BAB 2
LANDASAN TEORI

2.2. Landasan Teori

Landasan teori menjelaskan beberapa teori-teori dan penjelasan yang berkaitan dengan aplikasi mobile android yang akan dibangun. Teori – teori yang terkait dengan pembangunan aplikasi akan di jelasakan pada bab ini.

2.2.1. Laundry Sepatu

Saat ini sedang maraknya koleksi sepatu dengan segala jenis dikalangan anak muda. Sepatu dengan merek – merek yang terkenal banyak di pakai sekarang oleh anak muda untuk memenuhi kebutuhan sandang. Dengan maraknya tren ini kebanyakan pemilik sepatu tidak tahu bagaimana cara merawat sepatu yang berbeda jenis dan bahan yang sesuai, akibatnya sepatu yang dibeli dengan harga yang mahal selalu tidak bertahan lama dalam pemakaian.

Oleh karna itu dengan maraknya dengan tren tersebut, jasa laundry sepatu banyak dicari oleh kolektor sepatu dalam perawatan yang khusus untuk menjaga kondisi sepatu miliknya.

Laundry sepatu merupakan usaha yang jangka panjang karena selama kebutuhan orang terhadap sepatu semakin meningkat. Di Indonesia usaha laundry sepatu terbilang masih sedikit karena usaha laundry sepatu baru popular pada tahun 2016, tidak seperti laundry baju pada umumnya yang sudah ada lama dan sudah banyak di setiap daerah baik itu di kota maupun di desa.

2.2.2. Pelayanan laundry sepatu

Dalam pelayanan di laundry sepatu tidak hanya dalam cuci mencuci sepatu saja tetapi pelayanan laundry sepatu terdapat dalam perbaikan sepatu dan cat ulang sepatu juga. Pelayanan pada pencucian sepatu ada 2 jenis yaitu *fast cleaning* dan *deep cleaning*.

Fast cleaning merupakan pencucian sepatu dalam jangka waktu 15 menit – sampai 30 menit dengan teknik pembersihan general dan hanya pada bagian luar saja.

Deep cleaning merupakan pencucian sepatu dalam jangka waktu 1 hari – 3 hari dengan teknik pembersihan seluruh detail, focus setiap sudut dan pembersihan keseluruh sepatu.

2.2.3. Rekomendasi

Rekomendasi adalah saran yang sifatnya menganjurkan, membenarkan, atau menguatkan mengenai sesuatu atau seseorang. Rekomendasi sangat penting artinya untuk meyakinkan orang lain bahwa sesuatu bahwa seseorang tepat dan layak. Misalnya ketika seseorang akan menggunakan jasa online shop biasanya akan terlihat sebuah testimoni dari orang – orang yang sudah pernah bertransaksi sebelumnya, apakah banyak yang merekomendasikan atau tidak. Jika banyak testimoni positif maka akan menambah keyakinan seseorang untuk bertransaksi.

Rekomendasi biasanya dibuat dalam bentuk tertulis seperti review produk dan testimoni dari internet atau dalam bentuk yang lebih formal dalam bentuk surat rekomendasi.

2.3. Konsep Pemrograman Mobile

2.3.1. Aplikasi Mobile

Menurut Irwansyah & V.Moniaga dalam bukunya yang berjudul "Pengantar Teknologi Informasi", pengertian dari mobile applications adalah aplikasi perangkat lunak yang dibuat khusus untuk dijalankan di dalam tablet dan juga smartphone. Umumnya, developer mobile apps memerlukan IDE atau Intergrated Development Environments dan juga SDK untuk pengembangan dari mobile apps itu sendiri. Pada saat ini, pada smartphone dan juga tablet, ada satu aplikasi yang berguna untuk menyediakan berbagai macam aplikasi yang dapat dijalankan di device tersebut. Aplikasi ini sering disebut store. Contoh store yaitu apple apps store, samsung apps, amazon kindlefire, windows store dan google playstore [6].

Gambar 3.9 Store yang tersedia saat ini
(sumber:http://appshopper.com/http://www.brighthub.com/http://cliveric
h.com/http://www.geekwire.com/

Gambar 2. 1 Store Mobile

Jika membahas tentang Mobile Apps, umumnya kita tidak bisa tidak menyinggung soal Developer. Developer ialan badan usaha yang membuat aplikasi yang nantinya akan dijalankan dalam device. Pada dasarnya, aplikasi akan berjalan menggunakan tenaga baterai dan juga mendapat dukungan dari prosesor yang ukurannya lebih kecil dibanding dengan prosesor komputer. Sebelum dilempar ke pasar, umumnya mobile apps akan dites terlebih dahulu menggunakan emulator. Emulator merupakan salah satu cara untuk menghemat biaya yang dikeluarkan oleh developer untuk membuat mobile apps(Irwansyah & V.Moniaga:61-62) [6].

2.3.2. Teknologi GPS

Menurut Antonius Aditya Hartanto dalam bukunya yang berjudul "Mengenal Aspek Teknik dan Bisnis LBS" Bahwa Global Positioning System atau sering disingkat dengan GPS adalah sistem navigasi yang menggunakan satelit yang didesain agar dapat menyediakan posisi secara instan, kecepatan dan informasi waktu di hampir semua tempat dimuka bumi, setiap saat dan dalam kondisi cuaca apapun. Pada dasaranya, GPS merupakan aplikasi yang harus menunggu terlebih dahulu permintaan dari pengguna. Aplikasi ini menyediakan akurasi positioning atau penentuan posisi yang berkisar antara 100 meter(95% dari waktu), hingga 5 sampai 10 meter, juga sampai akurasi relatif pada submeter, dan bahkan tingkat subcentimeter. Secara umum, semakin tinggi akurasi yang dihasilkan akan memerlukan infrastruktur yang lebih canggih dan tentunya berhubungan dengan biaya yang harus dikeluarkan.

Pengguna GPS untuk penentuan posisi saat ini diantaranya adalah navigasi untuk kegiatan pribadi (hiking, pelayaramn, berburu, petunjuk ketika mengemudi dan lain sebagainya), navigasi di pesawat, survei di lepas pantai dan navigasi kapal, fleet tracking, pengendalian mesin, teknik sipil, survey daratan, GIS dan pemetaan, analisis deformasi dan sebagainya.

Program GPS dan operasionalnya saat ini sebagian besar bersumber pada Department of Defense (DoD). Amerika serikat yang dapat dikatakan merupakan pembuat sistem ini. Manajmennya sendiri dilaksanakan oleh US Air Force dengan panduan dari komite eksekutif DoD Positioning/Navigation. Komite ini menerima masukan dari komite yang sama dari Department of Transportation (DoT) yang bertindak sebagai suara sipil untuk urusan atauran GPS(NAPA, 1995). GPS asli hasil desain oleh US Department of Defense (DoD) terdiri atas tiga komponen utama, yaitu control segment, space segment, dan user segment.

1) GPS Control Segment

Control segment terdiri atas lima stasiun yang terletak di pangkalan Falcon Air Force, Colorado Springs, Hawaii, Ascension Island, Diego Garcia dan Kwajalein. Stasiun-stasiun ini adalah mata dan telinga bagi GPS, bertugas memonitor satelit-satelit tersebut sebagaimana mereka mengirimkan data overhead dengan mengukr jarak antarsatelit setiap 1.5 detik (Hofmann-Wellenhof, 1992). Data ini kemudian diperhalus dengan menggunakan informasi ionospheric dan meteorologi dan dikirikan ke Master Control Station yang berada di fasilitas US Air Force Space Command yang ada di Colorado Spring. Disinilah parameter-parameternya baru dapat menggambarkan perkiraaan dari orbit satelit dan clock performance. Informasi ini selanjutnya akan dikembalikan ketiga buah uplink station (ko-lokasi di stasiun pemonitor Ascension Island, Diego

Garcia dan Kwajalein) yang akan mengirimkan informasi tersebut ke satelit. Dengan mengacu pada luasnya daerah penyebaran dari control station, semua satelit GPS di-tracking selama 92% dari waktu.

2) GPS Space Segment

Space segment terdiri atas sebuah jaringan satelit dalam orbit lingkaran yang terdekat dengan tinggi nominar sekitar 20,183 km di atas permukaan bumi serta

dengan periode selama 12 jam. Konstelasi yang sesungguhnya adalah untuk 24 satelit, dalam 3 ruang orbit dan condong ke equator(Spilker, 1980). Akan tetapi, skenario ini telah mulai berubah dan satelit-satelit saat ini ditempatkan dalam enam ruang orbit yang berbeda, dengan emapt satelit di masing-masing ruang.

3) GPS User Segment

USER segement teridiri atas antenna dan prosessor receiver yang menyediakan positiong, kecepatan dan ketepatan waktu ke pengguna. Bagian ini menerima data dari satelit-satelit melalui sinyal radio yang dikirim-kan setelah mengalami koreksi oleh stasiun pengendali di daratan.

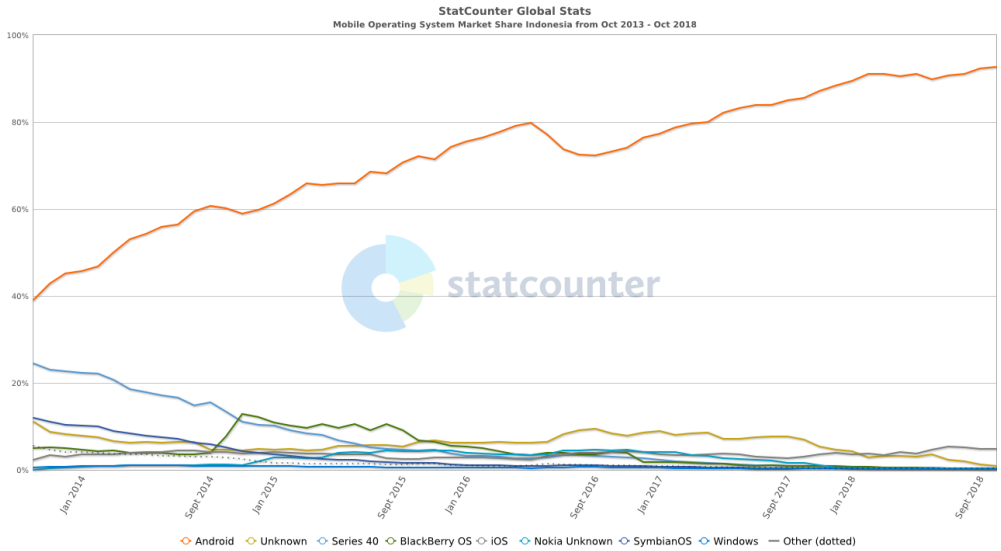
2.3 Android

Gambar 2. 2 Logo Android

Menurut Fachrul K & Gianto W dalam bukunya berjudul "Cepat Menguasi Pemrograman Android" bahwa, Android adalah salah satu sistem operasi yang pada awalnya, kemudian berkembang menjadi bahasa pemrograman yang banyak dicari dan digunakan oleh programmer. Pada dasarnya android adalah sistem operasi yang berbasis linux. Pengguna android pada awalna hanya digunakan untuk melengkapi sistem operasi pada *gadget-gadget* seluler seperti smartphone yang menggunakan layar sentuh. Tetapi karena sistem yang dikembangkan open source, mau tidak mau perkembangan dan penerimaan didunia industri IT menjadi lebih cepat juga [7].

Perusahaan android dibawah bendera android.inc, adalah pengembang pertama kali sistem operasi ini. Android.inc, pertama kali berdiri dikota Alto, salah satu kota erkenal di California Amerika Serikat, tepatnya pada bulan Oktober tahun 2003. Pendirinya terdiri dari tiga orang yang ahli dalam bidangn pengembangan aplikasi, mereka adalah Andy Rubin, Rich Miner, dan Chris White. Kemudian sejak tahun 2005, Google membeli dan selanjutnya di kembangkan oleh sumber daya Google sendiri sehingga mudah digunakan oleh khayalayak ramai. Dan sejat tahun 2007

secara resmi Google meluncurkan android sebagai sebuah sistem operasi baru khususnya untuk digunakan pada *smartphone* atau *gadget*, dan sistem operasi ini bersifat *open source* alias tidak diperjualbelikan. Sejak saat itu android bisa berkembang sedemikian rupa dan bisa menghasilkan berbagai macam aplikasi yang dibutuhkan oleh masyarakat untuk membantu kehidupan sehari-hari [7].



Gambar 2. 3 Statistik Pengguna OS Mobile di Indonesia

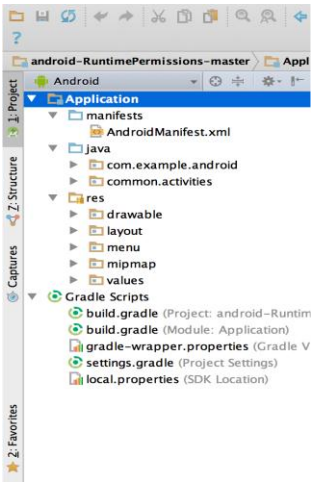
2.4 Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu - Integrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA . Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas Anda saat membuat aplikasi Android, misalnya:

1. Sistem versi berbasis Gradle yang fleksibel
2. Emulator yang cepat dan kaya fitur
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android
4. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru
5. Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh

6. Alat pengujian dan kerangka kerja yang ekstensif
7. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk Google Cloud Platform, mempermudah pengintegrasian Google Cloud Messaging dan App Engine.

1. Stuktur Proyek pada Android Studio



Gambar 2. 4 Tampilan File Struktur Android Studio

Setiap proyek di Android Studio berisi satu atau beberapa modul dengan file kode sumber dan file sumber daya. Jenis-jenis modul mencakup:

1. Modul aplikasi Android
2. Modul Pustaka
3. Modul Google App Engine

Secara default, Android Studio akan menampilkan file proyek Anda dalam tampilan proyek Android, seperti yang ditampilkan dalam gambar 1. Tampilan disusun berdasarkan modul untuk memberikan akses cepat ke file sumber utama proyek Anda.

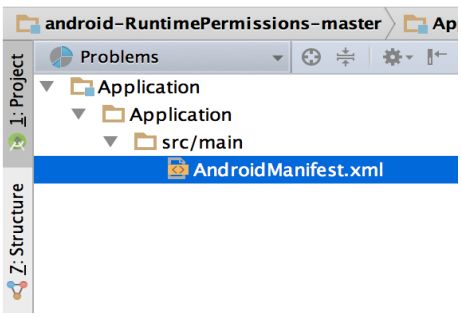
Semua file versi terlihat di bagian atas di bawah Gradle Scripts dan masing-masing modul aplikasi berisi folder berikut:

1. manifests: Berisi file AndroidManifest.xml.
2. java: Berisi file kode sumber Java, termasuk kode pengujian JUnit.

3. res: Berisi semua sumber daya bukan kode, seperti tata letak XML, string UI, dan gambar bitmap.

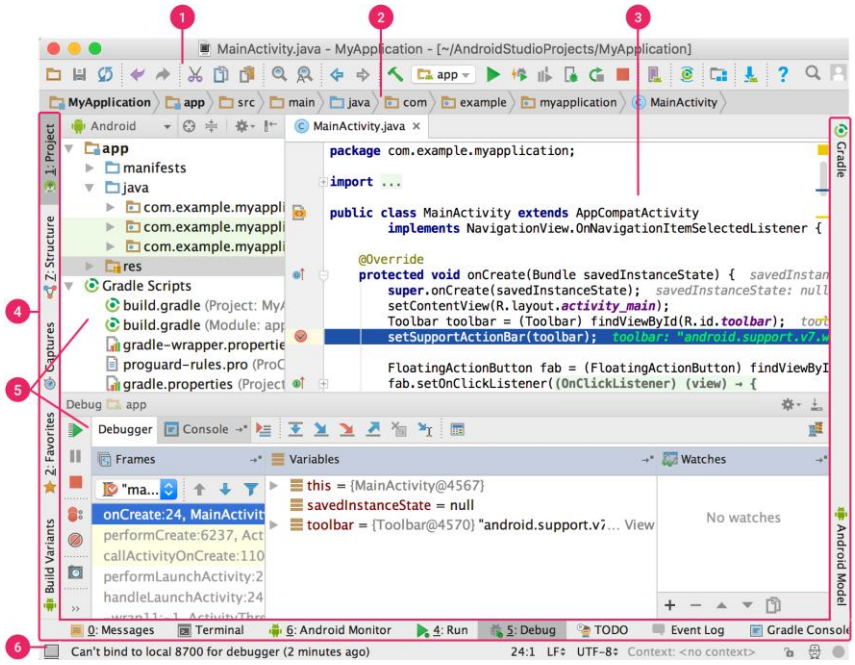
Struktur proyek Android pada disk berbeda dari representasi rata ini. Untuk melihat struktur file sebenarnya dari proyek ini, pilih Project dari menu tarik turun Project (dalam gambar 1, struktur ditampilkan sebagai Android).

Anda juga bisa menyesuaikan tampilan file proyek untuk berfokus pada aspek tertentu dari pengembangan aplikasi Anda. Misalnya, memilih tampilan Problems dari tampilan proyek Anda akan menampilkan tautan ke file sumber yang berisi kesalahan pengkodean dan sintaks yang dikenal, misalnya tag penutup elemen XML tidak ada dalam file tata letak.



Gambar 2. 5 Tampilan File Proyek Android Studio

2. Antarmuka Pengguna pada Android Studio



(sumber : https://www. developer.android.com/studio/intro/ 05-Oktober-2018).

Gambar 2. 6 Tampilan Jendela utama Android Studio

1. Bilah alat memungkinkan Anda untuk melakukan berbagai jenis tindakan, termasuk menjalankan aplikasi dan meluncurkan alat Android.
2. Bilah navigasi membantu Anda bernavigasi di antara proyek dan membuka file untuk diedit. Bilah ini memberikan tampilan struktur yang terlihat lebih ringkas dalam jendela Project.
3. Jendela editor adalah tempat Anda membuat dan memodifikasi kode. Bergantung pada jenis file saat ini, editor dapat berubah. Misalnya, ketika melihat file tata letak, editor menampilkan Layout Editor.
4. Bilah jendela alat muncul di luar jendela IDE dan berisi tombol yang memungkinkan Anda meluaskan atau menciutkan jendela alat individual.
5. Jendela alat memberi Anda akses ke tugas tertentu seperti pengelolaan proyek, penelusuran, kontrol versi, dan banyak lagi. Anda bisa meluaskan dan juga menciutkannya.
6. Bilah status menampilkan status proyek Anda dan IDE itu sendiri, serta setiap peringatan atau pesan.

2.4.1 Global Positioning System (GPS)

GPS merupakan sebuah infrastruktur satelit yang melayani penempatan posisi dari berbagai macam objek(Speikerman 2004). GPS pertama kali digunakan untuk kepentingan militer, tetapi pada tahun 1980-an pemerintah Amerikat Serikat memutuskan untuk membuat sistem *positioning* secara bebas dan dan tersedia untuk berbagai macam industry di dunia. Menurut (Roth 2004) sistem GPS terdiri atas 3 segmen yaitu [8].

1. ***User Segment*** yang terdiri atas perangkat bergerak dari pengguna (GPS *receivers).*
2. ***Space Segment*** yang terdiri atas satelit. Setiap satelit mempunyai berat anatra 1.5 sampai 2 ton dan mempunyai energy yang swatantra yang disuplai oleh sel matahari.
3. ***Control segment*** administrasi yang dibutuhkan oleh satelit sebagai koreksi dari internet data satelit (sistem waktu dan orbit).

GPS dapat melakukan perhitungan dan menentukan posisi user dan menampilkan dalam maps. Jika sudah dapat menyimpan posisi user selanjutnya GPS dapat menghitung informasi lain, seperti kecepatan, arah tuju, rute, tujuan perjalanan serta jarak tujuan. GPS ini juga dimanfaatkan untuk membangun sebuah aplikasi pencarian pet shop atau klinik hewan dimana si pengguna menetukan lokasi untuk mengetahui posisi pengguna dan lokasi pet shop atau klinik hewan.

2.4.1.1 Akurasi GPS

Posisi yang ditunjukkan oleh suatu GPS mempunyai faktor kesalahan atau juga disebut tingkat akurasi. Sebagai contoh suatu alat GPS menunjukkan titik koordinat dengan tingkat akurasi 5 meter, itu berarti posisi pengguna bisa berada dalam range radius 5 meter dari titik yang ditunjukkan tersebut. Mengapa tingkat akurasi yang terlihat bisa berubah-ubah? Kadang terlihat 10 meter, 15 meter, atau 5 meter. Ada beberapa hal yang mempengaruhi tingkat akurasi tersebut, antara lain:

1. Kesalahan Ephemeris. Terjadi jika satelit tidak dapat mentransmisikan posisinya di orbit dengan tepat.
2. Keadaan Ionosphere. Ionosphere berada pada jarak sekitar 43-50 mil di atas permukaan bumi.Walaupun GPS receiver berusaha untuk mengkoreksi/memperbaiki faktor keterlambatan yang terjadi tetap saja aktivitas tertentu dari plasma bisa menyebabkan kesalahan perhitungan.
3. Keadaan Troposphere. Troposphere adalah bagian terendah dari atmosfer sampai dengan ketinggian sekitar 11 mil dari permukaan tanah. Variasi pada temperatur, tekanan, dan kelembaban bisa menyebabkan perbedaan kecepatan penerimaan gelombang radio.
4. Kesalahan Waktu. Karena penempatan jam atom pada setiap GPS receiver tidak berjalan sebagaimana mestinya. Kesalahan waktu dari GPS receiver yang tidak presisi dapat menimbulkan ketidakakurasian.
5. Kesalahan Multipath. Terjadi karena sinyal satelit membentur permukaan keras (seperti bangunan atau tebing) sebelum mencapai GPS receiver. Hal tersebut bisa menyebabkan terjadinya delay sehingga perhitungan jarak menjadi tidak akurat.

6. Buruknya Sinyal Satelit. Keadaan langit yang terhalang akan menyebabkan GPS sulit menerima data satelit. Sebuah sinyal satelit yang pada hari tertentu diterima dengan sangat bagus belum tentu pada hari lain bisa diterima dengan kualitas yang sama walaupun user berdiri pada tempat yang sama.

2.4.1.2 Android dan GPS

Dewasa ini, teknologi berkembang dengan pesat. Dulu ponsel hanya sekedar digunakan untuk menelpon dan SMS saja. Sekarang ponsel sudah menjelma menjadi kotak kecil ajaib yang serba bisa. Salah satunya adalah ponsel dengan sistem operasi Android. Dengan standarisasi fitur dan hardware yang dimiliki, menjadikan ponsel Android ponsel canggih yang disukai banyak orang. Tidak lagi canggih karena adanya fitur MMS, radio, atau internet berkecepatan tinggi tapi juga karena ditanamkannya fitur teknologi satelit di dalamnya. Ya, perangkat GPS *receiver* yang dulu besar dan eksklusif, sekarang sudah bisa dimiliki dengan "hanya" membeli sebuah ponsel.

Dengan ponsel berteknologi satelit (GPS), banyak hal yang bisa dilakukan. Ingin melihat di mana posisi user sekarang dalam sebuah peta? Mengambil foto/video yang sudah dilengkapi dengan data koordinat? Ingin tahu jalur olahraga bersepeda yang sudah pernah user lalui? Tidak hanya itu, user juga dapat pergi ke tempat wisata tertentu dengan dipandu gambar dan suara dari sebuah ponsel! Bahkan lebih jauh lagi, GPS dapat digunakan untuk membantu memberikan peringatan dini terhadap terjadinya bencana alam. Sekarang ini banyak sekali pengembang - pengembang aplikasi untuk sistem operasi Android termasuk aplikasi-aplikasi GPS. Yang menyenangkan aplikasi-aplikasi tersebut jenisnya beragam dan jumlahnya pun banyak.

2.4.2 Location Basesd Services

Location based service merupakan layanan informasi yang dapat diakses menggunakan mobile devices, yang dilengkapi kemampuan untuk mengetahui keberadaan lokasi dari pengguna perangkat dan kemampuan memberikan informasi mengenai layanan yang tersedia berdasarkan lokasi mereka pada saat itu [9] .

Location Based Service (LBS) atau Layanan Berbasis lokasi adalah layanan informasi yang dapat diakses melalui mobile device dengan menggunakan mobile network yang dilengkapi kemampuan untuk memanfaatkan lokasi dari mobile device tersebut. LBS memberikan kemungkinan komunikasi dan interaksi dua arah. Oleh karena itu pengguna memberitahu penyedia layanan untuk memberi informasi,dengan referensi posisi pengguna tersebut. Layanan berbasis lokasi dapat digambarkan sebagaisuatu layanan yang berada pada pertemuan tiga teknologi yaitu: Geographic Information System, Internet Service, dan Mobile Devices, hal ini dapat dilihat pada gambar LBS adalah pertemuan dari tiga teknologi. Secara Garis besar jenis Layanan Berbasis Lokasi juga dapat dibagi menjadi dua, yaitu:

1. ***Pull Service***: Layanan diberikan berdasarkan permintaan dari pelanggan akan kebutuhan suatu informasi. Jenis layanan ini dapat dianalogikan seperti menggakses suatu web pada jaringan internet.
2. ***Push Service***: Layanan ini diberikan langsung oleh sevice provider tanpa menunggu permintaan dari pelanggan, tentu saja informasi yang diberikan tetap berkaitan dengan kebutuhan pelanggan.

   Teknologi ini dipakai dalam pembangunan aplikasi.

2.4.3 Clarifai

Gambar 2. 7 Logo Clarifai

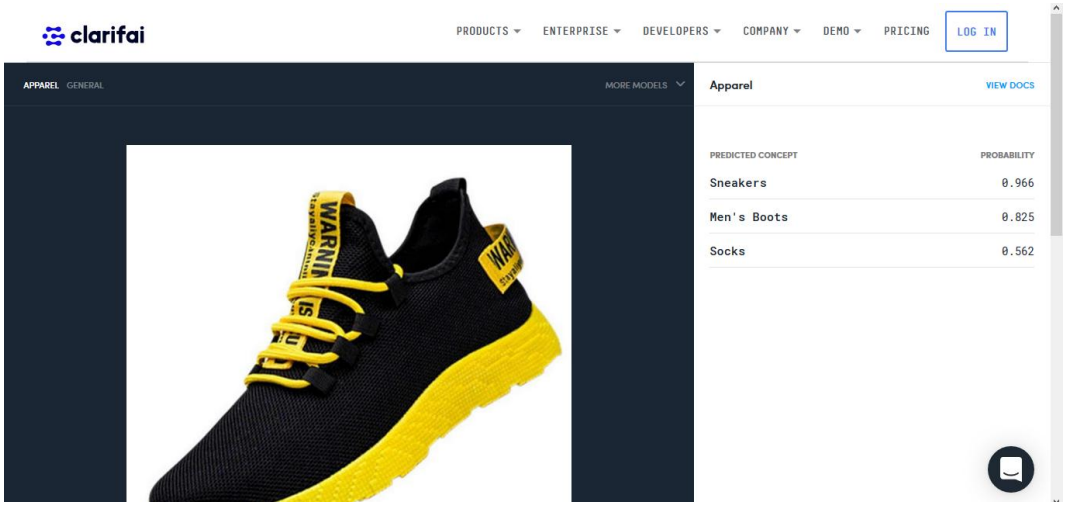
Clarifai merupakan sebuah website yang dapat memprediksi menggunakan sebuah foto atau video. Contoh sederhananya dimana anda memasukan foto atau

video dan kemudian anda akan mendapatkan hasil prediksi yang berhubungan dengan foto atau video yang anda masukan.

Berikut adalah model-model yang ada di *clarifai* :

1. Apparel Model

Apparel Model bias mengenali lebih dari 100 konsep yang berhubungan dengan *fashion* termasuk pakaian dan aksesoris. Contoh pengenalan objek menggunakan *Apparel* model pada gambar 2.8 adalah sebagai berikut :

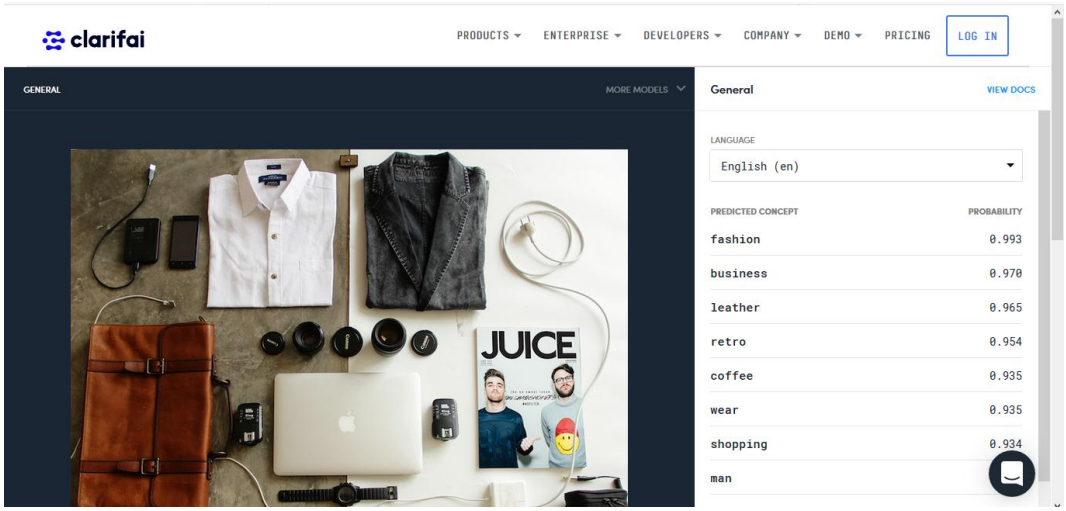


Gambar 2. 8 Contoh Apparel Model

2. General model

General Model berisi berbagai macam tag di berbagai topik yang berbeda.

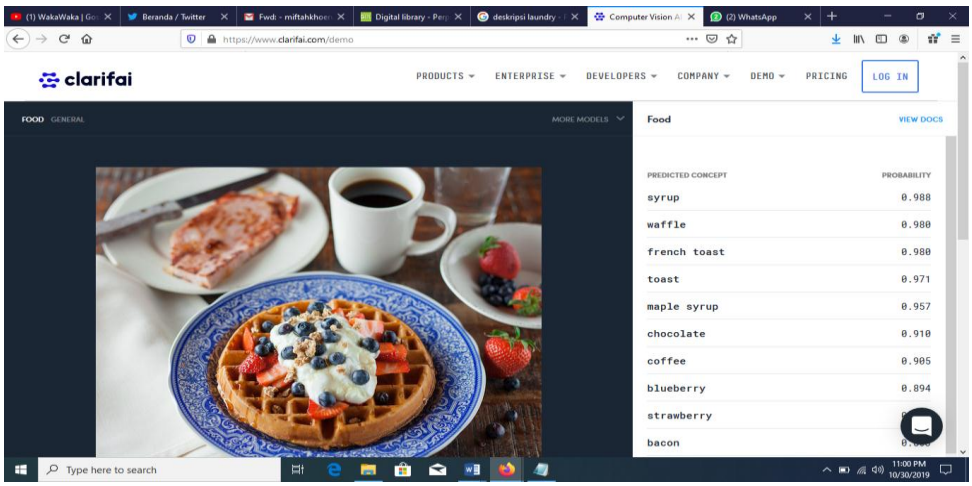
Dalam kebanyakan kasus, cukup akan mengenali objek apa yang terdapat pada gambar. Contoh pengenalan objek menggunakan *General* model seperti pada gambar 2.9 dibawah ini :



Gambar 2. 9 Contoh General Model

3. Food Model

Food modelmengenali objek dan memberikan prediksi yang terkait dengan objek. Model ini dirancang untuk mengidentifikasi item makanan tertentu dan bahan-bahan yang terlihat. Contoh pengenalan objek menggunakan *food* model pada gambar 2.10 adalah sebagai berikut :



Gambar 2. 10 Contoh Food Model

2.4.4 MidTrans

Gambar 2. 11 Logo Firebase

MidTrans adalah salah satu layanan payment gateway yang ada di Indonesia. Teknologi dari MidTrans ini bisa menjadi salah satu cara untuk memajukan bisnis Anda. Caranya adalah dengan pengembangan teknologi yang memfasilitasi pembayaran/transaksi secara online.

Hingga kini, sudah ada puluhan bank telah melakukan kerja sama dengan layanan payment gateway tersebut, yaitu termasuk Bank dari BUMN semacam Mandiri dan BNI. Pada sisi lain, telah berkembang ribuan toko online di Indonesia yang telah bekerja sama dengan MidTrans.

2.4.5 API (*Application Programing Interfaces*)

API merupakan software interface yang terdiri atas kumpulan instruksi yang disimpan dalam bentuk library dan menjelaskan bagaimana agar suatu software dapat berinteraksi dengan software lain. Penjelasan ini dapat dicontohkan dengan analogi apabila akan dibangun suatu rumah. Dengan menyewa kontraktor yang dapat menangani bagian yang berbeda, pemilik rumah dapat memberikan tugas yang perlu dilakukan oleh kontraktor tanpa harus mengetahui bagaimana cara kontraktor menyelesaikan pekerjaan tersebut. Dari analogi tersebut, rumah merupakan software yang akan dibuat, dan kontraktor merupakan API yang mengerjakan bagian tertentu dari software tersebut tanpa harus diketahui bagaimana prosedur dalam melakukan pekerjaan tersebut. Interface pada software merupakan suatu entry points yang digunakan untuk mengakses seluruh resources yang terdapat di dalam software tersebut. Dengan adanya API, maka terdapat aturan bagaimana software dapat berinteraksi dengan software lain untuk mengakses resources melalui interface yang telah tersedia.

2.5 Java Script 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).

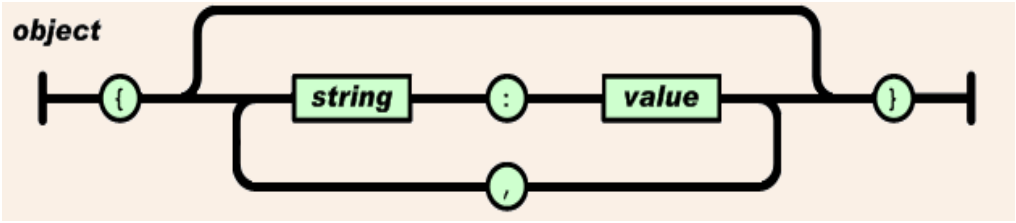
Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut :

2.5.1 Bentuk JSON

JSON menggunakan bentuk sebagai berikut:

2. Objek

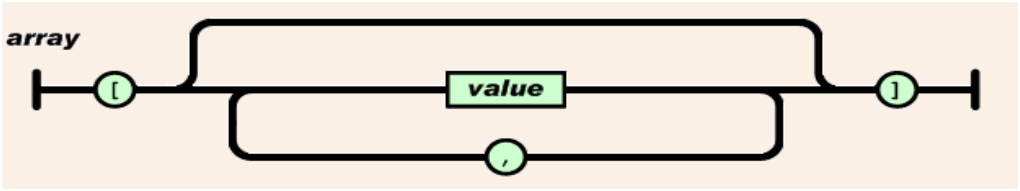
Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurang kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



Gambar 2. 12 Objek JSON

3. Larik

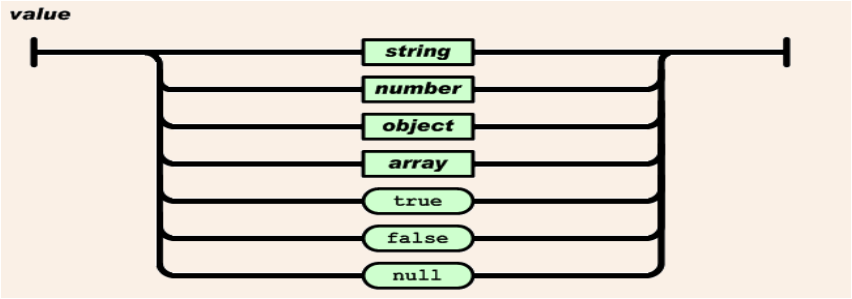
Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).



Gambar 2. 13 Array JSON

4. Value

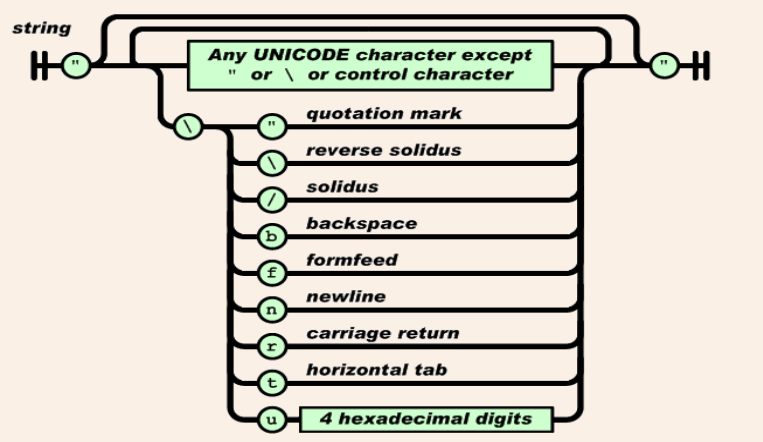
Nilai (value) dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau sebuah objek atau sebuah larik. Struktur- struktur tersebut dapat disusun bertingkat.



Gambar 2. 14 Value JSON

5. String

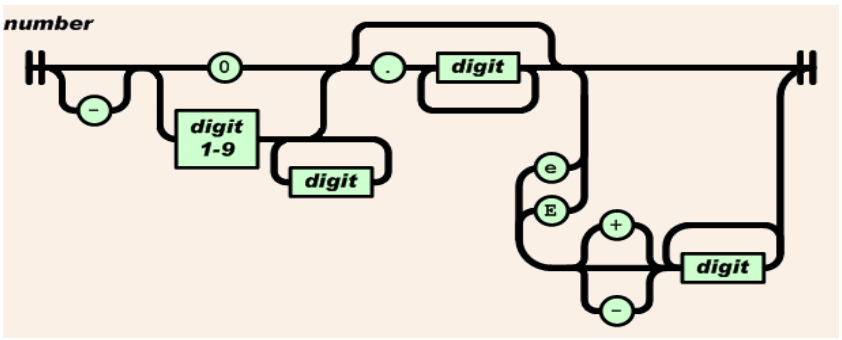
String adalah kumpulan dari nol atau lebih karakter Unicode, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan backslash escapes "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java.



Gambar 2. 15 String JSON

6. Angka

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



Gambar 2. 16 Number JSON

7. Number

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

2.6 Object Oriented Analysis Design

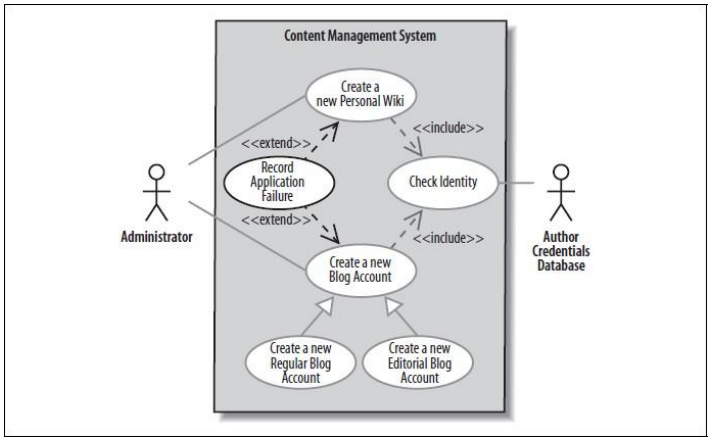
Analisis dan desain berorientasi objek adalah cara baru dalam memikirkan satu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Tujuan dari analisis berorientasi objek adalah untuk mengembangkan model yang menggambarkan perangkat lunak komputer karena bekerja untuk memenuhi seperangkat persyaratan yang ditentukan *user*. *Tools* yang dapat digunakan pada

pendekatan analisis pengembangan sistem secara objek dapat menggunakan UML [10].

Unified Modelling Language (UML) adalah sebuah bahasa yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak. UML menggunakan *class* dan *operation object* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa bahasa berorientasi objek. Dalam membangun block UML ada 3 hal yang harus diperhatikan, yaitu *object* (memodelkan konsep), *relationship* (mengkoneksikan *object*), dan *diagram* (*grouping* yang saling mengkoneksikan antara *object* dan *relationship* [10].

2.6.1 Use Case Diagram

Use Case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di- *include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang serupa. Sebuah *use case* juga dapat meng- *extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain. Dasar untuk menentukan sebuah *use case* adalah *use case* merupakan sesuatu yang menyediakan beberapa hasil terukur kepada pengguna atau sistem eksternal. *Use case* harus memiliki kriteria berhasil / gagal [11]. Berikut adalah contoh dari *use case* diagram pada gambar 2.17 :



Gambar 2. 17 Contoh Use Case Diagram

2.6..1.1 Use Case Scenario

Sebuah diagram yang menunjukkan *use case* dan aktor mungkin menjadi titik awal yang bagus, tetapi tidak memberikan detail yang cukup bagi desainer sistem untuk benar-benar memahami persis bagaimana sistem dapat terpenuhi. Cara terbaik untuk mengungkapkan informasi penting ini adalah dalam bentuk penggunaan *use case scenario* berbasis teks per *use case*-nya. Berikut adalah dasar format penulisan *use case scenario* pada table 2.2 [11] :

Tabel 2. 1 Dasar Pembangunan Use Case Scenario

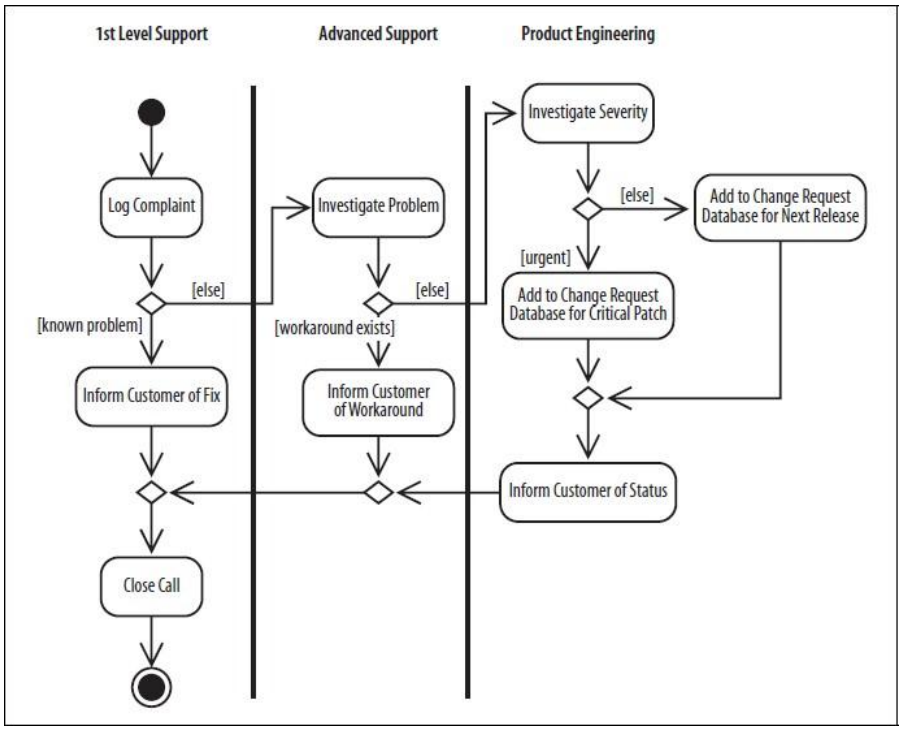
Use Case Name	Berisi nama dari *Use case* yang akan digunakan	
Goal In Context	Menjelaskan apa yang aktor coba untuk dapatkan dari *Use case*	
Description	Menjelaskan gambaran dari *Use case*	
Related Use Case	Daftar *Use case* yang berhubungan dengan *Use case* tersebut	
Successful End Condition	Kondisi *Use case* jika berhasil	
Failed End Condition	Kondisi *Use case* jika gagal	
Actors	Daftar aktor yang dapat mengakses *Use case*	
Trigger	Aktifitas yang dilakukan untuk mengawali *Use case*	
Main Flow	Step	Action
	1	Deskripsi urutan aksi dari aktifitas *Use case*
Extension	Step	Branching Action
	1.1	Deskripsi urutan aksi lain selain urutan aksi utama

2.6.2 Activity Diagram

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

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

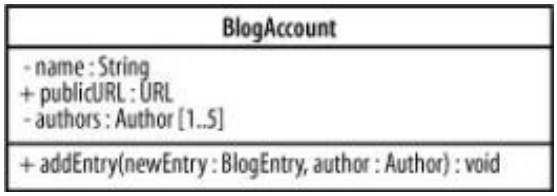
Standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behaviour* pada kondisi tertentu, untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis *horizontal* atau *vertikal*



Gambar 2. 18 Contoh Activity Diagram

2.6.3 Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) satu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class* diagram menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti cont*a*inment, pewarisan, asosiasi, dan lain-lain [11].



Gambar 2. 19 Contoh Class Diagram

Class memiliki tiga area pokok:

1. Nama dan stereotype
2. Atribut
3. Metoda

Atribut dan metoda dari *class* dapat memiliki salah satu sifat berikut:

1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
2. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.
3. *Public*, dapat dipanggil oleh siapa saja.

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class*. Sesuai dengan perkembangan *class* model, *class* dapat dikelompokkan menjadi *package*. *Class* memiliki tipe-tipe *relationship*, diantaranya [12] :

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus

mengetahui eksistensi *class* lain.

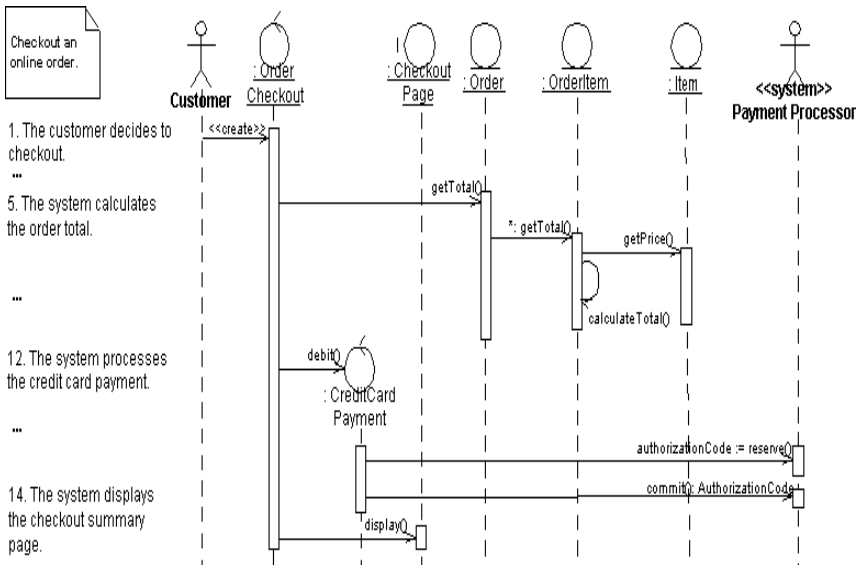
2. Agregasi, yaitu hubungan yang menyatakan bagian terdiri atas dimana ketika satu *class* di *share* atau direferensikan kepada objek yang ada di *class* lain.
2. Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
3. Komposisi, yaitu jenis relasi *class* diagram yang kuat dimana jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka *class* tersebut memiliki relasi Composition terhadap *class* tempat dia bergantung tersebut. Sebuah relationship composition digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.
4. Depedensi, salah satu jenis relasi class diagram yang lemah dimana objek dalam suatu *class* akan bekerja sangat singkat dengan objek yang ada pada *class* lain.

2.6.4 Squence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem berupa *message* yang digambarkan terhadap waktu. *Sequence* diagram terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence* diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan *Output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan.

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*.

Activation bar menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*. Untuk objek-objek yang memiliki



Gambar 2. 20 Contoh Sequnce Diagram

sifat khusus, standar UML mendefinisikan *icon* khusus untuk objek *boundary*, *controller* dan *persistent entity* [11].