BAB II

LANDASAN TEORI

2.1. Konsep Dasar Sistem

2.1.1. Definisi Sistem

Sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel-variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu. Terdapat dua kelompok pendekatan di dalam pendefinisian sistem, yaitu kelompok yang menekankan pada prosedur dan kelompok yang menekankan pada elemen atau komponennya.

Menurut Khadir (2014:61) menyatakan bahwa "sistem adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan".

Sedangkan menurut Sutabri dalam Rahmi dan Muryani (2018:143) menyatakan bahwa "Suatu sistem terdiri atas objek-objek atau unsur-unsur atau komponen-komponen yang berkaitan dan berhubungan satu sama lainnya sedimikian rupa sehingga unsur-unsur tersebut merupakan suatu kesatuan pemrosesan atau pengolahan yang tertentu".

Dari pendapat yang dikemukakan di atas dapat disimpulkan bahwa sistem adalah suatu kumpulan atau kelompok dari elemen atau komponen yang saling berhubungan atau saling berinteraksi dan saling bergantung satu sama lain untuk mencapai tujuan tertentu.

2.1.2. Karakteristik Sistem

Dalam pembuatan suatu sistem, pembuat sistem harus memehami ciri-ciri atau karakteristik yang terdapat pada sekumpulan elemen yang ada sebagai dasar pertimbangan dalam pembuatan sistem.

Menurut Tohari (2014:2-3), karakteristik sistem sebagai berikut:

1. Komponen atau Elemen (*Components*)

   Suatu sistem terdiri dari komponen-komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk suatu kesatuan

2. Batasan Sistem (*Boundary*)

   Batasan sistem merupakan daerah yang mebatasi antara sistem yang satu dengan ssitem 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 tetap saling berinteraksi. Dengan kata lain, batas sistem merupakan ruang lingkup atau *scope* dari sistem atau subsitem itu sendiri.

3. Lingkungan Luar Sistem (*Environment*)

   Lingkungan luar sistem adalah segala sesuatu diluar batas sistem yang memepengaruhi 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 dengn subsitem 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 sesuatu yang dimasukkan ke dalam 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*)

   Merupalan 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 (*Objekive*)

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

2.1.3. Klasifikasi Sistem

Sistem merupakan suatu bentuk integrasi antara satu komponen dengan komponen lainnya. Karena sistem memiliki sasaran yang berbeda untuk setiap kasus yang terjadi yang ada didalam sistem tersebut.

Menurut Ladjamudin (2013:6), sistem dapat diklasifikasikan dari berbagai sudut pandang, yaitu:

1. Sistem diklasifikasikan sebagai Sistem Abstrak dan Sistem Fisik

   Sistem abstarak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Sistem fisik adalah sistem yang ada secara fisik.

2. Sistem diklasifikasikan sebagai Sistem Alamiah dan Sistem Buatan Manusia

   Sistem Alamiah adalah sistem yang terjadi karena proses alam tidak dibuat oleh manusia (ditentukan dan tunduk kepada kehendak sang pencipta alam). Sistem Buatan Manusia adalah sistem yang dirancang manusia. Sistem buatan manusia yang melibatkan interaksi manusia dengan mesin disebut dengan *human machine system* atau ada yang menyebut dengan *machine system.*

3. Sistem diklasifikasikan sebagai Sistem tertentu (*Deterministic system*) dan Sistem tak tertentu (*Probabilistic system*)

   Sistem tertentu beroperasi dengan tingkah laku yang dapat diprediksi. Interaksi diantara bagian-bagiannya dapat dideteksi dengan pasti, sehingga keluaran dari sistem dapat diramalkan. Sistem tertentu relatif/konstan dalam jangka waktu yang lama. Sistem Tak Tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi, karena mengandung unsur probabilitas.

4. Sistem diklasifikasikan sebagai Sistem Tertutup dan Sistem Terbuka

   Sistem Tertutup merupakan sistem yang tidak berhubungan dan tidak berpengaruh dengan lingkungan luarnnya. Sistem Terbuka adalah yang menerima masukan dan menghasilkan keluaran subsistem lainnya.

2.1.4. Pengertian Sistem Informasi

Menurut Pratiwi dan Asti Herliana (2015:224) menyimpulkan bahwa "sistem informasi merupakan suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu oganisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan".

Menurut Kadir (2014:8) "sistem informasi mencakup sejumlah komponen (manusia, komputer, teknologi informasi, dan prosedur kerja), ada sesuatu yang diproses (data menjadi informasi), dan dimaksudkan untuk mencapai suatu sasaran atau tujuan".

Menurut Kadir (2014:71), sistem informasi mengandung komponen-komponen sebagai berikut:

1. Perangkat keras (*hardware*), yang mencakup piranti-piranti fisik seperti komputer dan printer.
2. Perangkat lunak (*software*) atau program, yaitu sekumpulan intruksi yang memungkinkan perangkat keras memproses data.
3. Prosedur, yaitu sekumpulan aturan yang dipakai untuk mewujudkan pemrosesan data dan pembangkitan keluaran yang dikehendaki.

4. Orang, yakni semua pihak yang bertanggung jawab dalam pengembangan sistem informasi, pemrosesan, dan keluaran sistem informasi.
5. Basis data (*database*), yaitu kumpulan tabel, hubungan, dan lain-lain yang berkaitan dengan penyimpanan data.
6. Jaringan komputer dan komunikasi data, yaitu sistem penghubung yang memungkinkan sumber (*resources*) dipakai secara bersama atau diakses oleh sejumlah pemakai.

2.1.5. Persediaan

Menurut Fauziah dan Ratnawati (2018:99) "Persediaan merupakan salah satu asset perusahaan. Peranan pengendalian intern dalam hal ini sangatlah penting dalam meningkatkan keamanan persediaan sebagai harta perusahaan".

Menurut Ristono (2013:1) mengemukakan bahwa "Persediaan dapat diartikan sebagai barang yang di simpan untuk digunakan atau dijual pada masa atau periode yang akan datang". Oleh karena itu persediaan wajib untuk dikelola dengan baik demi kelancaran operasi-operasi yang ada dalam klinik.

Adapun tujuan pengelolaan pesediaan adalah sebagai berikut:

1. Untuk dapat memenuhi kebutuhan atau permintaan konsumen dengan cepat (memuaskan konsumen).
2. Untuk menjaga kontinuitas atau menjaga agar perusahaan tidak mengalami kehabisan persediaan yang mengakibatkan terhentinya proses, hal ini dikarenakan alasan:
    a. Kmungkinan baarang menjadi langka sehingga sulit untuk diperoleh.
    b. Kemungkinan supplier terlambat mengirimkan barang yang dipesan

3. Untuk memepertahankan dan bila mungkin meningkatkan penjualan dan laba perusahaan.
4. Menjaga agar pembelian secara kecil-kecilan dapat dihindari, karena dapat mengakibatkan ongkos pesan menjadi besar.
5. Menjaga supaya penyimpanan dalam emplacement tidak besar-besaran, karena akan mengakibatkan biaya menjadi besar.

2.1.6. Pengertian Program

Menurut Kadir (2014:192) “Program sekumpulan intruksi yang digunakan untuk mengatur perangkat keras komputer agar melaksanakan tindakan tertentu”.

Sedangkan menurut Fadallah dan Rosyida (2018:61) “Program adalah kumpulan intruksi yang digunakan untuk mengatur komputer agar melakukan suatu tindakan tertentu.”

2.1.7. Pemograman Berorientasi Objek

Menurut Fadallah dan Rosyida (2018:61) “Pemogramman berorientasi objek adalah suatu cara baru dalam berfikir serta berlogika untuk menghadapi masalah-masalah yang akan dicoba atasi dengan buatan komputer”.

Menurut Fadallah dan Rosyida (2018:61-62) Objek Oriented Programming memiliki ciri sebagai berikut:

1. Objek

   Bentuk baik yang nyata atau tidak, seperti manusia, hewan, benda, konsep, aliran dan lain-lain. Objek merupakan inisiasi (turunan langsung) dari suatu kelas.

2. Kelas

   Kumpulan objek yang memiliki kemiripan perilaku (*method*), ciri atau karakteristik (*property*).

3. Method

   Perilaku dari objek atau kelas tertentu. Merupakan perwujudan aksi atau tindakan dari dunia nyata di dalam pemograman komputer.

4. Konstruktor

   Suatu fungsi yang dideklarasikan atau didefinisikan di dalam kelas, konstruktor harus mempunyai nama yang sama dengan fungsinya. Konstruktor dijalankan bersamaan dengan terciptanya kelas tersebut. Dalam suatu kelas biasanya terdapat lebih dari satu konstruktor. Konstruktor seperti *method* tetapi tidak mengembalikan nilai dan dapat didefinisikan tanpa paarameter atau memakainya.

5. De-konstuktor

   Fungsi yang dideklarasikan dalam kelas, nama sama dengan nama fungsinya. Tetapi dijalankan bersamaan dengan dimusnahkannya kelas tersebut.

6. Karakteristik / *Properties*

   Ciri yang dimiliki oleh suatu objek, karakteristik ini juga sebagai pembeda objek satu dengan objek lainnya dalamm kelas yang sama (konsep individu).

7. Variabel

   Tempat menampung data sementara, dalam pemograman objek biasanya disebut data, sedangkan dalam pemograman prosedural sering disebut dengan variabel.

8. Data

   Istilah lain dari variabel OOP. Dalam pemograman java biasa juga disebut field, data member atau *instance variable*.

9. Hak akses (*access attribute*)

   Hak akses digunakan untuk dapat menentukan data member mana yang dapat digunakan oleh kelas lain, dan mana yang tidak dapat digunakan.

2.1.8. *Visual Basic . NET*

Menurut Ruli (2017:11) *Microsoft Visual Basic . NET* adalah sebuah alat untuk mengembangkan dan membangun aplikasi yang bergerak di atas sistem *.NET Framework*, dengan mengguanakan bahasa *BASIC*. Dengan menggunakan alat ini, para programmer dapat memebangun aplikasi *Windows Form*, Aplikasi web berbasis *ASP.NET*, dan juga aplikasi *command line*. alat ini dapat diperoleh secara terpisah dari beberapa produk lainnya (seperi *Microsoft Visual C++*, *Visual C#*, atau *Visual J#*).

2.1.9. Basis Data

Basis data dijadikan sebagai media untuk menampung data-data yang dimasukkan ke dalam sistem informasi, kemudian diolah untuk mendapatkan keluaran yang berguna.

Menurut Sukamto dan M. Shalahuddin (2016:43) ”basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat unformasi tersedia saat dibutuhkan”.

Menurut Kadir (2014:218) “Basis data (*database*) adalah suatu pengorganisasian data yang saling terkait sehingga memudahkan aktivitas untuk memeperoleh informasi”.

Basis data dimaksudkan untuk mengatasi problem pada sistem yang memekai pendekatan berbasis berkas. Untuk mengelola basis data diperlukan perangkat lunak yang disebut Database Management System (DBMS). DBMS adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol, dan mengakses basis data dengan cara yang praktis dan efisien.

Pemanfaatan bentuk basis data dapat digunakan di berbagai aktivitas penggunanya, pemanfaatan basis data (Ruli, 2017:9) antara lain:

1. Kecepatan dan Kemudahan

   Memungkinkan untuk mudah dalam melakukan penyimpanan, perubahan data, dan pengambilan suatu data.

2. Efisiensi ruang penyimpanan

   Optimalisasi penggunaan ruang penyimpanan dapat dilakukan dengan menerapkan sejumlah pengkodean atau membuat keterhubungan antar kelompok data yang saling berhubungan.

3. Keakuratan

   Pemanfaatan pengkodean atau pembentukan keterhubungan antar kelompok data dengan menerapkan aturan atau batasan data.

4. Ketersediaan

   Perkembangan data sejalan dengan waktu membutuhkan ruang penyimpanan yang besar, sehingga data administator harus memilih mana yang merupakan

data utama, data master, data transaksi data histori sampai data yang sudah tidak digunakan lagi. Pemilihan ini untuk mempermudah dalam pencarian data dan mengefesiensi ruang penyimpanan.

5. Keamanan

   Untuk menentukan siapa saja yang boleh menggunakan aplikasi beserta objek-objek didalamnya dan jenis operasi yang digunakan.

6. Pemakaian bersama

   Masing-masing bagian dari suatu data dapat digunakan atau diakses bersama-sama dalm waktu yang bersamaan oleh pemakai untuk aplikasi yang berbeda.

2.1.10. Metode Pengembangan Perangkat Lunak

Metode pengembangan perangkat lunak dijadikan sebagai disiplin ilm u untuk mengembangkan/merancang sebuah sistem. Model air terjun (*waterfall*) merupakan salah satu model dari metode pengembangan perangkat lunak.

Menurut Sukamto dan M. Shalahuddin (2016:28) metode *waterfall* adalah "metode air terjun menyediakan pendekatan alur hidup perangkat lunak secara skuensial atau terurut dimulai dari analisa, desain, pengkodean, pengujian, dan pendukung (*support*)".

Maka dari itu dapat disimpulkan bahwa model *waterfall* merupakan model dari metode perangkat lunak yang melakukan pendekatan sistematis dan sekuensial mulai dari tahapan analisis, desain, pengkodean, pengujian dan pendukung.

Adapun penjelasan tahapan model *waterfall* (Sukamto dan M. Shalahuddin, 2016:28), yaitu:

1. Analisis Kebutuhan Perngkat Lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami prangkat lunak seperti apa yang dibutuhkan oleh user. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

2. Desain

   Desain tahap desain.

   perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antar muka dan prosedur pengkodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu untuk didokumentasikan.

3. Pembuatan Kode Program

   Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai desain yang telah dibuat pada tahap desain.

4. Pengujian

   Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*eror*) dan memastikan keluaran yang dihasilkansesuai dengan yang diinginkan.

5. Pendukung (*support*) atau Pemeliharaan (*maintenance*)

   Tahap pendukung atau pemeliharaan dapat mengulangi proses mulai dari tahap analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.2. Teori Pendukung *(Tool System)*

Analisa terstruktur merupakan suatu metode analisa dengan menggunakan peralatan pendukung atau sarana *(tool system)* yang mana sarana tersebut digunakan untuk membuat spesifikasi sistem yang terstruktur.

2.2.1. *Entity Relationship Diagram* (ERD)

Entity Relational Diagram merupakan pemodelan basis data dengan menggunakan diagram relasi antar entitas, dapat dilakukan dengan menggunakan suatu pemodelan basis data.

Menurut Sukamto dan M. Shallahudin (2016:53) "ERD adalah bentuk paling awal dalam melakukan perancangan basis data relasional. Jika menggunakan OODMBS maka perancangan ERD tidak perlu dilakukan". Sedangkan menurut Ladjamudin (2013:142) "ERD adalah suatu model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak". Adapun elemen-elemen Diagram Hubungan Entitas:

1. *Entity*

   Pada diagram E-R diagram, entity digambarkan dengan sebuah bentuk persegi panjang.

2. *Relationship*

   Pada E-R diagram, relationship dapat digambarkan dengan sebuah bentuk belah ketupat.

3. *Relationship Degree*

   Relationship degree atau Derajat *Relationship* adalah jumlah entitas yang berpartisipasi dalam satu relationship. Derajat *Relationship* yang sering dipakai di dalam ERD:

   a. *Urinary Relationship*

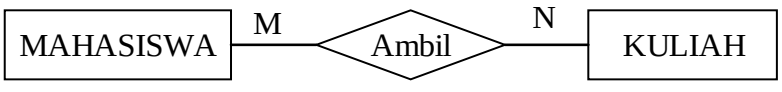
      Urinary Relationship adalah model *Relationship* yang terjadi diantara entity yang berasal dari entity set yang sama.

Sumber : Ladjamudin (2013:145)

Gambar II.1.
Diagram Relationship Unary

   b. *Binary Relationship*

      Binary Relationship adalah model *Relatonship* antara *instance-instance* dari suatu tipe entitas (dua *entity* yang berasal dari *entity* yang sama).

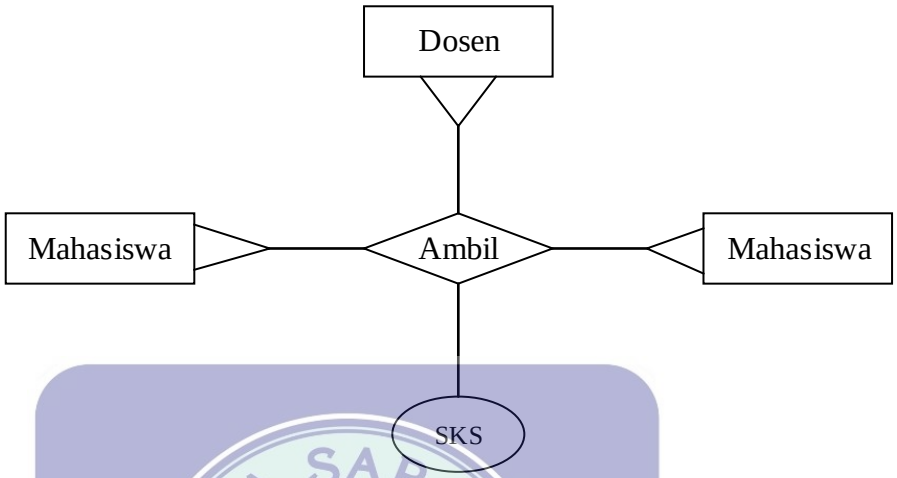


Sumber : Ladjamudin (2013:145)

Gambar II.2.
Diagram Relationship Binary

c. *Trenary Relationship*

Trenary Relationship merupakan relationship antara *instance-instance* dari tiga tipe entitas secara sepihak.



Sumber : Ladjamudin (2013:146)

Gambar II.3.
Diagram Relationship Ternary

2.2.2. *Logical Record Structure* (LRS)

Menurut Ladjamudin (2013:159) "*Logical record structure* (LRS) merupakan hasil transformasi ERD ke LRS yang memulai proses kardinalitas dan menghilangkan atribut-atribut yang saling berelasi".

Pentranformasikan ERD ke LRS ini memiliki aturan-aturan tertentu yang mempengaruhi langkah pentransformasianya itu kardinalitas. Adapaun kardinalitas tersebut (Ladjamudin, 2013:160) yaitu:

1. 1:1 (one to one)

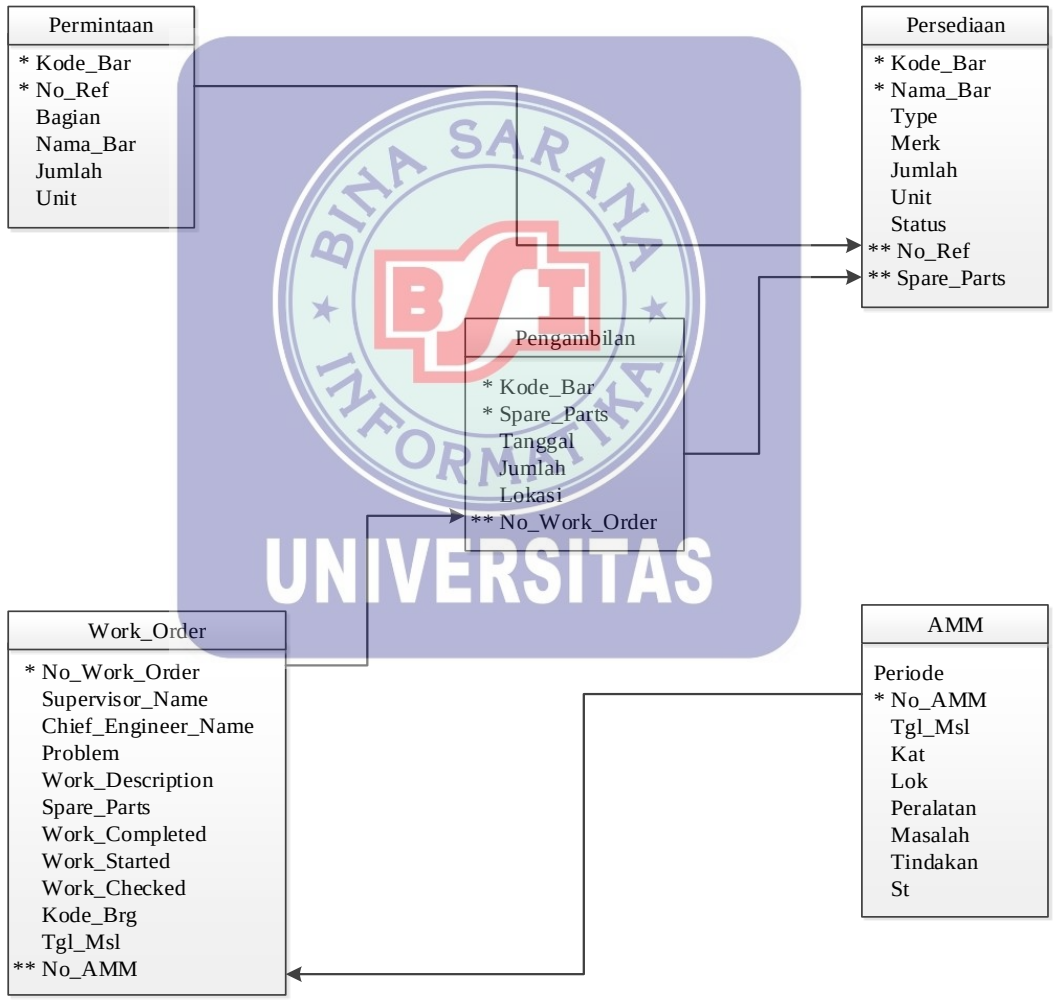
   Relasi yang terjadi antara satu entity dengan entity lainnya yang memiliki hubungan 1:1.

2. 1:M (one to many)

   Relasi yang terjadi antara satu entity dengan entity lainnya yang memiliki hubungan 1:M.

3. M:N (many to many)

   Relasi yang terjadiantara satu entity dengan entity lainnya yang memiliki hubungan M:N. Pada relasi ini biasa digunakan tabel bantuan untuk memecahkan relasi tersebut menjadi 1:1 atau 1:M.



Keterangan:
* : *Primary Key*
** : *Forgien Key*
Sumber : Ladjamudin (2013:202)

Gambar II.4.
***Transformasi Logical Record Structure* (LRS)**

2.2.3. *Unified Modeling Language* (UML)

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang di berbagai negaara dapat mengerti pemodelan perangkat lunak.

Menurut Sukamto dan M. Shalahuddin (2016:133) "UML (*Unified Modeling Language*) adalah salah standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, mambuat analisis & desain, serta manggambarkan arsitektur dalam pemograman berorientasi objek".

1. *Activity Diagram*

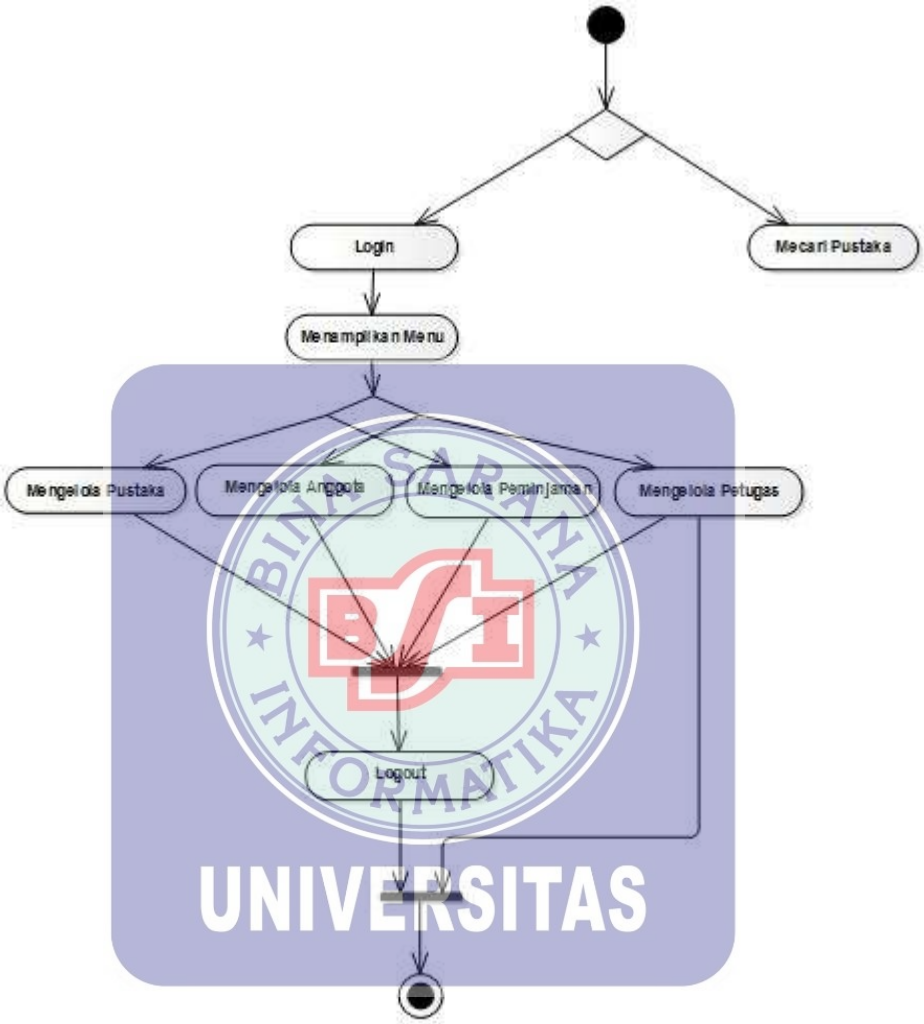
   Sukamto dan M. Shalahuddin (2016:161), Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor*, jadi activitas yang dapat dilakukan oleh sistem.

   Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

   a. Ranacangan proses bisnis dimana setuap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
   b. Urutan atau pengelompokan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

c. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

d. Rancangan menu yang ditampilkan pada perangkat lunak.



Sumber : Sukamto dan M. Shalahuddin (2016:234)

Gambar II.5.
Diagram Interaksi Studi Kasus

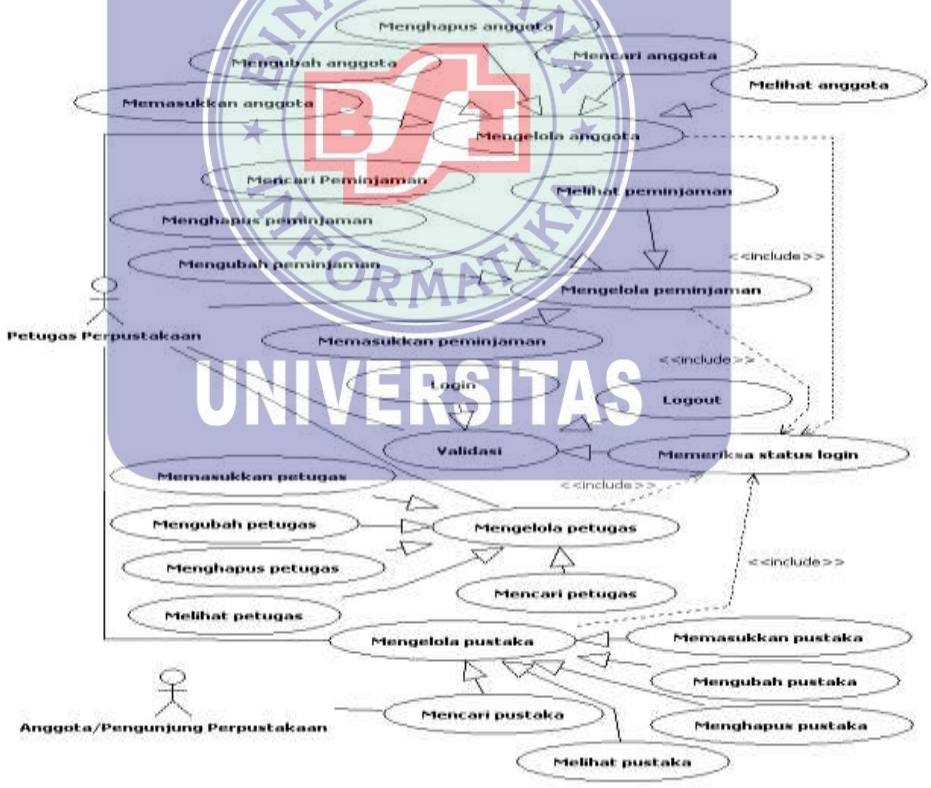
2. *Use Case Diagram*

Sukamto dan M. Shalahuddin (2016:155), Diagaram use case atau use case diagrammerupakan pemodelan untuk kelakuan (*behivor*) sistem informasi yang akan dibuat. *Use case* mendeskrifsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case*

digunakan untuk mengetahui fungsi apa saja yang ada dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada use case yaitu pendefinisian apa yang disebut *aktor* atau *use case*.

a. *Aktor* merupakan orang, proses atau sistem lain yang berinterkasi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
b. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau *aktor*.



Sumber : Sukamto dan M. Shalahuddin (2016:204)

Gambar II.6.
Diagram *Use Case* Perpustakaan

3. *Class Diagram*

Sukamto dan M. Shalahuddin (2016:141), 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 oprasi.

a. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.

b. Operasi atau metode adalah fungsi fungsi yang dimiliki oleh suatu kelas.

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan didalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Banyak berbagai kasus, perancangan kelas yang dibuat tidak sesuai dengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancang dan hasul jadinya tidak sesuai. Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga pembuat perangkat lunak atau *programmer* dapat membuat kelas-kelas didalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1) Kelas main

Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

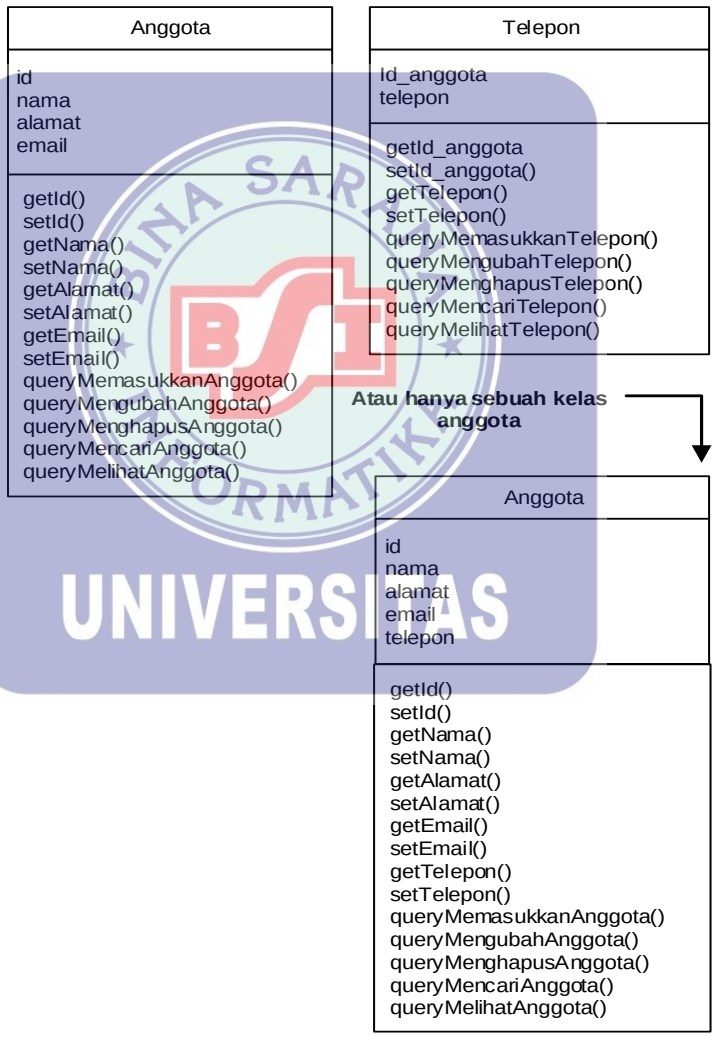
2) Kelas yang menangani tampilan sistem (*view*)

Kelas yang mendefinisikan dan mengatur tampilan ke pemakai

3) Kelas yang diambil dari pendefinisian *use case* (*controller*)

Kelas yang menangani fungsi-fungsi yang haru ada diambil dari pendefinisian *use case*, kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.

4) Kelas yang diambil dari pendefinisian data (*model*)
5) Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil namun akan disimpan ke basis data.

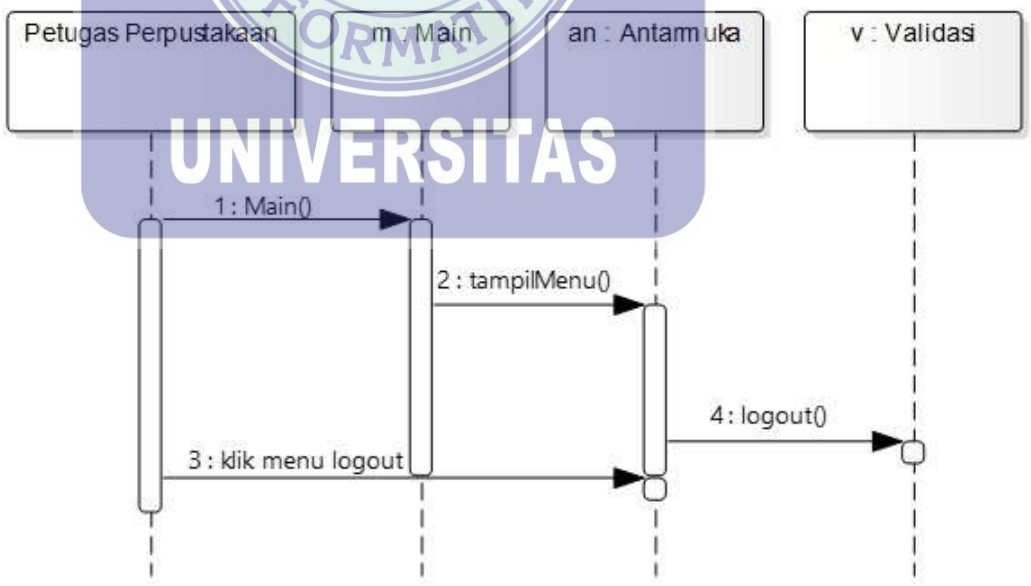


Sumber : Sukamto dan M. Shalahuddin (2016:143)

Gambar II.7.
Perancangan Kelas Data Untuk Tabel dari Atribut *Multivalue*

4. *Sequence diagram*

Sukamto dan M. Shalahuddin (2016:165), *Sequence diagram* 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 terlihat dalam sebuah *use case* beserta meode-metode yang dimiliki kelas yang diinstanisasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat sekenario yang ada pada *use case*. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian use case yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.



Sumber : Sukamto dan M. Shalahuddin (2016:210)

Gambar II.8.
Diagram Sekuen untuk *Use Case* Logout