

BAB II

LANDASAN TEORI

2.1 Notifikasi

Kamus Besar Bahasa Indonesia (2010) menguraikan Pengertian notifikasi berarti pemberitahuan atau kabar. Notifikasi dalam penelitian ini dijabarkan sebagai pemberitahuan yang terdapat pada jadwal imunisasi .

2.2 Firebase Cloud Messaging (FCM)

Hafidudin (2017: 01) menguraikan bahwa *Firebase Cloud Messaging* untuk *Android* (FCM) adalah layanan yang membantu pengembang mengirim data dari server untuk aplikasi mereka *Android* pada perangkat *Android*. Bisa menjadi pesan ringan memberitahu aplikasi *Android* bahwa ada data baru yang akan diambil dari server (misalnya, film diunggah oleh seorang teman), atau bisa juga pesan yang berisi sampai dengan 4KB data payload (sehingga aplikasi seperti instant messaging dapat mengkonsumsi pesan langsung). Layanan FCM menangani semua aspek antrian pesan dan pengiriman ke aplikasi target *Android* berjalan pada perangkat target.

2.3 Sistem Operasi

Pangera dan Ariyus (2010: 57) menguraikan bahwa sistem operasi adalah suatu sistem yang terdiri dari atas komponen-komponen kerja dan muatan metod kerja yang di gunakan untuk memanfaatkan mesin, sehingga mesin dapat bekerja sesuai dengan yang di impikan. Fungsi utama sistem operasi adalah untuk media interaksi manusa dengan mesin , artinya bagaimana manusia dapat memahami mesin dan sebaliknya sehingga merupakan partner yang saling memahami untuk melakukan suatu tugas. Berikut ini adalah jenis jenis sistem operasi :

- a. *DOS*
- b. *WINDOWS*
- c. *UNIX*
- d. *Linux*
- e. *Android*

2.4 *Android*

2.4.1 *Sejarah Android*

Kasman (2013:02) menguraikan bahwa “*Android*” merupakan sebuah sistem operasi yang berbasis *Linux* untuk perangkat *portable* seperti *smartphone* dan komputer tablet”. *Android* menyediakan *platform* terbuka bagi programmer untuk mengembangkan aplikasi sendiri pada berbagai perangkat dengan sistem operasi *Android*.

Android merupakan sistem operasi untuk telepon seluler berbasis *linux* sebagai kernelnya. *Android* menyediakan platform terbuka (*open source*) bagi para pengembang untuk menciptakan aplikasi mereka sendiri. Awalnya, perusahaan *search engine* terbesar saat ini, yaitu *Google Inc*, membeli *Android Inc*, pendatang baru yang membuat perangkat lunak untuk ponsel. *Android Inc*. Didirikan oleh *Andy Rubin*, *Rich Milner*, *Nick Sears* dan *Chris White* pada tahun 2003. Pada Agustus 2005 *Google* membeli *Android Inc*. Dimulai pada tahun 2005, *Android Inc*. dibawah naungan *Google Inc*. Berusaha membuat sebuah *operating system mobile* baru. Sejak saat itulah mulai beredar rumor bahwa *Google* akan melakukan ekspansi bisnis ke industri seluler. Akhirnya pada bulan September 2007 *Google* mengajukan hak paten atas produknya yang dinamai *Nexus One*.

Akhir tahun 2008, dibentuk sebuah tim kerja sama yang dinamai *Open Handset Alliance (OHA)*. *OHA* ini terdiri dari beberapa produsen perangkat telekomunikasi ternama dunia, antara lain *ASUS*, *Toshiba*, *Sony Ericsson* (sekarang *Sony*), *Garmin*, *Vodafone*, dan *Softbank*. *OHA* bekerja sama untuk mengembangkan sebuah kernel *Linux* yang akan dijadikan sebuah program untuk perangkat seluler. Hingga akhirnya *OHA* berhasil dan mengumumkan produk *operating system mobile* yang diberi nama *Android*. Ponsel yang mendapat kehormatan untuk mencoba pertama kali sistem operasi *Android* adalah *HTC Dream*.

2.4.2 *Komponen Android*

Kasman (2013 : 21) menguraikan bahwa *Android SDK (Software Development Kit)* merupakan sebuah *tools* yang diperlukan untuk mengembangkan aplikasi berbasis *Android* menggunakan bahasa

pemrograman *Java*. Pada saat ini *Android SDK* telah menjadi alat bantu dan *API (Application Programming Interface)* untuk mengembangkan aplikasi berbasis *Android*. *Android SDK* dapat anda lihat dan unduh pada situs resminya, yaitu *www.developer.Android.com*, *Android SDK* bersifat gratis dan bebas anda distribusikan karena *Android* bersifat *open source*.ty

2.4.3 Kelebihan Sistem Operasi *Android*

Kelebihan dari sistem operasi *Android* adalah sebagai berikut :

1. *Complete Platform*

Sistem operasi *Android* adalah sistem operasi yang banyak menyediakan *tools* yang berguna untuk membangun sebuah aplikasi yang kemudian aplikasi tersebut dapat lebih dikembangkan lagi oleh para *developer*.

2. *Open Source Platform*

Platform Android yang bersifat *open source* menjadikan sistem operasi ini mudah dikembangkan oleh para *developer* karena bersifat terbuka.

3. *Free Platform*

Developer dengan bebas bisa mengembangkan, mendistribusikan dan memperdagangkan sistem operasi *Android* tanpa harus membayar royalti untuk mendapatkan *license*.

2.4.4 Perkembangan Versi OS *Android*

Kasman (2013 :04) menguraikan bahwa Keunikan dari sistem operasi *Android* adalah dengan nama makanan hidangan penutup. Versi *Android* diawali dengan dirilisnya *Android beta* pada bulan November 2007. Versi komersial pertama, *Android 1.0*, dirilis pada September 2008. *Android* dikembangkan secara berkelanjutan oleh *Google* dan *Open Handset Alliance (OHA)*, yang telah merilis sejumlah pembaruan sistem operasi ini sejak dirilisnya versi awal.

Sejak April 2009, versi *Android* dikembangkan dengan nama kode yang dinamai berdasarkan makanan pencuci mulut dan penganan manis. Masing-masing versi dirilis sesuai urutan *alfabet*, yakni *Cupcake* (1.5), *Donut* (1.6), *Eclair* (2.0–2.1), *Froyo* (2.2–2.2.3), *Gingerbread* (2.3–2.3.7), *Honeycomb*

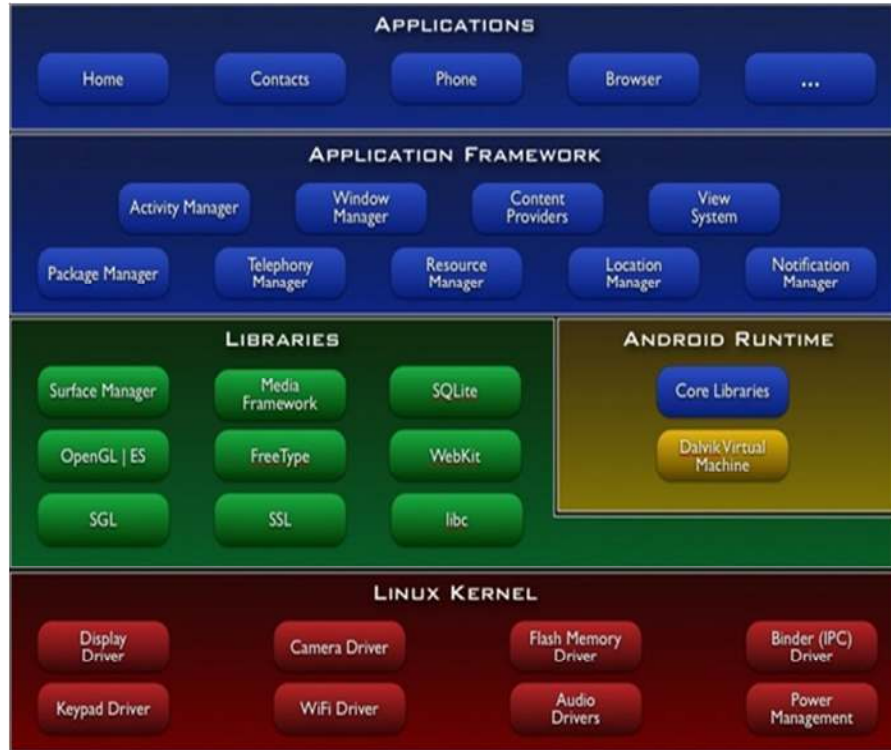
(3.0–3.2.6), *Ice Cream Sandwich* (4.0–4.0.4), *Jelly Bean* (4.1–4.3), *KitKat* (4.4+), *Lollipop* (5.0+), *Marshmallow* (6.0+), hingga yang terbaru adalah *Nougat* (7.0+) dan selanjutnya versi *Android* terbaru yang ditunggu-tunggu adalah *Android O* (8.0+), Berikut ini adalah table 2.1. yang menguraikan urutan versi *Android* dan *level API* pada *Android*:

Tabel 2.1. Urutan versi *Android*

Versi	Nama kode	Tanggal rilis	Level API	Distribusi
1.5	<i>Cupcake</i>	30 April 2009	3	
1.6	<i>Donut</i>	15 September 2009	4	
2.0–2.1	<i>Eclair</i>	26 Oktober 2009	7	
2.2	<i>Froyo</i>	20 Mei 2010	8	0,7%
2.3.3–2.3.7	<i>Gingerbread</i>	9 Februari 2011	10	11,7%
2.3–2.3.2	<i>Gingerbread</i>	6 Desember 2010	9	
3.1	<i>Honeycomb</i>	10 Mei 2011	12	
3.2	<i>Honeycomb</i>	15 Juli 2011	13	
4.0.3–4.0.4	<i>Ice Cream Sandwich</i>	16 Desember 2011	15	9,6%
4.1.x	<i>Jelly Bean</i>	9 Juli 2012	16	25,1%
4.2.x	<i>Jelly Bean</i>	13 November 2012	17	20,7%
4.3.x	<i>Jelly Bean</i>	24 Juli 2013	18	8%
4.4.x	<i>KitKat</i>	31 Oktober 2013	19	24,5%
5.x	<i>Lollipop</i>	15 Oktober 2014	21	
6.0	<i>Marshmallow</i>	19 Agustus 2015	23	
7.0	<i>Nougat</i>	22 Agustus 2016	24	Kurang dari 0.1%

2.4.5 Arsitektur *Android*

Kasman (2013 : 06) menguraikan bahwa Secara umum, arsitektur *Android* terdiri dari lapisan *software*, yaitu lapisan aplikasi, framework aplikasi, pustaka, *Android* runtime, dan kernel linux. Gambar 2.1 berikut merupakan arsitektur dari *Android* :



Gambar 2.1 Arsitektur *Android*

Penjelasan dari arsitektur *Android* adalah sebagai berikut :

1. *Framework Aplikasi* Dibawah aplikasi terdapat sejumlah *software* pendukung, meliputi layanan pengelola activity, view, sumber daya, pemberitahuan (notifikasi) dan lain lain.
2. Pustaka *Android* menyediakan sejumlah pustaka dasar seperti pustaka grafis 2D dan 3D, multimedia playback, browser engine, percetakan font, database dan lain – lain. Aplikasi yang memanfaatkan pustaka fungsi ini melalui lisan framework aplikasi.
3. *AndroidRunTime* menyediakan pustaka inti bagi pemrograman Java, dan terdapat *Dalvik Virtual Machine* yang akan menjalankan aplikasi. Tiap aplikasi akan dijalankan pada proses terpisah dengan Virtual Machine yang berbeda sehingga terisolasi satu sama lain. Aplikasi masih dapat berkomunikasi dengan aplikasi lain melalui mekanisme yang disediakan *framework* aplikasi. *Dalvik* VM bergantung pada lapisan dibawahnya (*kernel linux*) untuk multi-threading dan pengelolaan memori tingkat rendah.

4. *Kernel linux* bertanggung jawab menyediakan layar dasar seperti keamanan, pengelolaan proses, pengelolaan file, pengelolaan sumber daya memori dan hardware.

2.5 Android Studio

Satyaputra & Aritonang (2016 : 01) menguraikan bahwa *Android studio* adalah IDE resmi untuk membangun aplikasi *Android* berdasarkan IntelliJ IDEA. IntelliJ IDEA sendiri adalah *java Integrated Development Environment (IDE)* yang dikembangkan oleh JetBrains, untuk mengembangkan perangkat lunak komputer. IntelliJ IDE berfungsi dalam membantu anda dalam dunia per-coding-an baik dari segi navigasi, penyokong produktivitas, hingga code editor yang cerdas. *Android studio* dibangun dengan tujuan mempercepat proses pembangunan maupun pengembangan aplikasi yang berkualitas tinggi untuk setiap *device Android*.

2.6 ADT (Android Development Tools)

Nazruddin (2010 : 06) menguraikan bahwa *Android Development Tools (ADT)* adalah *plugin* untuk *Eclipse Intergrated Development Environment (IDE)* yang dirancang untuk memberikan lingkungan yang terpadu di mana untuk membangun aplikasi *Android*. *ADT* memperluas kemampuan *Eclipse* untuk membiarkan para developer lebih cepat dalam membuat proyek baru *Android*, membuat aplikasi *UI*, menambahkan komponen berdasarkan *Android Framework API*, debug aplikasi dalam penggunaan *Android SDK*, dan membuat *file APK* untuk mendistribusikan aplikasi.

2.7 Black-Box Testing

(Pressman, 2012 : 597) Menguraikan *Black Box Testing* atau Pengujian Kotak Hitam atau juga disebut Behavioral Testing, berfokus pada persyaratan fungsional dari perangkat lunak. Artinya, teknik *Black-Box Testing* memungkinkan untuk mendapatkan set kondisi masukan yang sepenuhnya akan melaksanakan semua persyaratan fungsional untuk suatu program.

Black-Box Testing bukan merupakan alternatif dari pengujian *White Box Testing*. Sebaliknya, *Black-Box Testing* adalah pendekatan komplementer yang mungkin

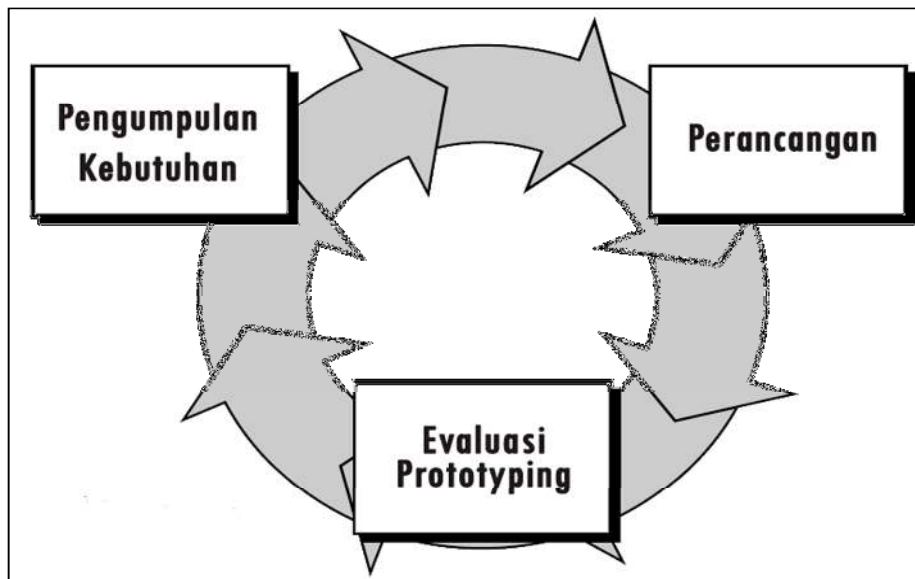
untuk mengungkap kelas yang berbeda dari kesalahan daripada metode *White Box Testing*.

Black Box Testing mencoba untuk menemukan kesalahan dalam kategori berikut.

1. Fungsi tidak benar atau hilang.
2. Kesalahan interface atau antarmuka.
3. Kesalahan dalam struktur data atau akses database eksternal.
4. Kesalahan kinerja atau perilaku.
5. Kesalahan inisialisasi dan terminasi.

2.8 Metode Pengembangan Perangkat Lunak Model *Prototype*

Pressman (2012 : 40) menguraikan bahwa *Prototyping* paradigma dimulai dengan pengumpulan kebutuhan, pengembang bertemu dengan pengguna dan mengidentifikasi objektif keseluruhan dari perangkat lunak, selanjutnya mengidentifikasi segala kebutuhan yang diketahui secara garis besar di mana definisi-definisi lebih jauh merupakan keharusan, kemudian dilakukan perancangan kilat, lalu diakhiri dengan evaluasi *prototyping* yang dapat dilihat pada gambar 2.2 berikut :



Gambar 2.2 Model *Prototype*

Tahap–tahap rekayasa *software* dalam *Prototype model* pada gambar 2.2 di atas adalah sebagai berikut :

1. Pengumpulan kebutuhan

Developer dan klien bertemu untuk menentukan tujuan umum, kebutuhan yang diketahui dan gambaran bagian-bagian yang akan dibutuhkan berikutnya. Detail kebutuhan mungkin tidak dibicarakan disini, pada awal pengumpulan kebutuhan. Selanjutnya peneliti akan melakukan analisis terhadap data apa saja yang dibutuhkan, seperti analisis terhadap sistem yang berjalan, analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis kebutuhan notifikasi imunisasi.

2. Perancangan

Perancangan dilakukan dengan cepat dan rancangan mewakili semua aspek *software* yang diketahui, dan rancangan ini menjadi dasar pembuatan *prototype*. Dalam tahap ini peneliti akan membangun sebuah versi *prototype* yang dirancang kembali dimana masalah-masalah tersebut diselesaikan.

3. Evaluasi *prototype*

Pada tahap ini, calon pengguna mengevaluasi *prototype* yang dibuat dan digunakan untuk memperjelas kebutuhan *software*. *Software* yang sudah jadi dijalankan dan akan dilakukan perbaikan apabila kurang memuaskan. Perbaikan termasuk dalam memperbaiki kesalahan/ kerusakan yang tidak ditemukan pada langkah sebelumnya

Kelebihan dari *Prototype Model* adalah sebagai berikut :

1. *End user* dapat berpartisipasi aktif.
2. Penentuan kebutuhan lebih mudah diwujudkan.
3. Mempersingkat waktu pengembangan *software*.

Kekurangan dari *Prototype Model* adalah sebagai berikut:

1. Proses analisis dan perancangan terlalu singkat.
2. Mengesampingkan alternatif pemecahan masalah.
3. Biasanya kurang fleksibel dalam menghadapi perubahan.

4. *Prototype* yang dihasilkan tidak selamanya mudah dirubah.
5. *Prototype* terlalu cepat selesai.

2.9 UML (*Unified Modeling Language*)

2.9.1 Pengertian UML

Nugroho A (2010 : 06) menguraikan bahwa UML merupakan bahasa untuk membangun dan mendokumentasikan *artifacts* (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, *artifact* tersebut dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak. Selain itu UML adalah bahasa pemodelan yang menggunakan konsep orientasi object. UML dibuat oleh *Grady Booch*, *James Rumbaugh*, dan *Ivar Jacobson* di bawah bendera *Rational Software Crop*. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai perspektif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namu hampir dalam semua bidang yang membutuhkan pemodelan.

2.9.2 Bagian-Bagian UML

Bagian-bagian utama dari UML adalah *view*, *diagram*, *model element*, dan *general mechanism*. Diagram berbentuk grafik yang menunjukan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu *view* tertentu dan ketika digambarkan biasanya dialokasikan untuk *view* tertentu. Adapun jenis diagram antara lain:

1. *Use Case Diagram*

Use case adalah abstraksi dari interaksi antara *system* dan *actor*. *Use case* bekerja dengan cara mendeskripsikan tipe intraksi antara lain *user* sebuah *system* dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah *system* dipakai. *Use case* merupakan konstruksi untuk mendeskripsikan bagaimana sistem akan terlihat di mata *user*. Sedangkan *Use case* diagram memfasilitasi komunikasi diantara analis dan pengguna serta analis dan *clinet*.

2. *Class Diagram*

Class adalah deskripsi kelompok obyek-obyek dengan *property*, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya *Class diagram* dapat memberikan pandangan global atas sebuah *system*. Hal tersebut tercermin dari *class-class* yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem biasanya mempunyai beberapa *class diagram*. *Class diagram* sangat membantu dalam visualisasi setruktur kelas dari suatu sistem.

3. *Activity Diagram*

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau intraksi.

2.9.3 Tujuan dan Keunggulan UML

Tujuan UML adalah sebagai berikut: Memodelkan suatu sistem (bukan hanya perangkat lunak) yang menggunakan konsep berorientasi *object*, menciptakan suatu bahasa pemodelan yang dapat digunakan baik oleh manusia maupun mesin.

Keunggulan menggunakan UML dibandingkan menggunakan metodologi terstruktur:

1. *Uniformity*

Pengembangan cukup menggunakan satu metodologi dari tahap analisis hingga perancangan. Memungkinkan merancang komponen antarmuka secara terintegrasi bersama perancangan perangkat lunak dan perancangan struktur data.

2. *Understandability*

Kode yang dihasilkan dapat diorganisasi kedalam kelas-kelas yang berhubungan dengan masalah yang sesungguhnya sehingga lebih mudah untuk dipahami.

3. *Stability*

Kode program yang dihasilkan relatif stabil sepanjang waktu, karena mendekati permasalahan yang sesungguhnya.

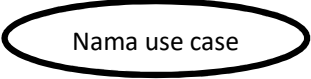


4. Reusability


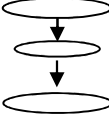
Dengan metodologi berorientasi objek, dimungkinkan penggunaan ulang kode, sehingga pada akhirnya akan sangat mempercepat waktu pengembangan perangkat lunak (atau sistem informasi).

2.9.4 Simbol-Simbol pada UML

Simbol-simbol yang terdapat dalam diagram UML. Dapat dilihat pada tabel 2.2 dibawah ini :

Tabel 2.2 Simbol Pada *Diagram* UML.



Simbol	Deskripsi
<p><i>Use Case</i></p> 	<p>Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan anatar unit atau aktor, biasanya akan diterangkan dengan menggunakan kata kerja diawal-diawal frase nama <i>use case</i>.</p>
<p>Aktor/<i>Actor</i></p>  <p>Nama Aktor</p>	<p>Orang, proses, atau sistem lain yang berintraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah orang, tapi aktor belum tentu merupakan orang. Biasanya akan dinyatakan menggunakan kata benda diawal frase nama aktor.</p>
<p>Asosiasi/<i>Association</i></p> 	<p>Komunikasi antar aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor.</p>


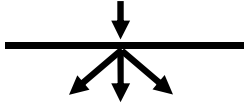
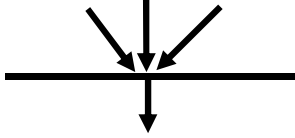
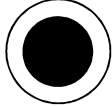
<p>Ekstensi/Extend</p> <p><<extend></p> 	<p>Case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan, misal .</p>  <p>Arah panah mengarah pada <i>use case</i> yang ditambahkan.</p>
<p>Uses</p> <p><<uses>></p>	<p>Digunakan sebagai kegiatan utama atau syarat menuju <i>use case</i> berikutnya.</p>

2.9.5 Activity Diagram

Diagram aktivitas digunakan untuk menggambarkan alur kerja suatu sistem informasi. Sebuah diagram aktivitas menunjukkan suatu alur kegiatan secara berurutan. Tabel 2.3 berikut ini adalah simbol-simbol yang ada pada diagram aktifitas:

Tabel 2.3 Simbol-Simbol *Activity Diagram*

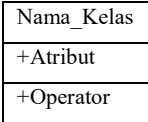



Simbol	Deskripsi
<p>Status awal</p> 	<p>Setatus awal aktivitas sistem, sebuah diagram aktivitas memiliki status awal</p>
<p>Aktivitas</p> 	<p>Aktivitas yang dilakukan sistem. Aktivitas biasanya diawali dengan kata kerja.</p>

<p>Pencabangan / decision</p> 	<p>Asosiasi penggabungan dimana lebih satu aktivitas.</p>
<p>Fork</p> 	<p>Digunakan untuk menunjukan kegiatan yang dilakukan secara paralel.</p>
<p>Penggabungan / Join</p> 	<p>Digunakan untuk menunjukan kegiatan yang digabungkan.</p>
<p>End Point</p> 	<p>Mengakhiri aktivitas sistem.</p>

2.9.6 Class Diagram

Diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki atribut yaitu variabel-variabel yang dimiliki oleh suatu kelas dan operasi atau metode yaitu fungsi-fungsi yang dimiliki oleh suatu kelas. Tabel 2.4 berikut ini adalah simbol-simbol yang ada pada *clas diagram* :

Tabel 2.4 Bagan *Class Diagram*

Simbol	Deskripsi
<p>Kelas</p> 	Kelas pada Struktur
<p><i>Interface</i></p>  <p>Nama <i>Interface</i></p>	Metode pada <i>interface</i> yang digunakan pada suatu kelas sama persis dengan yang ada pada <i>interface</i> .
<p>Asosiasi</p> 	Relasi antara kelas dengan makna umum.
<p>Asosiasi Berarah</p> 	Relasi antar kelas dengan makna kela yang satu digunakan pada kelas lain.

2.10 Kuesioner

Isti Pujihastuti (2010 : 01) menguraikan bahwa Kuesioner merupakan alat pengumpulan data primer dengan metode survei untuk memperoleh opini responden. Kuesioner dapat didistribusikan kepada responden dengan cara :

1. Langsung oleh peneliti (mandiri)
2. Dikirim lewat pos (mailquestionair)
3. Dikirim lewat komputer misalnya surat elektronik (e-mail).

Kuesioner dikirimkan langsung oleh peneliti apabila responden relatif dekat dan penyebarannya tidak terlalu luas. Lewat pos ataupun e-mail memungkinkan biaya yang murah, daya jangkau responden lebih luas, dan waktu cepat. Tidak ada prinsip khusus namun peneliti dapat mempertimbangkan efektivitas dan efisiensinya dalam hal akan dikirim lewat pos, e-mail ataupun langsung dari peneliti.

Kuesioner dapat digunakan untuk memperoleh informasi pribadi misalnya sikap, opini, harapan dan keinginan responden. Idealnya semua responden mau mengisi atau lebih tepatnya memiliki motivasi untuk menyelesaikan pertanyaan ataupun pernyataan yang ada pada kuesioner penelitian. Apabila tingkat respon (repon rate) diharapkan 100% artinya semua kuesioner yang dibagikan kepada responden akan diterima kembali oleh peneliti dalam kondisi yang baik dan kemudian akan dianalisis lebih lanjut.

2.10.1 Definisi Populasi

Populasi adalah wilayah generalisasi berupa subjek atau objek yang diteliti untuk dipelajari dan diambil kesimpulan. Sedangkan sampel adalah sebagian dari populasi yang diteliti. Dengan kata lain, sampel merupakan sebagian atau bertindak sebagai perwakilan dari populasi sehingga hasil penelitian yang berhasil diperoleh dari sampel dapat digeneralisasikan pada populasi.

Penarikan sampel diperlukan jika populasi yang diambil sangat besar, dan peneliti memiliki keterbatasan untuk menjangkau seluruh populasi maka peneliti perlu mendefinisikan populasi target dan populasi terjangkau baru kemudian menentukan jumlah sampel dan teknik sampling yang digunakan.

2.10.2 Ukuran Sampel

Untuk menentukan sampel dari populasi digunakan perhitungan maupun acuan tabel yang dikembangkan para ahli. Secara umum, untuk penelitian korelasional jumlah sampel minimal untuk memperoleh hasil yang baik adalah 30, sedangkan dalam penelitian eksperimen jumlah sampel minimum 15 dari masing-masing kelompok dan untuk penelitian survey jumlah sampel minimum adalah 100. Roscoe (1975) yang dikutip Uma Sekaran (2006) memberikan acuan umum untuk menentukan ukuran sampel. Ukuran sampel lebih dari 30 dan kurang dari 500 adalah tepat untuk kebanyakan penelitian. Jika sampel dipecah ke dalam subsampel (pria/wanita, junior/senior, dan sebagainya), ukuran sampel minimum 30 untuk tiap kategori adalah tepat. Dalam penelitian multivariate (termasuk analisis regresi berganda), ukuran sampel sebaiknya 10x lebih besar dari jumlah variabel dalam penelitian. Untuk penelitian eksperimental sederhana dengan kontrol eksperimen yang ketat, penelitian yang sukses adalah mungkin dengan ukuran sampel kecil antara 10 sampai dengan 20. Besaran atau ukuran sampel ini sangat tergantung dari besaran tingkat ketelitian atau kesalahan yang diinginkan peneliti. Namun, dalam hal tingkat kesalahan, pada penelitian sosial maksimal tingkat kesalahannya adalah 5% (0,05). Makin besar tingkat kesalahan maka makin kecil jumlah sampel. Namun yang perlu diperhatikan adalah semakin besar jumlah sampel (semakin mendekati populasi) maka semakin kecil peluang kesalahan generalisasi dan sebaliknya, semakin kecil jumlah sampel (menjauhi jumlah populasi) maka semakin besar peluang kesalahan generalisasi.

2.10.3 Teknik Sampling

Teknik sampling merupakan teknik pengambilan sampel yang secara umum terbagi dua yaitu probability sampling dan non probability sampling. Dalam pengambilan sampel cara probabilitas besarnya peluang atau probabilitas elemen populasi untuk terpilih sebagai subjek diketahui. Sedangkan dalam pengambilan sampel dengan cara nonprobability besarnya peluang elemen untuk ditentukan sebagai sampel tidak diketahui. Menurut Sekaran (2006), desain pengambilan sampel dengan cara probabilitas jika representasi sampel adalah penting dalam rangka generalisasi lebih luas. Bila waktu atau faktor lainnya, dan masalah generalisasi tidak diperlukan, maka cara *nonprobability* biasanya yang digunakan.

1. *Probability Sampling*

Probability sampling adalah teknik pengambilan sampel yang memberikan peluang yang sama kepada setiap anggota populasi untuk menjadi sampel. Teknik ini meliputi *simple random sampling*, *systematic sampling*, *proportionate stratified random sampling*, *disproportionate stratified random sampling*, dan *cluster sampling*

a. *Simple random sampling*

Teknik ini adalah teknik yang paling sederhana (simple). Sampel diambil secara acak, tanpa memperhatikan tingkatan yang ada dalam populasi.

b. *Sampling Sistematis*

Sampling Sistematis Adalah teknik sampling yang menggunakan nomor urut dari populasi baik yang berdasarkan nomor yang ditetapkan sendiri oleh peneliti maupun nomor identitas tertentu, ruang dengan urutan yang seragam atau pertimbangan sistematis lainnya.

c. *Proportionate Stratified Random Sampling*

Teknik ini hampir sama dengan *simple random sampling* namun penentuan sampelnya memperhatikan strata (tingkatan) yang ada dalam populasi. Teknik ini umumnya digunakan pada populasi yang diteliti adalah heterogen (tidak sejenis) yang dalam hal ini berbeda dalam hal

bidangkerja sehingga besaran sampel pada masing-masing strata atau kelompok diambil secara proporsional untuk memperoleh

d. Disproportionate Stratified Random Sampling

Disproporsional stratified random sampling adalah teknik yang hampir mirip dengan proportionate stratified random sampling dalam hal heterogenitas populasi. Namun, ketidakproporsionalan penentuan sample didasarkan pada pertimbangan jika anggota populasi berstrata namun kurang proporsional pembagiannya.

e. Cluster Sampling

Cluster sampling atau sampling area digunakan jika sumber data atau populasi sangat luas misalnya penduduk suatu propinsi, kabupaten, atau karyawan perusahaan yang tersebar di seluruh provinsi. Untuk menentukan mana yang dijadikan sampelnya, maka wilayah populasi terlebih dahulu ditetapkan secara random, dan menentukan jumlah sample yang digunakan pada masing-masing daerah tersebut dengan menggunakan teknik proporsional stratified random sampling mengingat jumlahnya yang bisa saja berbeda.

2. Non Probability Sampel

Non Probability artinya setiap anggota populasi tidak memiliki kesempatan atau peluang yang sama sebagai sampel. Teknik-teknik yang termasuk ke dalam Non Probability ini antara lain : *Sampling Sistematis, Sampling Kuota, Sampling Insidental, Sampling Purposive, Sampling Jenuh, dan Snowball Sampling.*

a. Sampling Kuota

Sampling Kuota adalah teknik sampling yang menentukan jumlah sampel dari populasi yang memiliki ciri tertentu sampai jumlah kuota (jatah) yang diinginkan.

b. Sampling Insidental

Insidental merupakan teknik penentuan sampel secara kebetulan, atau siapa saja yang kebetulan (insidental) bertemu dengan peneliti yang dianggap cocok dengan karakteristik sampel yang ditentukan akan dijadikan sampel.

c. *Sampling Purposive*

Purposive sampling merupakan teknik penentuan sampel dengan pertimbangan khusus sehingga layak dijadikan sampel. Misalnya, peneliti ingin meneliti permasalahan seputar daya tahan mesin tertentu. Maka sampel ditentukan adalah para teknisi atau ahli mesin yang mengetahui dengan jelas permasalahan ini. Atau penelitian tentang pola pembinaan olahraga renang. Maka sampel yang diambil adalah pelatih-pelatih renang yang dianggap memiliki kompetensi di bidang ini. Teknik ini biasanya dilakukan pada penelitian kualitatif.

d. *Sampling Jenuh*

Sampling jenuh adalah sampel yang mewakili jumlah populasi. Biasanya dilakukan jika populasi dianggap kecil atau kurang dari 100. Saya sendiri lebih senang menyebutnya total sampling.

2.11 Penelitian Terdahulu

Tabel 2.5 di bawah ini adalah jurnal penelitian terdahulu terkait dengan notifikasi berbasis *Android* :

Tabel 2.5 Jurnal Penelitian Terdahulu Terkait Dengan Notifikasi Berbasis *Android*

No	Judul Jurnal	Penulis	Tahun/terbit	Uraian
1	IMPLEMENTASI PUSH NOTIFICATION PADA INFORMASI PERKULIAHAN DAN KEGIATAN MAHASISWA BERBASIS <i>ANDROID</i>	Jefferson Setiawan, Edy Kristianto, Fredicia	Fakultas Teknik dan Ilmu Komputer Jurusan Teknik Informatika Universitas Kristen Krida Wacana – Jakarta, Juni 2015	Aplikasi menggunakan layanan google cloud messaging untuk fitur push notification, dan memanfaatkan api calendar contract untuk integrasi dengan aplikasi calendar
2	PENGEMBANGAN APLIKASI <i>ANDROID</i> BIMBINGAN SKRIPSI DENGAN FITUR NOTIFIKASI	Muhammad Zaky Faried, Anggraini Mulwinda, dan Yohanes Primadiyo	Jurusan Teknik Elektro, Fakultas Teknik, Universitas Negeri Semarang Kampus Sekaran, Gunungpati, Semarang, 50229,	Aplikasi <i>Android</i> Bimbingan Skripsi dengan fitur notifikasi berfokus pada mengetahui pentingnya fitur notifikasi pada aplikasi, sehingga

		no	Indonesia ,Desember 2017	kemampuan aplikasi masih terbatas pada dapat melakukan bimbingan skripsi, memvalidasi bimbingan skripsi, edit bimbingan skripsi, dan memberikan notifikasi pada bimbingan baru dan bimbingan yang telah divalidasi
3	RANCANG BANGUN APLIKASI MOBILE UNTUK NOTIFIKASI JADWAL KULIAH BERBASIS <i>ANDROID</i>	Taufik Ramadha, Victor G Utomo	Program Studi Teknik Informatika STMIK PROVISI Semarang, Agustus 2014	Aplikasi mobile dapat dikembangkan lebih lanjut dengan menambahkan fitur lain seperti push notifikasi untuk menghemat penggunaan memori smartphone mahasiswa. Aplikasi mobile dapat dikembangkan agar bisa berjalan pada sistem operasi smartphone selain <i>Android</i> untuk menambah lingkungan pengguna
4	PERANCANGAN DAN IMPLEMENTASI SISTEM NOTIFIKASI PADA SISTEM JEJARING KLASER BERBASIS <i>ANDROID</i>	Rendi, Kristanto, Suprihadi, Radius	Jurusan Komputer, Program Studi Teknik Informatika Universitas Kristen Satya Wacana ,Salatiga 2013	Berdasarkan pengujian aplikasi metode push notifikasi tidak membebani client, karena client dapat melakukan pemrosesan data jika ada notifikasi yang masuk