BAB 2
TINJAUAN PUSTAKA

2.1 Perencanaan

Pengertian perencanaan menurut Bintoro Tjokroamidjodjo adalah perencanaan dalam arti seluas-luasnya tidak lain adalah suatu proses mempersiapkan secara sistematis kegiatan-kegiatan yang akan dilakukan untuk mencapai suatu tujuan tertentu [10]. Perencanaan adalah proses penetapan dan pemanfaatan sumber-sumber daya secara terpadu sesuai dengan perhitungan dan penentuan yang tepat [11].

Berdasarkan pendapat tersebut, maka perencanaan merupakan suatu usaha atau tindakan-tindakan untuk pelaksanaan kegiatan-kegiatan yang disusun secara sistematis sehingga tujuan dapat dicapai dengan baik.

2.2 Asisten

Menurut KBBI (Kamus Besar Bahasa Indonesia) asisten merupakan seseorang yang membantu orang lain dalam melaksanakan tugas profesional, misalnya dalam pekerjaan, profesi, atau kedinasan.

Di era sekarang ini asisten bukan hanya seseorang yang membantu orang lain, namun asisten juga bisa berupa perangkat lunak yang dapat membantu seseorang dalam melaksanakan kegiatannya.

2.3 Touring

Touring merupakakan kegiatan berpergian dari satu daerah ke daerah lain secara sendiri/berkelompok menggunakan kendaraan bermotor. Touring biasanya dilaksanakan untuk menyalurkan kegemaran berkendara menggunakan sepeda motor.

Touring dapat dibagi menjadi beberapa jenis yaitu:

1. *Touring* berdasarkan TEMPAT/WILAYAH
   Domestik Tour, Overseas Tour, Countryside Tour, Intercity Tour dan Intersland Tour.
2. *Touring* berdasarkan WAKTU
   Halfday Tour, Fullday Tour, Overnight Tour dan Multipleday Tour.

3. *Touring* berdasarkan CARA PENYELENGGARAAN
   Regular Tour dan Irregular Tour.
4. *Touring* berdasarkan TUJUAN
   Educational Tour, Bussiness Tour, Cultural Tour dan Convention Tour.
5. *Touring* berdasarkan SARANA ANGKUTAN
   Overland Tour, Sea Tour, Air Touring dan *Combination.*
6. *Touring* berdasarkan KELAS/TARIF
   Deluxe Tour, First Class Tour, Standar Class Tour, Economy Class Tour dan Budget Class Tour.
7. *Touring* berdasarkan JUMLAH PESERTA
   Individual Tour, Grup Tour dan Mass Tour.
8. *Touring* berdasarkan MINAT
   General Tour dan Special Interest Tour.

2.4 Android

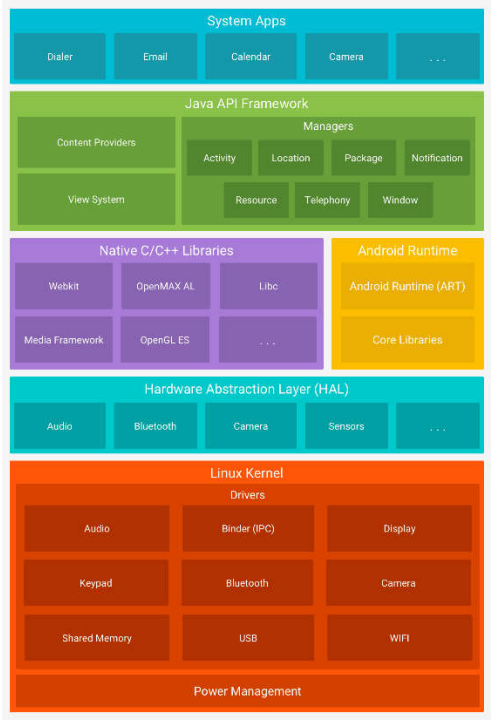
Android adalah sebuah sistem operasi perangkat *mobile* berbasis linux yang mencangkup sistem operasi, *middleware* dan aplikasi. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel atau smartphone. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia [12].

Dengan banyaknya pengguna dan perangkat yang mendukung berjalannya sistem operasi Android, berbanding lurus dengan kebutuhan aplikasi yang diperlukan oleh pengguna. Android merupakan salah satu *mobile operating system* yang saat ini banyak digunakan diberbagai perangkat *smartphone*. Dikarenakan saat ini Android merupakan sistem operasi yang banyak tersebar di masyarakat dan digemari oleh masyarakat, maka banyaknya aplikasi-aplikasi berbasis Android tidaklah aneh lagi [13]. Hingga saat ini Android telah banyak merilis versi-versi sistem operasinya. Berikut merupakan versi-versi Android hingga saat ini:

Tabel 2.1 Versi-versi Android

Versi	Nama Kode	Tanggal Rilis	Level API
1.5	Cupcake	27 April 2009	3
1.6	Donut	15 September 2009	4
2.0 – 2.1	Éclair	26 Oktober 2009	5
2.2 – 2.23	Froyo	20 Mei 2010	8
2.3 – 2.37	Gingerbread	9 Februari 2011	9 – 10
3.0 – 3.2.6	Honeycomb	22 Februari 2011	11 – 13
4.0 – 4.0.4	Ice Cream Sandwich	19 Oktober 2011	15
4.1 – 4.3.1	Jelly Bean	9 Juli 2012	16 – 18
4.4 – 4.4.4	Kitkat	31 Oktober 2013	19 – 20
5.1 – 5.1.1	Lolipop	12 November 2014	21 – 22
6.0 – 6.0.1	Marshmallow	5 Oktober 2015	23
7.1 – 7.1.2	Nougat	22 Agustus 2016	24 – 25
8.0 – 8.1	Oreo	25 Oktober 2017	26 – 27
9	Pie	6 Agustus 2018	28

Sistem Operasi Android memiliki beberapa komponen utama yang disebut dengan Arsitektur Platform Android. Berikut adalah diagram komponen-komponen utama dari platform Android:



Sumber: https://developer.android.com/guide/platform/index.html?hl=id [14]

Gambar 2.1 Arsitektur Platform Android

Berikut adalah penjelasan dari setiap komponen utama arsitektur pada platform android [14]:

1. Linux Kernel

   Fondasi *platform* Android adalah kernel Linux. Contohnya adalah *Android Runtime (ART)* yang bergantung pada kernel Linux untuk fungsionalitas dasar seperti *threading* dan manajemen memori tingkat rendah.

2. Hardware Abstraction Layer

   Hardware Abstraction Layer (HAL) memberikan antarmuka standar yang mengungkap kemampuan perangkat keras perangkat ke kerangka kerja API Java yang lebih tinggi. *HAL* terdiri atas beberapa modul pustaka, masing-masing menerapkan antarmuka untuk komponen perangkat keras tertentu, seperti modul kamera atau *bluetooth*. Ketika *API* kerangka kerja melakukan panggilan untuk mengakses perangkat keras, sistem Android memuat modul pustaka untuk komponen perangkat keras tersebut.

3. Android Runtime

   Untuk perangkat yang menjalankan Android versi 5.0 (API level 21) atau lebih tinggi, setiap aplikasi menjalankan proses masing-masing dengan tahap *Android Runtime (ART)*. *ART* ditulis guna menjalankan beberapa mesin virtual pada perangkat bermemori rendah dengan mengeksekusi file *DEX*, format *bytecode* yang dirancang khusus untuk Android yang dioptimalkan untuk *footprint* memori minimal. Buat rantai aplikasi, misalnya Jack, mengumpulkan sumber *Java* ke *bytecode DEX*, yang dapat berjalan pada *platform* Android.

4. Native C/C++ Libraries

   Banyak komponen dan layanan sistem Android inti seperti *ART* dan *HAL* dibuat dari kode bawaan yang memerlukan pustaka bawaan yang tertulis dalam C dan C++. *Platform* Android memungkinkan kerangka kerja *API Java* meningkatkan fungsi beberapa pustaka bawaan pada aplikasi.

5. Java API Framework

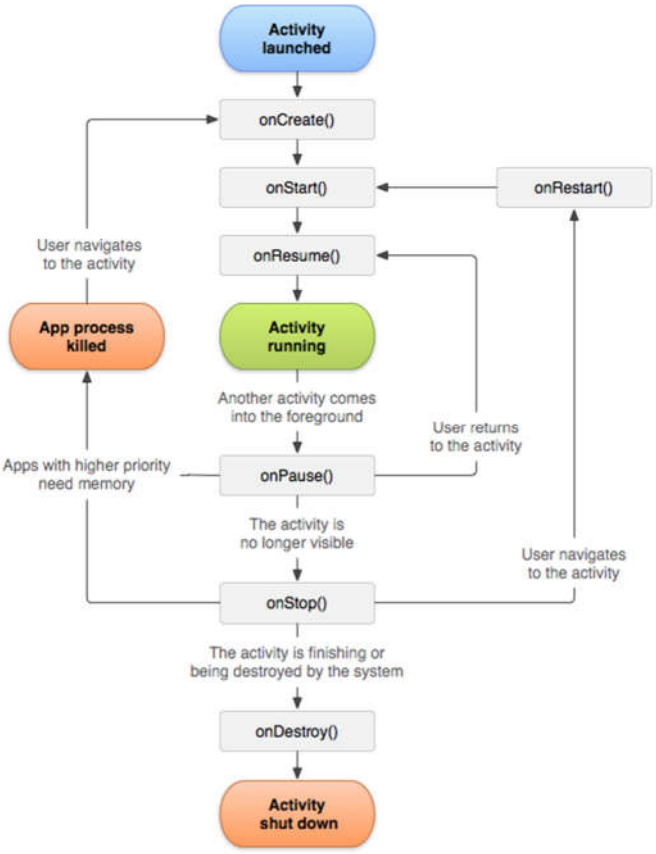
   Keseluruhan rangkaian fitur pada Android OS tersedia untuk melalui *API* yang ditulis dalam bahasa *Java*. *API* ini membentuk elemen dasar yang developer butuhkan untuk membuat aplikasi Android dengan menyederhanakan penggunaan *API* .

6. System Apps

   Android dilengkapi dengan serangkaian aplikasi inti untuk email, perpesanan SMS, kalender, menjelajahi internet, kontak, dll. Aplikasi yang disertakan bersama platform tidak memiliki status khusus pada aplikasi yang ingin dipasang pengguna.

   Aplikasi sistem berfungsi sebagai aplikasi untuk pengguna dan memberikan kemampuan kunci yang dapat diakses oleh developer dari aplikasi mereka sendiri. Misalnya, jika aplikasi Anda ingin mengirimkan pesan SMS, Anda tidak perlu membangun fungsionalitas tersebut sendiri sebagai gantinya Anda bisa menjalankan aplikasi SMS mana saja yang telah dipasang guna mengirimkan pesan kepada penerima yang Anda tetapkan.

Di dalam Sistem operasi Android terdapat siklus hidup dimana pada siklus ini untuk menavigasi antara tahap *Activity Life-Cycle*, Android itu sendiri dengan menyediakan 6 method inti callback yaitu *onCreate(), onStart(), onResume(), onPause(), onStop(),* dan *onDestroy()*. Dalam siklus hidup *method callback*, kita dapat mendeklarasikan cara perilaku *activity* saat pengguna meninggalkan dan memasuki kembali ke sebuah *activity* itu. Disitulah peranan *method callback* dalam sebuah aplikasi pada *platform* Android . Berikut adalah *Activity Life-Cycle* pada *platform* Android:



Sumber: https://developer.android.com/guide/components/activities/activity-lifecycle.html [15]

Gambar 2.2 Siklus Hidup Android

Keenam *method callback* tersebut memiliki pernananya masing-masing dan dijelaskan secara rinci sebagai berikut [15]:

1. onCreate()

   Method ini adalah *method* utama dari setiap *Activity*. *Method* ini akan dipanggil pertama kali ketika menjalankan sebuah sistem. Pengembang harus mengimplementasikan metode *onCreate()* untuk menjalankan logika memulai aplikasi dasar yang hanya boleh terjadi satu kali selama hidup aktivitas. Misalnya, implementasi *onCreate()* Anda harus mendefinisikan antarmuka pengguna dan mungkin membuat *instance* beberapa variabel dalam cakupan-kelas.

2. onStart()

   Method ini dipanggil ketika method *onCreate()* telah dipanggil. *onStart()* dipanggil ketika terlihat oleh *user*. *Method* ini selesai dengan cepat dan dilanjutkan dengan method setelahnya yaitu *onResume()*.

3. onResume()

   Method ini dipanggil ketika *method onStart()* selesai dipanggil. *Method* ini adalah keadaan dimana pengguna berinteraksi dengan aplikasi. Aplikasi akan tetap dalam keadaan ini sampai terjadi suatu *statement* dari aplikasi semisal menerima panggilan telepon atau mematikan layar *smartphone*.

4. onPause()

   Method ini dipanggil ketika pengguna meninggalkan *activity* (meskipun tidak selalu berarti *activity* dihancurkan). *Method* ini berguna untuk menghentikan sementara operasi yang sedang berjalan semisal menjeda pemutaran musik dan lain-lain.

5. onStop()

   Method ini dipanggil ketika *activity* tidak terlihat lagi oleh pengguna, dengan kata lain *activity* berhenti dijalankan. Hal ini dapat terjadi semisal ada aktivitas baru dijalankan meliputi seluruh layar. Sistem juga dapat menghubungi *method* ini ketika *activity* selesai berjalan, dan akan segera dihentikan.

6. onDestroy()

   Method ini adalah *method callback* ketika *activity* telah selesai dijalankan dan kemudian memanggil *method finish()* atau karena sistem untuk sementara menghancurkan proses yang berisi *activity* tersebut untuk menghemat ruang memori.

2.5 Android Studio

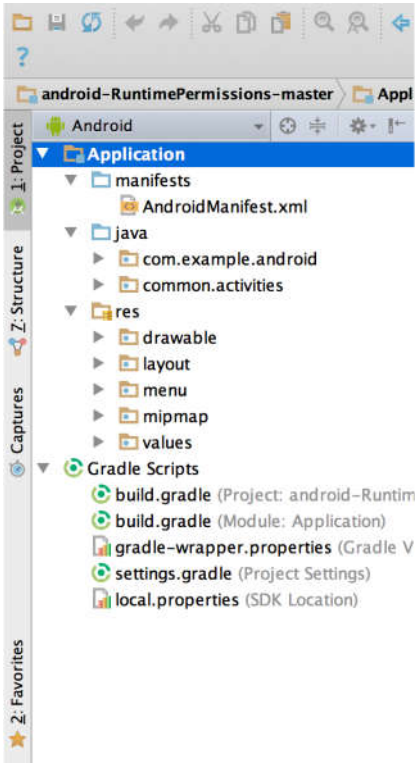
Android Studio dirilis sejak Mei 2013 dan dikembangkan secara terpisah dari aplikasi Eclipse ADT yang dirilis terlebih dahulu sejak android diluncurkan sebagai sistem operasi *open source*. Android Studio dikembangkan berbasis teknologi IntelliJ IDEA untuk penciptakan sistem yang lebih terintegrasi serta konsumsi

memori lebih efisien. Improvisasi paling signifikan adalah adanya dukungan konfigurasi berbasis *tool gradle* yang memberikan kemudahan *programmer* saat proses *versioning*, dukungan membangun *project*, dan ketidak tergantungan pada pustaka antar *project* secara otomatis serta teknologi DSL berbasis bahasa Groovy [16].

Android Studio mempunyai struktur project yang berisi satu atau beberapa modul dengan file kode sumber dan file sumber daya. Jenis-jenis modul mencakup [17]:

1. Modul Aplikasi Android
2. Modul Perpustakaan
3. Modul Google App Engine

Pada tampilan IDE Android studio, struktur project dapat terlihat pada bagian kiri IDE seperti Gambar berikut:



Sumber: https://developer.android.com/studio/intro/index.html?hl=id [17]

Gambar 2.3 Struktur *Project* Android Studio

2.6 Flutter

Flutter adalah teknologi untuk membangun suatu *mobile apps* yang dibuat oleh Google. Flutter dapat digunakan untuk membuat aplikasi Android dan iOS sekaligus. Flutter merupakan *cross-platform framework*, alias aplikasi yang dapat digunakan di lebih dari satu *platform*. Aplikasi yang dibuat dengan menggunakan Flutter dapat dijalankan baik di *platform* Android maupun iOS.

Flutter memiliki beberapa keunggulan diantaranya:

1. Flutter memungkinkan kita untuk membuat aplikasi yang indah (*beautiful*).
2. Flutter berjalan dengan sangat cepat (*fast*)
3. Flutter sangat produktif (*productive*)
4. Flutter bersifat terbuka (*open*)

2.7 Dart

Dart adalah bahasa pemrograman yang dikembangkan oleh Google untuk kebutuhan umum *(general-purpose programming language)*. Dart bisa digunakan untuk membuat aplikasi android, *front-end web*, IoT, *backend* (CLI), dan Game.

Berikut merupakan beberapa keunggulan dart:

1. Dioptimalkan untuk UI
2. Membuat pengembangan aplikasi lebih produktif
3. Cepat untuk semua *platform*

2.8 Javascript Object Notation (JSON)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data [18]. 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 pemrograman yang juga berdasarkan pada struktur data ini.

2.9 Kotlin

Kotlin adalah bahasa pemrograman berbasis *Java Virtual Machine* (JVM) yang dikembangkan oleh Jetbrains, perusahaan yang membuat IntelliJ IDEA. Kotlin merupakan bahasa pemrograman yang pragmatis untuk android yang mengkombinasikan *Object Oriented* (OO) dan pemrograman fungsional. Kotlin juga bahasa yang interpolabilitas yang membuat bahasa ini dapat digabungkan dalam satu proyek dengan bahasa Java [19]. Kotlin memiliki dua fitur yang menarik bagi pengembang android:

1. Kotlin sangat intuitif dan mudah dipelajari oleh para pengembang yang sebelumnya menggunakan bahasa pemrograman Java.
2. Terintegrasi dengan IDE Android Studio.

Dibandingkan dengan Java, Kotlin memiliki beberapa kenuggulan diantaranya:

1. Penulisan kode lebih ringkas dan mudah dibaca dibandingkan kode yang ditulis dengan bahasa pemrograman Java.
2. Dapat mengatasi *NullPoiterException* yang biasanya terjadi dalam bahasa pemerograman Java.

2.10 Hypertext Preprocessor (PHP)

PHP adalah bahasa pemrograman *server side scripting* yang menyatu dengan HTML untuk membuat halaman web yang dinamis [20]. Karena PHP merupakan

server side scripting maka sintaks dan perintah-perintah PHP akan dieksekusi di server kemudian hasilnya akan dikirimkan ke browser dengan format HTML.

PHP dapat berjalan diberbagai web server IIS *(Internet Information Server)*, PWS *(Personal Web Server)*, Apache, Xitami. PHP juga mampu berjalan di banyak sistem operasi yang beredar saat ini, diantaranya : Sistem Operasi Microsoft Windows (semua versi), Linux, Mac Os, Solaris. PHP dapat dibangun sebagai modul *web server* Apache dan sebagai *binary* yang dapat berjalan sebagai CGI *(Common Gateway Interface)*. PHP mempunyai koneksitas yang baik dengan beberapa DBMS seperti Oracle, Sybase, mSQL, MySQL, Microsoft SQL Server, Solid, PostgreSQL, Adabas, FilePro, Velocis, dBase, Unix dbm, dan tidak terkecuali semua *database* ber-*interface* ODBC.

Hampir seluruh aplikasi berbasis web dapat dibuat dengan PHP. Namun kekuatan utama adalah konektivitas basis data dengan web. Dengan kemampuan ini kita akan mempunyai suatu sistem basis data yang dapat diakses.

2.11 CodeIgniter

CodeIgniter adalah sebuah *framework* PHP yang dapat membantu mempercepat *developer* dalam pengembangan aplikasi web berbasis PHP dibanding menulis semua kode program dari awal [21].

Sumber: Buku Membangun Web Berbasis PHP dengan Framework CodeIgniter [21]

Gambar 2.4 Logo CodeIgniter

CodeIgniter pertama kali dibuat oleh Rick Ellis, CEO Ellislab, Inc, sebuah perusahaan yang memproduksi CMS *(Content Management System)* yang cukup handal, yaitu *Expression Engine*. Saat ini, CodeIgniter dikembangkan dan dimaintain oleh *Expression Engine Development Team*.

Adapun beberapa keuntungan menggunakan CodeIgniter, diantaranya:

1. Gratis

   CodeIgniter berlisensi dibawah Apache/BSD *opensource*.

2. Berukuran Kecil

   Ukuran CodeIgniter yang kecil merupakan keunggulan tersendiri. dibanding dengan *framework* lain yang berukuran besar.

3. Menggunakan Konsep MVC

   CodeIgniter menggunakan konsep MVC yang memungkinkan pemisahan *layer application-logic* dan *presentation*.

4. URL yang Sederhana

   Secara default, URL yang dihasilkan CodeIgniter sangat bersih dan *Serach Engine Friendly* (SEF).

5. Memiliki Paket Library yang Lengkap

   CodeIgniter mempunyai *library* yang lengkap untuk mengerjakan operasi-operasi yang umum dibutuhkan oleh sebuah aplikasi berbasis web, misalnya mengakses *database*, mengirim email, memvalidasi *form*, menangani *session* dan sebagainya.

6. Extensible

   Sistem dapat dikembangkan dengan mudah menggunakan *plugin* dan *helper*, atau dengan menggunakan *hooks*.

7. Tidak Memerlukan Template Engine

   Meskipun CodeIgniter dilengkapi dengan *template parser* sederhana yang dapat digunakan, tetapi hal ini tidak mengharuskan kita untuk menggunakannya.

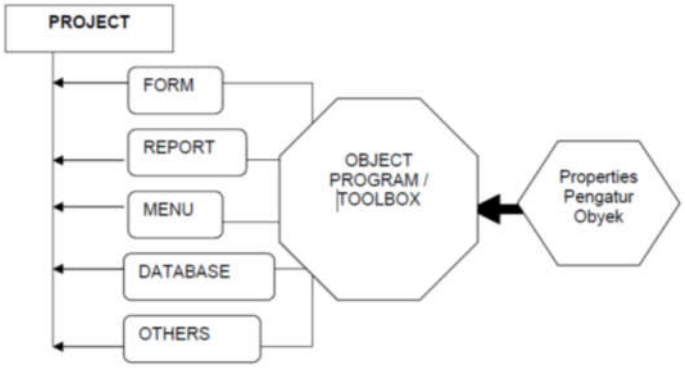
8. Dokumentasi Lengkap dan Jelas

   Dari sekian banyak *framework*, CodeIgniter adalah satu-satunya *framework* dengan dokumentasi yang lengkap dan jelas.

2.12 Object Oriented Programming (OOP)

Pemrograman berorientasi objek atau *object-oriented programming* merupakan suatu pendekatan pemrograman yang menggunakan *object* dan *class*. OOP merupakan cara pandang dalam menganalisa sistem dan permasalahan

pemrograman. Dalam OOP, setiap bagian dari program adalah *object*. Sebuah *object* mewakili suatu bagian program yang akan diselesaikan [22].



Sumber: E-Journal Infokom (2013) [22]

Gambar 2.5 Model Pemrograman *OOP*

2.13 Database

Database merupakan kumpulan koleksi data-data yang saling berhubungan secara logika yang isinya didesain untuk memenuhi kebutuhan informasi [23].

Ada beberapa istilah umum yang sering dipakai pada *database*, yaitu sebagai berikut:

1. *Field*, yaitu sekumpulan kecil dari kata atau sebuah deretan angka-angka.
2. *Record*, yaitu kumpulan dari *field* yang berelasi secara logis.
3. *File*, yaitu kumpulan dari *record* yang berelasi secara logis.
4. *Entity*, yaitu orang, tempat, benda, atau kejadian yang berkaitan dengan informasi yang disimpan.
5. *Attribute*, yaitu setiap karakteristik yang menjelaskan suatu *entity*.
6. *Primary key*, yaitu sebuah *field* yang nilainya unik yang tidak sama antara satu *record* dengan *record* yang lain.
7. *Foreign key*, yaitu sebuah field yang nilainya berguna untuk menghubungkan *primary key* yang berada pada tabel yang berbeda.

2.14 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL *(Database Management System)* atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL merupakan *Relational Database*

Management System (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL *(General Public License)* dimana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL *(Structured Query Language)* [24].

Dalam penggunaan *database* MySQL, setiap perintah yang diketikkan disebut *query*. Perintah MySQL dapat dikategorikan menjadi 3 sub perintah, yaitu DDL *(Data Definition Language)*, DML *(Data Manipulation Language)*, dan DCL *(Data Control Language)*.

1. DDL *(Data Definition Language)*

   Perintah dalam SQL yang pertama adalah perintah DDL. DDL sendiri mrupakan kependekan dari apa yang dikenal dengan nama *Data Definition Language*. DDL dapat berarti sebuah perintah yang berhubungan dengan pendefinisian dari suatu struktur database. Terdapat beberapa perintah DDL pada MySQL sebagai berikut:

   a. *CREATE* berfungsi untuk membuat *database* baru, tabel baru, *view* baru dan kolom.
   b. *ALTER* berfungsi untuk mengubah struktur tabel. Seperti mengganti nama tabel, menambah kolom, mengubah kolom, menghapus kolom maupun memberikan atribut pada kolom.
   c. *DROP* berfungsi untuk menghapus *database* dan tabel.
   d. *TRUNCATE* berfungsi untuk menghapus semua catatan dari tabel.
   e. *COMMENT* berfungsi untuk menambahkan komentar pada data.
   f. *RENAME* berfungsi untuk mengubah nama obyek.

   Adapun contoh sintaks dari perintah DDL pada MySQL sebagai berikut:

```
CREATE DATABASE NILAI;
ALTER TABLE Mahasiswa ADD (NoTelp char(8));
DROP INDEX nama_index ;
TRUNCATE TABLE table_barang;
```

```
RENAME table_barang to table_barang_gudang;
```

2. DML *(Data Manipulation Language)*

Data Manipulation Language (DML) ialah perintah yang digunakan untuk mengelola/memanipulasi data dalam *database*. Terdapat beberapa perintah DML pada MySQL sebagai berikut:

a. *SELECT* berfungsi untuk mengambil/menampilkan data dari *database*.
b. *INSERT* berfungsi untuk memasukkan data ke dalam tabel.
c. *UPDATE* berfungsi untuk memperbarui data dalam tabel.
d. *DELETE* berfungsi untuk menghapus data dari tabel.
e. *CALL* berfunsi untuk memanggil subprogram PL / SQL atau Java.
f. *EXPLAIN PLAN* berfungsi untuk menjelaskan jalur akses ke data.
g. *LOCK TABLE* berfungsi untuk mengunci tabel.

Adapun contoh sintaks dari perintah DML pada MySQL sebagai berikut:

```
INSERT INTO mahasiswa VALUES("08052926",
"Frenky","70");
SELECT nama_mahasiswa FROM mahasiswa;
DELETE FROM mahasiswa;
UPDATE mahasiswa SET nama_mahasiswa='ujang' WHERE
nim='08052926';
```

3. DCL *(Data Control Language)*

Data Control Language (DCL) ialah perintah yang digunakan untuk melakukan pengontrolan data dan server *database*-nya. Terdapat beberapa perintah DCL pada MySQL sebagai berikut:

a. *GRANT* berfungsi untuk memberikan hak akses pengguna ke *database*.
b. *REVOKE* berfungsi untuk menghilangkan hak akses yang telah diberikan dengan perintah *GRANT* [25].

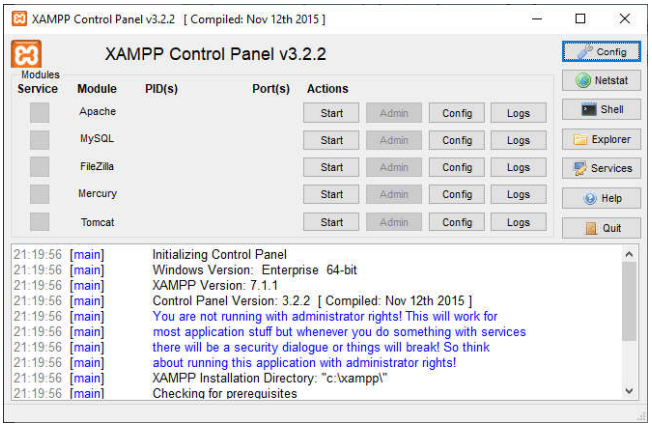
Adapun contoh sintaks dari perintah DML pada MySQL sebagai berikut:

```
GRANT all privileges on * to nm_user@localhost
identified by 'nm_passwd' with grand option;
REVOKE all on nm_db.nm_tbl from nm_user@localhost
identified by 'nm_passwd';
```

2.15 XAMPP

XAMPP adalah sebuah *software web server* apache yang didalamnya sudah tersedia *database* server MySQL dan dapat mendukung pemrograman PHP. XAMPP merupakan *software* yang mudah digunakan, gratis dan mendukung instalasi di Linux dan Windows. Keuntungan lainnya adalah cuma menginstal satu kali sudah tersedia Apache Web Server, MySQL Database Server, PHP Support (PHP 4 dan PHP 5) dan beberapa module lainnya [26].

Fungsi XAMPP sendiri adalah sebagai *server* yang berdiri sendiri *(localhost)*, yang terdiri beberapa program antara lain : Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Nama XAMPP sendiri merupakan singkatan dari X (empat sistem operasi apapun), Apache, MySQL, PHP dan Perl. Program ini tersedia dalam GNU *(General Public License)* dan bebas, merupakan *web server* yang mudah untuk digunakan yang dapat menampilkan halaman web yang dinamis.



Gambar 2.6 Aplikasi XAMPP

2.16 Web Server

Web Server adalah sebuah perangkat lunak yang bertugas menerima permintaan client melalui port HTTP maupun HTTPS dan merubahnya ke dalam format HTML. Terdapat beberapa format selain HTML yaitu PHP atau ASP, tetapi format – format tersebut hanyalah berfungsi untuk menghubungkan HTML dengan *database*.

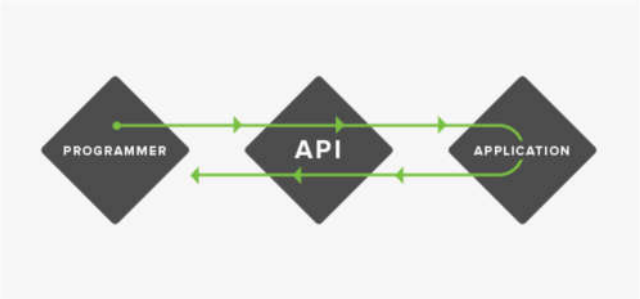
Web server yang umum diguanakan adalah Apache. Tugas utama dari *web server* adalah menerjemahkan permintaan ke dalam respon yang cocok untuk keadaan pada saat itu, ketika klien membuka komunikasi dengan Apache, Apache mengirimkan permintaan untuk sumber daya. Apache menyediakan sumber daya yang baik atau memberikan respon alternatif untuk menjelaskan mengapa permintaan tidak dapat terpenuhi. Setiap komunikasi HTTP dimulai dengan permintaan dan berakhir dengan jawaban. Executable Apache mengambil nama dari protokol, dan pada sistem Unix umumnya disebut httpd, kependekan daemon HTTP [27].

2.17 Application Programming Interface (API)

API adalah antarmuka yang digunakan untuk mengakses aplikasi atau layanan dari sebuah program. API memungkinkan pengembang untuk memakai fungsi yang sudah ada dari aplikasi lain sehingga tidak perlu membuat ulang dari awal. Pada konteks web, API merupakan pemanggilan fungsi lewat *Hyper Text Transfer Protocol* (HTTP) dan mendapatkan respon berupa *Extensible Markup Language* (XML) atau *JavaScript Object Notation* (JSON). Pemanggilan fungsi ke suatu situs tertentu akan menghasilkan respon yang berbeda kepada pengguna untuk membangun aplikasi *enterprise* di dalam *website*-nya [28].

Dengan API, panggilan-panggilan yang bolak-balik antar aplikasi diatur melalui *web service*. *Web service* adalah kumpulan standar teknis dan protokol, termasuk XML *(Extensible Markup Language)*, bahasa umum yang digunakan oleh aplikasi-aplikasi tersebut selama berkomunikasi di internet. API dan *web service* sepenuhnya bekerja di belakang layar.

Dengan demikian, API adalah standar komunikasi yang dibuka oleh perusahaan *software*, agar dapat dimanfaatkan oleh pengembang pihak ketiga untuk mendesain aplikasi yang memanfaatkan layanan mereka dengan mudah.



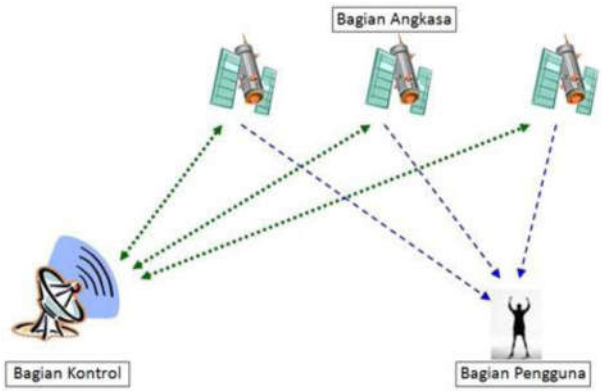
Sumber: sproutsocial.com

Gambar 2.7 Ilustrasi *API*

2.18 Global Positioning System (GPS)

Global Positioning System atau yang biasa disingkat dengan GPS adalah alat navigasi elektronik yang menerima informasi dari 4 - 12 satelit sehingga GPS bisa memperhitungkan posisi dimana kita berada di Bumi. Satelit GPS tidak mentransmisikan informasi posisi kita, yang ditransmisikan satelit adalah posisi satelit dan jarak penerima GPS kita dari satelit [29].

GPS terdiri dari tiga bagian yaitu sistem kontrol, satelit dan pengguna. Sistem kontrol adalah bagian yang mengontrol pergerakan satelit-satelit yang ada dan saling berinteraksi satu sama lain kemudian pengguna adalah alat navigasi yang digunakan seperti perangkat *mobile* yang kini sudah memiliki fitur GPS di dalamnya. GPS biasanya digunakan 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 dapat menghambat sinyal untuk mendapatkan lokasi pengguna [30].



Sumber: nusantride.com

Gambar 2.8 Cara Kerja *GPS*

1. GPS Control Segment

   Control segment GPS terdiri dari lima stasiun yang berada di pangkalan Falcon Air Force, Colorado Springs, Ascension Island, Hawaii, Diego Garcia dan Kwajalein. Kelima stasiun ini adalah mata dan telinga bagi GPS. Sinyal-sinyal dari satelit diterima oleh bagian kontrol, kemudian dikoreksi, dan dikirimkan kembali ke satelit. Data koreksi lokasi yang tepat dari satelit ini disebut data *ephemeris*, yang kemudian nantinya dikirimkan ke alat navigasi yang kita miliki.

2. GPS Space Segment

   Space Segment adalah terdiri dari sebuah jaringan satelit yang tediri dari beberapa satelit yang berada pada orbit lingkaran yang terdekat dengan tinggi nominal sekitar 20.183 km di atas permukaan bumi. Sinyal yang dipancarkan oleh seluruh satelit tersebut dapat menembus awan, plastik dan kaca, namun tidak bisa menembus benda padat seperti tembok dan rapatnya pepohonan. Terdapat 2 jenis gelombang yang hingga saat ini digunakan sebagai alat navigasi berbasis satelit. Masing-masingnya adalah gelombang L1 dan L2, dimana L1 berjalan pada frekuensi 1575.42 MHz yang bisa digunakan oleh masyarakat umum, dan L2 berjalan pada frekuensi 1227.6 Mhz dimana jenis ini hanya untuk kebutuhan militer saja.

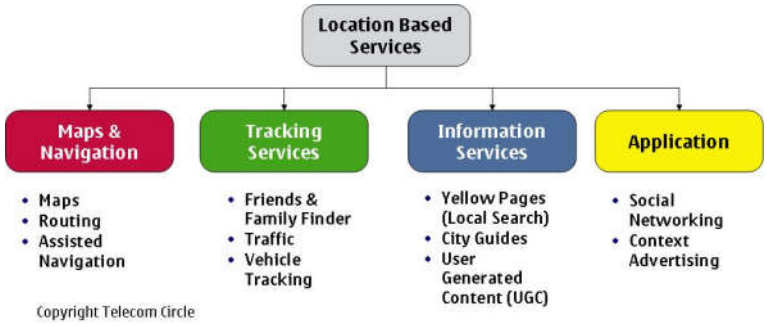
3. GPS User Segment

User segment terdiri dari antena dan *prosesor receiver* yang menyediakan *positioning*, kecepatan dan ketepatan waktu ke pengguna. Bagian ini menerima data dari satelit-satelit melalui sinyal radio yang dikirimkan setelah mengalami koreksi oleh stasiun pengendali *(GPS Control Segment)*.

2.19 Location Based Service (LBS)

Location Based Service merupakan layanan informasi yang dapat diakses menggunakan *mobile devices*, yang dilengkapi kemampuan untuk mengetahui keberadaan lokasi dari si pengguna perangkat dan kemampuan memberikan informasi mengenai layanan yang tersedia berdasarkan lokasi mereka pada saat itu. Menurut Schiller J, *Location Based Service* dapat didefinisikan sebagai "layanan yang mengintegrasikan lokasi perangkat *mobile* atau posisi dengan informasi lain sehingga dapat memberikan nilai tambah bagi pengguna" [31].

Layanan berbasis lokasi dapat diklasifikasikan ke dalam empat jenis yaitu *Maps* dan *Navigation*, *Tracking Services, Information Services* dan *Application* seperti yang digambarkan dalam gambar berikut:



Sumber: http://www.telecomcircle.com/2009/06/introduction-to-lbs/

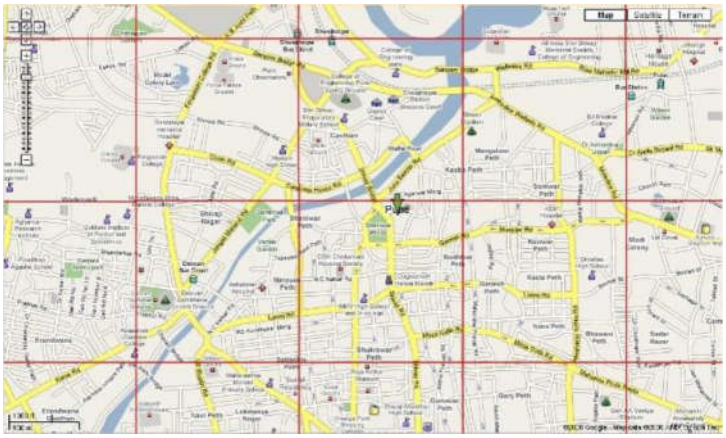
Gambar 2.9 Klasifikasi *LBS*

2.20 Google Maps

Google Map Service adalah sebuah jasa peta global virtual gratis dan online yang disediakan oleh perusahaan Google. Google Maps yang dapat ditemukan di website resmi *developer* Google. Google Maps menawarkan peta yang dapat diseret dan gambar satelit untuk seluruh dunia. Google Maps juga menawarkan pencarian suatu tempat dan rute perjalanan. Google Maps API adalah sebuah layanan *(service)* yang diberikan oleh Google kepada para pengguna untuk memanfaatkan

Google Maps dalam mengembangkan aplikasi. Google Maps API menyediakan beberapa fitur untuk memanipulasi peta, dan menambah konten melalui berbagai jenis *services* yang dimiliki [32].

Google Maps dibuat dengan menggunakan kombinasi dari gambar peta, *database*, serta obyek obyek interaktif yang dibuat dengan bahasa pemrograman HTML, Javascript dan AJAX, serta beberapa bahasa pemrograman lainnya. Gambar-gambar yang muncul pada peta merupakan hasil komunikasi dengan *database* pada *web server* Google untuk menampilkan gabungan dari potongan-potongan gambar yang diminta. Keseluruhan citra yang ada diintegrasikan ke dalam *database* pada Google Server, yang nantinya akan dapat dipanggil sesuai kebutuhan permintaan. Bagian-bagian gambar map merupakan gabungan dari potongan gambar-gambar bertipe PNG yang disebut *tile* yang berukuran 256 x 256 pixel seperti gambar berikut [33]:



Sumber : ankitjain.info

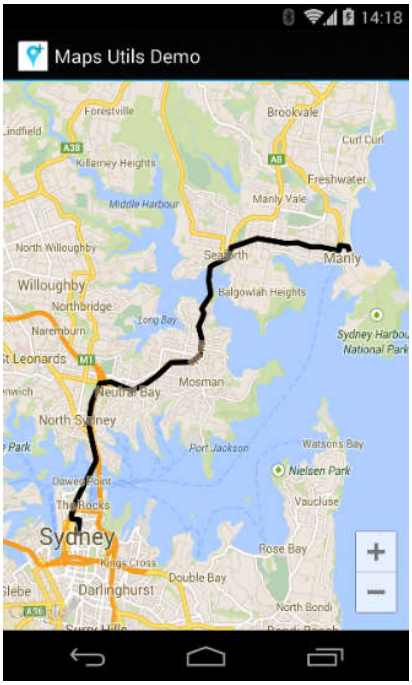
Gambar 2.10 *Google Maps Tile*

2.21 Google Maps Android API

Google Maps Android API adalah layanan untuk menampilkan peta di aplikasi android. Pengembang dapat menambahkan peta ke aplikasi berdasarkan data di Google Maps API secara otomatis menangani akses ke *server* Google Maps, mengunduh data, menampilkan peta, dan merespons gerakan peta. Anda juga bisa menggunakan panggilan API untuk menambahkan *marker*, poligon, dan *overlay* ke peta dasar, serta mengubah tampilan area peta tertentu ke pengguna. Semua objek ini memberikan informasi tambahan tentang lokasi peta, dan memungkinkan

interaksi pengguna dengan peta. API ini memungkinkan pengembang untuk menambahkan semua hal seperti berikut:

1. Ikon yang dikaitkan dengan posisi tertentu pada peta *(Marker)*.
2. Rangkaian segmen garis *(Polyline)*.
3. Segmen yang disertakan *(Polygon)*.
4. Gambar bitmap yang dikaitkan dengan posisi tertentu pada peta (*Overlay* Bumi).
5. Rangkaian gambar yang ditampilkan di bagian atas petak peta dasar (*Overlay* Petak) [34]



Sumber: https://developers.google.com/maps/documentation/android-sdk/images/utility-polycoding.png

Gambar 2.11 *Google Maps* di Android

2.22 Google Maps Direction API

Google Maps Directions API adalah layanan yang menghitung arah antar lokasi menggunakan permintaan HTTP. Dengan Google Maps Direction API pengembang dapat membangun fitur untuk mencari arah untuk beberapa moda transportasi, termasuk angkutan umum, mengemudi, berjalan atau bersepeda.

Google Maps Directions API bisa mengembalikan arah *multi*-bagian menggunakan serangkaian titik jalan [35].

Layanan ini biasanya didesain untuk menghitung arah alamat statis (sudah diketahui sebelumnya) untuk penempatan konten aplikasi pada peta; layanan ini tidak didesain untuk merespon masukkan pengguna secara *realtime*, misalnya. Untuk perhitungan arah dinamis (misalnya, dalam elemen antarmuka pengguna) [35].

Untuk menggunakan Google Maps Direction API ini, pengembang dapat memanggil URL:

```
https://maps.googleapis.com/maps/api/directions/outpu
t?parameters
```

Berikut merupakan contoh respon rute yang diberikan oleh Google Direction API.

```
{
   "geocoded_waypoints": [
      {
         "geocoder_status": "OK",
         "place_id": "ChIJndywNTbkaC4RGjiEIr70n6M",
         "types": [
            "establishment",
            "food",
            "point_of_interest",
            "restaurant"
         ]
      },
      {
         "geocoder_status": "OK",
         "place_id": "ChIJF5BgeTbkaC4RsjRIxYlaTv8",
         "types": [
            "car_repair",
            "establishment",
            "point_of_interest"
         ]
      }
   ],
   "routes": [
      {
```

```
         "bounds": {
            "northeast": {
               "lat": -6.881748699999999,
               "lng": 107.5528615
            },
            "southwest": {
               "lat": -6.883803400000001,
               "lng": 107.5516376
            }
         },
         "copyrights": "Map data ©2020",
         "legs": [
            {
               "distance": {
                  "text": "0,3 km",
                  "value": 266
               },
               "duration": {
                  "text": "1 min",
                  "value": 43
               },
               "end_address": "Jl. Jend. H. Amir Machmud No.364, Cibabat, Kec. Cimahi Tengah, Kota Cimahi, Jawa Barat 40513, Indonesia",
               "end_location": {
                  "lat": -6.883803400000001,
                  "lng": 107.5528615
               },
               "start_address": "Jl. Raya Cibabat No.114, Cibabat, Kec. Cimahi Utara, Kota Cimahi, Jawa Barat 40522, Indonesia",
               "start_location": {
                  "lat": -6.881748699999999,
                  "lng": 107.5516376
               },
               "steps": [
                  {
                     "distance": {
                        "text": "0,3 km",
                        "value": 266
                     },
                     "duration": {
                        "text": "1 min",
                        "value": 43
                     },
                     "end_location": {
                        "lat": -6.883803400000001,
```

```
                        "lng": 107.5528615
                     },
                     "html_instructions": "Ambil arah
<b>tenggara</b> di <b>Jl. Nasional III</b> menuju
<b>Jl. Karya Bhakti VI</b><div style=\"font-
size:0.9em\">Lewati Alfamart Cibabat 359 (di
kanan)</div><div style=\"font-size:0.9em\">Tujuan ada
di sebelah kiri.</div>",
                     "polyline": {
                        "points":
"|a_i@wdmoSj@WRIbAq@p@]TOj@Qz@a@NIb@UJGd@U"
                     },
                     "start_location": {
                        "lat": -6.881748699999999,
                        "lng": 107.5516376
                     },
                     "travel_mode": "DRIVING"
                  }
               ],
               "traffic_speed_entry": [],
               "via_waypoint": []
            }
         ],
         "overview_polyline": {
            "points":
"|a_i@wdmoS~@a@tBoATOj@QjAk@tAs@"
         },
         "summary": "Jl. Nasional III",
         "warnings": [],
         "waypoint_order": []
      }
   ],
   "status": "OK"
}
```


2.23 Firebase

Firebase memiliki produk utama, yaitu menyediakan *database realtime* dan *backend* sebagai layanan *(Backend as a Service)*. Layanan ini menyediakan pengembang aplikasi API yang memungkinkan aplikasi data yang akan disinkronisasi di klien dan disimpan di *cloud* Firebase ini. Firebase menyediakan *library* untuk berbagai *client platform* yang memungkinkan integrasi dengan Android, iOS, JavaScript, Java, Objective-C dan Node aplikasi Js dan dapat juga

disebut sebagai layanan DbaaS *(Database as a Service)* dengan konsep *realtime*, Firebase digunakan untuk mempermudah dalam penambahan fitur-fitur yang akan dibangun oleh *developer*.

Firebase memiliki beberapa fitur diantaranya:

1. *Firebase Analytics.*
2. *Firebase Cloud Messaging and Notification.*
3. *Firebase Authentication.*
4. *Firebase Remote Config.*
5. *Firebase Realtime Database.*
6. *Firebase Crash Reporting.*

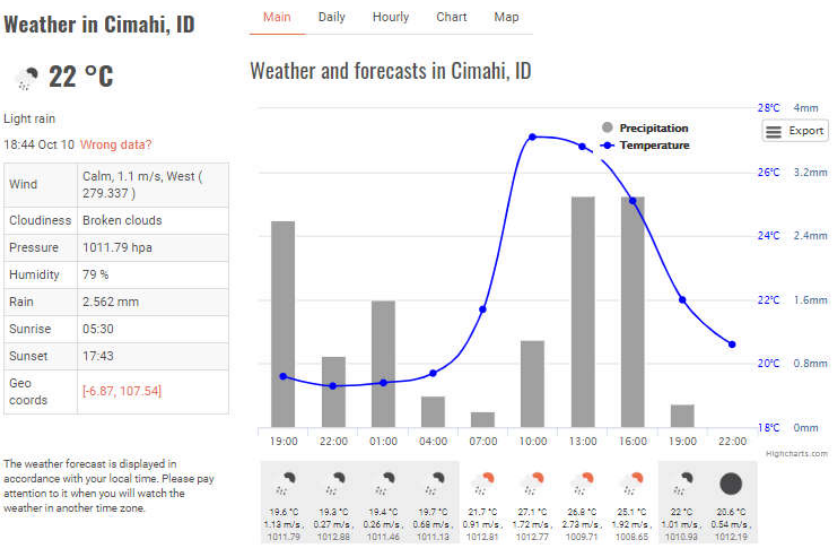
Sumber: https://firebase.google.com/brand-guidelines

Gambar 2.12 Logo Firebase

2.24 OpenWeatherMap

OpenWeatherMap adalah layanan online yang menyediakan data cuaca terbaru yang mencakup perkiraan cuaca dan data histori cuaca. OpenWeatherMap juga menyediakan *service* yang dapat digunakan oleh pengembang aplikasi untuk menggunakan data yang diperoleh.

Data yang didapatkan oleh OpenWeatherMap bersumber dari *meteorogical broadcast service*, data stasiun cuaca bandara, data stasiun radar, dan data lain dari statsiun cuaca resmi [36].



Sumber: https://openweathermap.org/

Gambar 2.13 Contoh Data Cuaca *OpenWeatherMap*

Berikut merupakan contoh respon prediksi cuaca yang diberikan oleh OpenWeatherMap API.


```
{
    "cod": "200",
    "message": 0,
    "cnt": 40,
    "list": [
        {
            "dt": 1578430800,
            "main": {
                "temp": 293.6,
                "feels_like": 295.63,
                "temp_min": 293.6,
                "temp_max": 293.6,
                "pressure": 1008,
                "sea_level": 1008,
                "grnd_level": 905,
                "humidity": 82,
                "temp_kf": 0
            },
            "weather": [
                {
                    "id": 804,
                    "main": "Clouds",
                    "description": "overcast clouds",
                    "icon": "04n"
                }
```

```
            ],
            "clouds": {
                "all": 85
            },
            "wind": {
                "speed": 0.65,
                "deg": 244
            },
            "sys": {
                "pod": "n"
            },
            "dt_txt": "2020-01-07 21:00:00"
        },
        {
            "dt": 1578441600,
            "main": {
                "temp": 295.13,
                "feels_like": 297.14,
                "temp_min": 295.13,
                "temp_max": 295.13,
                "pressure": 1010,
                "sea_level": 1010,
                "grnd_level": 907,
                "humidity": 77,
                "temp_kf": 0
            },
            "weather": [
                {
                    "id": 803,
                    "main": "Clouds",
                    "description": "broken clouds",
                    "icon": "04d"
                }
            ],
            "clouds": {
                "all": 78
            },
            "wind": {
                "speed": 0.97,
                "deg": 280
            },
            "sys": {
                "pod": "d"
            },
            "dt_txt": "2020-01-08 00:00:00"
        }
    ],
```

```
    "city": {
        "id": 1646448,
        "name": "Cimahi",
        "coord": {
            "lat": -6.8723,
            "lon": 107.5425
        },
        "country": "ID",
        "population": 493698,
        "timezone": 25200,
        "sunrise": 1578436856,
        "sunset": 1578481854
    }
}
```


2.25 Object Oriented Analysis and Design (OOAD)

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 [37].

Keuntungan menggunakan OOAD adalah memudahkan dalam memecah sistem yang kompleks menjadi lebih kecil, modul aplikasi lebih mudah dikelola, mudah merangkai modul kembali bersama-sama untuk membentuk suatu sistem informasi, menghemat waktu, meningkatkan interaksi antara pengguna dan analis. Tools yang digunakan untuk pengembangan sistem dengan pendekataan berorientasi objek ini adalah dengan menggunakan diagram UML.

2.26 Unified Modeling Language (UML)

Unified Modeling Language (UML) merupakan himpunan struktur dan teknik untuk pemodelan dan desain program berorientasi objek (OOP) serta aplikasinya. UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok *tool* untuk mendukung pengembangan sistem tersebut. UML mulai diperkenalkan oleh Object Management Group, sebuah organisasi yang telah mengembangkan model, teknologi, dan standar OOP sejak tahun 1980-an. Sekarang UML sudah mulai

banyak digunakan oleh para praktisi OOP. UML merupakan dasar bagi *tool* desain berorientasi objek dari IBM [38].

2.26.1 Use Case Diagram

Use case adalah teknik berdasarkan seknario untuk elistisasi persyaratan yang pertama kali diperkenalkan pada metode *Objectory*. Use case sekarang telah menjadi fitur dasar notasi UML untuk mendeskripsikan model sistem berorientasi objek. Dalam bentuknya yang paling sederhana, *use case* mengidentifikasi aktor yang terlibat dalam interaksi dan nama tipe interaksi tersebut [39].

Use case diagram adalah diagram yang menjelaskan interaksi sebuah sistem dengan aktor yang dinilai terlibat dengan sistem yang nantinya akan dibuat. *Use case* secara eksplisit menjelaskan fungsi-fungsi yang nantinya akan digunakan oleh aktor (pengguna), sehingga pembuatan use case harus diperhatikan karena use case yang nantinya akan menjadi acuan dalam pembuatan dan pengembangan perangkat lunak nantinya [40].

Sumber: Artikel Universitas Binus [41]

Gambar 2.14 Contoh *Use Case Diagram*

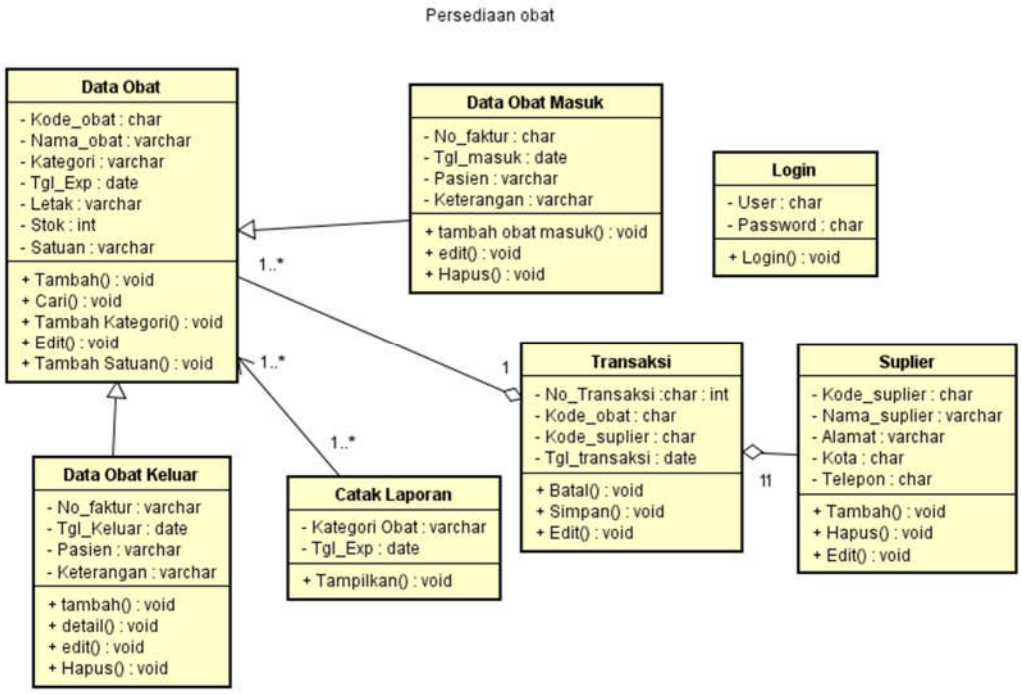
Use case description merupakan tabel yang digunakan untuk membuat dan menjelaskan keterangan terperinci mengenai tiap-tiap use case [42]. Use case description merupakan tabel yang akan menjelaskan bagaimana suatu fungsi terhubung dengan aktor berjalan. *Use case description* menjadi suatu acuan dalam pembuatan fungsi pada perangkat lunak sehinggga nantinya perangkat lunak yang dihasilkan sesuai dengan apa yang diharapkan.

Tabel 2.2 Contoh Tabel *Use Case Description*

Use Case Name	Ekspor data kecamatan	
Goal In Context	Melakukan ekspor data kecamatan ke dalam bentuk pdf/excel	
Description	Proses ini merupakan proses untuk melihat/mencetak data kecamatan	
Precondition	*User* berada pada menu mengolah data kecamatan	
Related Use Case	Mengolah data kecamatan	
Successful End Condition	Meng-ekspor data kecamatan	
Failed End Condition	-	
Actors	Pegawai	
Trigger	*User* memilih tombol unduh ke pdf/excel	
Main Flow	Step	Action
	1	Sistem menampilkan form data kecamatan
	2	*User* memilih tombol unduh ke pdf/excel

2.26.2 Class Diagram

Class diagram terdiri atas kelas-kelas yang memiliki nama, field-field di dalam kelas, dan tindakan-tindakan (kadang disebut sebagai metode) yang dilakukan atas kelas [43]. *Class diagram* merupakan diagram yang menggambarkan class yang terlibat di dalam sistem. *Class diagram* dapat saling berelasi satu sama lain.



Gambar 2.15 Contoh *Class Diagram*

Pada gambar terdapat keterangan berupa enkapsulasi terhadap variabel dan metoda. Terdapat 3 jenis enkapsulasi yaitu [44]:

1. Private

   Private tidak dapat dipanggil dari luar *class* yang bersangkutan yang dilambangkan dengan notasi (-).

2. Protected

   Protected hanya dapat dipanggil oleh kelas yang bersangkutan dan anak-anak yang mewarisinya yang dilambangkan dengan notasi (#).

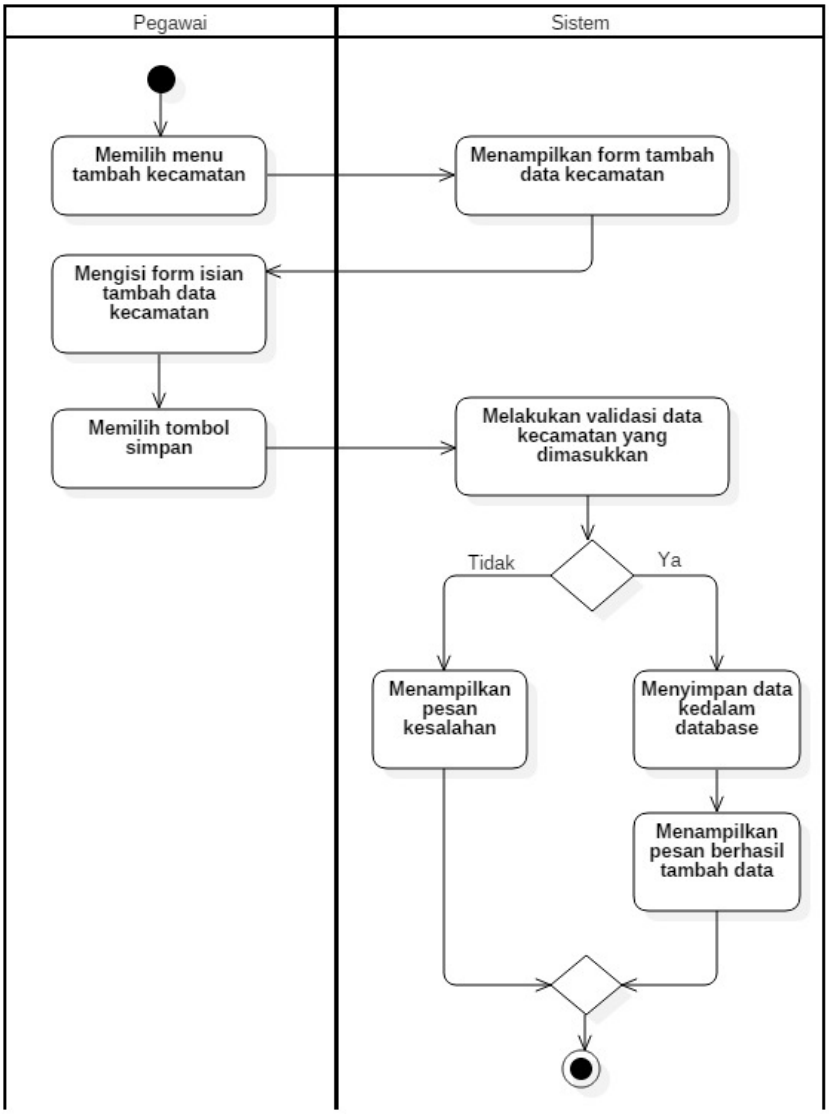
3. Public

   Public oleh seluruh kelas yang dilambangkan dengan notasi (+).

2.26.3 Activity Diagram

Activity diagram adalah diagram UML yang digunakan untuk menggambarkan alur aktivitas dari satu proses. Misalnya proses memesan makan, aktivitasnya adalah menelpon restoran, memesan makan, dan memberikan alamat kita, kemudian tunggu pesanan dan bayar. *Activity diagram* memungkinkan siapapun yang melakukan proses untuk memilih urutan dalam melakukannya, dengan kata

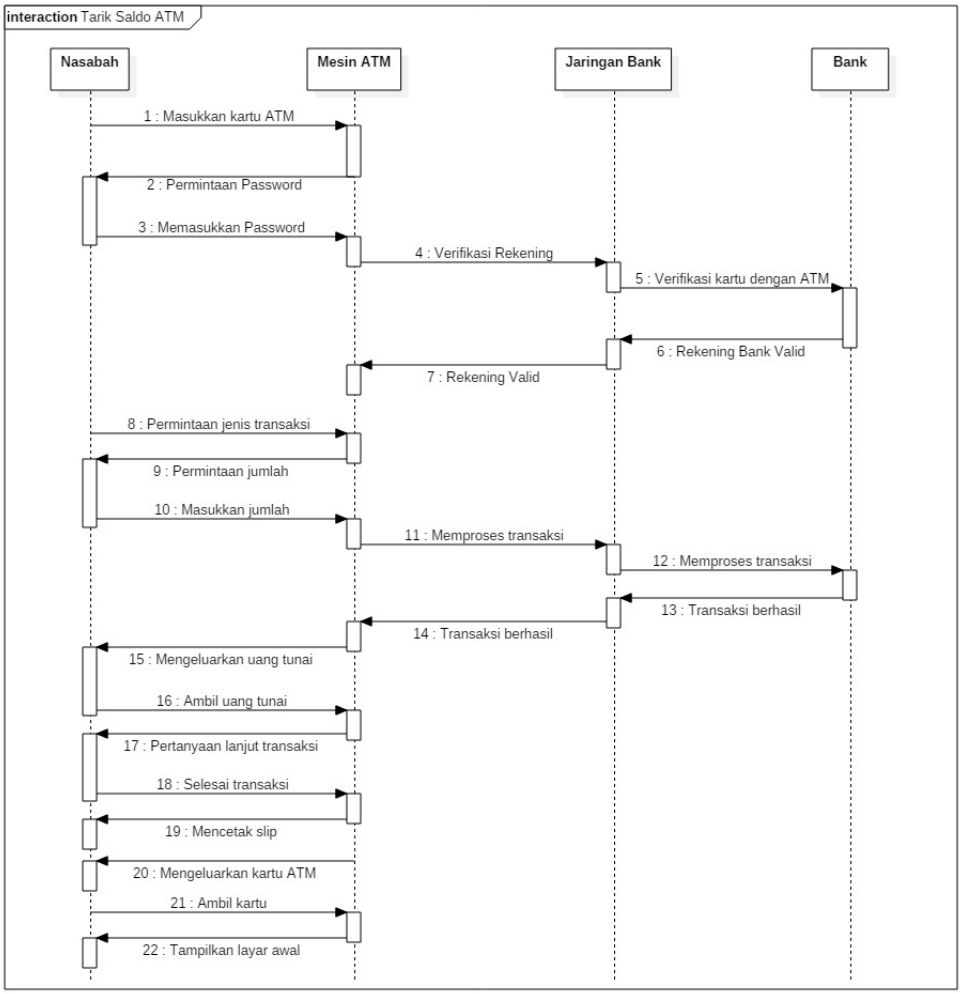
lain diagram hanya menyebutkan aturan-aturan rangkaian dasar yang harus kita ikuti [45].



Gambar 2.16 Contoh *Activity Diagram*

2.26.4 Sequence Diagram

Sequence diagram adalah diagram yang menggambarkan interaksi antar objek. *Sequence diagram* secara khusus menjabarkan *behavior* sebuah skenario tunggal. Diagram tersebut menunjukan sejumlah objek contoh dan pesan-pesan yang melewati objek ini dalam sebuah *use case* [45]. *Sequence diagram* merupakan diagram yang menjelaskan komunikasi data maupun pesan antar objek.



Sumber: Buku Rekayasa Perangkat Lunak Menggunakan UML dan Java [40]

Gambar 2.17 Contoh *Sequence Diagram*

2.27 Pengujian Alpha

Pengujian alpha dilakukan terhadap aplikasi untuk memastikan bahwa aplikasi dapat berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan [46]. Adapun 2 metode pengujian Alpha adalah sebagai berikut:

2.27.1 Pengujian White Box

Merupakan metode perancangan *test case* yang menggunakan struktur kontrol dari perancangan prosedural untuk mendapatkan *test case*. Dengan menggunakan metode *white box*, analis sistem akan dapat memperoleh keuntungan uji kasus sebagai berikut:

1. Menjamin seluruh *independent path* di dalam modul yang dikerjakan sekurang-kurangnya sekali.
2. Mengerjakan seluruh keputusan logikal.
3. Mengerjakan seluruh *loop* yang sesuai dengan batasannya.
4. Mengerjakan seluruh struktur data internal yang menjamin validitas.

2.27.2 Pengujian Black Box

Pengujian *Black Box* melakukan pengujian terhadap fungsi operasional *software*. Pendekatan ini biasanya dilakukan oleh penguji yang tidak ikut serta dalam pengujian *software*.

1. Pengujian *Black Box* berfokus pada kebutuhan fungsional pada *software*, berdasarkan spesifikasi kebutuhan *software*.
2. Pengujian *Black Box* bukan teknik alternatif daripada pengujian *white box*. Lebih dari pada itu pengujian *black box* merupakan pendekatan pelengkap dalam mencakup error dengan kelas yang berbeda dari metode pengujian *white box*.
3. Pengujian *Black Box* melakukan pengujian tanpa pengetahuan detail struktur internal dari sistem atau komponen yang dites juga disebut sebagai *behavioral testing*, *spesification-based testing*, *input/ouput testing* atau *functional testing*.
4. Pada pengujian *black box* terdapat jenis teknik desain tes yang dapat dipilih berdasarkan pada tipe *testing* yang akan digunakan diantaranya *equivalence class partioning*, *boundary value analysis*, *state transitions testing*, *cause-effect graphing*.

Kategori kesalahan yang akan diketahui melalui metode pengujian *black box* diantaranya:

1. Fungsi yang hilang atau tidak sesuai.
2. Kesalahan antar muka.
3. Kesalahan dari struktur data atau akses eksternal *database*.
4. Kesalahan kinerja dan tingkah laku.
5. Kesalan inisialisasi dan terminasi [47].

2.28 Pengujian Beta

Pengujian Beta dilakukan secara langsung terhadap pengguna dengan menggunakan kuesioner mengenai kepuasan pengguna atas sistem yang telah dibangun. Pengguna akan menjawab setiap pertanyaan mengenai kualitas dari sebuah sistem yang dibangun. Hasil jawaban pengguna tersebut kemudian menjadi dasar pengembang aplikasi apakah sistem yang dibangun dapat diterima atau tidak hingga menjadi bahan evaluasi untuk pengembangan lebih lanjut [46].

Tes beta merupakan tahap kedua dari pengujian perangkat lunak di mana pengguna mencoba aplikasi. Awalnya, tes alpha berarti tahap pertama pengujian dalam proses pengembangan perangkat lunak. Tahap pertama meliputi *unit testing*, pengujian komponen, dan pengujian sistem. Pengujian beta dapat dianggap "pengujian pra-rilis artinya sebelum aplikasi tersebut dirilis maka harus dipastikan dari sisi pengguna bahwa perangkat lunak tersebut terbebas dari cacat atau kegagalan [48].