BAB 2
TINJAUAN PUSTAKA

2.1 Kuliner

Ada juga yang mendefinisikan bahwa Kuliner adalah suatu bagian hidup yang erat kaitannya dengan konsumsi makanan sehari-hari karena setiap orang memerlukan makanan yang sangat dibutuhkan sehari-hari. Mulai dari makanan yang sederhana hingga makanan yang berkelas tinggi dan mewah. Semua itu, membutuhkan pengolahan yang serba berkualitas dan bergizi. Sebenarnya kuliner merupakan bagian/sub daripada esensi gastronomi. Sementara istilah kuliner itu sendiri adalah masakan atau dalam bahasa dapur mempunyai sinonim/arti yang sama dengan istilah *cuisine* .

Secara harafiah, kuliner adalah kata yang biasa digunakan untuk merujuk pada sesuatu yang berhubungan dengan memasak atau profesi kuliner. Profesi kuliner sendiri dapat diartikan profesi untuk memasak atau mempersiapkan produk makanan, seperti chef, management restaurant, ahli penata diet, ahli gizi dan sebagainya. Produk makanan merupakan hasil proses pengolahan bahan mentah menjadi makanan siap dihidangkan melalui kegiatan memasak.

2.2 Kota Cikampek

Cikampek Kota adalah salah satu Kota dalam Wilayah Kabupaten Karawang yang berada pada ketinggian ± 2.529 m dpl, dengan jarak ke Kabupaten Karawang 21 km. Saat ini Kecamatan Cikampek telah mengalami pemekaran wilayah menjadi 2 kecamatan, yaitu Kecamatan Kotabaru dan Kecamatan Purwasari.

Kecamatan Cikampek merupakan jalur strategis pantura dari Jakarta menuju wilayah Jawa Tengah dan Jawa Timur. Di Cikampek pula terdapat persimpangan rel kereta api antara Jakarta - Bandung dan Jakarta - Cirebon. Setiap mendekati hari raya lebaran,

Cikampek selalu menjadi langganan di layar kaca. Kemacetan arus mudik di pintu tol Kopo menjadi barometer padatnya arus lalu lintas di Pantura.

Cikampek memiliki sebuah terminal yang sebelum dibangun Jalan Tol Cikampek - Jakarta adalah terminal tersibuk di Jawa Barat karena merupakan persimpangan Jakarta - Bandung dan Jakarta - Cirebon. Tetapi sejak jalan tol dibangun secara berangsur - angsur fungsi terminal Cikampek berkurang. Saat ini terminal Cikampek hanya berfungsi sebagai tempat pemberhentian / terminal bayangan Angkutan Kota dan sedikit Bus Antar kota – Antar provinsi yang melayani Jakarta - Cikampek dan Cikampek – Tasikmalaya. Kecamatan Cikampek yang memiliki batas-batas fisik sebagai berikut :

a. Sebelah Utara berbatasan dengan Kecamatan Tirtamulya
b. Sebelah Selatan berbatasan dengan Kecamatan Klari
c. Sebelah Barat berbatasan dengan Kecamatan Purwasari'
d. Sebelah Timur berbatasan dengan Kecamatan Kotabaru 61

Luas Kecamatan Cikampek pada Tahun 2008 adalah 4.638 Ha dengan kondisi wilayah sebagian besar merupakan dataran rendah dan bersifat agraris.

Sumber pendapatan utama kota Cikampek ini adalah perindustrian, pertanian dan perdagangan. Kota Cikampek dikenal sebagai Sentra Perdagangan dan Industri. Perekonomian warga Cikampek sebagian besar adalah pedagang,karyawan pabrik,pegawai negeri,dan swasta.

adapun luas wilayah Kecamatan Cikampek secara terperinci dapat dilihat pada Tabel 2.1:

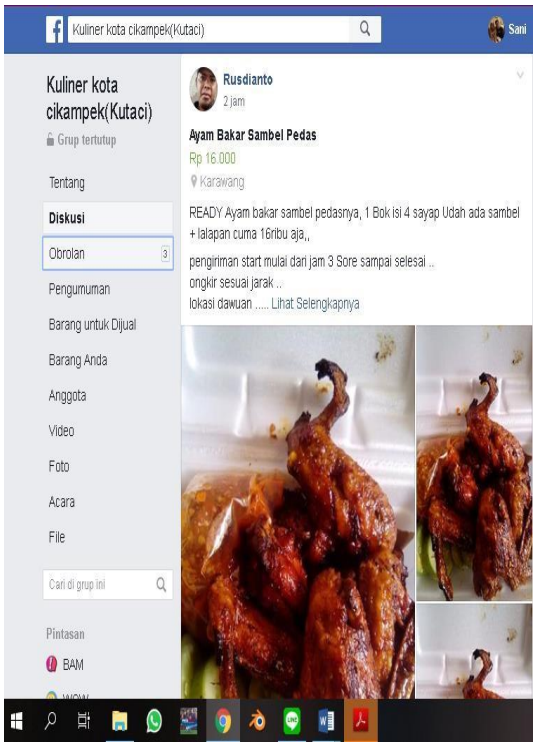
Tabel 2.1 Luas Wilayah Cikampek Tahun 2008

No	Desa	Luas Desa (Ha)	Proporsi(%)
1	Cikampek Kota	97	2,09
2	Cikampek Barat	194	4,18

No	Desa	Luas Desa (Ha)	Proporsi(%)
3	Cikampek Timur	112	2,41
4	Cikampek Selatan	122	2,63
5	Cikampek Pusaka	374	8,06
6	Kamojang	927	19,99
7	Dawuan Timur	293	6,32
8	Dawuan Tengah	322	6,94
9	Dawuan Barat	364	7,85
10	Kalihurip	1.833	39,52
Jumlah		4.638	100

2.3 Grup Kuliner Kota Cikampek (KUTACI)

Grup kuliner kota cikampek atau biasa di sebut juga (Kutaci) ini adalah salah satu grup *facebook* pecinta kuliner di kota cikampek, yang hingga saat ini terkenal di kota ini. Banyaknya masyarakat di kota ini memanfaatkan grup *facebook* untuk memesan makanan yang diinginkan ataupun hanya sekedar melihat-lihat saja makanan apa yang saat ini di gandrungi di kota tersebut. Awal mula adanya grup ini dikarenakan banyaknya permitaan masyarakat terhadap kuliner dan juga banyaknya masyarakat yang saat ini menggandrungi bisnis di bidang kuliner serta meningkatnya ketertarikan masyarakat terhadap kuliner yang semakin bervariasi. Maka dari itu masyarakat memberikan pemikirannya untuk membuat grup kutaci ini.



Gambar 2.1 screen shoot Grup kutaci

Sumber : grup *facebook* kuliner kota cikampek (kutaci)

2.4 Aplikasi Mobile

Aplikasi adalah program yang digunakan orang untuk melakukan sesuatu pada sistem komputer. *Mobile* dapat diartikan sebagai perpindahan yang mudah dari suatu tempat ke tempat yang lain, misalnya telepon *mobile* berarti bahwa terminal telepon yang dapat berpindah dengan mudah dari suatu tempat ke tempat lain tanpa terjadi pemutusan atau terputusnya komunikasi. Sistem aplikasi *mobile* merupakan aplikasi yang dapat digunakan walaupun pengguna berpindah dengan mudah dari suatu tempat ke-tempat lain tanpa terjadi pemutusan atau terputusnya komunikasi. Aplikasi ini dapat diakses melalui perangkat nirkabel seperti pager, seperti telepon seluler dan PDA [9].

Adapun karakteristik dari perangkat *mobile* adalah :

1. Ukuran yang kecil

Perangkat *mobile* memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan mobilitas.

2. Memori yang terbatas

   Perangkat *mobile* juga memiliki memory yang terkecil, yaitu *primary* (RAM), dan sekunder (Disk).

3. Daya proses yang terbatas

   Sistem *mobile* tidak setangguh desktop.

4. Mengkonsumsi daya yang rendah

   Perangkat *mobile* menghabiskan sedikit daya dibandingkan dengan mesin desktop.

5. Kuat dan dapat diandalkan

   Karena perangkat *mobile* selallu dibawa ke mana saja, mereka harus cukup kuat untuk menghadapi benturan, gerakan, dan sesekali terkena tetesan air.

6. Konektivitas yang terbatas

7. Perangkat *mobile* ini memiliki *bandwith* rendah, beberapa dari mereka kadang tidak tersambung.

8. Masa hidup yang pendek

   Perangkat – perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala

2.5 *Android*

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka.

Gambar 2.2 Logo *Android*

2.5.1 Pengertian *Android*

Android merupakan salah satu sistem operasi yang sangat berkembang saat ini, dengan berbasiskan Linux sistem operasi ini dirancang untuk mengembangkan perangkat seluler layar sentuh seperti *smartphone* dan juga komputer tablet. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi untuk digunakan oleh bermacam piranti gerak.

Salah satu penyebab mengapa sistem operasi *Android* begitu gampang diterima oleh pasar dan dengan cepatnya berkembang, itu dikarenakan *Android* menggunakan bahasa pemrograman *Java* serta kelebihannya sebagai *software* yang menggunakan basis kode komputer yang bisa didistribusikan secara terbuka (*open source*) sehingga pengguna dapat membuat aplikasi baru didalamnya. Dan hal tersebut mengakibatkan banyaknya pengembang *software* yang berbondong untuk mengembangkan aplikasi berbasis *Android*. Sehingga saat ini bila dibanding dengan OS yanglain untuk perangkat *handphone* dan PC tablet. *Android* adalah yang mempunyai dukungan aplikasi non berbayar terbanyak yang bisa diunduh oleh penggunanya melalui *Google Play*. Dengan terdapatnya fitur seperti browser, MMS, SMS, GPS, dan lain-lain maka sangat memudahkan penggunanya untuk mendapatkan informasi, mengetahui posisi, serta juga komunikasi antar para pengguna.

Adapun Adapun versi-versi *Android* yang pernah diliris sebagai berikut:

1. *Android* versi 1.1 (API level 1)

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

2. *Android* versi 1.5 cupcake (API level 3)

   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, mengunggah 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.

3. *Android* versi 1.6 Donut (API level 4)

   Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus kamera, camcorder dan galeri yang dintegrasikan CDMA / EVDO, 802.1x, VPN, Gestures, dan *Text-to-speech engine*; kemampuan dial 10 kontak teknologi *text to change speech.*

4. *Android* versi 2.0 - 2.1 Eclair (API level 5- 7)

   Pada 3 Desember 2009 kembali diluncurkan ponsel *Android* dengan versi 2.0/2.1 (*Eclair*), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan *Google Maps* 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 MP, digital *Zoom*, dan *Bluetooth* 2.1.

5. *Android* versi 2.2 – 2.2.3 Froyo :Frozen Yoghurt (API level 8)

Pada 20 Mei 2010, *Android* versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain 11 dukungan *Adobe Flash 10.1*, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, *intergrasi V8 JavaScript engine* yang dipakai *Google* Chrome yang mempercepat kemampuan menerjemah pada *browser*, pemasangan aplikasi dalam SD *Card*, kemampuan WiFi Hotspot portabel, dan kemampuan *auto update* dalam aplikasi *Android Market.*

6. *Android* versi 2.3 – 2.3.7 Gingerbread (API level 9 - 10)

   Android Gingerbread di rilis pada 6 Desember 2010. Perubahan-perubahan umum yang didapat dari *Android* versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (*reverb, equalization, headphone virtualization*, dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

7. *Android* versi 3.0 – 3.2 Honeycomb (API level 11 - 13)

   Android Honeycomb di rilis pada awal 2012. Merupakan versi *Android* yang dirancang khusus untuk device dengan layar besar seperti Tablet PC. Fitur baru yang ada pada *Android Honeycomb* antara lain yaitu dukungan terhadap *prosesso rmulticore* dan grafis dengan *hardware acceleration. User Interface* pada *Honeycomb* juga berbeda karena sudah didesain untuk tablet. Tablet pertama yang memakai *Honeycomb* adalah tablet Motorola Xoom yang dirilis bulan Februari 2011. Selain itu sebuah perangkat keras produksi Asus bernama *Eee Pad Transformer* juga menggunakan OS *Android Honeycomb* dan diharapkan akan masuk ke pasaran Indonesia pada Mei2011.

8. *Android* versi 4.0 – 4.0.4 Ice cream sandwich (API level 14 -15)

Android Ice Cream Sandwich diumumkan secara resmi pada 10 Mei 2011 di ajang *Google I/O Developer Conference* (San Francisco), pihak *Google* mengklaim *Android Ice Cream Sandwich* akan dapat digunakan baik di *smartphone* ataupun tablet. *Android Ice Cream Sandwich* membawa fitur *Honeycomb* untuk *smartphone* serta ada penambahan fitur baru seperti membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan *control*, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara offline, dan berbagi informasi dengan menggunakan NFC. Ponsel pertama yang menggunakan sistem operasi ini adalah Samsung Galaxy Nexus.

9. *Android* versi 4.1 – 4.3 Jelly bean (API level 16 - 18)

   Android Jelly Bean juga diluncurkan pada acara *Google* I/O 10 Mei 2011 yang lalu. *Android* versi ini membawa sejumlah keunggulan dan fitur baru, diantaranya peningkatkan *input keyboard*, desain baru fitur pencarian, UI yang baru dan pencarian melalui *Voice Search* yang lebih cepat. Versi ini juga dilengkapi *Google Now* yang dapat memberikan informasi yang tepat pada waktu yang tepat pula. Salah satu kemampuannya adalah dapat mengetahui informasi cuaca, lalu-lintas, ataupun hasil pertandingan olahraga. Sistem operasi *Android Jelly Bean* 4.1 pertama kali digunakan dalam produk tablet Asus, yakni *Google* Nexus7.

10. *Android* versi 4.4 kitkat (API level 19)

    Beberapa waktu yang lalu *Google* secara resmi memperkenalkan sistem operasi terbarunya, yaitu update Sistem Operasi *Android* versi 4.4 yang diberi nama KitKat. Kabar tersebut cukup mengagetkan banyak pihak terlebih pengambilan nama untuk versi terbaru *Android* tersebut ternyata diluar perkiraan. Sebelumnya banyak kabar yang berhembus bahwa update sistem operasi *Android* terbaru yang akan diusung oleh *Google* akan dinamai *Android Key Lime Pie*, namun ternyata *Google* menepis semua rumor tersebut dengan memperkenalkan *Android* KitKat.

11. *Android* versi 5.0 Lollipop (API level 21)

Android 5.0 pertama kali diperkenalkan di bawah codename "*Android* L" pada 25 Juni 2014 selama presentasi keynote pada konferensi pengembang *Google* I / O. Di samping Lollipop, presentasi difokuskan pada sejumlah platform *Android* yang berorientasi dan teknologi baru, termasuk *Android* TV, pada platform *Android* Auto, dapat dipakai pada platform komputasi *Android* Wear, dan platform pelacakan kesehatan *Google* Fit.Bagian dari presentasi didedikasikan untuk bahasa desain cross-platform baru yang disebut sebagai "material design". Memperluas pada "kartu" motif pertama kali terlihat di *Google* Now, adalah desain dengan peningkatan penggunaan layout berbasis grid, animasi dan transisi responsif, padding, dan efek kedalaman seperti pencahayaan dan bayangan.

12. *Android* versi 6.0 Marshmallow (API level 23)

    Android Marshmallow memberikan dukungan asli untuk pengenalan sidik jari, memungkinkan penggunaan sidik jari untuk membuka perangkat dan otentikasi Play Store dan pembelian *Android* Pay; API standar juga tersedia untuk melaksanakan otentikasi berbasis sidik jari dalam aplikasi lain. *Android* Marshmallow mendukung USB Type-C, termasuk kemampuan untuk menginstruksikan perangkat untuk mengisi daya perangkat lain melalui USB. Marshmallow juga memperkenalkan "pranala yang diverifikasi" yang dapat dikonfigurasi untuk membuka langsung dalam aplikasi tertentu mereka tanpa petunjuk pengguna lanjut.

13. *Android* versi 7.0 – 7.1 Nougat (API level 24 – 25)

    Android "Nougat" rilis 7.0 besar dari sistem operasi *Android*. Ini pertama kali dirilis sebagai pratinjau pengembang pada tanggal 9 Maret 2016, dengan gambar pabrik untuk perangkat Nexus saat ini, serta dengan "Program Beta" baru yang memungkinkan perangkat yang didukung ditingkatkan versinya ke versi *Android* Nougat melalui over-the-air update. Rilis terakhir adalah pada tanggal 22 Agustus 2016. Pratinjau akhir pembuatannya dirilis pada tanggal 18 Juli 2016, dengan nomorNPD90G.Padatanggal 19 Oktober 2016, *Google* merilis *Android* 7.1

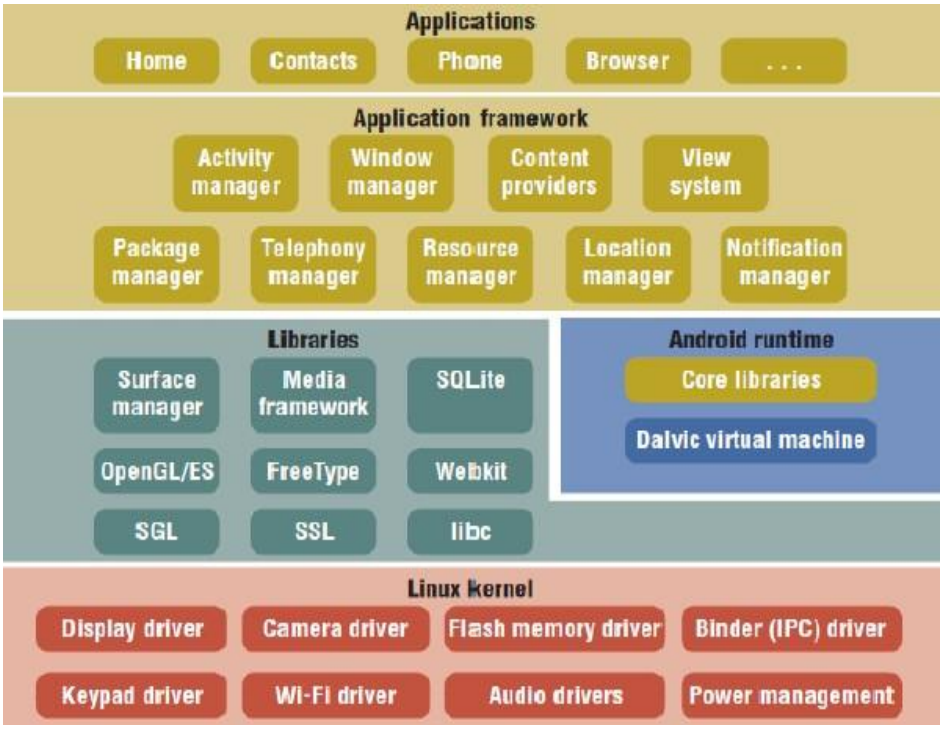
sebagai pratinjau pengembang untuk Nexus 5X, Nexus 6P dan Pixel C. Pratinjau kedua mulai tersedia pada 22 November 2016, sebelum versi final diluncurkan ke publik pada bulan Desember. 5, 2016.

14. *Android* versi 8.0 Oreo (API level 26)

    Android Oreo adalah rilis utama ke 8 dari sistem operasi *Android*. Ini pertama kali dirilis sebagai preview pengembang pada tanggal 21 Maret 2017 untuk perangkat Nexus dan Pixel saat ini. Pratinjau pengembang terakhir dirilis pada tanggal 24 Juli 2017, dengan rilis stabil yang diharapkan pada bulan Agustus atau September 2017.

2.5.2 Arsitektur *Android*

Secara garis besar arsitektur *Android* dapat dijalaskan dan digambarkan sebagai berikut [10]:



Gambar 2.3Arsitektur *Android*

a. Application and Widget

Application dan *widgets* ini adalah layer dimana pengguna berhubungan dengan aplikasi saja, dimana biasanya pengguna download aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut.

b. Application Frameworks

Application framework ini adalah layer dimana para pembuat aplikasi melakukan pengembangan atau pembuatan aplikasi yang akan dijalankan disistem operasi *Android*, karena pada layer inilah aplikasi dapat dirancang dan dibuat, seperti *content providers* yang berupa sms dan panggilan telepon.

c. Libraries

Libraries ini adalah layer dimana fitur – fitur *Android* berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya.
Berajalan diatas kernel, layer ini meliputi berbagai library C/C++ inti seperti Libc dan SSL, dll.

d. *Android* Runtime

Layer yang membuat aplikasi *Android* dapat dijalankan dimana dalam prosesnya menggunakan implementasi Linux. Dalvik virtual machine merupakan mesin yang membentuk dasar kerangka aplikasi *Android*.

e. Linux Kernel

Linux kernel adalah layer dimana inti dari *operating system* dari *Android* itu berada. Berisi *file – file system* yang mengatur sistem *processing*, *memory*, *resource drivers*, dan sistem – sistem operasi *Android* lainnya.

Alasan digunakannya *Android* pada penelitian ini karena aplikasi yang dibangun menggunakan *mobile Android* oleh karena itu perlu adanya pengetahuan tentang *Android* seperti apa saja yang perlu diperhatikan dan disiapkan dalam membangun aplikasi *Android*.

2.6 Application Programing Interface(API)

API adalah singkatan dari *Application Programming Interface*, dan memungkinkan developer untuk mengintegrasikan dua bagian dari aplikasi atau dengan aplikasi yang berbeda secara bersamaan. API terdiri dari berbagai elemen seperti *function*,protocols,dan *tools* lainnya yang memungkinkan *developers* untuk membuat aplikasi. Tujuan penggunaan API adalah untuk mempercepat proses *development* dengan menyediakan *function* secara terpisah sehingga *developer* tidak perlu membuat fitur yang serupa. Penerapan API akan sangaat terasa jika fitur yang diinginkan sudah sangat kompleks, tentu membutuhkan waktu untuk membuat yang serupa dengannya. Misalnya: integrasi dengan payment gateway. Terdapat berbagai jenis sistem API yang dapat digunakan, termasuk sistem operasi, library, dan *web* [11].

API memungkinkan sebuah aplikasi berkomunikasi dengan aplikasi lain di Internet melalui serangkaian panggilan (*call*). Sebuah API, berdasarkan definisinya, adalah sesuatu yang mendefinisikan cara dua entitas untuk berkomunikasi. Entitas di sini adalah sebuah *software* yang nyata berbeda (dalam layanan) dengan *software* lain. 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 sendiri merupakan sekumpulan kode *software* yang ditulis sebagai serangkaian pesan XML. Setiap pesan XML berhubungan dengan fungsi spesifik dari aplikasi yang akan diajak berkomunikasi. Sebagai contoh, pada API *Facebook*, terdapat pesan XML yang berhubungan dengan fungsi spesifik *wall post*, *wall comment*, *wall like*.Pengembang aplikasi pihak ketiga menggunakan pesan-pesan XML yang berhubungan dengan fungsi-fungsi spesifik dari layanan *web* yang akan diajak berkomunikasi. Pengembang bebas memilih fungsi khusus apa saja yang akan diajak berkomunikasi, dan ditampilkan pada aplikasi rancangannya. Sebagai contoh, kita bisa membuat *Facebook client* yang hanya menampilkan *update* status teman-teman kita.

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

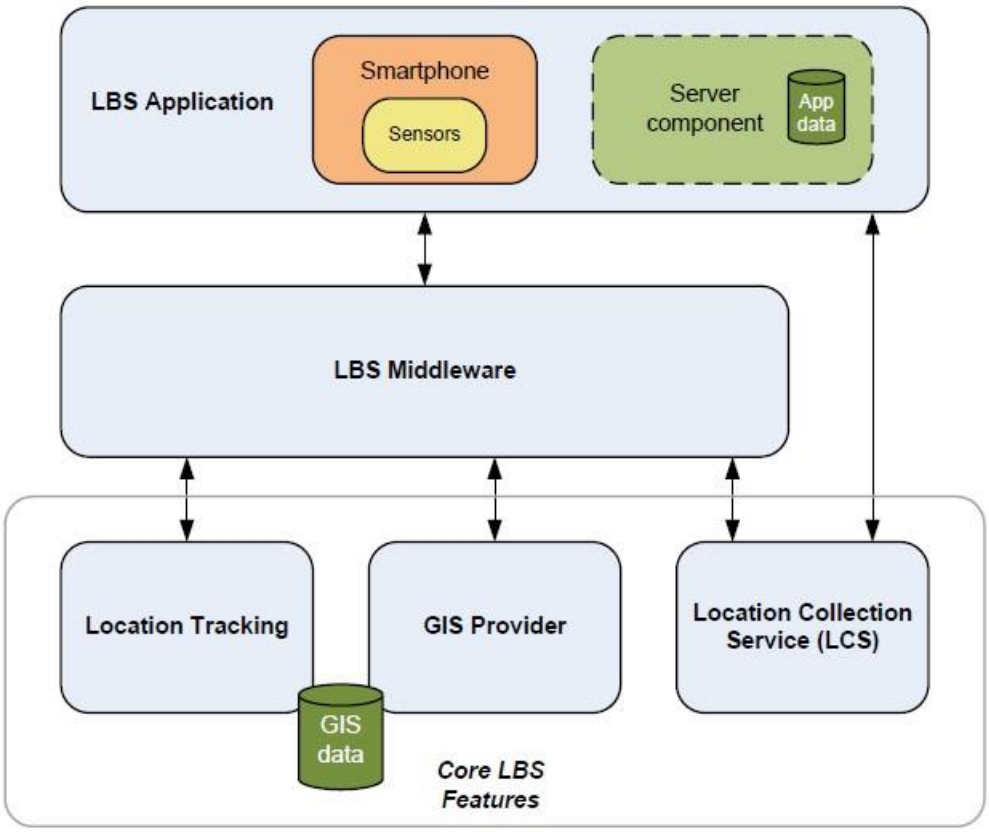
Gambar 2.4 Ilustrasi API (Application Programming Interface)

Sumber :http://www.condepolitan.com

Hubungan API (*Application Programming Interface*) pada penelitian ini yaitu karena aplikasi yang dibangun menggunakan berbagai macam API. Diantaranya: *Google Maps* API dan *Clarifai* API.

2.7 *Location Based Service* (LBS)

Ardiansyah menyatakan "LBS merupakan Layanan IP nirkabel yang menggunakan informasi geografis untuk memberikan layanan informasi lokasi kepada pengguna. Beberapa layanan aplikasi yang memberikan petunjuk posisi/lokasi piranti mobile berada". LBS memberikan layanan personalisasi kepada pengguna perangkat bergerak (*mobile device*) yang disesuaikan dengan lokasi mereka saat ini. LBS adalah layanan informasi yang dapat diakses melalui mobile device dengan menggunakan mobile network, yang dilengkapi kemampuan untuk memanfaatkan lokasi dari mobile device tersebut. LBS memberikan kemungkinan komunikasi dan interaksi dua arah. Oleh karena itu pengguna memberitahu penyedia layanan untuk mendapatkan informasi yang dia butuhkan, dengan referensi posisi pengguna tersebut [12].



Gambar 2.5Arsitektur LBS

Sumber gambar : Amit Kushuwa (2011) [12].

Sistem yang diusulkan akan memberikan fasilitas kepada pengguna ketika pengguna baru datang ke tempat manapun melalui aplikasi. Aplikasi peringatan kedekatan sistem yang diusulkan akan memberikan pemberitahuan kepada pengguna kedatangan tujuan yang ditentukan saat bepergian dalam bentuk alarm. Aplikasi lain dari sistem yang diusulkan adalah jalur terpendek yang akan memberikan layanan kepada pengguna dengan menambang jarak terpendek antara sumber dan tujuan. Aplikasi lain yang bernama pencari teman dari sistem yang diusulkan akan memberikan pemberitahuan kepada pengguna tentang teman-temannya di dekatnya.

Jika dua teman berada di lokasi yang sama, maka yang pertama datang yang dianggap terdekat [13].

Pencarian lengkap dari semua jalur yang mungkin akan menemukan rute terpendek, tetapi secara komputasi sulit untuk semua set kecil dari lokasi. Untuk masalah yang lebih besar, teknik pengoptimalkan diperlukan untuk mencari ruang solusi dan mencari solusi yang mendekati optimal. Secara matematis, masalah perjalanan dapat direpresentasikan sebagai grafik, dimana lokasi adalah node dan ujung (atau busur) mewakili rute langsung antara node. Berat masing-masing ujung adalah jarak antar simpul. Tujuannya adalah menemukan jalur dengan jumlah bobot terpendek. *Google* juga menyediakan cara untuk memecahkan TSP secara sederhana dari lokasi dunia nyata tanpa mengunduh *OR-Tools*. API *Google* memiliki kunci Arah sehingga dapat menyelesaikan TSP lokasi dunia nyata dengan API, *Google* juga menyediakan lokasi di URL dan akan mendapatkan tanggapan kembali sebagai JSON. API *Google* dapat digunakan secara gratis untuk pengembangan atau untuk penggunaan komersial

2.8 *Google* Maps API (Application Programming Interface)

API merupakan software interface yang terdiri atas kumpulan instruksi yang disimpan dalam bentuk library dan menjelaskan bagaimana agar suatu software dapat berinteraksi dengan software lain. Penjelasan ini dapat dicontohkan dengan analogi apabila akan dibangun suatu rumah. Dengan menyewa kontraktor yang dapat menangani bagian yang berbeda, pemilik rumah dapat memberikan tugas yang perlu dilakukan oleh kontraktor tanpa harus mengetahui bagaimana cara kontraktor menyelesaikan pekerjaan tersebut. Dari analogi tersebut, rumah merupakan software yang akan dibuat, dan kontraktor merupakan API yang mengerjakan bagian tertentu dari software tersebut tanpa harus diketahui bagaimana prosedur dalam melakukan pekerjaan tersebut.

Google Map adalah sebuah jasa peta global virtual gratis dan online yang disediakan oleh *Google* dan dapat ditemukan di http://maps.*Google*.com. *Google* Map memberikan layanan untuk menunjukkan jalan-jalan yang ada di seluruh dunia. Bentuknya seperti peta yang umum, tetapi *Google* Map hanya menunjukkan jalan-jalan saja, dan tidak ada nama gunung, sungai, ataupun batas-batas daerah yang biasa ditemukan di peta umum. Walaupun begitu, peta yang ada di *Google* Map ini sangat lengkap. Untuk kota-kota besar, *Google* Map juga menampilkan nama-nama jalan dan gedung. Selain dalam bentuk peta, *Google* Map juga bisa menampilkan dalam bentuk foto satelit [14].

Berikut ini adalah fasilitas dari *Google* Maps :

1. Legenda-legenda (*legend*) adalah keterangan tentang obyek-obyek yang ada di peta, seperti warna hijau adalah hutan, garis merah adalah jalan, simbol buku adalah universitas, dan sebagiannya.
2. Zoom in / out Peta di layar dapat diperbesar dengan zoom in dan diperkecil dengan zoom out.
3. Pan dengan fasilitas pan peta dapat digeser-geser untuk melihat daerah yang dikehendaki.
4. Kemampuan untuk mengubah tampilan dari peta jalan ke tampilan satelit maupun hybrid view dan akan mengkombinasikan tampilan satelit dengan nama jalan dan bangunan.
5. Search Bar yang berguna mencari suatu lokasi secara cepat
6. Mencari arah jalan dengan mudah dan secara otomatis akan ditampilkan rute perjalanannya.
7. Bisa memilih tampilan full screen ketika membuka peta jalan.
8. Bisa menyimpan lokasi-lokasi favorit anda dan menandainya sehingga bisa kembali menemukannya dengan mudah.
9. Bisa dengan mudah mengambil koordinat dari sebuah lokasi yang pilih.

Google maps API adalah fungsi-fungsi pemrograman yang disediakan oleh *Google* maps agar *Google* maps bisa diintegrasikan ke dalam Web atau aplikasi. *Google* Maps API merupakan aplikasi interface yang dapat diakses lewat *javascript* agar *Google* Map dapat ditampilkan pada sebuah halaman web. Pada *Google* Maps JavaScript API versi 2, untuk dapat mengakses *Google* Map pada sebuah halaman web diperlukan *Google* API Key. API Key adalah sederetan kode sebagai izin untuk menampilkan *Google* Map pada sebuah halaman web. Namun untuk versi 3 tidak memerlukan API Key, tetapi pihak *Google* menganjurkan menggunakan API Key untuk mempermudah mengontrol *Google* Maps API. API Key bersifat tunggal, hanya berlaku untuk sebuah URL. Salah satu syarat untuk mendapatkan API Key adalah mempunyai akun *Google*/Gmail untuk men- generate domain atau URL web pada link https://code.*Google*.com/apis/console/ . API Key bersifat gratis sampai batas 25.000 pengunjung per hari. Jika melebihi 25.000 pengunjung per hari maka diperlukan membeli kuota tambahan [14].

2.9 *Clarifai*

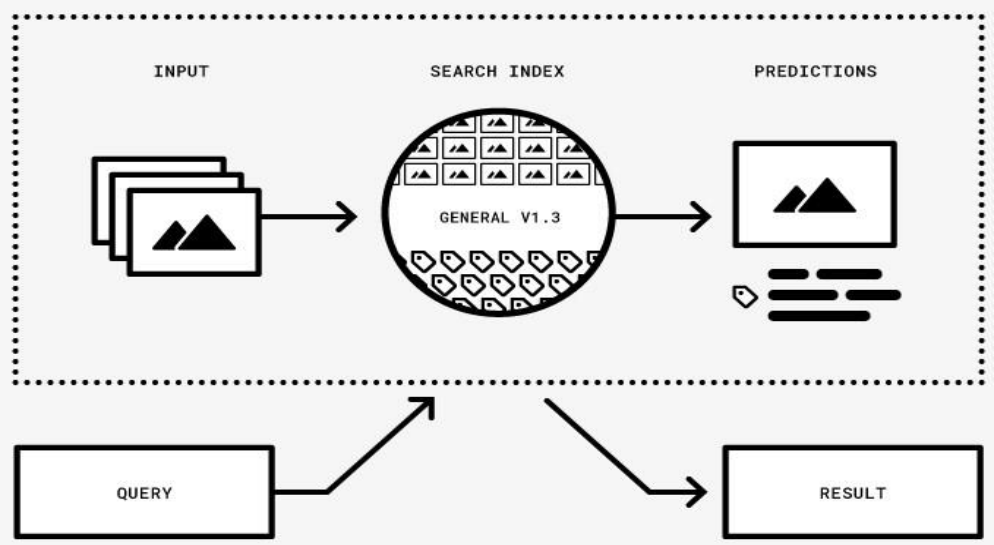
Saat ini, penggunaan alat eksternal atau *remote* yang menyediakan beberapa fungsi atau informasi. Sistem informasi yang mengolah seperti *grid and cloud computing* yang memungkinkan mengakses berbagai layanan dan kemampuan baru yaitu sistem *Clarifai*. Sistem ini menyediakan informasi teknologi untuk menganalisis gambar dan mengindentifikasi anotasi deskriptif yang ada keterkaitannya. *Clarifai* menawarkan *Application Programming Interface* (API) yang memperoleh 20 anotasi paling deskriptif dari gambar yang dikirimkan [5].

Teknologi *Clarifai* bergantung pada penggunaan *Convolutional Neural Networks* (CNN) untuk memproses gambar. CNN didefinisikan sebagai model pembelajaran mesin hierarkis yang mempelajari gambar yang kompleks representasi dari volume besar data beranotasi. Mereka banyak menggunakan lapisan transformasi dasar yang menghasilkan representasi yang sangat canggih. Pendekatan *Clarifai*

pertama kali diusulkan ke ImageNet pada tahun 2013 dimana sistem menghasilkan 5 hasil teratas [5].

Clarifai bekerja melalui analisis gambar untuk menghasilkan deskripsi dari gambar tersebut. Sistem ini juga akan memberikan kemungkinan terbesar dari gambar yang dimaksud. API *Clarifai* dapat diakses melalui website dengan menggunakan gambar sebagai inputannya, CNN akan menganalisa dengan memberikan kemungkinan beberapa jawaban [5].

Kasus umum untuk menggunakan *Clarifai* adalah untuk mendapatkan konsep yang diprediksi dalam sebuah gambar dan kemudian menggunakan konsep tersebut untuk mencari daya. API Pencarian memungkinkan Anda mengirim gambar (url atau byte) ke layanan dan meminta mereka diindeks oleh konsep model umum dan representasi visualnya. Setelah diindeks, dapat mencari gambar berdasarkan konsep atau dengan gambar. Cara kerja *Clarifai* dapat dilihat pada Gambar 2.6 [15]:



Gambar 2.6 Ilustrasi Cara Kerja *Clarifai*

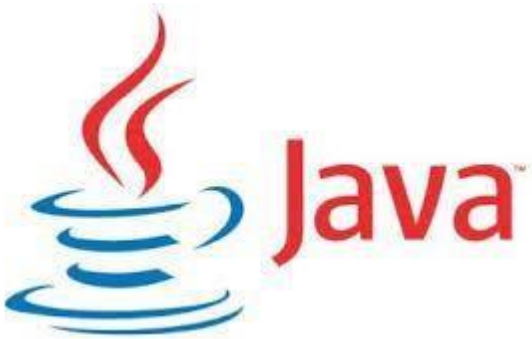
Sumber gambar : www.*Clarifai*.com/developer/guide [15].

Tabel 2.2 Studi Literatur

Judul	Kegunaan
Pembangunan aplikasi survei harga produsen pedesaan hortikultura di badan pusat statistic kabupaten subang dengan API *CLARIFAI*	Pada penggunaan aplikasi survei ini penggunaan *Clarifai* Api untuk mempermudah pencarian lokasi menggunakan gambar.
Pembangunan Aplikasi Smart Assistance monitoring meggunakan Api Carifai dan photo tangging berbasis *Android* pada iro adminitrasi Provinsi Banten	Pada penggunaan di aplikasi ini API *Clarifai* sebagai sarana photo tagging petugas monitoring yang harus melakukan foto secara langsung ke tempat yang akan dilakukan monitoring untuk disampaikan secara langsung ke kepala biro bahwa petugas monitoring telah melakukan pekerjaannya dengan benar
Pembangunan aplikasi intelligent Marketplace property memanfaatkan API *Clarifai* dan Api U*Class*ify berbasis *Android*	Pemanfaatan Api *Clarifai* disini yaitu sebagai fitur yang mempermudah pencarian tempat berdasarkan fotoyang digunakan untuk pencarian rumah.

2.10 Java

Java adalah salah satu bahasa pemrograman yang paling populer. *Java* dapat digunakan untuk berbagai hal, termasuk pengembangan perangkat lunak, aplikasi mobile, dan pengembangan sistem yang besar. Inilah yang membuat bahasa pemrograman *Java* sangat terkenal di lingkungan pengembang perangkat lunak [16].



Gambar 2.7 Logo Java

Sumber: jurnal pemrograman Aplikasi Mobile Smartphone

Seperti bahasa pemrograman lain, bahasa *Java* memiliki struktur sendiri, aturan sintaks, dan paradigma pemrograman. paradigma pemrograman bahasa Java didasarkan pada konsep *OOP*. Bahasa *Java* merupakan turunan bahasa C , sehingga aturan sintaks yang terlihat akan seperti bahasa C. Misalnya, blok kode yang modular dalam metode dan dibatasi oleh karakter '({' dan '})', dan variabel dideklarasikan sebelum digunakan. Secara struktural, bahasa *Java* diatur dengan *package*. Di Dalam *package* ada *Class*, dan dalam *Class* ada *method*, variabel, konstanta, dan banyak lagi [17].

Adapun sintak dalam bahasa pemrograman Java sebagai berikut :

```
public Class HelloWorld
{ 2.1 2.2   2.3
public static void 2.4  main(String[] args) {
```

2.10.1 Sejarah Perkembangan Java

Bahasa pemrograman Java pertama lahir dari *The Green Project*, yang berjalan selama 18 bulan, dari awal tahun 1991 hingga musim panas 1992. Proyek tersebut belum menggunakan versi yang dinamakan Oak. Proyek ini dimotori oleh *Patrick Naughton, Mike Sheridan, James Gosling dan Bill Joy*, beserta sembilan pemrogram

lainnya dari Sun Microsystems. Salah satu hasil proyek ini adalah maskot Duke yang dibuat oleh Joe Palrang. Pertemuan proyek berlangsung di sebuah gedung perkantoran Sand Hill Road di Menlo Park. Sekitar musim panas 1992 proyek ini ditutup dengan menghasilkan sebuah program Java Oak pertama, yang ditujukan sebagai pengendali sebuah peralatan dengan teknologi layar sentuh (touch screen), seperti pada PDA sekarang ini. Teknologi baru ini dinamai "*7" (Star Seven).

Setelah era Star Seven selesai, sebuah anak perusahaan Tv kabel tertarik ditambah beberapa orang dari proyek *The Green Project*. Mereka memusatkan kegiatannya pada sebuah ruangan kantor di 100 Hamilton Avenue, Palo Alto. Perusahaan baru ini bertambah maju, jumlah karyawan meningkat dalam waktu singkat dari 13 menjadi 70 orang. Pada rentang waktu ini juga ditetapkan pemakaian Internet sebagai medium yang menjembatani kerja dan ide di antara mereka. Pada awal tahun 1990-an, Internet masih merupakan rintisan, yang dipakai hanya di kalangan akademisi dan militer. Mereka menjadikan perambah (*browser*) Mosaic sebagai landasan awal untuk membuat perambah Java pertama yang dinamai *Web Runner*, terinsipirasi dari film 1980-an, *Blade Runner*. Pada perkembangan rilis pertama, *Web Runner* berganti nama menjadi *Hot* Java. Pada sekitar bulan Maret 1995, untuk pertama kali kode sumber Java versi 1.0a2 dibuka. Kesuksesan mereka diikuti dengan untuk pemberitaan pertama kali pada surat kabar San Jose Mercury News pada tanggal 23 Mei 1995. Sayang terjadi perpecahan di antara mereka suatu hari pada pukul 04.00 di sebuah ruangan hotel Sheraton Palace. Tiga dari pimpinan utama proyek, Eric Schmidt dan George Paolini dari Sun Microsystems bersama Marc Andreessen, membentuk Netscape. Nama Oak, diambil dari pohon oak yang tumbuh di depan jendela ruangan kerja "bapak java", James Gosling. Nama Oak ini tidak dipakai untuk versi *release* Java karena sebuah perangkat lunak sudah terdaftar dengan merek dagang tersebut, sehingga diambil nama penggantinya menjadi "Java". Nama ini diambil dari kopi murni yang digiling langsung dari biji (kopi tubruk) kesukaan Gosling. Konon kopi ini berasal dari

Pulau Jawa. Jadi nama bahasa pemrograman Java tidak lain berasal dari kata Jawa (bahasa Inggris untuk Jawa adalah Java) [18].

2.11 Adroid studio

Android Studio adalah Lingkungan Pengembangan Terpadu – *Integrated Development Environment* (IDE) untuk pengembangan aplikasi *Android* berdasarkan IntelliJ IDEA

Gambar 2.8 Logo *Android* Studio

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

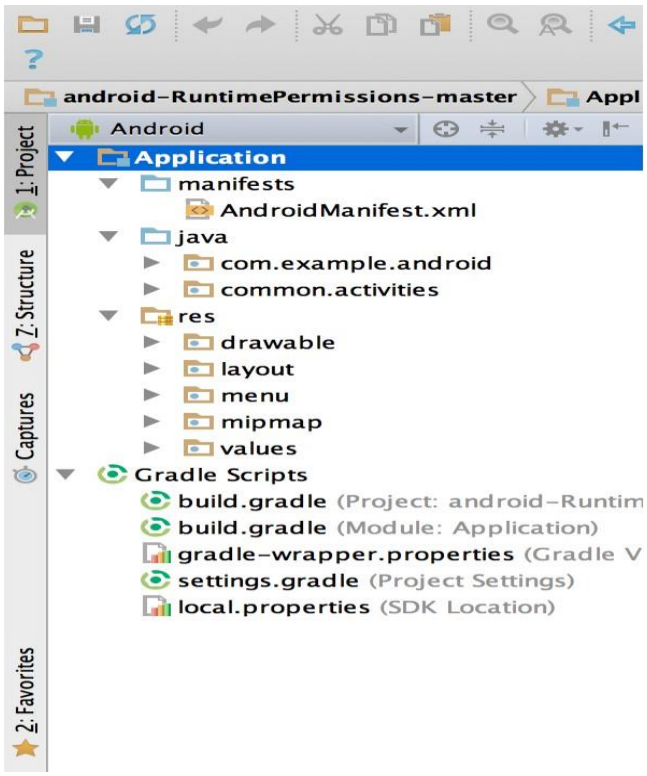
1. Sistem versi berbasis Gradle yang fleksibel.
2. Emulator yang cepat dan kaya fitur.
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat *Android*.
4. *Instant Run* untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru.

5. *Template* kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh.
6. Alat pengujian dan kerangka kerja yang ekstensif.
7. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain.
8. Dukungan C++ dan NDK.
9. Dukungan bawaan untuk *Google Cloud-Platform*, mempermudah pengintegrasian *Google Cloud Messaging* dan *App Engine*.

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

1. Modul aplikasi *Android*
2. Modul Perpustakaan
3. Modul *Google* App Engine [19].

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



Gambar 2.9 Struktur *Project Android* Studio.

Alasan digunakan *Android* Studio pada penelitian karena tools yang digunakan dalam pembangunan aplikasi *Android* yaitu menggunakan IDE *Android* Studio.

Android studio menyediakan alat pengembang *Android* yang terintegrasi dalam pengembangan dan debugging program.

2.12 Pemrograman Berorientasikan Objek

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak. sistem informasi, atau sistem lainnva). Pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objekobjek dunia nvata. Ada banvak cara untuk mengabstraksikan dan memodelkan objekobjek tersebut, mulai dan abstraksi objek, kelas, hubungan antar kelas sampai

abstraksi sistem. Saat mengabstraksikan dan memodelkan objek mi, data dan proses-proses yang dipunyai oleh objek akan dienkapsulasi (dibungkus) menjadi satu kesatuan.

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancangan, pemrograman, dan pengujian perangkat lunak. Ada berbagai teknik yang dapat digunakan pada masing-masing tahap tersebut, dengan aturan dan alat bantu pemodelan tertentu.

Sistem berorientasi objek merupakan sebuah sistem yang kompunennya dibungkus (dienkapsulasi) menjadi kelompok data dan fungsi. Setiap komponen dalam sistem tersebut dapat mewarisi atribut, sifat, dan kompunen lainnya yang dapat berinteraksi satu sama lain [20].

Terdapat beberapa konsep utama pada metodologi berorientasi objek, diantaranya [20]:

1) Kelas (*Class*), kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dari himpunan objek yang sama yang mungkin lahir atau diciptakan dari kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (*relationship*), dan arti. Suatu kelas dapat diturunkan dari kelas yang lain, dimana atribut dari kelas semula dapat diwariskan ke kelas yang baru.
2) Objek (*object*), abstraksi dari sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan.
3) Abstraksi (*abstraction*), prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi suatu bentuk model yang sederhana dengan mengabaikan aspekaspek lain yang tidak sesuai dengan permasalahan.

4) Enkapsulasi (*encapsulation*), pembungkusan atribut data dan layanan (operasioperasi) yang dimiliki objek untuk menyembunyikan implementasi dari objek sehingga objek lain tidak mengetahui cara kerjanya.
5) Pewarisan (*inheritance*), mekanisme yang memungkinkan suatu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dari dirinya.
6) Polimorfism (*polymorphism*), kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program. Hubungan OOP (*Object Oriented Programming*) pada penelitian ini adalah pada penelitian ini menggunakan konsep OOP yaitu membagi-bagi kode program menjadi kelas-kelas yang di buat berdasarkan objek-objek yang terlibat saat pembangunan program.

2.13 Unifiend Modelining Language (UML)

Unified Modeling Language (UML) adalah bahasa yang digunakan untuk menspesifikasikan, memvisualisasikan, dan mendokumentasikan artifak dari sebuah sistem berorientasi objek yang dalam tahap pengembangan. UML menyediakan standar defacto dalam analisis sistem berorientasi objek dan desain yang didasarkan dari pengalaman pengguna [21].

UML menawarkan sebuah standar untuk merancang model sebuah system. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi, dimana aplikasi tersebut dapat berjalan pada piranti keras, *system* operasi dan jaringan apapun, sertra ditulis dalam bahasa pemrograman apapun.

2.13.1 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. Di dalam *Use Case* terdapat aktor. Seorang / sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan

pekerjaanpekerjaan tertentu. Elemen-elemen yang ada pada diagram use case adalah aktor, use case, dan hubungan ketergantungan, generalisasi dan asosiasi [21].

1. Aktor

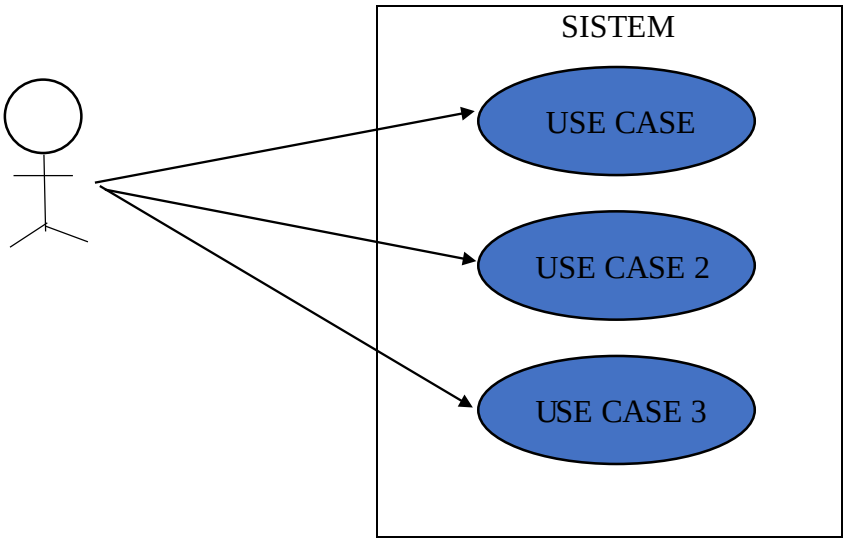
   Aktor adalah pemakai sistem, dapat berupa manusia atau sistem terotomatisasi lain. Aktor adalah sesuatu atau seseorang yang berinteraksi dengan sistem, yaitu siapa atau apa yang menggunakan sistem. Aktor berkomunikasi dengan sistem lewat pengiriman dan penerimaan pesan. Use case selalu diawali oleh aktor yang mengirim pesan.

2. Use Case

   Use case adalah cara spesifikasi penggunaan sistem oleh aktor. *Use case* menspesifikasikan perilaku sistem atau bagian sistem dan merupakan deskripsi sekumpulan sekuen aksi termasuk varian-varian yang dilakukan sistem untuk memproduksi hasil atau nilai ke aktor.

3. Hubungan antar *Use Case*

   Keterhubungan antar *use case* dengan *use case* lain berupa generalisasi antara *use case*, yaitu *include* dan *extend*. *Extend* adalah relasi *use case* tambahan ke sebuah *use case* dimana *use case*yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu. *Include* adalah relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya.



Gambar 2.10 Contoh *Use Case Diagram*

2.13.2 *Class* Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *Class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain [21].

Class memiliki 3 area pokok:

1. Nama memiliki fungsi mempresentasikan identitas dari sebuah *Class*.
2. Atribut memiliki fungsi mempresentasikan atribut-atribut yang ada di dalam suatu *Class*.
3. Metoda memiliki fungsi mempresentasikan metode-metode yang ada pada suatu *Class*.

Atribut dan metoda dapat memiliki salah satu sifat berikut:

1. *Private*, tidak dapat dipanggil dari luar *Class* yang bersangkutan.

2. *Protected*, hanya dapat dipanggil oleh *Class* yang bersangkutan dan anak-anak yang mewarisinya.
3. *Public*, dapat dipanggil oleh siapa saja.

Setiap *Class* pasti memiliki hubungan dengan *Class* lain. Terdapat 4 cara menghubungkan satu *Class* dengan *Class* lain tergantung hubungan *Class* – *Class* tersebut. Hubungan tersebut direpresentasikan dengan sebuah panah diantara *Class* tersebut[11]. Adapun 4 hubungan antar *Class* pada *Class diagram* sebagai berikut:

1. Asosiasi

   Asosiasi adalah hubungan statis antar *Class*. Umumnya menggambarkan clas yang memiliki atribut berupa *Class* lain, atau *Class* yang harus mengetahui eksistensi *Class* lain. Panah navigability menunjukkan arah query antar *Class*.

2. Agregasi

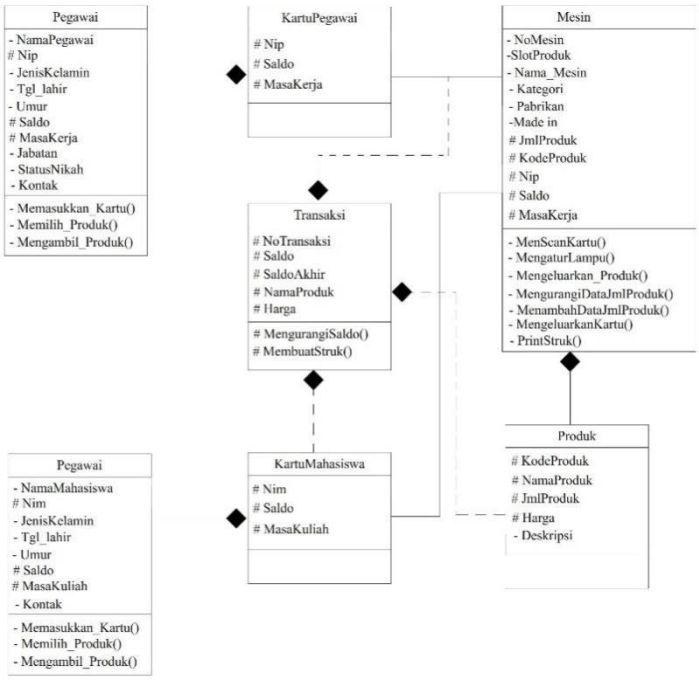
   Agregasi adalah hubungan yang menyatakan bagian “terdiri atas”.

3. Pewarisan

   Pewarisan adalah hubungan hirarkis antar *Class*. *Class* dapat diturunkan dari *Class* lain dan mewarisi semua atribut dan metoda *Class* asalnya dan menambahkan fungsionalitas baru sehingga ia disebut anak dari *Class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.

4. Hubungan dinamis

   Hubungan dinamis adalah rangkaian pesan yang di passing dari satu *Class* kepada *Class* lain. Hubungan dinamsi dapat digambarkan dengan menggunakan sequence diagram.

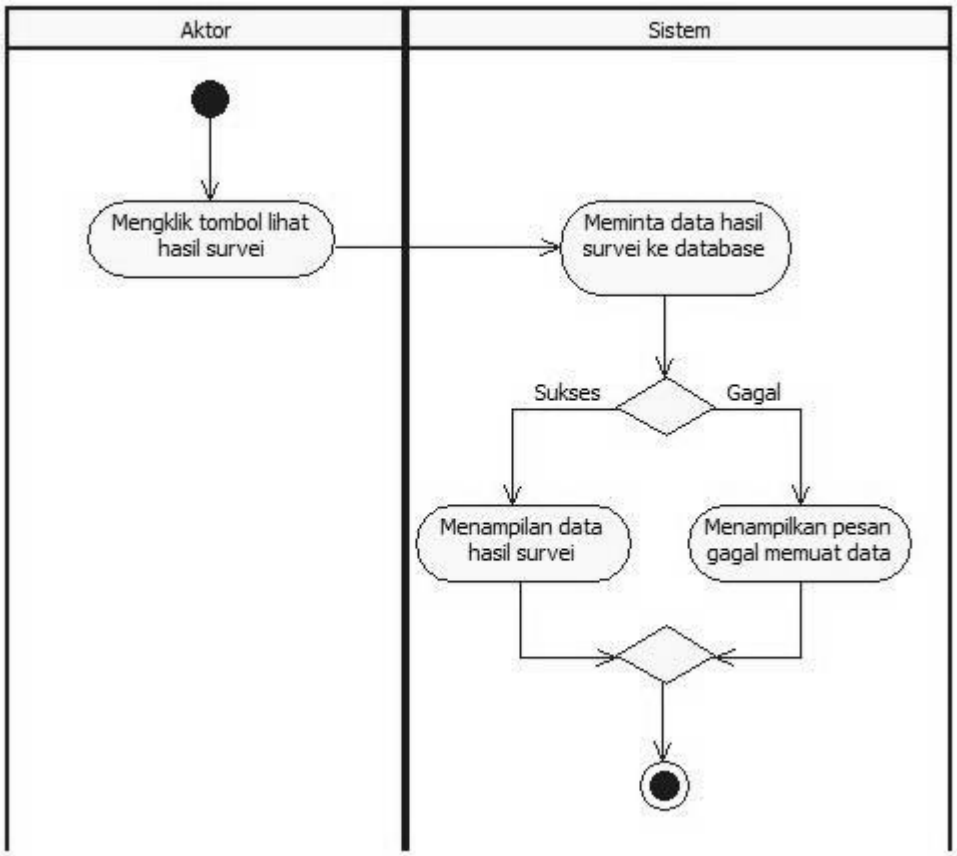


Gambar 2.11 Contoh *Class Diagram*

2.13.3 Activity Diagram

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

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas [21].



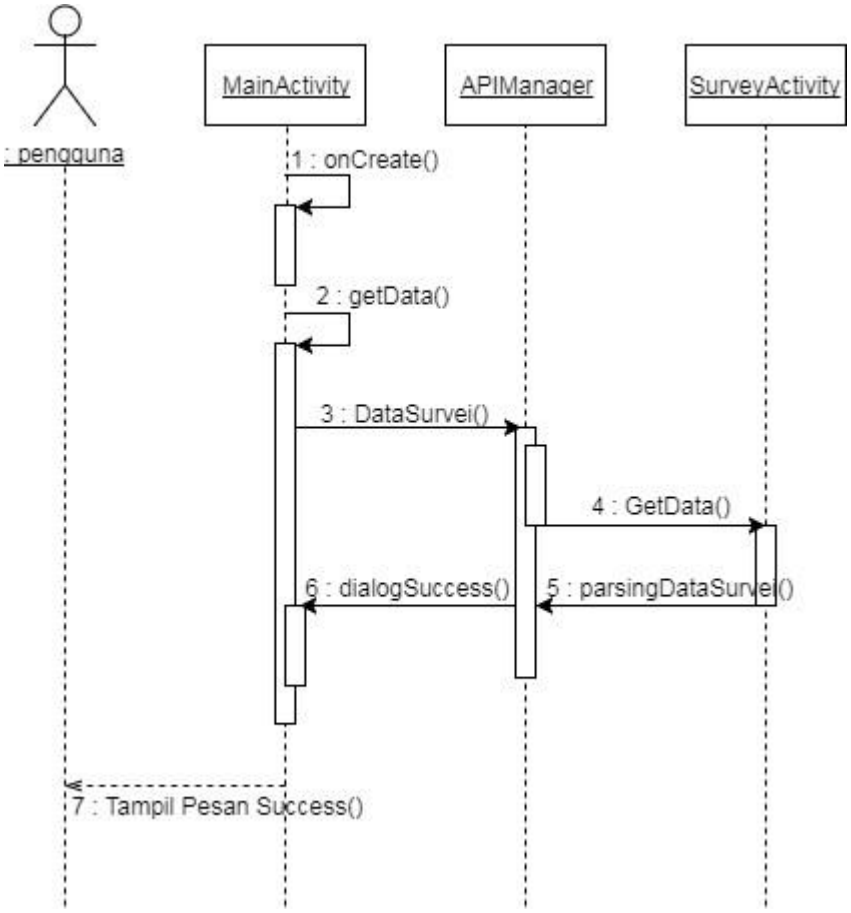
Gambar 2.12 Contoh *Activity Diagram*

2.13.4 Sequence Diagram

Sequence diagram dapat menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message*

digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *Class* [21].



Gambar 2.13 Contoh *Sequence Diagram*

Hubungan UML pada penelitian ini yaitu pada penelitian ini untuk memodelkan suatu aplikasi yang akan dibuat menggunakan konsep berorientasi objek. Sehingga bahasa pemodelan yang digunakan dalam perancangan aplikasi adalah UML

2.14 Internet

Internet merupakan rangkaian jaringan dalam jaringan yang menghubungkan komputer individu yang dimiliki oleh pemerintah, universitas, group dan non profit dan

perusahaan. Interkoneksi ini di hubungkan dengan standar protokol yang bebas dan terbuka (Turban, Rainer dan Potter, 2005, p478).

Internet adalah sistem informasi global berbasis komputer yang terbentuk dari jaringan-jaringan komputer-komputer yang saling terkoneksi satu sama lain (Nugroho, 2006, p25).

Secara harfiah, Internet (kependekan dari *interconnected-networking*) ialah sistem global dari seluruh jaringan komputer yang saling terhubung menggunakan str Internet Protocol Suite (TCP/IP) untuk melayani miliaran pengguna di seluruh dunia. Manakala Internet (huruf 'I' besar) ialah sistem komputer umum, yang berhubung secara global dan menggunakan TCP/IP sebagai protokol pertukaran paket (*packet switching communication protocol*). Rangkaian internet yang terbesar dinamakan Internet. Cara menghubungkan rangkaian dengan kaedah ini dinamakan *internetworking* [22].

Jadi, internet adalah jaringan komputer yang menghubungkan komputer yang berada di seluruh dunia yang dapat digunakan untuk menyampaikan informasi dan berkomunikasi

2.15 JSON (Javascript Object Natation)

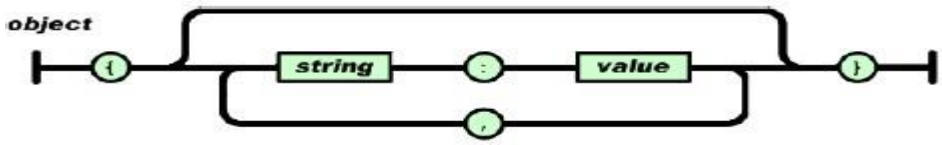
JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data. JSON terbuat dari dua struktur [23]:

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 modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. JSON menggunakan bentuk sebagai berikut :

1. Objek

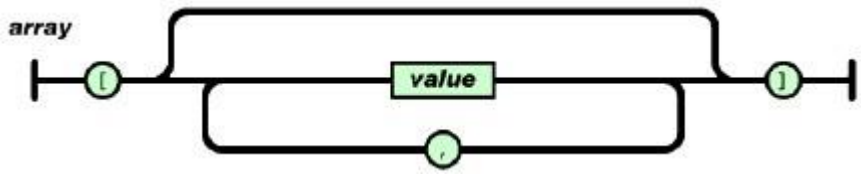
   Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan **{** (kurung kurawal buka) dan diakhiri dengan **}** (kurung kurawal tutup). Setiap nama diikuti dengan **:** (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh **,** (koma).



Gambar 2.14 Alur Bentuk Objek

2. Array

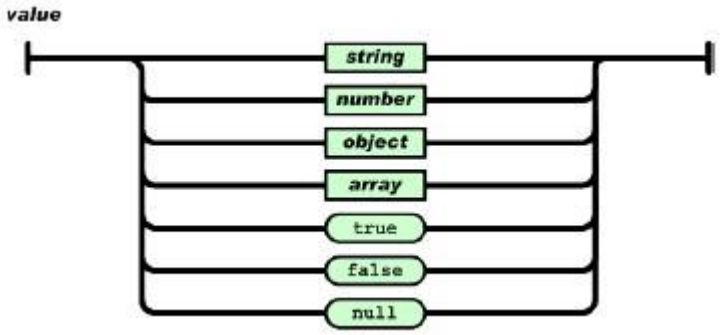
   Array adalah kumpulan nilai yang terurutkan. Larik dimulai dengan **[** (kurung kotak buka) dan diakhiri dengan **]** (kurung kotak tutup). Setiap nilai dipisahkan oleh **,** (koma).



Gambar 2.15 Alur Bentuk Array

3. Value

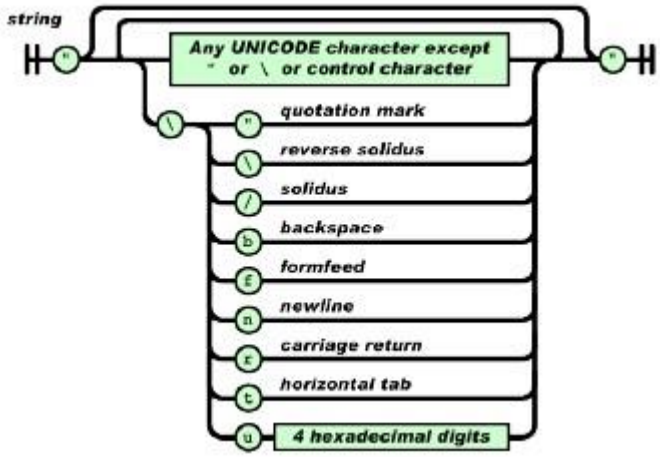
Value dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau **true** *atau* ***false*** atau ***null***, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.



Gambar 2.16 Alur Bentuk *Value*

4. String

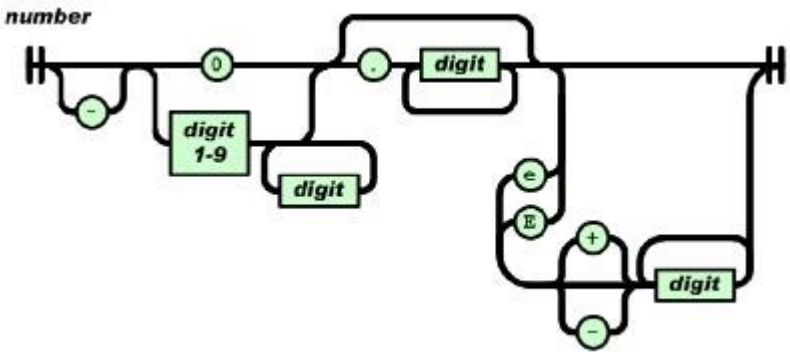
String adalah kumpulan dari nol atau lebih karakter Unicode, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan *backslash escapes* "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java.



Gambar 2.17 Alur Bentuk *String*

5. Number

Number (angka) sangat mirip dengan angka di C atau Java, kecuali format oktal dan heksadesimal tidak digunakan.



Gambar 2.18 Alur Bentuk *Number*

2.16 React Native

React Native adalah kerangka kerja JavaScript untuk menulis aplikasi mobile yang asli dan asli untuk iOS dan *Android*. Ini didasarkan pada React, perpustakaan JavaScript *Facebook* untuk membangun antarmuka pengguna, tetapi alih-alih menargetkan browser, ia menargetkan platform seluler. Dengan kata lain: pengembang web sekarang dapat menulis aplikasi seluler yang terlihat dan terasa benar-benar "asli",

semua dari kenyamanan perpustakaan JavaScript yang sudah kita kenal dan sukai. Plus, karena sebagian besar kode yang Anda tulis dapat dibagikan di antara berbagai platform, React Native memudahkan pengembangan secara bersamaan untuk *Android* dan iOS.

Mirip dengan React untuk Web, aplikasi React Native ditulis menggunakan campuran JavaScript dan markup XML-esque, yang dikenal sebagai JSX. Kemudian, di bawah tenda, React Native "bridge" memanggil API rendering asli di Objective-C (untuk iOS) atau Java (untuk *Android*). Dengan demikian, aplikasi Anda akan dirender menggunakan komponen UI seluler nyata, bukan tampilan web, dan akan terlihat dan terasa seperti aplikasi seluler lainnya. React Native juga memaparkan antarmuka JavaScript untuk platform API, sehingga aplikasi React Native Anda dapat mengakses fitur platform seperti kamera ponsel, atau lokasi pengguna.

React Native saat ini mendukung iOS dan *Android*, dan memiliki potensi untuk berkembang ke platform masa depan juga. Di buku ini, kami akan membahas iOS dan *Android*. Sebagian besar kode yang kita tulis adalah cross-platform. Dan ya: Anda benar-benar dapat menggunakan React Native untuk membangun aplikasi seluler yang siap produksi! Beberapa anekdota: *Facebook*, Palantir, dan TaskRabbit sudah menggunakannya dalam produksi untuk aplikasi yang menghadap pengguna [24].

2.17 XAMPP

XAMPP adalah distribusi Apache kecil dan ringan yang berisi pengembangan web paling umum teknologi dalam satu paket. Isinya, ukurannya yang kecil, dan mudah dibawa menjadikannya alat yang ideal untuk siswa mengembangkan dan menguji aplikasi dalam PHP dan MySQL. XAMPP tersedia secara gratis unduh dalam dua paket spesifik: lengkap dan ringan

Gambar 2.19 Logo *Android* Xampp

Sementara unduhan paket lengkap menyediakan lebar berbagai alat pengembangan, artikel ini akan fokus pada penggunaan XAMPP Lite yang berisi yang diperlukan teknologi yang memenuhi standar Kompetisi Keterampilan Ontario. Seperti namanya, versi ringan adalah paket kecil yang berisi Apache HTTP Server, PHP, MySQL, phpMyAdmin, Openssl, dan SQLite [25].

2.18 Pembangunan

Pembangunan secara umum pada hakekatnya adalah proses perubahan yang terus menerus untuk menuju keadaan yang lebih baik berdasarkan norma-norma tertentu. Mengenai pengertian pembangunan, para ahli memberikan definisi yang bermacammacam seperti halnya perencanaan. Istilah pembangunan bisa saja diartikan berbeda oleh satu orang dengan orang lain, daerah yang satu dengan daerah lainnya, Negara satu dengan Negara lain. Namun secara umum ada suatu kesepakatan bahwa pembangunan merupakan proses untuk melakukan perubahan [26].

Pengertian tentang pembangunan sebagai “Suatu usaha atau rangkaian usaha pertumbuhan dan perubahan yang berencana dan dilakukan secara sadar oleh suatu bangsa, negara dan pemerintah, menuju modernitas dalam rangka pembinaan bangsa (nation building)” [27]. Sedangkan dalam pengertian ekonomi murni, pembangunan adalah suatu usaha proses yang menyebabkan pendapatan perkapita masyarakat meningkat dalam jangka Panjang. Dengan demikian, proses pembangunan terjadi di

semua aspek kehidupan masyarakat, ekonomi, sosial, budaya, politik, yang berlangsung pada level makro (nasional) dan mikro. Makna penting dari pembangunan adalah adanya kemajuan/perbaikan (progress), pertumbuhan dan diversifikasi [28].

Sebagaimana dikemukakan oleh para para ahli di atas, pembangunan adalah semua proses perubahan yang dilakukan melalui upaya-upaya secara sadar dan terencana [29]