BAB II

LANDASAN TEORI

2.1 Aplikasi

Menurut Wahana Komputer (2003:122), “Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu”.

Serta menurut Jogiyanto (2005:12) mengemukakan bahwa “Aplikasi adalah penggunaan dalam suatu computer, intruksi (*intruction*) atau pernyataan (*statement*) yang disusun sedemikian rupa sehingga komputer dapat memproses *input* (masukan) menjadi *output* (keluaran)”.

Berdasarkan uraian di atas dapat disimpulkan bahwa, aplikasi adalah suatu program di dalam komputer yang dibuat untuk membantu pengguna dalam mengerjakan tugas khusus, dan biasanya disandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tetapi tidak secara langsung.

2.2 Implementasi

Menurut Usman (2002:70), “Implementasi adalah bermuara pada aktivitas, aksi, tindakan atau adanya mekanisme suatu sistem. Implementasi bukan sekedar aktivitas, tetapi suatu kegiatan yang terencana dan untuk mencapai tujuan kegiatan”.

Sedangkan menurut Setiawan (2004:39), “Implementasi adalah perluasan aktivitas yang saling menyesuaikan proses interaksi antara tujuan dan tindakan untuk mencapainya serta memerlukan jaringan pelaksana, birokrasi yang efektif”.

Dari pengertian implementasi yang dikemukakan di atas, dapat dikatakan bahwa implementasi merupakan suatu aktivitas yang sudah terencana, saling menyesuaikan antara proses interaksi antara tujuan dan tindakan untuk mencapai suatu tujuan kegiatan.

2.3 *Location Based Service*

Layanan berbasis lokasi adalah layanan informasi yang dapat diakses melalui *mobile device* dengan menggunakan *mobile network,* yang dilengkapi kemampuan untuk memanfaatkan lokasi dari *mobile device* tersebut. LBS memberikan kemungkinan komunikasi dan interaksi dua arah. (Safaat. 2013)

LBS (*Location Based Service*) merupakan suatu layanan yang bereaksi aktif terhadap perubahan entitas posisi sehingga mampu mendeteksi letak objek dan memberikan layanan sesuai dengan letak objek yang telah diketahui tersebut. Pada teknologi LBS berbasis jaringan selular, penentuan posisi sebuah peralatan komunikasi bergerak ditentukan berdasarkan posisi relative peralatan tersebut terhadap lokasi BTS (*Base Transceiver Station*). Terdapat dua unsur utama LBS, yaitu :

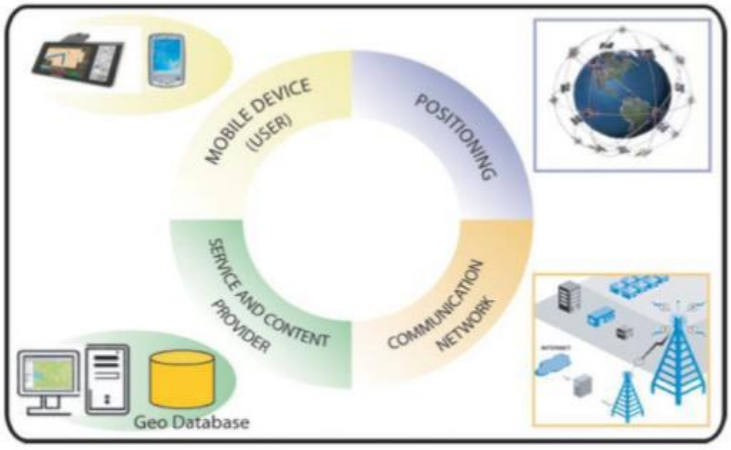
1. *Location Manager* (API *Maps*)

Menyediakan *tools/source* untuk LBS, *Application Programming Interface* (API) *Maps* menyediakan fasilitas untuk menampilkan, memanipulasi maps/peta beserta *feature-feature* lainnya seperti tampilan satelit, *street* (jalan), maupun gabungannya. Paket ini berada pada com.google.android.maps.

2. *Location Providers* (API *Location*)

Menyediakan teknologi pencarian lokasi yang digunakan oleh *device*/perangkat. API *Location* berhubungan dengan data GPS(*Global Positioning System*) dan data lokasi *real-time*. API *Location* berapa pada paket Android yaitu dalam paket android.location. dengan *Location Manager*, kita dapat menentukan lokasi kita saat ini, *track* gerakan/perpindahan, serta kedekatan dengan lokasi tertentu dengan mendeteksi perpindahan.

Menurut Safaat (2013) dalam menggunakan layanan berbasis lokasi elemen yang diperlukan antara lain :



Gambar 2.1 Komponen LBS (Imaniar, J., Arifin, Khalilullah, A.S., 2011)

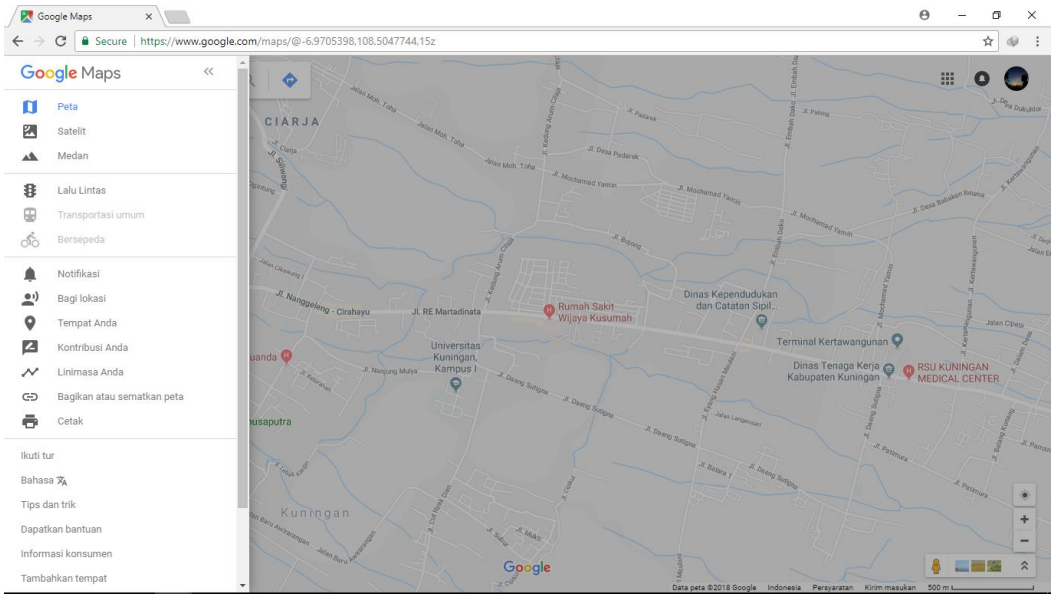
1. *Mobile Device* yaitu sebuah alat yang digunakan untuk meminta informasi yang dibutuhkan. Biasanya perangkat yang memungkinkan yaitu PDA, *Mobile Phone*, Laptop, dan perangkat lainnya yang mempunyai fasilitas navigasi.
2. *Communication Network* adalah jaringan seluler yang mengirimkan data pengguna dan permintaan layanan.
3. *Positioning Component* untuk pengolahan layanan biasanya posisi pengguna harus ditentukan. Posisi pengguna dapat diperoleh menggunakan jaringan komunikasi atau dengan menggunakan *Global Positioning System (GPS).*
4. *Service* dan *Application Provider* adalah penyedia layanan pengguna seluler yang bertanggung jawab untuk memproses layanan.
5. *Data and Content Provider* yaitu penyedia layanan informasi data yang dapat diminta oleh pengguna.

2.4 Google Maps

Menurut Riyanto (2010) Google Maps merupakan sebuah layanan peta dunia virtual berbasis web yang disediakan oleh Google. Layanan ini gratis dan dapat di temukan http://maps.google.com.

Google Maps menawarkan peta yang dapat digeser (*panned*), diperbesar (*zoom in*), diperkecil (*zoom out*), dapat diganti dalam beberapa mode (*map, satelit, hybrid*, dan lain-lain), fitur pencarian rute (*routing*), penunjuk arah dari satu objek peta ke objek lain (*direction*), dan juga pencari tempat (*place*) bisnis di Amerika, Kanada,

Jepang, Hongkong, Cina, Inggris, Irlandia (hanya pusat kota), dan beberapa bagian Eropa. Tampilan Google Maps pada *web browser* dapat dilihat pada Gambar 2.4



Gambar 2.2 Tampilan Google Map pada *Web Browser*

2.5 Hotel

Menurut Undang-Undang Nomor 10 Tahun 2009 Tentang Kepariwisataan Bab I Pasal 1 menyatakan ; Pariwisata adalah berbagai macam kegiatan wisata dan didukung berbagai fasilitas serta layanan yang disediakan oleh masyarakat, pengusaha, Pemerintah, dan Pemerintah Daerah. Selanjutnya pada Pasal 6 menyatakan ; Daerah tujuan pariwisata yang selanjutnya disebut Destinasi Pariwisata adalah kawasan geografis yang berada dalam satu atau lebih wilayah administrative yang di dalamnya terdapat daya tarik wisata, fasilitas umum, fasilitas pariwisata, aksesibilitas, serta masyarakat yang saling terkait dan melengkapi terwujudnya kepariwisataan.

Menurut Menteri Perhubungan, hotel adalah suatu bentuk akomodasi yang dikelola secara komersial, disediakan bagi setiap orang untuk memperoleh pelayanan penginapan berikut makan dan minum (SK. MenHub. RI. No. PM 10/PW.391/PHB-77). Menurut Sulastiyono (2008), "Hotel adalah suatu tempat dimana disediakan penginapan, makanan, dan minuman, serta pelayanan lainnya, untuk disewakan bagi para tamu atau orang-orang yang tinggal untuk sementara waktu".

2.6 *Unified Modeing Language* (UML)

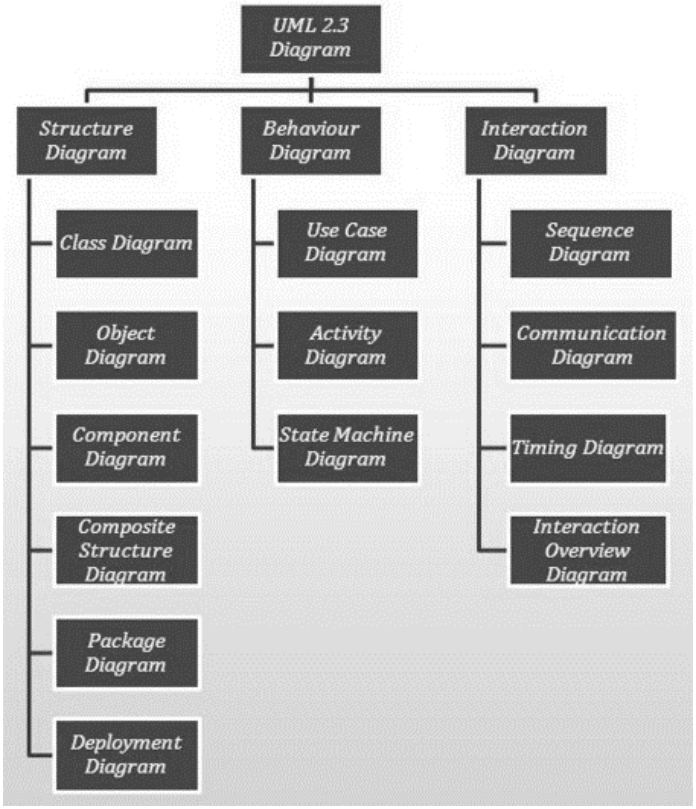
Menurut Bambang Hariyanto (2004:256), "UML adalah bahasa grafis untuk mendokumentasi, menspesifikasikan, dan membangun sistem perangkat lunak". UML berorientasi objek, menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi, pemaduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak, didukung oleh kakas-kakas yang diintegrasikan lewat XML (XMI).

Sebuah pernyataan "Sebuah standarisasi bahasa pemodelan untuk membangun perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML)". (Rosa. 2013:137)

Dari beberapa pengertian mengenai UML diatas, dapat disimpulkan bahwa UML merupakan bahasa pemodelan yang digunakan untuk menggambarkan pembangunan sistem berorientasi objek.

2.6.1 Diagram UML

Diagram UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sistem dengan menggunakan diagram dan teks-teks pendukung. (Rosa. 2013). Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar 2.1



Gambar 2.3 Diagram UML 2.3 (Rosa. 2013)

Berikut ini penjelasan singkat pembagian kategori pada gambar 2.1.

a. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk mengambarkan suatu struktur statis dari sistem yang dimodelkan.

b. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.

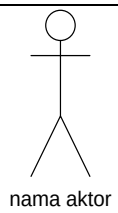
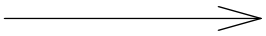
c. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

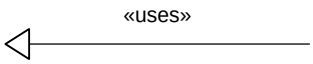
2.6.1.1 *Use Case* Diagram

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. (Rosa. 2013:155) Simbol-simbol pada *use case* dapat di lihat pada tabel 2.1

Tabel 2.1 simbol-simbol *use case*

Simbol	Deskripsi
Use case 	Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal di awal frase nama *use case*
Aktor/*actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol

Simbol	Deskripsi
	aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama actor
Asosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan actor
Ekstensi/*extend* <<extend>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya.

Simbol	Deskripsi
Menggunakan / *include* / *uses* <<include>>   	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

2.6.1.2 *Scenario Diagram*

Scenario use case adalah alur jalannya proses use case dari sisi aktor dan sistem. (Rosa. 2013) *Scenario use case* dapat dilihat pada tabel 2.2.

Tabel 2.2 *Scenario diagram*

Aksi actor	Reaksi aktor
Skenario normal	
Skenario alternatif	

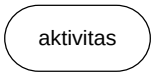
Scenario use case dibuat per *use case* terkecil, misalkan untuk generalisasi maka *scenario* yang dibuat adalah *use case* yang lebih khusus. Skenario normal adalah skenario bila sistem berjalan normal tanpa terjadi

kesalahan atau *error*. Sedangkan skenario alternatif adalah skenario bila sistem tidak berjalan normal, atau mengalami *error*. Skenario normal dan skenario alternatif dapat lebih dari satu. Alur dari skenario inilah yang nantinya menjadi dasar pembuatan diagram sekuen. (Rosa. 2013)

2.6.1.3 *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. (Rosa. 2013) Tabel 2.3 menunjukkan simbol-simbol yang ada pada diagram aktivitas.

Tabel 2.3 Simbol-simbol *activity diagram*

Simbol	**Deskripsi**
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / *join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu

Simbol	Deskripsi
Status akhir 	Status akhir yang dilakukan system, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane Nama swimlane Atau Nama swimlane	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

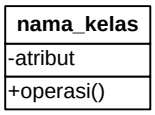
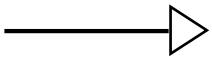
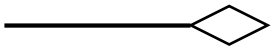
2.6.1.4 *Diagram Class*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki atribut dan metode atau operasi.

a. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. (Rosa. 2013)

Simbol-simbol yang ada pada diagram kelas dapat dilihat pada Tabel 2.4

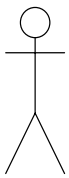
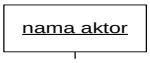
Tabel 2.4 Simbol-Simbol *Class Diagram*

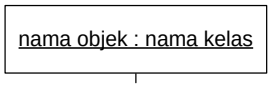
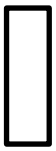
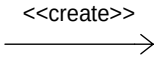
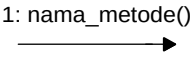
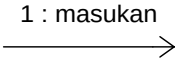
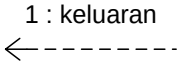
Simbol	**Deskripsi**
Kelas 	Kelas pada struktur sistem
Antarmuka / *interface* 	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah / *directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain , asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus)
Kebergantungan / *defendency* 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / *aggregation* 	Relasi antar kelas dengan makna semua-bagian (*whole-port*)

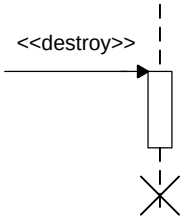
2.6.1.5 *Sequence Diagram*

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat scenario yang ada pada use case. (Rosa dan Shalahuddin, 2013) Simbol-simbol yang ada pada *sequence diagram* dapat dilihat pada Tabel 2.5

Tabel 2.5 Simbol-simbol *sequence diagram*

Simbol	**Deskripsi**
Aktor  nama aktor Atau  Tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat sendiri, jadi walaupun simbol dari actor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama actor
Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek

Simbol	Deskripsi
Objek 	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. Aktor tidak memiliki waktu aktif
Pesan tipe *create* 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
Pesan tipe *return* 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan arah panah mengarah pada objek tertentu.

Simbol	Deskripsi
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*.

2.7 *Tools* Perangkat Lunak dan Bahasa Pemrograman

Berikut adalah penjelasan *tools* perangkat lunak dan bahasa pemrograman yang akan digunakan penulis dalam membangun aplikasi lokasi penginapan hotel.

2.7.1 Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone*. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile dan Nvidia. (Safaat. 2015:1)

Android adalah sistem operasi dan *platform* pemrograman yang dikembangkan oleh Google untuk ponsel cerdas dan perangkat seluler lainnya. Android bisa berjalan di beberapa macam perangkat dari banyak produsen yang berbeda. Android menyertakan *kit development* perangkat lunak untuk

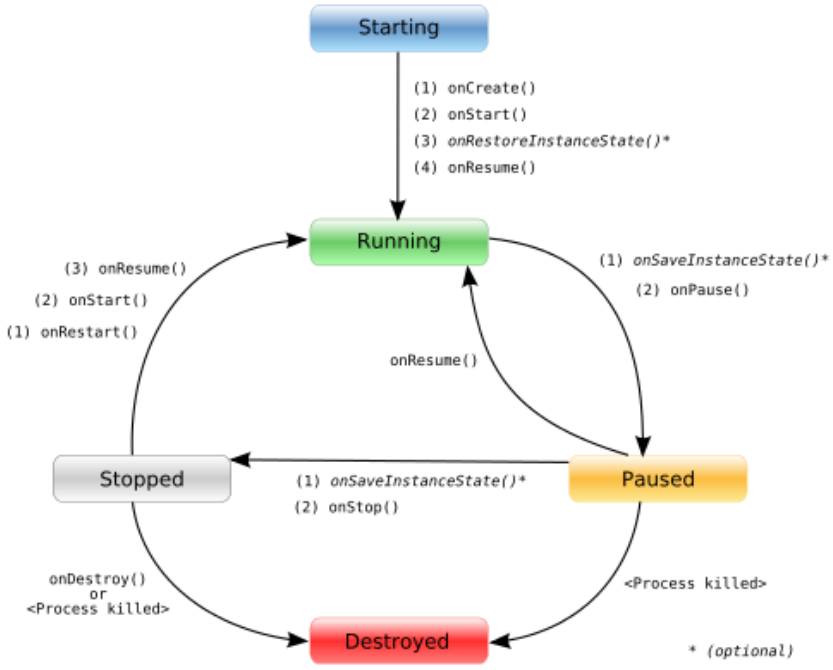
penulisan kode asli dan perakitan modul perangkat lunak untuk membuat aplikasi bagi pengguna Android.(Google Developer Training Team. 2016)

Dari dua pengertian Android diatas dapat di simpulkan bahwa Android merupakan sebuah sistem operasi yang berbasis linux yang terdapat di dalam *smartphone*, Android dapat berjalan pada perangkat dari banyak produsen selama itu masuk sebagai perangkat *mobile*. Terdapat beberapa versi Android yang penamaannya dengan abjad yang berurut, yaitu :

1. Android versi 1.1
2. Android versi 1.5 (*Cupcake*)
3. Android versi 1.6 (*Donut*)
4. Android versi 2.0/2.1 (*Eclair*)
5. Android versi 2.2 (*Froyo:Frozen Yoghurt*)
6. Android versi 2.3 (*Gingerbread*)
7. Android versi 3.0 (*Honeycomb*)
8. Android versi 3.1
9. Android versi 3.2
10. Android versi 4.0
11. Android versi 4.1
12. Android versi 4.2
13. Android versi 4.3 (Safaat. 2015)

2.7.1.1 Siklus Hidup Aplikasi Android

Menurut Arif (2013) Perangkat berbasis android hanya mempunyai satu layar antarmuka (interface). Setiap user interface diwakili oleh kelas *activity* (*activity class*). Setiap *activity* mempunyai siklus seperti yang dapat dilihat pada Gambar 2.4 Sebuah aplikasi dapat terdiri dari satu atau lebih *activity*.



Gambar 2.4 Siklus Hidup Aplikasi Android (Safaat. 2015)

Aplikasi Android terdiri dari 4 komponen, yaitu :

1. *Activity*

   Activity adalah bagian dari sebuah aplikasi yang dipakai untuk berinteraksi dengan pengguna aplikasi.

2. *Service*

   Service tidak memiliki *user interface*, namun berjalan secara *background*.

3. *Broadcast Receiver*

   Bagian ini dipakai untuk menerima isyarat dari sistem Android.

4. *Content Providers*

   Dengan *content providers*, data sebuah aplikasi bisa diakses atau digunakan dari aplikasi lainnya.

2.7.2 Java

Menurut Kadir. (2003:2) "Java adalah bahasa pemrograman serbaguna. Java dapat digunakan untuk membuat suatu program sebagaimana anda membuatnya dengan bahasa seperti Pascal atau ++. Yang lebih menarik, Java juga mendukung sumber daya Internet yang saat ini popular, yaitu *World Wide Web* atau yang sering disebut Web saja. Java juga mendukung aplikasi klien/server, baik dalam jaringan lokal (LAN) maupun jaringan berskala luas (WAN)".

Java menurut definisi dari Sun adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada computer *standalone* ataupun pada lingkungan jaringan. (Rosa. 2010:1)

Dari pengertian java diatas dapat ditarik kesimpulan bahwa Java merupakan sebuah bahasa pemrograman yang digunakan untuk membuat suatu program yang berbasis jaringan maupun *standalone*.

2.7.3 *Android* Studio

Gambar 2.5 Logo *Android* Studio

Android Studio merupakan lingkungan pengembanngan Android baru yang berdasarkan intelliJ IDEA. *Android* Studio ini mirip cara kerjanya dengan Eclipse + ADT *plugin*, *Android* Studio menyediakan alat pengembang android yang terintegrasi dalam pengembanngan dan debugging program. IntelliJ IDEA sendiri, merupakan sebuah komersial Java IDE powerfull layaknya Eclipse dan Netbeans yang dikembangkan oleh JetBrains. IDEA mendukung beberapa bahasa seperti : *Java, Java Script, HTML/XHTML/CSS, XML/XSL, ActionScript/MXML, Phyton, Ruby/JRuby, SQL* dan *PHP*. (Satyaputra. 2014)

Android Studio sebuah *Integrated Development Environment* (IDE) untuk platform *Android*. *Android* studio bersifat *free* dibawah *Apache Licence* 2.0. *Android* studio didesain khusus untuk *Android Development. Android* Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas saat membuat aplikasi *Android*, misalnya (https://developer.android.com) :

1. Mendukung pengembangan berbasis *Gradle*
2. Dapat menspesifikasikan pemfaktoran android dan melakukan perbaikan dengan cepat

3. Memiliki *Lint Tools* untuk menangkap kerja, penggunaan kompatibilitas versi dan masalah lainnya
4. Memiliki kemampuan penandaan aplikasi dan *ProGuard*
5. Memiliki kemampuan *wizard* berdasarkan template untuk membuat desain dan komponen android yang umum.
6. Sebuah layout editor yang kaya dan memungkinkan anda untuk melakukan *drag-drop* UI, pratinjau layout pada beberapa konfigurasi layer, dan banyak lagi.

2.7.4 *Android* SDK

Menurut Safaat (2015), Android SDK adalah tools API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware*, dan aplikasi kunci yang di release oleh Google.

Saat ini disediakan Android SDK sebagai alat bantu dan API untuk mulai mengembanngkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Sebagai *platform* aplikasi-netral, Android memberi anda kesempatan untuk membuat aplikasi yang dibutuhkan yang bukan merupakan aplikasi bawaan *Handphone/Smartphone.*

2.7.5 *Nox Player*

Nox Player merupakan android emulator gratis didedikasikan untuk memberikan pengalaman terbaik bagi pengguna dalam memainkan game dan aplikasi *Android* menggunakan PC atau Mac. Pengguna dapat menggunakan

keyboard dan *mouse* pada APK game dan aplikasi dengan mapping tombol yang mudah, serta kemudahan akses untuk berbagai fungsi seperti lokasi, setelan volume, dan banyak lagi lainnya. (https://id.bignox.com)

2.7.6 *MySQL*

MySQL adalah *multiuser database* yang menggunakan bahasa *Structured Query Language* (*SQL*). *MySQL* dalam operasi *client-server* melibatkan *server daemon MySQL* di sisi *server* dan berbagai macam program serta *library* yang berjalan di sisi *client*. *MySQL* mampu menangani data yang cukup besar. Perusahaan yang mengembangkan *MySQL* yaitu TcX, mengaku mampu menyimpan data lebih dari 40 database, 10.000 tabel dan sekitar 7 juta baris, totalnya kurang lebih 100 Gigabyte data. (Bimo. 2003:65)

Menurut website resmi *MySQL* menyebutkan, "*MySQL is the world's most popular open source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, Yahoo! and many more*." (https://www.mysql.com)

2.7.7 *XAMMP*

Xammp adalah sebuah *Software Web Server Apache* yang didalamnya sudah tersedia *database* server *MySQL* dan support PHP programming. *Xammp* merupakan *software* yang mudah digunakan, gratis dan mendukung instalasi pada *linux* dan *windows.* Keuntungan lainnya adalah menginstal satu kali sudah tersedia *Apache Web Server, MySQL, Database Server, PHP Support* (4 dan 5) dan beberapa modul lainnya.

Berikut ini adalah logo *Xammp* yang akan penulis gunakan:

Gambar 2.6 Logo *Xammp*

Xammp merupakan *tools* yang menyediakan paket perangkat lunak kedalam satu buah paket. Dengan menginstall *xammp* maka tidak perlu lagi melakukan instalasi dengan konfigurasi *Web Server Apache, PHP* dan *MySQL* secara manual secara manual. *Xammp* akan menginstalasi dan mengkonfigurasinya secara otomatis untuk anda atau auto konfigurasi. (Hakim, Lukmanul. 2008)

2.7.8 *PHP*

Menurut (Bimo. 2003:23) *PHP* adalah bahasa *server-side scripting* yang menyatu dengan HTML untuk membuat halaman *web* yang dinamis. Maksud dari *server-side scripting* adalah sintaks dan perintah-perintah yang diberikan akan sepenuhnya di jalankan di server tetapi disertakan pada dokumen HTML. Pembuatan *web* ini merupakan kombinasi antara PHP sendiri sebagai bahasa pemrograman dan HTML sebagai pembangun halaman *web*.

PHP atau *Hypertext Processor* adalah salah satu bahasa pemograman *open source* yang sangat cocok atau dikhusukan untuk pengembangan web dan dapat ditanamkan pada sebuah skrip HTML. (Hirin dan Virgi, 2011:25).

2.7.9 Rational Rose

Menurut Adi Nugroho (2005:20) *Rational Rose* berfungsi sebagai *tool* untuk pemodelan sistem yang menggambarkan proses-proses yang ada pada sistem ini. *Rational Rose* adalah kakas (*tools)* pemodelan visual untuk pengembangan system berbasis objek yang sangat handal untuk digunakan sebagai bantuan bagi para pengembang dalam melakukan analisis dan perancangan sistem.

Rational Rose digunakan untuk melakukan pemodelan sistem sebelum pengembang menulis kode-kode dalam Bahasa pemrograman tertentu. Ia juga membantu analisis sistem dengan cara pengembang membuat diagram *use case* untuk melihat fungsionalitas sistem secara keseluruhan sesuai dengan harapan dan keinginan pengguna.

2.7.10 Microsoft Visio 2013

Microsoft Visio adalah aplikasi utama untuk membuat semua diagram bisnis, mulai dari *flowchart*, *network diagram*, dan *organization charts*, untuk membuat denah dan *brainstorming diagram*. *Microsoft 2013* melanjutkan kegunaan dari kebiasaan user interface, atau dikenal sebagai keterkaitan, hal itu telah diperkenalkan pada *Visio 2010*. (Helmers. 2013:3)

Terlepas dari apa yang mungkin terpikirkan dari hubungannya dengan aplikasi *Microsoft Office* lainnya, dengan *Visio* rasanya seperti di rumah, terutama karena tujuan dari keterkaitan *user interface* gaya presentasi visual dari kelompok yang terkait fungsi, dan Visio termasuk didalamnya, pertama dan terutama, sebuah produk visual

2.8 Pengujian

Menurut Soetam Rizky (2011) menyimpulkan bahwa testing adalah sebuah proses yang diejawantahkan sebagai siklus hidup dan merupakan bagian dari proses rekayasa perangkat lunak secara integrasi demi memastikan kualitas dari perangkat lunak serta memenuhi kebutuhan teknis yang telah disepakati dari awal. Secara garis besar, terdapat dua jenis tipe testing yang paling umum digunakan di dalam lingkup rekayasa perangkat lunak. Dua jenis tersebut adalah *white box testing* dan *black box testing.*

2.8.1 *White Box*

White box testing secara umum merupakan jenis testing yang lebih berkonsentrasi terhadap "isi" dari perangkat lunak itu sendiri. (Soetam, 2011)

Menurut Bambang Hariyanto (2004) mengemukakan bahwa "Pengujian *white box* mengasumsikan bahwa logik spesifik adalah penting dan harus diuji untuk menjamin sistem melakukan fungsi dengan benar. Penggunaan utama *white box* adalah pengujian berbasis kesalahan ketika siap menguji semua objek di aplikasi dan semua metode eksternal atau publik dari objek. Pada pengujian *white box*, dicari *bug* yang mempunyai peluang eksekusi yang rendah, atau yang diimpelentasikan".

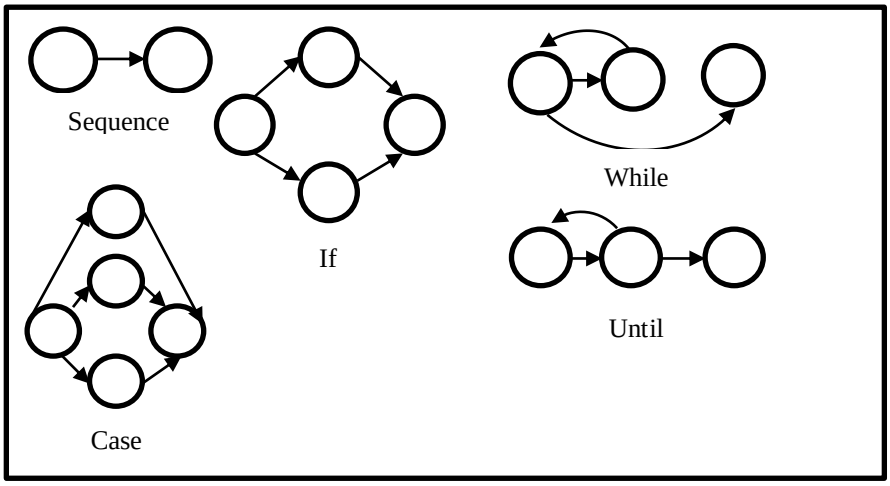
Pengujian kotak putih juga disebut pengujian struktur (*structural testing*). Sasaran dari pengujian ini adalah memeriksa semua pernyataan program. Adapun Teknik pengujian *white box* yaitu, antara lain:

1. *Basic path testing*

Metode ini melakukan pencarian ukuran kompleksitas perancangan prosedur dan menggunakannya untuk mendefinisikan sekumpulan jalur eksekusi dasar. Kasus uji diturunkan dari himpunan basis yang dijamin mengeksekusi seluruh pernyataan di program setidaknya satu kali selama pengujian. Menurut Romeo terdapat konsep utama *basis path,* yaitu:

a. Tiap basis path harus diidentifikasi, tidak boleh ada yang terabaikan (setidaknya dites 1 kali).
b. Kombinasi dan permutasi dari suatu *basis path* tidak perlu dites.

Alus logika suatu program dapat dipresentasikan dengan *flowgraph.* Berikut merupakan notasi *flowgraph* gambar 2.7:



Gambar 2.7 Notasi *flowgraph* (Romeo. 2003)

Kontruksi structural pada *flowgraph,* dimana tiap siklus melambangkan satu atau lebih pernyataan kode (*source code statement*). Panah pada *flowgraph,* disebut *edges* atau *links* (hubungan), mewakili alur pengiriman kendali dan merupakan analogi dari panah pada *flowchart*. Suatu *flowgraph* terbentuk dari:

a. *Nodes* (titik), mewakili pernyataan (sub program) yang akan ditinjau saat eksekusi program
b. *Edges* (anak panah), mewakili jalur alur logika program untuk menghubungkan satu pernyataan (sub program) dengan yang lainnya
c. *Branch nodes* (titik cabang), titik-titik yang mempunyai lebih dari satu anak panah keluaran
d. *Branch edges* (anak panah cabang), anak panah yang keluar dari suatu cabang
e. *Path* (jalur), jalur yang mungkin untuk bergerak dari satu titik kelainnya sejalan dengan keberadaan arah anak panah.

2. *Control structure testing*
   a. Conditional testing
   b. Data flow testing
   c. Loop testing

2.8.2 *Black Box*

Black box testing adalah tipe *testing* yang memperlakukan perangkat lunak yang tidak diketahui kinerja internalnya.(Soetam, 2011)

Salah satu teknik *testing* tipe *black box* yang akan dilakukan yaitu *Feature Test*, menurut Soetam, 2011 pada teknik ini, dilakukan proses *testing* terhadap spesifikasi dari perangkat lunak yang telah selesai dikerjakan. Misalnya pada perangkat lunak sistem informasi akademik. Dapat dicek apakah fitur untuk melakukan entri nilai telah tersedia, begitu dengan fitur entri data siswa maupun entri data guru yang akan melakukan entri nilai.