BAB II
TEORI PENUNJANG

2.1 *Android Operating System (Android OS)*

Android merupakan salah satu sistem operasi *Smartphone* yang sangat pesat penggunaannya saat ini. Tidak hanya *Smartphone* saja, saat ini penggunaan *Android* telah merambah ke berbagai aspek pengaplikasian, seperti *smart home*, perangkat elektronik; televisi, kulkas, mesin cuci, mobil dan sebagainya. Awalnya *Android* dikembangkan oleh Android Inc. Google pun membantu finansialnya dan membelinya pada tahun 2005.



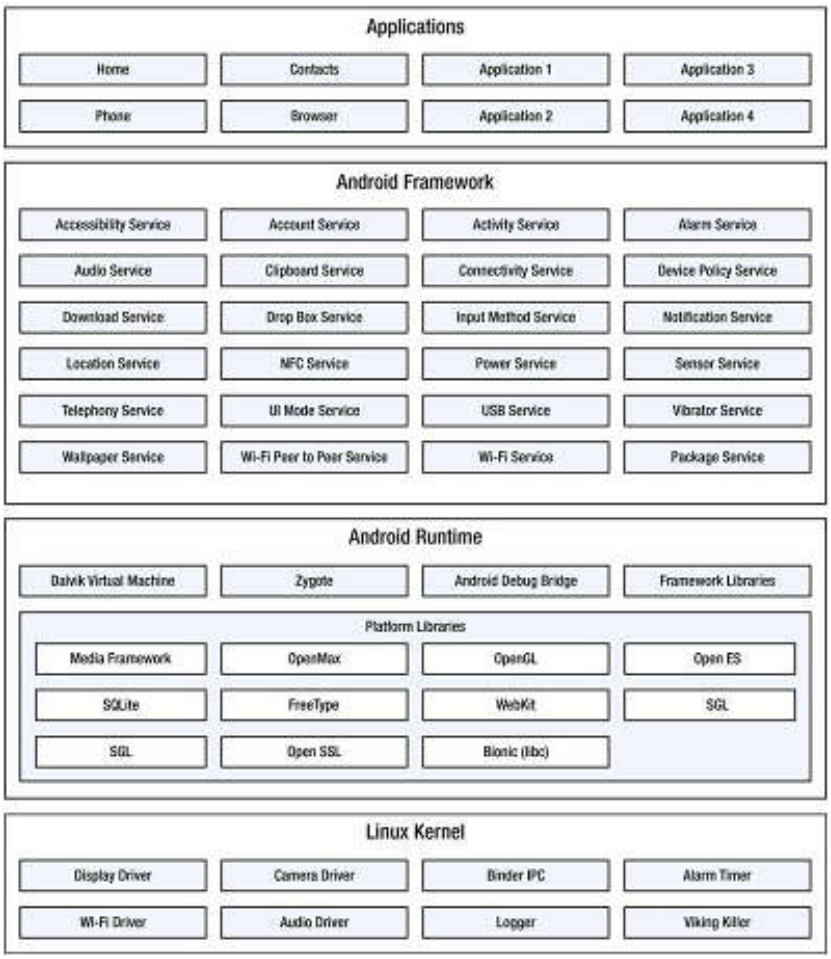
Gambar 2.1 Logo Android

Kemudian pada November 2007, dibentuk sebuah konsorsium yang dinamakan Open Handset Alliance untuk membuat dan mengembangkan standar terbuka (*open source*) bagi perangkat *mobile.* Produk pertama dari konsorsium ini adalah *Android Platform. Platform* ini menjadi *open source* yang dipimpin oleh Google dan di bawah naungan Open Handset Alliance. Tujuan dari *Android Open Source Project* ini adalah menyediakan *platform* terbuka sehingga siapa saja dapat mengembangkan pengalaman pengguna (*user experience*) khususnya perangkat *mobile.* [8]

Android adalah *platform mobile* pertama yang lengkap, terbuka, dan bebas, dengan maksud :

a. Lengkap. *Platform* Android merupakan Platform yang kuat, aman, mudah dikembangkan (*upgradable*), *Platform mobile* dengan antarmuka yang menarik dan *framework* yang luas. Mengizinkan para pengembang untuk mengembangkan dan secara penuh menggabungkan aplikasi mereka dengan *platform* ini.

b. Terbuka. *Platform* Android telah dikembangkan dan disediakan di bawah peraturan lisensi *open source* dari Apache. Tidak membedakan antara aplikasi yang telah terdapat pada p*latform Android* (*pre-load application*) atau dengan aplikasi pihak ketiga (*third-party application*). Pengembang memiliki akses penuh ke fitur dan layanan perangkat ketika mengembangkan aplikasinya.
c. Bebas. *Platform Android* tidak menuntut biaya lisensi, royalti, keanggotaan, ataupun sertifikasi dalam pengembangan aplikasi pada *platform* ini. *Source code* beserta *Software Development Kit* (SDK) disediakan secara bebas untuk para developer untuk mengembangkan aplikasinya.



Gambar 2.2 Arsitektur dari Platform Android

Pembuatan aplikasi dalam *Android* memiliki beberapa *tool* yang bisa digunakan oleh para pengembang (*developer*) atau biasa disebut *Interface Development Kit* (IDE) seperti *Eclipse, App Inventor, Android Studio* dan lainnya.

Nantinya aplikasi yang dibuat pengembang, didistribusikan oleh Google di *Digital Store* yang bernama *Google Play Store*. Nantinya para pengguna

Smartphone berbasis *Android* dapat mengunduh aplikasi dari *Store* tersebut dan dipasang pada *Smartphone.*

Saat ini, versi terbaru dari *Android* adalah *Android* versi 7.0 dan dinamakan *Android Nougat* yang diluncurkan pada Agustus 2016 setelah versi-versi sebelumnya :

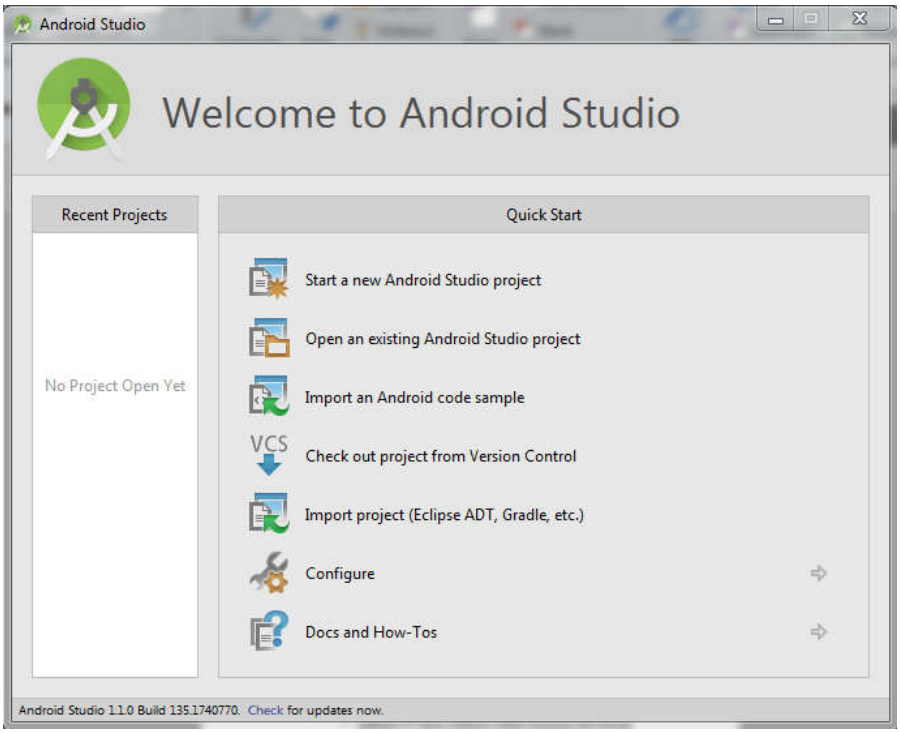
1. Android 1.5 (*Cupcake*)
2. Android 1.6 (*Donut*)
3. Android 2.0, 2.0.1, 2.1 (*Eclair*)
4. Android 2.2, 2.2.1, 2.2.2, 2.2.3 (*Froyo*)
5. Android 2.3, 2.3.3, 2.3.5, 2.3.6 (*Gingerbread*)
6. Android 3.0, 3.1, 3.2, 3.2.1, 3.2.2 (*Honeycomb*)
7. Android 4.0.1, 4.0.2, 4.0.3, 4.0.4 (*Ice Cream Sandwich*)
8. Android 4.1, 4.2, 4.3 (*Jelly Bean*)
9. Android 4.4, 4.4.1, 4.4.2, 4.4.3, 4.4.4 (*KitKat*)
10. Android 5.0, 5.0.1, 5.0.2, 5.1, 5.1.1 (*Lollipop*)
11. Android 6.0, 6.0.1, 6.1 (*Marshmallow*)

Di dalam *Android*, terdapat berbagai fitur untuk dapat dikembangkan menjadi berbagai aplikasi penunjang lain (*third-party application*) untuk para pengembang (*developer*) seperti fitur perpetaan (*Google Maps*) dengan *Google Maps API*, dukungan terhadap berbagai sensor, seperti *barometer, gyroscope, compass* serta *global positioning system (GPS)* dan sebagainya. Hal ini membuat banyak pengembang yang tertarik dan membuat aplikasi untuk sistem operasi *Android* ini.

2.2 Android Studio

Android Studio adalah sebuah peralatan (*tool*) atau yang biasa disebut dengan Integrated Development Environment (IDE) untuk pengembangan aplikasi yang berbasis *Android* dan dikembangkan oleh Google secara mandiri yang diperkenalkan pada 16 Mei 2013 dalam acara Google I/O. IDE ini berada di bawah lisensi *Apache 2.0* dan didasarkan pada *Software Intellij IDEA* milik *Jetbrain*.

Versi awal dari *Android Studio* yaitu versi 0.1 yang diluncurkan beberapa hari setelah pengenalan. Dan selanjutnya memasuki versi beta dimulai dari versi 0.8 pada Juni 2014. Dan akhirnya versi stabil dari *Android Studio* yaitu versi 1.0 dirilis pada Desember 2014. [11]



Gambar 2.3 User Interface Android Studio 1.1.0

Selama ini, *Eclipse* merupakan salah satu IDE yang paling banyak digunakan oleh para developer. Namun, menurut website *developer.google.com*, *Android Development Tools Plugin* untuk *Eclipse* tidak akan dikembangkan kembali, sehingga jika para *developer Android* yang masih menggunakan *Eclipse* sebagai dasar pembuatan aplikasi, maka disarankan untuk berpindah atau bermigrasi ke *Android Studio*.

Fitur-fitur yang terdapat pada *Android Studio* adalah :

1. *Live layout* : pemrosesan aplikasi secara *real-time*, pemrograman secara langsung, dan *WYSIWYG Editor*.
2. *Developer console* : saran untuk optimasi pembuatan aplikasi, bantuan penerjemahan sintaks, pencarian referensi, dsb.
3. Mendukung aplikasi yang masih dalam tahap pengembangan (versi beta)

4. Mendukung build berbasis *Gradle*.
5. *Lint tools* untuk melihat kinerja aplikasi, kegunaan, versi aplikasi untuk kompatibilitas dengan tipe *Android* yang digunakan.
6. Kemampuan untuk penanganan aplikasi dan *proguard.*
7. Langkah-langkah pembuatan berdasarkan *template* yang telah tersedia untuk pembuatan aplikasi *Android* secara umum.
8. *Layout editor* yang kaya akan fitur, seperti memungkinkan pengguna untuk men-*drag and drop* komponen UI (*User Interface*), pilihan untuk pratinjau layout pada beberapa konfigurasi layar.
9. Dukungan untuk *Android Wear* (*wearable device*)
10. Sudah termasuk mendukung *Google cloud platform* (perpesanan dan poros aplikasi / *app engine*).

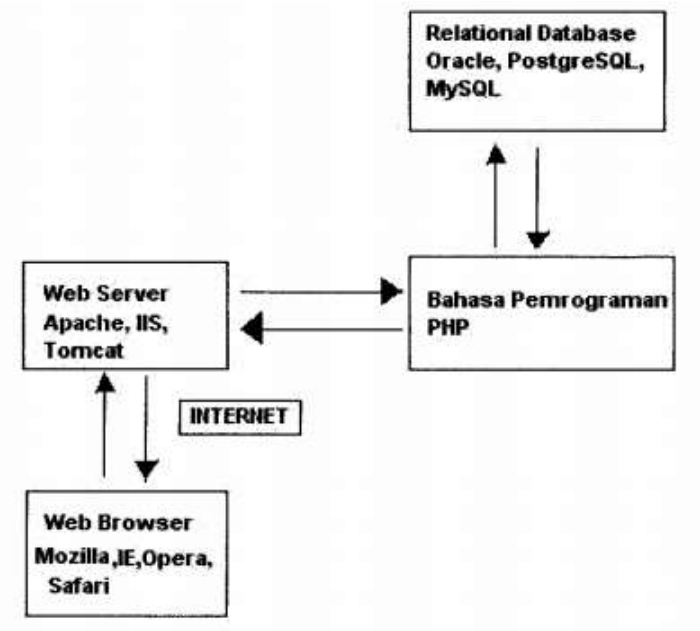
2.3 Pengelolaan *Database* pada Sistem

Dalam sistem yang dibuat, nantinya data-data yang digunakan akan disimpan ke dalam sebuah *database*. *Database* ini akan menggunakan bahasa pemrograman *PHP* dan *MySQL* sebagai *database*. Selanjutnya dijelaskan pada sub bab berikut.[1]

2.3.1 PHP

PHP atau *Hypertext Prepocessor* merupakan salah satu bahasa pemrograman dari sisi *server (web server-side)*, terintegrasi dengan HTML dan mampu membuat halaman *website* menjadi dinamis. Arti dari dinamis sendiri adalah halaman yang dibuat dan ditampilkan dari sisi *client* pada suatu browser atau pengguna akan mendapat pembaruan sesuai dengan apa yang terdapat di sisi *server* tersebut.

Pada tahun 1995 adalah pertama kali PHP dibuat. Versi awal dari PHP dibuat oleh Rasmus Lerdorf yang diberi nama FI (*Form Interpreted*) yang kemudian dirilis secara *open-source* dan akhirnya sebuah perusahaan bernama Zend merilis ulang PHP versi 2.0 yang sebelumnya telah disempurnakan oleh Rasmus sendiri, agar lebih baik, bersih, dan cepat. Hingga saat ini PHP telah memasuki versi 7.0.



Gambar 2.4 Bagan dari aplikasi web

Beberapa kelebihan dari PHP sebagai berikut :

a. Kesederhanaan. Selain itu juga memiliki banyak fungsi *built-in* untuk menangani kebutuhan standar pembuatan aplikasi web sehingga *user* tinggal memakai dan mengembangkannya.
b. Memiliki banyak referensi dalam bahasa *scripting* nya.
c. Merupakan bahasa *open source* yang dapat digunakan di berbagai sistem operasi, dijalankan secara *runtime* melalui *console* serta menjalankan perintah sistem.
d. Apache, IIS, Xitami adalah contoh *web server* yang mendukung PHP dengan konfigurasi yang mudah.
e. Mendukung banyak *database,* seperti Oracle, PostagreSQL, MySQL, dan lainnya.

2.3.2 *MySQL*

MySQL merupakan salah satu *Database Management System* (DBMS) *SQL* yang bersifat gratis di bawah lisensi GNU General Public License (GPL). Selain *MySQL*, banyak juga jenis DBMS seperti *Oracle*, *MS SQL, Postagre SQL* dan lainnya.

MySQL dipegang hak cipta kode sumbernya oleh MySQL AB yang didirikan oleh David Axmark, Allan Larson, dan Michael Monty Widenius.

Kelebihan dari MySQL adalah :

a. Bersifat Open Source.
b. Dapat berjalan stabil di berbagai sistem operasi, seperti Windows, Linux, FreeBSD, Solaris, dan sebagainya.
c. Bersifat *multiuser*, dapat digunakan banyak *user* dalan waktu bersamaan.
d. Dari sisi keamanan data, memiliki beberapa lapisan keamanan, seperti level *subnet mask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta kata sandi yang terenkripsi.
e. Memiliki antarmuka terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interfacing*).
f. Memiliki kecepatan yang baik dalam menangani *query* (perintah *SQL*) artinya dapat memproses lebih banyak *SQL* per satuan waktu.

2.4 *Google Map*

Google Map adalah sebuah layanan pemetaan yang disediakan oleh Google secara gratis. Menawarkan peta dan gambar satelit untuk seluruh dunia.

Google juga telah menyediakan *Application Programming Interface* (API) *Google Map,* layanan untuk para pengembang aplikasi *Android* yang ingin menggunakan layanan perpetaan ini. Terdapat dua *API Google Map* yang disediakan, yaitu standar dan untuk bisnis. Perbedaannya untuk bisnis sendiri, kuota akses dalam layanan *API Google Map* lebih besar dan dukungan langsung dari Google selaku pembuatnya.[5]

Awalnya, *API Google Map* versi pertama (*V1*) digunakan untuk pembuatan aplikasi perpetaan pada *Android* hingga akhirnya dihentikan pada akhir 2012. Selanjutnya digunakanlah *API Google Map* versi kedua (*V2*) untuk aplikasi *Android* yang ingin menampilkan perpetaan. Perbedaannya adalah :

1. *API Google Map V2* menggunakan SHA1 untuk mendapatkan *Google API Key*. Sedangkan sebelumnya menggunakan MD5.
2. *API Google Map V2* menggunakan Fragment menggantikan MapView.

3. Sebelum menggunakan *API Google Map V2*, harus instalasi *library google-play-service* pada *Android* SDK.
4. Adanya penambahan *marker* (penanda lokasi) sehingga bisa diganti ikon sendiri atau ikon standar dan bisa diberi keterangan yang disisipkan ketika ikon tersebut disorot atau ditekan.
5. Dan sebagainya.

Dalam pembutan program *API Google Map* menggunakan urutan sebagai berikut:

1. Memasukkan *Maps API JavaScript* ke dalam HTML.
2. Membuat *element div* dengan nama *map_canvas* untuk menampilkan peta.
3. Membuat beberapa objek literal untuk menyimpan properti-properti pada peta.
4. Menuliskan fungsi *JavaScript* untuk membuat objek peta.
5. Meng-inisiasi peta dalam *tag body* HTML dengan *event onload*.

Pada *API Google Map*, terdapat empat jenis pilihan model peta yang disediakan oleh Google, diantaranya adalah:

1. ***Roadmap***, untuk menampilkan peta biasa dua dimensi.
2. ***Satellite***, untuk menampilkan foto satelit.
3. ***Terrain***, untuk menunjukkan relief fisik permukaan bumi dan menunjukkan seberapa tingginya suatu lokasi, contohnya akan menunjukkan gunung dan sungai.
4. ***Hybrid***, akan menunjukkan foto satelit yang diatasnya tergambar pula apa yang tampil pada *roadmap* (jalan dan nama kota).

Berikut contoh *screenshot* nya:



Gambar 2.5 Tampilan Roadmap pada Google Maps Kecamatan Cibeunying Kaler

2.5 Global Positioning System (GPS)

Global Positioning System atau *GPS* adalah sistem yang dibuat untuk menentukan letak di permukaan bumi dengan bantuan penyelarasan (*synchronization*) sinyal satelit. Sistem ini menggunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke Bumi. Sinyal ini diterima oleh alat penerima di permukaan, dan digunakan untuk menentukan letak, kecepatan, arah, dan waktu. Sistem yang serupa dengan *GPS* antara lain *GLONASS* milik Rusia, *Galileo* milik Uni Eropa, *IRNSS* milik India, dan *BDS* milik Republik Rakyat Tiongkok (RRT). Sistem ini dikembangkan oleh Departemen Pertahanan Amerika Serikat, dengan nama lengkapnya adalah *NAVSTAR*. Kumpulan satelit ini dioperasikan oleh *50th Space Wing* Angkatan Udara Amerika Serikat.[6]

2.5.1 Prinsip Kerja

Setiap sudut-sudut atau daerah di atas permukaan bumi ini minimal terjangkau oleh 3-4 satelit. Pada prakteknya, setiap *GPS* terbaru bisa menerima sampai dengan 24 saluran satelit sekaligus. Kondisi langit yang cerah dan bebas dari halangan membuat *GPS* dapat dengan mudah

menangkap sinyal yang dikirimkan oleh satelit. Semakin banyak satelit yang diterima oleh *GPS*, maka akurasi yang diberikan juga akan semakin tinggi, tetapi apabila keadaan cuaca buruk seperti hujan atau badai maka sinyal akan sulit untuk didapatkan.

Satelit *GPS* berputar mengelilingi bumi selama 12 jam di dalam orbit yang sangat akurat dan mengirimkan sinyal informasi ke Bumi. *GPS receiver* mengambil informasi itu dan dengan menggunakan perhitungan *triangulasi* menghitung lokasi user dengan tepat.

GPS receiver membandingkan waktu sinyal dikirim dengan waktu sinyal tersebut diterima. Dari informasi itu didapat diketahui berapa jarak satelit. *GPS receiver* dapat melakukan perhitungan dan menentukan posisi user dan menampilkan dalam peta elektronik.

Cara kerja dari *GPS* meliputi 5 langkah, yaitu :

1. Memakai perhitungan *triangulasi* dari satelit.
2. Untuk perhitungan *triangulasi*, *GPS* mengukur jarak menggunakan *travel time* sinyal radio.
3. Untuk mengukur *travel time*, *GPS* memerlukan akurasi waktu yang tinggi.
4. Untuk perhitungan jarak, kita harus tahu dengan pasti posisi satelit dan ketinggian pada orbitnya.
5. Harus mengoreksi delay sinyal waktu perjalanan di atmosfer sampai diterima *receiver*.

Sehingga manfaat dari *GPS* itu sendiri adalah meliputi :

1. **Lokasi**. Dapat mengetahui tempat di mana seseorang atau target yang diinginkan.
2. **Navigasi.** Dapat menunjukkan arah ke tempat yang telah disepakati sebelumnya sehingga nantinya dapat menampilkan lintasan yang dapat dilewati.
3. **Pelacakan.** Dapat mengetahui tempat seseorang yang ingin dilacak keberadaannya.

4. **Waktu.** Dapat tercatat kapan tepatnya seseorang berada di lokasi tersebut.

2.5.2 Assisted-GPS (A-GPS)

Assisted GPS adalah jenis *GPS* yang tertanam pada perangkat *mobile* dan mengandalkan sebuah server bantuan, selain dari satelit. *A-GPS* dikembangkan untuk meningkatkan kinerja *GPS*. Server bantuan tersebut akan memberikan informasi tambahan ke perangkat yang dapat membantu dalam pencarian lokasi. Hal ini berguna apabila *chip GPS* mungkin mengalami kesulitan dalam mendapatkan sinyal satelit.

Server bantuan penyedia data informasi satelit yang dibutuhkan oleh *A-GPS* biasanya didukung oleh jaringan operator karena sering kali menara *BTS* memiliki unit penerima *GPS* dan secara terus-menerus akan mengunduh informasi data satelit dan kemudian memprosesnya. Data dari *server* bantuan bisa diberikan kepada pelanggan telepon seluler, bila diminta oleh perangkat *A-GPS* untuk mengidentifikasi lokasi pengguna berupa *latitude* dan *longitude*, lokasi dalam peta dan lain-lain.[10]

Dalam hal ini dibutuhkan 3 komponen dalam proses penentuan posisi: Satelit, *Assistance server (GSM), Receiver*. Perbedaan *A-GPS* dengan *GPS* hanya pada penambahan *assistance server* saja.

Adapun keuntungan dari *A-GPS* adalah :

1. Pengidentifikasian lokasi lebih cepat.
2. Membutuhkan power lebih kecil dalam proses komputasi data.
3. Cocok untuk lokasi perkotaan atau lokasi yang kurang optimal dalam menangkap sinyal satelit, seperti dalam gedung.

Sedangkan kekurangan dari *A-GPS* yaitu masih tergantung pada *coverage* jaringan operator.

2.5.3 Penentuan Keakuratan Geografis pada *Smartphone*

Dalam menentukan lokasi suatu tempat secara akurat, *smartphone* tidak hanya mengandalkan GPS saja, namun juga membutuhkan bantuan A-GPS. Selain itu ada beberapa metode dalam penentuan keakuratan posisi yang ada pada tabel 2.1.[8]

Tabel 2.1 Tingkat Keakurasian pada berbagai Metode Penempatan

Metode Penempatan	Akurasi	Penggunaan
GPS	± 8 m	Outdoor
Assisted GPS (A-GPS)	5 – 50 m	Indoor dan Outdoor
Cell-id	100 – 3000 m	Indoor dan Outdoor
GSM Cell Tower Triangulation	± 25 m	Indoor dan Outdoor
WLAN Positioning System	20 – 30 m	Indoor dan Outdoor

Smartphone yang beredar saat ini, telah memiliki dukungan WLAN sehingga pembacaan koordinat menjadi lebih akurat dengan toleransi hingga 20 – 30 m dari posisi yang diinginkan dengan data *longitude* dan *latitude* yang diterima oleh *smartphone.*

2.5.4 Pembacaan Data *Longitude* dan *Latitude*

Ada tiga format dalam pembacaan data *longitude* dan *latitude* pada GPS maupun peralatan elektronik lain, seperti pada *google map*, dan lainnya.[9]

Berikut formatnya :

1. Derajat, menit, dan detik (D°M'S")

Koordinat ini paling umum digunakan untuk GPS dengan akurasi. Contoh koordinat lokasi bumi dengan D°M'S" : 12°50'20.1"S 90°14'28.7"E

2. Derajat dan menit desimal (DMM)

Umunya dipakai pada perangkat elektronik. Contoh : 12 20.2220, 3 20.332. Jika Angka 12 adalah derajat, maka setelah 12 adalah angka dari pembagian 60 dari koordinat desimal.

3. Derajat desimal (DD°)

Digunakan untuk koordinat komputer. Paling sederhana dengan memasukan 2 angka koordinat desimal -12.40845, 107.392166.

Contoh perhitungan koordinat D°M'S" (GPS) ke kordinat DD° :
Jika pada layar GPS tertera : 12°27'55.1"S , 92°30'32.6"E, artinya :
S (South / *Latitude*) : 12 derajat 27 menit 55.1 detik
E (Bujur / *Longitude*) : 92 derajat 30 menit 36.2 detik.

Rumus :
Koordinat desimal = derajat + (menit/60) + (detik/3600), maka :
S = 7 + (48/60) + (10.2/3600) = 7.8028333
E =110 + (22/60) + (26.9/3600) = 110.374138

Tambahkan minus (-) pada koordinat *latitude* untuk lokasi yang berada di sebelah selatan garis *equator* (karena tertulis S atau *South* atau Lintang Selatan). Jadi, koordinat desimal yang didapat adalah :
***Latitude* (Lintang)** : -7.8028333
***Longitude* (Bujur)** : 110.374138

2.6 Persamaan Matematika Terhadap Jarak antar Koordinat

Ada beberapa persamaan dalam jarak antar koordinat, diantaranya seperti *Equirectangular Projection, Spherical Law of Cosines,* dan *Haversine.* Dalam tugas akhir ini, persamaan yang akan digunakan adalah *Haversine.*[5]

Haversine

Haversine adalah salah satu persamaan untuk menghitung lingkaran besar antar dua titik yang diibaratkan jarak terpendek dengan permukaan bumi.

Rumus dari *Haversine* adalah sebagai berikut :

$$\boldsymbol{x = (Lon_2 - Lon_1) \times \cos \frac{Lat_1 + Lat_2}{2}}$$

$$\boldsymbol{y = Lat_2 - Lat_1}$$

$$\boldsymbol{d = \sqrt{(x^2 + y^2)} \times R}$$

Keterangan:

Lon = Longitude (Lintang)

Lat = Lattitude (Bujur)

d = Jarak antar kedua koordinat

R = Radius Bumi = 6371 km

1 derajat= 0.0174532925 radian

Contoh Perhitungan Haversine :

Lokasi 1 : lon1= 119.800801, lat1= -0.790175

Lokasi 2 : lon2= 119.8428 , lat2= -0.8989

lat1 = -0.790175 * 0.0174532925 radian = -0.013791155 radian

lon1 = 119.800801 * 0.0174532925 radian = 2.090918422 radian

lat2 = -0.8989 * 0.0174532925 radian = -0.01569 radian

lon2 = 119.8428 * 0.0174532925 radian= 2.091651 radian

x = (lon2-lon1) * cos ((lat1+lat2)/2) = (2.091651-2.090918422) * cos ((-0.013791155 + (-0.01569)/2) = 0.0007329412

y = (lat2-lat1) = (-0.01569- (-0.013791155)) = -0.001897609

d = sqrt (x*x + y*y) * R = sqrt((0.0007329412*0.0007329412) + (-0.001897609*- 0.001897609))*6371 = sqrt (0.0000041381) * 6371 = **12.96012927 km**

2.7 Sistem Berorientasi Terstruktur

Dalam pembuatan suatu program, seorang *programmer* harus membuat dahulu langkah-langkah bagaimana perancangan suatu program yang dibuat. Pada tugas akhir ini, digunakan sebuah metode yang dinamakan Sistem Berorientasi Terstruktur dan diaplikasikan dalam *Data Flow Diagram* (DFD) dan *Entity Relationship Diagram* (ERD).[4]

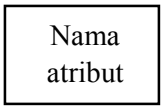
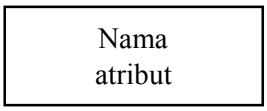
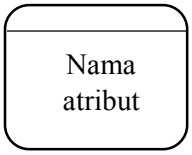
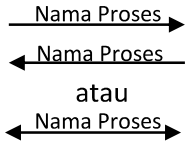
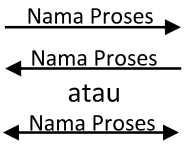
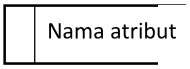
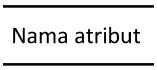
2.7.1 Data Flow Diagram (DFD)

Data Flow Diagram, yang selanjutnya disebut DFD, adalah permodelan dari sistem yang dibuat dan digambarkan ke dalam modul yang berbentuk notasi-notasi yang lebih terstruktur dan jelas sehingga nantinya

dapat dipahami oleh user tentang sistem yang dikerjakan. DFD menggambarkan model proses antara asal data, tujuan data, dan proses yang melibatkan keduanya.

DFD biasanya digambarkan dengan notasi-notasi dan ada beberapa pendapat, yaitu Gane/Sarson dan Yourdon/De Marco. Perbedaan notasi dari keduanya adalah seperti di bawah ini :

Tabel 2.2 Perbedaan Notasi DCD dari Gane/Sarson dan Yourdon/De Marco

Keterangan	Gane/Sarson	Yourdon/De Marco
Orang atau sebuah unit yang berinteraksi dengan proses		
Disebut dengan proses atau apa yang terjadi antara sumber unit dengan tujuan unit		
Aliran data masuk atau pun keluar sesuai mata penunjuknya.		
Simpanan data (data store)		

Dalam DFD ada beberapa tingkatan dalam pembuatan DFD yaitu :

a. *Data Context Diagram* (DCD) adalah level tertinggi dari DFD yang menggambarkan keseluruhan sistem dan penggambaran sistemnya (proses) dibatasi oleh garis putus-putus melingkar.
b. *DFD Level Diagram* (Diagram Rinci) adalah diagram yang menguraikan proses-proses yang ada di dalam proses utamanya (sistem).

2.7.2 Entity Relationship Diagram

Entity Relationship Diagram atau biasa disebut ERD merupakan diagram yang menggambarkan hubungan antara berbagai entity pada suatu sistem. Karena ERD sifatnya konsep, sehingga tidak dapat langsung

diterapkan pada database. Desain dari ERD ini mendekati penglihatan pengguna terhadap data.

ERD ini terdiri dari :

1. ***Entity***. *Entity* adalah obyek yang dapat dibedakan dalam dunia nyata. *Entity sets* adalah kumpulan dari *entity* yang sejenis.

   Entity dapat berupa :

   a. Obyek secara fisik : Rumah, Kendaraan, Peralatan, Mahasiswa.
   b. Obyek secara konsep : Pekerjaan, Perusahaan, Rencana, Matakuliah.
   c. Simbol : persegi panjang

2. ***Atribut***. *Atribut* yaitu karakteristik yang dimiliki oleh sebuah *Entity* atau *Relationship* sehingga dapat membedakan antara *Entity* lain. Biasanya memiliki *primary key*.

   Atribut memiliki berbagai jenis, sebagai berikut :

   a. Atribut sederhana/atomik : atribut yang tidak dapat dibagi-bagi menjadi atribut yang lebih mendasar.

      Contoh : atribut harga dari entity barang.

   b. Atribut komposit : atribut yang terdiri dari beberapa atribut yang lebih mendasar.

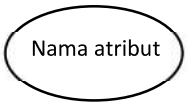
      Contoh : *Entity* mahasiswa memiliki atribut nama yang terdiri dari nama depan (first name), nama tengah (middle name) dan nama belakang (last name).

   c. Atribut berharga tunggal : atribut yang hanya memiliki satu nilai suatu *entity* tertentu. Contoh : atribut umur.
   d. Atribut berharga banyak : atribut yang memiliki beberapa nilai pada suatu *entity* tertentu. Contoh : atribut makanan favorit
   e. Atribut Derivatif : atribut yang berasal dari atribut lain. Contoh : atribut umur yang berdasarkan keterangan lahir.
   f. Atribut kunci : atribut unik yang membedakan antar *entity* atau disebut *primary key.* Selain *primary key*, ada yang disebut *foreign key* (atribut pada *entity* yang menunjuk *primary key* pada entity lain) dan *alternate key* (atribut yang tidak nyata pada suatu *entity* dan dipakai jika tidak ada yang mewakili suatu *entity* tersebut).

3. ***Relationship***. *Relationship* adalah hubungan yang terjadi antara satu atau lebih *entity*. *Relationship sets* adalah kumpulan dari *relationship* yang sejenis.
   Contoh : Karyawan berkerja di kantor. *Relationship* : bekerja

   Berikut adalah simbol-simbol dalam ERD :

   Tabel 2.3 Simbol-simbol dalam ERD

ERD	Simbol
Entity	
Atribut	
Relationship	