BAB 2
TINJAUAN PUSTAKA

2.1 Jurnalis

Istilah jurnalis dari kata jurnalistik yang berasal dari bahasa Perancis: "*Journal*", berarti catatan harian. Jurnalistik berkaitan dengan catatan harian yang dipublikasikan kepada masyarakat dan terbit secara teratur (YS. Gunadi, 1998: 64). "*Journal*" atau "*Dejour*" berarti hari, dimana segala berita atau warta sehari itu termuatdalam lembaran yang tercetak. Dalam perkembangannya istilahjurnalistik disenadakan dengan pers atau jurnalis (Dja'far Assegaf,1991: 10).

Jurnalis atau wartawan adalah seseorang yang bekerja dalam bidang jurnalisme atau peliputan sebuah objek artikel tertentu,misalnya sebuah berita atau kejadian, sebuah tempat, sebuah kondisi alam atau lingkungan, profil seseorang, tanggapan seseorang, atau pembahasan-pembahasan yang lain.

Hasil liputan oleh seorang jurnalis ini nantinya *dipublish* (disampaikan ke masyarakat luas) melalui media massa yang bersangkutan, baik media massa elektronik, maupun media massa cetak.

1) Media massa elektronik, contohnya: televisi, radio, internet.
2) Media massa cetak, contohnya: koran, majalah, tabloid.

Secara umum proses kerja jurnalis terdiri dari dua tahap, yaitu peliputan dan reportase, penulisan berita dan editing, namun sebelumnya perlu dibahas sedikit tentang apa itu berita, sebab berita adalah inti dari kegiatan jurnalistik yang akan dikerjakan oleh jurnalis, bahkan lebih dari 90% isi media cetak adalah berita yang disediakan oleh jurnalis. Meskipun untuk memberikan pengertian mengenai berita tidaklah mudah, tetaplah perlu untuk mencoba mencari pengertian yang tepat, pengertian dan batasan mengenai berita merupakan bekal bagi jurnalis karena dengan pengertian yang dimiliki tentang berita akan sangat menentukan tingkat kemampuan dan profesionalitas kewartawanan dalam menilai sesuatu sebagai berita[2].

2.2 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[3].

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

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

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

2.3 Android

Gambar 2.1 Logo Android

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

2.3.1 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 [4]:

a. 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.

b. 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.

c. 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.*

d. 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.

e. 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.*

f. 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.

g. 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 *prosessormulticore*dangrafisdengan*hardwareacceleration.*

UserInterface 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.

h. 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.

i. 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.

j. 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.

k. 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.

l. 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.

m. 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

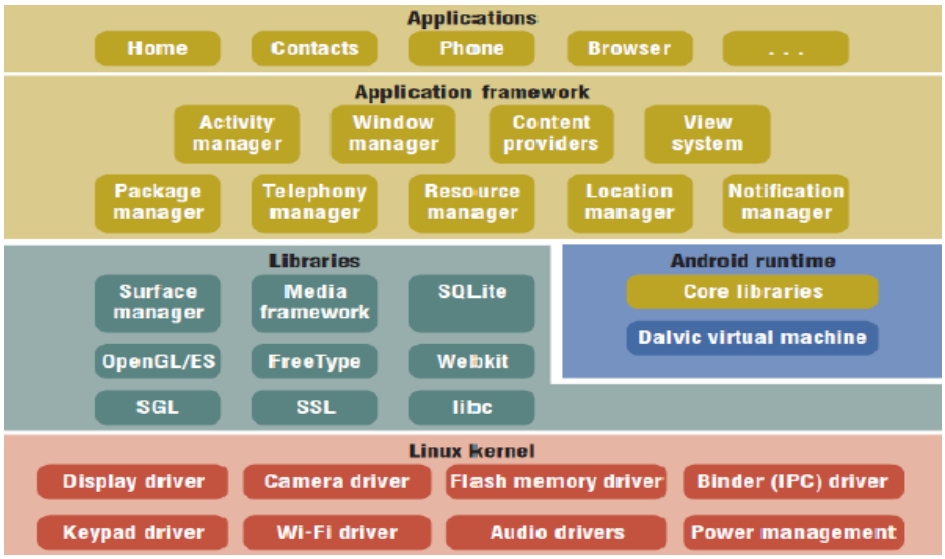
dengan "Program Beta" baru yang memungkinkan perangkat yang didukung ditingkatkan versinya ke versi Android Nougat melalui over-the-air update. Rilis terakhir adalah pada tanggal 22 Agustus 2016. Pratinjau akhir pembuatannya dirilis pada tanggal 18 Juli 2016, dengan nomorNPD90G.Padatanggal 19 Oktober 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.

n. 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.

2.3.2 Arsitektur Android

Secara garis besar arsitektur android dapat dijalaskan dan digambarkan sebagai berikut[5]:



Gambar 2.2 Arsitektur 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 *contentproviders* 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

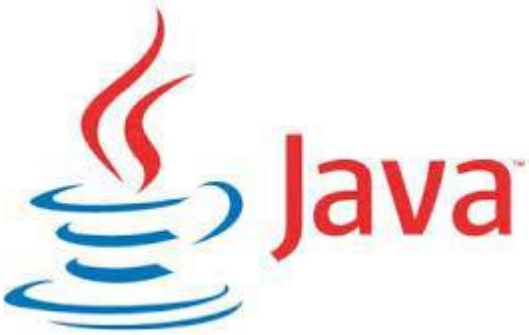
Layer yang membuat aplikasi android dapat dijalankan dimana dalam prosesnya menggunakan implementasi Linux. Dalvik virtual machine merupakan mesin yang membentuk dasar kerangka aplikasi android.

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.

2.4 JAVA



Gambar 2.3 Logo 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[6].

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[7].

Adapun sintak dalam bahasa pemrograman Java sebagai berikut :

```
public class HelloWorld {

      public static void main(String[] args) {
            System.out.println("Hello, World");
      }
}
```

Hubungan Java pada penelitian ini karena aplikasi yang dibuat menggunakan tools Android studio dengan bahasa pemrograman yang digunakan

yaitu bahasa pemrograman Java. Android Studio dipilih karena memiliki banyak fitur yang memudahkan saat pembangunan aplikasi android.

2.4.1 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).

Setelahera 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)[8]

2.5 Android Studio

Gambar 2.4 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:

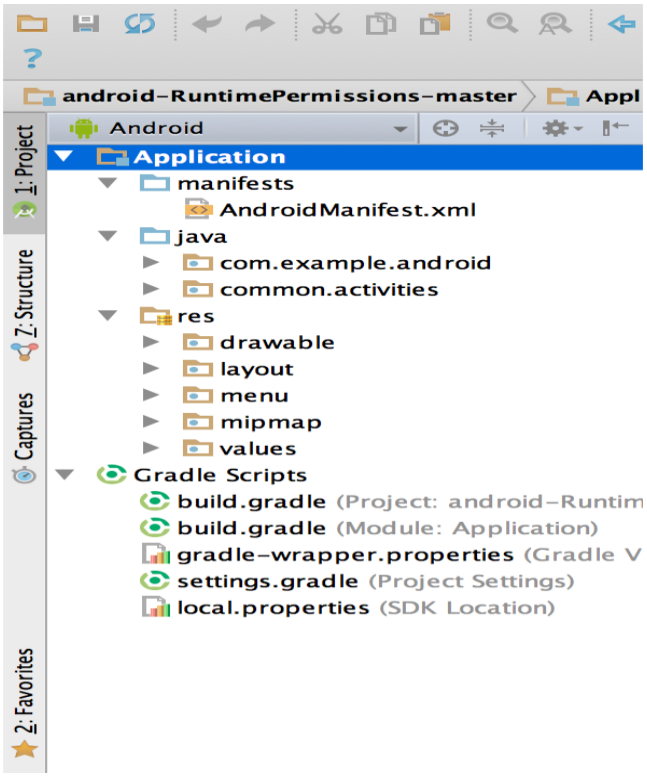
1. Sistem versi berbasis Gradle yang fleksibel.
2. Emulator yang cepat dan kaya fitur.
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat android.
4. *Instant Run* untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru.

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*[9]

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



Gambar 2.5 Stuktur *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.

2.6 Pemrograman Berorientasi 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 objek-objek dunia nvata. Ada banvak cara untuk mengabstraksikan dan memodelkan objek-objek 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[10].

Terdapat beberapa konsep utama pada metodologi berorientasi objek, diantaranya[10]:

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 aspek-aspek lain yang tidak sesuai dengan permasalahan.
4) Enkapsulasi (*encapsulation*), pembungkusan atribut data dan layanan (operasi-operasi) 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.

2.7 *Unified Modelling 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[11].

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.

2.7.1 *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 pekerjaan-pekerjaan tertentu. Elemen-elemen yang ada pada diagram use case adalah aktor, use case, dan hubungan ketergantungan, generalisasi dan asosiasi[11].

a. Aktor

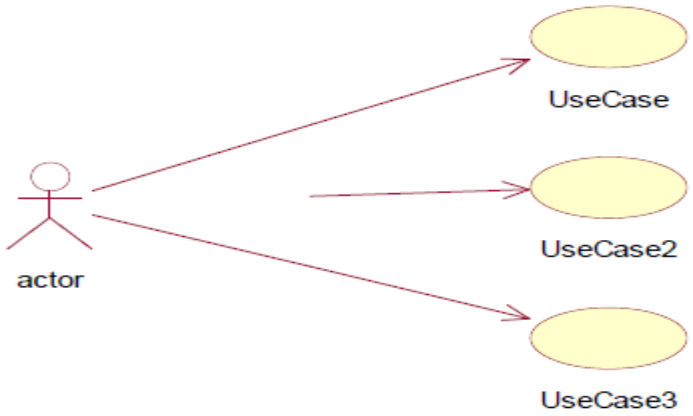
Aktor adalah pemakai sistem, dapat berupa manusia atau sistem terotomatisasi lain. Aktor adalah sesuatu atau seseorang yang berinteraksi dengan sistem, yaitu siapa atau apa yang menggunakan sistem. Aktor berkomunikasi dengan sistem lewat pengiriman dan penerimaan pesan. Use case selalu diawali oleh aktor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

b. 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.

c. 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 2.6 Contoh *Use Case Diagram*

2.7.2 *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[11].

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

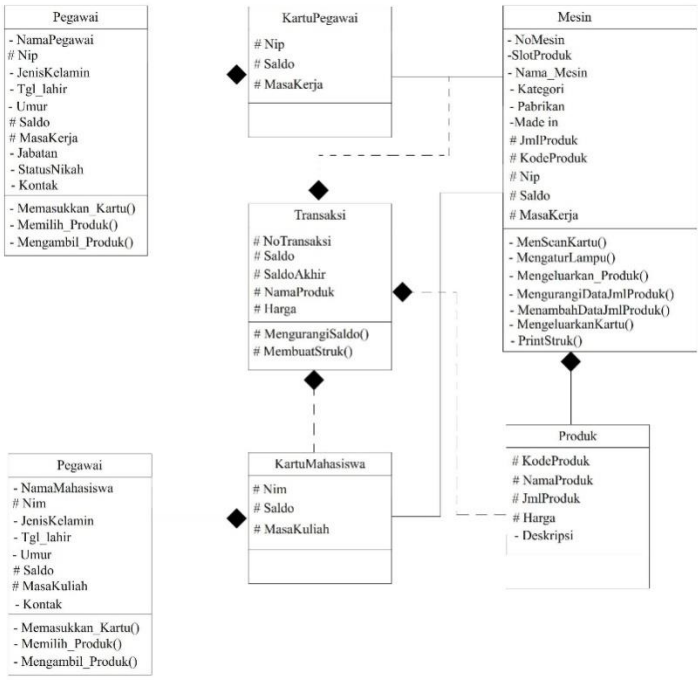
   Agregasi adalah hubungan yang menyatakan bagian “terdiri atas”.
3. Pewarisan

   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.

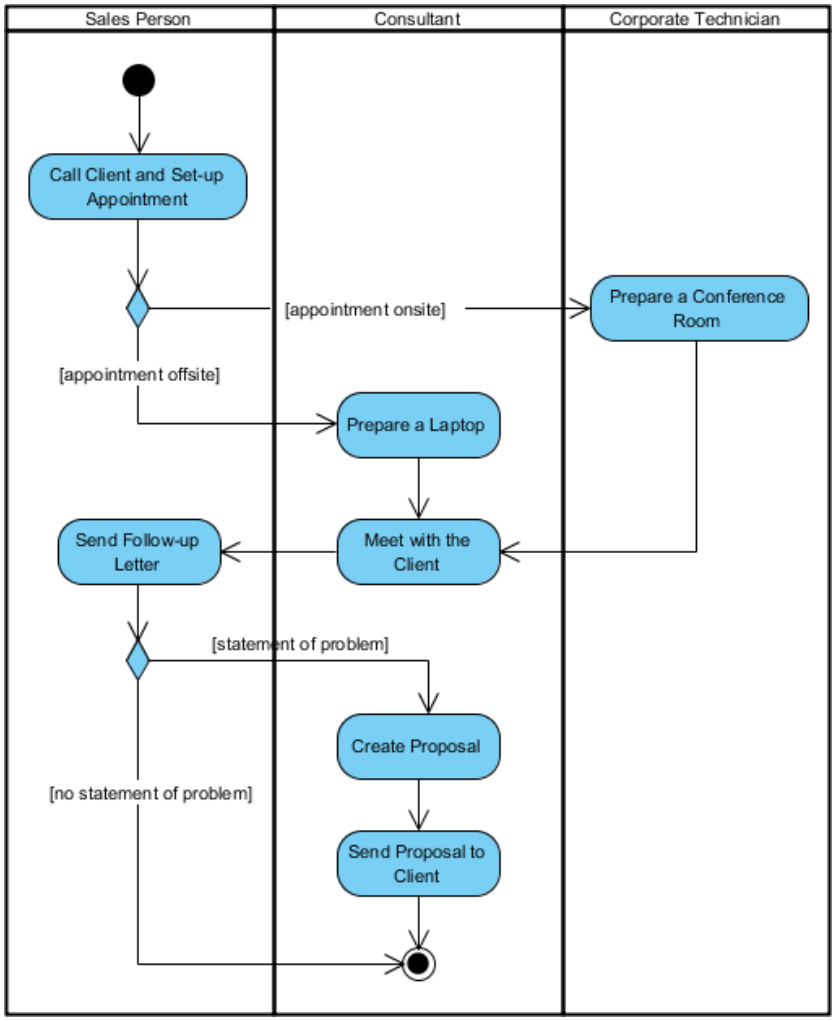


Gambar 2.7 Contoh *Class Diagram*

2.7.3 *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[11].



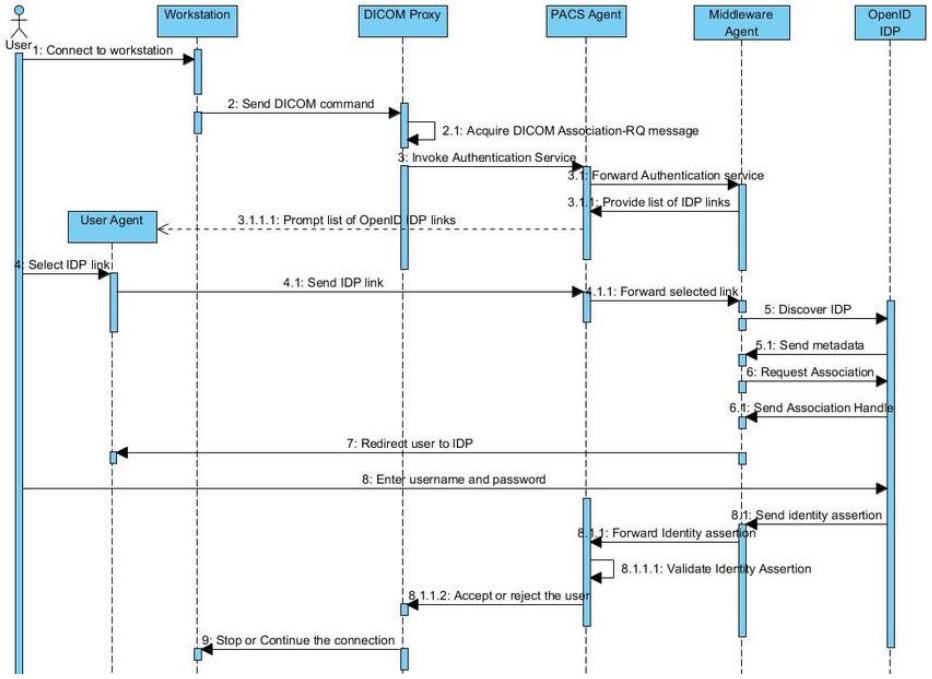
Gambar 2.8 Contoh *Activity Diagram*

2.7.4 *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*[11].



Gambar 2.9 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.

2.8 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*[12].

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

2.9 JSON (*Javascript Object Notation*)

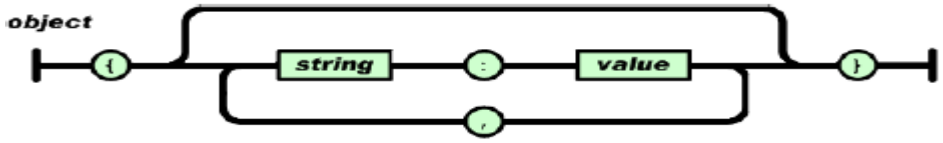
JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum 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[13] :

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

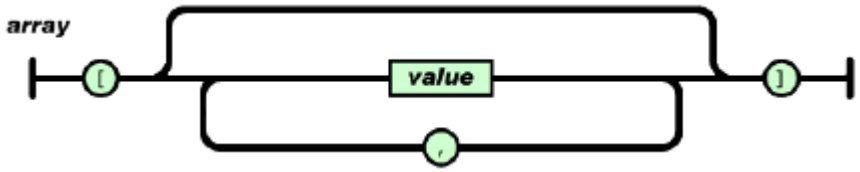
   Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan **{** (kurung kurawal buka) dan diakhiri dengan **}** (kurung kurawal tutup). Setiap nama diikuti dengan **:** (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh **,** (koma).



Gambar 2.10 Alur Bentuk Objek

2. Array

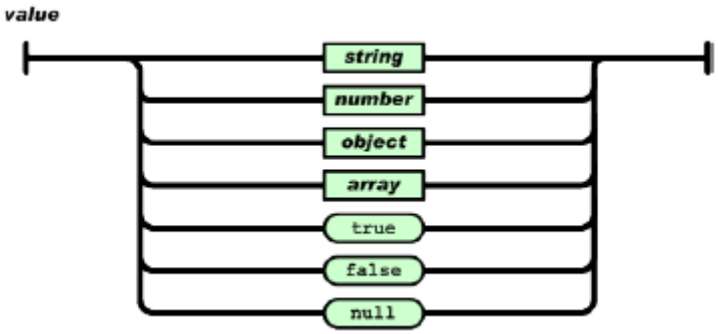
   Array adalah kumpulan nilai yang terurutkan. Larik dimulai dengan **[** (kurung kotak buka) dan diakhiri dengan **]** (kurung kotak tutup). Setiap nilai dipisahkan oleh **,** (koma).



Gambar 2.11 Alur Bentuk Array

3. Value

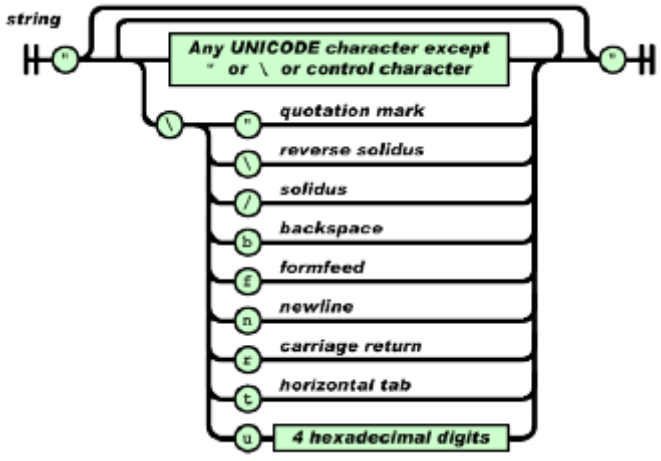
   Value dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau **true** *atau* ***false*** atau ***null***, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.



Gambar 2.12 Alur Bentuk *Value*

4. String

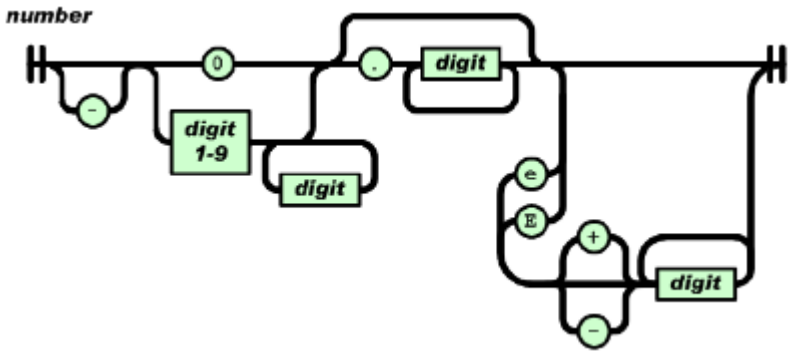
   String adalah kumpulan dari nol atau lebih karakter Unicode, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan *backslash escapes* "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java.



Gambar 2.13 Alur Bentuk *String*

5. Number

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



Gambar 2.14 Alur Bentuk *Number*

2.10 *Application Programming 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*[14].

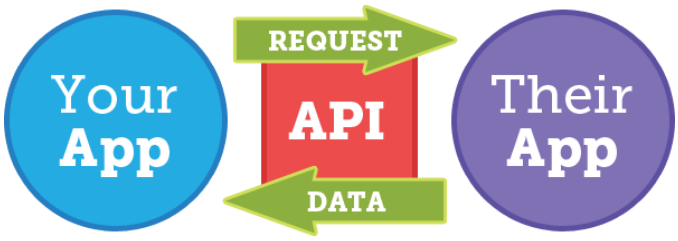
API memungkinkan sebuah aplikasi berkomunikasi dengan aplikasi lain di Internet melalui serangkaian panggilan (*call*). Sebuah API, berdasarkan definisinya, adalah sesuatu yang mendefinisikan cara dua entitas untuk berkomunikasi. Entitas di sini adalah sebuah *software* yang nyata berbeda (dalam layanan) 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 dimanfaatkan oleh pengembang pihak ketiga untuk mendesain aplikasi yang memanfaatkan layanan mereka dengan mudah.



Gambar 2.15 Ilustrasi API (*Application Programming Interface*)

Hubungan API (*Application Programming Interface*) pada penelitian ini yaitu karena aplikasi yang dibangun menggunakan berbagai macam API. Diantaranya: Google *Cloud Speech* API, Google *Cloud Text to Speech* API, Google *Translate* API dan Alchemy API.

2.10.1 Google *Translate* API

Google *Translate* adalah salah satu *translate* bahasa *online* paling terkenal dan paling banyak digunakan orang di seluruh dunia saat ini, banyak manfaat dari jasa Google *Translate* untuk menterjemahkan bahasa ke bahasa lain dengan mudah tanpa harus membuka kamus atau lainnya.

Google *Translate* merupakan layanan aplikasi yang disediakan oleh Google.Inc , yang berfungsi untuk membantu menterjemahkan suatu teks atau halaman *web* dari suatu bahasa ke bahasa lain. Menurut Wikipedians (2001), Google melakukan penerjemahkan dengan pendekatan yang disebut penerjemahan berdasar statistik. Penerjemahan demikian merupakan hasil penelitian Franz-Josef Och yang telah memenangkan kontes DARPA untuk kecepatan mesin terjemahan pada tahun 2003.

Menurut Michael Miller (2009), Google *translate* adalah lebih akurat dibanding dengan yang lain. Sedangkan kekurangannya adalah Google *Translate* menggunakan *translate* berbasis *word by word translation* yang tidak selalu memperhatikan stuktur bahasa. Sehingga membuat kalimat yang diterjemahkan terkadang menjadi sedikit kacau[15].

Hubungan google *translate* API pada penelitian ini yaitu karena aplikasi yang dibangun memanfaatkan API dari Google *Translate* untuk menerjemahkan teks ke bahasa tujuan agar mempermudah jurnalis ketika mewawancarai narasumber asing.

2.10.2 Google *Cloud Speech* API

Google *cloud speech* API diluncrukan pada tahun 2008 di Amerika Serikat untuk beberapa tipe *smartphone*. Google *cloud speech* API adalah sebuah *framework* yang dikembangkan oleh Google untuk mengenali suara, mengubahnya menjadi string (teks) dan memasukkanya ke dalam halaman pencarian Google sehingga akan tampil hasil pencarian berdasarkan input suara.

Dengan kata lain input suara yang diterima oleh perangkat Android (*smartphone*) akan dikirimkan ke *server* Google, yang selanjutnya *server* Google melakukan pengenalan dan mengubahnya menjadi teks. Hasil konversi suara menjadi teks kemudian dimasukkan dalam halaman pencarian Google kemudian *server* Google akan mengirimkan hasil pencarianya tersebut ke perangkat Android (Reddy & Mahender, 2013)[16]

Hubungan google *cloud speech* API pada penelitian ini yaitu karena aplikasi yang dibangun memanfaatkan API dari Google *Cloud Speech* untuk mentranslasi *file* suara menjadi teks. Agar jurnalis tidak perlu mendengarkan berulang-ulang *file* audio rekaman wawancara untuk dijadikan transkrip.

2.10.3 Google *Cloud Text to Speech* API

Google Cloud Text-to-Speech memungkinkan pengembang untuk menyintesis ucapan yang terdengar alami tersedia dengan 30 suara dalam berbagai bahasa. Dengan API yang mudah digunakan ini dapat membuat interaksi langsung dengan pengguna di banyak aplikasi dan perangkat.

Google Cloud Text-to-Speech mendukung aplikasi atau perangkat apa pun yang dapat mengirim permintaan REST atau gRPC termasuk ponsel, PC, tablet, dan perangkat IoT. Sebagai API yang mudah digunakan, Google Cloud Text-to-Speech adalah solusi fleksibel untuk mengubah teks menjadi audio yang dapat dikonsumsi sebagai audio[17]

Hubungan Google *Cloud Text to Speech* API pada penelitian ini yaitu karena aplikasi yang dibangun memanfaatkan API dari Google *Cloud Text to Speech* yang nantinya digunakan untuk melafalkan teks yang sudah di terjemahkan oleh Google *Translate*.

2.10.4 Alchemy API

Layanan *Cloud* IBM Alchemy API memungkinkan Klien untuk membuat aplikasi pintar yang menjalankan analisis konten pada teks dan gambar. Dengan menggunakan algoritma pengolahan bahasa alamiah, pengenalan gambar, dan mesin pembelajaran, Klien dapat memperkaya dan mengekstraksi meta-data semantik dari konten, seperti informasi mengenai orang, tempat, perusahaan, topik, fakta, hubungan, penulis, dan bahasa. Alchemy API juga digunakan untuk mengklasifikasikan konten dari sebuah teks, atau untuk melihat topik apa yang sedang tren dalam berita[18] .

Layanan *Cloud* ini menyediakan serangkaian *endpoint* API untuk melakukan analisis konten pada halaman *web* yang dapat diakses oleh Internet,

HTML yang dikirimkan atau kontek teks. Fitur-fitur yang disertakan dalam Layanan *Cloud* ini adalah[18] :

a. Layanan IBM Alchemy *Language* memberikan 12 fungsi API untuk analisis teks, yang masing-masing dari fungsi tersebut menggunakan teknik pengolahan bahasa alamiah yang canggih untuk menganalisis konten dan menambah informasi semantik tingkat tinggi. Klien dapat menggunakan API untuk menjalankan salah satu dari tugas berikut: ekstraksi entitas, analisis sentimen, ekstraksi kata kunci, penandaan konsep, ekstraksi relasi, klasifikasi taksonomi, ekstraksi penulis, deteksi bahasa, ekstraksi teks, penguraian kalimat dengan mikroformat, *feed detection*, dan *linked data support*.
b. Layanan IBM Alchemy Vision memberikan fungsi-fungsi untuk menganalisis adegan visual yang kompleks secara keseluruhan, tanpa memerlukan petunjuk tekstual, yang meningkatkan pendekatan menyeluruh untuk memahami beberapa objek dan area sekitar dalam foto *smartphone* dan gambar *online* yang umum. Klien dapat menggunakan API untuk menjalankan salah satu dari tugas berikut: *image recognition*, ekstraksi *image link*, dan *scene recognition*.
c. Layanan IBM Alchemy *Data News* memungkinkan Klien untuk meminta sumber berita dan blog seperti basis data. Klien dapat menggunakan API untuk meminta puluhan dari ribuan sumber-sumber berita dan konten blog.

Hubungan Alchemy API pada penelitian ini yaitu karena aplikasi yang dibangun memanfaatkan API dari alchemy yang digunakan untuk menganalisa teks hasil wawancara.

2.11 Metode Pengujian Sistem

Metode pengujian sistem terdiri dari Pengujian *black box* dan beta yang dilakukan untuk mengetahui efektifitas dari perangkat lunak (*software*) yang digunakan.

2.11.1 Pengujian *Black Box*

Pengujian *black box* berfokus pada spesifikasi fungsional dari perangkat lunak. Metode *black box* dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. Pengujian *black box* dapat menemukan kesalahan dalam kategori berikut[19] :

1. Fungsi yang tidak benar atau hilang.
2. Kesalahan antarmuka (*interface error*).
3. Kesalahan pada struktur data dan akses basis data.
4. Kesalahan performa (*performance error*).
5. Kesalahan inisialisasi dan terminasi.
6. Validitas fungsional.
7. Kesensitifan sistem pada nilai input tertentu.
8. Batasan dari suatu data.

2.11.2 Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif dimana diuji secara langsung ke lapangan, dengan menggunakan kuesioner mengenai tanggapan pengguna terhadap aplikasi yang telah dibangun. Adapun metode penilaian pengujian yang digunakan yaitu metode kuantitatif berdasarkan data sampel dari pengguna.

2.12 Pengujian Skala Likert (Kuesioner)

Kuesioner merupakan teknik pengumpulan data yang dilakukan dengan cara memberi beberapa pertanyaan atau pernyataan kepada responden untuk mendapatkan jawaban dari pertanyaan atau pernyataan tersebut. Untuk setiap satu pernyataan responden memilih salah satu tanggapan berupa SS = Sangat Setuju, S

= Setuju, Ragu=R, TS = Tidak Setuju, STS = Sangat Tidak Setuju. Tanggapan responden dikodekan kedalam bilangan 1 sampai dengan 5. Pemberian bilangan ini bergantung kepada bentuk pernyataan yang dijawab oleh responden.

Ada dua macam bentuk pernyataan dalam skala likert, pertama adalah pernyataan yang diharapkan untuk disetujui oleh responden, disebut bentuk positif. Yang kedua adalah pernyataan yang diharapkan untuk tidak disetujui oleh responden, bentuk negatif. Biasanya kuesioner berisikan pernyataan positif dan negatif agar responden berfikir dulu sebelum memberi jawaban.

Kemudian penulis olah kedalam bentuk *kuantitatif*, yaitu dengan cara menetapkan skor jawaban dari pernyataan dan pertanyaan yang telah dijawab oleh responden, dimana pemberian skor tersebut didasarkan pada ketentuan Sugiyono (2009:135)[20].

Tabel 2.1 Skala Likert

Pilihan Jawaban	Pernyataan	
	Positif	Negatif
Sangat Setuju	5	1
Setuju	4	2
Ragu	3	3
Tidak Setuju	2	4
Sangat Tidak Setuju	1	5

Pada bentuk positif sangat setuju memperoleh skor tinggi dan sangat tidak setuju memperoleh skor rendah. Pada bentuk negatif, sangat setuju memperoleh skor terendah dan sangat tidak setuju memperoleh skor tinggi.

Untuk mencari nilai presentase dari masing-masing jawaban kuesioner digunakan rumus skala likert sebagai berikut :

$$P = \frac{total\ nilai}{skor\ ideal}\ x\ 100$$

Berikut ini adalah keterangan dari rumus skala likert mencari nilai presentase dari jawaban kuesioner :

P : nilai persentase yang dicari

Total Nilai : jumlah frekuensi dikalikan dengan skor yang ditetapkan jawaban

Skor Ideal : nilai tertinggi dikalikan dengan jumlah sampel. Nilai tertingginya yaitu 5.

Skala Likert digunakan untuk mengukur pendapat jurnalis terhadap aplikasi *mobile* android yang dibangun. Skala Likert ini akan digunakan pada Bab 4 implementasi dan pengujian.

2.13 Format FLAC

Nama FLAC adalah singkatan dari *Free Lossless Audio Compression*. FLAC dirancang secara khusus untuk kompresi audio dan juga mendukung *streaming* dan arsip data audio. FLAC adalah gagasan dari Josh Coalson yang dikembangkan pada tahun 1999 yang kemudian ia memulai proyek FLAC nya di situs yang terkenal yaitu *Sourceforge* dengan merilis implementasi dari format FLAC tersebut. Sejak itu, banyak pengembang memberikan kontribusi nya untuk meningkatkan implementasi dari FLAC ini. Format FLAC memanfaatkan tingginya korelasi antar sample pada data audio. FLAC menggunakan prediksi linear untuk mengkonversi *sample* menjadi deretan angka yang kemudian disebut residu. FLAC menghasilkan rasio kompresi sebesar 50% hingga 60%[21]

Hubungannya dengan penelitian ini karena aplikasi yang dibangun menggunakan data masukkan *file* audio dengan format .flac.