

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Dalam penulisan penelitian ini, peneliti menggunakan referensi tinjauan pustaka yang berhubungan dengan kegiatan ini. Adapun beberapa referensi yang digunakan sebagai sumber informasi didalam penulisan penelitian ini didapat dari materi kuliah, jurnal, dan beberapa penulisan karya ilmiah yang ada kaitannya.

Adapun beberapa referensi atau jurnal pendukung yang digunakan sebagai acuan penulisan penelitian ini, antara lain:

- a. Penelitian yang dilakukan oleh Dwija Wisnu Brata pada tahun 2015 yang berjudul **“Perancangan Sistem KHS Mobile di STMIK ASIA Malang Menggunakan Android Programming dan JSON”** Vol. 9 No. 2, ISSN 0852-730X yang membahas tentang sistem khs *mobile* yang merupakan sarana mahasiswa dalam memperoleh informasi nilai setiap semester ketika menempuh pendidikan di kampus. Nilai senantiasa menjadi acuan seorang mahasiswa, karena suatu hal yang penting yang nantinya akan digunakan dikemudian hari terutama dalam dunia kerja. Banyak mahasiswa, baik di universitas negeri ataupun swasta senantiasa berusaha menjadi yang terbaik didalam kelas. Mengerjakan tugas, hadir didalam kelas, aktif didalam kelas serta dapat mengerjakan ujian akhir di setiap akhir semester merupakan kemampuan akses info nilai pada sistem informasi juga menjadi nilai tambah bagi sebuah kampus, artinya ada inovasi tentang penyampaian info nilai pada sistem informasi akademik dan disesuaikan dengan teknologi yang sedang berkembang. Hal ini menjadi perhatian dikarenakan banyak mahasiswa yang dari luar pulau atau sedang libur semester ketika ingin mengakses khs pada setiap semester, mahasiswa harus ke kampus untuk melihat pada *software* akademik dengan versi *offline*. Salah satu kendala tersebut bisa menjadi perhatian terutama STMIK ASIA sebagai institusi yang bergerak dalam hal menciptakan mahasiswa yang ahli dalam bidang teknik informatika untuk dapat menghasilkan produk-produk yang bermanfaat. Permasalahan akses info

nilai khs tersebut dapat teratasi dengan mengembangkan sistem informasi khs *mobile* dengan memanfaatkan JSON sebagai pengirim data, artinya aplikasi yang dirancang mengambil data dari *server* dengan melalui sebuah perantara *tool* yang mengirimkan data request melalui variabel melalui *browser*, dan *browser* akan memarsing data tersebut kepada *server*, dan *server* yang akan mengartikan, juga mengolahnya. Hal sangat bermanfaat apabila koneksi yang digunakan dalam kondisi lambat.

- b. Penelitian yang dilakukan oleh M. Rizki Alpiandi pada tahun 2016 yang berjudul **“Sistem Informasi Akademik Berbasis Web Di SMP Negeri 2 Kecamatan Gaung Anak Serka”** Vol. 5 No. 3, ISSN 2302-8149 yang membahas tentang perkembangan teknologi informasi yang semakin cepat dan pesat menjadi tantangan berat bagi sebuah perusahaan ataupun lembaga pendidikan, baik itu negeri maupun swasta. Setiap sektor pendidikan diharapkan dapat memanfaatkan teknologi informasi sebagai penunjang kegiatan operasional dalam menghasilkan suatu informasi. Sistem informasi akademik pada Sekolah Menengah Pertama Negeri 2 Kecamatan Gaung Anak Serka masih dilakukan secara manual, dengan sistem pendataan yang sekarang dirasakan masih banyak kekurangan yang terjadi, dikarenakan sistem yang ada masih menggunakan lembaran kertas dan arsip sehingga menyebabkan data-data yang ada sangat mungkin hilang ataupun rusak. Pengembangan sistem berupa pembuatan sistem informasi akademik berbasis web pada SMP Negeri 2 Kecamatan Gaung Anak Serka merupakan sistem yang memberikan informasi laporan siswa secara *online* yang berupa laporan nilai siswa yang bersangkutan, jadwal pelajaran, dan data pengajar, sehingga membantu kecepatan dan kualitas dalam penyampaian informasi. Sehingga dapat diharapkan dapat diimplementasikan dalam sebuah produk yang akan memperbaiki segala kekurangan yang ada pada sistem lama.
- c. Penelitian yang dilakukan oleh Muhammad Ibrohim, Arwan Hermawan pada tahun 2016 yang berjudul **“Pengembangan Sistem Informasi Akademik Universitas Serang Raya Berbasis Mobile Android”** Vol. 3, ISSN: 2406-7768 yang membahas tentang perkembangan teknologi dan realita yang ada, maka pengembangan sistem informasi berbasis *mobile*

Android sangat penting dilakukan untuk menunjang kegiatan mahasiswa mengenai kegiatan akademik. Pengembangan sistem ini nantinya akan terpasang pada perangkat *mobile* Android dalam bentuk aplikasi yang dihubungkan dengan database server website Universitas Serang Raya. Dengan aplikasi *mobile* ini mahasiswa dapat mengakses informasi akademik diantaranya biodata mahasiswa, biodata Dosen, mengisi Kartu Rencana Studi (KRS), informasi Kartu Hasil Studi (KHS), Buku-buku di Perpustakaan, dan Jurnal. Aplikasi ini juga memanfaatkan fitur notifikasi yang ada pada perangkat Android sehingga pengguna akan menerima informasi nilai matakuliah melalui notifikasi.

- d. Penelitian yang dilakukan oleh M. Iqbal Dzulhaq, Rahmat Tullah, dan Putra Satia Nugraha pada tahun 2017 yang berjudul **“Sistem Informasi Akademik Sekolah Berbasis Kurikulum 2013”** Vol. 7 No. 1, ISSN: 2088-1762 yang membahas tentang Sistem informasi akademik adalah suatu sistem informasi yang memberikan layanan informasi yang berupa data. Data-data yang berhubungan dengan akademik yaitu melayani registrasi siswa baru dan siswa lama, melakukan input data siswa dan input data guru kedalam database, membuat jadwal mengajar untuk guru, membuat jadwal pelajaran untuk siswa, menentukan pembagian kelas dan menerima nilai akhir dari guru. Sistem ini bertujuan untuk mendukung penyelenggaraan pendidikan, sehingga perguruan tinggi dapat menyediakan layanan informasi yang lebih baik dan efektif kepada komunitasnya, baik didalam maupun diluar sekolah tersebut melalui internet. Pada penelitian ini digunakan teknik analisis berupa pendekatan *Object Oriented Analysis (OOA)* atau analisis berorientasi obyek dengan UML. Proses analisis dilakukan terhadap hasil tahapan pengumpulan data dengan wawancara, observasi, dan studi pustaka untuk mendapatkan spesifikasi kebutuhan sistem yang akan dikembangkan. Metode perancangan sistem yang digunakan dalam penelitian ini yaitu metode *System Development Life Cycle (SDLC)*. Dengan adanya sistem ini maka pengolahan data dan nilai siswa dapat dengan cepat dilakukan.

- e. Penelitian yang dilakukan oleh Dwi Wijonarko, Betta Wahyu Retna Mulya pada tahun 2018 yang berjudul **“Pengembangan Antarmuka Pemrograman Aplikasi Menggunakan RESTful pada Sistem Informasi Akademik Politeknik Kota Malang”** Vol. 8 No. 2, ISSN: 2087-0256 yang membahas tentang kemudahan akses informasi antar perangkat dan media saat ini menjadi hal yang sangat lazim ditemukan. Satu fungsi penting dalam pengembangan aplikasi yaitu kemampuan untuk mengembangkan aplikasi menjadi lebih besar dan mampu diterapkan dalam berbagai macam *platform* serta perangkat yang ada saat ini. Pengembangan antarmuka pemrograman aplikasi adalah salah satu cara agar suatu aplikasi/sistem dapat diakses serta dimanfaatkan oleh pihak lain tanpa harus mengubah kode utama sistem dan basis data sistem serta memudahkan komunikasi antar sistem meskipun berbeda *platform*. Sistem Informasi Akademik Politeknik Kota Malang, menerapkan antarmuka pemrograman aplikasi menggunakan metode RESTful untuk berkomunikasi dengan sistem lain seperti aplikasi mobile, sistem kepegawaian dan keuangan. REST (*Representational State Transfer Protocol*) merupakan metode pengembangan web service yang menonjolkan kesederhaan dan menggunakan fitur yang ada pada HTTP untuk mendapatkan sumberdaya saat menggunakan *web service* tersebut.

2.2 Pengertian Perancangan

Perancangan adalah langkah pertama dalam fase pengembangan rekayasa produk atau sistem. Fase ini adalah inti teknis dari proses rekayasa perangkat lunak. Pada fase ini elemen-elemen dari model analisa dikonversikan. Dengan menggunakan satu dari sejumlah metode perancangan, fase perancangan akan menghasilkan perancangan data, perancangan antarmuka, perancangan arsitektur, dan perancangan prosedur (Manopo, Wowor, & Lumenta, 2016).

Perancangan adalah wujud *visual* yang dihasilkan dari bentuk-bentuk kreatif yang telah direncanakan. Langkah awal dalam perancangan desain bermula dari hal-hal yang tidak teratur berupa gagasan atau ide-ide kemudian melalui proses penggarapan dan pengelolaan akan menghasilkan hal-hal yang teratur, sehingga

hal-hal yang sudah teratur bisa memenuhi fungsi dan kegunaan secara baik (Hamzah, Soma, & Rahmawati, 2017).

Berdasarkan dari definisi yang telah di uraikan diatas, maka peneliti dapat menyimpulkan bahwa sistem adalah kumpulan dari beberapa komponen yang saling berinteraksi dalam sebuah prosedur untuk mencapai tujuan yang telah ditetapkan, dimana sistem dapat memproses masukan (*input*) dan menghasilkan pengeluaran (*output*) yang diinginkan.

2.3 Pengertian Sistem

Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu (Andalia & Setiawan, 2015).

Sistem adalah suatu kerangka kerja terpadu yang mempunyai satu sasaran atau lebih. Sistem ini mengkoordinasi sumber daya yang dibutuhkan untuk mengubah masukan-masukan menjadi keluaran (Brata, 2015).

Berdasarkan dari definisi yang telah di uraikan diatas, maka peneliti dapat menyimpulkan bahwa sistem adalah kumpulan dari beberapa komponen yang saling berinteraksi dalam sebuah prosedur untuk mencapai tujuan yang telah ditetapkan, dimana sistem dapat memproses masukan (*input*) dan menghasilkan pengeluaran (*output*) yang diinginkan.

2.4 Pengertian Informasi

Informasi adalah data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya. Dapat dikatakan bahwa data merupakan bahan mentah (Andalia & Setiawan, 2015).

Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau saat mendatang (Alpiandi, 2016).

Dari penjelasan diatas dapat disimpulkan bahwa informasi adalah data yang telah diolah menjadi suatu bentuk yang penting dalam pengambilan keputusan sekarang atau yang akan datang.

2.5 Pengertian Sistem Informasi

Sistem informasi adalah sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan (Brata, 2015).

Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan- laporan yang diperlukan (Alpiandi, 2016).

Jadi berdasarkan pengertian diatas dapat disimpulkan bahwa sistem informasi adalah kumpulan data yang terintegritasi dan saling melengkapi dengan menghasilkan *output* yang baik guna untuk memecahkan masalah dan pengambilan keputusan.

2.6 Pengertian Akademik

Akademik adalah suatu bidang yang mempelajari tentang kurikulum atau pembelajaran dalam fungsinya untuk meningkatkan pengetahuan dalam segi pendidikan / pembelajaran yang dapat dikelola oleh suatu sekolah atau lembaga pendidikan (Alpiandi, 2016).

Akademik adalah keadaan orang-orang bisa menyampaikan dan menerima gagasan, pemikiran, ilmu pengetahuan, dan sekaligus dapat mengujinya secara jujur, terbuka, dan leluasa (Eko Putra Membara, 2014).

2.7 Pengertian Sistem Informasi Akademik

Sistem informasi akademik adalah sistem informasi yang menangani masalah-masalah akademik guna memantau prestasi yang telah dicapai selama proses belajar mengajar. Prosedur kerja untuk kegiatan akademik yang dilakukan yaitu dari prosedur pendaftaran siswa yang akan mengikuti pembelajaran atau pelatihan sampai proses penilaian atas prestasi yang telah dicapai (Brata, 2015).

Sistem informasi akademik adalah perangkat lunak yang digunakan untuk menyajikan informasi dan menata administrasi yang berhubungan dengan kegiatan akademis (Sumarlin, 2015).

2.8 Pengertian *Mobile*

Aplikasi *Mobile* adalah sebuah aplikasi yang memungkinkan Anda melakukan mobilitas dengan menggunakan perlengkapan seperti PDA, telepon seluler atau *Handphone*. Dengan menggunakan aplikasi *mobile*, Anda dapat dengan mudah melakukan berbagai macam aktifitas mulai dari hiburan, berjualan, belajar, mengerjakan pekerjaan kantor, *browsing* dan lain sebagainya. Pemanfaatan aplikasi *mobile* untuk hiburan paling banyak digemari oleh hampir 70% pengguna telepon seluler, karena dengan memanfaatkan adanya fitur game, music player, sampai video player membuat kita menjadi semakin mudah menikmati hiburan kapan saja dan dimanapun (Kosidin & Farizah, 2016).

Perangkat *mobile* memiliki banyak jenis dalam hal ukuran, desain dan *layout*, tetapi mereka memiliki kesamaan karakteristik yang sangat berbeda dari sistem desktop, yaitu antara lain :

- a. Ukuran yang kecil perangkat *mobile* memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.
- b. Memori yang terbatas perangkat *mobile* juga memiliki *memory* yang kecil, yaitu *primary* (RAM) dan *secondary* (*disk*). Pembatasan ini adalah salah satu faktor yang mempengaruhi penulisan program untuk berbagai jenis dari perangkat ini. Dengan pembatasan jumlah dari *memory*, pertimbangan-pertimbangan khusus harus diambil untuk memelihara pemakaian dari sumber daya yang mahal ini.
- c. Daya proses yang terbatas sistem *mobile* tidaklah setangguh rekan mereka yaitu desktop. Ukuran, teknologi dan biaya adalah beberapa faktor yang mempengaruhi status dari sumber daya ini. Seperti *hard disk* dan RAM, mereka dapat ditemukan dalam ukuran yang pas dengan sebuah kemasan kecil.
- d. Mengkonsumsi daya yang rendah perangkat *mobile* menghabiskan sedikit daya dibandingkan dengan mesin desktop. Perangkat ini harus menghemat daya karena mereka berjalan pada keadaan dimana daya yang disediakan dibatasi oleh baterai-baterai.

- e. Kuat dan dapat diandalkan karena perangkat *mobile* selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapi benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.
- f. Konektivitas yang terbatas perangkat *mobile* memiliki *bandwidth* rendah, beberapa dari mereka bahkan tidak tersambung. Kebanyakan dari mereka menggunakan koneksi *wireless*.

Masa hidup yang pendek perangkat-perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala. Coba ambil kasus sebuah handphone, mereka *booting* dalam hitungan detik dan kebanyakan orang tidak mematikan handphone mereka bahkan ketika malam hari. PDA akan menyala jika anda menekan tombol power merek (Salamun, 2017).

2.9 Pengertian Android

Android adalah sebuah sistem operasi 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 atau *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*. Pada saat perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Di lain pihak, *Google* merilis kode-kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler (Hussein, Wahyuni, Irawan, & Mukhtar, 2018).

2.9.1 Sejarah Android

Perjalanan Android dimulai sejak Oktober 2003 ketika 4 orang pakar IT, Andi Rubin, Rich Miner, Nick Sears dan Chris White mendirikan *Android, Inc.*, di California US. Visi Android untuk mewujudkan *mobile device* yang lebih peka dan mengerti pemiliknya, kemudian menarik raksasa dunia maya *Google*. *Google* 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 bagi android untuk bersaing dan menyingkirkan *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* (Lengkong, Alicia, & Lumenta, 2015).

2.9.2 Fitur Android

(Lengkong, Alicia, & Lumenta, 2015), Fitur-fitur yang tersedia pada platform android adalah sebagaimana di uraikan berikut :

a. *Framework Aplikasi*

Fitur ini mendukung penggantian komponen dan penggunaan kembali komponen yang sudah dibuat (*reusable*). Seperti pada umumnya, *framework* memiliki keuntungan dalam proses pengkodean karena tidak perlu membuat kodongan untuk hal-hal yang pasti dilakukan seperti kodongan menampilkan gambar, kodongan konek *database*, dan lain-lain.

b. *Mesin Virtual Dalvik*

Lingkungan dimana aplikasi android akan bekerja.

c. *Integrated Browser*

Berdasarkan *Open Source engine WebKit*.

d. *Grafis*

Dengan adanya fitur ini, kita bisa membuat aplikasi grafis 2D dan 3D karena Android memiliki library *OpenGL ES 1,0*.

e. *SQLite*

Tugas dari fitur ini adalah berperan dalam penyimpanan data. Bahasanya mudah dimengerti dan merupakan sistem *database* nya android.

2.9.3 Perkembangan Android

a. Android versi 1.0

Sistem operasi ini bisa dibilang sebagai Android bayi yang masih sangat sederhana. Pengguna diajak beradaptasi dengan peradaban baru dalam menjajal sebuah perangkat telekomunikasi *mobile*. Jika hari ini kita terbiasa menggulir layar ke bawah untuk melihat notifikasi, perlu diketahui bahwa asal-muasalnya dari Android 1.0 ini. Sebab, di versi inilah *Google* untuk pertama kalinya memperkenalkan mekanisme pengecekan notifikasi yang kerap diistilahkan *pull-down notification* tersebut. Selain notifikasi, dua komponen pada Android 1.0 yang masih digunakan hingga kini adalah widget aplikasi, serta toko aplikasi *Google Play Store* yang kala itu masih bertajuk "Market". Sistem operasi ini juga menyatukan layanan Gmail.



Gambar 2. 1 Android Apple Pie

b. Android Versi 1.1

Android memang diluncurkan pertama kali pada tahun 2007, namun sistem operasi ini mulai dirilis dan diterapkan ke berbagai *gadget* pada tanggal 9 Maret 2009 silam. Android versi 1.0 merupakan Android awal yang dimana versi ini baru memberikan sentuhan di beberapa aplikasinya seperti sistem antar muka bagi pengguna (*user interface*) yang lebih baik, serta beberapa aplikasi yang lain.



Gambar 2. 2 Android *Banana Bread*

c. Android versi 1.5

Pada bulan Mei 2009 Android versi 1.5 atau yang dikenal sebagai *Android Cupcake*. Android generasi pertama yang diluncurkan secara komersil. Sistem operasi ini juga yang pertama kalinya diberikan nama kue atau camilan manis. Sistem operasi versi paling awal ini memiliki fitur yang terbilang sudah lazim saat ini seperti *widget*, *auto rotate*, dan *keyboard* virtual supaya memudahkan pengguna mengetik dan kustomasi *keyboard*. Bisa dibilang *Android Cupcake* hanya memiliki fitur Android dasar saja.



Gambar 2. 3 Android *Cupcake*

d. Android versi 1.6

Donut (versi 1.6) diluncurkan dalam tempo kurang dari 4 bulan semenjak peluncuran perdana *Android Cupcake*, yaitu pada bulan September 2009.

Di versi ini, *Google* sudah menyematkan fitur canggih seperti mesin penerjemah teks ke suara. Selain itu ada juga dukungan CDMA dan indikator penggunaan baterai. Secara umum, update yang ada di Android Donut hanya sekedar penambahan beberapa fitur dan perbaikan bug saja.



Gambar 2. 4 Android Donut

e. Android versi 2.0/2.1

Masih ditahun yang sama, Android kembali merilis operating sistem versi terbarunya, yaitu Android versi 2.0/2.1 *Eclair*. Android *Eclair* diluncurkan oleh *Google* 3 bulan setelah peluncuran. Android *Eclair* adalah versi Android yang dipersiapkan untuk menuju Hp pintar masa depan. Sistem operasi ini sudah dibekali dukungan fitur kamera seperti: *flash*, fokus, hingga efek warna. Selain itu, dari segi tampilan sistem operasi ini juga nampak segar dengan tambahan fitur *Live Wallpaper* yang bisa dibilang keren pada masanya.



Gambar 2. 5 Android Éclair

f. Android versi 2.2

Butuh 5 bulan bagi *Google* untuk melakukan regenerasi dari Android *Eclair* versi sebelumnya ke versi *Froyo Frozen Yoghurt*. Pada tanggal 20 Mei 2010, Android versi 2.2 alias Android *Froyo* ini dirilis. Versi Android ini punya performa dan optimalisasi yang cukup signifikan. Tersedia pula fitur yang cukup inovatif pada saat itu, seperti USB tethering, portabel WiFi *Hotspot*, dan *push notification*. Di versi ini, pengguna juga bisa memindahkan data-data aplikasi besar ke memori eksternal.



Gambar 2. 6 Android Froyo

g. Android versi 2.3

7 bulan kemudian Android kembali melakukan gebrakan dengan merilis kembali Android versi 2.3 atau yang dikenal sebagai Android *Gingerbread*. Versi ini cukup banyak menyedot perhatian publik dunia yang sebelumnya lebih banyak memiliki perangkat iOS, *Blackberry*, atau masih memakai *feature phone*. Pada *Gingerbread*, tampilan Android dibuat lebih simpel dan intuitif. Selain itu, juga ada fitur NFC untuk sistem pembayaran modern.



Gambar 2. 7 Android *Gingerbread*

h. Android versi 3.0/3.1

Pada bulan Mei 2011 Android versi 3.0/3.1 atau Android *Honeycomb* dirilis. Android *Honeycomb* merupakan sebuah sistem operasi Android yang tujuannya memang dikhususkan bagi penggunaan tablet berbasis Android. Tak banyak yang tahu mengenai kehadiran Android versi *Honeycomb*. Ini disebabkan karena sistem operasi ini hanya dikhususkan untuk perangkat tablet saja. Sistem operasi *Honeycomb* memiliki beberapa fitur unggulan di antaranya *System Bar*, dukungan *processor multi-core*, dan layar *home* atau *home screen* yang bisa dikustomisasi dengan bebas.



Gambar 2. 8 Android *Honeycomb*

i. Android versi 4.0

Android ICS atau *Ice Cream Sandwich* juga dirilis pada tahun yang sama dengan *Honeycomb*, yaitu pada bulan Oktober 2011. *Google* merasa merilis sistem operasi Android yang terpisah antara Hp dan tablet cukup merepotkan. Oleh karena itu, melalui *Ice Cream Sandwich* *Google* kembali menyatukan sistem operasi Android di Hp dan tablet menjadi satu. Karena digabung jadi satu kembali, untuk mengoptimasi sistem operasi akan jadi lebih mudah. Android *Ice Cream Sandwich* memiliki fitur unggulan Android Beam yang memungkinkan pengguna bisa transfer data dengan cepat melalui NFC.



Gambar 2. 9 Android Ice Cream Sandwich (ICS)

j. Android versi 4.1

Android *Jelly Bean* merupakan versi Android yang terbaru pada saat ini. Salah satu *gadget* yang menggunakan sistem operasi *Jelly Bean* adalah *Google Nexus 7* yang diprakarsai oleh ASUS, vendor asal Taiwan yang juga menjadi teman satu kampung halaman dengan Acer. Tak banyak fitur baru yang ditawarkan pada Android *Jelly Bean* dibanding pendahulunya (ICS). *Google* hanya fokus untuk meningkatkan performa dan keamanan sistem operasi Android pada versi ini. Jikapun ada fitur baru yang ada di *Jelly Bean* adalah dukungan konten gambar 4K UHD dan dukungan emoji.



Gambar 2. 10 Android *Jelly Bean*

k. Android versi 4.4

Kehadiran android *kitkat* merupakan peluncuran produk OS anyar yang diluncurkan pada 4 september 2013, sebelumnya banyak kabar beredar jikalau android akan meluncurkan OS baru yang bernama Android *Key Lime Pie* namun setelah di analisa tidak sesuai dengan ejaan orang umum, sehingga namanya diganti dengan OS Android *Kitkat* yang sebagian besar orang sudah familiar dengan itu.



Gambar 2. 11 Android *Kitkat*

l. Android versi 5.0.2

Android *Lollipop* merupakan keberadaan OS Android yang memang saat ini sudah menjadi trend baru di industri *smartphone*, hal ini tak lepas dari keunikan dan kelebihan yang banyak di miliki dari OS tersebut. Kehadiran android versi ini amat di nanti oleh sekian banyak orang karna

diharapkan sistem operasi *Lollipop* ini bias lebih baik dibandingkan versi-versi sebelumnya.



Gambar 2. 12 Android Lollipop

m. Android versi 6.0

Android 6.0 *Marshmallow* adalah versi dari sistem operasi *mobile* Android. Pertama kali diperkenalkan Mei 2015 di *Google I / O* di bawah kode nama Android M, secara resmi dirilis pada Oktober 2015. Android *Marshmallow* memperkenalkan model izin aplikasi didesain ulang sekarang ada hanya delapan kategori izin, dan aplikasi yang tidak lagi secara otomatis diberikan semua hak akses mereka ditentukan pada waktu instalasi.



Gambar 2. 13 Android Marshmallow

n. Android Versi 7.0

Android *Nougat* adalah sistem operasi yang dikembangkan khusus untuk perangkat android, Android *Nougat* merupakan generasi penerus dari versi sebelumnya yaitu *Marshmallow* yang di umumkan pada tanggal 1 Juli 2016 dan pertama kali di luncurkan untuk penggunaan publik pada 23 Agustus 2016. Ketika akan diluncurkan pihak *Google* mengundang pengembang aplikasi android untuk melihat seperti apa Android N, sehingga pihak pengembang nantinya bisa mengetahui aplikasi seperti apa yang harus dibangun untuk bisa di pasang pada Android N. Jadi ketika dirilis namanya yaitu Android *Nougat*, pengembang sudah bisa meluncurkan aplikasi yang bisa di pasang di dalamnya.



Gambar 2. 14 Android Nougat

o. Android Versi 8.0

Pada tanggal 21 Maret 2017 adalah Android *Oreo*. Untuk itu pada kesempatan kali ini Utopicomputers akan memberikan review seputar fitur, kelebihan dan kecanggihannya Android *Oreo* yang juga dikenal dengan Android 8.0. Ok dari pada kita berlama - lama mari langsung pada topik ulasanya. Android *Oreo* merupakan versi dari sistem operasi Android *mobile* yang menggunakan API level 8.0, Banyak sekali perubahan yang dilakukan oleh pengembang utamanya dari sisi keamanan dan performa yang ditawarkan. Tidak hanya itu banyak sekali kelebihan Android *Oreo* dibandingkan dengan versi sebelumnya. Dan dibawah ini merupakan fitur Android *Oreo* yang dapat ditemukan.



Gambar 2. 15 Android Oreo

p. Android Versi 9.0

Versi Android terbaru dengan nama kue *Pie* membawa lompatan baru dalam sejarah sistem operasi Hp. Android *Pie* sudah didukung kemampuan kecerdasan buatan (AI). Berkat AI, Hp Android jadi semakin pintar karena bisa mempelajari pola penggunaan secara otomatis. Android *Pie* bisa menyesuaikan konsumsi baterai dengan rutinitasmu sehari-hari. Aplikasi yang ditampilkan di halaman utama pun bisa disesuaikan dengan aplikasi apa yang paling sering digunakan dan bermanfaat saat menjalankan aktivitas sehari-hari. Android *Pie* juga bisa menganalisa waktu pemakaian Hp dan mengingatkanmu jika terlalu lama menggunakan Hp.



Gambar 2. 16 Android Pie

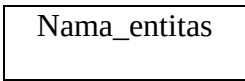
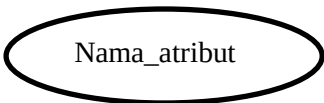
2.10 Pengertian Database

Database atau Basis data adalah kumpulan data yang saling berhubungan secara logikal serta deskripsi dari data tersebut, yang dirancang untuk memenuhi kebutuhan informasi suatu organisasi. Membangun basis data adalah langkah awal dari pembuatan sebuah aplikasi. Keberhasilan dalam membangun basis data akan menyebabkan program lebih mudah dibaca, mudah dikembangkan dan mudah mengikuti perkembangan perangkat lunak. Berikut ini adalah komponen-komponen yang terdapat dalam basis data, yaitu : *Tabel, Field, Record, Primary Key, Foreign Key dan Index* (Puspitasari, 2016).

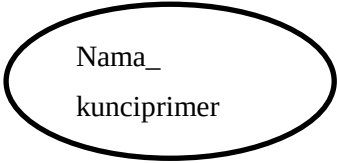
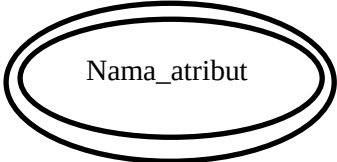
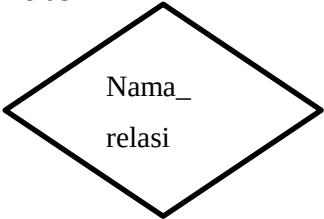
2.10.1 Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) adalah suatu rancangan atau bentuk hubungan suatu kegiatan di dalam sistem yang berkaitan langsung dan mempunyai fungsi di dalam proses tersebut. ERD adalah suatu pemodelan dari basis data relasional yang didasarkan atas persepsi di dalam dunia nyata, dunia ini senantiasa terdiri dari sekumpulan objek yang saling berhubungan antara satu dengan yang lainnya. Suatu objek disebut entity dan hubungan yang dimilikinya disebut relationship. Suatu entity bersifat unik dan memiliki atribut sebagai pembeda dengan entity lainnya (Puspitasari, 2016).

Tabel 2. 1 Simbol Entity Relationship Diagram

No	Simbol	Keterangan
1.	Entitas/ <i>entity</i> 	Entitas merupakan data inti yang akan disimpan tabel pada basis data, benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi computer, penamaan entitas biasanya lebih ke kata benda dan bukan merupakan nama tabel.
2.	Atribut 	<i>Field</i> atau kolom data yang butuh disimpan dalam suatu entitas.

Lanjutan Tabel 2.1 Simbol Entity Relationship Diagram

No	Simbol	Keterangan
3.	Atribut kunci primer 	<i>Field</i> atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses <i>record</i> yang diinginkan, biasanya berupa id, kunci primer dapat lebih dari satu kolom, asalkan kombinasi dari beberapa kolom tersebut dapat bersifat unik (berbeda tanpa ada yang sama).
4.	Atribut multivalor / <i>multivalor</i> 	<i>Field</i> atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu.
5.	Relasi 	Relasi yang menghubungkan antar entitas biasanya diawali dengan kata kerja.
6.	Asosiasi / association 	Penghubung antara relasi dan entitas di mana di kedua ujungnya memiliki multiplicity kemungkinan jumlah pemakaian. Kemungkinan jumlah maksimum keterhubungan antara entitas satu dengan entitas yang lain disebut dengan kardinalitas. Misalkan ada kardinalitas 1 ke N atau sering disebut dengan one to many menghubungkan entitas A dan entitas B.

2.10.2 Logical Relational Structure (LRS)

Logical Relational Structure (LRS) adalah Sebuah model sistem yang digambarkan dengan sebuah diagram-ER akan mengikuti pola/ aturan pemodelan tertentu dalam kaitannya dengan konversi ke LRS, maka perubahan yang terjadi

adalah mengikuti aturan – aturan berikut ini : Setiap entitas akan diubah kebentuk kotak, Sebuah atribut relasi disatukan dalam sebuah kotak bersama entitas jika hubungan yang terjadi pada diagram-ER 1:M (relasi bersatu dengan *cardinality* M) atau tingkat hubungan 1:1 (relasi bersatu dengan *cardinality* yang paling membutuhkan referensi), sebuah relasi dipisah dalam sebuah kotak tersendiri (menjadi entitas baru) jika tingkat hubungannya M:M (*many to many*) dan memiliki *foreign key* sebagai *primary key* yang diambil dari kedua entitas yang sebelumnya saling berhubungan (Puspitasari, 2016).

2.11 *Unified Modeling Language (UML)*

Menurut (Sulianta, 2017) “*Unifed Modeling language (UML)* merupakan kumpulan diagram-diagram yang sudah memiliki standar untuk membangun perangkat lunak berbasis objek”.

Menurut (Rosa & M.Shalahuddin, 2014) “Pada UML terdiri dari 13 macam diagram yang dikelompokkan menjadi 3 kategori”. Berikut ini penjelasan singkat dari pembagian kategori tersebut :

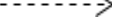
- a. *Structure diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. *Structure diagram* terdiri dari *class diagram*, *object diagram*, *component diagram*, *composite structure diagram*, *package diagram* dan *deployment diagram*.
- b. *Behavior diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. *Behavior Diagram* terdiri dari *Use Case Diagram*, *Activity Diagram*, *State Machine System*.
- c. *Interaction Diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem. *Interaction Diagram* terdiri dari *Sequence*”.

Berikut ini dijelaskan jenis-jenis UML (*Unified Model Language*) oleh Tri Hartati dalam jurnal Teknik dan Ilmu Komputer (2017:185), CSF (*Critical success Factor*) memiliki tipe dan sumber sebagai berikut :

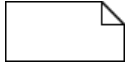
2.11.1 Use Case

Use Case Diagram atau diagram *use case* merupakan gambar dari beberapa atau seluruh aktor dengan tujuan mengenali interaksi mereka dalam suatu sistem.

Tabel 2. 2 Simbol Use Case

No	Gambar	Nama	Keterangan
1		Actor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case.
2		Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
3		Generalization	Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).
4		Include	Menspesifikasikan bahwa use case sumber secara eksplisit.
5		Extend	Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan.
6		Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.

Lanjutan Tabel 2.2 Simbol *Use Case*

No	Gambar	Nama	Keterangan
7		System	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		Use Case	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
9		Collaboration	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		Note	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

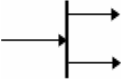
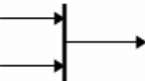
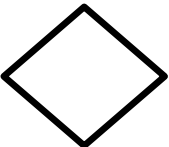
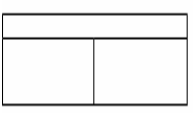
2.11.2 Activity Diagram

Diagram aktivitas atau *Activity Diagram* menggambarkan *workflow* (alur 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.

Tabel 2. 3 Simbol *Activity Diagram*

No	Gambar	Nama	Keterangan
1		Start Point	Menggambarkan titik awal siklus hidup suatu elemen

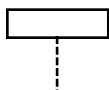
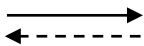
Lanjutan Tabel 2.3 Simbol *Activity Diagram*

No	Gambar	Nama	Keterangan
2		End Point	Menggambarkan titik akhir yang menjadi kondisi akhir suatu elemen
3		Activity	Memperlihatkan bagaimana masing-masing kelas anta saling berinteraksi
4		Fork	Menunjukkan adanya percabangan secara paralel dari aktivitas.
5		Joint	Menunjukkan adanya penggabungan aktivitas.
6		Decision	Decision adalah suatu titik / point pada activity diagram yang mengindikasikan suatu kondisi dimana ada kemungkinan perbedaan transisi
7		State Transition	State transition menunjukkan kegiatan apa berikutnya setelah suatu kegiatan Sebelumnya
8		Swimlane	Object swimlane untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

2.11.3 Sequence Diagram

Sequence Diagram atau 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 menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diintansiasi menjadi objek itu.

Tabel 2. 4 Simbol *Sequence Diagram*

No	Gambar	Nama	Keterangan
1		<i>Actor</i>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuatitu sendiri, jadi walaupun symbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2		<i>Object</i>	Menggambarkan pos-pos objek yang mengirim dan menerima pesan.
3		<i>Message</i>	Menggambarkan aliran pesan yang dikirim oleh pos-pos objek.
4		<i>Boundary</i>	Menggambarkan halaman <i>interface</i> yang dilalui oleh user dalam aksi yang dilakukan.
5		<i>Control</i>	Menggambarkan sebuah <i>class</i> dimana <i>class</i> tersebut terdapat beberapa kontrol seperti <i>sql control</i> , <i>event control</i> , dan lain-lain.
6		<i>Entity</i>	Menggambarkan sebuah <i>class</i> .
7		<i>Tabel</i>	Menggambarkan tabel yang ada pada sistem atau <i>database</i> .
8		Garis hidup/ <i>Lifeline</i>	Menyatakan kehidupan suatu objek

2.11.4 Class Diagram

Diagram kelas atau *Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Atribut mendeskripsikan properti dengan sebaris teks didalam kotak kelas tersebut. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Diagram kelas mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan dengan statis yang terdapat diantara mereka. Diagram kelas juga menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. Diagram kelas menggambarkan struktur dan deskripsi kelas, package dan objek beserta hubungan satu sama lain seperti containment, pewaris, asosiasi dan lain-lain.

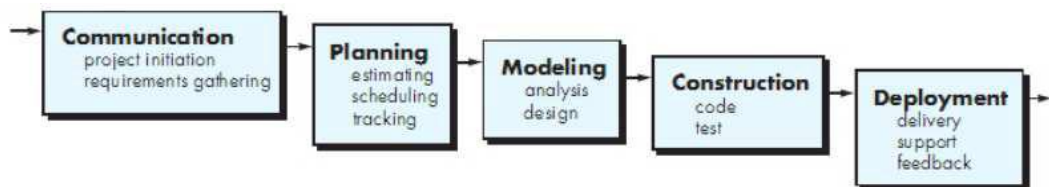
Tabel 2. 5 Simbol Class Diagram

No	Gambar	Nama	Keterangan
1		Generalization	Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).
2		Nary Association	Upaya untuk menghindari sosiasi dengan lebih dari 2 objek.
3		Class	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		Collaboration	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
5		Realization	Operasi yang benar-benar dilakukan oleh suatu objek.

2.12 Model Pengembangan Sistem

Menurut Pressman (2015:42), model *waterfall* adalah model klasik yang bersifat sistematis, berurutan dalam membangun *software*. Nama model ini sebenarnya adalah “*Linear Sequential Model*”. Model ini sering disebut juga dengan “*classic life cycle*” atau metode waterfall. Model ini termasuk kedalam model *generic* pada rekayasa perangkat lunak dan pertama kali diperkenalkan oleh Winston Royce sekitar tahun 1970 sehingga sering dianggap kuno, tetapi merupakan model yang paling banyak dipakai dalam *Software Engineering* (SE). Model ini melakukan pendekatan secara sistematis dan berurutan. Disebut dengan *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan.

Fase-fase dalam *Waterfall Model* menurut referensi Pressman :



Gambar 2. 17 Waterfall Pressman (Pressman, 2015:42)

a. Communication (Project Initiation & Requirements Gathering)

Sebelum memulai pekerjaan yang bersifat teknis, sangat diperlukan adanya komunikasi dengan *customer* demi memahami dan mencapai tujuan yang ingin dicapai. Hasil dari komunikasi tersebut adalah inisialisasi proyek, seperti menganalisis permasalahan yang dihadapi dan mengumpulkan data-data yang diperlukan, serta membantu mendefinisikan fitur dan fungsi *software*. Pengumpulan data-data tambahan bisa juga diambil dari jurnal, artikel, dan internet.

b. Planning (Estimating, Scheduling, Tracking)

Tahap berikutnya adalah tahapan perencanaan yang menjelaskan tentang estimasi tugas-tugas teknis yang akan dilakukan, resiko - resiko yang dapat terjadi, sumber daya yang diperlukan dalam membuat sistem, produk kerja yang ingin dihasilkan, penjadwalan kerja yang akan dilaksanakan, dan *tracking* proses pengerjaan sistem.

c. Modeling (*Analysis & Design*)

Tahapan ini adalah tahap perancangan dan permodelan arsitektur sistem yang berfokus pada perancangan struktur data, arsitektur *software*, tampilan *interface*, dan algoritma program. Tujuannya untuk lebih memahami gambaran besar dari apa yang akan dikerjakan.

d. Construction (*Code & Test*)

Tahapan *Construction* ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk/bahasa yang dapat dibaca oleh mesin. Setelah pengkodean selesai, dilakukan pengujian terhadap sistem dan juga kode yang sudah dibuat. Tujuannya untuk menemukan kesalahan yang mungkin terjadi untuk nantinya diperbaiki.

e. Deployment (*Delivery, Support, Feedback*)

Tahapan *Deployment* merupakan tahapan implementasi *software* ke *customer*, pemeliharaan *software* secara berkala, perbaikan *software*, evaluasi *software*, dan pengembangan *software* berdasarkan umpan balik yang diberikan agar sistem dapat tetap berjalan dan berkembang sesuai dengan fungsinya.

Kapan sebaiknya metode *waterfall* digunakan? Ada teori-teori yang menyimpulkan beberapa hal, yaitu :

- a. Ketika semua persyaratan yang diajukan sudah dipahami dengan baik pada awal pengembangan program.
- b. Definisi produk bersifat stabil dan tidak ada perubahan yang dilakukan saat pengembangan untuk alasan apapun. Oleh karena itu, teknologi yang digunakan juga harus sudah dipahami dengan baik.
- c. Menghasilkan produk baru, atau produk dengan versi baru.
- d. Sebenarnya, jika menghasilkan produk dengan versi baru maka itu sudah termasuk *incremental development*, yang setiap tahapannya sama dengan metode *waterfall* kemudian diulang-ulang.
- e. *Port-ing* produk yang sudah ada ke dalam *platform* baru.

Dengan demikian, metode *waterfall* dianggap pendekatan yang lebih cocok digunakan untuk proyek pembuatan sistem baru dan juga pengembangan *software* dengan tingkat resiko yang kecil serta waktu pengembangan yang cukup lama.

Tetapi salah satu kelemahan paling mendasar adalah menyamakan pengembangan *hardware* dan *software* dengan meniadakan perubahan saat pengembangan. Padahal, *error* diketahui saat *software* dijalankan, dan perubahan-perubahan akan sering terjadi.

Keuntungan menggunakan metode *waterfall* adalah prosesnya lebih terstruktur, hal ini membuat kualitas *software* baik dan tetap terjaga. Dari sisi *user* juga lebih menguntungkan, karena dapat merencanakan dan menyiapkan kebutuhan data dan proses yang diperlukan sejak awal. Penjadwalan juga menjadi lebih menentu, karena jadwal setiap proses dapat ditentukan secara pasti. Sehingga dapat dilihat jelas target penyelesaian pengembangan program. Dengan adanya urutan yang pasti, dapat dilihat pula perkembangan untuk setiap tahap secara pasti. Dari sisi lain, model ini merupakan jenis model yang bersifat dokumen lengkap sehingga proses pemeliharaan dapat dilakukan dengan mudah.

Kelemahan menggunakan metode *waterfall* adalah bersifat kaku, sehingga sulit melakukan perubahan di tengah proses. Jika terdapat kekurangan proses/prosedur dari tahap sebelumnya, maka tahapan pengembangan harus dilakukan mulai dari awal lagi. Hal ini akan memakan waktu yang lebih lama. Karena jika proses sebelumnya belum selesai sampai akhir, maka proses selanjutnya juga tidak dapat berjalan. Oleh karena itu, jika terdapat kekurangan dalam permintaan *user* maka proses pengembangan harus dimulai kembali dari awal. Karena itu, dapat dikatakan proses pengembangan *software* dengan metode *waterfall* bersifat lambat. Kelemahan lainnya menggunakan metode *waterfall* adalah membutuhkan daftar kebutuhan yang lengkap sejak awal. Tetapi, biasanya jarang sekali *customer* yang dapat memenuhi itu. Untuk menghindari pengulangan tahap dari awal, *user* harus memberikan seluruh prosedur, data, dan laporan yang diinginkan mulai dari tahap awal pengembangan. Tetapi pada banyak kondisi, *user* sering melakukan permintaan di tahap pertengahan pengembangan sistem. Dengan metode ini, maka *development* harus dilakukan mulai lagi dari tahap awal. Karena *development* disesuaikan dengan desain hasil *user* pada saat tahap pengembangan awal. Di sisi lain, *user* tidak dapat mencoba sistem sebelum sistem benar-benar selesai. Selain itu, kinerja personil menjadi kurang optimal karena terdapat proses menunggu suatu tahap selesai terlebih dahulu. Oleh karena itu,

seringkali diperlukan personil yang “*multi-skilled*” sehingga minimal dapat membantu pengerjaan untuk tahapan berikutnya (Pressman, 2015).

2.13 Aplikasi Pendukung

Aplikasi pendukung merupakan perangkat lunak/software yang membantu suatu aplikasi yang memiliki kekurangan untuk beroperasi sendiri. Mungkin kita sudah banyak mengetahui aplikasi pendukung itu bermacam-macam namun yang digunakan adalah sebagai berikut :

2.13.1 MIT App Inventor

Menurut (Amerkashi, 2015) “*App Inventor Android is an open-source web application originally provided by Google, and now maintained by the Massachusetts Institute of Technology (MIT). (App Inventor Android adalah sebuah aplikasi web open-source asli yang disediakan oleh Google, dan sekarang dikelola oleh Massachusetts Institute of Technology (MIT)).*”

Menurut (Kamriani & Roy, 2016) “*MIT App Inventor is a free, drag-and-drop, blocks-based visual programming language that enables people, regardless of their coding experience, to create mobile apps for Android devices. (MIT App Inventor adalah gratis, tahan-dan-lepas, suatu bahasa pemrograman visual berbasis blok yang dikenali orang, terlepas dari pengalaman koding mereka, untuk membuat aplikasi mobile untuk perangkat Android).*”

Berdasarkan pengertian-pengertian yang telah didefinisikan oleh para ahli tersebut maka dapat disimpulkan bahwa MIT App Inventor adalah aplikasi berbasis web open-source (sumber terbuka) yang disediakan oleh Google untuk membuat aplikasi perangkat Android.

2.13.2 MySQL

Menurut (MADCOM, 2016) “MySQL adalah sistem manajemen Database SQL yang bersifat Open Source dan paling populer saat ini. Sistem Database MySQL mendukung beberapa fitur seperti *multithreaded*, *multi-user* dan *SQL Database managemen system (DBMS)*”.

2.14 Pengujian

Pengujian *software* sangat diperlukan untuk memastikan *software* atau aplikasi yang sudah atau sedang dibuat dapat berjalan sesuai dengan fungsionalitas yang diharapkan. Pengembang atau penguji *software* harus menyiapkan sesi khusus untuk menguji program yang sudah dibuat agar kesalahan ataupun kekurangan dapat dideteksi sejak awal dan dikoreksi secepatnya. Pengujian atau *testing* sendiri merupakan elemen kritis dari jaminan kualitas perangkat lunak dan merupakan bagian yang tidak terpisahkan dari siklus hidup pengembangan *software* seperti halnya analisis, desain, dan pengkodean (Mustaqbal, Firdaus, & Rahmadi, 2015). Ada beberapa jenis pengujian perangkat lunak, antara lain :

2.14.1 White Box

White Box Testing adalah salah satu cara untuk menguji suatu aplikasi atau *software* dengan cara melihat modul untuk dapat meneliti dan menganalisa kode dari program yang di buat ada yang salah atau tidak. Kalau modul yang telah dan sudah di hasilkan berupa output yang tidak sesuai dengan yang di harapkan maka akan dikompilasi ulang dan di cek kembali kode-kode tersebut hingga sesuai dengan yang diharapkan (Mustaqbal, Firdaus, & Rahmadi, 2015).

- a. Penggunaan metode pengujian *white box* dilakukan untuk :
 1. Memberikan jaminan bahwa semua jalur independen suatu modul digunakan minimal satu kali.
 2. Menggunakan semua keputusan logis untuk semua kondisi *true* atau *false*.
 3. Mengeksekusi semua perulangan pada batasan nilai dan operasional pada setiap kondisi.
 4. Menggunakan struktur data internal untuk menjamin validitas jalur keputusan.
- b. Persyaratan dalam menjalankan strategi *white box testing*, yaitu :
 1. Mendefinisikan semua alur logika.
 2. Membangun kasus untuk digunakan dalam pengujian.
 3. Mengevaluasi semua hasil pengujian.
 4. Melakukan pengujian secara menyeluruh.

2.14.2 *Black Box*

Black Box Testing berfokus pada spesifikasi fungsional dari perangkat lunak. *Tester* dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. *Black Box Testing* bukanlah solusi alternatif dari *White Box Testing* tapi lebih merupakan pelengkap untuk menguji hal-hal yang tidak dicakup oleh *White Box Testing* (Mustaqbal, Firdaus, & Rahmadi, 2015).

Metode uji dapat diterapkan pada semua tingkat pengujian perangkat lunak seperti *unit*, *integrasi*, *fungsional*, sistem dan penerimaan. Ini biasanya terdiri dari kebanyakan jika tidak semua pengujian pada tingkat yang lebih tinggi, tetapi juga bias mendominasi unit testing juga.

Pengujian pada *black box* berusaha menemukan kesalahan seperti :

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