BAB 2
TINJAUAN PUSTAKA

2.1. Profil Dinas

Profile Dinas Tenaga Kerja di Kabupaten Tasikmalaya meliputi sejarah, visi dan misi, logo, struktur organisasi.

2.1.1. Sejarah Dinas

Sebagai perwujudan dari pelaksanaan otonomi daerah terbentuk Dinas Tenaga Kerja Kabupaten Tasikmalaya, yang merupakan Departemen Tenaga kerja dalam bidang ketenagakerjaan yang memberikan informasi tentang pencari kerja dan lowongan pekerjaan.

Dinas Tenaga Kerja Kabupaten Tasikmalaya dibentuk berdasarkan peraturan Daerah Kabupaten Tasikmalaya Nomor 7 Tahun 2016 tentang Pembentukan dan Susunan Perangkat Daerah. Serta untuk melaksanakan fungsi dan tugasnya telah di atur dalam Peraturan Daerah tersebut, kemudian dijabarkan oleh Peraturan Bupati Tasikmalaya Nomor 69 Tahun 2016 tentang Tugas Pokok, Fungsi dan Rincian Tugas Unit Dinas Tenaga Kerja Kabupaten Tasikmalaya.

Peraturan Pemerintah Nomor 38 Tahun 2007 tentang Pembagian Urusan Pemerintahan antara Pemerintah, Pemerintahan Daerah Provinsi dan Pemerintahan Daerah Kabupaten/Kota serta Peraturan Pemerintah Nomor 41 Tahun 2007 tentang Organisasi Perangkat Daerah, pembentukan, nomenklatur, kedudukan, tugas pokok, fungsi, susunan oraganisasi dan tata kerja Dinas Daerah Kabupaten Tasikmalaya ditinjau kembali (dirubah).

Kemudian ditetapkan peraturan Daerah KabupatenTasik Nomor 60 Tahun 2016 tentang Tugas Pokok, Fungsi, Rincian, Tugas Unit dan Tata kerja Dinas Tenaga Kerja Kabupaten Tasikmalaya yang bentujuan untuk membantu penyerapan tenaga kerja dan mendukung kesejahtraan masyarakat Kabupaten Tasikmalaya. Adapun lokasi kantor dinas ini yang terletak Jl. Mayor Utarya No. 1 telp./Fax. (0265) 333154.

2.1.2. Visi Dinas Tenaga Kerja

Visi Merupakan perencanaan yang strategis yang paling penting dalam suatu Instansi Dinas. Visi Dinas Tenaga Kerja Kabupaten Tasikmalaya yaitu

"Terwujudnya Penyelenggara Ketenagakerjaan Yang Berdaya Saing, Mandiri Dan Sejahtera".

2.1.3. Misi Dinas Tenaga Kerja

Dalam upaya mewujudkan Visi Dinas Tenaga Kerja Kabupaten Tasikmalaya dilaksanakan misi sebagai berikut :

1. Meningkatkan Pengelolaan Dan Perencanaan Pembangunan, Ketenagakerjaan Dan Transmigrasi Yang Akurat, Dinamis Dan Berkesinambungan Berbasis Teknologi Informasi.
2. Meningkatkan Kualitas Sdrn Pengelola Dan Pelaku Ketenagakerjaan Dan Transmigrasi.
3. Meningkatkan Kualitas, Produktivitas Dan Daya Saing Ketenagakerjaan Dan Transmigrasi Di Bidang Agribisnis.
4. Meningkatkan Kualitas Pelayanan Dan lejaring Ketenagakerjaan Dan Transmigrasi.

2.1.4. Logo Dinas

Dinas Tenaga Kerja Kabupaten Tasikmalaya memiliki logo sebagai berikut:

Gambar 2. 1 Logo Dinas Tenaga Kerja

Logo diatas memiliki arti sebagai berikut :

1. Perisai bersudut lima berwarna putih menunjukan sifat gotong royong yang berintikan Pancasila, melambangkan kepribadian, adat istiadat, kepercayaan dan kebudayaan rakyat daerah, sejak dulu sekarang dan kemudian.
2. Gunung melukiskan Gunung Galunggung berwarna biru yang melambangkan ciri tasikmalaya.
3. Simbol Industri melambangkan sebagian dari sumber penghidupan rakyat beserta kekayaan alam di daerah Kabupaten Tasikmalaya.
4. Tiga Buah Sungai melambangkan pemberi sumber kehidupan rakyat.
5. Sawah berwarna hijau terdiri dari 17 petak, melambangkan kesuburan/kemakmuran rakyat yang diproklamasikan pada tanggal 17 Agustus 1945.
6. Sawah berwarna kuning melambangkan sebagian penghidupan rakyat yang didapat dari kerajinan tangan.
7. Bambu Runcing terbuat dari haur kuning melambangkan sejarah perjuangan rakyat daerah Tasikmalaya dalam mengusir kaum penjajah.
8. Pita Kuning Melambai bertuliskan "Sukapura Ngadaun Ngora" melambangkan kemajuan yang abadi.
9. Warna Putih Mengkilat melambangkan tekad suci, warna hitam berarti kekal dan abadi, warna kuning melambangkan keadaan yang gilang gemilang (keemasan), warna hijaumelambangkan kehidupan yang tertinggi, adil dan subur makmur sedangkan warna biru berarti kesetiaan dan kejujuran.

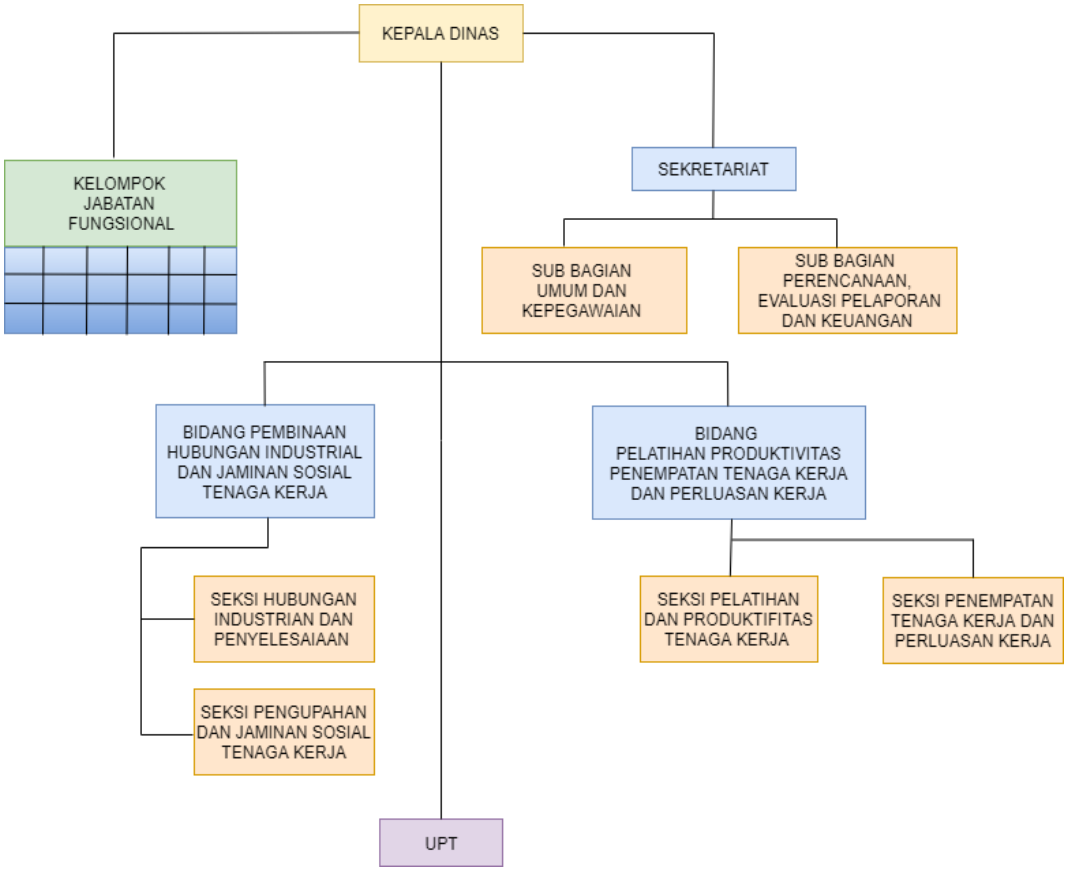
2.1.5. Struktur Organisasi

Struktur Organisasi adalah suatu susunan dan hubungan antara setiap bagian dan juga posisi yang ada pada suatu perusahaan dalam menjalankan suatu kegiatan operasional untuk mencapai suatu tujuan. Struktur Organisasi juga menggambarkan dengan jelas pemisahan suatu kegiatan pekerjaan antara yang satu dengan yang

lainnya. Dalam struktur organisasi harus ada yang berwenang siapa yang melapor kepada siapa, ada empat elemen yang ada dalam struktur organisasi yaitu :

1. Adanya spesialisasi kerja
2. Adanya standarisasi kegiatan kerja
3. Terbentuknya koordinasi dalam kegiatan kerja
4. Seluruh organisasi memiliki tanggung jawab yang sama

Struktur organisasi juga dapat membantu manajemen suatu perusahaan dalam mencapai tujuannya. Struktur organisasi juga menjelaskan bagaimana tugas akan dibagi dan dikelompokan dan dikoordinasikan. Dalam struktur organisasi di Dinas Tenaga Kerja terdapat Kepala Dinas yang bertanggung jawab untuk memimpin Dinas Tenaga Kerja di bantu oleh sekretaris dan kepala bidang pada tiap bagian.



Sumber : Tasikmalayakab, "Database Kelembagaan," Struktur Organisasi Dinas, 2016. [Online]. Available: http://klb.tasikmalayakab.go.id/2016.php. [Accessed 11 Oktober 2018]. [7].

Gambar 2. 2 Struktur Organisasi Dinas Tenaga Kerja

2.2. Landasan Teori

Landasa Landasan teori ini menjelaskan mengenai teori-teori atau penjelasan singkat yang terkait dengan pembangunan aplikasi. Adapun landasan teori pembangunan aplikasi pelayanan Kartu Kuning AK.1 ini adalah sebagai berikut.

2.2.1. Aplikasi

Aplikasi adalah penggunaan dalam suatu komputer, instruksi (*instructiom*) atau pernyataan (*statement*) yang disusun sedemikian rupa sehingga komputer dapat memproses *input* menjadi *output*. Menurut Kamus Kamus Besar Bahasa Indonesia, Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu. Aplikasi adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna. Aplikasi merupakan rangkaian kegiatan atau perintah untuk dieksekusi oleh komputer.

Program merupakan kumpulan *instruction set* yang akan dijalankan oleh pemroses, yaitu berupa *software*. Bagaimana sebuah sistem komputer berpikir diatur oleh program ini. Program inilah yang mengendalikan semua aktifitas yang ada pada pemproses. Program berisi konstruksi logika yang dibuat oleh manusia, dan sudah diterjemahkan ke dalam bahasa mesin sesuai dengan format yang ada pada *instructionset*. Program aplikasi merupakan program siap pakai yang dirancang untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain. Contoh aplikasi ialah program pemproses kata dan *web browser*. Aplikasi akan menggunakan sistem operasi (OS) komputer dan aplikasi yang lainnya yang mendukung. Istilah ini mulai perlahan masuk ke dalam istilah Teknologi Informasi semenjak tahun 1993, yang biasanya juga disingkat dengan *app*.

Dari definisi di atas dapat disimpulkan bahwa aplikasi adalah program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas tertentu untuk pengguna. Aplikasi yang akan dibangun dengan *platform mobile* dan sistem operasinya adalah *android,* dikarenakan saat ini *android* memiliki *market share* sebesar 87,7% dari seluruh *smart phone* yang diaktifkan [8]. Berdasarkan dari uraian masalah yang dipaparkan diatas, maka aplikasi menjadi sebuah media yang akan dibangun dengan tujuan untuk memenuhi kebutuhan pelayanan kartu kuning.

2.2.2. *Android*

Android adalah sistem operasi berbasis *Linux* yang dirancang untuk perangkat seluler layar sentuh seperti *smart phone* dan komputer tablet. Antarmuka pengguna *android* didasarkan pada manipulasi langsung. *Android* adalah sistem operasi dengan sumber terbuka (*open Source*), maka dari itu memungkinkan perangkat lunak untuk di modifikasi secara bebas dan di distribusikan oleh para pembuat perangkat, *operator nirkabel*, dan pengembang aplikasi.

Menurut Ableson, "*Android* merupakan sistem operasi *open source* milik *google* dan terbuka untuk pasar", *android* merupakan sistem operasi ponsel pintar yang paling banyak digunakan berdasarkan data diri *StatCounter Global* Status. Dalam penggunaannya *Android* ditunjang oleh aplikasi yang ada didalamnya untuk memaksimalkan kinerjanya [9]. Sistem operasi *Android* memiliki 2 jalur distribusi, jalur distribusi pertama didukung oleh *Google* atau *Google Mail Service* (GMS), jalur kedua OS menggunakan pola distribusi secara bebas tanpadukungan langsung *Google,* yang dikenal sebagai *Open Handset Distribution* (OHD) [10].

Dari uraian diatas sistem oprasi *android* digunakan untuk membangun suatu aplikasi pelayanan kartu kuning AK.1. *Android* yang digunakan adalah *Android* versi 6.0.1 (*Marshmallow*). Karna dengan banyaknya kelebihan dan fungsi yang bisa digunakan agar dapat maksimal dan sesuai dalam pembuatan sistem pelayanan kartu kuning AK.1.

2.2.2.1. Android SDK (*Software Development Kit*)

Android SDK adalah *tools* API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java*. *Android* merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di-*release* oleh *Google*. Saat ini disediakan *Android SDK* (*Software Development Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java* [10]. mempermudah dalam setiap pemprosesan program serta membantu *recontruction* sistem untuk menghasilkan suatu apk atau bisa disebut runing program yang utuh dan sudah siap unutuk digunakan.

2.2.2.2. *Arsitektur Android*

Secara Garis besar arsitektur *android* dapat dijalankan dan digambarkan sebagai berikut :

1. *Application dan Widgets*

   adalah *layer* dimana *user* berhubungan dengan aplikasi saja, dimana biasanya *user* men-*download* aplikasi, melakukan instalasi dan menjalankan aplikasi.

2. *Application Frameworks*

   Android adalah "*Open Development Platform*" yaitu *Android* menawarkan kepada pengembang atau member kemampuan untuk membangun aplikasi yang inovatif. Pegembang bebas untu mengakses perangkat keras, akses *informasi resource*, menjalankan *service background*, mengatur alarm, dan menambahkan status notifikasi. Komponen-komponen yang termasuk di dalam *applications frameorks* adalah sebagai berikut :

   a. *Views*
   b. *Content provider*
   c. *Resource manager*
   d. *Notification manager*
   e. *Activity manager*

3. *Libraries*

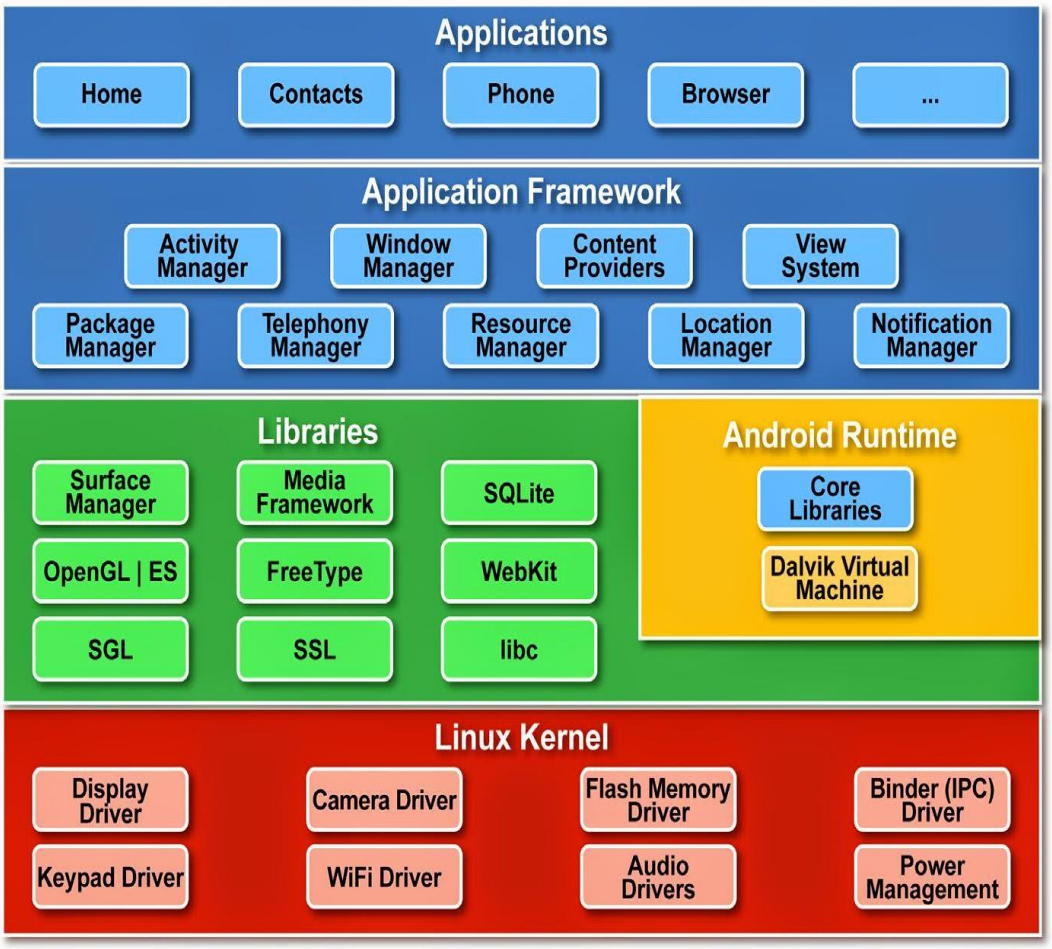
   Libraries adalah *layer* dimana fitur-fitur *Android* berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya.

4. *Android Runtime*

   Layer yang membuat aplikasi *Android* dapat djalankan dimana dalam prosesnya menggunakan Implementasi *Linux*. *Dalvik virtual machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi *Android*.

5. *Linux Kernel*

   Linux Kernel adalah *layer* dimana inti dari *operating system* dari *Android* itu berada. Berisi file-file *system* yang mengatur sistem *processing*, *memory*, *recource*, *drivers*, dan sistem-sistem operasi *android* lainnya.



Sumber : H. N. SAFAAT, PEMOGRAMAN APLIKASI MOBILE SMARTPHONE DAN TABLET PC BERBASIS ANDROID, Bandung: Informatika, 2015. [3].

Gambar 2.3 Struktur Organisasi Dinas Tenaga Kerja

2.2.2.3. *Fundamental Android*

Aplikasi *Android* ditulis dalam bahasa pemrograman *Java*. Kode *Java* dikompilasi bersama dengan data *file resource* yang dibutuhkan oleh aplikasi, dimana prosesnya di-*package* oleh *tools* yang dinamakan "*apt tools*" ke dalam paket *Android* sehingga menghasilkan file dengan *ekstensi apk*. *File apk* itulah yang disebut dengan apikasi. Ada empat jenis komponen pada aplikasi *Android* yaitu [11]:

1. *Activities*

   Suatu *activity* akan menyajikan *user interface* (UI) kepada pengguna, sehingga pengguna dapat melakukan interasksi. Satu *activity* biasanya

akan dipakai untuk meampilkan aplikasi atau yang bertindak sebagai *user interface* saat aplikasi diperlihatkan kepada pengguna.

2. *Service*

   Service tidak memiliki *Graphic User interface* (GUI), tetapi *service* berjalan secara *background*.

3. *Broadcast receiver*

   berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. *Broadcast receiver* tidak memiliki *user infterface* (UI), tetapi memiliki sebuah *activity* untuk merespon informasi yang diterima.

4. *Content provider*

   Content provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file system seperti *SQLite*. *Content provider* menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *activity*.

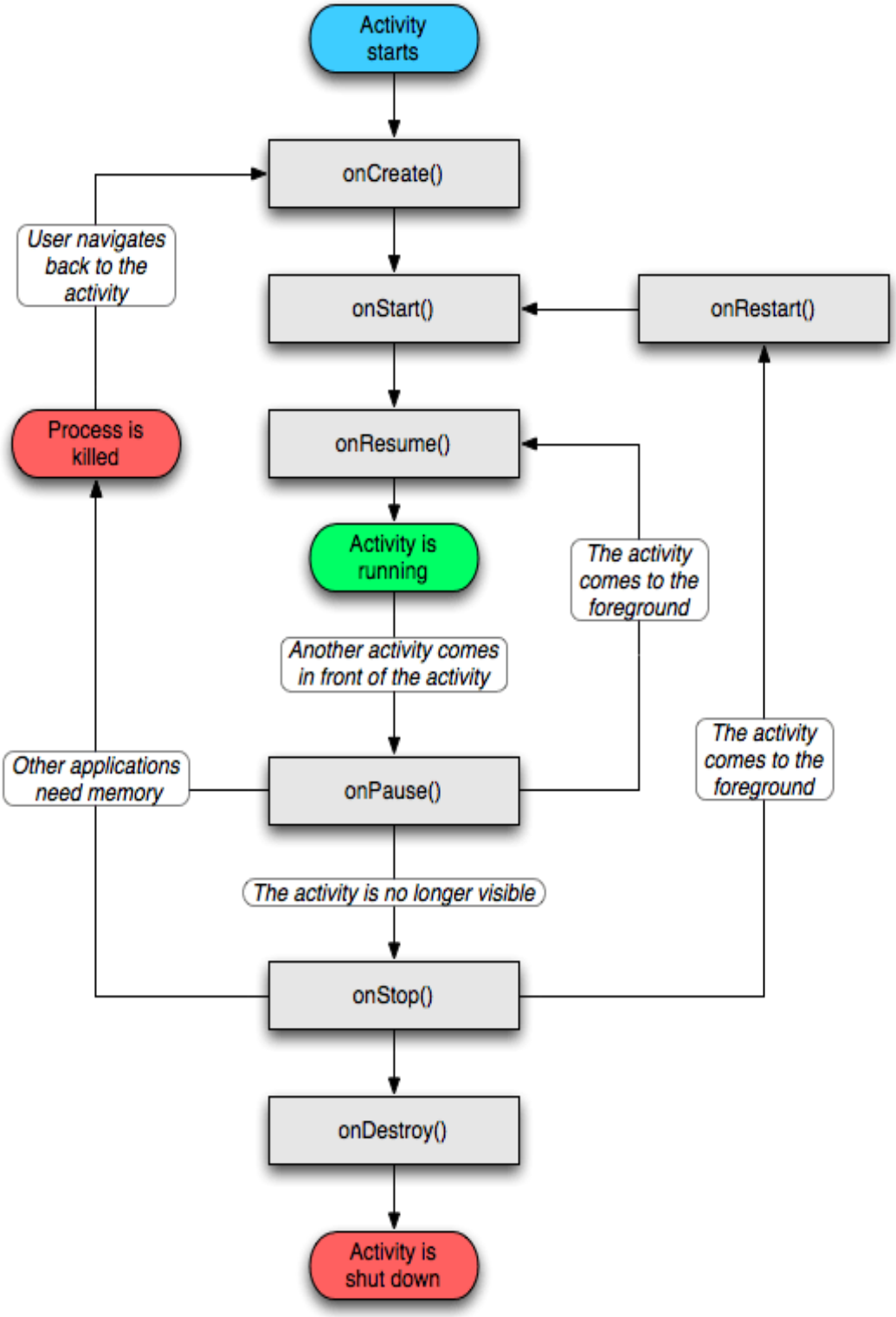
2.2.2.4. *Android Lifecycle*

Aplikasi *android Lifecycle* merupakan kumpulan dari beberapa *activity* yang tergabung secara bebas dengan yang lainnya. Yang mana kumpulan tersebut memiliki satu *activity* utama untuk memanggil *Activity* secara bertumpuk. *Activity* adalah suatu komponen pada aplikasi Android yang menyediakan tampilan, sehinggadapat berinteraksi dengan melakukan sesuatu, seperti dial telepon, mengambil foto, *play game*, *play music*, mengirim email. Setiap mendapatkan tampilan/*layout* sebagai antarmuka dengan *user* [12]. Sebuah aplikasi *android* akan terdiri atas beberapa komponen penting sebagai berikut :

1. *Activity*

   Activity merupakan layar tampilan pada sebuah aplikasi *android*. Secara sederhananya bahwa *activity* adalah layar yang pengguna dapat melihatnya. Sebuah aplikasi *android* dapat terdiri atas beberapa *activity* yang dapat dipindah dalam rentang waktu tertentu ketika menjalankan aplikasi. *Activity* ini merupakan komponen aplikasi yang umum dan sering berhubungan dengan tampilan, dimana *user* akan berinteraksi dengan aplikasi yang dibuat.

Skema *lifecycle* dari sebuah *activity* adalah sebagai berikut :



Gambar 2. 3 *Lifecycle Activity*

a. Ketika sebuah *activity* dipanggil, maka kode yang ada pada *methodonCreate()* akan dijalankan. Dan *method* ini juga akan memanggil satu parameter informasi status terakhir yang telah disimpan oleh *method onSaveInstanceState().*

b. *Method onStart()* digunakan untuk mengeksekusi *activity* ketika sebuah *user interface* ditampilkan.
c. *Method onPause()* akan dipanggil ketika ada *activity* lain yang akan mneggantikan *activity* yang ada.
d. *Method onStop()* digunakan ketika aplikasi sudah tidak dijalankan atau tidak dibutuhkan untuk sementara waktu.
e. *Method onRestart()* digunakan ketika *activity* di-*restart* dari posisi semula *onStop().*
f. *Method onDestroy()* akan dijalankan setelah *activity* aplikasi di-*destroy* atau ketika habis memori sehingga aplikasi akan di *terminate*.

2. Views

Views merupakan *widget interface* atau *antarmuka* pada tampilan aplikasi *android*, misalnya seperti *button*, *input text*. *Views* akan mempunyai atribut yang dapat diubah (*diklik*, *double klik*) atau diubah tampilannya (warna, ukuran).

3. Services

Services adalah layanan yang dilakukan dari perintah yang diberikan. Dan dijalankan di belakang layar atau tanpa menggunakan *user interface* pengguna. Misalnya seperti *MP3 player*, yang menjalankan musik selagi pengguna melakukan berbagai *activity*.

4. *Content Provider*

Content *provider* merupakan *interface* terstruktur untuk data.

5. Intents

Intents merupakan pesan yang bersifat asinkon yang akan memerintahkan aplikasi meminta sesuatu dari komponen yang ada pada

sistem *android*. Secara sederhananya, intents merupakan aksi yang akan dilakukan setelah mendapatkan perintah dari *services* atau *activity*.

6. *Broadcast Receiver*

Broadcast Receiver digunakan untuk menerima pesan dari sistem atau intents.

7. *Widgets Widgets*

meruapakan komponen interaktif yang ada pada layar utama *android*. Beberapa ikon *widget* akan berada interaktif pada layar yang memperbolehkan pengguna melakukan aksi tertentu.

2.2.3 *Object Oriented Programming* (OOP)

Pemrograman *Berorientasi Obyek* (*Object Oriented Programming* – OOP) adalah programming paradigma yang menggunakan *object* dan interaksinya untuk merancang aplikasi dan program komputer. OOP tidak banyak digunakan sebelum awal tahun 1990an. Tapi sekarang menjadi sesuatu yang sudah lumrah digunakan. Bahasa-bahasa pemrograman seperti keluarga *dotNet* dari *Microsoft* (*Visual Basic.Net, Visual C#, dan Visual J), Borland Delphi, Java, Phyton, PHP versi 5* keatas, *C++* dan banyak lainnya merupakan bahasa pemrograman yang mendukung konsep OOP.

Object Oriented Programming (OOP) dalam Bahasa Indonesia dikenal dengan pemrograman berorientasi *object* (PBO), sementara pada OOP penyelesaiian masalah dilakukan dengan cara memetakan subbagian program kedalam beberapa *class*. OOP adalah solusi untuk program besar karna OOP mudah dianalisis pada tiap *class* yang bersesuaiian sehingga jika ada suatu perubahan pada suatu *class*, maka *class* lainnya tidak terganggu [13].

2.2.3.1. Karakteristik Sistem *Object Oriented Programming* (OOP)

Karakteristik atau sifat-sifat yang dipunyai sebuah sistem berorientasi *object* adalah :

1. *Abstraksi*

Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk mode yang sederhana dengan mengabaikan aspek-aspek lain yang

2. *Enkapsulasi*.

   Pembungkusan atribut data dan operasi-operasi yang dipunyai objek.

3. Pewarisan

   Mekanisme yangmemungkinkan satu objek mewarisi sebagian atau seluruh definisi dari objek.

4. *Reusability*

   Pemanfaatan kembali objek yang sudah di definisikan untuk suatu permasalahan dan permasalahan lainnya yang melibatkan objek tersebut.

5. *Generalisasi* dan *Spesialisasi*

   Menunjukan hubungan antara kelasdan objek yang umum dengan kelas dan objek yang khusus.

6. Komunakasi antar objek

   Komunikasi antar objek dilakukan lewat pesan yang dikirim dari satu objek ke objek lainnya.

7. *Polymorphism*

   Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat bari program [14].

2.2.4 *Mobile*

Menurut Turban, *Mobile application* juga biasa disebut dengan *mobile apps* yaitu istilah yang digunakan untuk mendeskripsikan aplikasi internet yang berjalan pada *smart phone*. Aplikasi *mobile* biasanya membantu penggunanya untuk terkoneksi dengan layanan internet. Aplikasi *mobile* dikenali dalam dua bentuk yaitu :

1. Aplikasi Natif

   Merupakan aplikasi yang di program dan berjalan diatas perangkat *mobile*. Pada saat ini aplikasi natif sedang trend, penyebabnya adalah keberhasilan *Itunes AppStore* dan *Google Play Store.*

2. Aplikasi *Web Mobile*

   Merupakan aplikasi yang berjalan di dalam *browser web* perangkat *mobile*. Aplikasi *web* juga memanfaatkan fitur-fitur *HTML5* [15].

2.2.5. *Firebase*

Firebase pertama kali didirikan pada tahun 2011 oleh Andrew Lee dan James Tamplin. Produk yang pertama kali dikembangkan adalah Realtime Database, di mana developer dapat menyimpan dan melakukan sinkronasi data ke banyak user. Kemudian berkembang menjadi layanan penyedia pengembangan aplikasi. Pada Oktober 2014, perusahaan tersebut diakusisi oleh Google. Berbagai fitur terus dikembangkan hingga diperkenalkan pada Mei 2016 di Google I/O.

Untuk dapat mengimplementasikan layananan *Push Notification* diperlukan *cloud server*, salah satu *cloud server* yang bias digunakan adalah *Firebase*. *Firebase* adalah layanan pada *Google Cloud Messaging* (GCM) yang membantu pengembang mengirim data dari *server* untuk aplikasi mereka *Android*.

2.2.6. *Application Programming Interface* (API)

Antarmuka pemrograman aplikasi (*Application Programming Interface*/API) adalah sekumpulan perintah, fungsi, dan *protocol* yang dapat digunakan oleh *programmer* saat membangun perangkat lunak untuk *system* operasi tertentu. API memungkinkan *programmer* untuk menggunakan fungsi standar untuk berinteraksi dengan *system* operasi. API dapat menjelaskan cara sebuah tugas (task) tertentu dilakukan. Dalam *pemrograman procedural* seperti bahasa C, aksi biasanya dilakukan dengan media pemanggilan fungsi. Karena itu, API biasanya menyertakan penjelasan dari fungsi/rutin yang disediakannya [16].

API menyediakan fungsi dan perintah dengan bahasa yang lebih terstruktur dan lebih mudah untuk dipahami oleh *programer* bila dibandingkan dengan *System Calls*, hal ini penting untuk aspek editing dan pengembangan, sehingga *programer* dapat mengembangkan *system* dengan mudah. API juga dapat digunakan pada *System* Operasi mana saja asalkan sudah ada paket API nya. Dalam contoh program sederhana, dibutuhkan setidaknya ribuan *system calls* per detik. Oleh karena itu kebanyakan *programmer* membuat aplikasi dengan menggunakan *Application Programming Interface* (API). Dalam API itu terdapat fungsi-fungsi/perintah-perintah untuk menggantikan bahasa yang digunakan dalam *system calls* dengan bahasa yang lebih terstruktur dan mudah dimengerti oleh *programmer*. Fungsi yang dibuat dengan menggunakan API tersebut kemudian akan memanggil *system calls*.

2.2.6.1 Keuntungan menggunakan API

Keuntungan memprogram dengan menggunakan API adalah :

1. Portabilitas. *Programmer* yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah terinstall API tersebut. Sedangkan *system call* berbeda antar *system* operasi, dengan catatan dalam implementasinya mungkin saja berbeda.
2. Lebih Mudah Dimengerti. API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call*. Hal ini sangat penting dalam hal editing dan pengembangan.

Dengan adanya *System call interface* ini akan menerjemahkan perintah dalam API dan kemudian akan memanggil *system calls* yang diperlukan.

2.2.7. *Java*

Gambar 2. 4 Logo *Java*

Bahasa pemograman *java* merupakan salah satu dari sekian banyak bahasa pemograman yang dapat dijalankan diberbagai sistem oprasi termasuk telepon genggam (*Nofriadi,2015*) [17]. "*java* adalah bahasa pemrograman yang dapat dijalankan diberbagai komputer, termasuk telepon genggam". Adapun Bahasa pemrograman ini pertama kali dibuat oleh *james Gosling* saat masih bergabung *Sun Microsystems*. Bahasa pemrograman ini merupakan pengembangan dari bahasa C++ karena banyak mengadopsi sintak C dan C++.

Saat ini *Java* merupakan bahasa pemrograman yang paling popular digunakan. Kelebihan *Java* dari bahasa pemrograman yang lain adalah bahasa pemograman yang *berorientasi objek* (OOP) serta bisa dijalankan diberbagai jenis

sistem operasi sehingga dikenal juga bahasa pemrograman *multiplatform*, bersifat pemrograman *berorientasi objek* (PBO), memiliki library yang lengkap [18].

2.2.7.1 *Integrated Development Environment* (IDE)

Integrated Development Environment (IDE), merupakan sebuah *teks editor* untuk menuliskan *script* bahasa pemrograman *java*. Yaitu salah satunya *Netbeans*. *JasperRepot dan iReport*. *"JasperReport* sebuah *tool* yang sudah tidak asing lagi bagi pengguna *java*". *JasperReport hamper* menjadi *iconreporting* dalam *java, powerfull* untuk membuat laporan dalam bentuk *PDF, HTML, XLS, RTF, ODT, CSV, TXT,* dan *XML* (Huda, dkk, 2013).

JasperReport merupakan *software* (*library*) *open source* untuk *reporting* sebagai alternatif, terhadap *tools IReport* dengan (*library JasperReport*) [19].

Adapun yang menyebutkan bahwa *IReport* adalah *report designer visual* yang dibangun pada *JasperReports* yang mengisi kekurangan itu. *IReport* bersifat *intuitif* dan mudah digunakan pembuatan laporan *visual/designer* untuk *JasperReport* dan tertulis dalam kitab *Java*.

2.2.8. JSON (*JavaScript Object Notation*)

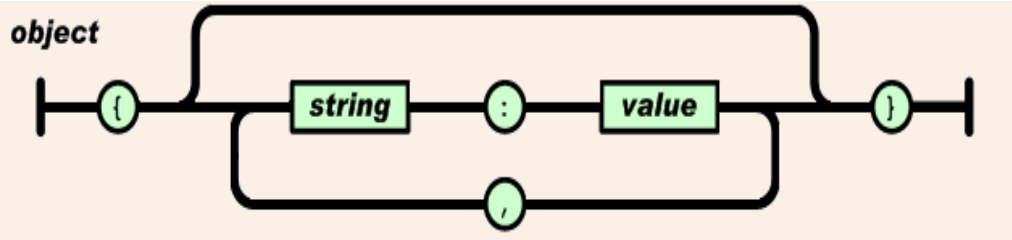
(*JavaScript Object Notation)* adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman *JavaScript, Standar ECMA-262* Edisi ke-3 - Desember 1999. *JSON*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

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau *associative array.*

2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

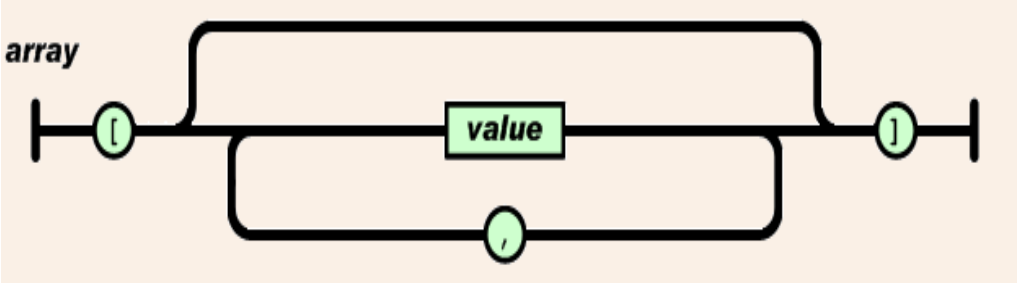
Struktur-struktur data ini disebut sebagai struktur data *universal*. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini, *JSON* menggunakan beberapa bentuk sebagai berikut :

1. 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). Objek biasanya digunakan untuk menyimpan data tunggal dalam bentuk *JSON* .



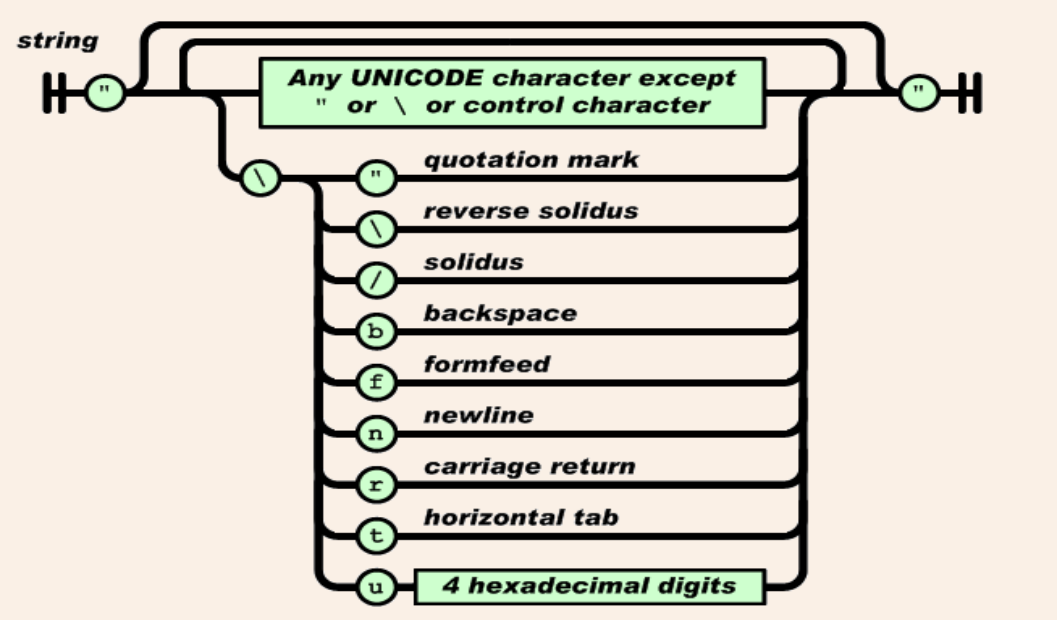
Gambar 2. 5 Objek *JSON*

2. Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh, (koma). Larik dalam *JSON* dapat digunakan sebagai *value* dari *JSON object* hal ini dapat berguna jika *JSON* menyimpan data bertingkat.



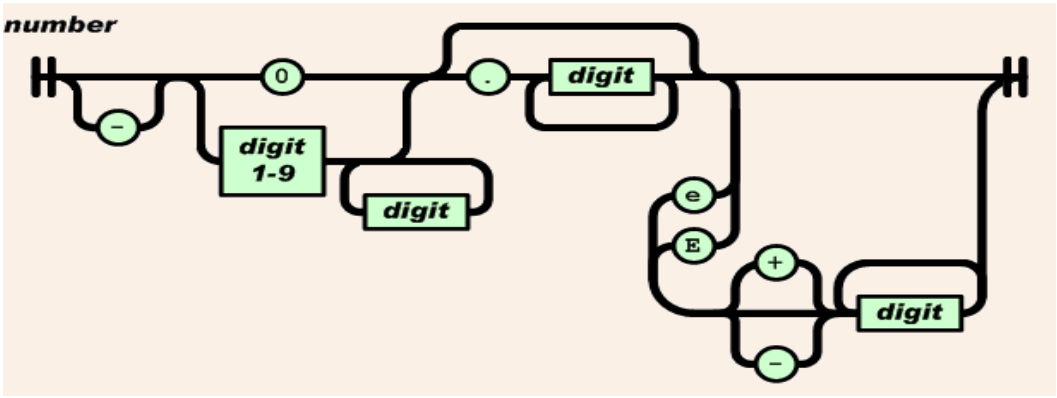
Gambar 2. 6 *Array JSON*

3. *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*.



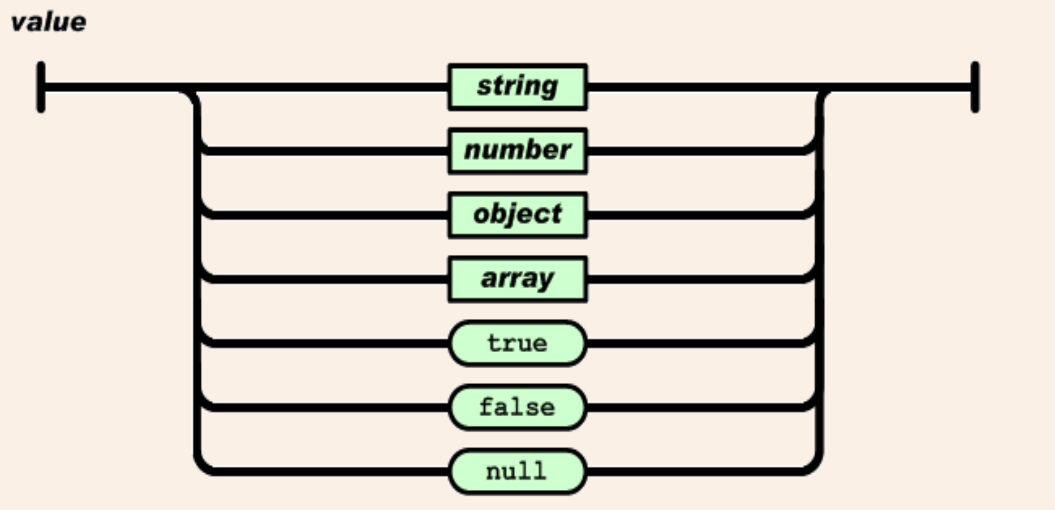
Gambar 2. 7 *String JSON*

4. Angka adalah sangat mirip dengan angka di C atau *Java*, kecuali format oktal dan heksadesimal tidak digunakan.



Gambar 2. 8 Angka *JSON*

5. Nilai (*value*) dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau *true* atau *false* atau *null*, atau sebuah objek atau sebuah larik. Strukturstruktur tersebut dapat disusun bertingkat.



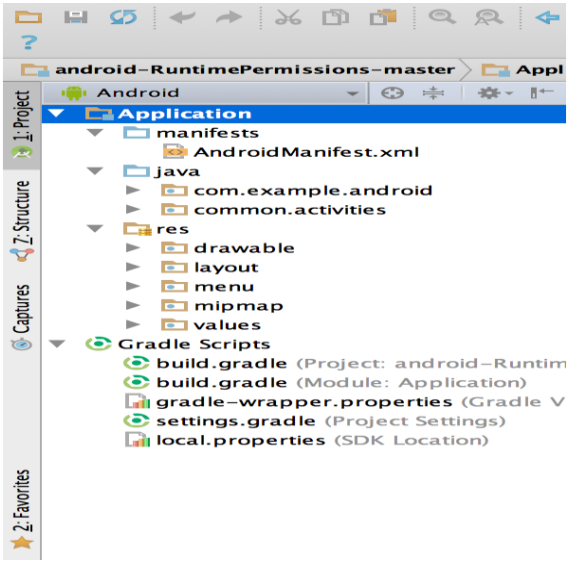
Gambar 2. 9 Nilai *JSON*

2.2.9. *Android Studio*

Android Studio adalah Lingkungan Pengembangan Terpadu - *Integrated Development Environment* (IDE) untuk pengembangan aplikasi *Android*, berdasarkan *IntelliJ* IDEA [11]. Selain merupakan *editor kode IntelliJ* dan alat pengembang yang berdaya guna, *Android Studio* menawarkan *fitur* lebih banyak untuk meningkatkan produktivitas Anda saat membuat aplikasi *Android*, dapat dimisalkan yaitu :

1. Sistem versi berbasis *Gradle* yang *fleksibel*
2. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat *Android*
3. *Instant Run* untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat *APK* baru
4. *Template kode* dan *integrasi GitHub* untuk membuat *fitur* aplikasi yang sama dan mengimpor kode contoh
5. Alat pengujian dan kerangka kerja yang *ekstensif*
6. Alat *Lint* untuk meningkatkan kinerja, kegunaan, *kompatibilitas* versi, dan masalah-masalah lain
7. Dukungan C++ dan NDK dan *Emulator* yang cepat dan kaya *fitur*
8. Dukungan bawaan untuk *Google Cloud Platform*, mempermudah pengintegrasian *Google Cloud Messaging* dan *App Engine*

1. Adapun struktur proyek pada *android* adalah sebagai berikut :



Gambar 2. 10 Tampilan File Struktur *Android Studio*

Setiap proyek di Android Studio berisi satu atau beberapa modul dengan file kode sumber dan file sumber daya. Jenis-jenis modul mencakup :

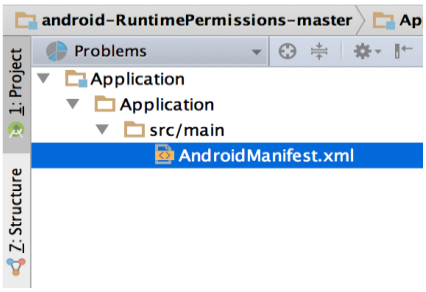
1. Modul aplikasi Android
2. Modul Pustaka
3. Modul Google App Engine

Secara default, Android Studio akan menampilkan file proyek Anda dalam tampilan proyek Android, seperti yang ditampilkan dalam gambar 1. Tampilan disusun berdasarkan modul untuk memberikan akses cepat ke file sumber utama proyek Anda. Semua file versi terlihat di bagian atas di bawah Gradle Scripts dan masing-masing modul aplikasi berisi folder berikut:

1. manifests: Berisi file AndroidManifest.xml.
2. java: Berisi file kode sumber Java, termasuk kode pengujian JUnit.
3. res: Berisi semua sumber daya bukan kode, seperti tata letak XML, string UI, dan gambar bitmap.

Struktur proyek Android pada disk berbeda dari representasi rata ini. Untuk melihat struktur file sebenarnya dari proyek ini, pilih Project dari menu tarik turun Project (dalam gambar 1, struktur ditampilkan sebagai Android).

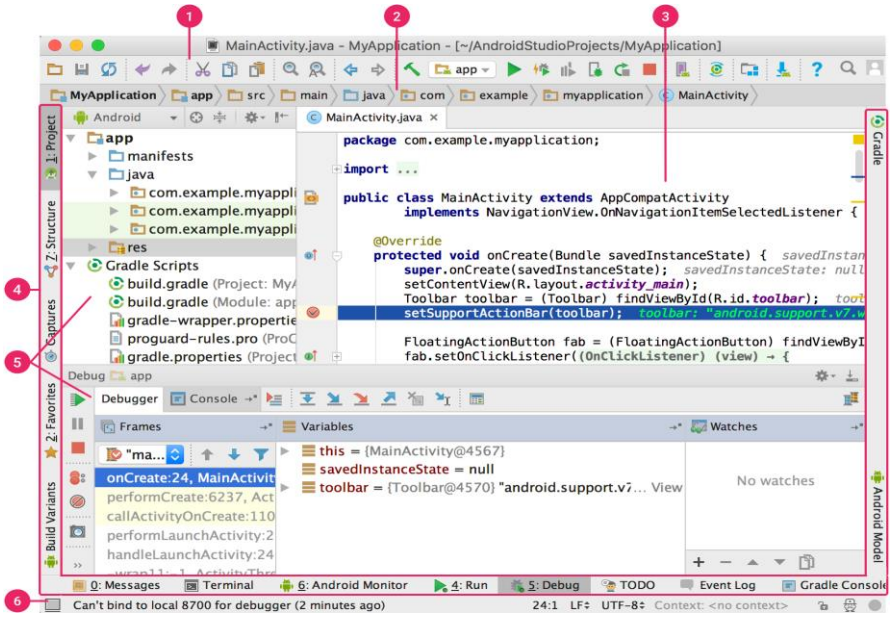
Anda juga bisa menyesuaikan tampilan file proyek untuk berfokus pada aspek tertentu dari pengembangan aplikasi Anda. Misalnya, memilih tampilan Problems dari tampilan proyek Anda akan menampilkan tautan ke file sumber yang berisi kesalahan pengkodean dan sintaks yang dikenal, misalnya tag penutup elemen XML tidak ada dalam file tata letak.



Gambar 2. 11 Tampilan *File* Proyek *Android Studio*

2. Antarmuka Pengguna pada *Android Studio*

menjalankan pengujian antarmuka Pengguna pada Android Studio merupakan bagian penting dari siklus development aplikasi Android. Pengujian yang ditulis dengan baik bisa membantu menangkap bug sejak dini dalam siklus development, sehingga lebih mudah diperbaiki, dan menambah kepercayaan dalam kode berikut adalah penjelasan dari Gambar Antarmuka Penggunaan pada Android Studio :



(sumber : https://www. developer.android.com/studio/intro/ 05-Oktober-2018)

Gambar 2. 12 Tampilan Jendela Utama *Android Studio*

1. Bilah alat memungkinkan Anda untuk melakukan berbagai jenis tindakan, termasuk menjalankan aplikasi dan meluncurkan alat Android.
2. Bilah navigasi membantu Anda bernavigasi di antara proyek dan membuka file untuk diedit. Bilah ini memberikan tampilan struktur yang terlihat lebih ringkas dalam jendela Project.
3. Jendela editor adalah tempat Anda membuat dan memodifikasi kode. Bergantung pada jenis file saat ini, editor dapat berubah. Misalnya, ketika melihat file tata letak, editor menampilkan Layout Editor.
4. Bilah jendela alat muncul di luar jendela IDE dan berisi tombol yang memungkinkan Anda meluaskan atau menciutkan jendela alat individual.
5. Jendela alat memberi Anda akses ke tugas tertentu seperti pengelolaan proyek, penelusuran, kontrol versi, dan banyak lagi. Anda bisa meluaskan dan juga menciutkannya.
6. Bilah status menampilkan status proyek Anda dan IDE itu sendiri, serta setiap peringatan atau pesan.

2.2.10. MySQL

MySQL merupakan software database yang paling populer di kalangan para pengembang aplikasi database. MySQL juga adalah suatu perangkat lunak yang menganut atau mengimplementasikan model basis data relasional maka MySQL disebut sebagai Relational Database Management System (RDBMS). MySQL merupakan software RDBMS (atau server database) yang dapat mengelola database dengan cepat, dapat menampung data dalam jumlah sangat besar, dapat diakses oleh banyak user (multi-user), dan dapat melakukan suatu proses secara sinkron atau berbarengan (multi-threaded), Dikarenakan faktor open source dan popular tersebut maka cocok untuk mendemontrasikan proses replikasi basis data.

Saat ini, MySQL banyak digunakan di berbagai kalangan untuk melakukan penyimpanan dan pengolahan data, mulai dari kalangan akademis sampai ke industri, baik industri kecil, menengah, maupun besar. Lisensi MySQL terbagi menjadi dua. Anda dapat menggunakan MySQL sebagai produk open source

dibawah GNU General Public License (gratis) atau dapat membeli lisensi dari versi komersialnya.

MySQL versi komersial tentu memiliki nilai lebih atau kemampuan-kemampuan yang tidak disertakan pada versi gratis. Pada kenyataannya, untuk keperluan industri menengah kebawah, versi gratis masih dapat digunakan dengan baik. Beberapa contoh aplikasi yang menggunakan MySQL adalah Joomla, Wordpress, MyBB, phpBB, dan masih banyak lagi [20].

2.2.11. Sublime Text 3

Sublime Text merupakan text editor berbagai bahasa pemrograman mulai dari bahasa c hingga java. Sublime Text merupakan editor HTML yang profesional untuk mendesain, menulis kode program, dan mengembangkan ke halaman website, halaman web, dan aplikasi web. Sublime Text 3 memiliki beberapa kelebihan yaitu:

1. Multi Platform

   Kelebihan dari Sublime Text adalah software ini tersedia dalam berbagai platform sistem operasi, antara lain Windows, Linux, dan MacOs.

2. Plugin

   Plugin-nya sangat beragam, sehingga bisa memudahkan programmer dalam mengembangkan software-nya. Sublime Text juga memiliki sangat banyak package. Untuk menginstall package kita bisa menjalankan Package Manager kemudian akan muncul list package, kita tinggal mencari (tentu saja menggunakan fuzzy search juga) package yang diinginkan.

3. Tema dan color scheme yang bervariasi

   Di Sublime Text, kita bisa mengganti Color Scheme (untuk area editor) maupun Theme (untuk interface sidebar, tab, console, search dll). Untuk mengedit color scheme, kita harus membuka file xml berekstensi .tmTheme dan mengedit isinya, atau bisa juga menggunakan .tmTheme editor.

4. Go To Anything

   Dengan Go to Anything, kita bisa membuka file di dalam project dengan cepat, tinggal tekan Ctrl + P kemudian ketik nama filenya. Untuk mencari nama file tidak harus mengetik secara tepat, karena adanya algoritma fuzy.

5. Drag & Drop

   Menyeret dan melepas file teks ke dalam editor akan membuka tab baru secara otomatis. Kita juga bisa menentukan lokasi tab pada saat menyeret file teks tersebut.

6. Membuka File Besar

   Sublime Text mampu membuka dan mengedit sebuah file teks yang sangat besar tanpa masalah.

7. Command Palette

   Sublime Text memiliki tampilan yang lebih simple dan sangat minim menu, bahkan tidak ada toolbar sama sekali. Kebanyakan perintah-perintahnya bisa kita akses menggunakan Shortcut, atau kalau belum hafal shortcut kita bisa mengakses Command Palette (tekan Ctrl + Shift + P) kemudian cari perintah yang kita inginkan (menggunakan fuzzy search).

8. Column Editing

   Baik Notepad++ maupun Sublime Text memiliki fitur ini. Di Notepad++ tekan Alt kemudian mouse didrag ke bagian yang ingin diedit. Di Sublime Text menggunakan drag tombol mouse tengah, atau bisa juga dengan Shift + drag tombol mouse kanan.

9. Split Editing

   Jika and memiliki monitor yang lebar maka kedua editor ini sangat cocok karena sama-sama memiliki fitur untuk Split Editing (membuka 2 atau lebih file).

10. Auto- Completion

    Sublime Text memiliki auto complete untuk beberapa bahasa yang saya pakai seperti PHP, CSS, Javascript. Fitur ini juga mendukung fuzzy search sehingga tidak harus mengetik secara tepat. Notepad++ juga memiliki auto complete, namun tidak senyaman Sublime Text dan harus diaktifkan dulu melalui Settings -> Preferences -> Backup/Auto-Completion.

11. Mini map / document map

    Sublime Text memiliki Minimap, semacam versi mini dari file untuk mempermudah melihat file secara keseluruhan.

12. Go To Definition

    Ini adalah fitur baru Sublime Text 3 , fitur ini sangat membantu menemukan function/class di dalam project, caranya dengan meletakkan cursor di nama function/class kemudian tekan F12, maka file yang berisi definisi function/class tersebut akan terbuka. Untuk melihat list semua function/class bisa menggunakan Go to Symbol in Project (tekan Ctrl + Shift + R) [21].

2.2.12. PHP

Hypertext Processor (PHP) merupakan bahasa standar yang digunakan dalam dunia website. Konsep PHP sangat sederhana sehingga dalam membuat dokumen PHP cukup membuat sebuah HTML biasa, hanya saja di tambahkan dengan kode-kode program yang diapit dalam tanda <?...?>. dalam hal ini interpreter PHP dalam mengeksekusi kode PHP ini berjalan pada sisi server (server-side). Sehingga sangat berbeda sekali dengan program pada sisi client (clienr-side). Spasi tidak berpengaruh pada penulisan perinta PHP.

PHP pertama sekali dibuat oleh Rasmus Lerdorf pada tahun 1995.Pada awal itu PHP bernama FI (Form Interpreted). Pada saat tersebut PHP adalah sekumpulan script yang di gunakan untuk mengolah data form dari web. Perkembangan selanjutnya adalah Rasmus melepaskan kode sumbernya tersebut dan menamakannya PHP/FI. Pada saat tersebut itu juga singkatan PHP/FI adalah Personal Home Page / Form Interpreter.Dengan pelepasan kode sumber ini menjadi open source, maka banyak programmer yang tertarik untuk mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilisan ini interpreter sudah diimplementasikan dalam C. Dalam rilisan ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan. Pada tahun ini juga sebuah perusahan yang bernama Zend, menulis ulang interpreter PHP menjadi lebih bersih, lebih baik dan lebih cepat. Kemudian pada tahun 1998 perusahaan tersebut merilis interpreter baru untuk PHP dan meresmikan nama rilisan itu menjadi PHP 3.0.

Pada tahun 1999, Zend merilis interpreterPHP baru dan rilisan tersebut dikenal dengan PHP 4.0 adalah versi PHP yang paling banyak dipakai. Versi ini

banyak dipakai sebab versi ini mampu membangun aplikasi web kompleks tetapi tetap memiliki kecepatan proses dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0.PHP versi 5 muncul untuk menangani kelemahan-kelemahan yang terdapat pada versi sebelumnya. Versi ini adalah versi mutahir dari PHP.Dalam versi ini, inti dari interpreterPHP mengalami perubahan besar. Dalam versi ini juga dikenal model pemrograman berorientasi objek baru untuk menjawab perkembangan bahasa pemrograman kearah pemrograman berorientasi objek [20].

2.2.13. *Unified Modeling Language* (UML)

UML memiliki 13 jenis diagram resmi yaitu activity diagram, Class, Comunnication, Component, Composite structure, Deployment, Interaction overview, Object, Package, Sequence, State mechine, Timing, dan diagram Use case. Meskipun jenis jenis diagram ini merupakan cara orang-orang memperlakukan UML. Para perancang UML tidak memandang diagram sebagai bagian yang sentral. Dan hasilnya, jenis-jenis diagram bukanlah hal yang mutlak. secara legal dapat menggunakan elemen-elemen satu jenis diagram untuk diagram yang lain [22]. Pada penelitian ini diagram yang digunakan adalah diagram Use case, Class, activity, dan Sequence.

2.2.13.1 *Use Case Diagram*

Use case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. Use case mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan. Sebuah use case adalah serangkaian skenarion yang dikemas menjadi satu oleh tujuan pengguna umum.

Dalam bahasa use case, para pengguna disebut sebagai aktor. Aktor merupakan sebuah peran yang dimainkan oleh seorang pengguna dalam kaitannya dengan sistem. aktor tidak harus manusia. Jika sebuah sistem melakukan layanan untuk sebuah sistem computer lain, sistem lain tersebut merupakan aktor.

Cockburn menjelaskan sebuah skema tingkatan use case. Terdapat tingkat sea level, fish level dan kite level.

1. Sea level

   Inti use case berada pada tingkat sea level. Use case sea level khususnya mewakili sebuah interaksi diskrit antara aktor utama dan sistem. use case semacam ini akan memberikan suatu nilai pada aktor utama dan biasanya memberi waktu beberapa menit sampai setengah jam bagi aktor utama untuk menyelesaikan.

2. Fish level

   Use case yang ada pada tinkat fish level adalah use case yang ada hanya karena mereka dimasukkan oleh use case sea level.

3. Kite level

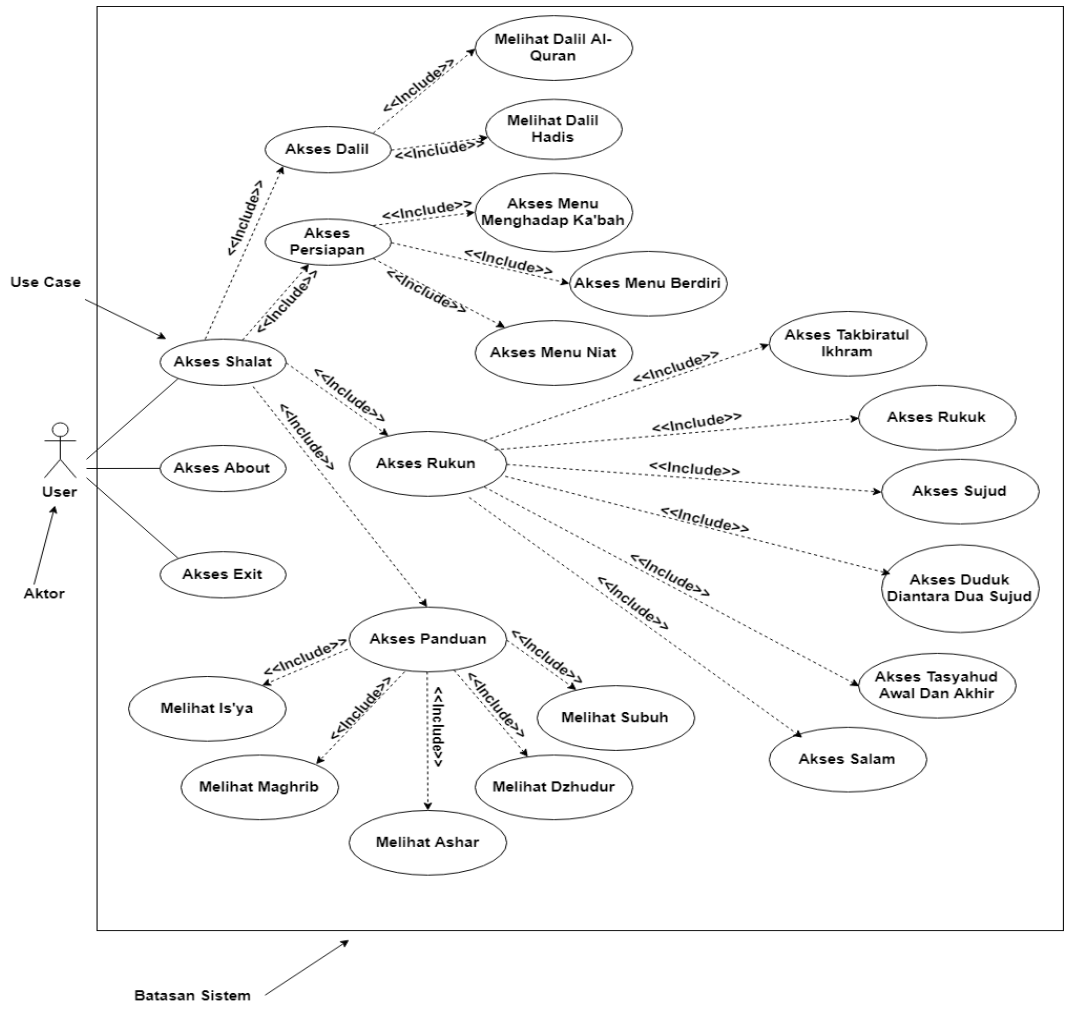
   Use case kite level menampilkan bagaimana use case sea level sesuai

Dengan interaksi bisnis yang lebih luas. Use case kite level biasanya merupakan use case bisnis.

Use case diagram menampilkan aktor, use case, dan hubungan antara mereka:

1. Aktor mana yang menggunakan use case mana.
2. Use case mana yang memasukkan use case lain.

Berikut ini adalah contoh dari diagram use case :



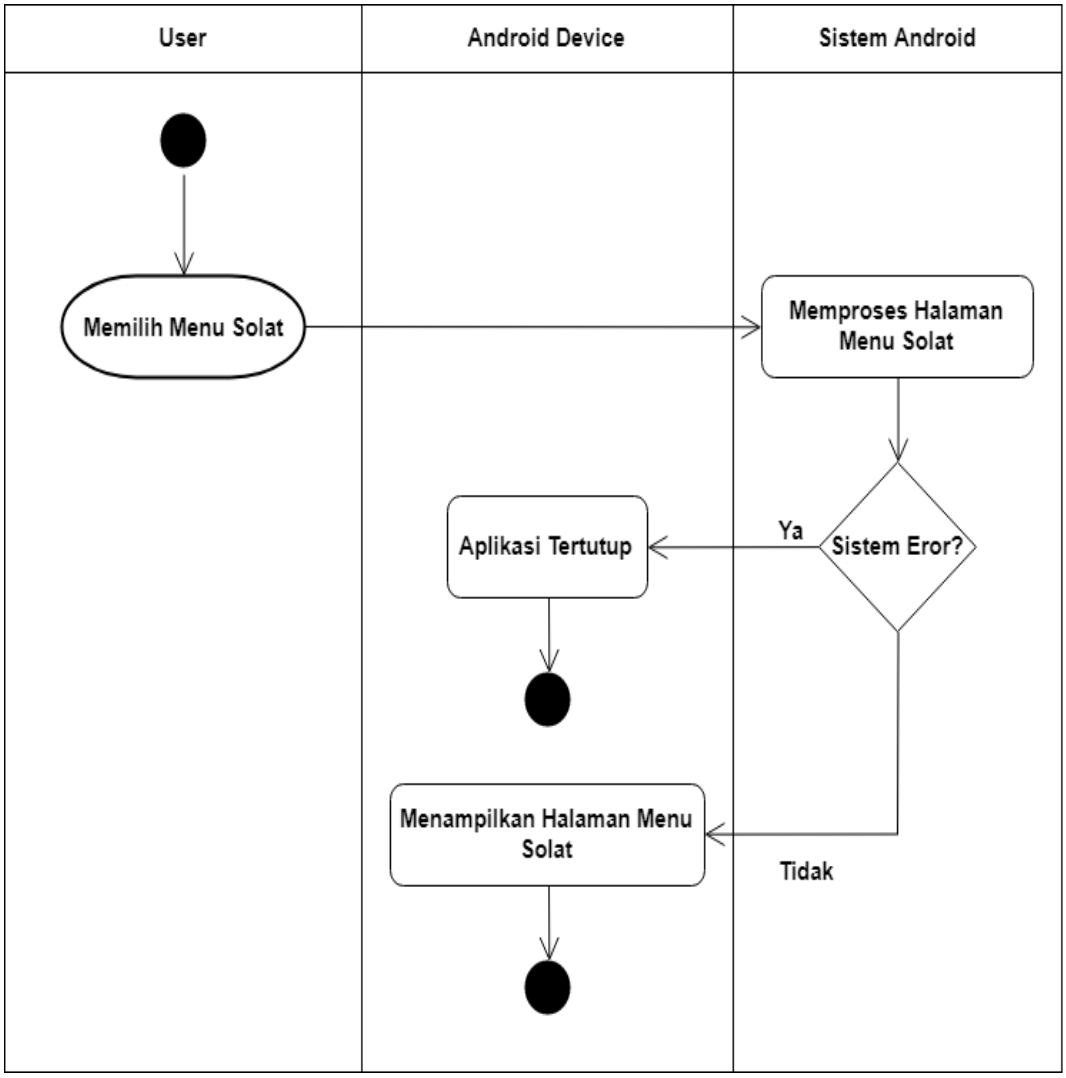
Sumber : S. H. Nazaruddin, Aplikasi Berbasis Android Berbagai Implementasi dan Pengembangan Aplikasi Mobile Berbasis Android, Bandung: Informatika, 2015 [22].

Gambar 2. 13 Contoh *Use Case Diagram*

2.2.13.2Activity Diagram

Activity diagram adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan alur kerja pada setiap use case. Dalam beberapa hal, activity diagram memainkan peran mirip sebuah diagram alir, tetapi perbedaan prinsip antara activity diagram dan diagram alir adalah diagram alir mendukung behavior pararel. Activity diagram dapat dipisahkan ke dalam partisi-partisi yang menampilkan action mana yang dilakukan oleh sebuah class atau organisasi. Model ini biasa disebut dengan swim lanes.

Berikut ini adalah contoh dari Activity Diagram :



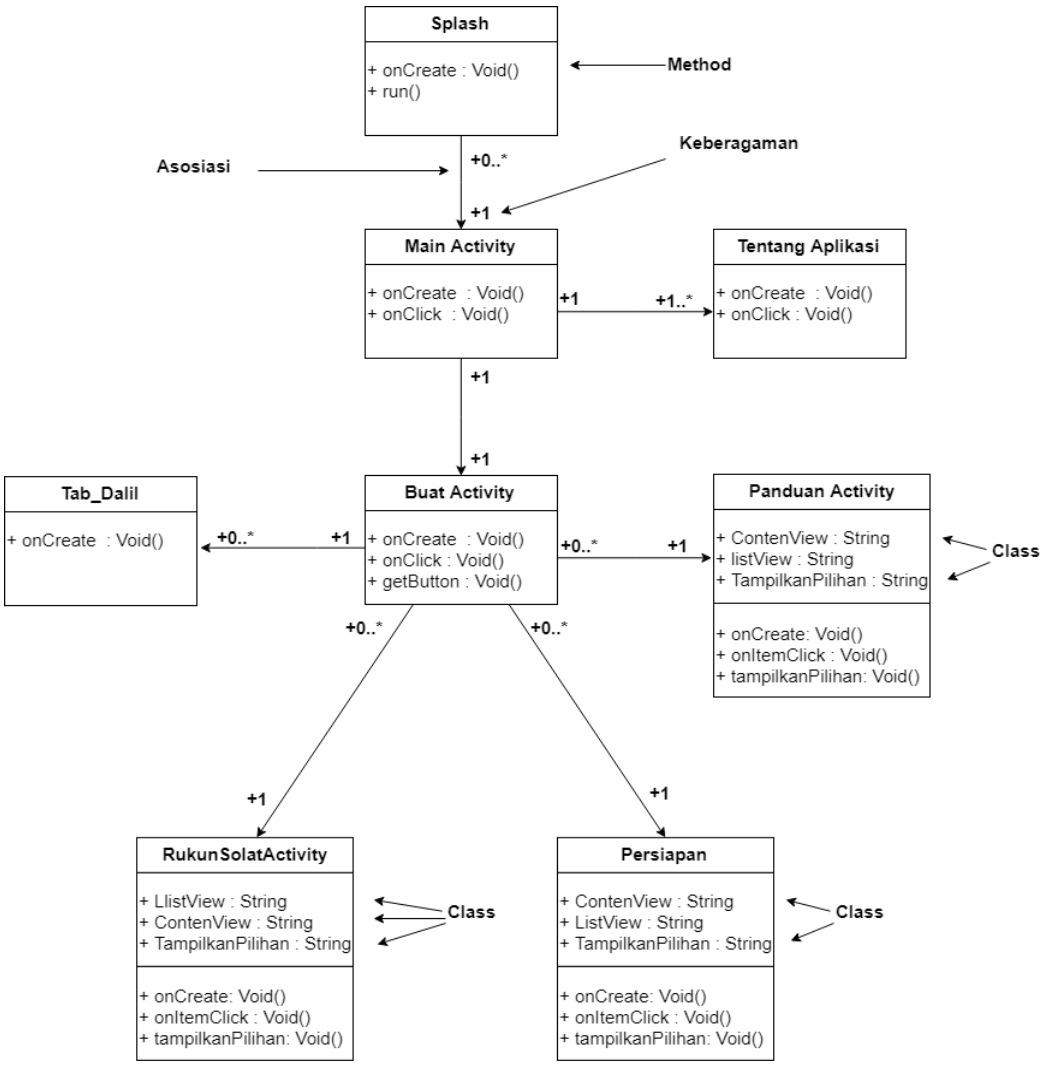
Sumber : S. H. Nazaruddin, Aplikasi Berbasis Android Berbagai Implementasi dan Pengembangan Aplikasi Mobile Berbasis Android, Bandung: Informatika, 2015 [22].

Gambar 2. 14 Contoh *Activity Diagram*

2.2.13.3 Class Diagram

Class diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara mereka. Diagram class juga menunjukan properti dan operasi sebuah class dan batasan-batasan yang terdapat pada sebuah hubungan-hubungan objek. UML menggunakan istilah fitur sebagai istilah umum yang meliputi properti dan operasi sebuah class.

Didalam class dibagi menjadi tiga bagian. Nama class, atributnya, dan operasinya. Berikut ini adalah contoh dari Class Diagram :



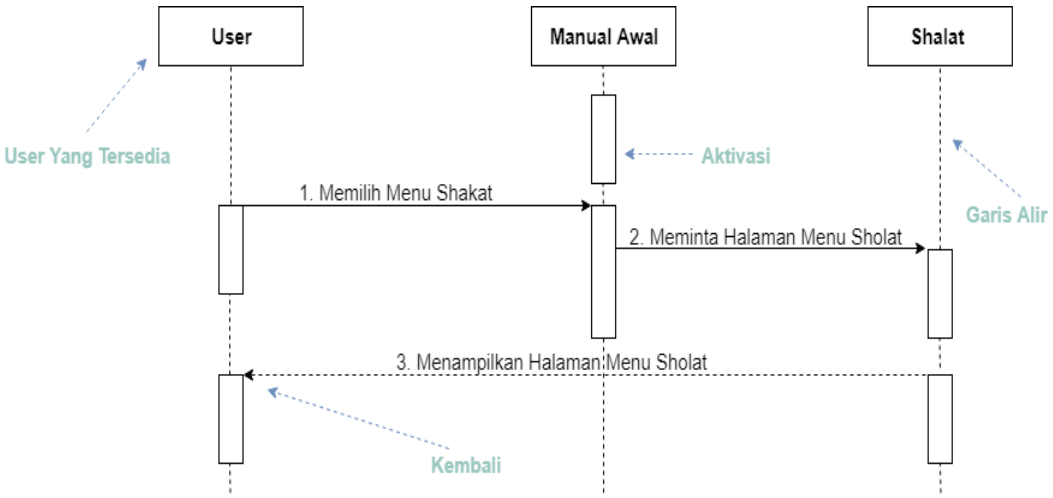
Sumber : S. H. Nazaruddin, Aplikasi Berbasis Android Berbagai Implementasi dan Pengembangan Aplikasi Mobile Berbasis Android, Bandung: Informatika, 2015 [22].

Gambar 2. 15 Contoh *Class Diagram*

2.2.13.4 Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display/form) berupa message yang digambarkan terhadap waktu. Sequence diagram terdiri atasdimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Sequence diagram biasa

digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event.*



Sumber : S. H. Nazaruddin, Aplikasi Berbasis Android Berbagai Implementasi dan Pengembangan Aplikasi Mobile Berbasis Android, Bandung: Informatika, 2015 [22].

Gambar 2. 16 Contoh Sequence Diagram

2.2.14 *Entity Relationship Diagram* (ERD)

Pemodelan awal basi data yang paling banyak digunakan adalah menggunakan *Entity Relationship Diagram* (ERD). ERD dikembangkan berdasarkan teori himpunan dalam bidang matematika. ERD digunakan untuk pemodelan basis data relasional. Sehingga jika penyimpanan basis data tidak perlu menggunakan ERD. ERD biasanya memiliki hubungan *binary* (satu relasi menghubungkan dua buah entitas). Beberapa metode perancangan ERD menoleransi hubungan relasi *ternary* satu relasi menghubungkan tiga buah relasi) atau N-ary (satu relasi menghubungkan banyak entitas) tapi banyak metode perancangan ERD yang tidak mengizinkan hubungan ternary dan N-ary [23].

2.2.15 *Data Flow Diagram* (DFD)

Data Flow Diagram (DFD) awalnya dikembangkan oleh Chris Gane dan Trish Sarson pada tahun 1979 yang termasuk dalam *Structured System Analysis* and *Design Methodology* (SSADM) yang ditulis oleh Chris Gane dan Tris Sarson. Sistem yang dikembangkan ini berbasi pada dekomposisi fungsional dari sebuah

sistem. Namun DFD Edward Yourdon dan Tom DeMarco populer digunakan sebagai model analisis sistem perangkat lunak untuk sistem perangkat lunak yang akan diimplementasikan dengan pemrograman terstruktur.

Informasi yang ada di dalam perangkat lunak dimodifikasi dengan beberapa transformasi yang dibutuhkan. DFD sering juga disebut sebagai aliran data atau representasi grafik yang menggambarkan aliran informasi dan transformasi informasi yang diaplikasikan sebagai data yang mengalir dari masukan dan keluaran. DFD dapat digunakan untuk merepresentasikan sebuah sistem atau perangkat lunak pada beberapa level abstraksi. DFD dapat dibagi menjadi beberapa level yang lebih detail untuk mereprensentasikan aliran informasi atau fungsi yang lebih detail.

DFD menyediakan mekanisme untuk pemodelan fungsional atau pemodelan aliran informasi. Oleh karena itu, DFD lebih sesuai digunakan untuk memodelkan fungsi-fungsi perangkat lunak yang akan diimplementasikan menggunakan pemrograman terstruktur karena pemrograman terstruktur membagibagi bagiannya dengan fungsi-fungsi dan prosedur-prosedur.

2.2.16 Metode Pengujian Sistem

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

2.2.16.1. Pengujian Black Box

Pengujian untuk mengetahui apakah semua fungsi perangkat lunak telah berjalan semestinya sesuai dengan kebutuhan fungsional yang telah didefinsikan. Metode Black Box memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program.

Pengujian Black Box dapat menemukan kesalahan dalam kategori berikut [24]:

1. Fungsi-fungsi yang tidak benar atau hilang
2. Kesalahan interface
3. Kesalahan dalam strutur data atau akses basisdata eksternal

4. Inisialisasi dan kesalahan terminasi
5. validitas fungsional
6. kesensitifan sistem terhadap nilai input tertentu
7. batasan dari suatu data

2.2.16.2. Skala Likert

Skala Likert digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang penelitian. Dengan skala likert, variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyusun item-item instrumen yang dapat berupa pernyataan atau pertanyaan.

Dalam skala Likert terdapat dua bentuk pernyataan yaitu pernyataan positif yang berfungsi untuk mengukur sikap positif, dan pertanyaan negative yang berfungsi untuk mengukur sikap negative objek. Skor pernyataan positif dimulai dari 1 untuk sangat tidak setuju (STS), 2 untuk tidak setuju (TS), 3 untuk ragu-ragu (RG), 4 untuk setuju (S), dan 5 untuk sangat setuju (SS). Skor pernyataan negative dimulai dari 1 untuk sangat tidak setuju (STS), 2 untuk tidak setuju (TS), 3 untuk raguragu (RG), 4 untuk setuju (S), dan 5 untuk sangat setuju (SS). Beberapa menghilangkan option “Ragu-ragu” dalam instrument untuk memudahkan dalam melihat angket yang responden isikan. Skala Likert digunakan untuk mengukur kesetujuan dan ketidaksetujuan seseorang terhadap sesuatu rencana program, pelaksanaan program ataupun tingkat keberhasilan suatu program.

Berikut ini adalah tabel Panduan Pemberian Skor untuk lebih detail nya dapat dilihat pada Tabel 2.1.

Tabel 2. 1 Panduan Pemberian Skor

Jenis Pertanyaan	**Alternatif Jawaban**				
	SS	**S**	**RG**	**TS**	**STS**
Skor	**5**	**4**	**3**	**2**	**1**

Kelebihan Skala Likert :

1. Mudah dibuat dan di terapkan.
2. Skala Likert lebih mudah membuatnya dibanding lain sepertti skala Thurstone.
3. Terdapat kebebasan dalam memasukan pertanyaan- pertanyaan, asalkan sesuai dengan konteks permasalahan yang diteliti.
4. Jawaban suatu item dapat berupa alternative, sehingga informasi mengenai item tersebut diperjelas.
5. Reliabilitas pengukuran bisa diperoleh dengan jumlah item tersebut diperjelas.

Kekurangan Skala Likert :

1. Kadangkala total skor dari individu tidak memberikan arti yang jelas, karena banyak pola respons terhadap beberapa item akan memberikan skor yang sama