jenis lensa yang dipakai untuk memotret foto.
3. Resolusi dari kamera yang digunakan untuk mengambil gambar. Digunakan atau tidaknya fitur flash pada kamera.
4. Lokasi foto tersebut diambil, jika kamera yang digunakan mendukung GPS.

II.12 EXIF Interface Android Class

Kelas *Exif Interface* adalah salah satu kelas dalam pemrograman android yang dapat digunakan untuk membaca dan menulis informasi tag Exif pada file yang berformat JPEG. Berikut adalahTabel 2.2 Tag EXIF informasi Exif yang dapat diambil [12].

Table II.2 Tag EXIF

No.	Tagging	Keterangan
1	TAG_APERTURE	Type is String.
2	TAG_DATETIME	Type is String.
3	TAG_DATETIME_DIGITIZED	Type is String.
4	TAG_EXPOSURE_TIME	Type is String.
5	TAG_FLASH	Type is int.
6	TAG_FOCAL_LENGTH	Type is rational.

No.	Tagging	Keterangan
7	TAG_GPS_ALTITUDE	The altitude (in meters) based on the reference in TAG_GPS_ALTITUDE_REF.
8	TAG_GPS_ALTITUDE_REF	0 if the altitude is above sea level.
9	TAG_GPS_DATESTAMP	Type is String.
10	TAG_GPS_LATITUDE	String
11	TAG_GPS_LATITUDE_REF	Type is String.
12	TAG_GPS_LONGITUDE	String
13	TAG_GPS_LONGITUDE_REF	Type is String.
14	TAG_GPS_PROCESSING_METHOD	Type is String.
15	TAG_GPS_TIMESTAMP	Type is String.
16	TAG_IMAGE_LENGTH	Type is int.
17	TAG_IMAGE_WIDTH	Type is int.
18	TAG_ISO	Type is String.
19	TAG_MAKE	Type is String.
20	TAG_MODEL	Type is String.
21	TAG_ORIENTATION	Type is int.
22	TAG_SUBSEC_TIME	Type is int.
23	TAG_SUBSEC_TIME_DIG	Type is int.
24	TAG_SUBSEC_TIME_ORIG	Type is int.
25	TAG_WHITE_BALANCE	Type is int.

II.13 *Google Geocoding*

Google Maps sudah memiliki fitur yang cukup lengkap, termasuk mencari lokasi kota pada peta (*geocoding*). *Geocoding* adalah proses menemukan koordinat geografis, yaitu bujur dan lintang, dari data geografis lain, misalnya alamat. Fitur *geocoding* dapat dimanfaatkan untuk menambahkan informasi *latitude* dan *longitude* pada foto yang tidak memiliki data *geotagging* [11].

II.14 *Structure Analysis and Design*

Structured analysis and design adalah pengidentifikasian komponen-komponen sistem informasi dengan tujuan untuk dikomunikasikan dengan pemakai (Sutanta, 2003). Tujuan *Structured analysis and design* secara umum adalah memberikan gambaran secara umum atau global kepada pemakai tentang

sistem yang akan dikembangkan dan berfungsi sebagai persiapan untuk tahap perancangan sistem.

II.15 Object Oriented Analysis Design

Konsep OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaiut analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang memerika 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 [15].

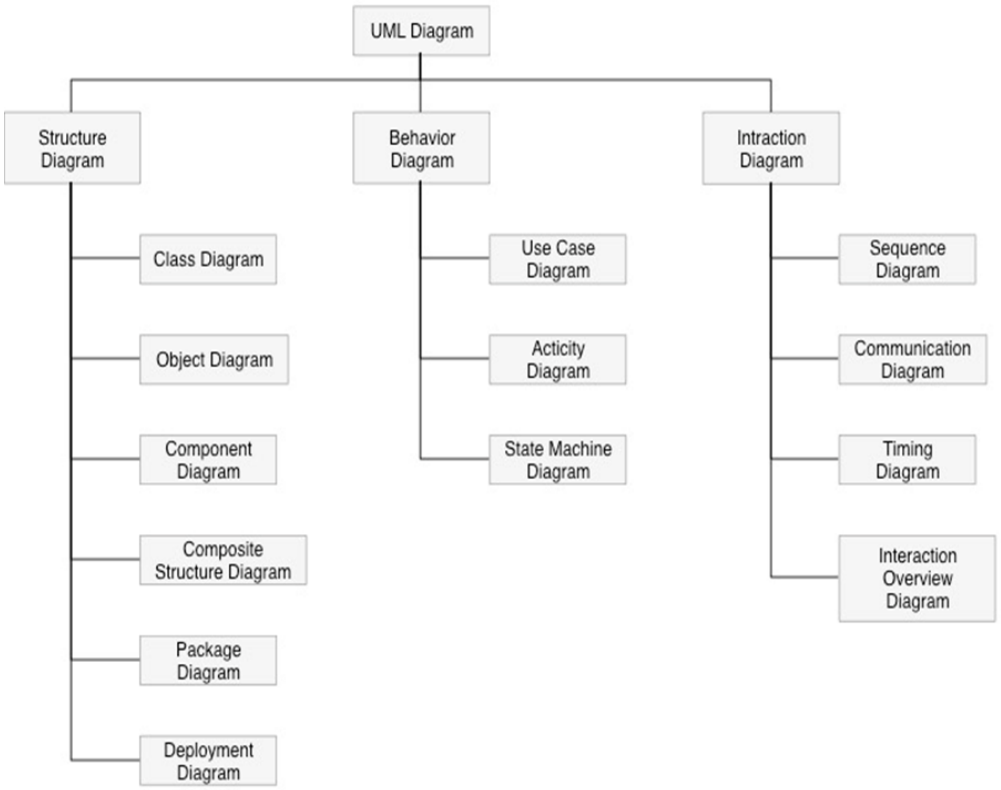
II.15.1 Unified Modelling Language (UML)

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). UML merupakan standar yang relatif terbuka yang dikontrol oleh *Object Management Group* (OMG), sebuah konsorsium terbukayang terdiri dari banyak perusahaan. OMG dibentuk untuk membuat standar-standar yang mendukung interoperabilitas, khususnya interoperabilitas sistem berorientasi objek. OMG mungkin lebih dikenal dengan standar-standar CORBA (*Common Object Request Broker Architecture*) [16]. UML terdiri dari 13 jenis diagram resmi seperti tertulis dalam Tabel II.10 Jenis Diagram Resmi UML [16].

Table II.3 Jenis Diagram Resmi UML [16]

Diagram	Kegunaan
Activity	Behavior prosedural dan parallel
Class	Class, fitur, dan hubungan-hubungan
Communication	Interaksi antar objek; penekanan pada jalur
Component	Struktur dan koneksi komponen

Composite structure	Dekomposisis *runtime* sebuah class
Deployment	Pemindahan artifak ke node
Interaction overview	Campuran sequence dan activity diagram
Object	Contoh konfigurasi dari contoh-contoh
Package	Struktur hirarki *compile-time*
Sequence	Interaksi antar objek; penekanan pada *sequence*
State machine	Bagaimana event mengubah objek selama aktif
Timing	Interaksi antar objek; penekanan pada *timing*
Use case	Bagaimana pengguna berinteraksi dengan sebuah system



Gambar II.11 Klarifikasi Jenis Diagram UML [16]

Adapun penjelasan dari *Class* Diagram, *Use Case* Diagram dan *Sequence* Diagram adalah sebagai berikut [16]:

1. Class Diagram

   Class diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antaranya. Class diagram juga menunjukan properti dan operasi sebuah class dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. Kotak-kotak yang terdapat di dalam diagram merupakan class, yang dibagi menjadi tiga bagian : nama class (cetak tebal), atributnya dan operasinya.

   a. Properti

      Properti mewakili fitur-fitur struktural dari sebuah class. Properti merupakan sebuah konsep tunggal, tetapi tampak seperti dua notasi yang sedikit berbeda: atribut dan asosiasi.

   b. Atribut

      Notasi atribut mendeskripsikan properti dengan sebaris teks di dalam kotak class tersebut.

   c. Operasi

      Operasi merupakan suatu kegiatan yang dimengerti sebuah class untuk dilakukan. Operasi secara jelas berhubungan dengan metode dalam sebuah class.

2. Use Case Diagram

   Use case diagram menampilkan aktor, use case dan hubungan antar keduanya :

   a. Aktor mana yang menggunakan use case mana
   b. Use case mana yang memasukkan use case lain

3. Sequence Diagram

   Sebuah sequence diagram secara khusus menjabarkan *behaviour* sebuah skenario tunggal. Diagram tersebut menunjukkan sejumlah objek contoh dan pesan-pesan yang melewati objek-objek ini di dalam use case.

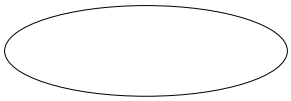
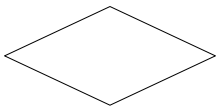
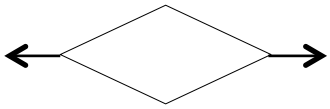
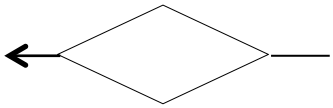
4. Activity Diagram

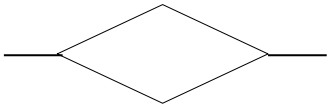
Activity diagram adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja. dalam beberapa hal, diagram ini seperti sebuah diagram alir, tetapi perbedaan prinsip antar diagram ini dan notasi diagram alir adalah diagram ini mendukung *behaviour* paralel.

II.16 Entity Relationship Diagram

Entity Relationship Diagram (ERD) adalah bagian yang menunjukkan hubungan antara entity yang ada dalam sistem. Simbol-simbol yang digunakan dapat dilihat dari tabel[17]. Komponen-komponen pembentuk ERD dapat di lihat pada Tabel 2.7 Komponen-komponen ERD [17].

Table II.4 Komponen-komponen ERD [17]

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

Notasi	Komponen	Keterangan
	Relasi N : N	Hubungan ini menunjukkan bahwa setiap entitas yang pertama dapat berhubungan dengan banyak entitas pada himpunan entitas yang kedua, demikian juga sebaliknya

II.17 Java

Java menurut definisi dari Sun adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer standalone ataupun pada lingkungan jaringan. Java dikembangkan pada bulan Agustus 1991, dengan nama semula Oak. Pada Januari 1995, karena nama Oak dianggap kurang komersial, maka diganti menjadi Java. Pada Desember 1998, Sun memperkenalkan nama "Java 2" (J2) sebagai generasi kedua dari java platform. Konvensi nama baru ini diterapkan untuk semua edisi Java yaitu Standard Edition (J2SE), Enterprise Edition (J2EE), dan Micro Edition (J2ME) [18].

Ada tiga platform Java yang masing-masing diarahkan untuk tujuan tertentu dan untuk lingkungan komputasi yang berbeda-beda:

1. Standard Edition (J2SE) : J2SE merupakan inti dari bahasa pemrograman Java. J2SE didesain untuk jalan pada komputer desktop dan komputer workstations.
2. Enterprise Edition (J2EE): Dengan built-in mendukung untuk servlets. JSP, dan XML, edisi ini ditujukan untuk aplikasi berbasis server.
3. Micro Edition (J2ME) : Didesain untuk piranti dengan memori terbatas, layar display terbatas dan power pemrosesan yang juga terbatas.

   Kelebihan Java

II.17.1 Kelebihan Java

Adapun beberapa kelebihan dari bahasa pemrograman Java adalah sebagai berikut [18] :

1. Sederhana dan Ampuh

Java dirancang untuk mudah dipelajari, terutama bagi programmer yang telah mengenal C/C++ akan mudah sekali untuk berpindah ke Java. Pemakai dapat belajar membuat program dengan Java secara cepat jika telah memahami konsep dasar pemrograman berorientasi objek. Java tidak memiliki hal-hal yang mengejutkan dan aneh. Java memberi anda kemampuan untuk menuangkan semua ide, karena bahasa pemrograman ini bukan merupakan scripting language (bahasa naskah) yang menghilangkan kemampuan kita untuk berinovasi, tetapi dengan cara berorientasi objek yang mudah dan jelas.

2. Aman

Java dirancang sebagai bahasa pemrograman yang handal dan aman. Aplikasi-aplikasi yang dibangun dengan bahasa Java sangat handal dengan manajemen memori yang bagus. Aplikasi Java juga dikenal sangat secure, yaitu kasus-kasus seperti buffer everflow yang umumnya menjadi lubang keamanan aplikasi-aplikasi berbasis C/C++ tidak terjadi di Java, karena pengaturan securitynya yang bagus.

3. Berorientasi Objek

Paradigma pemrograman berorientasi objek merupakan paradigma pemrograman masa depan. Java merupakan bahasa pemrograman berorientasi objek. Java bukan turunan langsung dari bahasa pemrograman manapun, juga sama sekali tidak kompetibel dengan semuanya. Java memiliki keseimbangan, Java secara cepat jika telah memahami konsep dasar pemrograman berorientasi objek. Java tidak memiliki hal-hal yang mengejutkan dan aneh. Java memberi anda kemampuan untuk menuangkan semua ide, karena bahasa pemrograman menyediakan mekanisme peng-class-an sederhana, dengan model antar muka dinamik yang intuitif hanya jika diperlukan.

4. Kokoh

Java membatasi anda dari beberapa hal kunci supaya anda dapat menemukan kesalahan lebih cepat saat mengembangkan program. Java langsung memeriksa program saat anda menuliskannya, dan sekali lagi ketika program di jalankan. Karena Java adalah bahasa yang sangat ketat dalam hal tipe data dan

deklarasi, banyak kesalahan umum terjadi saat kompilasi. Hal ini akan lebih menghemat waktu jika dibandingkan dengan keharusan menjalankan program terlebih dahulu dan memeriksa semua bagian program untuk melihat ketidakcocokan dinamis selama program berjalan. Ini adalah contoh di mana Java lebih luwes dan kokoh dari beberapa bahasa lain, tetapi dengan imbalan yang layak untuk kelebihan itu.

5. Interaktif

Java memiliki beberapa kemampuan yang memungkinkan program melakukan beberapa hal pada saat bersamaan, tanpa harus kesulitan menangani proses yang akan terjadi selanjutnya. Jalinan program-program Java yang mudah digunakan memungkinkan kita untuk memikirkan pembuatan perilaku khusus, tanpa harus mengintegrasikan perilaku tersebut dengan model pemrograman global yang mengatur perulangan kejadian.

6. Netral terhadap Berbagai Arsitektur

Java telah mengambil beberapa keputusan yang sulit dalam pembuatan bahasa Java dan bagaimana program dijalankan, jadi anda dapat sepenuhnya percaya “tulis sekali, jalan di mana saja, kapan saja, selamanya”.

7. Terintrepentasi dan Berkinerja Tinggi

Java dilengkapi keajaiban lintas platform yang luar biasa dengan kompilasi ke dalam representasi langsung yang disebut kode-byte Java (Java byte-code), yang dapat diterjemahkan oleh sistem manapun yang memilki program Java didalamnya. Java, bagaimanapun dirancang untuk tetap berkinerja baik pada CPU yang tidak terlalu kuat. Walaupun Java merupakan bahasa terinterpretasi, kodekode Java telah dirancang dengan hati-hati sehingga mudah diterjemahkan ke dalam bahasa asli suatu mesin untuk menghasilkan kinerja yang tinggi.

Sistem program Java yang melakukan optimasi tepat waktu tersebut tidak kehilangan keuntungan dari program yang netral terhadap platform. “lintas platform berkinerja tinggi” bukan sekedar omong-kosong. Dalam aplikasi Java (*.class) merupakan Java bytecode yang berjalan di atas jvm (Java Virtual Machine), yang kemudian jvm-lah yang akan menginterpresentasikan kode-kode

tersebut ke kode native atau kode mesin dari arsitektur yang bersangkutan. Hal sangat menarik karena urusan arsitektur mesin bukan jadi masalah bagi programmer tapi menjadi urusan kompiler pada bahasa pemrograman Java.

II.18 *Processor Hypertext Protocol* (PHP)

PHP adalah bahasa server-side –scripting yang menyatu dengan HTML untuk membuat halaman web yang dinamis. Karena PHP merupakan server-side-scripting maka sintaks dan perintah-perintah PHP akan diesksekusi diserver kemudian hasilnya akan dikirimkan ke browser dengan format HTML.

Dengan demikian kode program yang ditulis dalam PHP tidak akan terlihat oleh user sehingga keamanan halaman web lebih terjamin. PHP dirancang untuk membuat halaman web yang dinamis, yaitu halaman web yang dapat membentuk suatu tampilan berdasarkan permintaan terkini, seperti menampilkan isi basis data ke halaman web [19].

II.19 MySQL

MySQL adalah salah satu jenis database server yang sangat terkenal dan banyak digunakan untuk membangun aplikasi web yang menggunakan database sebagai sumber dan pengolahan datanya. MySQL dikembangkan oleh perusahaan swedia bernama MySQL AB yang pada saat ini bernama Tcx DataKonsult AB sekitar tahun 1994-1995, namun cikal bakal kodenya sudah ada sejak tahun 1979. Awalnya Tcx merupakan perusahaan pengembang software dan konsultan database, dan saat ini MySQL sudah diambil alih oleh Oracle Corp.

Kepopuleran MySQL antara lain karena MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses databasenya sehingga mudah untuk digunakan, kinerja query cepat, dan mencukupi untuk kebutuhan database perusahaan-perusahaan yang berskala kecil sampai menengah, MySQL juga bersifat open source (tidak berbayar). MySQL merupakan database yang pertama kali didukung oleh bahasa pemrograman script untuk internet (PHP dan Perl). MySQL dan PHP dianggap sebagai pasangan software pembangun aplikasi web yang ideal. MySQL lebih sering digunakan untuk membangun aplikasi berbasis

web, umumnya pengembangan aplikasinya menggunakan bahasa pemrograman script PHP [19].

II.20 *Google Maps Directions* API

Google Maps Directions API adalah layanan yang digunakan untuk menghitung arah antar lokasi menggunakan permintaan HTTP. Selain itu dapat digunakan menetapkan titik jalan untuk arah mengemudi, berjalan atau bersepeda. Titik jalan memungkinkan untuk menghitung rute melalui lokasi tambahan, di mana rute yang dikembalikan menyertakan persinggahan di setiap titik jalan yang diberikan[21].

Permintaan *Google Maps Directions API* mengambil bentuk berikut:

https://maps.googleapis.com/maps/api/directions/**output**?**parameters**

Dalam hal ini, output bisa berupa salah satu nilai berikut:

a. json (disarankan) menunjukkan output dalam *JavaScript Object Notation* (JSON)
b. xml menunjukkan output berupa XML

Untuk mengakses *Google Maps Directions API* melalui HTTP menggunakan:

http://maps.googleapis.com/maps/api/directions/**output**?**parameters**

URL *Google Maps Directions API* dibatasi sekitar 2000 karakter, Setelah Pengkodean URL. Karena sebagian URL *Google Maps Directions API* bisa melibatkan banyak lokasi sepanjang lintasan. Beberapa parameter tertentu diperlukan sementara yang lainnya bersifat opsional. Sebagaimana standar dalam URL, semua parameter dipisah menggunakan karakter *ampersand* (&). Daftar parameter dan kemungkinan nilainya disebutkan di bawah ini[21].

a. *origin* — Alamat, nilai garis lintang/garis bujur tekstual, atau ID tempat asal yang ingin dihitung arahnya.

   Jika meneruskan sebuah alamat sebagai *string*, layanan *Directions*

akan melakukan *geocode* atas *string* itu dan mengubahnya menjadi koordinat garis lintang/garis bujur untuk menghitung arah. Koordinat ini mungkin berbeda dengan yang dikembalikan oleh *Google Maps Geocoding API*. Misalnya pintu masuk bangunan dan bukan pusatnya. Jika meneruskan koordinat, itu akan digunakan tanpa diubah untuk menghitung arah. Pastikan tidak ada spasi di antara nilai garis lintang dan garis bujur.

ID Tempat harus diawali dengan place_id:. ID tempat hanya bisa ditetapkan jika permintaan menyertakan kunci API atau ID *klien Google Maps API for Work*.

b. *destination* — Alamat, nilai garis lintang/garis bujur tekstual, atau ID tempat tujuan yang ingin dihitung arahnya. Opsi untuk parameter *destination* sama dengan opsi untuk parameter origin yang dijelaskan di atas.
c. key — Kunci API aplikasi. Kunci ini mengidentifikasi aplikasi untuk keperluan manajemen kuota.

II.21 Metode Haversine

Metode Haversine digunakan untuk menghitung jarak antara titik di permukaan bumi menggunakan garis lintang (longitude) dan garis bujur (lattitude) sebagai variabel inputan. Haversine formula adalah persamaan penting pada navigasi, memberikan jarak lingkaran besar antara dua titik pada permukaan bola (bumi) berdasarkan bujur dan lintang .

Dengan mengasumsikan bahwa bumi berbentuk bulat sempurna dengan jari-jari R 6.367,45 km, dan lokasi dari 2 titik di koordinant bola (lintang dan bujur) masing-masing adalah lon1, lat1, dan lon2, lat2, maka rumus Haversine dapat ditulis dengan persamaan sebagai berikut [22]:

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

Gambar II.12 Rumus Haversine