BAB 2
LANDASAN TEORI

2.1 Kota Bandung

Kota Bandung terletak di wilayah Jawa Barat dan merupakan Ibukota Propinsi Daerah Tingkat I Jawa Barat. Kota Bandung terletak 107º Bujur Timur dan 6º 55’ Lintang Selatan. Lokasi Kotamadya Bandung cukup strateggis, dilihat dari segi komunikasi, perekonomian maupun keamanan. Hal tersebut disebabkan oleh **[**4**]** :

1. Kota Bandung terletak pada pertemuan poros jalan raya :
   a. Barat Timur yang memudahkan hubungan dengan Ibukota Negara.
   b. Utara Selatan yang memudahkan lalu lintas ke daerah perkebunan (Subang dan Pangalengan). [4]
2. Letak yang tidak terisolasi serta dengan komunikasi yang baik akan memudahkan aparat keamanan untuk bergerak ke setiap penjuru. [4]

Secara topograpis Kota Bandung terletak pada ketinggian 768 meter di atas permukaan laut, titik tertinggi di daerah Utara dengan ktinggian 1.050 meter dan terendah di sebelah Selatan adalah 675 meter di atas perumukaan laut. Di wilayah Kotamadya Bandung bagian Selatan permukaan tanah relatif datar, sedangkan di wilayah kota bagian Utara berbukit-bukit sehingga merupakan panorama yang indah.[4]

Keadaan Geologis dan tanah yang ada di Kota Bandung dan sekitarnya terbentuk pada zaman. Kwartier dan mempunyai lapisan tanah *alluvial* hasil letusan gunung Tangkuban Perahu. Jenis material di bagian Utara umumnya merupakan jenis *andosol*, dibagian selatan serta timur terdiri atas sebaran jenis alluvial kelabu dengan bahan endapan tanah liat. Di bagian tengah dan barat tersebar jenis *andosol*.[4]

Iklim Kota Bandung dipengaruhi oleh iklim pegunungan yang lembab dan sejuk. Pada tahun 1998 temperatur rata-rata 23,5º C, curah hujan rata-rata 200,4 mm dan jumlah hari hujan rata-rata 21,3 hari perbulan.[4]

2.1.1 Sejarah

Kota Bandung tidak berdiri bersamaan dengan pembentukan Kabupaten Bandung. Kota itu dibangun dengan tenggang waktu sangat jauh setelah Kabupaten Bandung berdiri. Kabupaten Bandung dibentuk pada sekitar pertengahan abad ke-17 Masehi, dengan Bupati pertama Tumenggung Wiraangunangun. Beliau memerintah Kabupaten Bandung hingga tahun 1681.[4]

Semula Kabupaten Bandung beribukota di Krapyak (sekarang Dayeuhkolot) kira-kira 11 kilometer ke arah Selatan dari pusat Kota Bandung sekarang. Ketika Kabupaten Bandung dipimpin oleh Bupati k-6, yakni R.A Wiranatakusumah II (1794-1829) yang dijuluki "Dalem Kaum I", kekuasaan di Nusantara beralih dari Kompeni ke Pemerintah Hindia Belanda, dengan Gubernur Jenderal pertama Hermak Willem Daendels (1808-1811). Untuk kelancaran menjalanjan tugasnya di Pulau Jawa, Daendels membangun Jalan Raya Pos (*Groote Postweg*) dari Anyer di ujung barat Jawa Barat ke Panarukan di ujung timur Jawa Timur (kira-kira 1000 km). pembangunan jalan raya itu dilakukan oleh rakyat pribumi di bawah pimpinan bupati daerah masing-masing.[4]

Di daerah Bandung khususnya dan daerah Priangan umumnya, Jalan Raya Pos mulai dibangun pertengahan tahun 1808, dengan memperbaiki dan memperlebar jalan raya yang telah ada. Di daerah Bandung sekarang, jalan raya itu adalah Jalan Jenderal Sudirman – Jalan Asia Afrika – Jalan A. Yani, berlanjut ke Sumedang dan seterusnya. Untuk kelancaran pembangunan jalan raya, dan agar pejabat pemerintah *colonial* mudah mendatangi kantor bupati, Daendels memlaui surat tanggal 25 Mei 1810 meminta Bupati Bandung dan Bupati Parakanmuncang untuk memindahkan ibukota kabupaten, masing-masing ke daerah Cikapundung dan Andawadak (Tanjungsari), mendekati Jalan Raya Pos. [4]

Rupanya Daendels tidak mengetahui, bahwa jauh sebelum surat itu keluar, Bupati Bandung sudah merencanakan untuk memindahkan ibukota Kabupaten Bandung, bahkan telah menemukan tempat yang cukup baik dan strategis bagi pusat pemerintah. Tempat yang dipilih adalah lahan kosong berupa hutan, terletak di tepi barat Sungai Cikapundung, tepi Selatan Jalan Raya Pos yang sedang

dibangun (pusat Kota Bandung sekarang). Alasan pemindahan ibukota itu antara lain, Krapyak tidak strategis sebagai ibukota pemerintahan, karena terletak di sisi selatan daerah Bandung dan sering di landa banjir bila musim hujan.[4]

Sekitar akhir tahun 1808/ awal tahun 1809, bupati beserta sejumlah rakyatnya pinda dari Krapyak mendekati lahan bakal ibukota baru. Mulamula bupati tinggal di Cikalintu (daerah Cipaganti), kemudia pindah ke Balubur Hilir, selanjutnya pindah lagi le Kampur Bogor (Kebon Kawung, pada lahan Gedung Pakuan sekarang).[4]

Tidak diketahui secara pasti, berapa lama Kota Bandung dibangun. Akan tetapi, kota itu dibangun bukan atas prakarsa Daendels, melainkan atas prakarsa Bupati Bandung, bahkan pembangunan kota itu langsung dipimpin oleh bupati. Dengan kata lain, Bupati R.A Wiranatakusumah II adalah pendiri (the founding father) kota Bandung. Kota Bandung diresmikan sebagai ibukota baru Kabupaten Bandug dengan surat keputusan tanggan 25 September 1810.[4]

2.1.1 Visi dan Misi

Berikut merupakan penjelasan dari Visi dan Misi dari Kota Bandung :

2.1.2.1 Visi

“Terwujudnya Kota Bandung Sebagai Kota Jasa yang Bermartabat (Bersih, Makmur, Taat dan Bersahabat)”[4]

Untuk merealisasikan keinginan, harapan, serta tujuan sebagaimana tertuang dalam visi yang telah ditetapkan, maka pemerintah bersama elemen seluruh masyarakat Kota Bandung harus memahami makna dari visi tersebut yaitu[4] :

Pertama : Kota Bandung sebagai Kota Jasa harus bersi dari sampah, dan bersi praktik Korupsi, Kolusi dan Nepotisme (KKN), penyakit masyarakat (judi, pelacuran, narkobam premanisme dan lainnya), dan perbuatan-perbuatan tercela lainnya yang bertentangan dengan moral dan agama dan budaya masyarakat ata bangsa[4];

Kedua : Kota Bandung sebagai Kota Jasa yang memberikan kemakmuran bagi warganya;[4]

Ketiga : Kota Bandung sebagai Kota Jasa harus memiliki warga yang taat terhadap agama, hukum dan aturan. Aturan yang ditetapkan untuk menjaga keamanan, kenyamanan, dan ketertiban kota;[4]

Keempat : Kota Bandung sebagai Kota Bandung harus memiliki warga yang bersahabat, santun, akrab dan dapat menyenangkan bagi orang yang berkunjung serta menjadikan kota yang bersahabat dalam pemahaman kota yang ramah lingkungan.[4]

Secara harfiah, Bermartabat diartikan sebagai harkat atau harga diri, yang menujukkan eksistensi masyarakat kota yang dapat dijadikan teladan karena kebersihan, kemakmuran, ketaatan, ketaqwaan dan kedisplinannya.[4]

Jadi Kota Jasa yang bermartabat adalah koa yang menyediakan jasa pelayanan yang didukung denga terwujudnya kebersihan, kemakmuran, ketaqwaan, dan kedisplinan masyarakatnya.[4]

Berdasarkan pemahaman tersebut, sangatlah rasional pada kurun waktu lima tahun kedepan diperlukan langkan dan tindakan pemantapan (revitalasi, reaktulisasi, reorientasi dan refungsionalisasi) yang harus dilakukan oleh pemerintah Kota Bandung berserta masyarakatnya serta didukung secara politis oleh pihak legislative melalui upaya-upaya yang lebih keras, cerdas dan terarah namun tetap ramah dalam meningkatkan akselerasi pembangunan guna tercapainya kemakmuran dan kesejahteraan masyarakat.[4]

2.1.2.2 Misi

Misi adalah tugas diemban Pemerintah Kota Bandung meliputi :

1. Mengembangkan sumber daya manusia yang handal yang religious, yang mencakup pendidikan, kesehatan dan moral keagamaan.
2. Mengembangkan perekonomian kota yang adil, yang mencakup peningkatan perekonomian kota tangguh, sehat dan berkeadilan dalam ranka meningkatkan pendapatan masyarakat, menciptakan lapangan kerja dan kesempatan berusaha.

3. Mengembangkan Sosial Budaya Kota yang ramah dan berkesadaran tinggi, serta berhati nurani, yang mencakup peningkatan partisipasi masyarakat dalam rangka menigkatkan ketenagakerjaan, meningkatkan kesejahteraan sosial, keluarga, pemuda dan olah raga serta kesetaraan gender.
4. Meningkatkan penataan kota, yang mencakup pemeliharaan serta peningkatan prasarana dan sarana kota agar sesuai dengan dinamika peningkatan kegiatan kota dengan tetap memperhatikan tata ruang kota dan daya dukung lingkungan kota.
5. Meningkatkan kinerja pemerintah kota secara professional, efektif, efisien akuntabel dan tranparan, yang mencakup pemberdayaan aparatur pemerintah dan masyarakat.
6. Mengembangkan sistem keuangan kota, mencakup sistem pembiayaan pembangunan yang dilaksanakan pemerintah. Swasta dan masyarakat.[4]

2.1.2 Arti Lambang



Gambar 2.1 Lambang Kota Bandung

Berikut merupakan penjelasan dari lambang Kota Bandung :

2.1.3.1 Lambang

Lambang kota Bandung ditetapkan dengan peraturan daerah kota besar Bandung tahun 1953, tertanggal 8 Juni 1953, yang diijinkan dengan Keputusan Presiden tertanggal 28 April 1953 No. 104 dan diundangkan dalam Berita

Provinsi Jawa Barat tertanggal 28 Agustus 1954 No. 4 lampiran No. 6 Lambang tersebut bertokoh Perisai yang berbentuk Jantung. Perisai tersebut terbagi dalam dua bagian oleh sebuah Balok – Lintang mendatar bertajuk empat buah, yang berwarna hitam dengan pelisir berwarna Putih (Perak) pada pinggir sebelah atasnya[4] :

1. Bagian atas latar Kuning (Emas) dengan lukisan sebuah Gunung berwarna Hijau yang bertumpu pada blok-lintang.
2. Bagian bawahh latar Putih (Perak) dengan lukisan empat bidan jalur mendatar berombak yang berwarna Biru.

Di bawah perisai itu terlukis sehelai Pita berwarna Kuning (Emas) yang melambai pada kedua ujungnya, pada pita itu tertulis huruf-huruf besar latin berwarna Hitam amsal dalam bahasa Kawi, yang berbunyi Gemah Ripah Wibawa Mukti. [4]

Sebagai tokoh lambang itu diambil bentuk perisai atau tameng, yang dikenal kebudayaan dan peradaban sebagai senjata dalam perjuangan untuk mencapai sesuatu tujuan dengan melindungi diri. Perkakas perjuangan yang demikian itu dijadikan lambang yang mempunyai arti menahan segala mara bahaya dan kesukaran.

1. Kuning (Emas), berarti : Kesejahteraan, Keluhungan.
2. Hitam (Sabel), berarti : Kokoh, tegak, kuat.
3. Hijau (Sinopel), berarti : Kemakmuran sejuk.
4. Putih (Perak), berarti : Kesucian.
5. Biru (Azuur), berarti : Kesetiaan.

Gemah Ripah Wibawa Mukti, berarti : Tanah subur rakyat makmur.[4]

2.1.3.2 Bendera



Gambar 2.2 Bendera Kotamadya Bandung

Bendera yang digunakan oleh Kotamadya Bandung adalah berdasarkan Surat Keputusan Dewan Perwakilan Rakyat Daerah Sementara Kota Besar Bandung tanggal 8 Juni 1953 No. 9938/53.[4]

Bentuk bendera tersebut adalah seperti yang tercantum oada dictum Keputusan tersebut diatas sebagai berikut :

1. Bendera yang dipergunakan oleh Kota Besar Bandung dan tiga bidang jalur mendatar, masing-masing berturut-turut dari atas kebawah berwarna Hijau, Kuning, dan Biru.
2. Perbandingan-perbandingan antara lebarnya dan jalur-jalur tersebut huruf a urutan dari atas kebawah adalah 2:1:1.
3. Perbandingan antara panjang dan lebarnya berbeda itu 7:5.[4]

2.2 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis *linux* yang mencakup 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 piranti lunak untuk ponsel/*smartphone*. Kemudian untuk mengembangkan android, dibentuklah *Open Handset Alliance*, *konsorsium* dari 34 perusahaan peranti keras, peranti lunak , telekomunikasi , termasuk Google, HTC, Intel, Motorola , Qualcomm, T-Mobile, dan Nvidia. [**2**]

Pada saat perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat mobile. Di lain pihak, Google merilis kode-kode Android di bawah lisensi *Apache*, sebuah lisensi perangkat lunak dan open *platform* perangkat seluler.[5]

Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).[5]

Pada masa saat ini kebanyakan vendor-vendor smartphone sudah memproduksi *smartphone* berbasis android, vendor-vendor itu antara lain HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, LG, Acer, Philips, T-Mobile, Nexian, IMO, Asus dan masih banyak lagi vendor *smartphone* didunia yang memproduksi android. Hal ini karena android itu adalah sistem operasi yang *open source* sehingga bebas didistribusikan dan dipakai oleh vendor manapun. [5]

2.2.1 Android SDK (*Software Development Kit*)

Android SDK adalah tools API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemograman *Java*. Android merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi., *middleware* dan aplikasi kunci yang di rilis oleh Google. Saat ini disediakan Android SDK (*Software Deveplmonet Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemograman *Java*. Sebagai platform aplikasi-netral, Android memberi anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan Handphone/*Smartphone*. Beberapa fitur-fitur Android yang paling penting adalah[5] :

a. *Framework* aplikasi yang mendukung penggantian komponen dan reusable.
b. Mesin Virtual Dalvik diptimalkan untuk perangkat mobile.

c. Integrated browser berdasarkan engine open source *WebKit.*
d. Grafis yang dioptimalkan dan didukung oleh libraries grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional akselerasi hardware).
e. SQLite untuk penyimpanan data.
f. Media Support yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM Telephony (tergantung hardware).
g. Bluetooth, EDGE, 3G, dan WiFi (tergantung hardware).
h. Kamera, GPS, kompas, dan *accelerometer* (tergantung hardware).
i. Lingkungan *Development* yang lengkap dan kaya termasuk perangkat emulator, tools untuk debugging, profil dan kinerja memori, dan plugin untuk IDE Eclipse.

2.2.2 ADT (*Android Development Tools*)

Android Development Tools (ADT) adalah plugin yang didesain untuk IDE Eclipse yang memberikan kita kemudahan dalam mengembangkan aplikasi android dengan menggunakan *IDE Eclipse*. Dengan menggunakan *ADT* untuk Eclipse akan memudahkan kita dalam membuat aplikasi *project* android, membuat *GUI* aplikasi, dan menambahkan komponen-komponen yang lainnya, begitu juga kita dapat melakukan running aplikasi menggunakan *Android SDK* melalui *Eclipse*. Dengan *ADT* juga kita dapat melakukan pembuatan package android (*.apk*) yang digunakan untuk distribusi aplikasi android yang kita rancang.[5]

2.2.3 Arsitektur Android

Secara garis besar Arsitektur Android dapat dijelaskan dan digambarkan sebagai berikut :

1. Applications dan Widgets

Applications dan Widgets ini adalah layer di mana kita berhubungan dengan aplikasi saja, di mana biasanya kita download aplikasi kemudian kkita lakukan instlasi dan jalankan aplikasi tersebut. Di layer terdapat aplikasi inti

termsauk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis menggunakan bahasa pemograman *Java*.[5]

2. Applications Frameworks

Android adalah "*Open Development Platform*" yaitu Android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat akses, akses informasi, *resources*, menjalankan *service background*, mengatur alarm, dan menambahkan status notifications, dan sebagainya. Pengembang memiliki akses penuh menuju *API* framework seperti yang dilakukan oleh aplikasi yang kategori inti. Arsitektur aplikasi dirancang supaya kita dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (reuse).[5]

Sehingga bisa kita simpulkan *Applications Frameworks* ini adalah layer di mana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi Android, karena pada layer inilah aplikasi dapat dirancang dan dibuat, seperti *content-providers* yang berupa sms dan panggilan telepon. [5]

3. Libraries

Libraries ini adalah *layer* di mana fitur-fitur Android berada, biasanya para pembuat aplikasi menagkses libraries untuk menjalankan aplikasinya. Berjalan di atas kernel, layer ini meliputi berbagai library C/C++ inti seperti *Libc* dan *SSL*, serta [5] :

a) *Libraries* media untuk pemutaran media audio dan video
b) *Libraries* untuk manajemen tampilan.
c) *Libraries Graphics* mencakup *SGL* dan *OpenGL* untuk grafis 2D dan 3D.
d) Libraries SQLite untuk dukungan *database.*
e) Libraries SSL dan *WebKit* terintegrasi dengan *web browser* dan *sequrity*.
f) Libraries LiveWebcore mencakup *modern web browser* dengan *engine embeded web view*.
g) *Libraries* 3D yang mencakup implementasi *OpenGL* ES 1.0 API's.

4. Android Run Time

Layer yang membuat aplikasi *Android* dapat dijalankan di mana dalam prosesnya menggunakan Implementasi *Linux*. *Dalvix Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android. Di dalam *Android Run Time* dibagi menjadi dua bagian yaitu[5] :

a) *Core Libraries* : Aplikasi Android dibangun dalam bahasa *Java*, sementara *Dalvik* serbagai virtual mesinnya bukan *Virtual Machine Java*, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menterjemahkan bahasa Java/C yang ditangani oleh Core Libraries.

b) *Dalvik Virtual Machine* : Virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi serta efisien , dimana merupakan pengembangan yang mampu membuat *linux kernel* untuk melakukan *threading* dan manajemen tingkat rendah.[5]

5. Linux Kernel

Linux Kernel adalah layer dimana inti dari operating sistem dari *Android* itu berada. Berisi file-file system yang mengatur sistem procressing, memory, *resource*, *drivers*, dan sistem-sistem operasi android adalah linux kernel release 2.6.[5]

2.2.4 Fundamental Aplikasi

Aplikasi Android ditulis dalam bahasa pemograman Java. Kode Java dikompilasi bersama dengan data file resource yang dibuthkan oleh aplikasi, di mana prosesnya dipackage oleh tools yang dinamakan “apt tools” ke dalam paket Android sehingga menghasilkan file dengan ekstensi apk [5]. Ada enam jenis komponen pada aplikasi Android yaitu :

a. Activities

Suatu activity akan menyajikan user interface (UI) kepada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi android bisa jadi hanya memili satu activity, tetapi umumnya aplikasi memiliki banyak activity tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut. Satu activity

biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai user interface (UI) saat aplikasi diperlihatkan kepada user.[5]

b. Service

Service tidak memiliki Graphic User Interface (GUI), tetapi service berjalan secara background, sebagai contoh dalam memainkan music, service mungkin memainkan music atau mengambil data dari jaringan, tetapi setiap service harus berada dalam kelas induknya. Misalnya, media player sedang memutar lagu dari list yang ada, aplikasi ini akan memiliki dua atau lebih activiry yang memungkinkan user untuk memilih lagu misalnya, atau menulis sms sambil player sedang berjalan. Untuk menjaga music tetap di jalankan, activiry player dapat menjalankan service.[5]

c. Broadcast Receiver

Broadcast Receiver berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contoh broadcast seperti notifikasi zona waktu berubah, baterai low, gambar telah selesai diambil oleh kamera, atau pengubahan referensi bahasa yang digunakan. Aplikasi ini juga dapat menginisiasi broadcast misalnya memberikan informasi pada aplikasi lain bahwa ada data yang telah di unduh ke perangkat dan siap untuk digunakan.[5]

d. Content Provider

Content Provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file sistem seperti database SQLite. Content Provider menyediakan cara untuk mengakses data yang dibuthkan oleh suatu activity, misalnya ketika kita menggunakan aplikasi yang membutuhkan peta (Map), atau aplikasi yang membutuhkan untuk mengakses data kontak dan navigasi, maka disinilah fungsi Content Provider. [5]

2.2.5 Versi Android

Telepon pertama yang memakai sistem Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Pada pengunjung tahun 2010 diperkirakan hampir semua vendor seluler di dunia menggunakan Android sebagai operating system. Adapun versi-versi Android yang pernah dirilis adalah sebagai berikut[5] :

1. *Android* versi 1.1

Pada 9 Maret 2009, *Google* merilis *Android* versi 1.1 Abdroid versi ini dilengkapi dengan pembaruan *estetis* pada aplikasi, jam, alarm, *voice search* (pencarian suara), pengiriman pesan dengan *Gmail*, dan pemberitahuan email.[5]

2. *Android* versi 1.5 (*Cupcake*)

Pada pertengahan Mei 2009, *Google* kembali merilis telepon seluler dengan menggunakan *Android* dan SDK (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengupload video ke *Youtube* dan gambar ke *Picasa* langsung dari telepon, dukungan *Bluetooth* A2DP, kemampuan terhubung secara otomatis ke *headset bluetooth*, animasi layar, dan *keyboard* pada layar yang dapat disesuaikan dengan sistem.[5]

3. *Android* versi 1.6 (*Donut*)

Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan control applet VPN.[5]

4. *Android* versi 2.0/2.1 (*Éclair*)

Pada 3 Desember 2009 kembali diluncurkan ponsel *Android* dengan versi 2.0/2.1 (Éclair). [5]

5. *Android* versi 2.2 (*Froyo: Frozen Yoghurt*)

Pada bulan Mei 2010 Android versi 2.2 Rev 1 diluncurkan. *Android* inilah yang sekarang sangat banyak beredar dipasaran, salah satunya adalah dipakai *Samsung FX Tab* yang sudah ada dipasaran.[5]

6. *Android* versi 2.3 (*Gingerbread*)

Android versi 2.3 diluncurkan pada Desember 2010.[5]

7. *Android* versi 3.0 (*Honeycomb)* dirilis Februari 2011 sebagai *Android* 3.0 revisi 1 serta Android versi 3.0 revision 2 telah dirilis pada Juli 2011.[5]
8. *Android* 3.1 dirilis Mei 2011, sedangkan android 3.1 revisi 2 juga dirilis Mei 2011, serta Android 3.1 revision 3 dirilis pada Juli 2011 [5]
9. Android versi 3.2 dirilis Juli 2011 [5]

10. Android versi 4.0 dirilis November 2011 [5]
11. Android versi 4.1 [5]
12. Android versi 4.2 [5]
13. Android versi 4.3 [5]

2.3 *Application Programming Interface* (API)

Berikut adalah penjelasan mengenai API :

2.3.1 Perkenalan

API adalah singkatan dari application programming interface. API dapat memberikan kait untuk rekan kerja, mitra, atau pengembang pihak ketiga untuk mengakes data dan layanan untuk membangun aplikasi. [**3**]

2.3.2 Bagaimana Sebuah API Digunakan

Ketika dua sistem (*website, desktop, smartphone*) menghubungkan melalui API, kami mengatakan mereka "terintegrasi." Dalam sebuah integrasi, Anda memiliki dua sisi, masing-masing dengan nama khusus. Satu sisi kita telah berbicara tentang: server. Ini membantu untuk mengingat bahwa API ini hanya program lain yang berjalan pada server 3. Ini mungkin bagian dari program yang sama yang menangani lalu lintas web, atau dapat menjadi salah satu benar-benar terpisah. Dalam kedua kasus, itu duduk, menunggu orang lain untuk meminta untuk data. **[**7**]**

Sisi lain adalah "klien." Ini adalah program terpisah yang tahu apa data tersedia melalui API dan dapat memanipulasi, biasanya atas permintaan dari pengguna. Sebuah contoh yang bagus adalah aplikasi smartphone yang sync dengan website. Ketika Anda menekan tombol refresh aplikasi Anda, itu berbicara ke server melalui API dan mengambil info terbaru.[7]

Prinsip yang sama berlaku untuk situs web yang terintegrasi. Ketika salah satu situs menarik di data dari yang lain, situs yang menyediakan data bertindak sebagai server, dan situs mengambil data adalah klien.[7]

2.3.3 Metode

Metode permintaan memberitahu server apa tindakan klien ingin server untuk mengambil. Bahkan, metode ini sering disebut sebagai permintaan "kata kerja." [7]

Empat metode yang paling sering terlihat di API adalah:

a. *GET* - Minta server untuk mengambil sumber daya
b. *POST* - Minta server untuk menciptakan sumber daya baru
c. *PUT* - Minta server untuk mengedit / memperbarui sumber daya yang ada
d. *DELETE* - Minta server untuk menghapus sumber daya [7]

Berikut ini adalah contoh untuk membantu menggambarkan metode ini. Katakanlah ada sebuah restoran pizza dengan *API* yang dapat digunakan untuk menempatkan pesanan. Anda memesan dengan membuat permintaan *POST* ke server restoran dengan detail pesanan Anda, meminta mereka untuk membuat pizza Anda. Segera setelah Anda mengirimkan permintaan, namun, Anda sadar bahwa Anda memilih salah gaya kerak, sehingga Anda membuat permintaan *PUT* untuk mengubahnya.[7]

2.3.4 *Google Maps API*

Aplikasi dan situs web yang menggabungkan data atau fungsionalitas dari dua sumber atau lebih disebut sebagai mashup. *Mashup* menjadi semakin popular dan telah merevolusi cara informasi digunakan dan divisualisasikan. Solusi pemetaan merupakan salah satu unsur penting dalam banyak mashup ini. Google Maps API memungkinkan anda memanfaatkan kekuatan Google Maps untuk digunakan dalam aplikasi anda sendiri untuk menampilkan aplikasi anda sendiri (atau lainnya) dengan cara yang efisien dan bermanfaat. [**5**]

2.3.4.1 Memuat *Google Maps JavaScript API*

Untuk memuat *Google Maps JavaScript API*, harus menggunakan *tag script* seperti contoh berikut ini **[**9**]**:

```
<script                                    async                                    defer

    src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMa
p">

  </script>
```

URL yang terdapat dalam tag script adalah lokasi file *JavaScript* yang memuat semua simbol dan definisi yang dibutuhkan untuk menggunakan *Maps JavaScriptAPI*. Tag script ini diperlukan. [9]

Atribut *async* memungkinkan browser merender bagian selebihnya dari situs web yang akan dibuat saat *Maps JavaScript API* dimuat. Bila API sudah siap, ia akan memanggil fungsi yang ditetapkan menggunakan parameter *callback*. Parameter *key* berisi kunci API aplikasi yang dibuat.[9]

2.3.4.2 Elemen *DOM* Peta

Untuk memuat *Google Maps JavaScript API*, harus menggunakan *tag script* seperti contoh berikut ini[9]:

```
<script                                    async                                    defer

    src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMa
p">

  </script>
```

URL yang terdapat dalam tag script adalah lokasi file *JavaScript* yang memuat semua simbol dan definisi yang dibutuhkan untuk menggunaka *Maps JavaScriptAPI*. *Tag script* ini diperlukan.[9]

Atribut async memungkinkan *browser* merender bagian selebihnya dari situs web yang akan dibuat saat *Maps JavaScript API* dimuat. Bila API sudah siap,

ia akan memanggil fungsi yang ditetapkan menggunakan parameter *callback*. Parameter *key* berisi kunci API aplikasi yang dibuat.[9]

2.3.4.3 Opsi Peta

Ada dua opsi yang diperlukan untuk setiap peta : *center* dan *zoom.*[9]

```
map = new google.maps.Map(document.getElementById('map'), {
  center: {lat: -34.397, lng: 150.644},
  zoom: 8
});
```

2.3.4.4 Tingkat Zoom

Resolusi awal untuk menapilkan peta disetel oleh properti *zoom*, dalam hal ini *zoom* 0 menyatakan peta bumi yang diperkecil maksimum, dan tingkat *zoom* yang lebih besar akan memperbesar dengan resolusi lebih tinggi. Menetapkan tingkat *zoom* sebagai *integer*.[9]

```
zoom: 8
```

Gambar peta dalam *Google Maps* dan *Maps JavaScript API* akan dipecah menjadi “petak” peta dan “tingkat *zoom*”. Pada tingkat *zoom* rendah, satu rangkaian kecil petak peta mencakup area yang luas; di tingkat *zoom* yang lebih tinggi, petak memiliki resolusi lebih tinggi dan mencakup area yang lebih kecil. Daftar ini menunjukkan perkiraan tingkat detail yang di harapkan untuk terlihat di setiap tingkat *zoom* [9] :

1. 1 : Dunia
2. 5 : Daratan luas/benua
3. 10 : Kota
4. 15 : Jalan
5. 20 : Bangunan

2.3.4.5 Objek Peta

Kelas *JavaScript* yang menyatakan peta adalah kelas *Map*. Objek di kelass ini mendifinisikan sebuah peta pada lama. (Bisa membuat lebih dari satu instance kelas ini – setiap objek akan mendefinisikan peta terpisah pada laman.) kita membuat sebuah instance baru dari kelas ini menggunakan operator *new JavaScript*.[9]

```
map = new google.maps.Map(document.getElementById("map"), {...});
```

Bila membuat *instance* peta baru, menetapkan elemen *HTML* di laman sebagai *container* untuk peta. Simpul *HTML* adalah adak dari objek *document JavaScript*, dan kita memperoleh referensi ke elemen ini melalui metode document.getElementById().[9]

2.3.4.6 Meminta Detail Tempat

Permintaan detail tempat adalah sebuah *HTTP URL* dari *form* berikut[9] :

https://maps.googleapis.com/maps/api/place/details/output?parameters

Dimana *output* mungkin salah satu dari nilai berikut :

1. Json (disarankan) menunjukkan output di *JavaScript Object Notation* (JSON)
2. *Xml* menunjukkan output sebagai *XML*.[9]

Seperti standar dalam *URL*, semua parameter dipisahkan menggunakan karakter *ampersand (&)*. Berikut adalah daftar parameter dan nilai yang mungkin mereka dapatkan.[9]

3. *Key*(wajib) – Kunci API aplikasi anda. Kunci ini mengidentifikasi aplikasi anda untuk tujuan pengelolaan kuota dan sehingga tempat yang ditambahkan dari aplikasi anda dibuat segera tersedia untuk aplikasi anda.[9]
4. Entah *placeid* atau *reference* (kita harus menyediakan salah satu dari ini, tapi tidak keduanya).[9]
5. P*laceid* – Pengenal teks yang secara unik mengidentifikasi tempat, kembali dari penelusuran tempat.[9]

6. *Language* (optional) – Kode bahasa, yang menujukkan bahasa mana hasilnya harus dikembalikan, jika memungkinkan. Perlu di catat bahwa beberapa bidang mungkin tidak tersedia dalam bahasa yang diminta.[9]
7. *Region* – Kode wilayah, yang ditentukan sebagai kode ccTLD (kode negara tingkat atas domain) dua karakter. Kebanyakan kode ccTLD identik dengan kode ISO 3166-1, dengan beberapa pengecualian. Parameter ini hanya akan berpengaruh, tidak sepenuhnya membatasi, hasil.[9]

2.3.4.7 Permintaan Arah

Sebuah permintaan *Google Maps Directions API* mengambil *form* sebagai berikut :

https://maps.googleapis.com/maps/api/directions/outputFormat?parameters

Seperti standar dalam *URL*, semua parameter dipisahkan menggunakan karakter *ampersand* (&). Berikut adalah daftar parameter dan nilai yang mungkin mereka dapatkan.[9]

1. Parameter Wajib

a. *Origin* – Alamat , nilai lintang/bujur teks, atau ID tempat dari mana anda ingin menghitung arah.
b. *Destination* – Alamat, nilai linta/bujur teks, atau ID tempat yang anda inginkan untuk menghitung arah.
c. *Key* – Kunci aplikasi. Kunci ini mengidentifikasi aplikasi yang ada untuk tujuan pengelolaan kuota.[9]

2. Parameter Pilihan

a. *Mode* (defaults to *driving*) - Menentukan moda transportasi untuk digunakan saat menghitung arah.[9]
b. *Waypoints* – menentukan sebuah array titik arah. Titik arah mengubah rute dengan mengarahkannya melalui lokasi yang ditentukan. Titik arah ditentukan sebagai koordinat garis lintang/bujur, polyline yang dikodekan, ID tempat atau alamat yang akan dikodekan.[9]

c. *Alternatives* – Jika disetel ke true, tetapkan bahwa layanan arah dapat memberikan lebih dari satu rute alternatif dalam respon. Perhatikan bahwa rute alternative dapat meningkatkan waktu respons dari server.[9]

d. *Avoid* – Menujukkan bahwa rute yang dihitung harus menghindari fitur yang ditunjukkan. Parameter ini mendukung argument berikut :

   a) *tolls* – Menunjukkan bahwa rute yang dihitung harus menghindari jalan tol/jembatan.
   b) *highways* – Menunjukkan bahwa rute yang dihitung harus menghindari jalan raya.
   c) *ferries* – Menunjukkan bahwa rute yang dihitung harus menghindari feri.
   d) *indoor* – Menujukkan bahwa rute yang dihitung harus mengindari langkah-langkah dalam ruangan untuk petunjuk arah berjalan dan transit.[9]

a. *Language* – Bahasa untuk mengembalikan hasil. Google biasanya menupdate bahasa yang didukung. Jika bahasa tidak disertakan, API mencoba menggunakan bahasa pilihan seperti yang ditemtukan dalam header Accep-Language, atau bahasa asli dari domain tempat permintaan dikirim.[9]

b. *Units* – Menentukkan sistem unit yang akan digunakan saat menampilkan hasilnya.[9]

c. *Regions* – Menentukakkan kode wilayah, yang ditentukan sebagai nilai dua karakter ccltd (“domain tingkat atas”).[9]

d. *Arrival_time* – Menentukkan waktu kedatangan yang diinginkan untu petunjuk arah transit, dalam hitungan detik sejak tengan malam, 1 Januari 1970 UTC. Kita dapat menentukan departure_time atau arrival_time, namun tidak keduanya. Perhatikan bahwa arrival_time harus ditentukan sebagai integer.[9]

e. *Departure_time* – Menentukan waktu keberangkatan. Kita bisa menentukan waktunya sebagai bilangan integer dalam hitungan detik sejak tengah malam, 1 Januari 1970 UTC.[9]
f. *Traffic_model (default to best_guess)* – Menentukan asumsi yang digunakan saat menghitung waktu lalu lintas. Setelan ini memengaruhi nilai yang dikembalikan di field duration_in_traffic dalam respon, yang berisi perkiraan waktu lalu lintas berdasarkan rata-rata historis. Parameter traffic_model hanya dapat ditentukan untuk petunjuk arah mengemudi dimana permintaan mencakup derpature_time. [9]
g. *Transit_mode* – Menentukan satu atau lebih mode transit pilihan.[9]
h. *Transit_routing_preference* – Menentukan preferensi untuk rute transit. Dengan menggunakan parameter ini, kita dapat memilah opsi yang dikembalikan, daripada menerima rute default terbaik yang dipilih oleh API.[9]

2.3.4.8 Pencarian Tempat

Google Places API Web Service memungkin kita untuk menanyakan informasi tempat pada berbagai kategori, seperti : tempat usaha, tempat menarik, lokasi geografis, dan lainnya. Kita dapat mencari tempat baik dengan kedekatan atau *string* teks.[9]

2.3.4.9 Permintaan Pencarian Sekitar

Penelusuran di sekitar memungkinkan kita mencari tempat dalam area tertentu. Kita dapat memperbaiki permintaan pencarian kita dengan menyediakan kata kunci atau menentukan jenis tempat yang kita cari.[9]

Permintaan penelusuran di sekitar adalah *URL HTTP* dari *form* berikut[9] :
https://maps.googleapis.com/maps/api/place/nearbysearch/output?parameters

Parameter tertentu diperlukan untuk memulai permintaan Pencarian Terdekat. Seperti standar dalam *URL*, semua parameter dipisahkan menggunakan karakter *ampersand* (&).[9]

1. Parameter Wajib
    a. *Key* – Kunci API aplikasi yang dibangun. Kunci ini mengidentifikasi aplikasi yang dibangun untuk tujuan pengelolaan kuota dan sehingga tempat yang ditambahkan dan aplikasi yang dibangun.
    b. *Location* - Lintang/bujur di mana untuk mengambil informasi tempat. Ini harus ditentukan sebagai garis lintang, bujur.
    c. *Radius* – Tetapkan jarak (dalam meter) di mana untuk mengembalikan hasil tempat radius maksimum yang diizinkan adalah 50.000 meter. Perhatikan bahwa radius tidak disertakan jika rankby=distance.
    d. Jika *rankby=distance* ditentukan, maka satu atau beberapa kata kunci, nama, atau jenis diperlukan.[9]
2. Parameter Optional
    a. *Keyword* – Sebuah istilah yang akan disesuaikan dengan semua konten yang telah diindeks Google untuk tempat ini, termasuk namun tidak terbatas pada nama, jenis, dan alamat, serta ulasan pelanggan dan konten pihak ketiga lainnya.
    b. *Language* – Kode bahasa, yang menunjukkan bahasa mana hasilnya harus dikembalikan, jika memungkinkan.
    c. *Minprice and maxprice* (optional) – Batasi hasil hanya pada tempat di kisaran yang ditentukan. Nilai yang valid berkisar antara 0 (paling terjangkau) sampai 4 (aling mahal), inklusif. Jumlah pasti yang ditunjukkan oleh nilai tertentu akan bervariasi dari satu wilayah ke wilayah lainnya.
    d. *Name* – Sebuah istilah yang harus disesuaikan dengan semua konten yang telah diindeks Google untuk tempat ini. Setara dengan kata kunci. Bidang nama tidak lagi dibatasi untuk menempatkan nama.
    e. *Opennow* – Mengembalikkan hanya tempat-tempat yang terbuka untuk bisnis pada saat query dikirim. Tempat yang tidak menentukan jam buka di basis data *Google Places* tidak akan dikembalikkan jika kita memasukkan parameter ini ke *query* kita.[9]

f. *Rankby* – Menentukan urutan hasil yang tercantum. Perhatikan bahwa *rankby* tidak boleh disertakan jika radius ditentukan. Kemungkinan values :

a) *Prominence (default)* – Pilihan ini mengurutkan hasil berdasarkan kepentingan mereka. Peringkat akan menyukai tempat- tempat menonjol di dalam area yang ditentukan. Keunggulan dapat dipengaruhi oleh peringkat suatu tempat di indeks *Google*, popularitas global, dan faktor lainnya.

b) *Distance* – Pilihan ini bias mencari hasil dalam urutan naik menurut jarak dari *location* yang ditentukan. Bila *distance* ditentukan, satu atau beberapa kata *keyword*, *name*, atau *type* diperlukan. [9]

c) *Type* - Batasi hasilnya ke tempat yang sesuai dengan jenis yang ditentukan. Hanya satu jenis yang dapat ditentukan (jika lebih dari satu jenis disediakan, semua jenis mengikuti entri pertama diabaikan).[9]

d) *Pagetoken* – Mengembalikan 20 hasil berikutnya dari pencarian yang sebelumnya dijalankan. Menetapkan parameter pagetoken akan melakukan pencarian daengan parameter yang sama yang digunakan sebelumnya – semua parameter selain pagetoken akan diabaikan.

e) *Region* - Kode wilayah, yang ditentukan sebagai kode cctld (*country code top-level domain*) dua karakter. Kebanyakan kode ccltd identic dengan kode ISO 3166-1, dengan beberapa pengecualian. Parameter ini hanya akan berpengaruh, tidak sepenuhnya membatasi, hasil pencarian. Jika hasil yang lebih relevan ada di luar wilayah uang ditentukan, mereka mungkin disertakan.[9]

2.3.5 *Twitter* API

API Twitter atau *Application Programming Interface* (API) *twitter* adalah suatu aplikasi yang dibuat oleh *twitter* untuk mempermudah *developer* untuk mengakses informasi yang ada di *twitter*. **[**10**]**

Hal yang akan di ambil dari API twitter adalah :

1. *Tweets*

Tweets adalah bagian dasar dari *twitter*, karena *tweets* adalah status yang di tweetkan oleh *user*. Termasuk dengan *reply*, *retweet*, *delete*, dan *likes*. Untuk mencari *tweet* berdasarkan *keyword URL* yang digunakan untuk pencarian ini adalah ”*https://api.twitter.com/1.1/search/tweets.json”* dengan menggunakan parameter *“?q”* untuk mencari kata yang akan dicari dan parameter *“?count”* untuk menentukan jumlah *tweet* yang akan diunduh.[10]

2. *Users/Accounts*

Seseorang yang telah mendaftar pada *twitter*, bisa siapa saja dan siapapun. Mereka dapat mengetweet, mengikuti, membuat jadwal dan lain-lain.[10]

2.3.6 *Instagram* API

Platform API Instagram dapat digunakan untuk membuat non-otomatis, otentik, aplikasi dan *service* kualitas tinggi yang : membantu individu membagikan konten mereka sendiri dengan aplikasi pihak ke-3. Membantu mereka dan pengiklan memahami, mengelola pelanggan dan hak media mereka. Membantu penyiar dan penerbit menemukan konten, mendapatkan hak digital terhadap media, dan membagikan media dengan atribusi yang tepat. **[**11**]**

Ada beberapa bagian yang penting dalam *API Instagram* setelah anda melakukan register, anda akan mendapatkan kemudahan untuk memulai meminta data dari Instagram, ini berada di *API Endpoints*. Semua endpoints hanya bisa di akses *via https* dan berlokasi di *api.instagram.com* dan ada beberapa *endpoints* penting yang akan ditampilkan yaitu[11] :

1. *Users*

Pengguna dari Instagram yang sudah terdaftar, dan dapat melakukan upload foto atau video lalu melike foto atau video juga melakukan *search*.

a. *Get/users/self* : biasanya digunakan untuk mendapatkan informasi tentang pengguna dari *acces_token* dan menggunakan *URL* sebagai berikut : *“”*
b. *Get/users/user-id* : biasanya digunakan untuk mendapatkan informasi mengenai pengguna *public_content* ruang lingkup diperlukan jika pengguna bukan pemilik dari *access_token* dan biasanya menggunakan *URL* sebagai

berikut : *https://api.instagram.com/v1/users/{user-id}/?access_token=ACCESS-TOKEN*

c. *Get/user/self/media/recent* : biasanya digunakan untuk mendapatkan media yang paling baru di publikasikan oleh pemilik dari *access_token* dan biasanya menggunakan *URL* sebagai berikut : *"https://api.instagram.com/v1/users/self/media/recent/?access_token=ACCESS-TOKEN"*

d. *Get/users/user-id/media/recent* : biasanya digunakan untuk mendapatkan media yang paling baru di publikasikan pemilik dari *access_token* dan biasanya menggunakan *URL* sebagai berikut : "*https://api.instagram.com/v1/users/{user-id}/media/recent/?access_token=ACCESS-TOKEN"*

e. Get/users/self/media/liked : biasanya digunakan untuk mendapatkan list media yang paling baru di liked oleh owner dari access_token biasanya menggunakan URL sebagai berikut : "*https://api.instagram.com/v1/users/self/media/liked?access_token=ACCESS-TOKEN".*

f. *Get/users/search* : biasanya digunakan untuk mendapatkan list user yang sesuai dengan query dan biasanya menggunakan URL sebagai berikut : "*https://api.instagram.com/v1/users/search?q=jack&access_token=ACCESS-TOKEN"* [11]

2. *Media*

Pengguna dari Instagram yang sudah mendaftar sebagai client akan dapat menggunakan :

a. *Get/media/media-id* : biasanya digunakan untuk mendapatkan tentang media object. Memakai *type field* untuk membedakan antara media gambar dan video di dalam respon. Anda juga akan mendapatkan *user_has_liked field* yang mana akan memberi tahu anda dimana *owner* dari *access_token* telah menyukai media ini. Biasanya menggunakan *URL* sebagai berikut : "*https://api.instagram.com/v1/media/{media-id}?access_token=ACCESS-TOKEN"*

b. *Get/media/shortcode/shortcode* : *Endpoint* ini mengembalikan respon yang sama sebagai *Get/media/media-id*. Biasanya menggunakan *URL* : "*https://api.instagram.com/v1/media/shortcode/D?access_token=ACCESS-TOKEN"*

c. *Get/media/search* : Biasanya digunakan untuk mencari media baru di *area* yang telah diberikan. Biasanya menggunakan *URL* : "*https://api.instagram.com/v1/media/search?lat=48.858844&lng=2.294351&access_token=ACCESS-TOKEN"* [11]

3. *Comments*

Pengguna yang sudah terdaftar juga dapat mendapatkan *Comment endpoints*.

a. *Get/media/media-id/comments* : Biasanya ini digunakan untuk mendapatkan list komentar terbaru di *object* media. Cakupan *public_content* dibutuhkan untuk media yang bukan milik dari pemilik *access_token*. Biasanya menggunakan *URL* : "*https://api.instagram.com/v1/media/{media-id}/comments?access_token=ACCESS-TOKEN"*

b. *Post/media/media-id/comments* : Biasanya digunakan untuk membuat sebuah komen pada objek media dengan beberapa peraturan seperti berikut :

1) Panjang total komentar tidak boleh melebihi 300 karakter.
2) Komentar tidak boleh berisi lebih dari 4 hashtag.
3) Komentar tidak boleh berisi lebih dari 1 URL.
4) Komentar tidak bisa terdiri dari semua huruf besar.

Cakupan *public_content* dibutuhkan untuk media yang bukan bagian dari pemilik *access_token*. Biasanya menggunakan URL :

"Curl –F 'access_token=ACCESS-TOKEN'
-F 'text=This+is+my+comment'
https://api.instagram.com/v1/media/media/{media-id}/comment"

c. *Del/media/media-id/comments/comment-id* : Biasanya digunakan untuk menghapus sebuah komentar baik pada objek media pengguna yang diotektikasi atau ditulis oleh pengguna yang diautentikasi. Biasanya

menggunakan *URL* : "*curl -X DELETE https://api.instagram.com/v1/media/{media-id}/comments/{comment-id}?access_token=ACCESS-TOKEN"*

4. *Tags*

Berlaku sama bagi yang sudah mendaftar sebagai *client* akan dapat mengakses *Tag Endpoints* yang berup :

a. *Get/tags /tag-name* : Biasanya digunakan untuk mendapatkan informasi tentang objek *tag*. Biasanya menggunakan *URL* : "*https://api.instagram.com/v1/tags/{tag-name}?access_token=ACCESS-TOKEN"*

b. *Get/tags/media/recent* : Biasanya digunakan untuk mendapatkan list dari daftar media yang baru ditandai. Biasanya menggunakan *URL* : "*https://api.instagram.com/v1/tags/{tag-name}/media/recent?access_token=ACCESS-TOKEN"*

c. *GET/tags/search* : Biasanya digunakan untuk mencari tags dengan nama. Biasanya menggunakan *URL* : "*https://api.instagram.com/v1/tags/search?q=snowy&access_token=ACCESS-TOKEN"* [11]

5. *Location*

Pengguna yang telah mendaftar akan mendapatkan *access endpoint* dari *location*. *Endpoint* tersebut adalah :

a. *GET/locations/location-id* : Biasanya digunakan untuk mendapatkan informasi tentang lokasi dan biasanya menggunakan *URL* : *"https://api.instagram.com/v1/locations/{location-id}?access_token=ACCESS-TOKEN"*

b. *GET/locations/location-id/media/recent* : Biasanya digunakan untuk mendapatkan daftar objek media baru dari lokasi tertentu dan biasanya menggunakan *URL* : "*https://api.instagram.com/v1/locations/{location-id}/media/recent?access_token=ACCESS-TOKEN"*

c. *GET/locations/search* : Biasanya digunakan untuk mencari untuk lokasi dengan menggunakan koordinat geografik dan biasanya menggunakan *URL* : *"https://api.instagram.com/v1/locations/search?lat=48.858844& lng=2.294351&access_token=ACCESS-TOKEN"* [11]

2.3.7 *API Zomato*

API Zomato memberikan akses ke informasi paling baru dan paling lengkap untuk lebih dari 1.5 juta juta restoran di lebih dari 10.000 kota di dunia. **[**12**]** Untuk mendapatkan *API* yang dibutuhkan kita dapat menggunakan kategori :

1. *Location*

Untuk mendapatkan API *Get Foodie Index, Nightlife Index, Top Cuisines* dan *Best Rated restaurants* di lokasi yang sudah diberikan dengan menggunakan *GET/location_details*. Sedangkan untuk mendapatkan Zomato *locations* dengan *keywords,* dibekali dengan koordinat untuk pencarian yang lebih bagus dengan menggunakan *get/locations* [12]

2. *Restaurant*

Untuk mendapatkan data *API restaurant* dengan mendapatkan informasi detail restaurant dengan menggunakan Zomato ID, jika ingin mengakses foto dan *reviews* anda harus menjadi partner terlebih dahulu dengan Zomato. Tapi jika tidak, hanya dibutuhkan *get/restaurant*. [12]

2.4 *Location Based Service* (LBS)

Layanan lokasi dapat didefinisikan sebagai layanan yang mengintegrasikan lokasi atau posisi perangkat mobile dengan informasi lainnya sehingga memberikan nilai tambah bagi pengguna. [**10**]

Location Service memiliki tradisi yang panjang. Sejak tahun 1970an, *Departement of Defense A.S* telah mengoperasikan sistem penentuan posisi global (*GPS*), infrastruktur satelit yang melayani penentuan posisi orang dan objek. Awalnya, *GPS* dipahami untuk tujuan militer, namun pemerintah A.S. memutuskan pada tahun 1980an untuk membuat data posisi sistem tersedia secara

bebas untuk industri lain di seluruh dunia. Sejak saat itu, banyak industri telah mengambil kesempatan untuk mengakses data posisi melalui *GPS* dan sekarang menggunakannya untuk meningkatkan produk dan layanan mereka. Misalnya, industri otomotif telah mengintegrasikan sistem navigasi ke mobil untuk beberapa waktu.[13]

Dalam sistem penentuan posisi tradisional, informasi lokasi biasanya diturunkan oleh perangkat dan dengan bantuan sistem satelit (misalnya, penerima *GPS*) .1 Namun, minat yang meluas pada layanan berbasis lokasi (*LBS*) dan teknologi yang mendasarinya seperti yang dibahas di buku ini benar-benar mulai berkembang hanya di akhir tahun 1990an, ketika teknologi pelokalan tipe baru dan minat pasar baru terhadap layanan data dipicu oleh operator jaringan seluler. Pada sekitar tahun 1997, jaringan bergerak banyak digunakan di Eropa, Asia, dan Amerika Serikat, dan pendapatan dari layanan telepon telah terbukti signifikan bagi operator seluler. Namun, meski layanan suara seluler terus menjadi penghasil pendapatan utama telco, pertumbuhan telepon seluler terbatas dan harga per menit menurun.[13]

2.5 *Global Positioning System* (GPS)

Global Positioning System (*GPS*) adalah sistem navigasi berbasis satelit yang dikembangkan oleh Departemen Pertahanan A.S pada awal 1970-an. Awalnya, *GPS* dikembangkan sebagai sistem militer untuk memenuhi kebetuhan militer A.S. namun, kemudian tersedia untuk warga sipil., dan sekarang sistemnya dual-use yaitu bisa diakses oleh militer dan warga sipil. **[**14**]**

GPS memberikan informasi postioning dan timing yang terus menerus, dimana saa di dunia dalam kondisi cuaca apapun. Karena melayani jumlah pengguna yang tidak terbatas serta digunakan untuk alasan keamanan, *GPS* adalah sistem satu arah (pasif). Artinya pengguna hanya bisa menerima sinyal satelit.[14]

2.6 *Text Mining*

Berikut merupakan penjelasan mengenai *text mining* :

2.6.1 Definisi

Text mining dapat difeinisikan secara luas sebagai proses pengetahuan intesif dimana pengguna berinteraksi denngan koleksi dokumen dari waktu ke waktu dengan menggunakan rangkaian alat analisis. Dicara yang serupa dengan data mining, *text mining* berusaha mengekstrak informasi yang bermanfaat dari sumber data melalui identifikasi dan ekplorasi pola menarik. Namun, dalam text mining, sumber data adalah kumpulan dokumen, dan pola yang menarik tidak ditemukan di antara catatan database yang diformalkan tapi juga di data tekstual tidak terstruktur dalam koleksi dokumen ini. **[**15**]**

2.6.2 Proses Text Mining

Pada tingkat fungsional, mengikuti model umum *text mining* yang disediakan oleh beberapa aplikasi *data mining* klasik dan dengan demikian dibagi menjadi empat bidang utama : (a) *preprocessing tasks*, (b) *core mining operations*, (c) *presentation layer components*, dan (d) *refinement techniques*. **[**15]

1. ***Preprocessing Tasks,*** mencakup semua rutinitas, proses, dan metode yang dibutuhkan untuk mempersiapkan data operasi pengetahuan inti *text mining*. tugas-tugas ini biasanya berpusat pada aktivitas preprocessing dan kategorisasi sumber data. *Preprocessing tasks* umumnya mengubah informasi dari masing-masing orisinil sumber data menjadi format kanonik sebelum menerapkan berbagai jenis fitur metode ekstrasi terhadap dokumn-dokumen ini untuk membuat kumpulan dokumen baru. Terkadang, tugas preprocessing bahkan mungkin mencakup metode yang dirancang khusus digunakan dalam pengambilan awal data “mentah” yang sesuai dari data asli yang berbeda sumber.[15]
2. ***Core Mining Operations***, adalah jantung dari sistem *text mining* dan termasuk pola penemuan, analisis kecenderungan, dan algoritma penemuan pengetahuan inkremental. Diantara pola yang umum digunakan untuk penemuan pengetahuan dalam data tekstual distribusi (dan proporsi), rangkaian konsep yang sering dan cukup sering. Operasi *core mining* juga dapat memperhatikan perbandingannya. Sistem *text mining* yang canggih atau

berorientasi domain, atau keduanya, bisa juga meningkatkan kualitas berbagai operasi mereka dengan memanfaatkan latar belakang sumber-sumber pengetahuan. Operasi *core mining* ini dalam sistem text mining juga disebut sebagi, kolektif, sebagai yang sudah diketahui proses *distilation*.[15]

3. ***Presentation Layer Components***, termasuk GUI dan fungsi pola penjelajahan serta akses kepada *query* bahasa. Alat visualisasi dan *user-facing query* dan *optimizers* juga termasuk alat berbasis karakter aau grafis untuk membuat atau memodifikasi konsep cluster serta untuk membuat profil beranotasi untuk spesifik konsep atau pola.[15]
4. ***Refinement Techniques***, paling sederhana, sertakan metode yang menyaring informasi yang berlebihan dan cluster data yang mungki terkait tapi yang mungkin akan tumbuh, dalam sistem *text mining* tertentu, untuk mewakili rangkaian lengkap penekanan, pemesanan, pemangkasanan, generalisasi, dan pendekatan pengelompokkan yang ditujukan untuk optimasi penemuan. Teknik ini juga telah digambarkan di *post processing*.[15]

2.7 Media Sosial

Berikut ini adalah definisi dari media sosial yang berasal dari berbagai literatur penelitian (lihat Fuchs, 2014: 35-36):

1. Menurut Mandibergh (2012), media sosial adalah media yang mewadahi kerja sama di antara pengguna yang menghasilkan konten (*user-generated content*). **[**16**]**
2. Menurut Shirky (2008), media sosial dan perangkat lunak sosial merupakan alat untuk meningkatkan kemampuan pengguna untuk berbagi (*to share*), bekerja sama (*to co-operate*) di antara pengguna dan melakukan tindakan secara kolektif yang semuanya berada di luar kerangka institusional maupun organisasi.[16]
3. Boyd (2009) menjelaskan media sosial sebagai kumpulan perangkat lunak yang memungkinkan individu maupun komunias untuk berkumpul, berbgai, berkomunikasi, dan dalam kasus tertentu saling berkolaborasi atau bermain. Media sosial memiliki kekuatan pada *user-generated content* (UGC) di

mana konten dihasilkan oleh pengguna, bukan oleh editor sebagaimana di institusi media massa.[16]

4. Menurut Van Dijk (2013), media sosial adalah platform media yang memfokuskan pada eksistensi pengguna yang memfasilitasi mereka dalam beraktivitas maupun berkolaborasi. Karena itu, media sosial dapat dilihat sebagai medium (*fasilitator*) online yang menguatkan hubungan antar pengguna sekaligus sebagai sebuah ikatan sosial.[16]
5. Meike dan Young (2012) mengartikan kata sosial media sebagai konvergensi antara komunikasi personal dalam arti saling berbagi di antara individu (*to be shared one-to-one*) dan media publik untuk berbagi kepada siapa saja tanpa ada kekhususan individu.[16]

2.7.1 Definisi Media

Terlepas dari cara pandang melihat media dari bentuk dan teknologinya, pengungkapan kata "media" bisa dipahami dengan melihat dari proses komunikasi itu sendiri (Meyrowitz, 1999; Moores, 2005; Williams, 2003). Proses terjadinya komunikasi memerlukan tiga hal, yaitu objek, organ, dan medium. Saat menyaksikan sebuah program ditelevisi, televise adalah objek dan mata adalah organ. Perantara antara televisi dan mata adalah gambar atau visual. Contoh sederhana ini membuktikan bahwa media merupakan wadah untuk membawa pesan dari proses komunikasi. [16]

Beragam kriteria bisa dibuat untuk melihat bagaimana media itu. Ada yang membuat kriteria media berdasarkan teknologinya, seperti media cetak yang menunjukkan bahwa media tersebut dibuat dengan mesin cetak dan media elektronik yang dihasilkan dari perangkat elektronik. Dari sumber atau organ yang menjelaskan bagaimana cara mendapatkan atau bagaimana kode-kode pesan itu di olah, misalnya media audio-visual yang diakses menggunakan organ pendengaran dan penglihatan. Ada juga yang menuliskannya berdasarkan bagaimana pesan itu disebarkan. Contohnya, pesan, seperti stasiun televisi, dan pesan itu disebarkan serta bisa dinikmati oleh siapa saja asal memilii pesawat televisi. Atau berdasarkan teknologi, pola penyebaran, sampai pada bagaimana khalayak

mengakses media, seperti media lama (*old media*) dan media baru (*new media*).[16]

2.7.2 Definisi Sosial

Kata “sosial” dalam media sosial secara teori semestinya didekati oleh ranah sosiologi. Inilah yang menurut Fuchs (2014) ada beberapa pertanyaan dasar ketika melihat kata sosial, misalnya terkait dengan informasi dan kesadaran. Ada pertanyaan dasar, seperti apakah individu itu adalah manusia yang selalu berkrakter sosial atau individu itu baru dikatakan sosial ketika ia secara sadar melakukan interaksi. Bahkan, dalam teori sosiologi disebutkan bahwa media pada dasarnya adalah sosial karena media merupakan bagian dari masyarakat dan aspek dari masyarakat yang dipresentasikan dalam bentuk perangkat teknologi yang digunakan.[16]

2.7.3 Karakteristik Media Sosial

Media sosial merupakan salah satu *platform* yang muncul di media siber. Karena itu, melihat media sosial yang ada tidak jauh berbeda dengan karakteristik yang dimiliki oleh media siber. Bahkan Gane dan Beer (2008) secara khusus memberikan konsep-konsep kunci untuk memahami media siber.

Meski karakteristik media siber bisa dilihat melalui media sosial, media sosial memiliki karakteristik khusus yang tidak dimiliki oleh beberapa jenis media siber lainnya. Ada batasan-batasan dan ciri khusus tertentu yang hanya dimiliki oleh media sosial di banding dengan media lainnya. Salah satunya adalah media sosial beranjak dari pemahaman bagaimana media tersebut digunakan sebagai sarana sosial di dunia virtual.[16]

Adapun karakteristik media sosial, yaitu :

1. Jaringan (*Network*).
2. Informasi (*Information*).
3. Arsip (*Archive*).
4. Interaksi (*Interactivity*).
5. Simulasi sosial (*Simulation of Society*).

6. Konten oleh pengguna (*User-Generated Content*).[15]

2.7.4 Jenis-jenis Media Sosial

Banyak sumber, terutama liputan media maupun kajian literatur, yang membagi jenis sosial media. Ada yang berdasarkan model jaringan yang terbentuk, berdasarkan karakteristik penggunanya, sampai berdasarkan pada *file* atau berkas apa saja yang disebarkan (*sharing*) di antara pengguna. Dapat di ambil kesimpulan dari berbagai sumber tersebut setidaknya ada enam kategori besar untuk melihat pembagian media sosial, yakni [16] :

1. Media jejaring sosial (*Social Networking*).
2. Jurnal online (*Blog*).
3. Jurnal online sederhana atau mikroblog (*Micro-blogging*).
4. Media berbagi (*Media Sharing*).
5. Penanda sosial (*Social Bookmarking*).
6. Media konten bersama atau Wiki.

2.8 *Java*

Java adalah bahasa *object-oriented* baru yag mendapat perhatian luas dari industry dan akademisi. Java dikembangkan oleh James Gosling dan timnya di Sun Microsystems di California. Bahasa ini didasarkan pada C dan C++ dan pada awalnya ditujukan untuk menulis program yang mengendalikan peralatan konsumen seperti toaster, microwave, oven, dan lainnya. Bahasa itu pertama kali disebut Oak, dinamai setelah pohon oak di luar dari kantor Gosling, tapi nama itu sudah diambil, jadi tim menamainya Java.

Java sering digambarkan sebagai bahasa pemograman web karena penggunaannya di meluis program yang disebut applet yang berjalan dalam browser web. Browser web untuk mengeksekusi applet Java. Applet memungkinkan lebih dinamis dan fleksibel di Internet, dan fitur ini saja membuat Java menjadi bahasa yang menarik untuk dipelajari. Namun, kita tidak terbatas pada menulis applet di Java. Kita bisa juga menulis aplikasi Java. Aplikasi Java adalah sebuah program lengkap yang berdiri sendri tidak memerlukan Web

Browser . aplikasi Java analog terhadap program yang kita tulis dalam bahasa pemograman lainnya.[17]

2.9 *Unified Modelling Language* (UML)

Unified Modelling Language (UML) adalah salah satu alat bantu pengembangan sistem yang berorientasi obyek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang sistem untuk membuat cetak biru (*design*) atas visi (*ide*) mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain. **[**18**]**

2.9.1 *Diagram Use Case*

Use case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara pengguna sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah system dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan system disebut *scenario.* Setiap *scenario* mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, system yang lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat bisa dikatakan usecase adalah serangkaian *scenario* yang digabungkan bersama-sama oleh tujuan umum pengguna.

Di dalam *usecase* biasanya pengguna disebut dengan actor. *Actor* adalah sebuah peran yang bisa dimainkan oleh pengguna dalam interaksinya dengan sistem.[18]

2.9.2 *Diagram Sequence*

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah scenario. Diagram ini menunjukkan sejumlah contoh objek dan *message* (pesan) yang diletakkan diantara obyek-obyek ini di dalam *use case*.

Komponen utama sequence diagram terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan denga *progress vertical*.[18]

2.9.3 *Diagram Activity*

Activity Diagram adalah teknik untuk mendeskripsikan logika *procedural*, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaanya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku parallel sedangkan *flowchart* tidak bisa.[18]

2.9.4 *Diagram Class*

Diagram Class menggambarkan jenis objek dalam sistem dan aneka ragamnya jenis hubungan statis yang ada diantara mereka. *Diagram Class* juga menunjukkan sifat dan operasi kelas dan batasan yang berlaku untuk cara benda terhubung *UML* menggunakan istilah fitur sebagai istilah umum yang mencakup properti dan operasi kelas.[18]

2.10 Analisis dan Desain Berbasis Objek

Analisis dan desain berorientasi objek adalah cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, satu entitas. Model berorientasi objek bermanfaat untuk memahami masalah, komunikasi dengan ahli aplikasi, pemodelan suatu organisasi, menyiapkan dokumentasi serta perancangan program dan basis data. [**15**]

2.10.1 Karakteristik dari Objek

Identitas berari bahwa data diukur mempunyai nilai tertentu yang membedakan entitas dan disebut objek. Suatu paragraph dari dokumen, suatu *windows* dari *workstation*, dan raja putih dari buah arsip dalam sistem, atau

koseptual seperti kebijakan pendajwalan dalam *multiprocessing* pada sistem operasi. Setiap objek mempunai sifat yang melekat pada identitasnya.[19]

2.10.2 Sejarah Berorientasi Objek

Pemograman berorientasi objek pertama-tama dibicarakan pada akhir tahun 1960 menggunakan bahasa SIMULA. Pada tahun 1970, bahasa pemograman Smaltalk dikembangkan oleh Xerox PARC. Pada saat itu sebagian lain dari dunia menggunakan COBOL dan PORTRAN dengan metode dekomposisi funsgional. Perubahan terjadi selama beberap decade sampai dikenalnya faktor utama, yaitu :

1. Konsep dasar berorientasi objek

   Konsep dasar pendekatan berorientas objek mencapai kematangan pada saat masalah analisis dan desai menjadi lebih diperhatikan dari pada masalah *coding*.[19]

2. Teknologi dasar pembangunan sistem

   Gagasan tentang *coding* sangat dipengaruhi oleh bahasa permograman yang tersedia. Untuk memikirkan pemograman terstruktur akan sulit bila yang tersedia adalah *Assembler*, tetapi lebih mudah bila menggunakan *Pascal*. Kesulitan yang sama akan terjadi pada saat pembuatan coding berorientasi objek bila bahasa pemograman yang dipilih COBOL atau FORTRAN. Hal ini lebih mudah dilakukan bila menggunaka C++, Borland, Pascal versi baru, Smalltalk dan Ada.[19]

3. Kondisi sistem

   Sistem yang dibangun pada saat ini berbeda dengan sistem beberapa tahun yang lalu. Sistem sekarang lebih besar, kompleks dan juga dapat merpakan sistem yang *interaktif*. Sebagian besar *code* pada sistem yang modern memperhatikan *user interface*, seperti manipulasi *window*, *icon*, *mouse*, dan lain-lain.[19]

4. Penggunaan model

   Banyak organisasi mendapatkan bahwa sistem yang dibangun pada saat ini cenderung berorientasi data dari pada sistem yang dibangun sekitar tahun

1970 dan 1980. Pada waktu itu penggunaan model telah mendapatkan prioritas lebih besar dibandingkan dengan kompleksitas fungsional.[19]

2.10.3 Metodologi Berorientasi Objek

Metode berorientasi objek diperkenalkan pada tahun 1980, menggunakan perangkat kerja dan teknik-teknik yang dibutuhkan dalam pengembangan sistem, yaitu *dynamic* dan *static object oriented model*, *state transition* dan *case scenario.*[19]

1. Analisis

   Analisis Berorientasi Objek atau *Object-Oriented Analysis* (OOA) dimulai dengan menyatakan suatu masalah, analisis membuat model situasi dari dunia nyata, menggambarkan sifat yang penting. Dalam menganalisa suatu sistem, analis harus bekerha dengan pihak yang membutuhkan sistem untuk memahami masalah tersebut dengan jelas. Model analisis adalah abstraksi yang ringkas dan tepat dari apa yang harus dilakukan oleh sistem, dan bagaimana melakukannya. Objek dalam model harus merupakan konsep domain dari aplikasi, dan bukan merupakan implementasi komputer seperti struktur data.[19]

2. Desain

   Desain Berorientasi Objek atau *Object-Oriented Design* (OOD) merupakan tahap lanjutan setelah Analisis Berorientasi Objek dimana tujuan sistem diorganiassi ke dalam sub-sistem berdasar struktur analisis dan arsitektur yang dibutuhkan. Desainer sistem (*System designer*) menentukan karakteristik penampilan ecara optimal, enentukan startegi memecahkan masalah, dan menentukan pilihan alokasi sumber daya.[19]

Desain model yang digunakan berdasarkan model analisis dengan dilengkapi rincian untuk implmentasi. Eksekusi dari desain objek (*object design*) adalah perencanaan struktur data dan *algoritma* yang diperlukan untuk implementasi setiap kelas.[19]