

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Tinjauan Studi**

Penelitian yang berhubungan dengan *Personal Computer* telah dibuat oleh Aditya Ardi Nugraha, Faizatul Amalia, dan Adam Hendra Brata pada tahun (2018), yang membahas tentang Pengembangan Media Pembelajaran Perakitan Komputer Dengan Menerapkan Metode Agile Software Development. Dalam pengembangan media pembelajarannya masih banyak SMK yang menerapkan media konvensional. Penelitian ini ditujukan untuk melakukan implementasi dan menguji media pembelajaran komputer yang telah dikembangkan. Pengembangan dengan menggunakan *agile software development* diharapkan dapat membantu mengembangkan media pembelajaran perakitan komputer yang ada. Karena metode *agile software development* memudahkan dan memberikan ruang bagi media pembelajaran agar dapat dikembangkan ke arah yang lebih luas lagi, dalam bentuknya sebagai media pembelajaran.

Penelitian yang berhubungan dengan *Personal Computer* telah dibuat oleh Jhon Veri dan Eko Prasetya pada tahun (2017), yang membahas tentang Perancangan Dan Pembuatan Aplikasi Media Pembelajaran Perakitan Komputer Berbasis *Android*. Dari hasil observasi yang dilakukan di SMK Negeri 6 Padang guru masih menyampaikan materi di depan kelas dengan sarana papan tulis dengan spidol dan juga menggunakan alat presentasi *proyektor* dengan menggunakan *slide power point* untuk memberikan contoh atau gambaran kepada siswa didiknya. Dengan penyampaian materi pelajaran seperti disebutkan diatas, kualitas ilmu yang tersampaikan kepada siswa cenderung monoton dan kurang maksimal. Salah satu media pembelajaran yang sering dipakai untuk mengatasi masalah rendahnya minat siswa dalam membawa buku ke sekolah adalah dengan pembuatan aplikasi yang di pasang pada *smartphone android*. Media

pembelajaran berbasis *android* merupakan salah satu media yang ekonomis dan efisien dibandingkan dengan media lainnya seperti *E-Learning* yang membutuhkan koneksi internet untuk mengaksesnya. Pembelajaran melalui media *smartphone* akan lebih praktis dilakukan dimana saja dan kapan saja sehingga dapat membuat siswa lebih mudah dalam belajar.

Penelitian yang berhubungan dengan *Personal Computer* telah dibuat oleh Stefanus Santosa dan April Firman Daru pada tahun (2016), yang membahas tentang Penerapan *Learning Technology System Architecture* (LTSA) Pada Multimedia Pembelajaran Perangkat PC. Dari hasil tes pembelajaran dapat dinyatakan bahwa metode pembelajaran perangkat PC menggunakan alat pengajaran berbasis multimedia dapat mendukung pembelajaran yang interaktif, menarik, efisien, efektif dan bermakna. Selain itu hasil tes juga menunjukkan perbedaan yang signifikan dibandingkan metode pengajaran konvensional. Siswa yang menggunakan sistem pembelajaran konvensional hanya memperoleh skor rata-rata 49,6. Sedangkan siswa menggunakan sistem pembelajaran dengan menggunakan alat pembelajaran dengan menggunakan alat pembelajaran animasi dan visualisasi memperoleh nilai rata-rata 80,09.

Dari tinjauan studi, peneliti tidak menemukan penelitian dengan judul yang sama seperti judul penelitian penulis. Namun peneliti mengangkat beberapa penelitian lain yang berkaitan sebagai referensi dalam memperkaya bahan kajian pada penelitian penulis. Terdapat perbedaan antara penelitian diatas dengan penelitian yang dibuat dalam tugas akhir ini yaitu penelitian ini akan membuat aplikasi *upgrade Personal Computer* berbasis *Android*. Aplikasi ini dapat memberitahu tipe-tipe komponen *Personal Computer* yang kompatibel antara satu sama lain. Metode pengumpulan data dilakukan dengan wawancara, studi pustaka. Model pengembangan perangkat lunak menggunakan model *waterfall*. Bahasa pemrograman yang digunakan yaitu *Java Android*.

## 2.2 Perancangan

Menurut Adi Nugroho (2004) Perancangan adalah analisis sistem, persiapan untuk merancangan dan implementasi agar dapat menyelesaikan apa yang harus diselesaikan serta mengkonfigurasi komponen-komponen perangkat lunak ke perangkat keras. Sedangkan perancangan menurut definisi lain yaitu strategi untuk memecahkan masalah dan mengembangkan solusi terbaik bagi permasalahan itu.

## 2.3 Aplikasi

Menurut Nazruddin Safaat (2012) Perangkat lunak aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media. Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau *suite* aplikasi (*application suite*).

### 2.3.1 Klarifikasi Aplikasi

Menurut Nazruddin Safaat (2012) Aplikasi dapat digolongkan menjadi beberapa kelas, antara lain:

- a. Perangkat lunak perusahaan (*enterprise*).
- b. Perangkat lunak infrastruktur perusahaan.
- c. Perangkat lunak informasi kerja.
- d. Perangkat lunak media dan hiburan.
- e. Perangkat lunak pendidikan.
- f. Perangkat lunak pengembangan media.
- g. Perangkat lunak rekayasa produk.

## 2.4 *Upgrade*

Menurut Irawan (2009) *Upgrade* adalah mengganti perangkat keras dengan model terbaru atau yang lebih baik. Istilah *upgrade* lebih cenderung digunakan pada bidang perangkat keras, terkadang juga untuk perangkat lunak.

## 2.5 **Personal Computer**

Menurut Sutarman (2009) Istilah komputer mempunyai arti yang luas dan berbeda untuk orang yang berbeda. Kata komputer (*computer*) berasal dari bahasa latin *computare* yang berarti menghitung. Berikut beberapa pengertian komputer adalah sebagai berikut:

a. Menurut buku *Computer Annual* (Robert H. Blissmer)

Komputer adalah suatu alat elektronik yang mampu melakukan beberapa tugas sebagai berikut:

1. Menerima *input*
2. Memproses *input* tadi sesuai dengan programnya
3. Menyimpan perintah-perintah dan hasil dari pengolahan
4. Menyediakan *output* dalam bentuk informasi

b. Menurut buku *Computer Today* (Donald H. Sanders)

Komputer adalah sistem elektronik untuk memanipulasi data yang cepat dan tepat serta dirancang dan diorganisasikan agar secara otomatis menerima dan menyimpan data input, memprosesnya, dan menghasilkan output di bawah pengawasan suatu langkah-langkah instruksi program yang tersimpan pada memori.

c. Menurut buku *Computer Organization* (V.C. Hamacher, ZG. Vranesic, S.G. Zaky)

Komputer adalah mesin penghitung elektronik yang dengan cepat dapat menerima informasi input digital, memprosesnya sesuai dengan suatu program yang tersimpan di memorinya dan menghasilkan output informasi.

- d. Menurut buku *Introduction to the Computer, The Tool of business* (William M. Fuori)

Komputer adalah suatu pemroses data yang dapat melakukan perhitungan yang besar dan cepat, termasuk perhitungan aritmatika yang besar atau operasi logika, tanpa campur tangan dari manusia mengoperasikan selama pemrosesan.

- e. Menurut buku *Introduction to Computers* (Gordon B. Davis)

Komputer adalah tipe khusus alat penghitung yang mempunyai sifat tertentu yang pasti.

Berdasarkan beberapa pengertian diatas, dapat disimpulkan bahwa komputer adalah :

- a. Alat elektronik yang dapat melakukan perhitungan *numeric*
- b. Alat yang dapat membaca input data dan mengolahnya sesuai dengan program yang ditetapkan untuk menghasilkan informasi yang merupakan output hasil pemrosesan input data.
- c. Alat yang dapat melakukan penyimpanan data, yaitu program, input, maupun output hasil pengolahan.
- d. Alat yang bekerja secara otomatis sesuai dengan aturan yang sudah ditetapkan dalam program.

### **2.5.1 Sistem Komputer**

Menurut Sutarman (2009) Komputer dapat melakukan rangkaian pekerjaan secara otomatis melalui instruksi (program) yang diberikan, dan alat pengolah data menjadi informasi melalui proses tertentu. Agar komputer dapat digunakan untuk mengolah data, maka harus berbentuk sistem komputer. Tujuan pokok dari sistem komputer adalah mengolah data untuk menghasilkan informasi. Agar tujuan pokok tersebut terlaksana, maka harus ada elemen-elemen yang mendukungnya. Berikut ini elemen-elemen dari sistem komputer, yaitu:

- a. *Hardware* (perangkat keras/ piranti keras) adalah peralatan pada sistem komputer yang secara fisik terlihat dan dapat disentuh.

- b. *Software* (perangkat lunak/ piranti lunak) adalah program yang berisi perintah-perintah untuk melakukan pengolahan data.
- c. *Brainware* adalah manusia yang terlibat di dalam pengoperasian serta pengaturan sistem komputer.

Ketiga elemen sistem komputer tersebut harus saling berhubungan dan membentuk satu kesatuan. *Hardware* tanpa adanya *software*, tidak akan berfungsi sesuai dengan yang diharapkan, hanya berupa benda mati saja. *Software* akan mengoperasikan *hardware*. *Hardware* yang sudah didukung oleh *software* juga tidak akan berfungsi jika tidak ada manusia yang mengoperasikannya. Akan tetapi jika ketiga elemen ini telah dikombinasikan sesuai dengan fungsinya masing-masing, maka akan terjadi suatu proses yang akan menghasilkan suatu informasi sesuai dengan yang diharapkan.

### 2.5.2 Keunggulan Komputer

Sutarman (2009) menjelaskan, Komputer sebagai produk teknologi memiliki keunggulan, antara lain:

- a. Mampu mengakses dengan cepat dan tepat.
- b. Menghasilkan informasi dari data yang telah lama.
- c. Mampu memproses data yang sangat besar menjadi informasi.
- d. Mampu menyimpan data yang sangat banyak (sampai berukuran *tera byte*).
- e. Mampu melakukan importing dan exporting data yang dirancang secara khusus.

Komputer dapat bekerja menurut perintah manusia yang menggunakannya. Manusia memberi perintah kepada komputer dengan menggunakan bahasa manusia. Kemudian *interpreter* / *compiler* (penterjemah) akan membantu untuk menerjemahkan bahasa manusia ke bahasa mesin yang dimengerti oleh komputer. Selanjutnya komputer akan bekerja sesuai dengan perintah dan jika telah selesai akan menghasilkan informasi.

## 2.6 *Android*

Menurut Nazruddin Safaat (2012) *Android* adalah sistem operasi yang berbasis *linux* atau *open source*. Selain *Android* SDK untuk pengembangan aplikasi, *android* juga tersedia bebas dalam bentuk *operating sistem*, hal ini sebenarnya yang menyebabkan vendor-vendor *smartphone* berlomba-lomba untuk memproduksi *smartphone* dan *tablet* PC berbasis *android*, *Android OS* dapat di *download* dari situs resmi *google*. Inilah yang menjadi peluang besar bagi vendor-vendor *smartphone* dan *tablet* PC untuk memproduksi *smartphone* dan *Tablet* PC *Android*. *Android OS* 3.0 yang dapat di *download* dari situs resminya tersebut sudah banyak digunakan oleh *table pc* yang banyak beredar di pasaran. Dan sekarang *Android* sudah menyebar bukan hanya di *smartphone* tetapi juga di *tablet/gadget pc*.

### 2.6.1 Sejarah *Android*

Perjalanan *Android* dimulai sejak Oktober 2003 ketika 4 orang pakar IT, Andi Rubin, Rich Minner, Nick Sears dan Chris White mendirikan *Android.Inc*, di California US. Visi *Android* untuk mewujudkan *mobile device* yang lebih peka dan mengerti pemiliknya, kemudian mengakuisisi *Android* pada Agustus 2005. OS *Android* dibangun berbasis *platform* Linux yang bersifat *open source*, senada dengan Linux, *Android* juga bersifat *Open Source*. Dengan nama besar *Google* dan konsep *open source* pada OS *Android*, tidak membutuhkan waktu lama lagi *android* untuk bersaing dan menyisihkan *Mobile OS* lainnya seperti *Symbian*, *Windows Mobile*, *Blackberry* dan *iOS*. Kini siapa yang tak kenal *Android* yang telah menjelma menjadi penguasa *Operating System* bagi *Smartphone*.

### 2.6.2 Arsitektur *Android*

Secara garis besar arsitektur *Android* dapat dijelaskan dan digambarkan sebagai berikut :

a. *Applications dan Widgets*

*Application* dan *Widgets* ini adalah *layer* dimana berhubungan dengan aplikasi saja, dimana biasanya kita *download* aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut, di *layer* inilah terdapat aplikasi inti termasuk *clien email*, program SMS, kalender, peta, *browser*, kontak dan lain-lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.

b. *Applications Frameworks*

*Applications Frameworks* ini adalah *layer* dimana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem android, karena pada *layer* inilah aplikasi dapat dirancang, seperti *content-providers* yang berupa SMS dan lain sebagainya.

Komponen-komponen yang termasuk di dalam *Applications Frameworks* adalah sebagai berikut:

1. *Views*
2. *Content Provider*
3. *Resource Manager*
4. *Notification Manager*
5. *Activity Manager*

c. *Libraries*

*Libraries* ini adalah *layer* dimana *feature-feature* Android berada, biasanya para pembuat aplikasi kebanyakan mengakses *Libraries* untuk menjalankan aplikasinya. Berjalan di atas *kernel*, *layer* ini meliputi berbagai *library* C/C++ inti seperti *Libc* dan *SSL*, serta:

1. *Libraries* media untuk pemutaran media *audio* dan *video*.
2. *Libraries* untuk manajemen tampilan.
3. *Libraries Graphics* mencakup *SGL* dan *OpenGL* untuk grafis 2D dan 3D.
4. *Libraries SSL* dan *WebKit* terintegrasi dengan *web browser* dan *security*.



5. *Libraries LiveWebcore* mencakup modem *web browser* dengan *engine embedded web view*.

6. *Libraries 3D* yang mencakup implementasi *OpenGL ES 1.0 API's*.

d. *Android Run Time*

Layer yang membuat aplikasi Android dapat dijalankan dimana dalam prosesnya menggunakan implementasi linux. *Dalvik Virtual Mechine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi android.

Didalam *Android Run Time* dibagi menjadi dua bagian yaitu:

1. *Core Libraries*: Aplikasi android yang dibangun dalam bahasa java, sementara *Dalvik* sebagai *virtual* mesinnya bukan *virtual* mesin java, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menterjemahkan bahasa java/c yang ditangani oleh *core libraries*.
2. *Dalvik Virtual Mechine*: *Virtual* mesin yang berbasis *register* yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat *linux kernel* untuk *threading* dan manajemen tingkat rendah.

e. *Linux Kernel*

*Linux Kernel* adalah layer dimana inti dari *operating system* dari android itu sendiri, berisi *file-file* sistem yang mengatur *system processing*, *memory*, *resource*, *drivers*, dan sistem-sistem operasi android lainnya.



**Gambar 2. 1** Arsitektur *Android*

Menurut Nazruddin Safaat (2012) Di lapisan teratas ada aplikasi itu sendiri. Di lapisan inilah kamu akan menemukan fungsi-fungsi dasar smartphone seperti menelepon dan mengirim pesan singkat, menjalankan *web browser*, mengakses daftar kontak, dan lain-lain. Inilah lapisan yang akan sering diakses oleh para pengguna. Mereka mengakses fungsi-fungsi dasar tersebut melalui *user interface*.

### 2.6.3 Fitur *Android*

Menurut Nazruddin Safaat (2012) *Android* merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di *release* oleh *Google*. Saat ini disediakan *Android SDK (software Development kit)* sebagai alat bantu dan API diperlukan untuk mulai mengembangkan aplikasi pada platform *Android* menggunakan bahasa pemrograman *Java*.

- a. *Application framework* mendukung pemakaian komponen-komponen yang mudah diganti dan digunakan kembali.

- b. *Dalvik virtual machine* dioptimalkan untuk perangkat *mobile*.
- c. *Integrated browser* dirancang berdasarkan *engine browser open source Webkit*.
- d. *Optimized graphics* didukung oleh *custom 2D graphics library*, *3D graphics* berdasarkan *OpenGL ES 1.0 spesifcation* (dukungan *hardware acceleration optional*).
- e. *SQLite* untuk penyimpanan *database* terstruktur.
- f. *Media support* untuk mendukung berbagai *audio*, *video*, dan *still image formats* (*MPEG4*, *H.264*, *MP4*, *AAC*, *AMR*, *JPG*, *PNG*, *GIF*).
- g. *GSM Telephony*, *Bluetooth*, *EDGE*, *3G*, and *WiFi* (*hardware dependant*).
- h. Kamera, GPS, *compass*, and *accelerometer* (*hardware dependant*).
- i. *Rich development environment* meliputi *device emulator*, *tools* untuk *debugging*, *memory* dan *performance pro filling*, serta plugin untuk *Eclipse IDE*.

#### **2.6.4 Kelebihan Dan Kekurangan OS Android**

Tentunya setiap OS memiliki kelebihan dan kekurangan masing-masing. Karena tidak ada yang *software* yang berjalan 100% sempurna.

##### **2.6.4.1 Kelebihan OS Android**

Menurut Nazruddin Safaat (2012) Karena *Android* sendiri adalah OS yang berbasis *Linux* jadi dapat dengan mudah dikembangkan oleh siapa saja.

- a. Akses yang mudah ke *Android App Market*. Di *Android App Market* bisa *download game* dan aplikasi yang mendukung *Android*. Fasilitas ini mirip seperti *Apple Store* di *Apple iOS*.
- b. Mudah dijangkau masyarakat. Tidak seperti *iOS* yang khusus pada *Apple Phone* saja, *Android* terdapat di banyak *platform-platform ponsel*. Mulai dari *Nokia*, *Samsung*, *SE*, sampai *HTC*.
- c. Fasilitas penuh *USB*. Kamu bisa mengganti baterai, *mass storage*, *disk drive*, dan *USB tethering*.

- d. *Easy Notification*. Kamu tidak akan terlewat satupun SMS, *Miss Call*, dan *Email*, bahkan artikel terbaru dari *RSS Reader* yang masuk.
- e. Mendukung Semua Layanan *Google*. Karena diciptakan oleh *google*, Android sangat mendukung layanan-layanan *Google*. Mulai dari *Gmail*, *Google+*, *Google Map*, *Google Talk*, sampai *Google reader*.
- f. *Install ROM* modifikasi. Kita kadang mendapati *ROM* yang tidak resmi. Maksudnya adalah versi yang telah rilis tidak sesuai dengan spesifikasi ponsel kita, jalan terakhir kita adalah modifikasi. Jangan khawatir ada banyak *custom ROM* yang bisa Anda pakai di ponsel Android, dan dijamin tidak akan membahayakan perangkat anda.

#### **2.6.4.2 Kekurangan OS Android**

Menurut Nazruddin Safaat (2012) selain kelebihan, *Android* juga mempunyai beberapa kekurangan diantaranya:

- a. Selalu *Online*. *Android* sangat membutuhkan *internet* untuk menunjang kinerjanya, setidaknya harus ada koneksi dengan jaringan GPRS saja di tempat kamu.
  - b. Terkadang Perusahaan perangkat lambat dalam mengeluarkan versi resmi dari *Android* milik anda. Meskipun kadang tidak ada perbedaan mencolok dalam hal *UI (User Interface)*.
  - c. *Android Market App* terkadang kurang mendapat dari pengelolanya. Jadi kadang masih terdapat *malware*.
  - d. Sebagai penyedia layanan langsung, terkadang pengguna sangat sulit sekali terhubung dengan pihak *Google*.
  - e. Karena aplikasinya yang mudah dan gratis, terkadang terdapat iklan di atas atau bawah tampilan aplikasi. Tetapi tidak begitu mengganggu.
- Boros Baterai. Dikarenakan *Android* menjalankan banyak program di bawahnya, *Android* menjadi lebih boros daripada *OS* lainnya.

### 2.6.5 Versi *Android*

Semakin berkembangnya zaman versi *android* pun semakin meningkat, yang dimulai dari android 1.5 *Cupcake* hingga saat ini. Berikut versi-versi *android*:

a. *Android 1.5 Cupcake*

*Cupcake* dirilis 30 April 2009. *Cupcake* menjadi versi *android* pertama yang menggunakan nama makanan. Konon katanya versi ini seharusnya versi 1.2, namun *Google* memutuskan untuk membuat revisi besar dan membuatnya menjadi versi 1.5 *Cupcake* adalah kue kecil yang dipanggang dalam cetakan berbentuk *cup*.

b. *Android 1.6 Donut*

*Android V1.6, codename Donut*, dirilis pada 15 September 2009. Pada versi ini diperbaiki beberapa kesalahan *reboot*, perubahan fitur foto dan video dan integrasi pencarian yang lebih baik. Donat merupakan panganan berbentuk cincin bulat bolong tengah. Adonan donat dimasak dengan cara digoreng dan biasanya disajikan dengan topping di atasnya.

c. *Android 2.0/2.1 Eclair*

*Android 2.0/2.1 Eclair* dirilis 26 Oktober 2009. *Eclair* adalah makanan penutup yakni kue yang biasanya berbentuk persegi panjang yang dibuat dengan krim di tengah dan lapisan cokelat di atasnya.

d. *Android 2.2 Froyo*

Dirilis 20 Mei 2010. Menggunakan *codename Froyo*, yang merupakan makan penutup yang nama merek sebuah produk yang terbuat dari *Yoghurt*. *Froyo* singkatan dari *Frozen Yoghurt*, *Froyo* adalah *yoghurt* yang telah mengalami proses pendinginan sehingga secara terlihat sama seperti es krim.

e. *Android 2.3 Gingerbread*

*Android* versi 2.3 *Gingerbread* dirilis resmi tanggal 6 Desember 2010. *Gingerbread* merupakan jenis kue kering yang dengan rasa jahe. Kue jahe biasanya dibuat pada perayaan hari libur akhir tahun di Amerika. Biasanya cemilan kering ini dicetak berbentuk tubuh manusia.

f. *Android 3.0 Honeycomb*

Dirilis tanggal 22 Februari 2011. *Honeycomb* adalah sereal sarapan manis yang sudah dibuat tahun 1965 oleh *Posting Sereal*. Seperti namanya, *Honeycomb*/sarang lebah, sereal ini terbuat dari potongan jagung berbentuk sarang lebah dengan rasa madu.

g. *Android 4.0 Ice Cream Sandwich*

*Android 4.0-4.0.2* API Level 14 dan *4.0.3-4.0.4* API Level 15 pertama dirilis 19 Oktober 2001. Dinamai *Ice Cream Sandwich*. *Ice Cream Sandwich* adalah lapisan es krim, biasanya rasa *vanilla* yang terjepit di antara dua kue coklat, dan biasanya berbentuk persegi panjang.

h. *Android 4.1 Jelly bean*

*Android Jelly Bean* diluncurkan pertama kali pada Juli 2012, dengan berbasis *Linux Kernel 3.0.31*. Terdiri dari *Android 4.1* API Level 16, *Android 4.2* API Level 17, *Android 4.3* API Level 18. Penamaan *Jelly Bean* mengadaptasi nama sejenis permen dalam beraneka macam rasa buah. Ukurannya sebesar kacang merah. Permen ini keras di luar tapi lunak di dalam serta lengket bila di gigit.

i. *Android 4.4 KitKat*

*Android 4.4 KitKat* API Level 19. *Google* mengumumkan *Android KitKat* (dinamai dengan izin dari *Nestle* dan *Hershey*) pada 3 September 2013. Dengan tanggal rilis 31 Oktober 2013. *KitKat* merupakan merk sebuah coklat yang dikeluarkan oleh *Nestle*. Rilis berikutnya setelah nama *KitKat* diperkirakan banyak pengamat akan diberi nomor 5.0 dan dinamai '*Key Lime Pie*'.

j. *Android 5.0 Lollipop*

*Android 5.0 Lollipop* pertama kali diperkenalkan pada Mei 2014. *Android lollipop* merupakan perancangan ulang terbesar untuk *Android*. *Smartphone Google Nexus 6*, bersama dengan *tabletNexus 9*-nya, merupakan perangkat pertama yang memiliki *Lollipop* yang telah terpasang sebelumnya. Peningkatan terbesar yang dilakukan oleh *Lollipop* adalah pengenalan *Material Design* yang dengan cepat menjadi bahasa desain terpadu

yang diterapkan di seluruh produk *Google*. Berikut adalah fitur yang dimilikinya yaitu, dukungan pengaturan cepat yang lebih baik, masa pakai baterai yang disempurnakan dengan *mode Battery Saver* yang baru, Layar kunci baru, *Fitur Smart Lock* melalui Layanan *Google Play*, *Mode* tamu untuk berbagi perangkat, Pemasangan tombol.

k. *Android 6.0 Marshmallow*

*Android 6.0 (Marshmallow)* dirilis pada tahun 2015. Ini Perangkat pertama yang dikirim bersama *Marshmallow* yang telah terpasang sebelumnya adalah smartphone *Google Nexus 6P* dan *Nexus 5X*, dengan *tablet Pixel C*-nya. Tujuan *marshmallow* memoles sudut kasar dan membuat *versi Lollipop* lebih baik lagi. Berikut adalah fitur yang dimilikinya yaitu, Dukungan sidik jari resmi untuk perangkat, dukungan untuk pembayaran seluler melalui *Android Pay*, Model perizinan yang lebih baik untuk aplikasi, *Google Now* di *Tap*, *Deep* menghubungkan Aplikasi.

l. *Android 7.0 Nougat*

*Android 7.0 (Nougat)* Dirilis pada Tahun ,2016. Sebelum *Nougat* terungkap "*Android N*" dirujuk secara internal oleh *Google* sebagai "*New York Cheesecake*". Berikut adalah fitur yang dimilikinya yaitu, *Doze on the Go* untuk waktu siaga yang lebih baik lagi, *Multi Window* untuk penggunaan dua aplikasi secara bersamaan, aplikasi setelan yang lebih baik, hapus semua di layar aplikasi baru-baru ini, Balas langsung ke pemberitahuan, notifikasi dibundel, pengaturan cepat akan mengubah kustomisasi.

m. *Android 8.0 Oreo*

Pada bulan Maret 2017, *Google Rilis Android 8.0 Oreo*, bulan Agustus, *Google* mengkonfirmasi *Oreo* akan menjadi nama publik untuk *Android 8.0*. Sperti yang kita ketahui Ini adalah kedua kalinya *Google* memilih nama merek dagang untuk *Android (Oreo* dimiliki oleh *Nabisco*). Adapun sekarang *versi* ini adalah semua yang baru yang dimilikinya, diantaranya adalah sebagai berikut yaitu, pemberitahuan untuk prioritas dan kategorisasi yang lebih baik, pengelolaan warna lebih

baik, *Android O* memiliki koleksi *emoji* baru yang telah didesain ulang, waktu *boot* lebih cepat: Pada perangkat *Pixel*, sekarang bisa mengalami waktu *boot* dua kali lebih cepat dibandingkan dengan *Nougat*, mengisi otomatis dan mengingat kata sandi dalam aplikasi.

#### **2.6.6 Android SDK**

Menurut Nazruddin Safaat (2012) *Android SDK (Software Development Kit)* menyediakan *tools* dan *API (Application Programming Interface)* yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* android menggunakan bahasa pemrograman Java.

#### **2.6.7 Android Development Tools (ADT)**

Menurut Nazruddin Safaat (2012) *Android Development Tools (ADT)* didesain untuk memudahkan merancang dan mengembangkan aplikasi *android* sehingga aplikasi yang dihasilkan nantinya dapat di gunakan oleh *user* dengan mudah. Android juga dapat mendistribusikan *package* dengan merubahnya menjadi (.apk) agar dapat terpasang di telepon genggam android secara mudah oleh pengguna. ADT juga memungkinkan *Android studio* untuk membuat aplikasi *android* baru, lalu membuat *User Interface*, kemudian menambahkan komponen berdasarkan *framework API Android*, serta *debug* aplikasi, dan pemaketan aplikasi *Android*.

#### **2.6.8 Android Virtual Device (AVD)**

Menurut Nazruddin Safaat (2012) *Android Virtual Device (AVD)* merupakan *emulator* untuk membantu menjalankan aplikasi android yang sudah dirancang. Tiap *emulator* mempunyai versi yang sesuai dengan versi android yang sudah ada saat ini.



## 2.7 Model Pengembangan Perangkat Lunak *Waterfall*

Menurut Sommerville (2011) Dalam pengembangan perangkat lunak pada penelitian ini menggunakan model *waterfall* yang merupakan salah satu model dari metode *System Development Life Cycle* (SDLC). Model *Waterfall* ini merupakan contoh dari sebuah proses yang bersifat *plan-driven* dimana semua aktivitas yang akan dilakukan harus direncanakan terlebih dahulu sebelum mengerjakannya. Sommerville (2011) menjelaskan bahwa, Model *Waterfall* memiliki tahapan-tahapan sebagai berikut:

### a. *Requirements analysis and definition*

Layanan sistem, kendala, dan tujuan ditetapkan oleh hasil konsultasi dengan pengguna yang kemudian didefinisikan secara rinci dan berfungsi sebagai spesifikasi sistem.

### b. *System and software design*

Tahapan perancangan sistem mengalokasikan kebutuhan-kebutuhan sistem baik perangkat keras maupun perangkat lunak dengan membentuk arsitektur sistem secara keseluruhan. Perancangan perangkat lunak melibatkan identifikasi dan penggambaran abstraksi sistem dasar perangkat lunak dan hubungannya.

### c. *Implementation and unit testing*

Pada tahap ini, perancangan perangkat lunak direalisasikan sebagai serangkaian program atau unit program. Pengujian melibatkan verifikasi bahwa setiap unit memenuhi spesifikasinya.

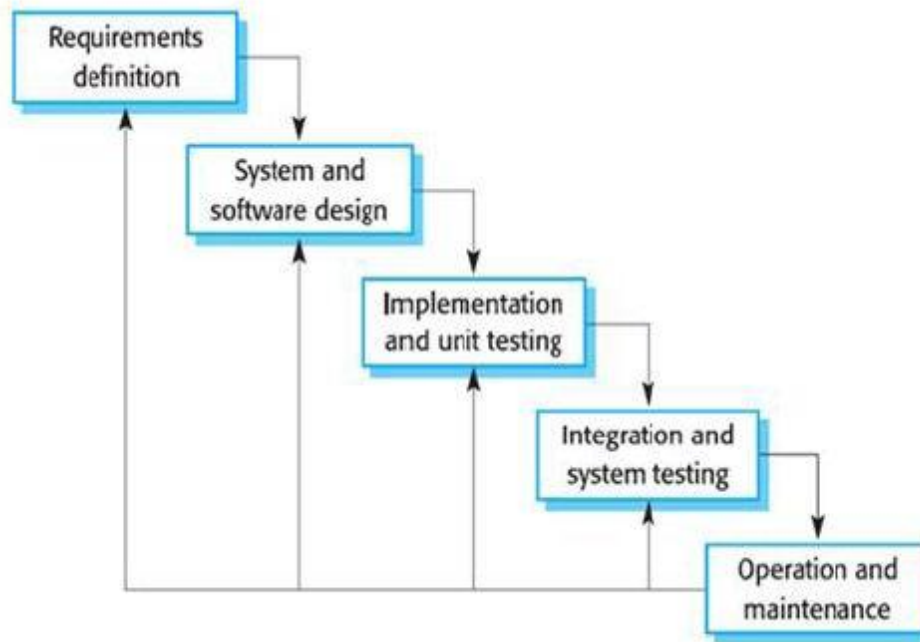
### d. *Integration and system testing*

Unit-unit individu program atau program digabung dan diuji sebagai sebuah sistem lengkap untuk memastikan apakah sesuai dengan kebutuhan perangkat lunak atau tidak. Setelah pengujian, perangkat lunak dapat dikirimkan ke *customer*.

### e. *Operation and maintenance*

Biasanya (walaupun tidak selalu), tahapan ini merupakan tahapan yang paling panjang. Sistem dipasang dan digunakan secara nyata. *Maintenance* melibatkan pembetulan kesalahan yang tidak ditemukan pada tahapan-tahapan sebelumnya,

meningkatkan implementasi dari unit sistem, dan meningkatkan layanan sistem sebagai kebutuhan baru.



**Gambar 2. 2** Model Waterfall

## 2.8 Java

Menurut Budi dkk (2007) Java adalah bahasa pemrograman yang disusun oleh James Gosling yang dibantu oleh rekan-rekannya seperti Patrick Naughton, Chris Warth, Ed Frank, dan Mike Sheridan di suatu perusahaan perangkat lunak yang bernama Sun Microsystems, pada tahun 1991. Bahasa pemrograman ini mula-mula diinisialisasi dengan nama “Oak”, namun pada tahun 1995 diganti namanya menjadi “Java”.

Alasan utama pembentukan bahasa Java adalah untuk membuat aplikasi-aplikasi yang dapat diletakkan diberbagai macam perangkat elektronik, seperti *microwave oven* dan *remote control*, sehingga Java harus bersifat portable atau yang sering disebut dengan *platform independent* (tidak tergantung pada *platform*). Itulah yang menyebabkan dalam dunia pemrograman Java, dikenal adanya istilah ‘*write once, run everywhere*’, yang berarti kode program hanya ditulis sekali, namun dapat

dijalankan di bawah *platform* manapun, tanpa harus melakukan perubahan kode program.

### 2.8.1 Arsitektur Java

Secara arsitektur, Java tidak berubah sedikit pun semenjak awal mula bahasa tersebut dirilis. Kompiler Java (yang disebut dengan **javac** atau *Java Compiler*) akan mentransformasikan kode-kode dalam bahasa Java ke dalam suatu *bytecode*. Apa itu *bytecode*? *Bytecode* adalah sekumpulan perintah hasil kompilasi yang kemudian dapat dieksekusi melalui sebuah mesin komputer abstrak, yang disebut dengan JVM (*Java Virtual Machine*). JVM juga sering dinamakan sebagai interpreter, karena sifatnya yang selalu menerjemahkan kode-kode yang tersimpan dalam *bytecode* dengan cara baris demi baris.

### 2.8.2 Java Versi Lama (Java 1)

Pada awal perilisannya, versi Java masih disebut dengan JDK (*Java Development Kit*). Dalam JDK, semua kebutuhan untuk pengembangan program dan eksekusi program masih tergabung jadi satu. Penamaan ini berlaku sampai Java 1.1. Namun sekarang, setelah Java 1.2, *Sun Microsystems* menamainya dengan JSDK (*Java Software Development Kit*) dalam hal ini kebutuhan untuk pengembangan program dipisahkan dengan kebutuhan eksekusi. Bagian *software* yang digunakan untuk kebutuhan eksekusi program disebut dengan JRE (*Java- Runtime Enviroment*). Selanjutnya, Java 1.2 disederhanakan penamaannya menjadi “Java 2”.

### 2.8.3 Java 2

*Sun Microsystems* telah mendefinisikan tiga buah edisi dari Java 2, yaitu sebagai berikut:

- a. Java 2 *Standard Edition* (J2SE), yang digunakan untuk mengembangkan aplikasi-aplikasi desktop dan applet (aplikasi Java yang dapat dijalankan di dalam browser web).

- b. Java 2 *Enterprise Edition* (J2EE), merupakan superset dari J2SE yang memperbolehkan kita untuk mengembangkan aplikasi-aplikasi berskala besar (*enterprise*), yaitu dengan melakukan pembuatan aplikasi-aplikasi di sisi server dengan menggunakan EJBs (*Enterprise JavaBeans*), aplikasi web dengan menggunakan *Servlet* dan JSP (*JavaServer Pages*) dan teknologi lainnya seperti COBRA (*Common Object Request Broker Architecture*) dan XML (*Extensible Markup Language*).
- c. Java 2 *Micro Edition* (J2ME), merupakan subset dari J2SE yang digunakan untuk menangani pemrograman di dalam perangkat-perangkat kecil, yang tidak memungkinkan untuk mendukung implementasi J2SE secara penuh.

#### 2.8.4 Fitur Java

Terdapat beberapa fitur java, yaitu sebagai berikut:

- a. *Applet*

Program Java yang dapat berjalan di atas *browser*, yang dapat membuat halaman HTML lebih dinamis dan menarik.

- b. *Java Networking*

Sekumpulan API (*Application Programming Interface*) yang menyediakan fungsi – fungsi untuk aplikasi – aplikasi jaringan, seperti penyediaan akses untuk TCP, UDP, IP Address dan URL. Tetapi *Java Networking* tidak menyediakan akses untuk ICMP dikarenakan alasan *security* dan pada kondisi umum hanya *administrator* ( *root* ) yang bisa memanfaatkan protokol ICMP.

- c. *Java Database Connectivity* (JDBC)

JDBC menyediakan sekumpulan API yang dapat digunakan untuk mengakses database seperti *Oracle*, *MySQL*, *PostgreSQL*, *Microsoft SQL Server*.

d. *Java Security*

*Java Security* menyediakan sekumpulan API untuk mengatur security dari aplikasi Java baik secara *high level* atau *low level*, seperti *public/private key management* dan *certificates*.

e. *Java Swing*

*Java Swing* menyediakan sekumpulan API untuk membangun aplikasi – aplikasi GUI (*Graphical User Interface*) dan model GUI yang diinginkan bisa bermacam – macam, bisa model Java, model Motif/CDE atau model yang dependent terhadap platform yang digunakan.

f. *Java RMI*

*Java RMI* menyediakan sekumpulan API untuk membangun aplikasi – aplikasi Java yang mirip dengan model RPC (*Remote Procedure Call*) jadi *object - object* Java bisa di call secara *remote* pada jaringan komputer.

g. *Java 2D/3D*

*Java 2D/3D* menyediakan sekumpulan API untuk membangun grafik – grafik 2D/3D yang menarik dan juga akses ke printer.

h. *Java Server Pages*

Berkembang dari *Java Servlet* yang digunakan untuk menggantikan aplikasi – aplikasi CGI, JSP (*Java Server Pages*) yang mirip ASP dan PHP merupakan alternatif terbaik untuk solusi aplikasi *Internet*.

i. *JNI (Java Native Interface)*

*JNI* menyediakan sekumpulan API yang digunakan untuk mengakses fungsi – fungsi pada *library* (\*.dll atau \*.so) yang dibuat dengan bahasa pemrograman yang lain seperti *C*, *C++*, dan *Basic*.

j. *Java Sound*

*Java Sound* menyediakan sekumpulan API untuk memanipulasi *sound*.

k. *Java IDL + CORBA*

*Java IDL (Interface Definition Language)* menyediakan dukungan Java untuk implementasi *CORBA (Common Object Request Broker)* yang merupakan model *distributed-Object* untuk solusi aplikasi besar di dunia *networking*.

1. *Java Card*

*Java Card* utamanya digunakan untuk aplikasi – aplikasi pada *smart card*, yang sederhana wujudnya seperti *SIM Card* pada *handphone*.

m. *JTAPI (Java Telephony API)*

*Java Telephony API* menyediakan sekumpulan API untuk memanfaatkan *devices* – *devices telephony*, sehingga akan cocok untuk aplikasi – aplikasi CTI (*Computer Telephony Integration*) yang dibutuhkan seperti ACD (*Automatic Call Distribution*), PC-PBX dan lainnya.

### 2.8.5 *Java Development Kit (JDK)*

Menurut Budi dkk (2007) *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*. Baisanya, setiap *JDK* berisi satu atau lebih *JRE* dan berbagai alat pengembangan lain seperti sumber *compiler java*, *bundling*, *debuggers*, *development libraries* dan lain sebagainya.

## 2.9 *Android Studio*

Menurut Anisa dkk (2017) *Android Studio* adalah sebuah *Integrated Development Environment (IDE)* utama *Google* untuk mengembangkan pada *platform Android*. Karena *Android Studio* merupakan IDE dari *Google*, maka software ini dapat secara langsung terintegrasi dengan *Google Maps* menggunakan *API Key* yang dibuat di laman yang disediakan dari *Google Maps API* untuk mengintegrasikan peta dengan *software* sehingga peta akan secara otomatis ditampilkan di aplikasi yang dibuat. Selain terintegrasi dengan *Google Maps*, *Android Studio* juga dapat terintegrasi dengan *database SQLite Manager*, *plugin* untuk pengolahan dan penyimpanan informasi yang saling berkaitan untuk kemudian dibuat algoritma dari tiap data yang akan ditampilkan.

## 2.10 UML

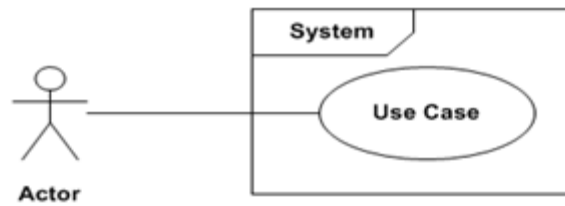
Menurut Whitten dkk (2004) UML merupakan bahasa yang digunakan sebagai standar dalam suatu industri untuk melakukan visualisasi, merancang, dan mendokumentasikan sebuah sistem piranti lunak. UML juga memberikan jasa untuk merancang model sistem yang dibuat untuk segala jenis aplikasi piranti lunak dan memungkinkan aplikasi tersebut dapat dijalankan pada piranti keras, sistem operasi, dan jaringan apapun. Karena UML menggunakan *class* dan *operation* pada konsep dasarnya, maka penulisan piranti lunak yang baik dilakukan pada bahasa yang berorientasi objek seperti C++, Java, C# atau VB.NET, namun UML juga dapat digunakan untuk modeling aplikasi prosedural dalam bahasa VB dan C.

Sama seperti bahasa-bahasa lain, UML juga digunakan untuk mendefinisikan notasi dan *syntax*. Notasi yang ada pada UML merupakan bentuk khusus yang digunakan untuk menggambarkan berbagai macam diagram piranti lunak. Bentuk-bentuk yang ada pada notasi UML memiliki makna tertentu, dan UML *syntax* yang bertugas untuk mendefinisikan bentuk-bentuk tersebut agar dapat dikombinasikan. 3 Notasi UML yang sudah ada sebelumnya diantaranya:

- a. Grady Booch OOD (*Objec-Oriented Design*).
- b. Jim Rumbaugh OMT (*Object Modelling Technique*).
- c. Ivar Jacobson OOSE (*Object-Oriented Software Engineering*).

### 2.10.1 Use Case Diagram

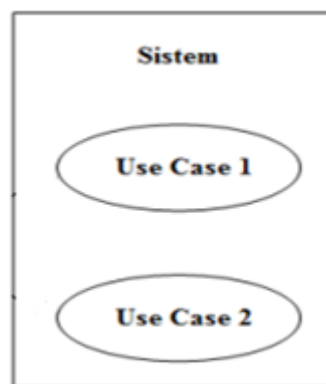
Menurut Whitten dkk (2004) *Use case diagram* ialah sebuah diagram yang digunakan untuk menggambarkan fungsionalitas yang ada pada sistem serta interaksi antara sistem, eksternal sistem, dan pengguna. Pada *use case diagram* ini akan dijelaskan mengenai apa yang diperbuat oleh sistem dan bukan bagaimana sebuah *use case* merepresentasikan interaksi antara aktor dan sistem. *Use case diagram* biasanya digunakan untuk membantu pada saat penyusunan *requirement* pada sebuah sistem, pada saat menjelaskan rancangan pada klien, dan pada saat merancang *test case* yang dilakukan untuk seluruh aplikasi pada sistem.



**Gambar 2. 3** *Use Case Diagram*

#### **2.10.1.1** *Use Case*

*Use case* yaitu uraian yang saling terkait sistematis antar sistem untuk membentuk pola tingkah laku yang digambarkan melalui garis dan diawasi oleh *actor*.

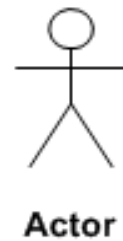


**Gambar 2. 4** Contoh *Use Case Diagram*

#### **2.10.1.2** *Actor*

*Actor* adalah user yang berinteraksi dengan sistem yang berfungsi untuk bertukar informasi. *Actor* dapat digambarkan dengan bentuk *stick man*. *Actor* juga dapat dibedakan menjadi 3 katagori utama yaitu *primary system actor*, *primary business actor* dan *external receiver actor*.





**Gambar 2. 5** *Actor Use Case Diagram*

### 2.10.1.3 Relationship

*Relationships* yaitu hubungan antar kelas yang terjalin antara *actor* dan sistem maupun sistem dengan *use case*. *Association* akan terjadi jika antara dua kelas mengetahui adanya kegiatan.

#### a. Association

*Association* merupakan suatu hubungan antar *use case* dan *actor* dalam sistem yang terstruktur. *Association* dapat dilambangkan garis lurus.



**Gambar 2. 6** *Association*

#### b. Aggregation

*Aggregation* merupakan hubungan antar *class* yang berkaitan lemah, dimana jika salah satu kelas hilang maka kelas yang bergantung masih dapat berjalan tanpa ada nya kelas tersebut.



**Gambar 2. 7** *Aggregation*

c. *Depedency* (Keberuntungan)

Hubungan yang mandiri antar *use case* maupun antar *actor*, hubungan antar *use case* akan berubah dan akan mempengaruhi *use case* lain jika ada perubahan dari *use case* yang mandiri.

### 2.10.2 *Class Diagram*

Menurut Whitten dkk (2004) *Class diagram* merupakan kumpulan gambar yang terstruktur yang didalamnya terdapat kelas objek yang ada pada suatu sistem dan hubungan antar kelas. *Class* dapat diartikan pendiskripsian kumpulan dari kelompok objek-objek.




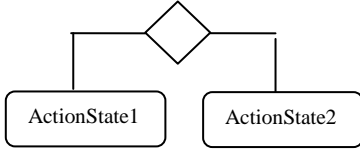
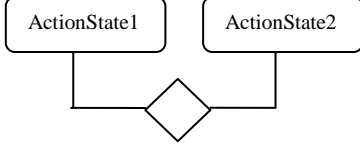
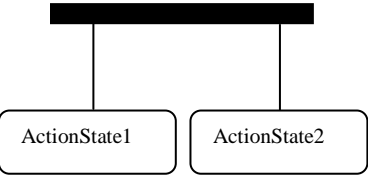
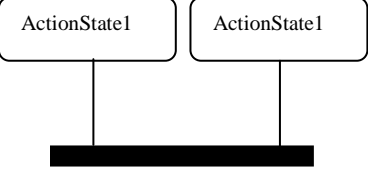
### 2.10.3 *Activity Diagram*


Menurut Whitten dkk (2004) *Activity diagram* ialah sebuah diagram yang digunakan untuk menggambarkan jalur aktivitas pada sistem yang sedang dirancang. *Activity diagram* juga digunakan untuk menggambarkan *decision* yang mungkin terjadi pada saat sistem dirancang serta digunakan untuk menggambarkan sebuah proses parallel yang mungkin terjadi pada saat eksekusi.

*Activity diagram* merupakan sebuah *state diagram* yang khusus karena sebagian *state* merupakan *action*. *Activitydiagram* tidak menggambarkan *behavior* internal dari sebuah sistem secara eksak, melainkan menggambarkan proses-proses dan jalur-jalur aktivitas secara umum.

Aktivitas dapat digambarkan dan direalisasikan dengan menggunakan satu *use case* atau lebih. Aktivitas juga menggambarkan proses yang sedang berjalan, sementara *use case* menggambarkan aktor yang sedang menggunakan sistem untuk melakukan aktivitas.

Tabel 2. 1 Simbol *Activity Diagram*

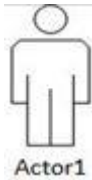
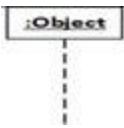



Gambar	Nama	Keterangan
	<i>Initial Node</i>	Simbol yang menandakan dimulainya suatu aktivitas atau proses
	<i>Actions</i>	Simbol yang menandakan suatu aktivitas pada proses
	<i>Flows</i>	Simbol yang menandakan jalur yang menghubungkan suatu aktivitas dengan aktivitas lain pada proses
	<i>Decision</i>	<i>Decision</i> adalah simbol yang menandakan jalur aktivitas yang dapat dipilih
	<i>Merge</i>	<i>Merge</i> adalah simbol yang menandakan penggabungan aktivitas yang dipisahkan oleh decision
	<i>Fork</i>	<i>Fork</i> adalah simbol yang menandakan dua aktivitas yang berjalan secara bersamaan
	<i>Join</i>	<i>Join</i> adalah simbol yang menandakan proses parallel yang sudah selesai

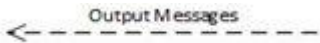


	<i>Activity Final</i>	Simbol yang menandakan berakhirnya suatu aktivitas atau proses
---	-----------------------	--

#### 2.10.4 *Sequence Diagram*

Menurut Whitten dkk (2004) *Sequence diagram* merupakan sebuah gambaran yang menjelaskan bagaimana objek berinteraksi satu sama lain melalui pesan dalam eksekusi dari sebuah *use case* atau sebuah operasi. Diagram ini menggambarkan bagaimana pesan dikirim dan diterima antar objek dan urutannya.

Tabel 2. 2 Simbol *Sequence Diagram*

Gambar	Nama	Keterangan
	<i>Actor</i>	Simbol yang menunjukkan orang yang berinteraksi dengan suatu aktivitas pada proses
	<i>Objects</i>	Simbol yang digunakan untuk menandakan objek-objek pada kelas
	<i>Life Lines</i>	Simbol berupa garis putus-putus yang digunakan untuk menandakan suatu objek yang berjalan pada proses
	<i>Activation Bars</i>	Simbol yang digunakan untuk menunjukkan lamanya suatu objek berjalan pada proses
	<i>Input Message</i>	Simbol yang digunakan untuk menyampaikan input dari suatu objek

	<i>Output Message</i>	Simbol yang digunakan untuk mengirim balik input yang dikirim dari suatu objek
	<i>Self Call</i>	Simbol yang digunakan untuk menandakan pengulangan input pada objek itu sendiri
	<i>Frame</i>	Simbol yang digunakan untuk menggambarkan suatu keadaan pada diagram. Frame terdiri dari 3, yaitu: pengulangan (loop), pemilihan (alternate), dan penentuan (optional)

## 2.11 Software Testing

Pengujian perangkat lunak dilakukan untuk menjamin kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean. Berikut ini adalah teknik pengujian yang digunakan :

### 2.11.1 White Box Testing

Menurut Pressman (2003) *White-Box Testing* adalah metode desain *test case* yang menggunakan struktur kontrol desain prosedural untuk memperoleh *test case*. Dengan menggunakan metode pengujian ini akan didapatkan *test case* yang :

- Memberikan jaminan bahwa semua jalur independen pada suatu modul telah digunakan paling tidak satu kali
- Menggunakan semua keputusan logis pada sisi *true* dan *false*
- Mengeksekusi semua *looping* pada batasan tertentu
- Dan menggunakan struktur data internal yang menjamin validitasnya

### **2.11.2 *Black Box Testing***

Menurut Pressman (2003) *Black-Box Testing* adalah metode pengujian yang berfokus pada persyaratan fungsional perangkat lunak. Pengujian ini berusaha menemukan kesalahan dalam kategori sebagai berikut :

- a. Fungsi – fungsi yang tidak benar atau hilang.
- b. Kesalahan *interface*.
- c. Kesalahan dalam struktur data atau akses database eksternal.
- d. Kesalahan kinerja.