BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Beberapa penelitian tentang Stok Opname Obat dengan hasil yang relevan antara lain : Purbasari [3], melakukan sebuah penelitian pada Apotek Merben, dimana Apotek Merben memiliki sebuah masalah pada proses penjualan dan pengelolaan data obat yang masih menggunakan sistem manual. Sistem manual yang dimaksud adalah dengan menggunakan Alat Tulis Kantor (ATK) dalam penjualan dan pengelolaan data obatnya. Sehingga dibutuhkan suatu sistem yang dapat membantu sistem penjualan dan pengelolaan data obat. Sistem ini berbasis *localhost*. Manfaat yang didapat dari sistem informasi ini yaitu mampu menghasilkan informasi yang cepat, dan tepat waktu sehingga memudahkan dalam pengolahan data penjualan obat dan data stok obatnya.

Penelitian lain yang dilakukan oleh Saiddinur [4], pada Apotek Santoso yang memiliki permasalahan berupa proses mengelola persediaan obat. Masalah lain yang dihadapai berupa kehabisan stok obat dalam setiap periode bulannya, dikarenakan banyaknya permintaan dari konsumen, serta sering terjadinya persediaan yang menumpuk dan berlebihan yang dapat menyebabkan terjadinya kadaluarsa pada obat yang ada. Sistem ini dikembangkan menggunakan metode haan. *Supply Chain Management* (SCM) dan dibantu dengan metode *Lot Sizing* khususnya *Economic Order Quantity* (EOQ) dan didukung *Reorder Point* (ROP) untuk menentukan berapa obat yang harus dipesan dan kapan harus melakukan pemesanan. Manfaat yang didapat dari sistem informasi ini ialah dapat mengendalikan stok dan menghindari penumpukan obat.

Penelitian selanjutnya juga pernah dilakukan oleh Nurdesni [5], pada Apotek Adir Farma yang memiliki permasalahan pada proses pencatatan pada gudang mengumpulkan bon penerimaan dan bon pengeluaran obat yang dilakukan secara manual sehingga sering terjadinya keterlambatan pembuatan laporan persediaan obat yang disebabkan oleh penumpukan data transaksi barang masuk dan data barang keluar. Sistem ini dikembangkan menggunakan metedologi *Model-Driven Development* (MDD) yang memanfaatkan alat bantu *seperti Use Case Diagram, Activity Diagram, Class Diagram*, dan *Sequence Diagram*. Manfaat yang didapat dari sistem informasi

persediaan obat ini yaitu mampu mengatasi permasalahan yang sering timbul dan membantu kegiatan pencatatan data. Dari penelitian yang akan dilakukan memiliki sebuah perbedaan dengan penelitan – penelitian sebelumnya. Perbedaan tersebut diantaranya yaitu Metode yang digunakan ialah metode FIFO *(First In First Out*) sehingga akan meminimalisir terjadinya stok obat yang *expired* date. Sistem ini berbasis web agar mempermudah karyawan apotek dalam pengelolaan Stok Opname Obat, Pengelolaan Obat Masuk dan Obat Keluar serta penyampaian laporan Stok Opname Obat, Obat Masuk dan Obat Keluar. Tahap pengembangan sistem informasi ini menggunakan metode SDLC (*System Development Life Cycle*) model *Waterfall* yang dibangun menggunakan bahasa pemrograman PHP (*Hypertext Preprocessor*) dan *MySQL* sebagai aplikasi *database*.

2.2 Landasan Teori

2.2.1 Sistem Informasi

Sistem Informasi adalah suatu 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. Dengan berkembangnya sistem informasi saat ini, banyak sistem informasi pada organisasi yang ingin mencapai tahap sistem informasi secara cepat, relevan dan akurat. Sistem Informasi merupakan cara yang terorganisir untuk mengumpulkan, memasukan, dan memroses data dan penyimpananya, mengelola, mengontrol dan melaporkanya sehingga dapat mendukung perusahaan atau organisasi untuk mencapai tujuan. Jadi dapat disimpulkan bahwa sistem informasi adalah sebuah rangkaian prosedur yang menggabungkan subsistem-subsistem yang memepertemukan kebutuhan organisasi dengan laporan yang diperlukan [6].

Adapun definisi sistem informasi menurut beberapa para ahli yaitu :

1. Menurut James A. Hall Bahwa sistem Informasi adalah serangkaian prosedur formal dimana data dikumpulkan, diproses menjadi informasi dan didistribusikan ke para pengguna [7].
2. Menurut Jogianto, Sistem Informasi merupakan suatu system di dalam suatu organisasi yang merupakan kombinasi dari orang-

orang, fasilitas, teknologi, media, prosedur-prosedur dan pengendalian [8].

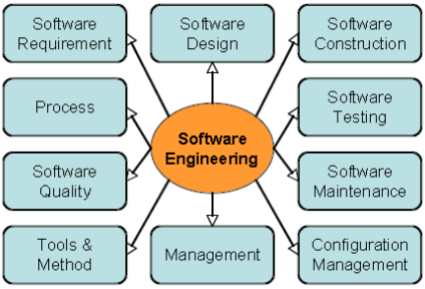
3. Menurut Indrajit, dalam buku Hanim Tohari tahun 2017 sistem informasi dapat dianalogikan sebagai sebuah permintaan (demand) dari masyarakat industri, ketika kebutuhan akan sarana pengelolahan data dan komunikasi yang cepat dan murah [9].

2.2.2 Rekayasa Perangkat Lunak

Perangkat lunak(*Software*) adalah sebuah perintah program dalam sebuah computer, yang apabila di ekseusi oleh user akan memberikan fungsi dan untuk bekerja seperti yang diharapkan oleh user. Rekayasa Perngkat Lunak (RPL) merupakan proses kegiatan perangkat lunak guna mengembangkan, memelihara, dan membangun kembali dengan menggunakan prinsip rekayasa untuk menghasilkan perangkat lunak yang dapat bekerja lebih efektif dan efisien untuk user. Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut [10] :

1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (*maintainability*)
2. Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi (*dependability* dan *robust*)
3. Efisien dari segi sumber daya dan penggunaan
4. Kemapuan untuk dipakai sesuai dengan kebutuhan (*usability*)

Ruang lingkup dari RPL dapat digambarkan sebagai berikut :



Gambar 2.1 Ruang lingkup RPL

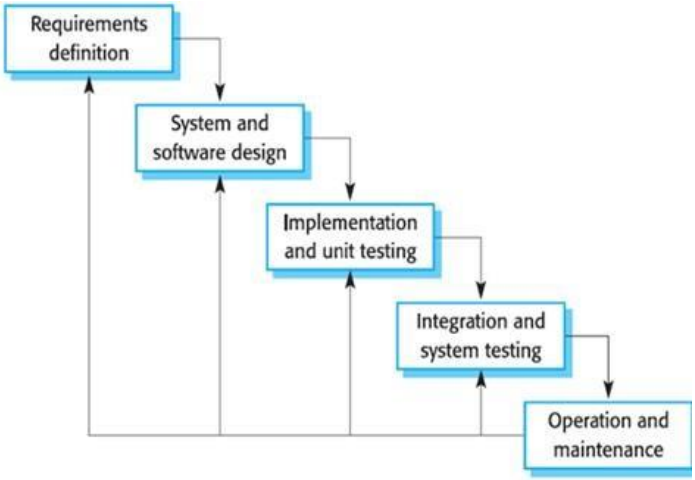
1. *Software Requirment* merupakan kegiatan yang dilakukan untuk mengidentifikasi dan menganalisis kebutuhan perangkat lunak. Hasil akhir tahapan ini adalah spesifikasi dan model perangkat lunak.
2. *Software Design* adalah tahapan perancangan arsitektur, komponen, antar muka, dan karakteristik lain dari perangkat lunak.
3. *Software Construction* berhubungan dengan detail pengembangan perangkat lunak, termasuk algoritma, pengkodean, pengujian dan pencarian kesalahan.
4. *Software Testing* meliputi pengujian pada keseluruhan perilaku perangkat lunak.
5. *Software Maintenance* mencakup upaya-upaya perawatan ketika perangkat lunak telah dioperasikan.
6. *Software Configuration Management* berhubungan dengan usaha perubahan konfigurasi perangkat lunak untuk memenuhikebutuhan tertentu.
7. *Software Engineering Management* berkaitan dengan pengelolaan dan pengukuran RPL, termasuk perencanaan proyek perangkat lunak.
8. *Software Engineering Tools and Methods* mencakup kajian teoritis tentang alat bantu dan metode RPL.
9. *Software Engineering Process* berhubungan dengan definisi, implementasi pengukuran, pengelolaan, perubahan dan perbaikan proses RPL.
10. *Software Quality* menitik beratkan pada kualitas dan daur hidup perangkat lunak.

2.2.2.1 Metode Pengembangan Sistem

Tahap pengembangan sistem dalam penelitian ini menggunakan meggunakan SDLC (*System Development Life Cycle*). SDLC *(System Devlopment Life Cycle)* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan untuk mengembangkan sistem perangkat lunak sebelumnya berdasarkan *best paratice* atau cara-cara yang sudah disebut baik. SDLC yang digunakan dalam pengembangan sistem menggunakan SDLC model *Waterfall* menurut Ian Sommerville [11].

Model *waterfall adalah sebuah model pengembangan sistem* dengan model pendekatan sekuensi linear atau alur hidup klasik.

Pengembangan sistem dikerjakan secara sistematis dan juga berurutan, berikut adalah tahap-tahap dalam model *waterfall* [12] :



Gambar 2.2 Model *Waterfall* menurut Ian Sommerville

1. *Requirements Definition*
   Tahap ini merupakan tahap pengumpulan data dengan melakukan penelitian, wawancara atau studi literature. Hasil dari analis yaitu adanya sebuah dokumen dengan user requirement atau bisa dikatakan data yang berhubungan dengan keinginan user dalam pembuatan sistem.
2. *System and Software Design*
   Tahap ini merupakan tahapan yang akan menghasilkan dokumen perancangan dari sistem. Perancangan sistem ini berasal dari data yang telah diperoleh dari identifikasi kebutuhan sistem. Dari tahap ini akan menghasilkan dokumen software requirement specification. Dokumen ini akan di gunakan sebagai alur proses sistem yang akan dibuat.
3. *Implementation and Unit Testing*
   Tahap ini merupakan tahap mengubah hasil perancangan menjadi kode program sesuai dengan apa yang telah dirancang.

Semua algoritma diterapkan pada bagian ini untuk menyesuaikan dengan kebutuhan pengguna.

4. *Integeration and System Testing*
   Tahap merupakan proses pengujian atau testing untuk memastikan apakah sesuai dengan kebutuhan perangkat lunak atau tidak, dimana dalam tahap pengujian ini menggunakan metode *blackbox testing*.
5. *Operation and Maintenance*
   Satu tahap akhir dari penerapan dan pengembangan sebuah sistem yakni kegiatan yang dilakukan oleh user untuk mengoprasikan sistem yang sudah jadi menjadi terverifikasi. Seiring berjalannya waktu, software yang telah di bangun dan di terapkan kepada user pasti akan mengalami perubahan. Perubahan terjadi karena adanya kesalahan, karena software harus menyesuaikan dengan lingkungan seperti peripheral atau sistem operasi baru atau karena membutuhkan perkembangan fungsional (belum dilakukan).

2.2.2.2 Metode Pengujian Sistem

Tahap pengujian sistem ini menggunakan metode *Black box testing*. *Black box testing* merupakan pengujian perangkat lunak yang berfokus pada detail kebutuhan perangkat lunak seperti tampilan perangkat lunak dan fungsionalitas perangkat lunak. Metode ini digunakan untuk mengetahui apakah perangkat lunak sudah berfungsi dengan benar. Pengujian black box testing bertujuan untuk menemukan kesalahan dalam kategori kesalahan antarmuka, kesalahan pada struktur data, kesalahan perfomansi, kesalahan inisialisasi dan terminasi [13].

2.2.3 Pemrograman Berorientasi Objek

Metodelogi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan informasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secar sistematis. Metodologi berorientasi objek sidasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metodologi berorientasi objek merupakan rangkaian aktivitas analisis orientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek [14].

Beberapa konsep dasar yang harus dipahami dalam pemrograman berorientasi objek berikut ini :

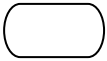
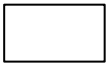
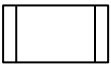
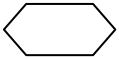
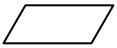
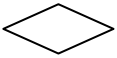
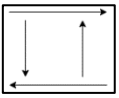
1. Kelas (*class*)
   Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungkin lahir atau diciptakan dari kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/*method*), hubungan (*relationship*), dan arti. Suatu kelas dapat diwariskan ke kelas yang baru.
2. Objek (*object*)
   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan.
3. Metode (*method*)
   Operasi atau metode atau *method* pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri. Operasiatau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek.
4. Atribut (*attribute*)
   Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya. Atribut sebaiknya bersifat privat untuk menjaga konsep enkapsulasi.
5. Abstraksi (*abstraction*)
   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.
6. Enkapsulasi (*encapsulation*)
   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

7. Pewarisan (*inheritance*)
   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.
8. Antarmuka (*interface*)
   Antarmuka atau interface sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah *interface* dapat diimplementasikan oleh kelas lain. Sebuah kelas dapat mengimplementasikan lebih dari satu antarmuka dimana kelas ini akan mendeklarasikan metode pada antarmuka yang dibutuhkan oleh kelas itu sekaligus mendefinisikan isinya pada kode program kelas itu.
9. *Reusability*
   Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.
10. Generalisasi dan Spesifikasi
    Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus.
11. Komunikasi Antar Obyek
    Komunikasi antar-objek dilakukan lewat pesan (*message*) yang dikirim dan satu objek ke objek lainnya.
12. Polimorpisme (*polymorphism*) Kemampuan suatu objek untuk digunakan di banyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.
13. *Package*
    Package adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam *package* yang berbeda.

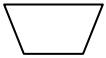
2.2.4 *Flowchart*

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program. Flowchart adalah bentuk gambar/diagram yang mempunyai aliran satu atau dua arah secara sekuensial. Berikut Tabel 2.1 merupakan symbol-simbol yang terdapat pada *flowchart* :

Tabel 2.1 Simbol-simbol *Flowchart* [15]

No.	Simbol	Nama	Keterangan
1.		*Terminal*	Memulai dan mengakhiri suatu program
2.		*Proses*	Proses perhitungan atau proses pengolahan data
3.		*Predefined Process* (sub program)	Permulaan sub program atau proses pengolahan data
4.		*Preparation*	Proses inisialisasi atau pemberian harga awal
5.		*Input – Output*	Memasukan data maupun menunjukan hasil dari suatu *process* tanpa tergantung dengan jenis peralatannya
6		*Decision*	Memilih proses berdasarkan kondisi yang ada.
7.		*Flow*	Menghubungkan antara simbol satu dengan simbol yang lain atau menyatakan jalannya arus dalam suatu proses. Simbol arus ini sering disebut juga dengan *connecting line*.

Tabel 2.2 Simbol-simbol *Flowchart* [15]

No	Simbol	Nama	Keterangan
1.		*Document*	Merupakan simbol untuk data yang terbentuk informasi.
2.		*On Page Connector*	Penghubung bagian-bagian *flowchart* yang berada pada satu halaman
3.		*Off Page Connector*	Penghubung bagian-bagian *flowchart* yang berada pada halaman berbeda
4.		*Manual Operation*	Menunjukkan pengolahan yang tidak dilakukan oleh komputer/*pc*.
5.		*Manual Input*	Memasukan data secara manual *on-line keyboard.*

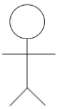
2.2.5 UML (Unified Modeling Language)

UML (*Unified Modeling Language*) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggambarkan diagram dan teks-teks pendukung [17]. UML memiliki banyak diagram, berikut penjelasan mengenai diagram yang ada di UML.

a. Use Case Diagram

Use Case Diagram merupakan pemodelan yang memiliki kemampuan dalam menggambarkan interaksi diantara aktor dan sistem. Use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu [18]. Berikut Tabel 2.2 adalah simbol-simbol yang ada pada diagram *use case diagram* :

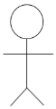
Tabel 2.3 Simbol-simbol *Use Case Diagram* [16]

No.	Simbol	Nama	Keterangan
1.		*Use case*	Deskripsi dari urutan aksi – aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*.
2.		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
3.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
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.		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas

b. Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang dikirim dan diterima oleh objek. Banyaknya diagram sekuen yang akan dibangun sesuai dengan pendefenisian usecase yang memiliki proses sendiri [19]. Berikut Tabel 2.3 adalah simbol-simbol yang ada pada *Sequence Diagram* :

Tabel 2.4 Simbol-simbol *Sequence Diagram* [16]

No.	Simbol	Nama	Keterangan
1.		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2.		*Actor*	Menggambarkan *user* atau pengguna.
3.		*Message*	Spesifikasi dari komunikasi antar *objek* yang memuat informasi – informasi tentang aktivitas yang terjadi.
4.		*Boundary*	Menggambarkan sebuah *form*.
5.		*Control Class*	Menghubungkan *boundary* dengan Tabel.
6.		*Entity Class*	Menggambarkan hubungan kegiatan yang akan dilakukan.

2.2.6 Basis Data (*Database*)

Basis data atau *database* adalah sekumpulan relasi data logika, dan deskripsi dari data yang dirancang untuk memenuhi kebutuhan informasi organisasi. Basisi data atau *databas*e adalah kumpulan data yang mewakili berbagai macam entitas dan hubungannya yang dapat digunakan secara bersamaan oleh banyak pengguna dan dirancang untuk memenuhi kebutuhan informasi organisasi [21]. Basis data memungkinkan tempat penyimpanan datayangbesar dan dapat digunakan secara bersamaan oleh banyak departemen dan pengguna. Database mewakili entitas, atribut, dan hubungan logis antara entitas. Basis data terdiri dari kumpulan data yang terorganisir, relasi antar data, dan objektifnya. Objektif utama adalah kecepatan dan kemudahan berinteraksi dengan data yang dikelola atau diolah. Selain itu terdapat pengertian bahwa basis data adalah sekumpulan data persisten yang digunakan oleh aplikasi sistem dari perusahaan.

a. DBMS (*Database Management* System)

DBMS *(Data Base Management System)* adalah koleksi terpadu dari aplikasi program *(system software)* yang dapat digunakan dalam melakukan definisi membuat melakukan akses dan memelihara *database* [22]. DBMS menyediakan berbagai fasilitas yaitu [22] :

1. DDL *(Data Definition Language)* merupakan perintah-perintah yang biasa digunakan administrator database untuk mendefinisikan skema dan subskema database. Perintah yang termasuk di dalamnya yaitu :
   a. *CREATE* : Digunakan untuk membuat, termasuk diantaranya membuat database dan tabel baru.
   b. *ALTER* : Digunakan untuk mengubah struktur tabel yang telah dibuat
   c. *DROP* : Digunakan untuk menghapus *database* dan tabel
2. DML *(Data Manipulation Language)* merupakan merupakan perintah-perintah yang memungkinkan pengguna melakukan akses dan manipulasi data sebagaimana yang telah diorganisasikan sebelumnya dalam model data yang tepat, *(Data Manipulation Language)* digunakan untuk memanipulasi *database* yang telah didefinisikan dengan DDL. Perintah yang termasuk DML :

a. *INSERT* : Digunakan untuk menyisipkan atau memasukan dalam tabel
b. *SELECT* : Untuk mengambil data atau menampilkan data dari suatu tabel atau beberapa tabel
c. *UPDATE* : Digunakan untuk memperbaharui data lama menjadi data terkini
d. *DELETE* : Digunakan untuk menghapus data dari tabel.

b. SQL (*Structured Query Language*)

SQL *(Stuctured Query Language)* adalah bahasa yang digunakan untuk mengelola data pada RDBMS, SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus [24]. MySQL merupakan basis data yang dikembangkan dari bahasa SQL *(structure query language).* MySQL dapat dikatakan sebagai *relational database management system* (RDBMS), yaitu hubungan antar table yang berisi data data pada suatu database dengan demikian dapat.

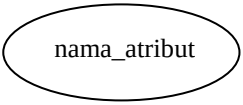
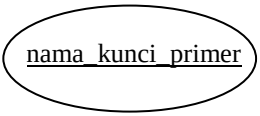
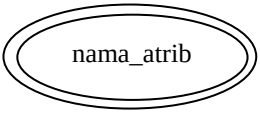
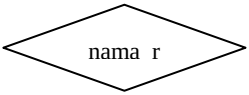
c. ERD (*Entity Relationship Diagram*)

ERD (*Entity Relationship Diagram*) merupakan tools yang digunakan untuk memodelkan struktur data dengan menggambarkan entitas dan hubungan antar enititas (*relationship*) secara abstrak (*konseptual*). ERD berfungsi untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan symbol [25]. Berikut Tabel 2.5 adalah daftar symbol dalam ERD :

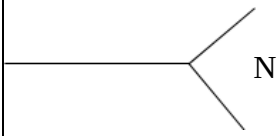
Tabel 2.7 Simbol-simbol ERD [16]

No	Simbo	Keterangan
1.	Entitas / *entity* nama_entitas	Entitas merupakan data inti yang akan disimpan; bakal tabel pada basis data; benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer; penamaan entitas biasanay lebih ke kata benda dan belum merupakan nama tabel.

Tabel 2.8 Simbol-simbol ERD [16]

No	Simbol	Keterangan
2.	Atribut 	Filed atau kolom data yang butuh disimpan dalam suatu entitas.
1.	Atribut kunci primer 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* 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).
2.	Atribut multi nilai / multivalue 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih.
3.	Relasi 	Relasi yang menghubungkan antar entitas; biasanya diawali dengan kata kerja

Tabel 2.9 Simbol - Simbol ERD

No	Simbol	Keterangan
1.	Asosiasi / *Association* 	Penghubung antara relasi dan entitas dimana 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* yang menghubungkan entitas A dan entitas B.

2.2.7 Apotek

Berdasarkan Surat Keputusan Kementrian Kesehatan Republik Indonesai Nomor 1027/MENKES/SK/IX/2004 Tahun 2004, Apotek merupakan tempat tertentu, tempat dilaksanakannya sebuah pekerjaan farmasi serta penyaluran sebuah persediaan farmasi, dan juga penyaluran pembekalan kesehatan lainya kepada sosial masyarakat [1].

2.2.8 Logistik

Logistik adalah proses perencanaan, implementasi dan kontrol serta penentuan kebutuhan pengadaan, penyimpanan, dan pemeliharaan serta penghapusan material. Manajemen logistik di apotek merupakan salah satu aspek penting, kesediaan obat saat ini menjadi salah satu tuntutan kesehatan. Analisis Pengelolaan Logistik obat di Apotek yang meliputi tahap perencanaan, pengadaan, penyimpanan, penghapusan, evaluasi dan monitoring yang saling terkait satu sama lain, sehingga harus terkoordinasi dengan baik agar dapat berfungsi secara optimal [26].

2.2.9 FIFO *(First In First Out)*

Metode FIFO merupakan metode dimana barang pertama yang masuk berarti barang tersebutlah yang pertama keluar. Metode FIFO atau masuk pertama keluar pertama mengasumsikan bahwa barang yang dibeli awal dianggap akan lebih awal dijual atau digunakan, dan harga pokok perolehan barang yang dibeli lebih awal akan dibebankan lebih dahulu sebagai harga pokok penjualan. Pada pencatatan secara fisik, metode ini beranggapan bahwa barang yang ada paling awal dianggap dijual paling awal [27].

~Halaman Ini Sengaja Dikosongi~