BAB 2
LANDASAN TEORI

2.1 *Factory Outlet*

Factory Outlet adalah toko pengecer yang menyediakan produk berbeda dari yang lain, menyediakan pakaian yang sudah *outdate* atau pakaian stok lama yang dijual langsung ke konsumen.

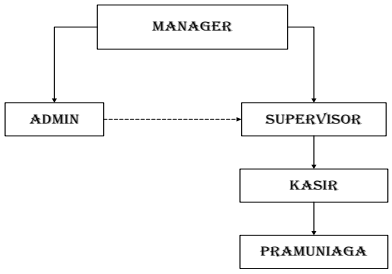
Berdasarkan uraian diatas dapat disimpulkan bahwa *Factory Outlet* adalah sebuah toko yang menjual produk pakaian yang sudah *outdate* dengan produk yang berkualitas yang dapat ditawarkan kepada konsumen dengan harga miring / terjangkau dan dijual langsung kepada konsumen [6].

2.2 Struktur Organisasi *Factory Outlet Black and White*

Struktur organisasi merupakan kerangka kerja yang mengambarkan hubungan kerja, wewenang dan tanggung jawab setiap tingkat yang ada dalam organissasi tersebut, untuk melaksanakan kegiatan ke arah tercapainya tujuan dari organisasi yang telah ditetapkan, sehingga tercapainya kerjasama dan koordinasi usaha diantara setiap unit organisasi dalam mengambil tindakan dan mencapai tujuan. Mempunyai struktur organisasi yang baik dan merupakan suatu yang penting bagi perusahaan, karena dengan struktur organisasi yang baik dan tepat dapat membantu kelancaran jalannya usaha yang baik dan teratur.

Dengan organisasi yang baik dan tepat setiap karyawan akan ditempatkan pada posisi yang sesuai dengan minat dan keahlian dari masing – masing staf dan karyawannya. Agar tercapai efisiensi dan efektivitas bagi setiap karyawan dalam bekerja, perusahaan perlu menyusun dan menetapkan bagan organisasi yang disertai uraian tugas dan tanggung jawab di lingkungan perusahaan.

Berikut gambar 2.1 struktur organisasi *Factory Outlet Black and White*



Gambar 2.1 Struktur Organisasi Bidang Operasional *Factory Outlet Black and White*

Berdasarkan bagan struktur orgaisasi *Factory Outlet Black and White* kota Tangerang secara garis besar pada gambar 2.1.

Dalam membantu kelancaran kerja pimpinan, pemimpin mendapat bantuan dari para staf. Garis wewenang tetap berada pada pimpian, sedangkan staf hanya memiliki wewenang staf saja.

2.2.1 Deskripsi Jabatan

Adapun tugas dari masing-masing departemen dari struktur organisasi diatas diantaranya:

1. ***Manager***
   a. Menyusun dan menetapkan rencana kerja secara periodik dalam upaya pencapaian target
   b. Menyusun strategi penjualan
   c. Menerima semua laporan kegiatan operasional
2. ***Supervisor Outlet***
   a. Mengkoordinator semua kegiatan operasional yang berada di bawah tanggungjawabnya, memback up admin,
   b. Bertanggung jawab atas permasalahan yang ada di outlet baik dari pihak pelanggan maupun karyawan staf operasional.
3. **Admin**
   a. Status mengolah data barang

4. **Kasir**
    a. Menerima pembayaran dari produk yang terjual
    b. Membuat laporan penjualan setiap hari
    c. Dapat menyelesaikan permasalahan jika pelanggan ada keluhan, dan memberitahukan keluhan pelanggan kepada supervisor outlet
5. **Pramuniaga**
    a. Melayani pelanggan pada saat mencari barang
    b. Menjaga dan membina hubungan baik dengan customer
    c. Menjaga kebersihan outlet
    d. Membantu mendisplay barang yang tersedia di outlet

2.3 Aplikasi

Aplikasi adalah penggunaan dalam suatu komputer, instruksi (*Instruction*) atau pernyataan (*Statement*) yang disusun sedemikian rupa sehingga komputer dapat memproses input menjadi output.

Aplikasi merupakan perangkat lunak yang berada didalam sistem dan digunakan untuk mengolah data untuk menjadi suatu informasi yang berguna bagi penggunanya. Aplikasi juga dapat diartikan sebagai suatu sistem program yang siap untuk digunakan yang dibuat untuk fungsi – fungsi tertentu dengan harapan dapat membantu memudahkan penggunanya [7].

2.4 Data dan Informasi

2.4.1 Data

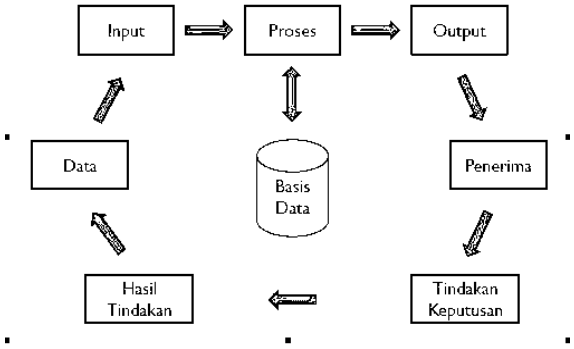
Data merupakan sesuatu yang tidak mempunyai arti bagi penggunanya dan masih memerlukan adanya suatu pengolahan terhadap data tersebut. Data bisa berbentuk suatu kejadian, benda, aktifitas, gambar, huruf dan sebagainya. Data adalah bahan mentah untuk menjadi suatu informasi [8].

2.4.2 Informasi

Informasi merupakan data yang telah diproses atau diolah sehingga memiliki nilai tertentu bagi penggunanya. Keberadaan data sangatlah penting karena data merupakan bahan utama untuk didapatkannya suatu informasi yang dibutuhkan [8].

2.4.2.1 Siklus Informasi

Data yang masih merupakan bahan mentah yang harus diolah, agar dapat menghasilkan suatu informasi melalui suatu model. Model yang digunakan untuk mengolah data tersebut disebut model pengolahan data atau dikenal dengan siklus pengolahan data (*siklus informasi)*



Gambar 2.2 siklus informasi [8].

2.5 *Global Positioning System* (GPS)

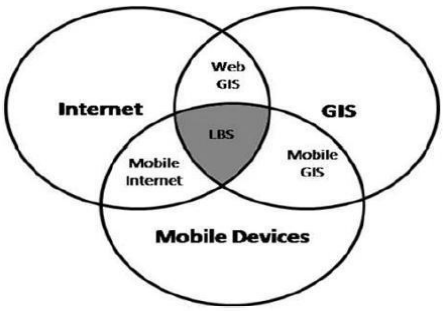
Global Positioning System (GPS) merupakan sistem navigasi menggunakan satelit dalam penggunaannya. GPS terdiri dari tiga bagian yaitu sistem kontrol, satelit dan pengguna [9].

alat navigasi yang digunakan seperti perangkat mobile yang kini sudah memiliki fitur GPS di dalamnya. GPS biasanya di gunakan untuk menunjukan suatu lokasi yang berada di permukaan bumi dengan tingkat akurasi yang cukup baik yaitu kurang dari 10 meter selama tidak ada benda padat yang menghambat sinyal untuk mendapatkan lokasi pengguna.

2.6 *Location Based Service*

Location Based Service (LBS) atau layanan berbasis lokasi adalah layanan informasi yang dapat diakses melalui *mobile device* dengan menggunakan *mobile network,* yang dilengkapi dengan kemampuan untuk memanfaatkan lokasi dari *mobile device* tersebut. LBS memungkinkan untuk melakukan komunikasi dan interaksi dua arah.

LBS ini dapat digambarkan sebagai suatu layanan yang berada pada pertemuan tiga teknologi yaitu *Geographic Information System* (GIS), Internet *Service* dan *mobile device,* seperti pada gambar 2.3 [9].



Gambar 2.3 Teknologi *Location Based Service* [9].

2.6.1 Metode Penentu Lokasi

LBS merupakan layanan yang mampu bereaksi aktif pada suatu perubahan posisi sehingga dapat mendeteksi letak objek dan memberikan layanan sesuai dengan letak objek yang telah diketahui tersebut. Pada teknologi LBS berbasis jaringan seluler, penentuan posisi pada sebuah peralatan komunikasi ditentukan berdasarkan posisi relatif terhadap lokasi BTS (*Base Transceiver Station*).

Dalam menentukan posisi dari sebuah perangkat *mobile* yang sedang aktif, secara umum terdapat tiga tingkat metode yang digunakan saat ini, diantaranya adalah sebagai berikut :

1. Metode *Basic Positioning*

   Metode *Basic Positioning* yang berbasis pada *Cell Identification* (Cell ID) Penentuan posisi didasarkan pada daerah geografis yang tercakup oleh sebuah *Cell* berhubungan dengan daerah cakupan dari sinyal radio. Ketika sebuah *handphone* terhubung secara aktif dengan sebuah *base station*, berarti *handphone* tersebut diasumsikan berada dalam *cell* dari *base station* tersebut. Untuk mengukur jarak dan arah *handset* dari *base station* tidak dapat diketahi dengan pasti, oleh karena itu untuk lebih meningkatkan lagi akurasi hasil pencarian pada metode ini seringkali dikombinasikan dengan metode lain seperti:

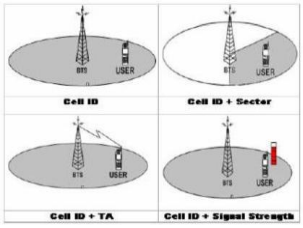
   a. *Timing Advance* (TA)

      Dengan menggunakan TA ini metode Cell ID akan ditambahkan sebuah fungsionalitas untuk menghitung *Round Trip Time* (RTT), yaitu waktu transmisi sebuah *frame* (dari *base station* ke *handphone*) dan waktu sebuah *frame* (dari *handphone* ke *station*).

Dengan tambahan ini jarak antara *handphone* dan *base station* dapat ditentukan dengan keakuratan 50 Meter.

b. *Network Measurement Report (NMR)*

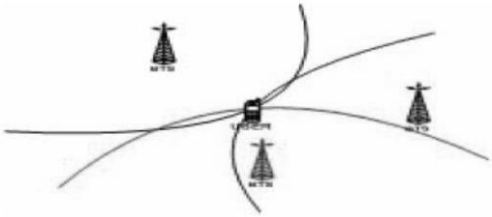
Berdasarkan pada besar kecilnya sinyal (*Receiver Signal Strength*) yang diterima *handphone* yang ada di suatu "*sector cell*", maka posisi itu dapat ditentukan dengan lebih akurat.



Gambar 2.4 *Network Measurement Report* [9].

c. *Metode Enhanced Posittioning*

Metode ini menggunakan pendekatan *Observe Time Difference* (OTD) dengan jaringan GSM yang sering digunakan seperti *Enhanced* - OTD (E-OTD). E-OTD adalah metode pencarian posisi yang berdasarkan pada waktu.



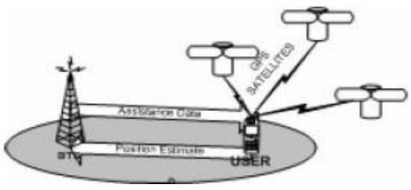
Gambar 2.5 Enhanced Positioning [9].

2. *Metode Advance Positioning*

Metode ini menggunakan *teknologi Assisted-Global Positioning System* (A-GPS). A-GPS ini merupakan metode yang berbasis pada waktu.

Pada metode ini, akan dilakukan pengukuran waktu tiba dari sebuah sinyal yang dikirim dari tiga buah satelit GPS. Hal ini berarti handset harus memiliki fasilitas untuk mengakses GPS.

A-GPS juga menghasilkan akurasi secara *vertical* dan estimasi jarak yang baik. Akurasinya pun kurang dari 10 m.



Gambar 2.6 Advance Positioning [9].

2.6.2 Komponen *Location Based Service*

Dalam menggunakan *Location Based Service* atau layanan berbasis lokasi terdapat beberapa elemen yang diperlukan, antara lain :

1. *Mobile Devices*

   Sebuah alat yang digunakan untuk meminta informasi yang dibutuhkan seperti perangkat *mobile smartphone* yang memiliki fasilitas navigasi.

2. *Communication Network*

   Jaringan selular yang mengirimkan data pengguna dan permintaan layanan.

3. *Positioning Component*

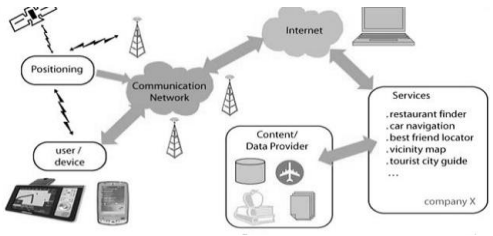
   Untuk dapat mengolah layanan harus menentukan lokasi pengguna. Posisi pengguna ini dapat diperoleh menggunakan jaringan komunikasi atau dengan menggunakan *Global Positioning System* (GPS).

4. *Service and Application Provider*

   Penyedia layanan pengguna selular yang bertanggung jawab untuk memproses suatu layanan.

5. *Data and Content Provider*

   Penyedia layanan informasi data yang dapat diminta oleh pengguna.



Gambar 2.7 Contoh Komponen dasar LBS [9].

2.6.3 *Latitude* (Garis Lintang)

Latitude adalah garis yang melintang di antara kutub utara dan kutub selatan yang menghubungkan sisi bagian bumi dari timur dan barat. Garis lintang memiliki posisi membentangi bumi seperti halnya garis khatulistiwa tetapi dengan kondisi nilai tertentu [10].

2.6.4 *Longitude* (Garis Bujur)

Longitude adalah garis bujur yang menghubungkan antara sisi utara dan sisi selatan permukaan bumi. Garis bujur digunakan untuk mengukur sisi barat dan timur koordinat suatu titik di belahan bumi. *Longitude* dibedakan menjadi dua wilayah yaitu bujur timur dan bujur barat dimana nilai koordinat yang berada di timur selalu negatif dan nilai koordinat dibagian barat selalu positif [10].

2.6.4.1 Rumus *Harversine* untuk menentukan *Latitude* dan *Longitude*.

Rumus *Haversine* Formula adalah persamaan penting dalam bidang navigasi, formula ini dapat memberikan jarak lingkaran besar antara dua titik pada sebuah bentuk bola dari garis lintang dan garis bujur.

$$Haversin(\frac{d}{r}) = haversin(\phi_2 - \phi_1)+cos(\phi_1)cos(\phi_2)haversin(\lambda_2 - \lambda_1)$$

dimana, $haversine(\phi)= sin^2(\frac{\phi}{2})= 1-\frac{cos(\phi)}{2}$ .....(1)

keterangan: d = jarak antara dua titik, r = jari-jari bola, ϕ_1, ϕ_2 = garis lintang titik 1 dan garis lintang titik 2, $\lambda_1 - \lambda_2$ = bujur titik 1 dan bujur titik 2

Gambar 2.8 Contoh Perhitungan Jarak menggunakan rumus *Harversine* untuk menentukan *Latitude* dan *Longitude*.

Rumus dari formula hervasine ini diperlukan untuk mencari titik terdekat lokasi Factory Outlet Black and white dari lokasi user berada, sehingga jika rumus hervasine formula ini diterjemahkan dalam SQL menjadi sebagai berikut :

*$query="SELECTid,(3959*acos(cos(radians($a))*cos(radians(lat))*cos(radians (lng)-radians($b))+sin(radians($a))*sin(radians(lat)))) AS distance FROM markes HAVING distance < 25 ORDER BY distance LIMIT 0,20";*

Keterangan :

$a = Letitude User

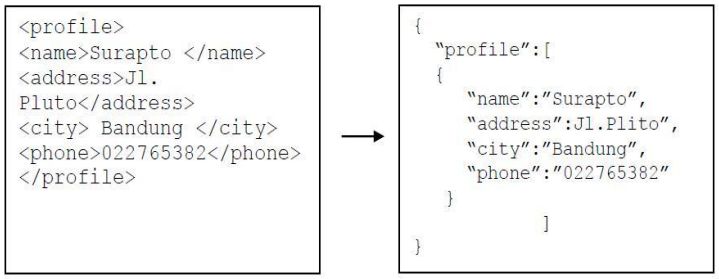
$b = Longitude User

Angka 3959 jika pencarian lokasi dalam satu mil, sedangkan 6371 bila pencarian dalam kilometer. Setelah dilakukan perhitungan menggunakan rumus hervasine, maka sistem dapat memperoleh jarak terdekat antara *Factory Outlet Black and White* dan keberadaan user [10].

2.7 *JSON (Javascript Object Notation)*

JSON atau *Javascript Object Notation* adalah format untuk pertukaran data

yang lebih ringan,mudah dibaca dan ditulis oleh manusia serta mudah mengerti dan dapat diterjemahkan oleh komputer. JSON lebih unggul dari pada format pertukaran data yang lain seperti XML. Mulai dari kecepatan, penulisan yang lebih mudah dan *coding parsing* yang lebih ringkas dan sederhana. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman lain / apapun karena menggunakan bahasa pemrograman yang umum digunakan yaitu keluarga C seperti C, C#, C++, Java, Javascript, Perl, Python dan lain – lain [11]. Berikut adalah contoh konversi data XML ke dalam format JSON :



Gambar 2.9 Contoh Script JSON [11].

2.8 Google *Maps*

Google *Maps* adalah sebuah layanan gratis dari Google tentang berbagai macam teknologi pemetaan. Google *Maps* menampilkan peta dunia secara digital sehingga lebih memudahkan ketika ingin mengetahui suatu lokasi tertentu. Google *Maps* menyediakan beberapa keunggulan seperti peta jalan, menunjukan rute untuk suatu perjalanan yang diinginkan baik itu melalui jalur mobil sampai pejalan kaki [12].

2.8.1 Google *Maps* API

Google *Maps* API adalah sebuah layanan yang memungkinkan Google *Maps* agar dapat diintegrasikan ke dalam aplikasi yang memang membutuhkan fitur dari Google *Maps* ini. Google *Maps* API adalah sebuah *library Javascript*. Dengan menggunakan Google *Maps* API ini dapat memudahkan pengembang dalam membuat aplikasi yang membutuhkan teknologi peta di dalamnya dan tidak perlu memikirkan cara membuatnya secara manual karena dengan menggunakan Google *Maps* API ini sudah semua yang dibutuhkan terkait teknologi peta digital sudah cukup lengkap dan dapat dimanfaatkan untuk aplikasi yang dibutuhkan

Untuk penggunaan *Google Maps API* pada perangkat *mobile* misalnya android *Google Maps API* yang bisa digunakan saat ini telah mencapai versi 2. Perkembangan pada *Google Maps Versi* 2 ini memiliki fitur – fitur yang lengkap dari versi sebelumnya salah satunya dapat menampilkan peta dengan sudut yang diingkan dan berupa 3D *view*. Selain itu pada Google *Maps API* terdapat beberapa pilihan model peta yang ditampilkan seperti *Roadmap*, *Satellite*, *Terrain* dan *Hybrid* [12].

2.9 *Object Oriented Programming* (OOP)

Object-Oriented Programming (OOP) adalah sebuah pendekatan untuk pengembangan atau *development* suatu *software* dimana dalam struktur *software* tersebut didasarkan kepada interaksi *object* dalam penyelesaian suatu proses atau tugas. Interaksi tersebut mengambil form dari pesan-pesan dan mengirimkannya kembali antar *object* tersebut. *Object* akan merespon pesan tersebut menjadi sebuah tindakan atau metode [13].

Adapun konsep pada pemrograman berorientasi objek ini yaitu:

1. *Object*

   Object adalah sebuah struktur yang menggabungkan data dan prosedur untuk bekerja bersama-sama.

2. *Abstraction*

   Ketika membangun *objects* dalam aplikasi OOP, adalah penting untuk menggabungkan konsep *abstraction* ini. Jika membangun aplikasi shipping, maka harus membangun *object* produk dengan atribut seperti ukuran dan berat. Warna adalah contoh informasi yang tidak ada hubungannya dan harus dibuang. Tetapi ketika akan membangun *order-entry application*, warna menjadi penting dan harus termasuk atribut *object* produk.

3. *Enkapsulasi*

   Sebuah proses dimana tidak ada akses langsung ke data yang diberikan, bahkan hidden. Jika ingin mendapat data, harus berinteraksi dengan *object* yang bertanggung jawab atas data tersebut.

4. *Polymorphisms*

   Kemampuan dua buah *object* yang berbeda untuk merespon pesan permintaan yang sama dalam suatu cara yang unik. Contohnya, dapat mengirim pesan *print* ke *object* printer yang akan mencetak pada printer, dan juga dapat mengirim pesan yang sama ke *object screen* yang akan menuliskan pada *screen* monitor. Dalam OOP, diterapkan tipe *polymorphism* melalui proses yang disebut *overloading*. Dapat dilakukan dalam implementasi metode yang berbeda pada sebuah *object* yang mempunyai nama yang sama.

5. *Inheritance*

   Penggunaan *inheritance* dalam OOP untuk mengklasifikasikan *objects* dalam program sesuai karakteristik umum dan fungsinya. Hal ini akan membuat pekerjaan bersama *object* lebih mudah dan lebih intuitif. Hal ini juga membuat programming lebih mudah karena memungkinkan untuk mengkombinasikan karakteristik umum kedalam *object parent* dan mewariskan karakteristik ini ke *child object*.

2.10 *Entitas Relation Diagram (ERD)*

ERD merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. ERD untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan simbol.

Menurut salah satu para ahli, Brady dan Loonam *Entity Relationship diagram (ERD)* merupakan teknik yang digunakan untuk memodelkan kebutuhan data dari suatu organisasi, biasanya oleh System Analys dalam tahap analisis persyaratan proyek pengembangan system [13].

Sementara seolah-olah teknik diagram atau alat peraga memberikan dasar untuk desain database relasional yang dapat mendasari sistem informasi yang dikembangkan. ERD bersama-sama dengan detail pendukung merupakan model data yang pada gilirannya digunakan sebagai spesifikasi untuk database. Entitas adalah objek dalam dunia nyata yang dapat dibedakan dengan objek lain, sebagai

contoh mahasiswa,dosen,departemen. Entitias terdiri atas beberapa atribut sebagai contoh atribut dari entitas mahasiswa adalah nim, nama, alamat, email, dll. Atribut nim merupakan unik untuk mengidentifikasikan / membedakan mahasiswa yg satu dengan yg lainnya. Pada setiap entitas harus memiliki 1 atribut unik atau yang disebut dengan primary key.

Atribut adalah Setiap entitas pasti mempunyai elemen yang disebut atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasikan isi elemen satu dengan yang lain [13].

Relasi adalah hubungan antara beberapa entitas. sebagai contoh relasi antar mahaiswa dengan mata kuliah dimana setiap mahasiswa bisa mengambil beberapa mata kuliah dan setiap mata kuliah bisa diambil oleh lebih dari 1 mahasiswa. relasi tersebut memiliki hubungan banyak ke banyak. Berikut adalah contoh ERD Kardinalitas menyatakan jumlah himpunan relasi antar entitias [13].

Pemetaan kardiniliat terdiri dari :

a. one-to-one : sebuah entitas pada A berhubungan dengan entitas B paling banyak 1 contoh diatas relasi pegawai dan departemen dimana setiap pegawai hanya bekerja pada 1 departemen
b. one-to-many : sebuah entitas pada A berhubungan dengan entitas B lebih dari satu contoh diatas adalah 1 depertemen memiliki banyak pegawai
c. many-to-many : sebuah entitas pada A berhubungan dengan entitas B lebih dari satu dan B berhubungan dengan A lebih dari satu jugan contoh diatas adalah relasi mahasiswa dengan mata kuliah.

   Berikut adalah metode/tahap untuk membuat ERD :

   a. Menentukan Entitas
   b. Menentukan Relasi
   c. Menggambar ERD sementara
   d. Mengisi Kardinalitas
   e. Menentukan Kunci Utama
   f. Menggambar ERD berdasar Key

g. Menentukan Atribut
h. Memetakan Atribut
i. Menggambar ERD dengan Atribut

2.11 *Unified Modeling Language* (UML)

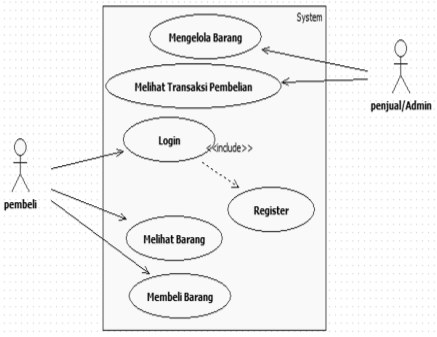
Unified Modelling language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak khususnya dalam sistem yang dibangun menggunakan pemrograman berorientasi objek (OO).

Definisi ini merupakan definisi yang sederhana. Pada kenyataanya, pendapat orang-orang tentang UML berbeda satu sama lain. Hal ini dikarenakan oleh sejarahnya sendiri dan oleh perbedaan persepsi tentang apa yang membuat sebuah proses rancang-bangun perangkat lunak efektif [13].

2.11.1 *Use Case* Diagram

Use case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. Sebuah *use case* juga menggambarkan cara sistem berperilaku untuk memenuhi kebutuhan. Use *case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang sistem tersebut digunakan. Dalam bahasan *Use case*, para pengguna disebut sebagai aktor. Aktor merupakan sebuah peran yang dimainkan seorang pengguna dalam kaitannya dengan sistem. Aktor tidak harus manusia, jika sebuah sistem melakukan sebuah layanan untuk sistem komputer lain, sistem lain tersebut merupakan aktor.

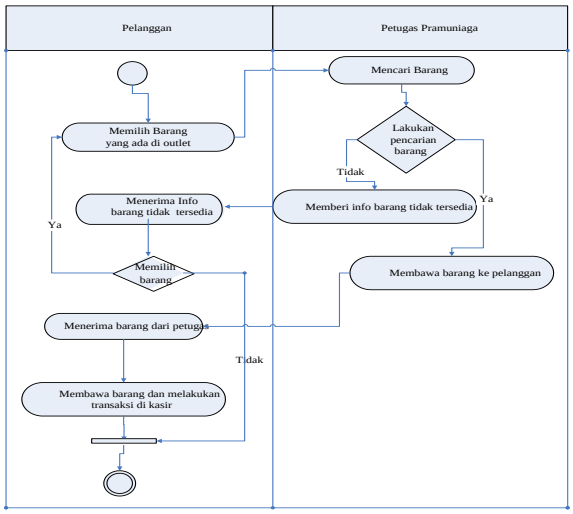
Berikut contoh bentuk *Use Case Diagram* yang mengkomunikasikan antara aktor *Use Case* di dalam sebuah sistem :



Gambar 2.11 Contoh Use Case Diagram [13].

2.11.2 *Activity Diagram*

Activity diagram adalah teknik untuk menggambrakan logika prosedural, proses bisnis, dan jalur kerja. Dalam beberapa hal, diagram ini memainkan peran mirip sebuah diagram alir, tetapi perbedaan prinsip antar diagram ini dan notasi diagram alir yang bersifat pararel. *Activity diagram* memungkinkan siapapun yang melakukan proses untuk memilih urutan dalam melakukannya. Contoh sederhana penggunaan *activity diagram* dapat dilihat pada gambar 2.12 :

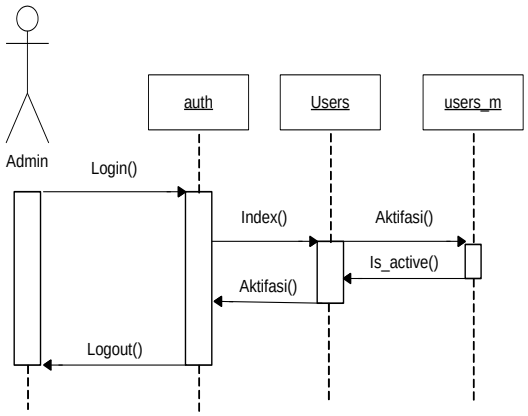


Gambar 2.12 Contoh Activity Diagram [13].

2.11.3 *Sequence Diagram*

Sequence Diagram menunjukan interaksi antar diagram yang berkolaborasi dalam beberapa sifat. Sebuah *sequence* diagram secara khusus menjabarkan sifat skenario tunggal. Diagram tersebut menunjukan sejumlah objek contoh dan pesan-pesan yang melewati objek - objek ini di dalam *Use case*.

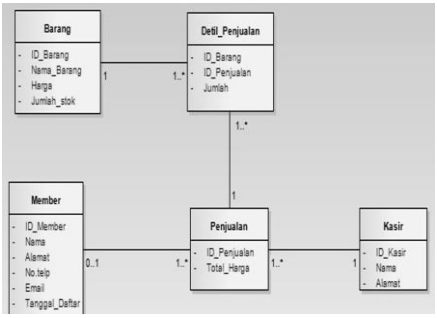
Berikut contoh bentuk dari *sequence* diagram :



Gambar 2.13 Contoh Sequence Diagram [13].

2.11.4 *Class* Diagram

Class diagram dapat dideskripsikan sebagia jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antara mereka. *Class diagram* juga menunjukan properti dan operasi sebuah class dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. UML menggunakan istilah fitur sebagai istilah umum yang meliputi properti dan operasi sebuah *class*.



Gambar 2.14 Contoh Class Diagram [13].

2.12 *Database*

2.12.1 Pengertian *Database*

Database terdiri dari dua penggalan kata yaitu "data" dan "*base*" yang artinya berbasiskan pada data. Secara konseptual database memiliki arti sebuah koleksi atau kumpulan data yang saling berhubungan, disusun menurut urutan tertentu secara logis sehingga menghasilkan informasi [14].

Database adalah suatu kumpulan data yang terhubung (*interrelated data*) yang disimpan secara bersama-sama pada suatu media, tidak terulang (*controlled*

redundancy) dan dikontrol dengan cara tertentu sehingga mudah digunakan atau ditampilkan kembali, dapat digunakan oleh satu atau lebih program aplikasi secara optimal, data disimpan tanpa ketergantungan kepada program yang akan menggunakannya, dapat ditambah, diambil, dimodifikasi dengan mudah dan terkontrol. Menurut Gordon C. Everest, *database* adalah koleksi atau kumpulan data yang mekanis, terbagi / *shared*, terdefinisi secara formal dan dikontrol terpusat pada organisasi. Menurut C.J. Date, database adalah koleksi "data operasional" yang tersimpan dan dipakai oleh sistem aplikasi dari suatu organisasi. Terdapat data input adalah data yang masuk dari luar sistem, data output adalah data yang dihasilkan sistem, dan data operasional adalah data yang tersimpan pada sistem. Berdasarkan pengertian *database* dari beberapa tokoh di atas dapat disimpulkan bahwa *database* adalah suatu kumpulan data.

Database merupakan salah satu komponen yang penting di sistem informasi, karena berfungsi sebagai basis penyedia informasi bagi para pemakainya. Penerapan database dalam sistem informasi disebut dengan database sistem. Sistem basis data (*database system*) ini adalah suatu sistem informasi yang mengintegrasikan kumpulan dari data yang saling berhubungan satu dengan lainnya dan membuatnya tersedia untuk beberapa aplikasi yang bermacam-macam didalam suatu organisasi. Dalam database hal ini tidak boleh dan tidak dapat terjadi karena antara file yang satu dengan file yang lain saling berhubungan. Jika satu data yang sama Anda ubah, data tersebut di file yang lain akan berubah juga. Sehingga tingkat keakuratan/kebenaran data sangat tinggi. Penamaan *database* biasanya disesuiakan dengan isinya, misal database perpustakaan, database perikanan, statistik dan sebagainya. JAMES,F.C *at al*.

Sistem database adalah sekumpulan database yang dapat dipakai secara bersama-sama, personal-personal yang merancang dan mengelola database, teknik-teknik untuk merancang dan mengelola database serta piranti untuk mendukungnya [14].

2.12.2 Kriteria Database.

Kriteria database meliputi :

a. Bersifat data *oriented*, bukan program *oriented*,
b. Dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah databasenya,
c. Dapat berkembang dengan mudah baik volume maupun strukturnya,
d. Dapat memenuhi kebutuhan sistim-sistim baru dengan mudah,
e. Dapat digunakan dengan cara-cara yang berbeda
f. Kerangkapan data (*data redundancy*) minimal

2.12.3 Arsitektur Database.

Arsitek database terdiri atas tiga tingkatan, *extenal*, *conceptual* dan *interna*l. Tingkatan eksternal menegaskan bagaimana pengguna mengerti pengorganisasian dari data, data tunggal dengan berbagai bentuk merupakan tingkatan eksternal.

Tingkatan internal menegaskan bagaimana data secara fisik disimpan dan diproses dengan sistim komputer, tingkatan internal meliputi biaya, penampilan, *scalability*, dan kegiatan operasioal lainnya.

2.12.4 Komponen Sistem *Database*

Database sendiri adalah sekumpulan data yang berhubungan dengan secara logika dan memiliki beberapa arti yang saling berpautan utuh biasanya terdiri dari :

a. *Hardware*

   Hardware merupakan sistem computer actual yang digunakan untuk menyimpan dan mengakses databse. Dalam sebuah organisasi berskala besar, hardware terdiri jaringan dengan sebuah server pusat dan beberapa program client yang berjalan di komputer desktop.

b. *Software* beserta *utility*

   Software adalah DBMS yang aktual. DBMS memungkinkan para user untuk berkomunikasi dengan database. Dengan kata lain DBMS merupakan mediator antara database dengan user. Sebuah database harus memuat seluruh data yang diperlukan oleh sebuah organisasi.

c. Prosedur

Bagian integral dari setiap sistem adalah sekumpulan prosedur yang mengontrol jalannya sistem, yaitu praktik-praktik nyata yang harus diikuti user untuk mendapatkan, memasukkan, menjaga, dan mengambil data

d. Data

Data adalah jantung dari DBMS. Ada dua jenis data. Pertama, adalah kumpulan informasi yang diperlukan oleh suatu organisasi. Jenis data kedua adalah metadata, yaitu informasi mengenai *database*.

e. *User*

Ada sejumlah user yang dapat mengakses atau mengambil data sesuai dengan kebutuhan penggunaan aplikasi-aplikasi dan *interface* yang disediakan oleh DBMS, antara lain adalah

1. *Database administrator* adalah orang atau *group* yang bertanggungjawab mengimplementasikan sistem database di dalam suatu organisasi.
2. *Enduser* adalah orang yang berada di depan *workstation* dan berinteraksi secara langsung dengan sistem.
3. *Programmer* aplikasi, orang yang berinteraksi dengan database melalui cara yang berbeda.

2.12.5 Elemen Sistem *Database*

Sistem database mempunyai beberapa elemen penting yaitu : database sebagai inti sistim database, perangkat lunak untuk mengelola database, perangkat keras dan sebagai pendukung operasi pengolahan data, serta manusia yang mempunyai peran penting dalam sistim tersebut.

2.12.6 Tujuan Sistem Database

Keberhasilan suatu sistim informasi managemen sangat dipengaruhi oleh sistim database yang merupakan salah satu elemen penyusun sistim tersebut, di mana semakin lengkap/akurat/mudah menampilkan kembali data-data dalam sistem database maka semakin meningkatkan kualitas SIM. Tujuan sistem database meliputi :

a. Penyediaan sarana akses yang fleksibel,
b. Pemeliharaan integritas data
c. Proteksi data dari kerusakan dan penggunaan yang tidak legal
d. Penyediaan sarana untuk penggunaan bersama (*share*)

2.12.7 Manfaat Sistem Database

Penyediaan sarana penggunaan bersama database bertujuan :

a. Meminimalkan kerangkapan data (redudancy),
b. Menghilangkan ketergantungan data pada program-program aplikasi,
c. Menstandarkan definisi elemen data
d. Meningkatkan produktivitas personil sistem informasi

2.12.8 Tipe Database

Ada beberapa tipe "database" yang dikelompokkan berbasarkan fungsinya dan secara garis besarnya dibagi atas :

a. *Flat-file Database*

   Tipe flat-file database sesuai digunakan apabila jumlah data tidak terlalu banyak. Pada tipe ininya bahwa data selalu siap apabila diperlukan dan dapat diedit dengan tangan. Data disusun dalam satu file atau lebih, namun dapat dikatakan bahwa kita tidak dapat menyimpan data yang lebih kompleks pada flat-file database. Salah satu permasalahan utama dalam penggunaan flat-file for even semi-active database adalah bahwa fakta sangat memberi peluang untuk korupsi. Pada tipe ini tidak ada mekanisme untuk mendeteksi kapan suatu file telah digunakan atau dimodifikasi.

b. *Relational Database.*

   Relational databases seperti *MySQL*, *Microsoft SQL Server & Oracle*, memiliki lebih banyak menggunakan struktur logikal dalam penyimpanan data. Tabel dapat digunakan untuk menampilkan obyek ril, dengan setiap kejadian sebagai atribut. Contoh, Tabel disebut "books" dapat mempunyai judul kolom, penulis dan ISBN, yang menjelaskan secara terperinci setiap buku dimana setiap lajur dalam tabel adalah buku baru. "*Relation*", hubungan datang dari kenyataan bahwa tabel-tabel dihubungkan yang satu dengan yang lainnya, misal penulis buku dapat dihubungkan dengan tabel

"*authors*" untuk dapat menyediakan lebih banyak informasi tentang penulis. Salah satu keuntungan utama "relational model" ialah bahwa apabila database ditujukan untuk efisiensi, tidak akan ada duplikasi berbagai data, membantu menjaga integritas data. Relational database juga berfungsi untuk "*built-in*" yang membantu to retrieve, singkat dan mengedit data dalam berbagai cara.

c. *Operational Databases*

tipe ini digunakan untuk menyimpan data operasi harian organisasi, mencakup *inventory management, purchases*, *transctions* (transaksi) dan *financials* (pembiayaan). Semua data dikumpulakan dalam database yang biasanya diberi berbagai nama seperti database operasi/produksi, database subyek – area (SADB) atau database transaksi.

d. *Database Warehouses*

bahwa secara umum suatu organisasi/instansi menginginkan penyimpanan data dalam beberapa tahun. Data lama juga dan masih merupakan sumber informasi yang penting melalui analisis dan perbandingan data lampau dan data saat ini mempermudah menentukan "*key trend*" yang berjaya. Semua data tahunan terbut disimpan dalam "*Database Warehouse*". Data yang disimpan telah diskrening, pengeditan, dan pengintegrasian dan tidak memerlukan terlalu banyak lagi pengeditan atau alterasi.

Pada tipe penyimpanan data seperti, spesifikasi perangkat lunak (*software requirement specification* (SRS) yang dibutuhkan telah disetujui pada perencanaan kualita proyek (*projet quality plan*)

e. *Distributed Databases*

bahwa banyak organisasi atau perusahan yang memiliki banyak lokasi kantor, manufacturing plants, kantor regional, kantor cabang, dan satu kantor pusat yang secara geografis letaknya berbeda dari kelompok kerja tersebut kemungkinan memiliki database tersendiri yang secara bersama membentuk database organisasi atau perusahaan.

f. *End- User Databases*

bahwa di sana ada perbedaan ketersedian data pada lokasi kerja dari semua pengguna akhir pada banyak organisasi / instansi. Setiap lokasi kerja merupakan suatu database skala kecil tersendiri yang termasuk *data in spreadsheets*, *presentation* (penyajian), *word file*, *note pads* dan *download files*. Semua yang seperti database kecil (small databases) membentuk suatu tipe database yang disebut : *End-User Database*.

g. *External database*

bahwa diluar organisasi banyak sekali informasi yang kemungkinan dibutuhkan oleh organisasi/instansi. Semua database di luar organisasi yang dapat digunakan oleh banyak perusahaan tetapi dengan akses terbatas (pendaftaran) disebut : *External Database*.

h. *Hypermedia Database*

bahwa umumnya website memiliki berbagai halaman media interkoneksi yang dapat meliputi : teks, vidio clips, audio clips, fotografi dan grafik. Semua itu disimpan dan dipanggil dari suatu tempat apabila ingin membuat halaman web. Semua dari itu secara bersama membentuk " *Hypermedia Database*".

i. *Navigational Database*

bahwa *navigational database* memiliki aemua items yang referensinya dari obyek lain. Dalam hal ini, satu refensi dapat ke lainnya atau satu obyek ke obyek lainnya. Pada tipe database ini dapat menggunakan sistim moderen seperti *XPath*, seperti yang diaplilakasikan pada pengelolaan pengaturan lalu lintas udara.

j. *In - Memory Database*

bahwa pada tipe ini data disimpan dimemori utama komputer dengan menggunakan "*disk-based system*", yang cepat dan lebih reliable dibanding piringan (*disk*). Tipe ini diaplikasikan pada peralatan " *telecommunication network*".

k. *Document-Oriented Database*

bahwa tipe ini berbeda dengan berbagai tipe yang ada di mana tipe ini berorientasi kepada dokumen. Data disimpan dalam bentuk " *teks record*" dan kadang juga disimpan dalam bentuk tabel.

l. *Real-Time Database*

bahwa data ditangai secara tetap mengikuti perubahan. Contoh, adalah *stock market database* di mana pengaruh perubahan nilai (*share changes*) setiap menit harus diperbaharui dalam "*real-time database*". Tipe database juga dapat digunakan pada analisis dalam bidang kesehatan dan ilmu pengetahuan, perbankan, akunting, kontrol proses, sistim reservasi dan sebagainya. Essentialnya bahwa data yang dibutuhkan dapat diperoleh dengan cepat dan selalu *real time*.

m. *Analytical Database*

bahwa tipe ini digunakan untuk menyimpan informasi dari berbagai tipe database seperti : *selected operational databases* dan *external databases*. Nama lain dari *analytical database* adalah *information databases*, *Management databases* atau *multi-dimension databases*. Data yang disimpan dalam suatu " *Analytical Database*" digunakan oleh management untuk penggunaan analisa. Pada tipe ini, para analis dapat bekerja secara langsung, atau mengkreasi secara terpisah analisis database untuk Online Analytical Processing. Contoh, perusahaan yang dapat mengambil data pencatatan penjualan untuk menganalisis keefektifan dari reklame dan promosi penjualan lainnya pada tingkat agregasi.

2.12.9 Keunggulan dan Kelemahan Sistem *Database*

Keunggulan Pemakaian *Database*

a) Mengurangi kerangkapan data
b) Mencapai independensi data
c) Mengintegrasi data dari beberapa file
d) Mengambil data dan informasi secara cepat
e) Keamanan data terjamin

f) Data dapat dipakai secara bersama-sama oleh beberapa *system* program aplikasi secara *batch* maupun *online* pada saat bersamaan
g) Terpeliharanya keselarasan (kekonsistenan) data apabila ada perubahan data pada aplikasi yang berbeda-beda
h) Dapat diterapkan standarisasi

Kelemahan Pemakaian *Database*

a) Perangkat lunak yang mahal
b) Dibutuhkan tenaga yang terampil dalam mengelola data
c) *Storage* (tempat penyimpanan data) yang digunakan besar
d) Kerusakan pada sistem database dapat mempengaruhi departemen lain yang terkait

2.12.10 Pengertian DBMS

Database Management System (DBMS) adalah seperangkat program komputer yang mengontrol pembuatan, penanganan, dan penggunaan database. Kroncke at al (1997 dan 2007) menjelaskan bahwa *Database Management System* (DBMS) terdiri atas perangkat lunak yang mengoperasikan database, menyediakan penyimpanan, akses, keamanan, *back up* dan fasiilitas lainnya [14].

Database Management System (DBMS) adalah sistem pengorganisasian dan pengolahan data base pada komputer. Sistem ini dirancang untuk mampu melakukan berbagai data dengan beberapa referensi data yang sama. DBMS ini mampu diakses oleh berbagai aplikasi. Database Manajement System (DBMS) merupakan software yang digunakan untuk membangun sebuah sistem basis data yang berbasis komputerisasi. DBMS membantu dalam pemeliharaan dan pengolahan kumpulan data dalam jumlah besar. Sehingga dengan menggunakan DBMS tidak menimbulkan kekacauan dan dapat digunakan oleh pengguna sesuai dengan kebutuhan.

Sistem pengelolaan database dapat dikategorikan berdasarkan : model data yang didukung, seperti "*relational database*" atau XML, tipe komputer yang didukung, seperti "*server cluster*" atau "*mobile phone*", bahasa untuk mengakses database, seperi SQL atau Xquery, penampilan "*trade-ofif*" seperti "maximum scale atau "maximum speed" atau lainnya. Beberaba DBMS mencakup lebih dari

kategori i.e didukung beberapa bahasa akses seperti yang dilakukan pada DBMS MySQL, PostgreSQL, Microsoft Access, SQL Server, FileMaker, Oracle, RDBMS, dBASE, Clipper, FoxPro dan sebagainya [14].

2.12.10.1 Arsitektur DBMS Multi Pengguna

Arsitektur yang biasanya digunakan untuk dapat mengimplementasikan sisitem basis data yang pengguna, antara lain :

1. *Teleprocessing*

   Arsitektur tradisional untuk sistem multi pengguna adalah *teleprocessing*, dari suatu komputer dengan sebuah CPU dan sejumlah terminal komputer. Semua pemrosesan dikerjakan dalam batasan fisik komputer yang sama. Terminal untuk pengguna berjenis ‘dumb’, yang tidak dapat berfungsi sendiri dan masing-masing dihubungkan ke komputer pusat. Terminal tersebut mengirimkan pesan melalui subsistem pengontrol komunikasi pada sistem operasi ke program aplikasi, yang bergantian menggunakan layanan DBMS. Dengan cara yang sama, pesan dikembalikan ke terminal pengguna.

2. *File Server*

   Proses didistribusikan ke dalam jaringan, sejenis Local Area Network (LAN). File server mengendalikan file yang diperlukan oleh aplikasi dan DBMS. Meskipun aplikasi dan DBMS dijalankan pada masing-masing workstation tetapi tetap meminta file dari file server jika diperlukan. Dengan cara ini, file server berfungsi sebagai hard disk yang digunakan secara bersamaan. DBMS yang ada pada setiap workstation meminta data ke file server untuk semua data yang di inginkan oleh DBMS.

3. *Client Server*

   Client server menunjukkan cara komponen software berinteraksi dalam bentuk sistem. Sesuai dengan namanya, ada sebuah pemroses client yang membutuhkan sumber dan sebuah server yang menyediakan sumbernya. Server diletakkan pada satu sisi dalam LAN dan client pada sisi yang lain.

2.13 *Android*

Perkembangannya saat ini telah menjadikan *android* menjadi *platform* yang sangat cepat dalam melakukan inovasi. Setelah diakuisisi oleh *Google*, *android* kini menjadi *platform* yang terdiri dari sistem operasi *Linux* dan juga GUI (*Graphical User Interface*) dimana memudahkan untuk para pengembang untuk membuat aplikasi berbasis android [15].

2.13.1 Sejarah *Android*

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis *linux* yang mencakup kedalam sistem operasi, *middleware* dan aplikasi. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, *Google Inc*. membeli *Android Inc*. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone*.

Kemudian untuk dapat mengembangkan *Android* dibentuklah *Open Handset Alliance, konsorsium* yang berasal dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi termasuk *Google, HTC, Intel Motorola, Qualcomm, T-Mobile,* dan *Nvidia*.

Pada saat perilisan perdana *Android*, 5 November 2007, *Android* bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Di dunia ini terdapat dua jenis distributor yang *system* operasi *android*. Pertama yang mendapat dukungan penuh dari *Google* atau *Google Mail Services* (GMS) dan kedua. distribusinya tanpa dukungan langsung *Google* atau dikenal sebagai *Open Handset Distribution (OHD)*. Pada kisaran September 2007 *Google* mengenalkan *Nexus One*, salah satu jenis smartphone yang menggunakan *android* sebagai *system* operasinya.

Seiring pembentukan *Open Handset Alliance*, OHA adalah produk perdana mereka, android, perangkat *mobile* yang merupakan modifikasi *kernel Linux* 2.6. Sejak *android* dirilis telah dilakukan berbagai pembaruan berupa perbaikan bus dan penambahan fitur baru. Pada masa saat ini sebagian besar *vendor – vendor smartphone* sudah memproduksi *smartphone* berbasis *android*, Hal ini karena android itu adalah sistem operasi yang *open source* sehingga bebas didistribusikan dan dipakai oleh *vendor* manapun.

Pesatnya pertumbuhan *android* selain faktor yang disebutkan diatas adalah karena *android* itu sendiri adalah *platform* yang sangat lengkap baik itu sistem operasinya, aplikasi dan *Tool* Pengembangannya, market aplikasi android serta dukungan yang sangat tinggi dari komunitas *open source* di dunia, sehingga android terus berkembang pesat baik dari segi teknologi maupun dari segi jumlah *device* yang ada di dunia [15].

2.13.2 Versi Android

Terdapat keunikan dari nama sistem operasi android yaitu digunakannya nama makanan penutup (*Dessert*) dan setiap kenaikan versi nama dalam sistem operasinya memiliki huruf awalan berutan sesuai abjad. Berikut versi – versi android yang pernah dirilis :

1. Android 1.1

   Awalnya versi sistem operasi android ini dirilis perangkat T-Mobile G1. Versi ini merupakan update dari untuk memperbaiki beberapa bugs dan mengganti API serta menambahkan beberapa fitur lain.

2. Android 1.5 (Cupcake)

   Versi ini pertama kali dirilis pada pertengahan Mei 2009. Dimulai dari versi ini penamaan versi android mulai menggunakan nama makanan pencuci mulut (dessert). Sistem operasi ini berbasiskan kernel Linux dan menambahkan beberapa update serta UI baru dari android yang sebelumnya.

3. Android 1.6 (Donut)

   Dirilis pertama kali pada September 2009. Memiliki peningkatan fitur pencarian dan UI yang lebih user friendly. Pada versi ini sudah mendukung teknologi CDMA/EVDO, 802.1x, VPNs kemudian support layar dengan resolusi WVGA.

4. Android 2.0/2.1 (Éclair)

   Versi ini dirilis pertama kali, karena terjadi penambahan fitur untuk pengoptimalan hardware, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML5.

5. Android 2.2 (Froyo : Frozen Yoghurt)

   Dirilis pertama kali pada smartphone *Google Nexus One*. Pada versi ini sudah support terhadap Adobe Flash Player 10.1, peningkatan pada kecepatan membuka dan menutup aplikasi serta penggunaan SD Card sebagai media penyimpanan aplikasi.

6. Androidd 2.3 (Gingerbread)

   Pertama kali dirancang untuk memaksimalkan kemampuan aplikasi dan game serta mulai digunakannya *Near Field Communication (NFC).*

7. Android 3.0/3.1 (Honeycomb)

   Dirilis pada 22 Februari 2011 dan *Motorola Xoom* adalah yang pertama kali menggunakannya. Versi ini didesain khusus untuk pengoptimalan penggunaan tablet PC.

8. Android 4.0 (Ice Scream Sandwidch)

   Dirilis pada 19 Oktober 2011 dan yang pertama kali digunakan pada *smartphone* Samsung Galaxy Nexus. Secara teori semua perangkat android versi sebelumnya, Gingerbread dapat di-update ke android 4.0 *Ice Scream Sandwidch*.

9. Android 4.1 (Jelly Bean)

   Android Jelly Bean dibuat dengan memiliki keunggulan dan fitur baru. Penambahan baru itu diantaranya adalah meningkatkan input *keyboard*, desain baru fitur pencarian, UI yang baru serta pencarian melalui *Voice Search* yang lebih cepat. *Android Jelly Bean* ini pertama kali digunakan pada produk tablet *Asus* yaitu *Google Nexus* 7.

2.13.3 Arsitektur Android

Arsitektur android terdiriri dari beberapa lapisan yang mendukung fungsi–fungsi spesifik dari sistem operasi. Berikut lapisan – lapisan arsitektur android :

1. *Applications and Widgets*

   Merupakan layer yang berhubungan dengan aplikasi saja. Biasanya dengan mendownload aplikasi dan kemudian dilakukan instalasi yang dapat menjalankan sistem aplikasi tersebut. Pada layer terdapat aplikasi yang seperti klien email, program SMS, kalender, peta, browser.

2. *Applications Frameworks*

   Android termasuk kedalam *Open Development Platform* yang bersifat menawarkan kepada pengembang untuk membangun aplikasi bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi resources, menjalankan service background , mengatur alarm, menambah status notification dan lain – lain. Contoh dari *service* yang berjalan di background adalah *Asyncronous Task* yang merupakan sebuah *class* abstrak yang digunakan untuk melakukan operasi pada background dan menampilkannya di UI. Pengembang juga memiliki akses menuju API *framework* seperti yang dilakukan aplikasi pada kategori inti.

   Komponen yang termasuk didalam *Applications Framework* adalah sebagai berikut :

   a. Views
   b. Content Provider
   c. Resource Manager
   d. Notification Manager
   e. Activity Manager

3. *Libraries*

   Merupakan layer dimana fitur – fitur android berada dan menyediakan satu set *library-library* dalam bahasa C/C++ yang dapat digunakan oleh berbagai komponen android. Berikut beberapa library yang ada di android :

   a. *Libraries* media utnuk pemutaran media gambar, audio maupun video.
   b. *Libraries* untuk manajeman tampilan
   c. *Libraries Graphics* SGL dan OpenGL untuk grafis 2D dan 3D
   d. *Libraries* SQLite untuk dukungan database
   e. *Libraries* SSL dan WebKit terintegrasi dengan web browser dan security.

f. *Libraries Live Webcore* mencakup modern *web browser* dengan *engine embedded web view*.

g. *Libraries* 3D untuk implementasi OpenGL ES 1.0 API's.

4. Android *Runtime*

Merupakan layer yang membuat aplikasi android dapat dijalankan dimana dalam prosesnya menggunakan implementasi *Linux*. *Dalvik Virtual Machine* (DVM) adalah mesin bentuk dasar kerangka aplikasi android. Terdapat dua bagian dalam android *Runtime*, yaitu :

a. *Core Libraries*

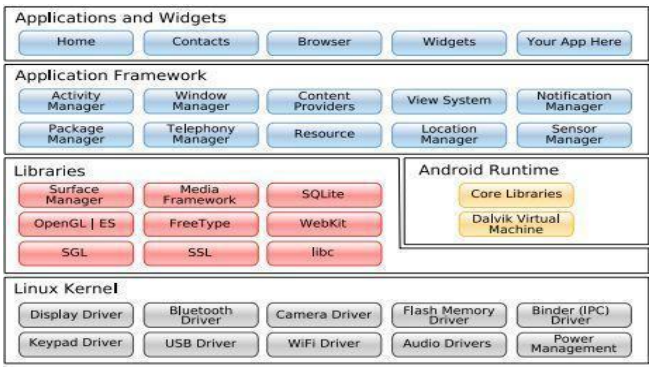
Aplikasi android di buat menggunakan bahasa pemrograman *Java*. *Dalvik Virtual Machine* bukan merupakan *Virtual Machine Java*

b. *Dalvik Virtual Machine* (DVM)

Virtual mesin berbasis register dioptimalkan agar berfungsi secara efisien dalam pengembangan dari Linux kernel untuk melakukan *threading.*

5. Linux Kernel

Merupakan layer yang dimana inti dari system operasi android itu berada. Keberadaannya dapat mengatur sistem *processing, memory, resources, drivers* dan sistem operasi android lainnya. Linux kernel yang digunakan android adalah *linux kernel release* 2.6. Untuk lebih mudah arsitektur android dapat dilihat pada gambar 2.15:



Gambar 2.15 Arsitektur Android [15].

2.13.4 Komponen Android

Aplikasi android dibangun menggunakan bahasa pemrograman Java untuk membuat aplikasi android terdapat beberapa komponen utama didalamnya, yaitu :

1. *Activites*

   Merupakan suatu tampilan (*User Interface*) untuk pengguna melakukan interaksi dengan aplikasi android biasanya memiliki lebih dari satu *activity* tergantung tujuan dan desain aplikasi tersebut.

2. *Service*

   Service tidak memiliki *Graphic User Interface*, tetapi *service* dapat berjalan secara *background*. Sebagai contoh media player sedang memutar musik dari daftar lagu yang ada, aplikasi ini akan memiliki dua atau lebih activity yang memungkinkan pengguna untuk aktifitas lain seperti menulis sms selama musiknya masih berjalan. Untuk melakukan itu, *activity* media *player* dapat menjalankan *service* dimana *service* tersebut akan dijalankan pada thread utama dari proses aplikasi.

3. *Broadcast Receiver*

   Broadcast Receiver adalah komponen aplikasi yang bersifat *system-wide broadcast announcements*. Beberapa contoh *broadcast* dikirimkan oleh sistem operasi android adalah munculnya notifikasi ketika aplikasi baru yang ter-install/ter-*update*, layar mati, *booting* sistem operasi selesai, atau telepon masuk, dan sebagainya. *Broadcast Receiver* tidak memiliki tampilan, tetapi dapat menjalankan suatu *activity* atau menampilkan notifikasi di *Notification* Bar.

4. *Content Provider*

   Sebuah *content provider* mengatur sekumpulan data aplikasi terbagi Dengan *content provider* dapat menyimpan data di *file system*, sebuah database SQLite, di web, atau di metode penyimpanan data lainnya yang bisa di akses oleh aplikasi melalui *content provider*. Sebagai contoh, sistem android menyediakan *content provider* yang mengatur informasi kontak user.

5. *Intent*

Intent merupakan pesan *Asynchronous* yang aplikasi untuk memberikan request secara fungsional dari komponen berbeda dalam sistem android.

2.14 JAVA

Java merupakan bahasa pemrograman berorientasi objek yang bersifat *open source* dan bisa dijalankan diberbagai platform. *Java* mengadopsi sintaks dari bahasa C dan C++. Setiap program yang ditulis dengan bahasa Java awalnya dikompilasi menjadi sebuah kode objek yang disebut *bytecode*. Hasil dari *bytecode* adalah file berekstensi **.class*, selanjutnya *file *.class* tersebut akan diterjemahkan baris demi baris (interpreter). Dalam terminology *Java* terdapat istilah *"write once, run anywhere"*. Ini berarti bahwa sekali menulis program *Java* dan melakukan kompilasi terhadap program *Java* tersebut, maka *bytecode* bisa dijalankan didalam platform manapun selama platform tersebut terdapat *Java Virtual Machine* (JVM) [15].

2.14.1 Sejarah Java

Bahasa pemrograman terlahir dari sebuah proyek yang diberi nama *The Green Project*. Proyek tersebut berlangsung selama 18 bulan, mulai dari awal tahun 1991 hingga musim panas 1992. Proyek ini yang dikerjakan oleh Patrick Naughton, Mike Sheridan, James Gosling dan Billy Joy dan beserta sembilan pemrogram lainnya yang berasal dari Sun Microsystems [15].

2.15 MySQL

MySQL atau yang sering dibaca dengan ejaan *"mai es que el"* ini adalah sebuah database yang berbasiskan *server database system*. Kemampuannya dalam menangani RDBMS (*Relasional Database Management System*) mengakibatkan *database* ini menjadi *database* yang sangat popular saat ini. *Database MySQL* mampu menangani data yang sangat besar hingga ukuran Giga Byte, kemampuan daya tampung data ini maka *MySQL* sangat cocok digunakan untuk mengcover data perusahaan baik kecil sampai perusahaaan besar [15].

2.16 *Eclipse*

Eclipse merupakan sebuah IDE (*Integrated Development Environment*) untuk mengembangkan suatu aplikasi dan dapat dijalankan di berbagai platform (*Platform-Independent*) Eclipse termasuk *tools* yang bersifat *open source* [16]. Eclipse memiliki beberapa sifat diantaranya adalah sebagai berikut :

1. *Multi-platform*

   Target *system* operasi eclipse ini adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.

2. *Multi-language*

   Dikembangkan dengan bahasa pemrograman *Java*. *eclipse* mendukung pengembangan aplikasi dengan bahasa pemrograman lain seperti C/C++, Cobol, Python, Perl, PHP dan lain – lain.

3. *Multi-role*

   Eclipse dapat digunakan untuk aktifitas siklus pengembangan aplikasi seperti dokumentasi, pengujian, pengembangan dan sebagainya.

2.16.1 Arsitektur Eclipse

Eclipse merupakan sebuah kernel yang berarti bahwa eclipse adalah fungsi dari pluf-in yang sudah dipasang (di-*install*). Basis dari eclipse dinamakan *Rich Client Platform* (RCP). Berikut adalah komponen yang membentuk RCP :

1. Core Platform
2. OSGi
3. SWT (Standard Widget Toolkit)
4. JFace
5. Eclipse Workbench

Eclipse dilengkapi dengan JDT (*Java Development Tools*), plug-in yang membuat eclipse dapat kompatibel untuk pengembangan program Java dan PDE (*Plug-in Development Environment*). Eclipse beserta plug-in yang dimilikinya diimplementasikan dalam bahasa pemrograman Java [16].

2.16.2 *Versi Eclipse*

Eclipse Foundation telah mengkoordinasikan peluncuran *Eclipse* secara rutin dikenal dengan nama *simultaneous Release*. Setiap versi yang diluncurkan

terdiri dari *eclipse platform* dan sejumlah proyek yang terlibat langsung dalam pembuatan proyek *eclipse* [16].

2.17 *Software Development Kit (SDK)*

Software Development Kit (SDK) adalah sekumpulan *tools* untuk membuat dan menjalankan aplikasi. SDK juga merupakan kerangka kerja perangkat lunak, platform perangkar keras, sistem komputer, konsol video game, sistem operasi dan sebagainya [16].

2.18 *Development Kit (JDK)*

Java Development Kit (JDK) adalah produk dari *Sun Microsystems* untuk pengembang *Java*. Sejak *Java* di perkenalkan, hingga saat ini banyak yang telah menggunkan Java SDK [16].

2.19 Skala *Likert*

Skala Likert adalah suatu skala psikometrik yang umum digunakan dalam kuesioner, dan merupakan skala yang paling banyak digunakan dalam riset berupa survei. Nama skala ini diambil dari nama Rensis Likert, yang menerbitkan suatu laporan yang menjelaskan penggunaannya. Saat menanggapi pertanyaan dalam skala *likert*, responden menentukan tingkat persetujuan mereka terhadap suatu pernyataan dengan memilih salah satu dari pilihan yang tersedia. Biasanya disediakan lima pilihan skala dengan format seperti :

a. Sangat tidak setuju
b. Tidak setuju
c. Netral
d. Setuju
e. Sangat setuju

Penulis kuesioner harus memutuskan apakah memasukkan titik tengah atau tidak sesuai dengan pernyataan yang diberikan kepada responden. Meskipun penggunaan respon kategori tengah tidak mempengaruhi reliabilitas dan validitas dalam penelitian ini, namun direkomendasikan bahwa penilaian [17].