# BAB 2
# LANDASAN TEORI

## 2.1 Memasak

Memasak adalah kegiatan menyiapkan makanan untuk dimakan dengan cara memanaskan pada bahan makanan agar bahan makanan tersebut bisa dikonsumsi. Memasak terdiri dari berbagai macam metode, teknik, peralatan, dan kombinasi bumbu dapur untuk mengatur rasa memudahkan makanan untuk dicerna dan mengubah makanan dari segi warna, rupa, rasa, tekstur, penampilan dan nilai nutrisi. Memasak secara umum adalah persiapan dan proses memilih, mengatur kuantitas, dan mencampur bahan makanan dengan urutan tertentu dengan tujuan untuk medapatkan hasil yang diinginkan. Memanaskan bahan makanan umumnya, walaupun tidak selalu, perubahan bahan makanan tersebut secara kimiawi, mengakibatkan adanya perubahan rasa, tekstur, penampilan, dan nilai nutrisi.[1]

## 2.2 Sensor

Sensor adalah alat yang dapat digunakan untuk mengukur, menganalisa, memantau suatu kondisi dan kemudian merespon terhadap perubahan di sekitarnya. Alat ini dapat ditemukan pada perangkat modern seperti smartphone dengan sistem operasi android, sebagai ponsel pintar yang memiliki sepaket teknologi canggih.[6]

### 2.2.1 Sensor Proximity

Sensor *proximity* adalah sebuah sensor yang bisa mendeteksi keberadaan benda tanpa kontak fisik. Sensor *proximity* memancarkan medan elektromagnetik atau sinar radiasi elektromagnetik (misalnya inframerah) dan mendeteksi perubahan bidang dengan mengembalikan sinyal. Ada empat jenis teknologi sensor *proximity*, diantaranya *Electrical* (*Inductive* dan *Capacitive*), *Optical* (IR dan Laser), *Magnetic*, *Sonar* .

Dari beberapa jenis sensor *proximity*, jenis sensor *proximity* dengan tipe *optical* paling banyak digunakan, karena jenis sensor *proximity* dengan tipe *optical*

lebih murah dan lebih sederhana dibandingkan dengan jenis sensor yang lainnya, dan bisa mendeteksi benda hingga jarak 5CM. Sensor *proximity* pada ponsel android biasanya terdapat di depan ponsel, tepatnya diatas layar ponsel seperti pada Gambar II-1




Gambar II-1 Sensor Proximity Pada Smartphone

Pada ponsel android sensor *proximity* biasa digunakan untuk mematikan layar saat terjadi panggilan masuk. Hal ini bertujuan untuk menghemat konsumsi batere saat pengguna sedang melakukan panggilan telepon. Sensor *proximity* akan aktif otomatis dan mendeteksi ketika benda dalam jangkauan sensor, kemudian sensor akan mengirimkan informasi ke sistem ketika benda yang mendekati sensor, dan sistem akan mematikan layar selama benda dekat dengan sensor selama panggilan terjadi.

Proximity Switch atau Sensor Proximity adalah alat pendeteksi yang bekerja berdasarkan jarak obyek terhadap sensor. Karakteristik dari sensor ini adalah menditeksi obyek benda dengan jarak yang cukup dekat, berkisar antara 1 mm sampai beberapa centi meter saja sesuai type sensor yang digunakan. Proximity Switch ini mempunyai tegangan kerja antara 10-30 Vdc dan ada juga yang menggunakan tegangan 100-200VAC.



Gambar II-2 Proximity Switch

Hampir di setiap mesin mesin produksi sekarang ini menggunakan sensor jenis ini, sebab selain praktis sensor ini termasuk sensor yang tahan terhadap benturan ataupun goncangan, selain itu mudah pada saat melakukan perawatan ataupun perbaikan penggantian.

Proximity Sensor terbagi dua macam, yaitu:

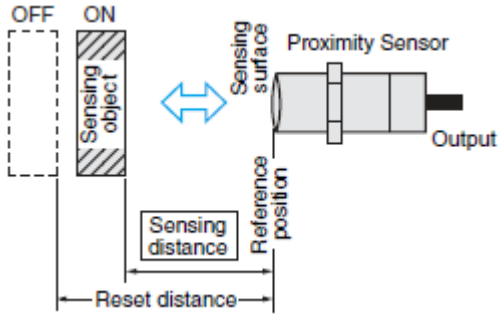
- Proximity Inductive
- Proximity Capacitive

Proximity Inductive berfungsi untuk mendeteksi obyek besi/metal. Meskipun terhalang oleh benda non-metal, sensor akan tetap dapat mendeteksi selama dalam jarak (nilai) normal sensing atau jangkauannya. Jika sensor mendeteksi adanya besi di area sensingnya, maka kondisi output sensor akan berubah nilainya.

Proximity Capacitive akan mendeteksi semua obyek yang ada dalam jarak sensingnya baik metal maupun non-metal.

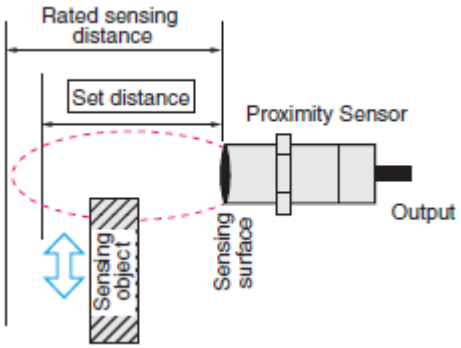
**Jarak Diteksi**

Jarak diteksi adalah jarak dari posisi yang terbaca dan tidak terbaca sensor untuk operasi kerjanya, ketika obyek benda digerakkan oleh metode tertentu.



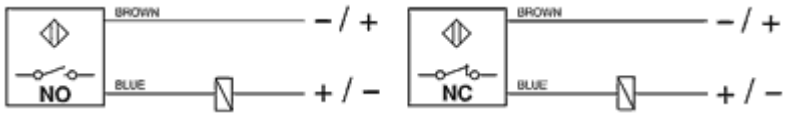

**Pengaturan jarak**

Mengatur jarak dari permukaan sensor memungkinkan penggunaan sensor lebih stabil dalam operasi kerjanya, termasuk pengaruh suhu dan tegangan. Posisi objek (standar) sensing transit ini adalah sekitar 70% sampai 80% dari jarak (nilai) normal sensing.

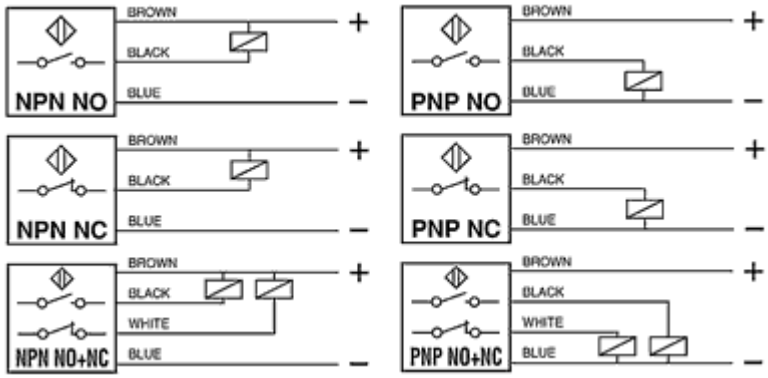



Nilai output dari Proximity Switch ini ada 3 macam, dan bisa diklasifikasikan juga sebagai nilai NO(Normally Open) dan NC (Normally Close). Persis seperti fungsi pada tombol, atau secara spesifik menyerupai fungsi limit switch dalam suatu sistem kerja rangkaian yang membutuhkan suatu perangkat pembaca dalam sistem kerja kontinue mesin.

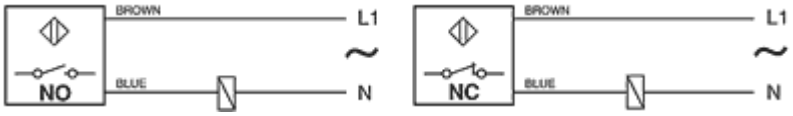
Tiga macam ouput Proximity Switch ini bisa dilihat pada gambar dibawah.




Output 2 kabel VDC




Output 3 dan 4 kabel VDC




Output 2 kabel VAC

Dengan melihat gambar diatas kita dapat mengenali type sensor Proximity Switch ini, yaitu type NPN dan type PNP. Type inilah yang nanti bisa dikoneksikan dengan berbagai macam peralatan kontrol semi digital yang membutuhkan nilai nilai logika sebagai input untuk proses kerjanya.

Beberapa jenis Proximity Switch ini hanya bisa dikoneksikan dengan perangkat PLC tergantung type dan jenisnya. Sensor ini juga bisa dikoneksikan langsung dengan berbagai macam peralatan kontrol semi digital seperti Sensor Controller dan counter relay digital. Pada prinsipnya fungsi Proximity Switch ini dalam suatu rangkaian pengendali adalah sebagai kontrol untuk memati hidupkan suatu sistem interlock dengan bantuan peralatan semi digital untuk sistem kerja berurutan dalam rangkaian kontrol.

## 2.3 Sistem Informasi

Sistem adalah kumpulan dari komponen-komponen peralatan model requirements, function dan interface. Sistem Informasi merupakan kombinasi teratur apapun dari sumber daya manusia, hardware, software, data, dan jaringan. Daur hidup suatu sistem meliputi [7]:

1. Investigasi
2. Analisis
3. Desain
4. Implementasi
5. Pemeliharaan

### 2.3.1 Komponen Sistem Informasi

Perkembangan Sistem informasi dapat digambarkan sebagai sistem yang terdiri dari berbagai komponen. Komponen ini saling berinteraksi satu dengan yang lain membentuk suatu kesatuan untuk mencapai sasaran [7]:

1. Komponen Masukan

Mewakili sejumlah data yang masuk ke dalam sistem informasi. Input tersebut termasuk metode-metode dan media untuk memperoleh data yang akan dimasukkan, yang dapat berupa dokumen-dokumen dasar.

Komponen Proses

Terdiri dari kombinasi prosedur, logika, dan model matematika yang akan memanipulasi data input dan data yang tersimpan di database dengan cara tertentu untuk menghasilkan keluaran (output) yang diinginkan.

2. Komponen Keluaran

Produk dari sistem informasi adalah keluaran yang merupakan informasi dan dokumentasi yang dapat digunakan untuk semua tingkatan manajemen dan semua pemakai sistem.

3. Komponen Teknologi

Teknologi merupakan *tool-box* dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari system secara keseluruhan.

4. Komponen Hardware

Hardware berperan penting sebagai suatu peralatan vital bagi sistem informasi yang digunakan untuk menyimpan dan mengolah informasi .

5. Komponen Software

Software berfungsi sebagai kumpulan perintah komputerisasi yang berguna untuk mengolah, menghitung dan memanipulasi data yang diambil dari hardware untuk menciptakan suatu informasi.

6. Komponen Basis data

Database merupakan kumpulan dari data yang saling berhubungan satu sama lainnya, tersimpan pada perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. Pengelolaan database umumnya dikenal dengan nama DBMS (Database Management System).

7. Komponen Kendali

Bagian pengendalian dirancang dan diterapkan untuk memelihara system dari hal-hal yang dapat merusaknya, seperti faktor-faktor alamiah temperatur, air, api, debu, virus, sabotase/hijacking, dan sebagainya.

## 2.4 User-generated Content

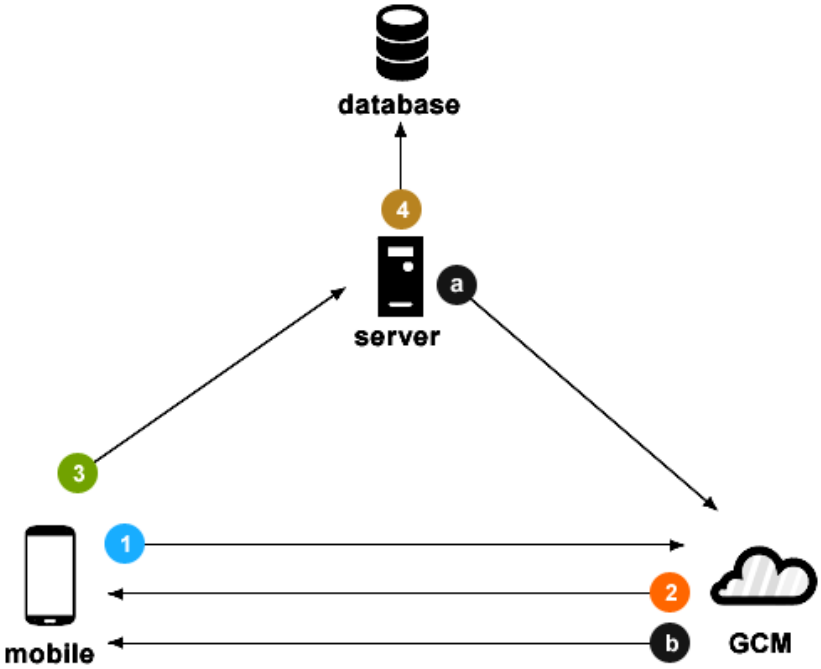
User-generated Content atau disingkat UGC, dikenal juga dengan istilah consumer-generated media (CGM) atau user-created content (UCC) adalah merujuk pada berbagai tipe konten materi atau media yang tersedia untuk umum dan diproduksi oleh end-user atau para non-media profesional. Kemajuan teknologi web memungkinkan konten website tidak lagi dimonopoli oleh profesional media, melainkan dapat dibuat oleh para penggunanya. UGC merupakan salah satu ciri

dominan Web 2.0. Contoh penerapan pada UGC adalah YouTube hampir semua konten yang dimiliki situs tersebut dibuat dan diupload oleh penggunanya [11].

Konsep UGC pada dasarnya telah banyak merubah cara berinteraksi pengguna dengan internet begitu juga dalam media periklanan. Bagi media periklanan jejaring sosial dengan konsep UGC memiliki potensi besar menyediakan market yang lebih terarah dan terpusat bagi mereka [11].

## 2.5 Google Cloud Messaging

Google Cloud Messaging adalah teknologi untuk membantu pengembang mengirim data dari server menuju device android secara langsung melalui server google, pemanfaatan teknologi ini biasanya digunakan untuk *push messaging*. Pengiriman pesan menggunakan teknologi ini mengharuskan adanya API key, key ini terkait langsung dengan akun google yang digunakan untuk mempublikasikan aplikasi melalui google play [9].




Gambar II-3 GCM Model [8]

Berdasarkan ilustrasi dari Gambar II-3 GCM Model terdapat dua server yang terlibat dalam proses *push messaging* , pada setiap server yang terlibat terdapat proses pengiriman data yang berbeda-beda diantaranya [9]:

1. Device android mengirimkan sender id dan application id menuju GCM Server untuk registrasi.
2. Setelah berhasil registrasi, GCM server akan mengirimkan registration id menuju device android.
3. Setelah device android mendapatkan registration id, device tersebut akan mengirimkan registration id menuju server aplikasi.

Server aplikasi akan menyimpan registrasi id yang dikirimkan untuk digunakan sebagai id saat menggunakan *push messaging*.

## 2.6 Aplikasi *Mobile*

Menurut Buyens, aplikasi mobile berasal dari kata *application* dan *mobile*. *Application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju sedangkan *mobile* dapat di artikan sebagai perpindahan dari suatu tempat ke tempat yang lain.

Maka aplikasi *mobile* dapat di artikan sebuah program aplikasi yang dapat dijalankan atau digunakan walaupun pengguna berpindah – pindah dari satu tempat ke tempat yang lain serta mempunyai ukuran yang kecil. Aplikasi *mobile* ini dapat di akses melalui perangkat nirkabel, pager, PDA, telepon seluler, *smartphone*, dan perangkat sejenisnya [11].

### 2.6.1 Karakteristik Perangkat *Mobile*

Perangkat mobile memiliki banyak jenis dalam hal ukuran, desain dan layout, tetapi mereka memiliki karakteristik yang sangat berbeda dari sistem desktop. Berikut karakteristik perangkat mobile, diantaranya :

1. Ukuran yang kecil

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

2. Memory yang terbatas

Perangkat mobile juga memiliki memory yang kecil, yaitu *primary* (RAM) dan *secondary* (*disk*). Pembatasan ini adalah salah satu faktor yang mempengaruhi penulisan program untuk berbagai jenis dari perangkat ini. Dengan pembatasan jumlah dari memory, pertimbangan-pertimbangan khusus harus diambil untuk memelihara pemakaian dari sumber daya yang mahal ini.

3. Daya proses yang terbatas

Sistem mobile tidaklah setangguh rekan mereka yaitu desktop. Ukuran, teknologi dan biaya adalah beberapa faktor yang mempengaruhi status dari sumber daya ini. Seperti harddisk dan RAM, Anda dapat menemukan mereka dalam ukuran yang pas dengan sebuah kemasan kecil.

4. Mengkonsumsi daya yang rendah

Perangkat mobile menghabiskan sedikit daya dibandingkan dengan mesin desktop. Perangkat ini harus menghemat daya karena mereka berjalan pada keadaan dimana daya yang disediakan dibatasi oleh baterai-baterai.

5. Kuat dan dapat diandalkan

Karena perangkat mobile selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapi benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.

6. Konektivitas yang terbatas

Perangkat mobile memiliki bandwith rendah, beberapa dari mereka bahkan tidak tersambung. Kebanyakan dari mereka menggunakan koneksi wireless.

7. Masa hidup yang pendek

Perangkat-perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala. Coba ambil kasus sebuah handphone, mereka booting dalam hitungan detik dan kebanyakan orang tidak mematikan handphone mereka bahkan ketika malam hari. PDA akan menyala jika anda menekan tombol power mereka [11].

## 2.7 Android

Android adalah sistem operasi yang berbasis *Linux* untuk telepon seluler seperti telepon pintar (*Smartphone*) dan komputer tablet. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti begerak. Awalnya, *Google Inc*. membeli *Android Inc.*, pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan *Android*, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan piranti keras, piranti lunak dan telekomunikasi, termasuk *Google, HTC, Intel, Motorola, Qualcomm, T-Mobile* dan *Nvidia.*

Beberapa pengertian lain dari *Android*, yaitu :

1. Merupakan *platform* terbuka (*Open Source*) bagi para programmer untuk membuat atau mengembangkan aplikasi.
2. Merupakan sistem operasi yang dibeli *Google Inc* dari *Android Inc*.

Bukan bahasa pemrograman, akan tetapi hanya menyediakan lingkungan hidup atau *runtime environment* yang disebut DVM (*Dalvik Virtual Machine*) yang telah dioptimasi untuk *device* dengan sistem memori yang kecil [12].

### 2.7.1 Sejarah Android

Pada Juli 2000, *Google* bekerjasama dengan *Android Inc*., perusahaan yang berada di Palo Alto, California, Amerika Serikat. Para pendiri *Android Inc*. bekerja pada *Google*, diantaranya Andy Rubin, Rich Miner, Nick Sears dan Chris White. Saat itu banyak yang menganggap fungsi *Android Inc*. hanyalah sebagai perangkat lunak pada telepon seluler. Sejak saat itu muncul rumor bahwa *Google* hendak memasuki pasar telepon seluler. Di perusahaan *Google*, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yang didukung oleh *Kernel Linux*. Hal ini menunjukan indikasi bahwa *Google* sedang bersiap menghadapi persaingan dalam pasara telepon seluler. Versi *android* terbaru yaitu versi 5.0. (*KitKat*). *Android* juga sudah bergabung dengan beberapa *smart mobile* seperti *LG, Samsung, Sony Ericsson, HTC, Motorola* dan lainnya.

Sekitar September 2007 sebuah studi melaporkan bahwa *Google* mengajukan hak paten aplikasi telepon seluler (akhirnya *Google* mengenalkan *Nexus One,* salah satu jenis telepon pintar GSM yang menggunakan *Android* pada sistem operasinya. Telepon seluler ini diproduksi oleh *HTC Corporotaion* dan tersedia di pasaran pada 5 Januari 2010).

Pada Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja *Android ARM Holdings, Atheros Communications,* diproduksi oleh *Asustek Computer Inc. Garmin Ltd, Softbank, Sony Ericsson Toshiba Corp* dan *Vodafone Group Plc.* Seiring pembentukan *Open Handset Alliance,* OHA mengumumkan produk perdana mereka, *Android,* perangkat bergerak (*Mobile*) yang merupakan modifikasi *kernel Linux* 2.6. sejak *Android* dirilis telah dilakukan berbagai pembaruan berupa perbaikan *bug* dan penmbahan fitur baru [12].

### 2.7.2 Life cycle Android

Aplikasi android terdiri dari beberapa fungsi dasar seperti mengedit catatan, memutar file musik, membunyikan alarm, atau membuka kontak telepon. Fungsi-fungsi tersebut dapat diklasifikasikan ke dalam empat komponen android yang berbeda seperti ditunjukkan pada , klasifikasi tersebut berdasarkan kelas-kelas dasar java yang digunakan [9].

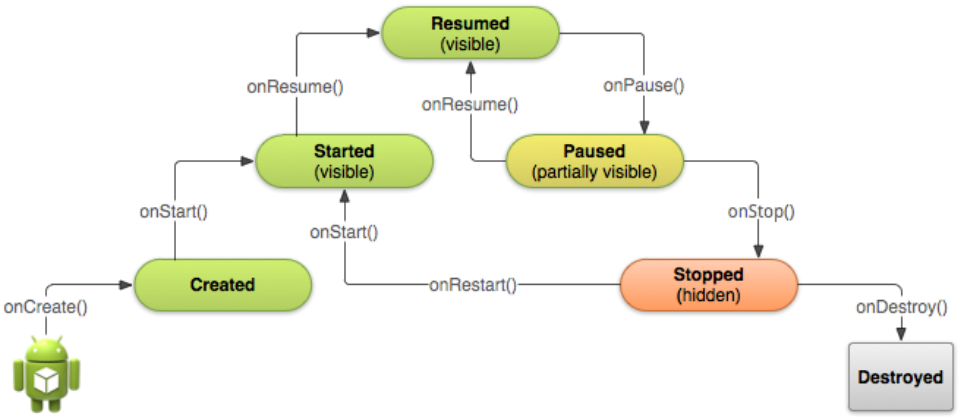
Tabel II.1 Komponen Aplikasi Android

| Functionality | Java Base Class | Examples |
|---|---|---|
| Focused thing a user can do | Activity | Edit a note, play a game |
| Background process | Service | Play music, update weather icon |
| Receive messages | BroadcastReceiver | Trigger alarm upon event |
| Store and retrieve data | ContentProvider | Open a phone contact |

Setiap aplikasi pasti menggunakan minimal satu dari komponen tersebut, akan tetapi terdapat beberapa komponen yang mengharuskan mencantumkan

*specified permission* sebelum digunakan seperti komponen *Service*, *BroadcastReceiver*. *ContentProvider* [9].

Android memiliki paradigma pemrograman lain tidak seperti paradigma pemrograman biasa di mana aplikasi yang dijalankan pada fungsi main(), sistem android menjalankan kode dalam method Activity dengan menerapkan metode callback tertentu yang sesuai dengan tahap tertentu dari siklus hidup. Setiap aplikasi yang berjalan dalam sistem operasi android memiliki siklus hidup yang berbeda dengan aplikasi desktop atau web, Hal ini dikarenakan aplikasi mobile memiliki tingkat interupsi proses yang lumayan tinggi seperti ketika *handling* panggilan masuk aplikasi diharuskan menghentikan proses sementara, Penerapan siklus hidup juga berguna untuk memastikan aplikasi tidak menghabiskan sumber daya baterai pengguna [8].




Gambar II- 4 Siklus Hidup Android [8]

Terdapat beberapa *state* dalam siklus hidup android yang terjadi seperti diilustrasikan pada Gambar II- 4 Siklus Hidup Android, akan tetapi hanya beberapa dari state tersebut yang menjadi statis diantaranya:

**1. Resumed**

Resumed terjadi ketika aplikasi berjalan setelah state *paused* . State ini akan menjalankan perintah program yang ditulis pada method onResume() [8].

**2. Paused**

Dalam keadaan ini aktivitas yang terjadi dihentikan secara sementara tetapi masih terlihat oleh pengguna karena terdapat proses yang memiliki prioritas lebih tinggi seperti panggilan telepon. Aplikasi tidak dapat menjalankan perintah apapun ataupun menampilkan apapun dalam state ini [8].

**3. Stopped**

Dalam keadaan ini, aplikasi benar-benar tidak ditampilkan dan tidak terlihat oleh pengguna tetapi masih meninggalkan service dibackground [8].

State lain seperti Created dan Started bersifat sementara dan sistem dengan cepat menjalankan state berikutnya dengan memanggil metode *life cycle callback* berikutnya. Artinya, setelah sistem OnCreate() dipanggil, dengan cepat sistem akan memanggil method OnStart(), krmudian diikuti oleh onResume() [9].

### 2.7.3 Fitur

Android memiliki beberapa fitur utama yang sering digunakan dalam proses pembangunan aplikasi diantaranya adalah:

1. Multi-proses dan App Widgets

Sistem operasi android tidak melarang prosesor menjalankan lebih dari satu aplikasi dalam satu waktu. Sistem operasi android dapat mengatur aplikasi dan thread yang berjalan secara *multitasking*. Keuntungan yang didapat adalah ketika aplikasi berjalan dan berinteraksi dengan pengguna di layer depan sistem operasi, proses dari aplikasi lain dapat berjalan untuk melakukan pembaruan informasi. Sebagai contoh misalnya ketika pengguna memainkan game, proses lain dapat berjalan di belakang aplikasi seperti memeriksa harga saham dan memunculkan peringatan [8].
App Widgets adalah mini aplikasi yang dapat *embedded* dalam aplikasi seperti home screen. App widgets dapat menjalankan proses request seperti musik streaming atau mendeteksi suhu ruangan secara *background* [8].
*Multi-proses* dapat memberikan manfaat berupa *user experience* yang lebih banyak, namun penggunaan fitur tersebut dapat menghabiskan banyak energi baterai jika penggunaan tidak benar [9].

2. Touch Gestures dan Multi-touch

*Touchscreen* adalah *user interface* intuitif yang digunakan banyak *smartphone* di dunia. Dengan fitur ini interaksi dapat dibuat lebih mudah karena cukup dengan menggunakan jari tangan. *Multi-touch* adalah kemampuan yang dapat melakukan tracking lebih dari satu tangan dalam satu waktu, Fitur ini sering digunakan untuk interaksi memperbesar atau memutar objek. Selain itu pengembang dapat membuat interaksi baru dengan memanfaatkan fitur tersebut [9].

3. Hard dan Soft Keyboard

Salah satu fitur pada perangkat smartphone adalah tombol fisik dan non fisik, tombol fisik digunakan untuk navigasi pendukung dalam pengoperasian android.. Pengembang aplikasi tidak perlu secara manual untuk mengintegrasikan tombol tersebut dalam aplikasi. Tombol non fisik adalah tombol yang dibuat oleh sistem operasi seperti keyboard virtual, dan tombol navigasi aplikasi [9].

### 2.7.4 Prinsip Desain

Android memiliki beberapa prinsip desain yang dapat menjadi acuan dalam membuat desain aplikasi android diantaranya adalah

1. Multiple Assets

Android mendukung jutaan smartphone, tablet dan perangkat lain dalam berbagai ukuran layar dan ukuran, untuk itu *Multiple Assets* sangat disarankan digunakan untuk mengatasi fragmentasi pada android. Seperti ilustrasi pada Gambar II- 5 Klasifikasi Ukuran Ikon, android menciptakan beberapa klasifikasi ukuran icon yaitu MDPI, HDPI, XHDPI, XXHDPI dan XXXHDPI. MDPI dan HDPI dikhususkan untuk icon yang akan digunakan pada device berukuran smartphone sedangkan untuk XHDPI, XXHDPI dan XXXHDPI digunakan pada device berukuran tablet [10].




Gambar II- 5 Klasifikasi Ukuran Ikon

2. Touch Feedback

Touch Feedback dalam android digunakan sebagai respon setiap objek yang ditekan pengguna. Hal ini bertujuan untuk memberi tahu pengguna objek mana yang berinteraksi dengan pengguna [10].




Gambar II- 6 Touch Feedback

3. Pattern Gesture

Touch gesture berguna untuk meningkatkan experience pengguna dalam menggunakan aplikasi. Terdapat beberapa gesture yang didukung oleh android diataranya adalah [10]:

1. Touch

Touch adalah gesture default sebagai event untuk item tertentu yang paling sering digunakan dalam perangkat android, hampir seluruh kontrol utama aplikasi ini menggunakan touch.

2. Long Press

Biasanya digunakan untuk seleksi data , dengan gesture ini dimungkinkan untuk memilih satu atau lebih item dalam sebuah tampilan dan menjalankan suatu fungsionalitas tertentu [10].

3. Swipe Or Drag

Swipe adalah menyentuh sebuah titik pada layar dan menggerakkan jari yang tetap tersentuh pada layar ke titik lain pada layar. Swipe biasanya dilakukan untuk kebutuhan berpindah layar atau tergantung dari aplikasi. Swipe dapat dilakukan dari dan ke arah mana saja [10].

4. Double Touch

Pada Smartphone dan tablet android, melakukan dua kali tapping secara berturut-turut pada satu obyek, fungsinya berbeda dengan double klik mouse komputer. Pada Android, teknik ini biasanya dipakai untuk melakukan zoom in dan zoom out atau memperbesar / memperkecil sebuah obyek gambar [10].

5. Pinch Open

Teknik lain yang biasa digunakan adalah dengan menggunakan dua jari, di mana kedua jari tersebut menyentuh dua titik pada layar yang terpisah di mana ujung dari dua jari tersebut tidak bersentuhan, kemudian kedua jari tersebut sambil tetap menyentuh layar bergerak saling mendekati. Teknik ini digunakan untuk membuka aplikasi tertentu [10].

6. Pinch Close

Teknik Pinch Close adalah kebalikan dari Pinch Open di mana spread dilakukan dengan berawal dari dua jari yang bersentuhan pada ujungnya dan ditempelkan sebuah titik yang sama pada layar, kemudian kedua jari tersebut bergerak memisahkan atau menjauhi satu sama lain Gerakan ini untuk menutup layar [10].

### 2.7.5 Versi Android

Banyak *smartphone* dan PC Tablet menggunakan sistem operasi dengan versi yang berbeda. Semakin tinggi versi, fiturnya semakin canggih dan banyak. Telepon pertama yang memakai sistem operasi *Android* adalah HTC Dream yang dirilis pada tanggal 22 Oktober 2008. Beberapa uraian versi *android* seperti dibawah ini :

**1. Android versi 1.1**

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

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

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

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

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 pada kamera, *camcorder* dan galeri yang diintegrasikan pada CDMA / EVDO, 802.1x, VPN, *Gestures*, dan *Text-to-spech engine*. Kemampuan dial kontak teknologi *text to change speech* (tidak tersedia pada semua ponsel).

**4. Android versi 2.0/2.1 (*Eclair*)**

Pada 3 Desember 2009 kembali diluncurkan ponsel *Android* dengan versi 2.0/2.1 (*Eclair*), perubahan yang dilakukan adalah mengoptimalkan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML 5, daftar kontak yang baru, dukungan *flash* untuk kamera 3.2 MP, *digital zoom*, dan *Bluetooth* 2.1. Untuk bergerak cepat dalam persaingan perangkat

generasi berikut, Google melakukan investasi dengan mengadakan kompetisi aplikasi *mobile* terbaik (*killer apps*-aplikasi unggulan). Kompetisi ini berhadiah $25,000 bagi setiap pengembang aplikasi terpilih. Kompetisi diadakan selama dua tahap yang tiap tahapnya dipilih 50 aplikasi terbaik. Dengan semakin berkembangnya dan semakin bertambahnya jumlah *handset Android*, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi *Android*. Aplikasi terkenal yang diubah ke dalam sistem operasi *Android* adalah Shazam, *Backgrounds* dan *WeatherBug*. Sistem operasi *Android* dalam situs internet juga dianggap penting untuk menciptakan aplikasi *Android* asli, contohnya oleh *MySpace* dan *Facebook*.

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

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

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

Pada 6 Desember 2010, Android versi 2.3 (*Gingerbread*) diluncurkan. Perubahan-perubahan umum yang didapat dari versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antarmuka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM. efek audio baru (*reverb, equalization, headphone virtualization,* dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC) dan dukungan jumlah kamera yang lebih dari satu.

**7. Android versi 3.1 (*Honeycomb*)**

*Android Honeycomb* dirancang khusus untuk tablet. *Android* versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada *Honeycomb* juga berbeda karena sudah didesain untuk tablet. *Honeycomb* juga mendukung multi prosesor dan juga akselerasi perangkat keras (*hardware*) untuk grafis. Tablet peratam yang dibuat dengan menjalankan *Honeycomb* adalah Motorola Xoom.

**8. Android versi 4.0 (*Ice Cream Sandwich*)**

Diumumkan pada tanggal 19 Oktober 2011, membawa fitur *Honeycomb* untuk *smartphone* dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara *offline* dan berbagi informasi dengan menggunakan NFC.

**9. Android versi 4.1 (*Jelly Bean*)**

Pada tanggal 27 Juni 2012 Google mengumumkan *Android* versi 4.1 (*Jelly Bean*) dalam konferensi Google I/O berdasarkan kernel Linux 3.0.31. *Jelly Bean* adalah pembaruan penting yang bertujuan untuk meningkatkan fungsi dan kinerja antarmuak pengguna (*User Interface*). Pembaruan ini diwujudkan dalam proyek *Butter*, perbaikan ini termasuk antisipasi sentuh, *triple buffering*, perpanjangan waktu vsync dan peningkatan *frame rate* hingga 60fps untuk menciptakan UI yang lebih halus. Perangkat pertama yang menggunakan sistem operasi ini adalah tablet Nexus 7, yang dirilis pada 13 Juli 2012.

**10. Android versi 4.4 (KitKat)**

*Android* versi 4.4 (KitKat) direncanakan akan dirilis pada bulan Oktober 2013. Setelah sebelumnya beredar rumor bahwa Android versi berikutnya setelah *Jelly Bean* diperkirakan akan diberi nomor 5.0 dan dinamai *Key Lime Pe* [12].

### 2.7.6 Komponen Aplikasi Android

Fitur penting Android adalah bahwa satu aplikasi dapat menggunakan elemen dari aplikasi lain (untuk aplikasi yang memungkinkan). Sebagai contoh, sebuah aplikasi memerlukan fitur *scroller* dan aplikasi lain telah mengembangkan fitur *scroller* yang baik dan memungkinkan aplikasi lain menggunakannya. Maka pengembang tidak perlu lagi mengembangkan hal serupa untuk aplikasinya, cukup menggunakan *scroller* yang telah ada. Agar fitur tersebut dapat bekerja, sistem harus dapat menjalankan aplikasi ketika setiap bagian aplikasi itu dibutuhkan, dan pemanggilan objek java untuk bagian itu. Oleh karenanya Android berbeda dari sistem-sistem lain, Android tidak memiliki satu tampilan utama program seperti fungsi main() pada aplikasi lain. Sebaliknya, aplikasi memiliki komponen penting yang memungkinkan sistem untuk memanggil dan menjalankan ketika dibutuhkan [12].

#### 2.7.6.1 Activities

*Activity* merupakan bagian yang paling penting dalam sebuah aplikasi, karena *Activity* menyajikan tampilan visual program yang sedang digunakan oleh pengguna. Setiap *Activity* dideklarasikan dalam sebuah kelas yang bertugas untuk menampilkan antarmuka pengguna yang terdiri dari *Views* dan respon terhadap *Event*. Setiap aplikasi memiliki sebuah *activity* atau lebih. Biasanya pasti akan ada *activity* yang pertama kali tampil ketika aplikasi dijalankan.

Perpindahan antara *activity* dengan *activity* lainnya diatur melalui sistem, dengan memanfaatkan *activity stack*. Keadaan suatu *activity* ditentukan oleh posisinya dalam tumpukan *acitivity*, LIFO (*Last In First Out*) dari semua aplikasi yang sedang berjalan. Bila suatu *activity* baru dimulai, *activity* yang sebelumnya digunakan maka akan dipindahkan ketumpukan paling atas. Jika pengguna ingin menggunakan *activity* sebelumnya, cukup menekan tombol *Back*, atau menutup *activity* yang sedang digunakan, maka *activity* yang berada diatas akan aktif kembali. *Memory Manager* Android menggunakan tumpukkan ini untuk

menentukan prioritas aplikasi berdasarkan *activity*, memutuskan untuk mengakhiri suatu aplikasi dan mengambil sumber daya dari aplikasi tersebut [12].

Ketika *activity* diambil dan disimpan dalam tumpukkan *activity* terdapat 4 kemungkinan kondisi transisi yang akan terjadi :

**1. *Active***

Setiap *activity* yang berada ditumpukan paling atas, maka dia akan terlihat, terfokus, dan menerima masukkan dari pengguna. Android akan berusaha untuk membuat *activity* aplikasi ini untuk untuk tetap hidup dengan segala cara, bahkan akan menghentikan *activity* yang berada dibawah tumpukkannya jika diperlukan. Ketika *activity* sedang aktif, maka yang lainnya akan dihentikan sementara [12].

**2. *Paused***

Dalam beberapa kasus *activity* akan terlihat tapi tidak terfokus pada kondisi inilah disebut *paused*. Keadaan ini terjadi jika *activity* transparan dan tidak *fullscreen* pada layar. Ketika *activity* dalam keadaan *paused*, dia terlihat *active* namun tidak dapat menerima masukkan dari pengguna. Dalam kasus ekstrim, Android akan menghentikan *activity* dalam keadaan *paused* ini, untuk menunjang sumber daya bagi *activity* yang sedang aktif [12].

**3. *Stopped***

Ketika sebuah *activity* tidak terlihat, maka itulah yang disebut *stopped*. *Activity* akan tetap berada dalam memori dengan semua keadaan dan informasi yang ada. Namun akan menjadi kandidat utama untuk dieksekusi oleh sistem ketika membutuhkan sumberdaya lebih. Oleh karenanya ketika suatu *activity* dalam kondisi *stopped* maka perlu disimpan data dan kondisi antarmuka saat itu. Karena ketika *activity* telah keluar atau ditutup, maka dia akan menjadi *inactive*.

**4. *Inactive***

Kondisi ketika *activity* telah dihentikan dan sebelum dijalankan. *Inactive activity* telah ditiadakan dari tumpukan *activity* sehingga perlu

*restart* ulang agar dapat tampil dan digunakan kembali. Kondisi transisi ini sepenuhnya ditangani oleh manajer memori Android. Android akan memulai menutup aplikasi yang mengandung *activity inactive*, kemudian *stopped activity*, dan dalam kass luar biasa *paused activity* juga akan ditutup [12].

#### 2.7.6.2 Services

Suatu *service* tidak memiliki tampilan antarmuka, melainkan berjalan di *background* untuk waktu yang tidak terbatas. Komponen *service* diproses tidak terlihat, memperbarui sumber data dan menampilkan notifikasi. *Service* digunakan untuk melakukan pengolahan data yang perlu terus diproses, bahkan ketika *Activity* tidak aktif atau tidak tampak [12].

#### 2.7.6.3 Intents

*Intents* merupakan sebuah mekanisme untuk menggambarkan tindakan tertentu, seperti memilih foto, menampilkan halaman *web*, dan lain sebagainya. *Intents* tidak selalu dimulai dengan menjalankan aplikasi, namun juga digunakan oleh sistem untuk memberitahukan ke aplikasi bila terjadi suatu hal, misal pesan masuk. *Intents* dapat eksplisit atau implisit, contohnya jika suatu aplikasi ingin menampilkan *URL*, sistem akan menentukan komponen apa yang dibutuhkan oleh *Intents* tersebut [12].

#### 2.7.6.4 Broadcast Receiver

*Broadcast Receivers* merupakan komponen yang sebenarnya tidak melakukan apa-apa kecuali menerima dan bereaksi menyampaikan pemberitahuan. Sebagian besar *Broadcast* berasal dari sistem misalnya, Batre sudah hampir habis, informasi zona waktu telah berubah, atau pengguna telah merubah bahasa *default* pada perangkat. Sama halnya dengan *service*, *Broadcast Receivers* tidak menampilkan antarmuka pengguna. Namun, *Broadcast Receivers* dapat

menggunakan *Notification Manager* untuk memberitahukan sesuatu kepada pengguna [12].

#### 2.7.6.5 Content Providers

*Content Providers* digunakan untuk mengelola dan berbagi *database*. Data dapat disimpan dalam file sistem, dalam database *SQLite*, atau dengan cara lain yang pada prinsipnya sama. Dengan adanya *Content Provider* memungkinkan antar aplikasi untuk saling berbagi data. Komponen ini sangat berguna ketika sebuah aplikasi membutuhkan data dari aplikasi lain, sehingga mudah dalam penerapannya [12].

### 2.7.7 Tipe Aplikasi Android

Terdapat tiga kategori aplikasi pada Android :

**1. *Foreground Activity***

Aplikasi yang hanya dapat dijalankan jika tampil pada layar dan tetap efektif walaupun tidak terlihat. Aplikasi dengan tipe ini mempertimbangkan siklus hidup *activity*, sehingga perpindahan antar *activity* dapat berlangsung dengan lancar [12].

**2. *Background Service***

Aplikasi yang memiliki interaksi terbatas dengan user, selain dari pengaturan konfigurasi, semua dari prosesnya tidak tidak tampak pada layar. Contohnya aplikasi penyaringan panggilan atau sms *auto respon* [12].

**3. *Intermittent Activity***

Aplikasi yang masih membutuhkan beberapa masukkan dari pengguna, namun sebagian sangat efektif jika dijalankan di *background* dan jika diperlukan akan memberi tahu pengguna tentang kondisi tertentu. Contohnya pemutar musik. Untuk aplikasi yang kompleks akan sulit untuk menentukan kategori aplikasi tersebut apalagi aplikasi memiliki ciri-ciri dari semua kategori. Oleh karenanya perlu

pertimbangan bagaimana aplikasi tersebut digunakan dan menentukan kategori aplikasi yang sesuai [12].

### 2.7.8 Kelebihan Android

Sudah banyak *platform* untuk perangkat selular saat ini, termasuk didalamnya *Symbian*, *iPhone*, *Windows Mobile*, *BlackBerry*, *Java Mobile Edition*, *Linux Mobile* (LiM), dan banyak lagi. Namun ada beberapa hal yang menjadi kelebihan Android. Walaupun beberapa fitur-fitur yang ada telah muncul sebelumnya pada *platform* lain, Android adalah yang pertama menggabungkan hal seperti berikut :

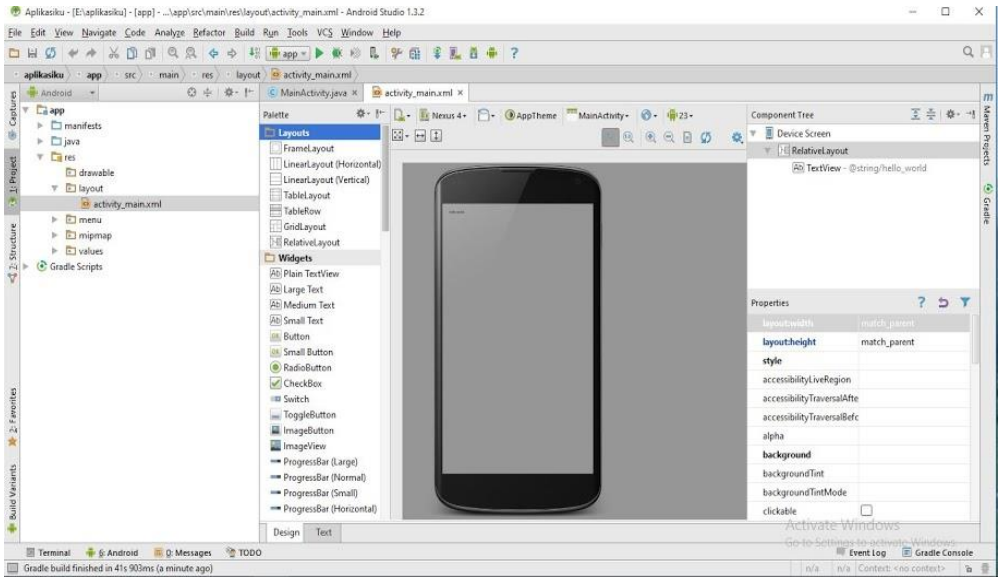
1. Keterbukaan, Bebas pengembangan tanpa dikenakan biaya terhadap sistem karena berbasiskan *Linux* dan *open source*. Pembuat perangkat menyukai hal ini karena dapat membangun *platform* yang sesuai yang diinginkan tanpa harus membayar *royality*. Sementara pengembang *software* menyukai karena Android dapat digunakan diperangkat manapun dan tanpa terikat oleh *vendor* manapun.
2. Arsitektur komponen dasar Android terinspirasi dari teknologi *internet Mashup*. Bagian dalam sebuah aplikasi dapat digunakan oleh aplikasi lainnya, bahkan dapat diganti dengan komponen lain yang sesuai dengan aplikasi yang dikembangkan.
3. Banyak dukungan *service*, kemudahan dalam menggunakan berbagai macam layanan pada aplikasi seperti penggunaan layanan pencarian lokasi, *database* SQL, *browser* dan penggunaan peta. Semua itu sudah tertanam pada Android sehingga memudahkan dalam pengembangan aplikasi.
4. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga kerja sistem menjadi lebih stabil. Pengguna tak perlu khawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.

5. Dukungan grafis dan suara terbaik, dengan adanya dukungan 2D grafis dan animasi yang diilhami oleh Flash menyatu dalam 3D menggunakan *OpenGL* memungkinkan membuat aplikasi maupun *game* yang berbeda.
   *Portabilitas* aplikasi, aplikasi dapat digunakan pada perangkat yang ada saat ini maupun yang akan datang. Semua program ditulis dengan menggunakan bahas pemrograman Java dan dieksekusi oleh mesin *virtual Dalvik*, sehingga kode program *portable* antara ARM, X86, dan arsitektur lainnya. Sama halnya dengan dukungan masukan seperti penggunaan *Keyboard*, layar sentuh, *trackball* dan resolusi lyr semua dapat disesuaikan dengan program [12].

## 2.8 Android Studio

Android studio adalah lingkungan pengembangan terpadu (*IDE)* resmi untuk pengembangan platform android, hal itu disampaikan pada tanggal 16 mei 2013 Google I/O,android studio sudah tersedia secara bebas dibawah lisensi Apache 2.0.

Android studio pada awalnya tahap *preview* versi 0.1 yang dipakai pada tanggal 1 mei 2013 dan memasuki tahap beta pada bulan juni 2014 dan mulai stabil dirilis pada Desember 2014 dengan versi 1.0 ,berdasarkan *jetBrains* ‘IDEA *Intellij Software*, android studio dirancang khusus untuk pengembangan android yang tersedia untuk Windows,Mac OS X dan Linux sebagai pengganti *Eclipse* [13].



Sumber Gambar : Xavier Ducrohet [13]

Gambar II- 7 Layout Android Studio

### 2.8.1 Struktur Android Studio

Berikut rincian dari struktur Android Studio :

1. App : berfungsi untuk menyimpan file file pendukung aplikasi yang akan diolah.

   *A. Manifest*

   tempat dimana konfigurasi konfigurasi pendukung project serta izin (permision) untuk mengakses hardware.

   B. Java

   tempat berdiamnya *code/scripts* java untuk menjalankan aplikasi yang akan dibuat

   C. *Res* :

   1. *Drawable* : tempat untuk menyimpan gambar atau *icon* pendukung aplikasi.
   2. *Layout* : tempat berkumpulnya design tampilan yang akan tampak pada tampilan aplikasi anda.

3. *Menu* : berisi *code* berupa *file* xml yang didalamnya berisi *array* tentang menu yang akan tampil dalam *project* anda, misalkan *file*, *new*,*open*,*close*,dsb.
4. *Mipmap* : tempat berkumpulnya *icon launcher* (icon utama aplikasi) yang ukurannya menyesuaikan *device* yang digunakan.
5. *Values* : untuk menyimpan file .xml yang menggambarkan sumber dari berbagai jenis. Sebagai contoh folder berisi *file strings*.xml dengan definisi sumber *string*. Seperti yang akan anda lihat dari menambahkan bagian *color , folder* tata letak juga dapat berisi, misalnya, deskripsi warna. Di dalam file dimens.xml, terdapat *string-string* untuk konfigurasi dimensi aplikasi. Tidak jauh berbeda dengan dimens.xml maupun *strings*.xml, pada file styles.xml terdapat deklarasi variable untuk theme.

2. *Gradle scripts* : berfungsi sebagai *configurasi* aplikasi anda dengan *device,* serta menghubungkan *library* pendukung aplikasi yang anda buat. Penggunaan android studio ini untuk membuat aplikasi rumah pintar di *smartphone* android yang sebagai perangkat bantuan untuk mengontrol rumah pintar [13].

## 2.9 Database

Beberapa definisi basis data sebagai berikut :

1. Database adalah sekumpulan data store (bias dalam jumlah yang besar) yang tersimpan dalam *magnetic disk, optical disk, magnetic drum* atau media penyimpanan.
2. Database adalah sekumpulan program-program aplikasi umum yang bersifat "*batch*" yang mengeksekusi dan memproses data secara umum (seperti pencarian,peremajaan,penambahan, dan penghapusan terhadap data)
3. Database terdiri dari data yang akan digunakan atau diperuntukan terhadap banyak user (baik menggunakan teknik pemrosesan yang bersifat batch atau on-line) akan menggunakan data tersebut sesuai dengan tudas dan fungsinya, dan user lain dapat juga menggunakan data tersebut dalam waktu yang bersamaan.

### 2.9.1 SQL

SQL adalah sebuah bahasa permintaan database yang terstruktur. Bahasa SQL dibuat sebagai bahasa yang dapat merelasikan beberapa tabel dalam database maupun merelasikan antar database. Bahasa SQL ditulis langsung dalam sebuah program databases sehingga pengguna dapat melihat langsung permintaan yang diinginkan sekaligus melihat hasilnya. SQL tidak hanya mengambil informasi dari tabel-tabel *database* SQL dapat digunakan dalam membuat elemen dalam *database*, memasukan, mengubah dan menghapus data dari *database*. Dengan memanfaatkan bahasa web seperti HTML, dan PHP, SQL merupakan tool penting yang tidak dapat ditinggalkan untuk pengembangan aplikasi database berbasis web. Query dasar yang sering digunakan adalah *create, read, update* dan *delete*.

1. *Create*

Digunakan untuk menginputkan data kesebuah tabel di database, contoh querynya sebagai berikut:

[ INSERT INTO anggota(nama_anggota, umur) VALUES(‘Asep’, 21) ]

2. *Read*

*Read* digunakan untuk menampilakan data sesuai dengan ketentuan tertentu, contoh querynya sebagai berikut :

[ SELECT * FROM anggota ]

3. *Update*

*Update* digunakan untuk mengubah data sesuai dengan apa yang diinginkan, contoh querynya adalah sebagai berikut:

[ UPDATE anggota SET umur = ‘21’ WHERE id_anggota = 1 ]

4. *Delete*

Delete digunakan untuk menghapus data yang ingin dihilangkan, contoh querynya sebagai berikut:

[ DELETE FROM anggota WHERE id_anggota =5 ]

#### 2.9.1.1 DDL

DDL atau *Data Definition Language* adalah sebuah metode *query* sql yang berguna untuk mendefinisikan data pada sebuah *database*, adapun *query* yang dimiliki adalah:

1. CREATE : Digunakan untuk melakukan pembuatan tabel dan *database*
   contoh querynya sebagai berikut :

   [ Create : mysql > CREARE DATABASE nama_database;]

2. DROP : Digunakan untuk melakukan penghapusan tabel maupun *database*
   contoh querynya sebagai berikut :

   [ Drop : mysql > DROP DATABASE nama_database; ]

3. ALTER : Digunakan untuk melakukan perubahan struktur tabel yang telah dibuat, baik menambah *field* (*add*), mengganti nama *field* (*change*) ataupun menamakannya kembali (*rename*) serta menghapus (*drop*)
   contoh querynya sebagai berikut :

   [ Alter : mysql > ALTER nama_tabel alter_spesifikasi kondisi; ]

#### 2.9.1.2 DML

DML atau *Data Manipulation Language* adalah sebuah metode *query* yang dapat digunakan apabila DDL telah terjadi, sehingga fungsi dari *query* ini adalah untuk melaukan pemanipulasian *database* yang telah ada atau telah dibuat sebelumnya. Adapun *query* yang termasuk di dalamnya adalah:

1. INSERT : Digunakan untuk melakukan penginputan pada tabel di *database.*
   contoh querynya sebagai berikut :

   [ INSERT : mysql > INSERT INTO nama_tabel VALUES (isi_field); ]

2. UPDATE : Digunakan untuk melakukan pengubahan atau peremajaan terhadap data yang ada pada tabel.
   contoh querynya sebagai berikut :

[ UPDATE : mysql > UPDATE nama_tabel SET isi_field=“data_baru“; ]

3. DELETE : Digunakan untuk melakukan penghapusan data pada tabel. Penghapusan ini dapat dilakukan secara sekaligus (seluruh isi tabel) maupun hanya beberapa *Recordset.*

   contoh querynya sebagai berikut :

   [ DELETE : DELETE nama_tabel WHERE kondisi; ]

#### 2.9.1.3 DCL

DCL atau *Data Control Language* merupakan bahasa yang digunakan untuk mengendalikan pengaksesan data pada *database*, adapun perintah yang dimiliki DCL adalah:

1. GRANT : Untuk memberikan hak akses atau izin oleh *administrator server* kepada pengguna.
2. REVOKE : Untuk menghilangkan atau mencabut hak akses yang telah diberikan kepada *user* oleh *administrator*.

### 2.9.2 MySQL

MySQL adalah sebuah program *database* server yang mampu menerima dan mengirimkan datanya dengan sangat cepat, *multi user* serta menggunakan perintah standar SQL (*Structured Query Laguage*). MySQL merupakan *database* server *multi user* dan *multi threaded* yang tangguh. Dengan memiliki banyak *feature* MySQL bisa bersaing dengan *database* komsersial sekalipun. Tidak mengejutkan MySQL menjadi *database* pilihan untuk banyak pengguna PHP. MySQL termasuk RDBMS (*Relational Database Management System*) itulah sebabnya istilah seperti tabel, baris dan kolom digunakan pada MySQL. Pada MySQL sebuah *database* mengandung satu atau sejumlah tabel. Tabel terdiri dari sejumlah baris dan setiap baris mengandung satu atau beberapa kolom.

MySQL mampu menangani data yang cukup besar. Perusahaan yang mengembangkan MySQL yaitu TcX, mengaku menyimpan data lebih dari 40 *database*, 10.000 tabel dan sekitar 7 juta baris, totalnya kurang lebih 100 *Gigabyte* data. SQL adalah sebuah standar yang digunakan untuk mengakses *database server*. Bahasa ini pada awalnya dikembangkan oleh IBM, namun telah diadopsi dan digunakan sebagai standar industri.

Dengan menggunakan MySQL, proses akses *database* menjadi lebih *userfriendly* dibandingkan dengan menggunakan *dBASE* atau *Clipper* yang masih menggunakan perintah-perintah pemrograman. MySQL merupakan *software database* yang masih yang masih popular di lingkungan *Linux*, kepopuleran ini sangat ditunjang oleh performa *query* dari databasenya yang saat ini bisa dikatakan paling cepat dan jarang bermaslah. MySQL ini juga sudah berjalan pada lingkungan windows [14].

MySQL juga memiliki beberapa kelebihan, antara lain :

1. *Portability*

   MySQL dapat berjalan stabil pada berbagai sistem operasi seperti *windows*, *Linux, FreeBSD, Solaris* dan lain-lain.

2. *Open Source*

   MySQL didistribusikan secara *open source* (gratis), dibawah lisensi GPL sehingga dapat digunakan cuma-Cuma.

3. *Multi User*

   MySQL dapat digunakan oleh beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

4. *Performance Tuning*

   MySQL memiliki kecepatan yang menakjubkan dalam menangani *query* sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.

5. *Coloumn Types*

MySQL memiliki tipe kolom yang sangat kompleks, seperti *integer, double, char, text, date* dan lain-lain.

6. *Command and Function*

MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *select* dan *where* dalam *query*.

7. *Security*

MySQL memiliki beberapa lapisan sekuritas seperti level *subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta password terenkripsi.

8. *Scability and Limits*

MySQL mampu menangani *database* dalam skala besar, dengan jumlah *records* lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas *indeks* yang dapat ditampung mencapai 32 *indeks* pada tiap tabelnya.

9. *Connectivity*

MySQL dapat melakukan koneksi dengan *clients* menggunakan protokol TCP/IP, *Unix socket* (UNIX) atau *Named Pipes* (NT).

10. *Localisation*

MySQL dapat mendeteksi pesan kesalahan pada *client* dengan menggunakan lebih dari dua puluh bahasa. Meskipun demikian, bahasa Indonesia belum termasuk didalamnya.

11. *Interface*

MySQL memiliki *interface* (antar muka) terhadap berbagai aplikasi dan bahasa pemograman dengan menggunakan fungsi API (*Aplication Programming Interface*).

12. *Clients and Tools*

MySQL dilengkapi dengan berbagai tools yang dapat digunakan untuk administrasi *database* dan pada setiap tool yang ada disertakan petunjuk online.

13. Struktur Tabel

MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani *ALTER TABLE*, dibandingkan database lainnya semacam *PosgreeSQL* ataupun *Oracle.*

Kelemahan MySQL dari dulu sampai saat ini adalah *feature-creep* artinya MySQL berusaha kompatibel dengan beberapa standar serta berusaha memenuhinya namun jika itu diungkapkan kenyataannya bahwa fitur-fitur tersebut belum lengkap dan belum berperilaku sesuai standar. Contoh fitur *SUB-SELECT* (nesting *SELECT* dalam *SELECT*) yang tidak optimal dan sering salah parsing *query* SQL dan jalan keluarnya dengan memecah menjadi beberapa query [14].

## 2.10 Javascript

JavaScript adalah bahasa *scripting* yang popular di internet dan dapat bekerja di sebagian besar *browser* popular seperti *Internet Explorer* (IE), *Mozilla FireFox*, *Netscape*, dan *Opera*. Kode JavaScript dapat disisipkan dalam halaman web menggunakan tag SCRIPT.

Keunggulan javascript yang juga dikenal dengan nama ECMAScript yaitu dapat berjalan di semua *platform* dengan *browser* yang mendukung JavaScript, dan hampir semua *platform* dan *browser* saat ini mendukung JavaScript. Contoh dari aplikasi yang dibangun dengan JavaScript adalah *Google Maps* yang dapat berjalan di atas *Linux, Windows,* dan *Mac OS*. JavaScript juga semakin dan terus berkembang, seperti pertumbuhan pustaka (*library*) yang memudahkan untuk menavigasi dokumen, memlilih elemen DOM, membuat animasi, menangani *event* dan mengembangkan aplikasi *Ajax*. JavaScript adalah bahasa pemograman *client-side* yang *cross-platform* (berjalan di banyak *platform*) dan bersifat bebas (untuk dimodifikasi dan gratis) juga diadopsi secara universal [15].

## 2.11 JSON

JSON merupakan bagian dari bahasa pemrograman JavaScript (Standard ECMA-262 3rd *Edition – December* 1999). JSON merupakan format teks yang sepenuhnya independen tetapi menggunakan konvensi yang familiar dengan bahasa pemrograman dari keluarga-C, termasuk C, C++, C#, Java, JavaScript, Perl, Python, dan sebagainya. Kelebihan inilah yang membuat JSON menjadi sebuah bahasa *data-interchange* yang ideal.

JSON dibangun dalam dua struktur:

1 Beberapa pasangan dari nama/nilai. Dalam beberapa bahasa pemrograman biasa disebut dengan istilah *object, record, struct, tabel hash, key list* atau *associative array*.
2 Nilai-nilai yang terusun secara *ordered list*. Biasa disebut dengan *array, vector, list* atau daftar dalam bahasa pemrograman.

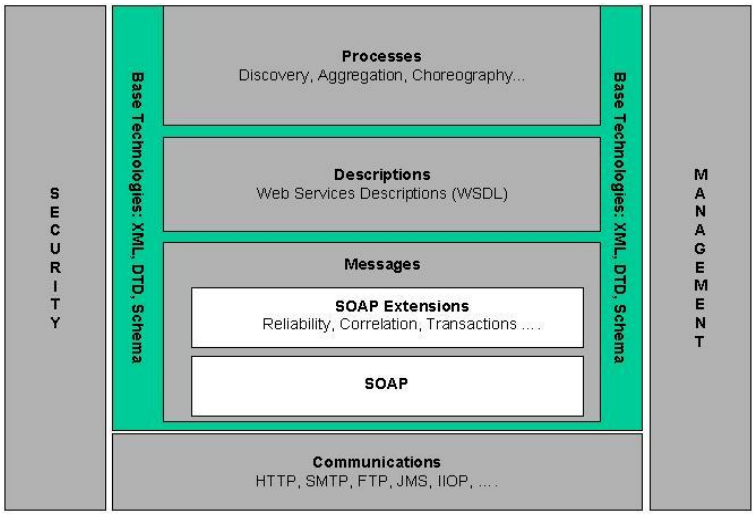
JSON adalah struktur data yang *universal*, dalam artian bisa digunakan dalam berbagai bahasa pemrograman. Hampir semua bahasa pemrograman mendukung penuh JSON dalam berbagai format. Hal ini memungkinkan format data yang dapat dipertukarkan menggunakan bahasa pemrograman juga menggunakan dasar dari struktur JSON.

## 2.12 Web Service

*Web service* adalah sebuah entitas komputasi yang dapat diakses melalui jaringan internet maupun intranet dengan standar protokol tertentu dalam *platform* dan antarmuka bahasa pemograman yang independen. Tujuan pengembangannya adalah untuk menjembatani komunikasi antar program, sehingga aplikasi yang satu dan aplikasi yang lain yang terdapat pada suatu jaringan yang sama atau pada jaringan berbeda dapat saling berkomunikasi asalkan menggunakan standart protokol yang ditetapkan oleh *web service*. Hal ini bisa terjadi, karena standart protokol itu tidaklah terikat pada suatu *platform* atau bahasa pemograman. Protokol itu sendiri dibangun oleh *Extensible Markup Language* (XML) yang memang

kenyataannya telah didukung oleh banyak *paltform*, bahasa pemograman, dan oleh *developer* di seluruh dunia.

Arsitektur *web service* dibangun oleh beberapa layer dan teknologi yang saling berhubungan. Banyak cara untuk memvisualisaskan *service*, sama banyaknya dengan cara untuk membangun dan menggunakan *web service* gambar berikut hanyalah salah satu cara untuk menggambarkan arsitekturnya.




Sumber gambar : Purba, Siregar [16]

Gambar II- 8 Arsitektur Web Service

*Web service* disusun oleh tiga komponen standart, yaitu :

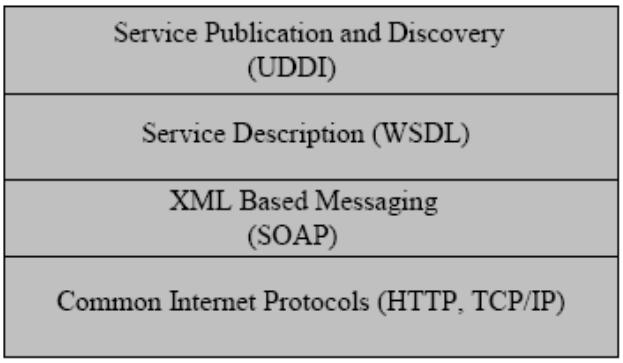
1. *Simple Object Access Protocol* (SOAP), yaitu protokol yang bertanggungjawab dalam pertukuran informasi dalam lingkungan jaringan terdistribusi.
2. *Web service definition Language* (WDSL), dokumen standart yang dituliskan pada format XML, dan mendifinikasan kehadiran web *service* dalam suatu jaringan.
3. *Universal description, Discovery and Inttegration* (UDDI), yaitu suatu lokasi direktori yang beriskan *service* (layanan) dan bersifat bebas *platform* (*platform independent*), dituliskan berbasis XML dan dapat diakses oleh entittas yang berada di dalam dan luar jaringan.

Adanya standart tersebut membuat web *service* mudah di akses melalui berbagai antarmuka dan juga memberi peluang dimungkinkannya berbagai sistem yang dibangun pada platform yang berbeda dan bahasa yang berbeda untuk berkolaborasi pada dalam suatu pekerjaan [16].

### 2.12.1 Komponen-Komponen *Web Service*

*Web service* secara keseluruhan memiliki empat layer komponen yaitu:

1. Layer 1: Protokol internet standar seperti HTTP, TCP/IP
2. Layer 2: *Simple Object Access Protocol* (SOAP), merupakan protokol akses objek berbasis XML yang digunakan untuk proses pertukaran data/informasi antar layanan.
3. Layer 3*: Web Service Definition Language* (WSDL), merupakan suatu standar bahasa dalam format XML yang berfungsi untuk mendeskripsikan seluruh layanan yang tersedia.




Sumber gambar : Purba, Siregar [16]

Gambar II- 9 Komponen-Komponen Web Service

## 2.13 Perancangan Berorientasi Objek

Teknologi objek menganalogikan sistem aplikasi seperti kehidupan nyata yang didominasi oleh objek. Didalam membangun sistem berorientasi objek akan menjadi lebih baik apabila langkah awalnya didahului dengan proses analisis dan perancangan yang berorientasi objek. Tujuannya adalah mempermudah *programmer* didalam mendesain program dalam bentuk objek-objek dan hubungan antar objek tersebut untuk kemudian dimodelkan dalam sistem nyata. Suatu

perusahaan *software* yaitu *Rational Software*, telah membentuk konsarium dengan berbagai organisasi untuk meresmikan pemakaian *Unified Modelling Language* (UML) sebagai bahasa standar dalam *Object Oriented Analysist Design* (OOAD) [17].

### 2.13.1 UML (Unified Modeling Language)

UML merupakan suatu cara untuk menyelesaikan suatu masalah dengan mendeskripsikannya yang telah menjadi standar dalam dunia industri untuk memvisualisasikan, merancang dan mendokumentasikan sistem perangkat lunak. Dengan menggunakan UML kita dapat membangun model untuk segala bentuk dan jenis aplikasi perangkat lunak, yang mana aplikasi yang dibangun dapat berjalan pada perangkat lunak dengan sistem operasi dan jaringan apapun. Selain itu, dengan menggunakan UML, programmer dapat mengerti, memahami dengan jelas maksud, tujuan dan arah rancangan sistem, sehingga sistem perngakat lunak yang dirancang dapat dibangun dengan bahasa program apapun. Namun karena UML merupakan suatu model pengembangan sistem perangkat lunak yang berbasis *object oriented* sehingga menggnuakan bentuk *class* dan *operation* dalam konsep dasarnya, maka lebih cocok dalam pemrogramannya dengan menggunakan bahasa-bahasa pemrograman yang berbasis OOP (*Object Oriented Programming*) seperti C, C++, Java, VB dan lain sebagainya [17].

Secara lebih mendalam, UML lebih dari sekedar sebuah standard dan penemuan dari suatu notasi-notasi yang disatukan, tetapi juga berisi konsep-konsep baru yang menarik yang tidak ditemukan secara umum dalam komunitas *object oriented*. Dalam membangun suatu model perangkat lunak dengan UML, digunakan bentuk-bentuk diagram atau symbol untuk merepresentasikan elemen-elemen dalam sistem. UML ini digunakan untuk perancangan sistem yang nanti akan digunakan pada Bab 3 pada tugas akhir ini. Bentuk diagram yang digunakan untuk merepresentasikannya adalah sebagai berikut:

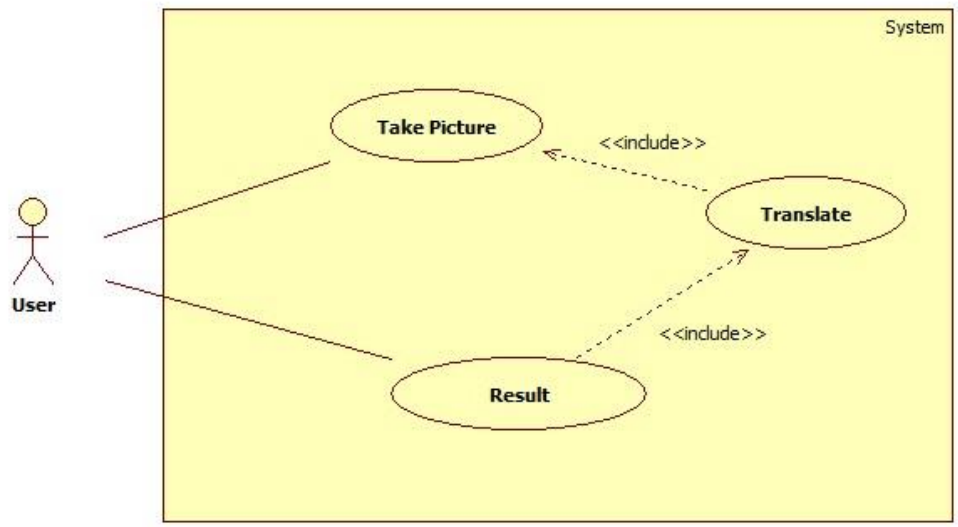
1. *Use-case* Diagram.
2. *Class* Diagram.
3. *Sequence* Diagram.

4. *Collaboration* Diagram.
5. *Activity* Diagram.
6. *Component* Diagram.
7. *Deployment* Diagram.

#### 2.13.1.1 Use-case Diagram

*Use-case diagram* merupakan suatu bentuk diagram yang menggambarkan fungsi-fungsi yang diharapkan dari sebuah sistem yang dirancang. Dalam *Usecase diagram* penekanannya adalah "*apa*" yang diperbuat oleh sistem, dan bukan "*bagaimana*". Sebuah *use-case* akan merepresentasikan sebuah interaksi antara pelaku atau *actor* dengan sistem.

*Use-case diagram* yang digunakan dalam mercancang suatu sistem dapat sangat membantu pada saat menyusun *requirement* sebuah sistem, mengomunikasikannya dengan klien, dan merancang pengujian untuk semua fitur yang terdapat dalam sistem. Dalam suatu sistem aplikasi *database*, *use-case diagram* sangat membantu *requierement* apa saja yang diperlukan [18].




Gambar II- 10 Use Case Diagram

### 2.13.1.2 *Class* Diagram

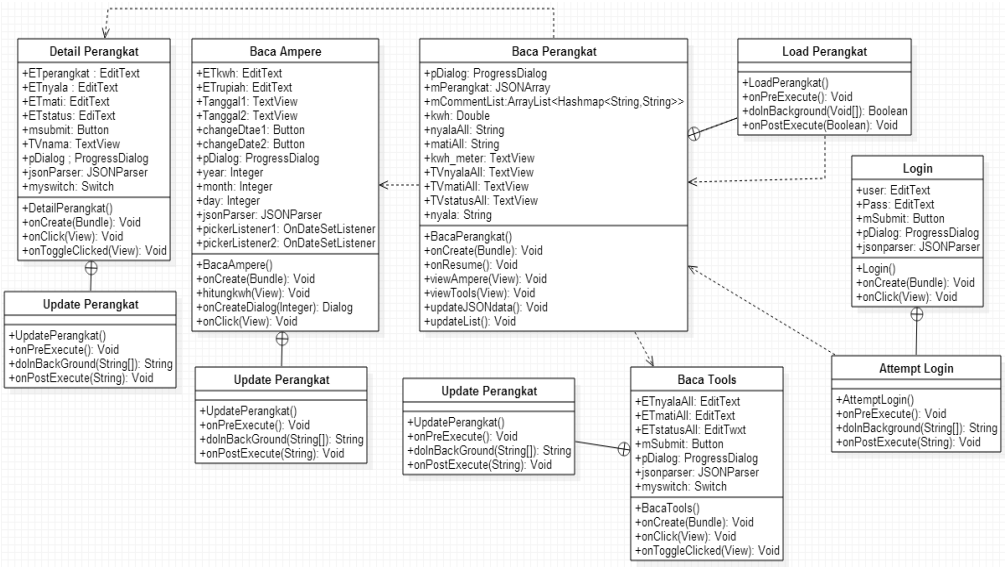
Sebuah *Class Diagram* menunjukkan struktur yang statis dari beberapa *class* dalam suatu sistem. *Class-class* merepresentasikan suatu keadaan (atribut/properti) dan yang akan dikerjakan oleh sistem (metoda/fungsi). *Class* memiliki tiga area pokok:

1. Nama (dan *stereotype*)
2. Atribut
3. Metoda

Atribut dan metoda dalam class diagram dapat memiliki salah satu sifat seperti berikut di bawah ini:

1. *Private*, hanya dapat diakses oleh *class* itu sendiri.
2. *Protected*, hanya dapat diakses oleh *class* itu sendiri dan turunan dari *class* tersebut.
3. *Public*, dapat diakses oleh *class* selain dari *class* yang bersangkutan.

   *Class* dapat direpresentasikan dalam sebuah *interface* atau sebaliknya merupakan implementasi dari sebuah *interface* yang berupa *class* abstrak yang hanya tidak memiliki *attribute* dan hanya memiliki metoda. Berikut merupakan bentuk *class diagram* secara umum.

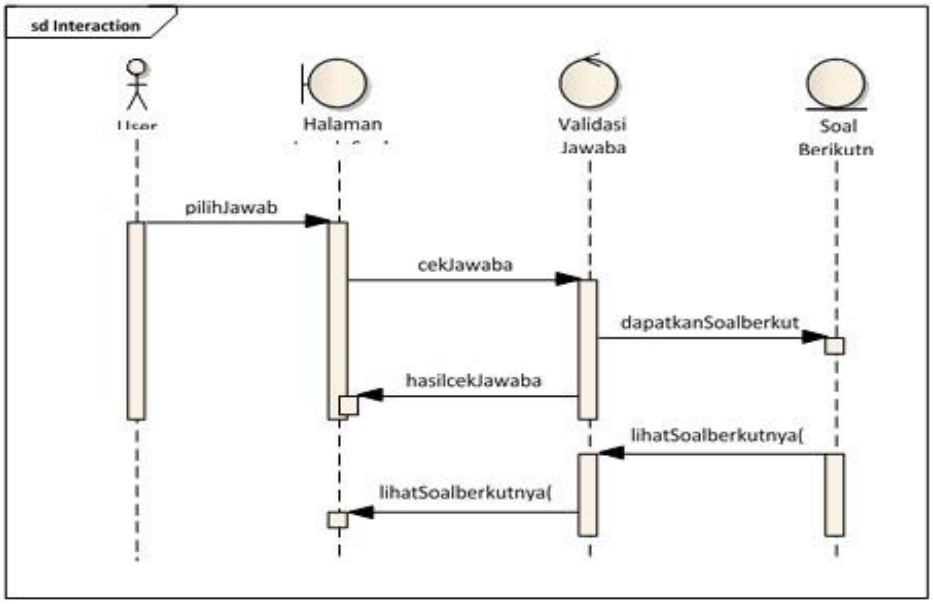



Gambar II- 11 Class Diagram

### 2.13.1.3 Sequence Diagram

*Sequence Diagram* merupakan diagram yang mengambarkan kolaborasi yang dinamis antara obyek satu dengan yang lain. Kolaborasi ini ditunjukkan dengan adanya interaksi antar obyek di dalam dan di sekitar sistem yang berupa pesan atau instruksi yang berurutan.

*Sequence diagram* umumnya digunakan untuk menggambarkan suatu skenario atau urutan langkah-langkah yang dilakukan baik oleh *actor* maupun sistem yang merupakan respon dari sebuah kejadian untuk mendapatkan hasil atau *output*.



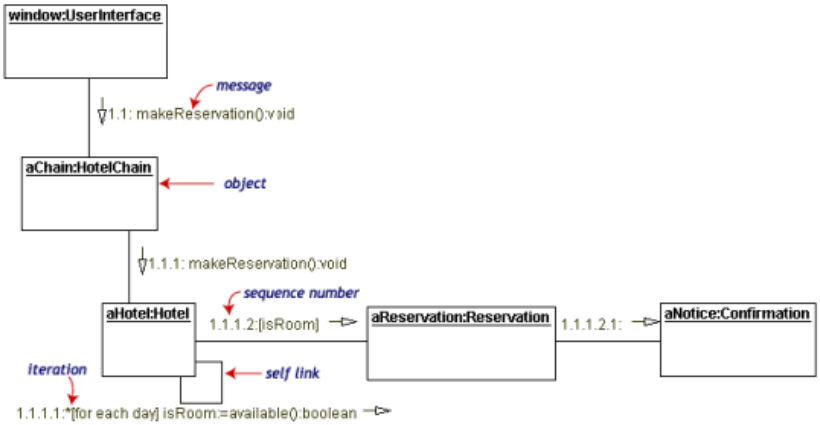

Sumber gambar :Sri Dharwiyanti dan romi Satria Wahono [18]

Gambar II- 12 Sequence Diagram

### 2.13.1.4 Collaboration Diagram

Diagram kolaborasi adalah diagram interaksi yang menekankan organisasi strutkural dari objek-objek yangmenerima serta mengirim pesan (*message*).Sebuah *collaboration diagram* menunjukkan kolaborasi yang dinamis yang mirip dengan *sequence diagram*. *Collaboration diagram* digambarkan sebagai sebuah *obiject*

diagram dimana sejumlah obyek ditunjukkan disekitarnya dengan hubungan-hubungannya.

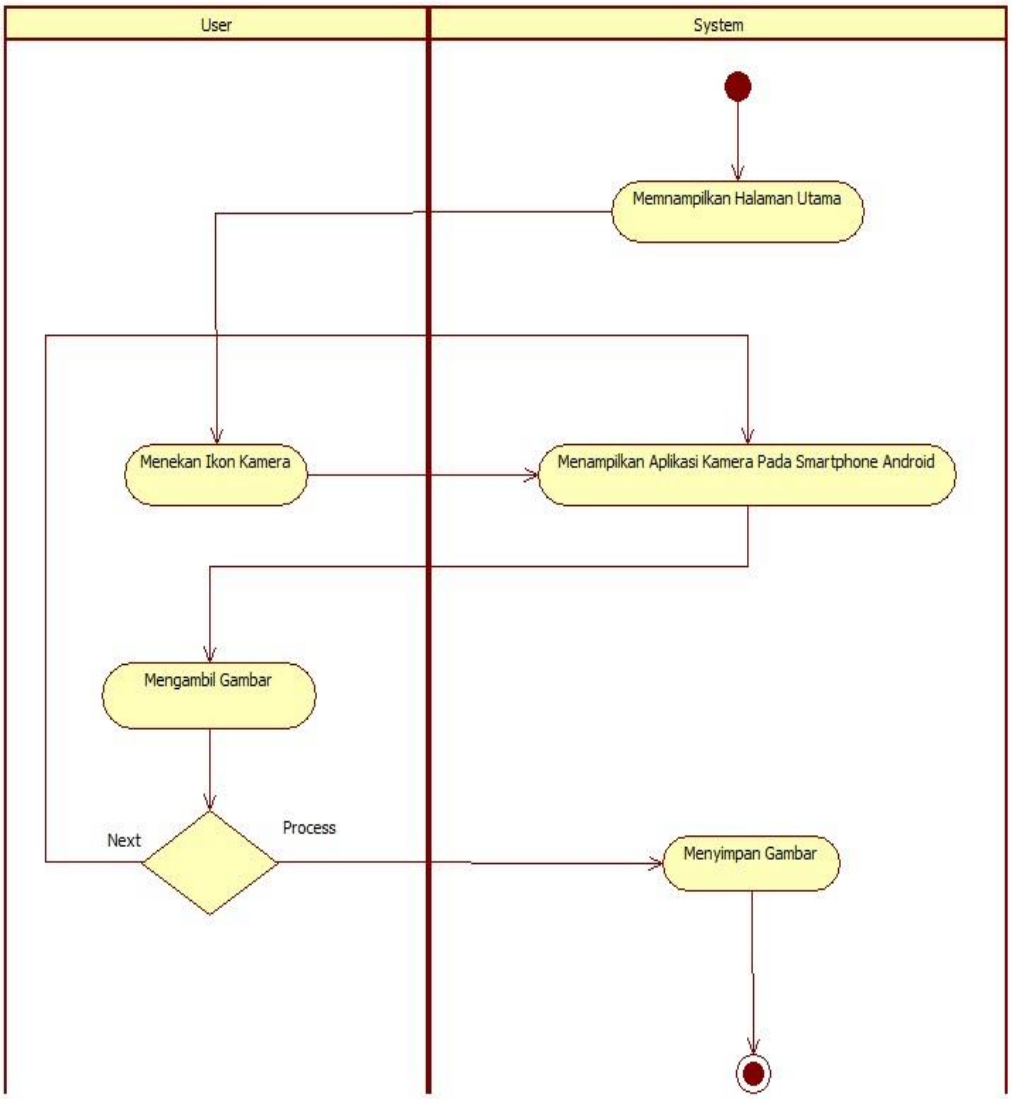



Sumber gambar :Sri Dharwiyanti dan romi Satria Wahono [18]

Gambar II- 13 Collaboration Diagram

### 2.13.1.5 Activity Diagram

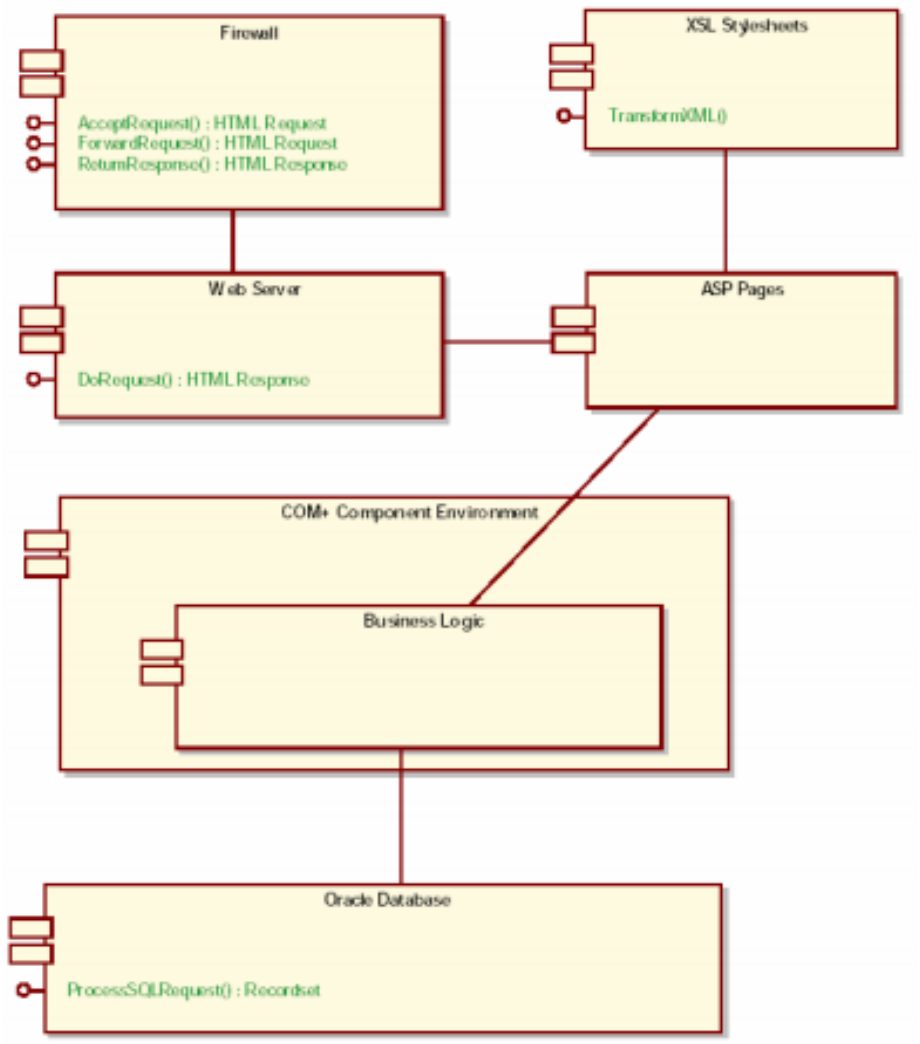
Sebuah *Activity Diagram* menunjukkan suatu alur kegiatan secara berurutan.*Activity Diagram* digunakan untuk mendiskripsikan kegiatan-kegiatan dalam sebuah operasi meskipun juga dapat digunakan untuk mendeskripsikan alur kegiatan yang lainnya seperti *use case* atau suatu interaksi.




Gambar II- 14 Activity Diagram

### 2.13.1.6 Component Diagram

*Component Diagram* menunjukkan struktur dan hubungan antar komponen *software* termasuk ketergantungan (*dependency*) diantara komponen-komponen tersebut. Komponen pada piranti lunak adalah berupa modul-modul yang berisikan *code*, baik *library* maupun *executable*. Umumnya komponen yang terbentuk dari beberapa *class* dan/atau *package*, atau juga dapat dari komponenkomponen yang lebih kecil.

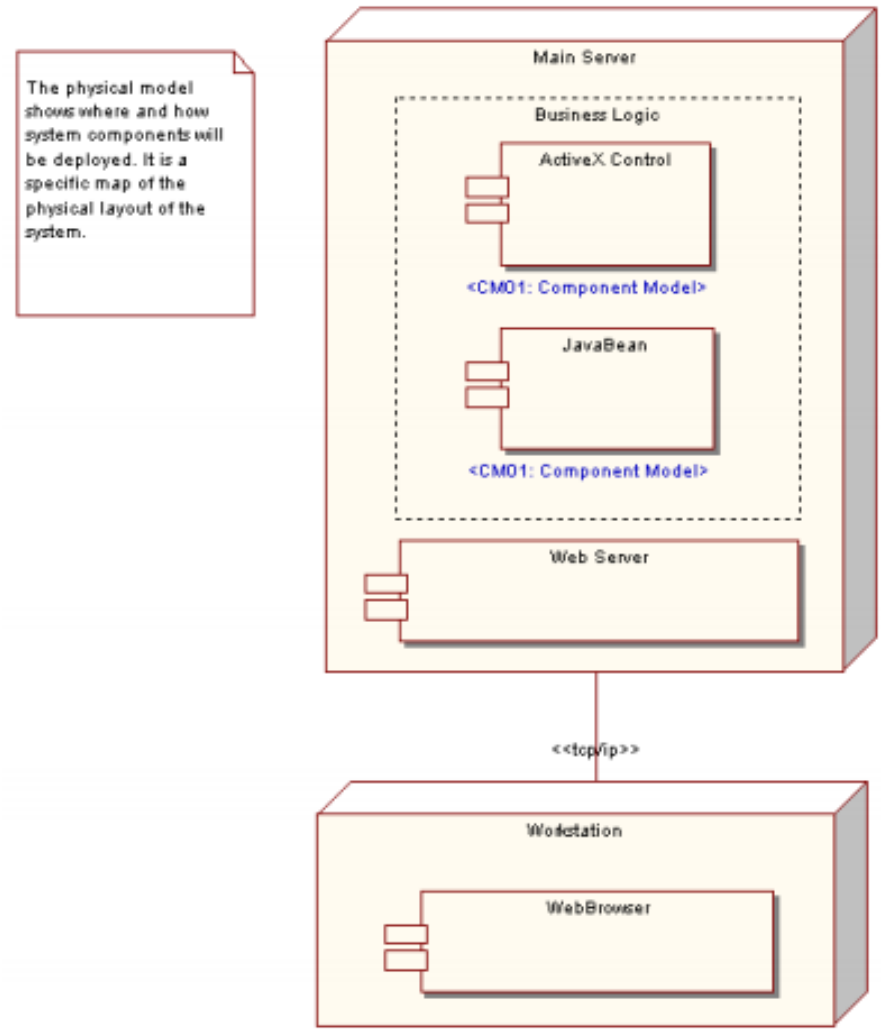



Sumber gambar :Sri Dharwiyanti dan romi Satria Wahono [18]

Gambar II- 15 Component Diagram

### 2.13.1.7 Deployment Diagram

*Deployment Diagram* menunjukkan arsitektur fisik pada *hardware* dan *software* pada suatu sistem yang dirancang. *Deployment diagram* juga dapat menunjukkan perngkat-perangkat dan *nodes* diantara hubungan yang dimilikinya antar komponen.




Sumber gambar :Sri Dharwiyanti dan romi Satria Wahono [18]

Gambar II- 16 Deployment Diagram

## 2.14 Software Development Kit (SDK)

*Software Development Kit (SDK)* adalah suatu kit atau library dari bahasa pemrograman untuk pengembangan atau pembangunan suatu perangkat lunak dan biasanya SDK terdiri dari kumpulan tools yang dibutuhkan. Misalnya bahasa pemrograman java, mempunyai SDK yang berisi suatu *library* yang dapat digunakan untuk membuat suatu aplikasi berbasis java.

## 2.15 Java Development Kit (JDK)

*Java Development Kit* (JDK) adalah sekumpulan perangkat lunak yang dapat kamu gunakan untuk mengembangkan perangkat lunak yang berbasis Java, Sedangkan JRE adalah sebuah implementasi dari *Java Virtual Machine* yang benar-benar digunakan untuk menjalankan program java.Biasanya, setiap JDK berisi satu atau lebih JRE dan berbagai alat pengembangan lain seperti sumber *kompiler java, bundling, debuggers, development libraries* dan lain sebagainya. Perbedaan JDK dengan SDK (*Software Development Kit*) yaitu JDK adalah sebuah SDK tetapi sebuah SDK tidak harus menjadi sebuah JDK