BAB II
LANDASAN TEORI

Untuk mendukung pembuatan laporan ini, maka perlu dikemukakan hal-hal atau yang berkaitan dengan permasalahan dan ruang lingkup pembahasan sebagai landasan dalam pembuatan laporan ini.

2.1 Definisi Dasar Sistem

Sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen atau variable yang terorganisir, saling berinteraksi, saling tergantung satu sama lain, dan terpadu. (Sutabri, 2012: 10).

Sistem adalah kumpulan elemen yang saling berhubungan dan berinteraksi dalam satu kesatuan untuk menjalankan suatu proses pencapaian suatu tujuan utama. (Sutarman, 2012:13).

Sistem bisa berupa abstrak atau fisis. Sistem yang abstrak adalah susunan yang teratur dari gagasan-gagasan atau konsepsi yang saling bergantung. Sedangkan sistem yang bersifat fisis adalah serangkaian unsur yang bekerjasama untuk mencapai suatu tujuan. (Gordon B. Davis dalam Sutabri, 2012:17).

Berdasarkan definisi di atas dapat disimpulkan mengenai sistem. Sistem yaitu suatu kumpulan dari bagian-bagian yang saling berhubungan dan berinteraksi membentuk satu kesatuan untuk mencapai tujuan tertentu.

2.2 Konsep Dasar Informasi

2.2.1 Definisi Informasi

Gordon B. Davis (dalam Sutabri, 2012: 1) menjelaskan kaitannya data dengan informasi dalam bentuk definisi berikut "Informasi adalah data yang telah diproses ke dalam suatu bentuk yang mempunyai arti bagi si penerima dan mempunya nilai nyata dan terasa bagi keputusan saat itu atau keputusan mendatang".

Informasi adalah data hasil pemrosesan yang memiliki makna, biasanya menceritakan suatu hal yang belum diketahui kepada pengguna (Mcle1od, 2012: 11).

Berdasarkan definisi diatas dapat disimpulkan pengertian informasi. Informasi adalah data hasil pemrosesan yang memiliki arti bagi si penerima tentang sesuatu yang belum diketahui.

2.3 Konsep Dasar Sistem Informasi

2.3.1 Definisi Sistem Informasi

Turban Mclean, dan Wetherbe (dalam Darmawan, 2013: 26) menyatakan bahwa sistem informasi adalah sebuah sistem informasi yang mempunyai fungsi mengumpulkan, memproses, menyimpan, menganalisis, dan menyebarkan informasi untuk tujuan yang spesifik.

Bodnar dan Hopwood (dalam Darmawan, 2013: 27) menyatakan bahwa sistem informasi adalah kumpulan perangkat keras dan lunak yang dirancang untuk mentransformasikan data ke dalam bentuk informasi yang berguna.

Alter (dalam Darmawan, 2013: 27) menyatakan sistem informasi adalah kombinasi antara prosedur kerja, informasi, orang dan teknologi informasi yang diorganisasikan untuk mencapai tujuan dalam sebuah perusahaan.

2.3.2 Komponen-Komponen Sistem Informasi

Sistem informasi mempunyai enam buah komponen yaitu :

a) Komponen Input

   Input merupakan data yang masuk ke dalam sistem informasi. Komponen ini harus ada karena merupakan bahan dasar dalam pengolahan informasi.

b) Komponen Model

   Informasi yang dihasilkan oleh sistem informasi berasal dari data yang diambil dari basis data yang di olah lewat suatu model-model tertentu. Model logika yang menunjukkan suatu proses perbandingan logika atau matematik yang menunjukkan proses perhitungan matematika.

c) Komponen Output

   Produk dari sistem informasi adalah Output berupa informasi yang berguna bagi para pemakainya.

d) Komponen Teknologi

   Teknologi merupakan komponen yang penting . tanpa adanya Teknologi yang mendukung, maka sistem informasi tidak akan dapat menghasilkan informasi yang tepat waktunya.

e) Komponen Basis Data

   Kumpulan dari data yang saling berhubungan satu dengan yang lainya, tersimpan di perangkat keras computer dan digunakan perangkat lunak untuk memanipulasinya.

f) Komponen Kontrol

Komponen kontrol ini digunakan untuk menjamin bahwa informasi yang dihasilkan oleh sistem informasi merupakan informasi yang kuat.

2.4 Android

2.4.1 *Definisi Android*

Android menurut Satyaputra dan Aritonang (2014: 2) adalah sebuah sistem operasi untuk *smartphone* dan tablet. Sistem operasi dapat diilustrasikan sebagai jembatan antara piranti *(device)* dan penggunaannya, sehingga pengguna dapat berinteraksi dengan *device*-nya dan menjalankan aplikasi-aplikasi yang tersedia pada *device.*

Referensi lain ditemukan bahwa Arif Akbarul Huda (2013: 1-5) berpendapat mengenai *Android* merupakan sistem operasi berbasis Linux yang khusus untuk perangkat bergerak seperti *smartphone* atau tablet.

2.4.2 *Sejarah Android*

Onur Cinar (2012: 28) mengemukakan bahwa "*Android Inc. was founded in Silicon Valley, California, in October 2003, with the idea of providing a mobile platform that is more aware of user's location and preferences*".

Perkembangan *Android* dimulai dengan berdirinya *Android*, Inc. pada Oktober 2003 dengan tujuan *mobile device* yang lebih pintar untuk menyaingi *Symbian* dan *Windows Mobile* yang populer pada saat itu dimana iPhone dan Blackberry belum dirilis. Pada tahun 2005, *Android* diakuisisi oleh Google. Pengembangan terus dilanjutkan sampai *Android* versi beta diluncurkan pada tanggal 5 November 2007. Hingga saat ini tanggal 5 November diperingati sebagai hari jadi *Android*. Seminggu setelahnya yaitu pada tanggal 12 November 2007 *Android* SDK *(Software Development Kit)* diluncurkan, sehingga pengguna dapat membuat dan mengembangkan aplikasi-aplikasi *Android* mereka sendiri (Satyaputra dan Aritonang, 2014: 5).

2.4.3 Versi-versi Android

Sistem operasi Android ini sangatlah unik dan mampu memberikan kemudahan bagi para pengguna karena nama sistem operasinya selalu berdasarkan nama makanan dan diawali dengan abjad yang berurutan seperti:

Tabel 2.1 Versi *Android*

Versi	API LEVEL	Tanggal Rilis	Nama
1.0	1	23 September 2008	Android 1.0
1.1	2	09 Februari 2009	Android 1.1
1.5	3	30 April 2009	Cupcake
1.6	4	15 September 2009	Donut
2.0	5	26 Oktober 2009	Eclair
2.0.1	6	03 Desember 2009	Eclair
2.1	7	12 Januari 2010	Eclair
2.2	8	20 Mei 2010	Froyo
2.2.1	8	18 Januari 2011	Froyo
2.2.2	8	22 Januari 2011	Froyo
2.2.3	8	21 November 2011	Froyo
2.3	9	06 Desember 2010	Gingerbread
2.3.1	9	Desember 2010	Gingerbread
2.3.2	9	Januari 2011	Gingerbread
2.3.3	10	09 Februari 2011	Gingerbread
2.3.4	10	28 April 2011	Gingerbread
2.3.5	10	25 Juli 2011	Gingerbread
2.3.6	10	02 September 2011	Gingerbread
2.3.7	10	21 September 2011	Gingerbread
3.0	11	22 Februari 2011	Honeycomb
3.1	12	10 Mei 2011	Honeycomb
3.2	13	15 Juli 2011	Honeycomb
3.2.1	13	20 September 2011	Honeycomb
3.2.2	13	30 Agustus 2011	Honeycomb
3.2.3	13	-	Honeycomb
3.2.4	13	Desember 2011	Honeycomb
3.2.5	13	Januari 2012	Honeycomb
3.2.6	13	Februari 2012	Honeycomb
4.0	14	19 Oktober 2011	Ice Cream Sandwich
4.0.1	14	21 Oktober 2011	Ice Cream Sandwich
4.0.2	14	28 November 2011	Ice Cream Sandwich
4.0.3	15	16 Desember 2011	Ice Cream Sandwich
4.04	15	29 Maret 2012	Ice Cream Sandwich
4.1	16	09 Juli 2012	Jelly Bean

4.1.1	16	23 Juli 2012	Jelly Bean
4.1.2	16	09 Oktober 2012	Jelly Bean
4.2	17	13 November 2012	Jelly Bean
4.2.1	17	27 November 2012	Jelly Bean
4.2.2	17	11 Februari 2013	Jelly Bean
4.3	18	23 Juli 2013	Jelly Bean
4.4	19	21 Oktober 2013	Kitkat
5.0	21	25 Juni 2014	Lolipop
6.0	23	-	Marsmallow
7.0	24	22 Agustus 2016	Nougat

Sumber: https://id.wikipedia.org/wiki/Daftar_versi_Android (diakses pada 10 Agustus 2017)

2.4.4 *Komponen Aplikasi* Android

*M*enurut Arif Akbarul Huda (2013: 4-5) komponen aplikasi merupakan bagian penting dari sebuah *Android*. Setiap komponen mempunyai fungsi yang berbeda, dan antara komponen satu dengan yang lainnya bersifat saling berhubungan. Berikut ini komponen aplikasi yang harus diketahui, yaitu:

2.4.4.1 Activities.

Activity merupakan satu halaman antarmuka yang bisa digunakan oleh *user* untuk berinteraski dengan aplikasi. Biasanya dalam satu *activity* terdapat *button*, *spinner*, *list view*, *edit text*, dan sebagainya. Satu aplikasi dalam *Android* dapat terdiri atas lebih dari satu *activity*.

2.4.4.2 Services.

Services merupakan komponen aplikasi yang dapat berjalan secara *background*, misalnya digunakan untuk memuat data dari *server database*. Selain itu, aplikasi pemutar musik atau radio juga memanfaatkan servis supaya aplikasinya bisa tetap berjalan meskipun pengguna melakukan aktivitas dengan aplikasi lain.

2.4.4.3 Contact Provider.

Komponen ini digunakan untuk mengelola data sebuah aplikasi, misalnya kontak telepon. Siapapun bisa membuat aplikasi *Android* dan dapat mengakses kontak yang tersimpan pada sistem *Android*. Oleh karena itu, agar dapat mengakses kontak, user memerlukan komponen *contact provider*.

2.4.4.4 Broadcast Receiver.

Fungsi komponen ini sama seperti bahasa terjemahannya yaitu penerima pesan. Kasus beterai lemah merupakan kasus yang sering dialami *handphone Android*. Sistem *Android* dirancang untuk menyampaikan "pengumuman" secara otomatis jika baterai habis. Apabila aplikasi yang dibuat dilengkapi dengan komponen *broadcast receiver*, maka *user* dapat mengambil tindakan menyimpan kemudian menutup aplikasi atau tindakan yang lain.

2.5 UML *(Unified Modeling Language)*

*2.5.1 Definisi UML (*Unified Modeling Language*)*

Menurut Alan Dennis (2012:513), UML (*Unified Modeling Language*) merupakan kosakata umum berbasis objek dan diagram teknik yang cukup efektif untuk memodelkan setiap proyek pengembangan sistem mulai tahap analisis sampai tahap desain dan implementasi.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk:

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.
5. Blok pembangunan utama UML adalah diagram. Beberapa diagram ada yang rinci (jenis *timing* diagram) dan lainnya ada yang bersifat umum (misalnya diagram kelas).Para pengembang sistem berorientasi objek menggunakan bahasa model untuk mengambarkan, membangun dan mendokumentasikan sistem yang mereka rancang. UML memungkinkan para anggota *team* untuk bekerja sama dengan bahasa model yang sama dalam mengaplikasikan beragam sistem. Intinya, UML merupakan alat komunikasi yang konsisten dalam men*support* para pengembang sistem saat ini.Diagram *Use Case,* Diagram Aktivitas (*Activity Diagram*), Diagram *Sequence,* dan Diagram *Class.*

2.5.2 Fokus Unified Modeling Language *(UML)*

Pemetaan (*mapping*) Unified Modeling Language (UML) bersifat 2 (dua) arah, yaitu:

1. Generasi kode bahasa pemrograman tertentu dari Unified Modeling Language (UML) *foward engineering*.
2. Generasi kode belum sesuai dengan kebutuhan dan harapan pengguna, pengembang dapat melakukan langkah baik bersifat *iterative* dari implementasi ke Unified Modeling Language (UML) hingga didapat sistem atau piranti lunak yang sesuai dengan harapan pengguna dan pengembang .

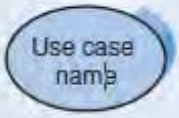
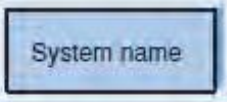
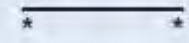
2.5.3 Diagram – Diagram UML (Unified Modeling Language)

Beberapa *literature* menyebutkan bahwa UML menyediakan Sembilan jenis diagram. Namun kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan. Diagram yang sering digunakan adalah Diagram *Use Case*, Diagram Aktivitas (*Activity Diagram*), Diagram *Sequance*, dan Diagram *Class*.

2.5.4 Use Case Diagram

Use Case Diagram secara grafis menggambarkan, interaksi secara sistem, sistem eksternal dan pengguna. Dengan kata lain *use case* diagram secara grafis mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna (*user*) mengharapkan interaksi dengan sistem itu. *Use case* secara naratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari tiap interaksi.*Use case* diagram merupakan suatu diagram yang menangkap kebutuhan bisnis untuk sistem dan untuk menggambarkan interaksi antara sistem dan lingkungannya. (Dennis *et al*, 2012:513).

Tabel 2.2 Notasi *Use Case* Diagram (Dennis *at al*:2012)

Term and Definition	Symbol
An actor ■ Is a person or system that derives benefit from and is external to the system. ■ Is labeled with its role. ■ Can be associated with other actors by a specialization/superclass association, denoted by an arrow with a hollow arrowhead. ■ Is placed outside the system boundary.	 Actor role name
A use case ■ Represents a major piece of system functionality. ■ Can extend another use case. ■ Can use another use case. ■ Is placed inside the system boundary. ■ Is labeled with a descriptive verb–noun phrase.	
A system boundary ■ Includes the name of the system inside or on top. ■ Represents the scope of the system.	
An association relationship ■ Links an actor with the use case(s) with which it interacts.	

1. Aktor (actor), menggambarkan pihak-pihak yang berperan disebuah sistem.
2. Use case, aktifitas / sarana yang disiapkan oleh bisnis / sistem.
3. System boundary, adalah sebuah kotak yang mewakili sebuah sistem.
4. Hubungan (link), aktor mana saja yang terlibat dalam Use case, dan bagaimana hubungan Use case dengan Use case lain. ada hubungan antar Use case. Digolongkan menjadi 2 : yaitu extend digambarkan dengan keterangan <<extend>>, dan include digambarkan dengan keterangan <<include>>, berikut perbedaanya dijelaskan pada tabel dibawah ini:

Tabel 2.3 Perbedaan *include* dan *extend* pada *Use Case* (Dennis *et al*:2012)

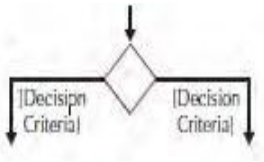
Include	*Extend*
Use case terpanggil (*included use case*) selalu diperlukan oleh *use case* dasar	*Use case* ekstensi tidak selalu dibutuhkan oleh *use case* dasar
Yang memutuskan kapan dipanggilnya *use case included* adalah *use case* dasar	Yang memutuskan kapan dipanggilnya *use case extend* adalah *use case extend* itu sendiri
Panah hubungan dari *use case* dasar ke *use case include*	Panah hubungan dari *use case extend* ke *use case* dasar

2.5.5 Activity Diagram

Secara grafis untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *use case*. *Activity Diagram* dapat juga digunakan untuk memodelkan *action* yang akan dilakukan saat operasi dieksekusi, dan memodelkan hasil dari *action* tersebut.Pengertian diagram *Activity* adalah yang menggambarkan alur kerja bisnis independen dari *class*, aliran kegiatan dalam *Use case*, atau desain rinci sebuah metode (Dennis *et al* 2012 :516).

Tabel 2.4 Notasi Pemodelan *Activity Diagram* (Dennis *et al*, 2012 : 516)

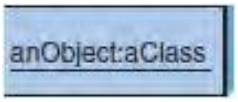
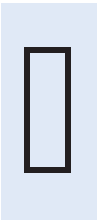
Actor Digunakan untuk melakukan tindakan .	
Activity Digunakan untuk mewakili serangkaian tindakan.	
Object Node Digunakan untuk mewakili suatu objek yang terhubung ke satu set Arus Obyek.	Class Diagram
Control Flow Menunjukkan urutan eksekusi.	

Object Flow Menunjukkan arus dari sebuah objek dari satu kegiatan (atau tindakan) untuk kegiatan lain (atau tindakan).	---------->
Initial Node Menggambarkan awal dari serangkaian tindakan atau kegiatan.	
Initial activity Node Digunakan untuk menghentikan semua arus kontrol dan arus objek dalam suatu kegiatan (atau tindakan).	
Decision Node Digunakan untuk mewakili kondisi tes untuk memastikan bahwa aliran kontrol atau aliran objek hanya turun satu jalur.	
Fork Node Adalah node kontrol yang memiliki satu dan dua atau lebih aliaran keluar.	
Join Node Adalah gabungan dari satu atau lebih activity aliran masuk.	
Swimline Digunakan untuk memecah sebuah diagram aktivitas dalam baris dan kolom untuk menetapkan aktivitas individu (atau tindakan) kepada individu atau benda yang bertanggung jawab untuk melaksanakan kegiatan (atau tindakan).	

2.5.6 DiagramSequence

Diagram *Sequence* merupakan urutan model dinamis yang menggambarkan contoh *class* yang berpartisipasi dalam *use case* dan pesan yang lewat di antara mereka dari waktu ke waktu. (Dennis *et al* 2012 : 540).*Sequence diagram* merupakan diagram interaksi yang disusun berdasarkan urutan waktu. Cara membaca diagram sekuensial dari atas ke bawah. Setiap diagram sekuensial mempresentasikan satu *flow* dari beberapa *flow* didalam *use case*.

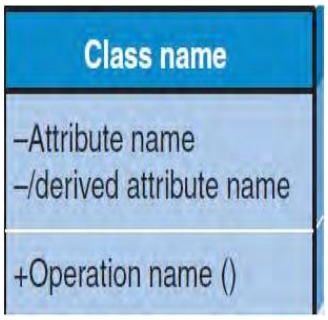
Tabel 2.5 Notasi Pemodelan Komponen *Sequence* Diagram(Dennis *et al* :2012)

Term and Definition	*Symbol*
An Actor (Aktor) • Orang atau sistem yang berasal dari luar sistem yang dapat memberikan manfaat. • Berpartisipasi secara berurutan oleh mengirim dan / atau menerima pesan. • Ditempatkan dibagian atas diagram.	
An Object (sebuah Objek) • Berpartisipasi secara berurutan oleh mengirim dan / atau menerima pesan. • Ditempatkan dibagian atas diagram.	
A Lifeline (sebuah garis hidup) • Menandakan kehidupan sebuah objek selama squance. • Berisi sebuah X pada titik dimana kelas tidak lagi berinteraksi	
A Focus of Control (Sebuah Fokus kontorl) • Menandakan sebuah persegi panjang yang sempit ditempatkan diatas sebuah garis hidup. • Menandakan ketika suatu objek mengirim atau menerima pesan.	
A Massege (sebuah Pesan) • Menyampaikan informasi dari satuobjek ke objek yang lain.	
Object destruction (Objek Penghabisan) • Merupakan sebuah X ditempatkan pada akhir suatu garis hidup untuk menunjukkan bahwa itu akan keluar dari eksistensi.	

2.5.7 Class Diagram

Class Diagram adalah ilustrasi antara *class* yang dimodelkan didalam sistem.*Class Diagram* sangat mirip dengan diagram hubungan entitas (ERD). Diagram *class* menggambarkan *class* yang meliputi atribut, perilaku dan *states*, sementara dalam ERD hanya mencakup atribut.(Dennis*et al*, 2012:513). Komponen *class diagram* :

Tabel 2.6 Komponen *Class* Diagram (Dennis *et al* :2012)

Term and Definition	*Symbol*
A Class (sebuah *class*) • Mewakili jenis orang, tempat atau hal yang sistem harus menangkap dan menyimpan informasi. • Memiliki nama yang diketik dengan huruf tebal dan berpusat diatas kompartemen. • Memiliki daftar atribut ditengah Kompartemen. • Memiliki daftar operasi	 Class name –Attribute name –/derived attribute name +Operation name ()
An Attribut (sebuah atribut) • Merupakan sifat yang menggambarkan bagian suatu objek. • Dapat diturunkan dari atribut lain, ditunjukkan oleh penempatan garis miring sebelum nama atribut.	*Attribut name / derived attribut name*
A Method (sebuah metode) • Merupakan tindakan atau fungsi bahwa sebuah class dapat melakukan. • Dapat diklasifikasikan sebagai konstruktor, query, atau memperbaharui operasi. • Termasuk tanda kurung yang mungkin mengandung parameter khusus atau informasi yang dibutuhkan untuk melakukan operasi.	*Operation name ()*

An Association (sebuah asosiasi) • Merupakan hubungan antara beberapa *class* atau *class* dirinya sendiri. • Diberi label oleh kata kerja frase mana yang merupakan hubungan yang tepat. • Bisa ada diantara satu atau lebih *class* • Berisi banyaknya simbol yang mewakili minimum dan maximum misalnya waktu class dapat dikaitkan dengan contoh class lain.	1..* *verb phrase* 0..1

Class Diagram menggambarkan *class* dan hubungan antar-*class* di dalam sistem.*Class Diagram* dibangun berdasarkan *use case diagram, sequence diagram,* atau *collaboration diagram* yang telah dibuat sebelumnya.

Diagram *Class* memberikan pandangan secara luas dari suatu sistem dengan menunjukkan kelas-kelasnya dan hubungan mereka. Diagram *Class* bersifat statis, menggambarkan hubungan apa yang terjadi bukan yang terjadi jika mereka berhubungan. Sebuah *class* memiliki tiga area pokok, yaitu:

3. Nama, merupakan nama dari sebuah kelas.
4. Atribut, merupakan properti dari sebuah kelas. Atribut melambangkan batas nilai yang mungkin ada pada objek dari kelas.
5. Operasi, merupakan sesuatu yang bisa dilakukan oleh kelas lain terhadap sebuah kelas.

2.6 Perangkat Lunak Pendukung

2.6.1 Edraw Max

Edraw Max adalah perangkat lunak diagram teknis bisnis 2D yang membantu membuat diagram alir, bagan organisasi, peta pikiran, diagram jaringan, denah lantai, diagram alur kerja, diagram bisnis, dan diagram teknik.

2.6.2 Balsamiq mockup

Mockup adalah sebagai sebuah model dari suatu struktur atau alat baik *full size* ataupun berupa miniatur yang digunakan untuk pembelajaran, demo, *test* desain, promosi, dsb.

Balsamiq mockup adalah program aplikasi yang digunakan dalam pembuatan tampilan user interface sebuah aplikasi. Software ini sudah menyediakan tools yang dapat memudahkan dalam membuat desain prototyping aplikasi yang akan kita buat. Software ini berfokus pada konten yang ingin digambar dan fungsionalitas yang dibutuhkan oleh pengguna.