

## **BAB II**

### **TINJAUAN PUSTAKA DAN LANDASAN TEORI**

#### **2.1 Tinjauan Pustaka**

Penelitian terdahulu oleh Khairul Rizal (2018) yang mengusulkan perancangan Sistem Informasi Penjualan Obat di Apotek Zam-zam berbasis desktop dengan menggunakan model *Waterfall*. Permasalahan yang terjadi pada Apotek Zam-zam yaitu proses pembayaran dan pengecekan resep yang masih manual mengakibatkan lambatnya proses transaksi dan penyampaian laporan. Untuk meminimalisir kekurangan pada sistem yang masih manual, maka dibangunlah Sistem Informasi Penjualan Obat di Apotek Zam-zam. Sistem informasi yang dibuat akan mempermudah Petugas dan Pemilik Apotek dalam mengelola data transaksi dan data laporan [6].

Penelitian lain oleh Prasetya Adi Nugraha (2019), mengusulkan perancangan sistem Informasi penjualan di Apotek Kayba untuk menggantikan sistem di Apotek Kayba yang masih manual. Adapun cara-cara manual tersebut pertama, pihak apotek akan membuat surat pemesanan dan dikirim ke *supplier*. Kedua, *supplier* menyediakan obat-obatan yang dikehendaki sesuai surat pesanan dengan menyertakan faktur untuk dikirim ke apotek. Ketiga, penjualan obat akan ditulis di sebuah buku penjualan dengan menuliskan nama obat, jumlah dan harga. Sistem ini berbasis website dengan menggunakan bahasa pemrograman PHP dan MySQL sebagai *database*, kemudian dalam pembuatan sistemnya menggunakan Metode Prototype [7].

Penelitian yang dilakukan oleh Nurwahyunita, Iin Kurniawati Dewi, dan Syaiful Zuhri Harahap (2019), mengusulkan perancangan Sistem Informasi Penjualan Obat pada Toko Obat Anugerah Rantauprapat berbasis website yang bertujuan membangun sistem informasi menggunakan perancangan UML (*Unified Modelling Language*) dengan bahasa pemrograman PHP dan *database* MySQL. Dengan dibuatnya sistem ini diharapkan dapat meningkatkan kinerja dalam proses *input* dan pencarian data penjualan serta pemasaran pada Toko Obat Anugerah Rantauprapat [8].

Penelitian lainnya yaitu oleh Suci Anggareni (2020), yang mengusulkan perancangan sistem informasi penjualan obat-obatan pada Apotek Najwa Farma berbasis website. Permasalahan dalam studi kasus

tersebut yaitu Pengolahan data penjualan pada Apotek Najwa Farma ini masih dilakukan dengan cara sederhana yaitu pencatatan pada buku dan komputer untuk mengetahui informasi barang yang dijual dan yang terjual. Pembuatan sistem menggunakan metode *Research And Development* (R&D), bahasa pemrograman PHP dan MySQL sebagai *database* [9].

Penelitian yang dilakukan oleh penulis berbeda dengan penelitian sebelumnya, yaitu sistem yang dibuat untuk pengelolaan data obat, pengelolaan transaksi pembelian, pengelolaan transaksi penjualan, dan pengelolaan laporan. Sistem ini dilengkapi dengan *barcode scanner* untuk melakukan transaksi penjualan. Sistem ini dibuat dengan harapan dapat membantu apoteker dan karyawan dalam proses pengolahan data obat serta bisa mempermudah proses transaksi terutama transaksi penjualan.

## 2.2 Landasan Teori

Dalam menunjang penelitian ini, maka diperlukannya teori-teori mendasar. Teori-teori yang digunakan dalam penelitian ini adalah:

### 2.2.1 Sistem Informasi

Sistem adalah suatu jaringan dari prosedur yang saling berhubungan yang bekerja sama untuk melakukan aktivitas atau mencapai tujuan tertentu [2].

Berikut beberapa karakteristik yang menjadikan sebuah sistem berjalan dengan baik [10]:

1. Komponen sistem (*Component*)  
Suatu sistem yang memiliki komponen-komponen yang berinteraksi antar sistem lain untuk membentuk satu kesatuan dan saling bekerja sama..
2. Lingkungan Luar Sistem (*Environments*)  
Lingkungan luar sistem merupakan lingkungan luar sistem yang mempengaruhi operasi sistem yang dapat menguntungkan atau merugikan sistem itu sendiri.
3. Batasan Sistem (*Boundary*)  
Batasan sistem merupakan daerah luar sistem yang dibatasi oleh ruang lingkup (*scope*) antara satu sistem dengan sistem yang lain .
4. Penghubung sistem (*Interface*)  
Penghubung *interface* sistem merupakan media yang menghubungkan sistem dengan subsistem lainnya, dimana *output* dari subsistem akan menjadi *input* untuk subsistem lain.

5. **Masukkan Sistem (*Input*)**  
Masukkan sistem merupakan energi atau sumber daya yang dimasukkan ke dalam sistem agar sistem dapat beroperasi dan diproses sebagai *output*.
6. **Keluaran sistem (*Output*)**  
Keluaran sistem merupakan hasil energi setelah pemrosesan *inputan* keluaran.
7. **Pengolahan sistem**  
Bagian pengolahan yang dimiliki oleh suatu sistem atau sistem itu sendiri sebagai pengolahnya yang merubah *input* menjadi *output*.
8. **Sasaran sistem**  
Sasaran sistem merupakan sasaran (*objective*) atau tujuan (*goal*) yang menentukan masukkan yang dibutuhkan dan keluaran yang dihasilkan sistem.

Informasi merupakan hasil pengolahan data yang menggambarkan suatu kejadian nyata untuk pengambilan keputusan. Informasi dapat dihasilkan dari data yang baru masuk, data yang telah disimpan atau bisa didapat dari gabungan antara data yang disimpan dengan data yang baru masuk [11].

Sistem informasi merupakan suatu komponen-komponen yang dibuat oleh manusia dengan tujuan untuk menyajikan sebuah informasi. Sistem informasi juga dapat didefinisikan sebagai kumpulan prosedur organisasi yang memberikan informasi untuk mengendalikan organisasi, mempertemukan kebutuhan pengolahan transaksi [12].

### **2.2.2 Black Box Testing**

Metode *Black Box Testing* adalah sebuah metode pengujian software yang dilakukan dengan memasukkan data pada setiap *form* yang telah dibuat dengan tujuan mengetahui apakah program tersebut berjalan sesuai dengan yang dibutuhkan perusahaan [13].

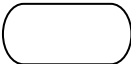


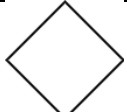

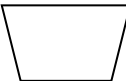
Pengujian *blackbox testing* memberi gambaran atas kondisi masukan dan pengujian fungsional program. Permasalahan yang diselesaikan dengan menggunakan metode *blackbox testing* adalah sebagai berikut:


- a. Kesalahan fungsi.
- b. Kesalahan antarmuka (*interface*).
- c. Kesalahan struktur data dan basis data.
- d. Fungsi yang salah atau hilang
- e. Kesalahan deklarasi dan terminasi [14].

### 2.2.3 Flowchart

*Flowchart* adalah bagan-bagan yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah. penggambaran secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program. *Flowchart* menolong analis dan programmer untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian [15]. Simbol *flowchart* dapat dilihat pada tabel 2.1.

**Tabel 2. 1** Simbol-simbol *flowchart*

No.	Simbol	Nama	Keterangan
1.		<i>Terminator</i>	Simbol yang berfungsi untuk mengawali dan mengakhiri suatu program.
2.		<i>Process</i>	Simbol yang menyatakan proses yang dilakukan komputer.
3.		Data	Simbol yang menggambarkan <i>input</i> (masukkan) dan <i>output</i> (keluaran) data di dalam suatu proses tanpa tergantung peralatan
4.		<i>Decision</i>	Simbol yang menunjukkan suatu kondisi tertentu yang akan menghasilkan pilihan ya atau tidak.
5.		<i>Document</i>	Simbol yang menunjukkan bahwa proses <i>input</i> berasal dari dokumen dan proses <i>output</i> yang perlu dicetak.
6.		<i>Manual Operation</i>	Simbol yang menyatakan suatu proses yang tidak dilakukan komputer.

No.	Simbol	Nama	Keterangan
7.		Simbol arus/ <i>flow</i>	Simbol yang menghubungkan simbol yang satu dengan simbol lain yang menyatakan jalannya arus suatu proses.


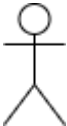
### 2.2.4 *Unified Modelling Language (UML)*



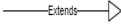
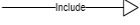
UML (*Unified Modeling Language*) adalah bahasa yang menjadi spesifikasi standar yang digunakan untuk mendokumentasi, menspesifikasi, dan membangun perangkat lunak [16]. Berikut beberapa jenis UML:

#### 1. *Use Case Diagram*

*Use case diagram* atau diagram *usecase* merupakan pemodelan untuk melakukan (*behaviour*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi aktor dengan sistem informasi yang akan dibuat. Secara kasar, *usecase* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu [9]. Simbol *Usecase* dapat dilihat pada tabel 2.2

**Tabel 2. 2** Simbol-Simbol *Use Case Diagram*








No.	Simbol	Nama	Keterangan
1.		<i>Use case</i>	Fungsi yang disediakan yaitu Abstraksi dan interaksi antara sistem dan aktor.
2.		<i>Actor</i>	Mewakili orang atau alat untuk berkomunikasi dengan <i>usecase</i> .

No.	Simbol	Nama	Keterangan
3.		<i>Association</i>	Sebagai penghubung atau komunikasi antara <i>aktor</i> dan <i>usecase</i> .
4.		<i>Generalization</i>	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah <i>usecase</i> di mana fungsi yang satu adalah lebih umum dari lainnya.
5.		<i>Extend</i>	Relasi <i>usecase</i> tambahan ke sebuah <i>usecase</i> dimana <i>usecase</i> yang ditambahkan dapat berdiri sendiri.
6.		<i>Include</i>	Relasi <i>usecase</i> tambahan ke sebuah <i>usecase</i> dimana <i>usecase</i> yang ditambahkan memerlukan <i>usecase</i> ini untuk menjalankan fungsinya.

## 2. *Sequence Diagram*

*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 menggambarkan diagram *sequence* maka harus diketahui objek-objek yang terlihat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansikan menjadi objek itu. Membuat diagram *sequence* juga dibutuhkan untuk melihat skenario yang ada pada *use case* [9]. Simbol-simbol *sequence diagram* bisa dilihat pada Tabel 2. 3.







**Tabel 2. 3** Simbol-Simbol *Sequence Diagram*

No.	Simbol	Nama	Keterangan
1.		Aktor	Menggambarkan <i>user</i> atau pengguna yang berinteraksi dengan sistem
2.		<i>Boundary object</i>	Simbol yang menggambarkan gambaran <i>form</i>
3.		<i>Entity object</i>	Berisi kumpulan kelas yang membentuk gambaran awal sistem
4.		<i>Control object</i>	Menggambarkan <i>Boundary</i> dengan tabel
5.		<i>Life line</i>	Digambarkan garis putus yang terhubung dengan objek yang menggambarkan tempat mulai dan berakhirnya sebuah pesan.
6.		<i>Activation bar</i>	Menyatakan objek dalam aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya. Menggambarkan waktu yang diperlukan untuk memproses pesan dari suatu objek.
7.		<i>Message</i>	Komunikasi antar objek untuk mengirimkan pesan antar class.

### 3. *Class Diagram*

*Class diagram* adalah deskripsi kelompok obyek-obyek dengan properti, perilaku dan relasi yang sama. Sehingga dengan adanya *class diagram* dapat memberikan pandangan global atas sebuah sistem. Hal tersebut tercermin dari *class-class* yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem biasanya mempunyai beberapa *class diagram* [17]. Dapat dilihat pada Tabel 2.4.

**Tabel 2. 4** Simbol-simbol *Class Diagram*



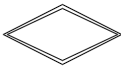


No	Simbol	Nama	Fungsi
1.		<i>Generalization</i>	Hubungan dimana objek anak ( <i>descendent</i> ) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk ( <i>ancestor</i> ).
2.		<i>Nary Association</i>	Upaya untuk menghindari <i>asosiasi</i> dengan lebih dari 2 objek.
3.		<i>Class</i>	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4.		<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu objek.
5.		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri ( <i>independent</i> ) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
6.		<i>Association</i>	Penghubung antara objek satu dengan objek yang lain.





### 2.2.5 Entity Relationship Diagram (ERD)

*Entity Relationship Diagram* (ERD) merupakan hubungan penterjemah yang berisi komponen-komponen himpunan entitas dan himpunan relasi yang dilengkapi dengan atribut-atribut dimana untuk menghubungkan *entity* tersebut digunakan key field (*primary key*) dari masing-masing *entity*. Pemodelan awal basis data yang paling banyak digunakan adalah menggunakan *Entity Relationship Diagram* (ERD). ERD dikembangkan berdasarkan teori himpunan dalam bidang matematika. ERD digunakan untuk pemodelan basisdata relasional. Sehingga jika penyimpanan basis data menggunakan OODBMS maka perancangan basis data tidak perlu menggunakan ERD [15].

**Tabel 2. 5** Simbol-simbol ERD

No	Simbol	Nama	Fungsi
1.		<i>Entity</i>	Suatu objek yang dikumpulkan dan diidentifikasi dalam lingkungan pemakai.
2.		<i>Relationship</i>	Relasi menunjukkan hubungan yang terjadi di antara sejumlah entitas yang berbeda.
3.		<i>Identify Relationship</i>	Suatu <i>relationship</i> dimana keberadaan <i>Child entity</i> bergantung pada induknya, dan PK <i>Child entity</i> memuat komponen PK <i>Parent entity</i> .
4.		<i>Attribute</i>	<i>Attribute</i> , karakteristik dari entitas atau relasi yang menjelaskan secara detail tentang entitas.
5.		<i>Key Attribute</i>	Atribut yang mengidentifikasi suatu <i>entity</i> dengan sangat spesifik atau unik.

No	Simbol	Nama	Fungsi
6.		<i>Multi Attribute</i>	Attribute yang mempunyai lebih dari satu (multivalue) nilai dari attribute yang bersangkutan.
7.		Alur	Hubungan antara relasi, entitas, dan atribut.

### 2.2.6 MySQL

MYSQL adalah sistem manajemen *database* yang menggunakan SQL (*Structured Query Language*) untuk mengolah *database*. MySQL bersifat rational, artinya manipulasi data akan lebih cepat karena data yang dikelola di dalam *database* diletakkan pada tabel yang terpisah. SQL memungkinkan penggunaanya mengetahui letak lokasi atau bagaimana informasi tersebut disusun. Bahasa SQL jauh lebih rumit dibandingkan dengan pengolah data, tetapi lebih mudah digunakan jika dibandingkan dengan bahasa pemrograman [18].

MySQL merupakan software RDMS (*Relational Database Management System*) yang sangat cepat dalam mengelola *database* dan penampungan datanya sangat besar. Selain itu, MySQL dapat diakses oleh banyak pengguna yang melakukan proses secara sinkron atau bersamaan [19].

### 2.2.7 C Sharp (C#)

C# atau C Sharp merupakan bahasa pemrograman berorientasi objek yang dikembangkan oleh Microsoft. Bahasa C# telah dibuat bebasiskan C++ yang dipengaruhi oleh bahasa pemrograman lain seperti Java, Delphi, Visual Basic dalam aspek-aspek atau fitur bahasa dengan beberapa penyederhanaan [20].

C# merupakan salah satu aplikasi yang memiliki kemampuan dalam penguatan Framework.NET. C# dibuat sejalan dengan perkembangan Framework. NET, C# sendiri dikembangkan oleh Microsoft. Dalam penerapannya C-Sharp (C#) menjanjikan produktifitas, fleksibilitas serta kemudahan yang ada dari aplikasi sebelumnya yaitu Visual Basic, Java dan C++. C# mengadopsi kemampuan dari peggabungan aplikasi sebelumnya [21].

Struktur dasar yang sering digunakan dalam penulisan program C# ada 5 yaitu[22] :

- a. *Resource and Library*  
Merupakan pendefinisian *library* yang digunakan untuk proses penulisan program selanjutnya atau *library* yang diimpor.
- b. *Namespace*  
Merupakan nama *project* yang akan dibuat.
- c. *Class*  
Berisi nama *Class* yang akan dibuat atau bisa ditambahkan penanda *Main Class* sebagai penanda bahwa *Class* tersebut adalah *Class* Utama.
- d. Deklarasi *Method*  
Untuk menjalankan *method* atau suatu perintah maka dilakukan pendeklarasian *method*. Suatu *method* akan dijalankan oleh *compailer* pertama kali, jika *method* tersebut didefinisikan dengan “*Main*”.
- e. *Method* atau *Command*  
Struktur terakhir dalam penulisan C# adalah *method* dan atau perintah yang akan dieksekusi *compailer*. *Command* atau catatan singkat program digunakan untuk mengetahui fungsi dari *Class* tersebut.

Kelebihan dari Bahasa pemrograman C# yaitu:

- a. C# bisa digunakan untuk membuat aplikasi desktop, aplikasi web, dan aplikasi berbasis *webservices* serta dapat mengembangkan aplikasi yang berjalan di Windows dengan sangat baik.
- b. Program C# bersifat *flexible* artinya, dapat dieksekusi pada komputer atau melalui web.
- c. *Powerful*: Kumpulan perintah pada C# sama seperti C++ yang memiliki banyak fitur.
- d. *Easiertouse*: perintah yang digunakan pada C# sama dengan perintah yang ada pada C++. Selain itu, C# memberi tahu letak *error* dalam proses membuat aplikasi.
- e. Memiliki keamanan (*secure*) untuk menghindari aksi kejahatan dari pihak lain atau *hacker* [22].

### 2.2.8 Usability

*Usability* adalah atribut kualitas yang mengukur seberapa mudah penggunaan suatu antarmuka (*interface*). “*Usability*” merujuk pada metode untuk meningkatkan kemudahan pemakaian selama proses perancangan. *Usability* memiliki lima aspek yaitu *learnability*, *efficiency*, *memorability*, *errors*, dan *satisfaction*. Pengukuran *usability* bergantung pada kemampuan pengguna menyelesaikan serangkaian tes [23]. Adapun nilai uji *usability* dapat dihitung menggunakan rumus:

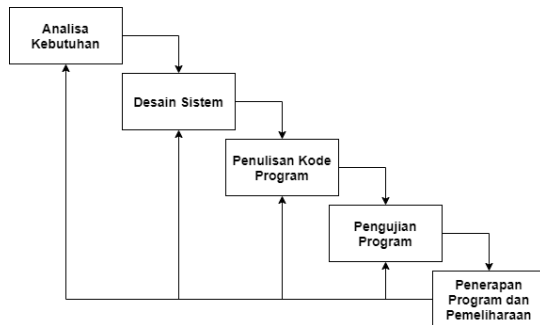
$$U = \frac{np}{nm} \times 100\%$$

Keterangan:

- U = Nilai hasil uji usability
- np = Nilai perolehan, jumlah seluruh hasil kuesioner/jawaban responden
- nm = Nilai maksimal, jumlah maksimal yang dapat diperoleh (jumlah responden x nilai maksimal skala *likert* yang dipakai x jumlah pertanyaan)

### 2.2.9 Metode Waterfall

Metode waterfall adalah proses evolusioner yang diikuti dalam menerapkan sistem atau subsistem informasi berbasis komputer. Metode waterfall terdiri dari serangkaian tugas yang erat, mengikuti langkah-langkah pendekatan sistem [24]. Metode waterfall sering disebut metode air terjun karena dalam proses pengerjaan sistem dibuat secara berurutan tahap demi tahap. Metode waterfall digunakan karena mempermudah dalam melakukan pengembangan sistem karena harus melalui tahapan-tahapan yang harus dilakukan [6]. Berikut adalah tahap metode pengembangan *waterfall* menurut Bassil:



**Gambar 2. 1** Metode *Waterfall* Menurut Bassil

1. **Analisa Kebutuhan**  
Merupakan proses pengumpulan kebutuhan sistem informasi. Untuk memahami dasar dari program yang akan dibuat, seorang analisis harus mengetahui ruang lingkup informasi, fungsi-fungsi yang dibutuhkan, kemampuan kinerja yang ingin dihasilkan dan perancangan antarmuka pemakai sistem informasi tersebut [6].
2. **Desain Sistem**  
Desain perangkat lunak adalah proses multi langkah yang focus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya [6].
3. **Penulisan Kode Program**  
Pengkodean sistem informasi merupakan proses penulisan bahasa program agar sistem informasi tersebut dapat dijalankan oleh mesin [6].
4. **Pengujian Program**  
Pengujian fokus pada perangkat lunak secara dari segi logik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (error) dan memastikan keluaran yang dihasilkan sesuai yang diinginkan [6].

#### 5. Penerapan Program dan Pemeliharaan

Tahapan terakhir yaitu Sistem yang dibuat akan diterapkan atau diimplementasikan kepada user. Setelah diimplementasikan masuk ke tahap pemeliharaan yang dilakukan untuk memperbaiki kesalahan pada sistem yang tidak ditemukan pada langkah sebelumnya.

Penerapan (implementasi) dilakukan setelah sistem lolos pada tahap pengujian. Aplikasi yang telah diimplementasi diharapkan dapat dipakai dan tidak berhenti di tengah jalan. Agar dapat dipergunakan dengan semestinya, perangkat lunak dipelihara dengan memperhatikan beberapa aspek, diantaranya:

- a. Menangani perkembangan data dengan seiring berjalannya waktu.
- b. Menangani ancaman dari program penyusup lainnya.
- c. Memperbaiki apabila ditemukan error atau bug pada aplikasi yang sedang dijalankan.
- d. Penambahan fitur seiring dengan berjalannya waktu.
- e. Mampu menangani perkembangan dan kemajuan teknologi[24].

#### **2.2.10 Apotek**

Apotek adalah pelayanan kesehatan yang menjual obat-obatan. Apotek rakyat adalah sarana kesehatan tempat yang menawarkan pelayanan kefarmasian dan penyerahan obat yang bertujuan meningkatkan dan memperluas akses masyarakat dalam memperoleh obat serta meningkatkan pelayanan kefarmasian [9].

#### **2.2.11 Obat**

Menurut PERMENKES No. 58 th. 2014 yang mengatur tentang Standar Pelayanan Farmasi Rumah Sakit, “obat adalah suatu bahan atau kombinasi bahan, termasuk produk biologi yang digunakan dalam rangka penetapan diagnosis, pencegahan, penyembuhan, pemulihan, peningkatan kesehatan dan kontrasepsi atau alat pencegah kehamilan pada manusia untuk mempengaruhi atau menyelidiki sistem fisiologis atau kondisi patologis”.

### **2.2.12 Pembelian**

Pembelian merupakan kegiatan utama yang dimulai dengan pemilihan sumber sampai memperoleh barang [15]. Urutan kegiatan pembelian Prosedur pembelian yang dilaksanakan oleh beberapa bagian dalam perusahaan yaitu prosedur pembelian. Bagian dalam perusahaan merupakan bagian yang terkait dalam prosedur ini. Pembelian dapat dilakukan dengan cara pembelian langsung dan pembelian tak langsung [4].

### **2.2.13 Penjualan**

Secara umum penjualan dapat diartikan sebagai kegiatan menjual atau pemindahan hak kepemilikan barang atau jasa dari penjual ke pembeli untuk mendapatkan keuntungan [25]. Setiap perusahaan ataupun pemilik bisnis melakukan penjualan dengan tujuan mendapat laba atau keuntungan. Laba yang diperoleh berasal dari pendapatan-pendapatan bisnis [26]. Penjualan adalah upaya untuk mengembangkan rencana strategis guna mendapatkan penjualan yang menghasilkan laba atau keuntungan [4].

### **2.2.14 Barcode**

*Barcode* adalah kumpulan garis berbentuk batang (bar) dengan ketebalan yang bervariasi. Setiap garis batang menggambarkan angka atau huruf yang dapat dibaca menggunakan sebuah alat barcode reader. Kode baris yang disusun berderet secara horisontal dalam bentuk bar dan spasi berwarna hitam tebal dan tipis. Di bawah kode baris biasanya dicantumkan angka atau huruf untuk membantu pembacaan manual [27].