BAB II
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Sistem

Istilah sistem bukanlah hal yang asing bagi kebanyakan orang. Seringkali, sistem mengacu pada komputer seperti IBM PC, atau Macintosh, tetapi juga bisa ke arah yang lebih luas seperti sistem tata surya atau bahkan ke hal – hal yang lebih spesifik seperti sistem respirasi mamalia

Sistem merupakan sekumpulan prosedur yang saling berkaitan dan saling terhubung untuk melakukan suatu tugas bersama – sama. Secara garis besar, sebuah sistem informasi terdiri atas tiga komponen utama. Ketiga komponen tersebut mencakup *software*, *hardware*, dan *brainware* yang saling berkaitan satu sama lain [2].

Software mencakup semua perangkat lunak yang dibangun dengan bahasa pemrograman tertentu, pustaka, untuk kemudian menjadi sistem operasi, aplikasi, dan *driver*. Sistem operasi, aplikasi, *driver*, saling bekerja sama agar komputer dapat berjalan dengan baik. *Hardware* mencakup semua perangkat keras (*motherboard, processor, VGA,* dan lainnya) yang disatukan menjadi sebuah komputer. Dalam konteks yang luas, bukan hanya sebuah komputer, namun sebuah jaringan komputer, *brainware* mencakup kemampuan otak manusia, yang mencakup ide, pemikiran analisis, di dalam menciptakan dan menggabungkan *hardware* dan *software* dengan bantuan *brainware* inilah (melalui sejumlah prosedur) yang dapat menciptakan sebuah sistem yang bermanfaat bagi pengguna [2].

Sistem merupakan suatu bentuk integrasi antara suatu komponen dengan komponen lain karena sistem memiliki sasaran yang berbeda untuk setiap kasus yang terjadi dalam sistem tersebut [3].

Sistem adalah kumpulan atau himpunan dari unsur atau variabel – variabel yang saling terkait, saling berinteraksi, dan saling tergantung satu sama lain untuk mencapai tujuan [2].

Karakteristik suatu sistem [4]:

1. Komponen atau elemen (*Components*)

Suatu sistem terdiri dari komponen – komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan.

2. Batas sistem (*Boundary*)

   Batas sistem merupakan daerah yang membatasi antara sistem yang satu dengan sistem yang lainnya atau dengan lingkungan luarnya. Adanya batas sistem, maka sistem dapat membentuk suatu kesatuan, karena dengan batas sistem ini, fungsi dan tugas dari subsistem satu dengan yang lainnya berbeda tetapi saling berinteraksi. Dengan kata lain, batas sistem merupakan ruang lingkup atau *scope* dari sistem atau subsistem itu sendiri.

3. Lingkungan Luar Sistem (*Environment*)

   Lingkungan luar sistem adalah segala sesuatu diluar batas sistem yang mempengaruhi operasi suatu sistem. Lingkungan luar sistem dapat bersifat menguntungkan atau merugikan. Lingkungan luar sistem yang bersifat menguntungkan harus dipelihara dan dijaga supaya tidak hilang pengaruhnya. Sedangkan, lingkungan yang bersifat merugikan harus dihilangkan supaya tidak mengganggu operasi dari sistem.

4. Penghubung Sistem (*Interface*)

   Penghubung sistem merupakan suatu media (penghubung) antara satu subsistem dengan subsistem lainnya yang membentuk satu kesatuan, sehingga sumber – sumber daya mengalir dari subsistem yang satu ke subsistem lainnya. Dengan kata lain, melalui penghubung, *output* dari subsistem akan menjadi *input* bagi subsistem lainnya.

5. Masukan (*Input*)

   Input adalah energi atau suatu yang dimasukkan kedalam suatu sistem yang dapat berupa masukkan yaitu energi yang dimasukkan supaya sistem dapat beroperasi atau masukkan sinyal yang merupakan energi yang diproses untuk menghasilkan suatu luaran.

6. Luaran (*Output*)

   Merupakan hasil dari energi yang diolah dan diklasifikasikan menjadi luaran yang berguna, juga merupakan luaran atau tujuan akhir dari sistem.

7. Pengolah (*Process*)

   Suatu sistem mempunyai bagian pengolah yang akan mengubah *input* menjadi *output*.

8. Sasaran (*Objective*)

   Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan luaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.

2.1.2 Informasi

Informasi merupakan hasil penghasilan data dari satu atau berbagai sumber, yang kemudian diolah, sehingga memberikan nilai, arti, dan manfaat. Proses pengelolaan ini memerlukan teknologi. Berbicara mengenai teknologi memang tidak harus selalu berkaitan dengan komputer, namun komputer sendiri merupakan salah satu bentuk teknologi. Dengan kata lain, alat tulis dan mesin ketik pun dapat dimasukkan sebagai salah satu teknologi yang digunakan selain komputer dan jaringan komputer [2].

Informasi adalah data yang telah diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi penerimanya. Sumber informasi adalah data. Data kenyataan yang menggambarkan suatu kejadian – kejadian dan kesatuan nyata. Kejadian – kejadian (*event*) adalah kejadian yang terjadi pada saat tertentu [5].

Untuk menghasilkan informasi diperlukan proses pengolahan data yang akurat, spesifik, dan tepat waktu. Hal ini sangat penting agar informasi mempunyai suatu nilai dan memberikan pemahamam kepada pembacanya.

Pada proses pengolahan data, untuk dapat menghasilkan informasi, juga dilakukan proses verifikasi secara akurat, spesifik, akurat, dan tepat waktu. Hal ini penting agar informasi dapat memberikan nilai dan pemahaman kepada pengguna. Pengguna dalam hal ini mencakup pembaca, pendengar penonton, bergantung bagaimana cara pengguna tersebut menikmati sajian informasi dan melalui media apa media tersebut disajikan [2].

Informasi adalah sebagai data yang telah diproses sedemikian rupa sehingga meningkatkan pengetahuan seseorang yang menggunakan data tersebut. Informasi

adalah data yang telah diolah menjadi suatu bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang [6].

2.1.3 Sistem Informasi

Berdasarkan definisi mengenai sistem dan informasi yang telah dijelaskan diatas, maka dapat dinyatakan bahwa sistem informasi merupakan gabungan dari empat komponen utama. Keempat komponen utama tersebut mencakup perangkat lunak (*software*), perangkat keras (*hardware*), infrastruktur dan sumber daya manusia (SDM) yang terlatih. Keempat bagian utama ini saling berkaitan untuk menciptakan sebuah sistem yang dapat mengolah data menjadi informasi yang bermanfaat [2].

Di dalamnya juga termasuk proses perencanaan, kontrol, koordinasi, dan pengambilan keputusan. Sehingga, sebagai sebuah sistem yang mengolah data menjadi informasi yang akan disajikan dan digunakan oleh pengguna, maka sistem informasi merupakan sebuah sistem yang kompleks. Bukan hanya komputer saja yang bekerja (beserta *software* dan *hardware* di dalamnya), namun juga manusia (dengan *brainware* yang dimiliki). Manusia (pengguna/aktor) dalam hal ini menggunakan seluruh ide, pemikiran, perhitungan, untuk dituangkan ke dalam sistem informasi yang di gunakan [2].

2.2 *Prototyping*

Prototyping merupakan Teknik pengembangan system yang menggunakan prototype untuk menggambarkan system, sehingga pengguna atau pemilik system mempunyai gambaran pengembangan system yang akan dilakukannya.

Dengan Teknik prototyping, pengembangan bisa membuat prototype lebih dahulu sebelum mengembangkan system yang sebenarnya.

Prototype sering diwujudkan dalam bentuk user interface program apikasi dan contoh-contoh reporting yang akan dihasilkan, sehigga dengan demikian pengguna system akan mempunyai gambaran tentang system yang akan digunakan nanti [7].

1. Reaksi Awal Dari Pengguna

   Saat penganalisis sistem menampilkan sebuah *prototype* sistem informasi, anda akan tertarik dengan reaksi pengguna dan pihak manajemen terhadap *prototype*. Anda ingin tahu secara mendetail bagaimana reaksi mereka saat bekerja dengan *prototype* serta apakah fitur – fitur sistem yang di-*prototype* sudah sesuai dengan kebutuhan mereka. Reaksi yang terkumpul dalam lembar observasi, wawancara, dan umpan balik (kemungkinan juga kuesioner) dirancang untuk mengurangi setiap pendapat pengguna mengenai. Melalui reaksi pengguna semacam itu, penganalisis akan menemukan beberapa perspektif mengenai *prototype*, termasuk apakah pengguna tampak senang dengannya atau apa ada kesulitan dalam menjual atau menerapkan sistem.
2. Saran – Saran dari pengguna

   Penganalisis juga tertarik dengan saran – saran pengguna dan pihak manajemen perbaikan atau pengubahan *prototype* yang ditampilkan. Saran – saran merupakan hasil dari interaksi pengguna dengan *prototype* serta refleksi mereka atas interaksi tersebut. Saran – saran yang diperoleh dari pengguna memberi petunjuk untuk menganalisis cara – cara memperbaiki, mengubah, atau "menghentikan" *prototype* sehingga bisa memenuhi kebutuhan pengguna dengan lebih baik.
3. Inovasi

   Inovasi adalah kemampuan – kemampuan sistem baru yang tidak dianggap berhubungan dengan waktu saat pengguna mulai berinteraksi dengan *prototype*. Inovasi – inovasi ini memberi nilai tambah terhadap fitur – fitur yang di-*prototypekan* sebelumnya dengan menambahkan sesuatu yang baru atau yang lebih inovatif.
4. Rencana Revisi

   Prototype menggambarkan sistem di masa datang. Rencana revisi membantu mengidentifikasi prioritas – prioritas apa yang akan di-*prototype* selanjutnya.

2.2.1 Tahapan – tahapan *Prototyping*

Berikut ini adalah tahapan – tahapan dalam *prototyping* [8]:

1. Pengumpulan kebutuhan

   Pelanggan dan *developer* bersama – sama mendefinisikan format keseluruhan perangkat lunak, mendefenisikan semua kebutuhan, dan garis besar sistem yang akan dibuat.

2. Membangun *prototype*

   Membangun *prototype* dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misal dengan *input* dan format *output*).

3. Evaluasi *prototyping*

   Evaluasi ini dilakukan oleh pelanggan apakah *prototyping* yang sudah dibangun sudah sesuai dengan keinginan pelanggan. Jika sudah sesuai maka tahapan berikutnya akan diambil. Jika tidak *prototyping* direvisi dengan mengulangi langkah pertama, kedua, dan ketiga.

4. Mengkodekan Sistem

   Dalam tahap ini yang sudah disepakati diterjemahkan kedalam Bahasa pemograman yang sesuai.

5. Menguji Sistem

   Setelah sistem sudah menjadi perangkat lunak yang siap dipakai, harus diuji coba dahulu sebelum digunakan. Pengujian ini dilakukan dengan *white box, black box, bassic path,* pengujian arsitektur dan lain – lain.

6. Evaluasi Sistem

   Pelanggan mengevaluasi apakah sistem yang sudah jadi sesuai dengan yang diharapkan. Jika iya tahapan selanjutnya dilakukan, jika tidak ulangi langkah ke empat dan ke lima

7. Menggunakan Sistem

   Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan.

2.3 *Use Case* Diagram

Use Case adalah sebuah kegiatan yang dilakukan oleh sistem, biasanya dalam menanggapi permintaan dari pengguna sistem [9].

Use Case adalah rangkaian atau uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang akan dilakukan atau diawasi oleh sebuah aktor. *Use case* digunakan untuk membentuk tingkah laku benda dalam sebuah model serta direalisasikan oleh sebuah kolaborasi [4].

Diagram *use case* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Hal yang ditekankan pada diagram ini adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah aktivitas atas pekerjaan tertentu, misalnya *registration* ke sistem, meng-*create* sebuah daftar belanja, dan lain sebagainya, aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan – pekerjaan tertentu [4].

2.3.1 Elemen – elemen Diagram *Use Case*

Berikut adalah beberapa elemen – elemen dari diagram *use case* [4]:

- **Sistem**

  Sistem yang menyatakan batasan sistem dalam relasi dengan *actor – actor* yang menggunakannya (di luar sistem) dan fitur – fitur yang harus disediakan (dalam sistem).

  Sistem digambarkan dengan segi empat yang membatasi semua *use case* dalam sistem terhadap pihak mana sistem akan berinteraksi. Sistem disertai label yang menyebutkan nama dari sistem, tapi umumnya tidak digambarkan karena tidak terlalu memberi arti tambahan pada diagram.

- ***Actor***

  Actor atau aktor dapat berupa merupakan manusia, sistem, atau *device* yang memiliki peranan dalam keberhasilan operasi dari sistem. Digambarkan dengan *icon* yang mungkin bervariasi namun konsepnya sama:

  1. Umumnya, untuk orang, digambarkan dengan sosok lengkap seperti, dengan kepala, badan, tangan, dan kaki.
  2. Umumnya, untuk sistem, digambarkan dengan segi empat disertai notasi "<<*actor*>>" di atas label nama.

- ***Use Case***

Use Case mengidentifikasi fitur kunci dari sistem. Tanpa fitur ini, sistem tidak akan memenuhi permintaan *user/actor*. Setiap *use case* mengekspresikan *goal* dari sistem yang harus dicapai. Diberi nama sesuai dengan *goal*-nya dan digambarkan dengan elips (dengan nama di dalamnya). *Focus* tetap pada *goal*, bukan "bagaimana" mengimplementasikannya walaupun *use case* berimplikasi pada prosesnya nanti.

- ***Assosiation***

  Assosiation berfungsi untuk mengidentifikasi interaksi antara setiap *actor* tertentu dengan setiap *use case* tertentu. Digambarkan dengan garis antara *actor* terhadap *use case* yang bersangkutan. Asosiasi bisa berarah (garis dengan anak panah) jika komunikasi satu arah, namun umumnya terjadi kedua arah (tanpa anak panah) karena selalu diperlukan demikian.

- ***Stereotape***

  Stereotape memungkinkan perluasan UML tanpa memodifikasinya. Berperan sebagai *kualifier* pada suatu elemen model. Menyediakan informasi lebih banyak mengenai peranan dari elemen tanpa menyebutkan implementasinya. Terutama untuk menggambarkan:

  1. *Use Case dependency*
  2. *Class – class*
  3. *Package – package*
  4. *Classifier*

  Notasi dalam diagram dengan *guile met* "<<…>>"

2.3.2 *Dependency*

Secara umum dependensi dilambangkan dengan tanda <<*include*>> yang berguna untuk :

1. Mengidentifikasi hubungan antar dua *use case* di mana yang satu memanggil yang lain.
2. Jika pada beberapa *use case* terdapat bagian yang memiliki aktivitas yang sama maka bagian aktivitas tersebut biasanya dijadikan *use case* tersendiri dengan relasi dependensi setiap *use case* semula ke *use case* yang baru ini sehingga memudahkan pemeliharaan.

3. Digambarkan dengan garis putus – putus bermata panah dengan notasi <<*include*>> pada garis.
4. Arah mata panah sesuai dengan arah pemanggilan.
5. Dependensi <<*extend*>>
    a. Jika pemanggilan memerlukan adanya kondisi tertentu maka berlaku dependensi <<*extend*>>.
    b. *Note*: konsep "*extend*" ini berbeda dengan "*extend*" dalam *java*!
    c. Digambarkan serupa dengan dependensi <<*include*>> kecuali arah panah berlawanan.

2.3.3 *Generalization*

Generalization berguna untuk mengidentifikasi relasi antara dua *actor* atau dua *use case*. Salah satunya meng-*inherit* dan menambahkan atau *override* sifat dari yang lainnya. Penggambaran menggunakan garis bermata panah indekosong dari yang meng-*inherit* mengarah ke yang di-*inherit* [4].

Sebagai acuan dalam membangun diagram *use case*, dapat menerapkan langkah – langkah [4]:

1. Set konteks dari target sistem.
2. Identifikasi semua *actor*.
3. Identifikasi semua *use case*.
4. Definisikan asosiasi antara setiap *actor* dan setiap *use case*.
5. Evaluasi setiap actor dan setiap use case untuk mendapatkan kemungkinan *refinement*.
6. Evaluasi setiap *use case* untuk dependensi <<*include*>>.
7. Evaluasi setiap *use case* untuk dependensi <<*entend*>>.
8. Evaluasi setiap *actor* dan setiap *use case* untuk generalisasi [4].

2.4 *Activity Diagram*

Activity Diagram memodelkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini sangat mirip dengan *flowchart* karena memodelkan *workflow* dari satu aktivitas ke aktivitas lainnya atau dari aktivitas ke status. Membuat *activity* diagram pada awal pemodelan proses cukup menguntungkan untuk membantu memahami keseluruhan proses. *Activity* diagram juga bermanfaat untuk menggambarkan *parallel behavior* atau menggambarkan interaksi antara beberapa *use* case [4].

Elemen – elemen dari *activity diagram* meliputi [4]:

1. Status *start* (mulai) dan *end* (akhir).
2. Aktivitas yang merepresentasikan sebuah langkah dalam *workflow*.
3. *Transition* menunjukkan terjadinya perubahan status aktivitas (*transitions show what state follows another*).
4. Keputusan yang menunjukkan *alternative* dalam *workflow*.
5. *Synchronization bars* yang menunjukkan *subflow parallel*. *Synchronization bars* dapat digunakan *concurrent threads* pada *workflow* proses bisnis.
6. *Swimlanes* yang merepresentasikan *role* bisnis yang bertanggung jawab pada aktivitas yang berjalan.

2.5 PIECES

Untuk mengidentifikasi masalah, harus dilakukan analisis terhadap kinerja, informasi, ekonomi, keamanan aplikasi, efisiensi, dan pelayanan pelanggan. Panduan ini dikenal dengan analisis PIECES (*Performance*, *informance*, *economy*, *control*, dan *services*). Dari analisis ini biasanya didapatkan beberapa masalah utama. Hal ini penting karena biasanya didapatkan beberapa masalah utama. Hal ini penting karena biasanya yang muncul di permukaan bukan masalah utama, tetapi hanya gejala dari masalah utama [6].

P = Analisis Kinerja

Masalah kinerja terjadi ketika tugas - tugas bisnis yang dijalankan tidak mencapai sasaran. Kinerja diukur dengan jumlah produksi dan waktu tanggap.

I = Analisis Informasi

Evaluasi terhadap kemampuan sistem informasi dalam menghasilkan informasi yang bermanfaat perlu dilakukan untuk menyikapi peluang dan menangani masalah yang muncul.

E = Analisis Ekonomi

Alasan ekonomi merupakan motivasi paling umum bagi suatu proyek, pijakan dasar bagi suatu manager adalah biaya atau rupiah, persoalan ekonomis dan peluang berkaitan dengan biaya

Biaya biasanya terdiri dari biaya tidak diketahui, biaya tidak dapat dilacak ke sumber dan biaya terlalu tinggi.

C = Analisis Keamanan

Tugas - tugas bisnis perlu dimonitor dan dibetulkan jika ditemukan kinerja yang dibawah standar. Kontrol dipasang untuk meningkatkan kinerja sistem, mencegah, atau mendeteksi kesalahan sistem, menjamin keamanan data, informasi, dan persyaratan

E = Analisis Efisiensi

Efisiensi menyangkut bagaimana menghasilkan output sebanyak banyaknya dengan inputan sekecil mungkin

S = Layanan

Kualitas layanan suatu sistem bisa dikatakan buruk apabila sistem menghasilkan produk yang tidak akurat, tidak konsisten, tidak percaya, sistem tidak mudah dipelajari, tidak mudah digunakan, sistem canggung dan tidak fleksibel [6].

2.6 Kos

Kos merupakan suatu tempat tinggal yang disewakan kepada pihak lain dengan fasilitas-fasilitas tertentu dengan harga yang lebih terjangkau daripada di hotel/penginapan. Rumah indekos lebih akrab digunakan sebagai domisili, karena kebanyakan tempat indekos disewa dalam jangka waktu yang cukup lama dari pada hotel atau penginapan yang menggunakan hitungan hari. Dan juga istilah tempat indekos sangatlah berdampingan dengan mahasiswa, karena pada umumnya tempat indekos disewakan untuk mahasiswa walaupun tidak jarang juga tempat indekos yang

disewakan untuk umum. Tempat indekos sangatlah bermacam-macam, dari cara penyewaannya, fasilitas-fasilitas dan harga yang bervariasi. Dan tempat indekos ini adalah merupakan suatu investasi yang cukup menjanjikan yang dimana kita dapat menghitung biaya perbulan dengan yang dihasilkan disetiap bulannya [1].

2.7 *Android*

Android adalah sistem operasi berbasis Linux yang dimodifikasi untuk perangkat bergerak (*mobile devices*) yang terdiri dari sistem operasi, *middleware*, dan aplikasi - aplikasi utama. Awalnya, *Android* dikembangkan oleh *Android* Inc. perusahaan ini kemudian dibeli oleh Google pada tahun 2005. Sistem operasi *Android* kemudian diluncurkan bersamaan dengan dibentuknya organisasi *Open Handset Alliance* tahun 2007. Selain google, beberapa nama - nama besar juga ikut serta dalam *Open Handset Alliance,* antara lain Motorola, Samsung, LG, *Sony Ericsson*, t-*mobile*, Vodafone, Toshiba, dan Intel [10].

Android telah mengalami sejumlah pembaruan sejak pertama kali dirilis. Rata-rata, versi terbaru dari *Android* dirilis setiap 6 bulan. Table 1.1 menunjukan beberapa jenis *Android* dan nama kodenya. Penamaan kode menggunakan nama makanan dan huruf depannya urut sesuai abjad.

Tabel 2. 1 Daftar *Android*

Versi	Tanggal rilis	Kode
1.1	9 Februari 2009	
1.5	30 April 2009	Cupcake
1.6	15 September 2009	Donut
2.0/2.1	26 Oktober 2009	Éclair
2.2	20 Mei 2010	Frozen Yoghurt (Froyo)
2.3	6 Desember 2010	Gingerbread
3.0	22 Februari 2011	Honeycomb

4.0	19 Oktober 2011	Ice Cream Sandwich
4.1	27 Juni 2012	Jelly Bean
4.2	29 Oktober 2012	Jelly Bean
4.3	24 Juli 2013	Jelly Bean
4.4	3 September 2013	Kitkat
5.0/5.1	15 Oktober 2014	Lollipop
6.0	1 September 2015	Marshmallow
7.0	30 Juni 2016	Nougat
8.0	1 Agustus 2017	Oreo

Berikut beberapa jenis *Android* dan penjelasannya:

a. *Cupcake*

Android 1.5 (API Level 3) yang dikenal dengan nama kode *Cupcake* dirilis pada April 2009. API Level 3 membawa perbaikan dari versi sebelumnya dan tentu saja fitur baru. Misalnya, dukungan *On-screen keyboard*, video *recording*, *home screen widget*, fitur *auto pairing* pada *Bluetooth*, *kernel* Linux versi 2.6.7 ,dan kemampuan memperbaiki *filesystem* SD *card* yang rusak.

b. *Donut*

Android 1.6 (API Level 4) yang dirilis pada Oktober 2009 ini merupakan rilis minor dengan fitur utama berupa dukungan teknologi CDMA, *gesture, text-to-speech engine*, dan fitur pencarian cepat yang memungkinkan anda mencari semua hal seperti kontak, riwayat jelajah internet, *bookmark*, dan lainnya.

c. *Eclair*

Android 2.1 (API Level 7) yang dirilis pada Januari 2010 mengusung fitur baru seperti *live wallpaper* dan berupa fitur baru seperti *Android* 2.0, misalnya menggunakan banyak akun, kontak cepat, *Bluetooth* 2.1, dan profil *Bluetooth* baru yaitu *Object Push Profile* (OPP) dan *Phone Boom Access Profile* (PBAP).

d. *Froyo*

Android 2.2 (API Level 8) dirilis pada Mei 2010. Fitur baru yang dibawa adalah dukungan *OpenGL* ES 2.0, instalasi aplikasi penyimpan eksternal (SD *card*), dan *Android Cloud to Device Messaging* yang memungkinkan aplikasi melakukan *push messaging* dan *portable hostspot* yang memungkinkan piranti *Android* menjadi *hotspot Wi-Fi* untuk berbagi koneksi internet.

e. *Gingerbread*

Android 2.3 (API Level 9 dan 10) mengusung beberapa fitur, antara lain dukungan banyak kamera, *Near Field Communication (NFC)*, *download manager service*, dukungan terhadap sensor lain seperti giroskop, dan barometer.

f. *Honeycomb*

Android 3.0 (API Level 11) membawa perubahan besar terutama pada tampilan UI yang berubah drastis supaya optimal untuk piranti layar besar seperti tablet. Pada versi ini diperkenalkan *fragment*, *action bar*, *system clipboard*, dan *cursor loader*.

g. *Ice Cream Sandwich*

Android 4.0 (API level 11) membawa perubahan besar terutama pada tampilan UI *Android* 3.0 supaya cocok untuk layar kecil sehingga memungkinkan aplikasi anda tampak konsisten di tablet maupun ponsel. Fitur baru seperti *Android beam* dan *Wi-Fi Direct* juga ditambahkan.

h. *Jelly Bean*

Android 4.1 (API Level 16) adalah rilis minor yang membawa perbaikan fungsionalitas dan performa *rendering User-Interface* (UI). Versi 4.2 (API Level 17) mengusung fitur baru seperti *gesture typing* dan dukungan *multiuser* di piranti tablet. Versi 4.3 (API Level 18) membawa perbaikan berupa dukungan *bluetooth low energy* dan *Open* GL ES 3.0. Beberapa fitur baru antara lain dukungan terhadap Bahasa internasional dan penulisan teks dua arah (kiri ke kanan atau kanan ke kiri untuk Bahasa-bahasa tertentu seperti Bahasa Arab).

i. *Kitkat*

Android 4.4 (API Level 19) ini mengusung sejumlah perbaikan dan fitur baru, terutama dukungan teknologi *Near Field Communication* (NFC) melalui *host card emulation,* pencetakan ke printer nirkabel, *webview* dengan *rendering engine Chromium*, dan dukungan yang lebih baik bagi piranti yang menggunakan RAM rendah.

2.8 Basis Data

Basis data adalah kumpulan file yang disimpan dan dapat dibagikan jika sewaktu-waktu ada pengguna yang membutuhkannya. Basis data diperiksa menggunakan suatu program computer untuk memperoleh informasi dari basis data tersebut [11].

Tujuan efektifitas dari basis data adalah sebagai berikut [11]:

1. memastikan bahwa data dapat dibagi di antara pengguna untuk berbagi aplikasi.
2. memelihara data yang akurat dan konsisten.
3. mengumpulkan semua data untuk aplikasi yang akan tersedia di masa mendatang.

2.9 kamus data

Kamus data adalah suatu proses penyusunan data yang dilakukan oleh penganalisis system. Hal ini bertujuan untuk membimbing penganalisis selama melakukan kegiatan analisis dan desain. Sebagai suatu dokumen, kamus data akan mengumpulkan istilah istilah data tertentu dan menjelaskan arti dari setiap istilah yang ada.

Kamus data sangat berguna karena memiliki kapasitas dalam mereferensikan item-item data, dengan demikian memungkinan dilakukannya perubahan-perubahan program terhadap semua program yang berbagi suatu elemen biasa [11].