BAB II
DASAR TEORI

BAB II
DASAR TEORI

2.1 Tinjauan Pustaka

Penelitian tentang sistem informasi jasa penjahit busana sudah dilakukan sebelumnya oleh Chintia (2018) dengan judul "Sistem Informasi Jasa Penjahit Busana Wanita" yang menggunakan studi kasus dari *Olive Collection*. Sistem informasi pada *Olive Collection* yaitu meningkatkan proses pelayanan terhadap konsumen dengan mempermudah proses pemesanan yang terkomputerisasi dan dapat dilakukan dimanapun dan kapanpun. Sistem informasi ini menggunakan metodologi waterfall. Hasil produk yang dihasilkan dari penelitian ini adalah sebuah aplikasi pemesanan busana berbasis website [1].

Penelitian serupa dilakukan oleh Anggie (2018) dengan judul "Sistem Informasi Pola Dasar Pakaian Berbasis Website". Pengembangan pada sistem informasi ini yaitu merancang sebuah sistem yang mampu memudahkan penjahit melihat kembali pola-pola dasar yang telah dibuat sebelumnya. Sistem ini dibangun dengan menggunakan bahasa pemrograman PHP, HTML, CSS, MySQL dan JavaScript. Hasil daripada penelitian dan pembangunan sistem ini yaitu adanya fitur terkait informasi yang dibutuhkan berdasarkan pola busana yang dicari. Sehingga sistem ini mempermudah penjahit dalam menerima informasi terkait pola-pola jahitan [2].

Penelitian serupa dilakukan oleh Ridho (2018) dengan judul "Sistem Informasi Pemesanan Online Jasa Menjahit Pada Ardhina Tailor" yang menggunakan studi kasus di Ardhina Tailor. Sistem informasi yang dibangun pada Ardhina Tailor yaitu meningkatkan mutu pelayanan guna menarik minat lebih banyak konsumen dengan memberikan kemudahan kepada konsumen untuk melakukan pemesanan busana secara online dan memantau progres pesanan secara berkala. Metodologi penelitian yang digunakan yaitu *prototype*. Hasil daripada penelitian dan pembangunan sistem ini yaitu sebuah aplikasi pemesanan busana berbasis website [3].

Penelitian serupa dilakukan oleh Clara (2018) dengan judul "Promosi Jasa Menjahit Berbasis Web Dengan Fitur *Live Chat*" yang menggunakan studi kasus di Okta Tailor. Pengembangan sistem pada penelitian ini adalah mengembangkan media promosi untuk memperluas daerah pemasaran. Metodologi yang digunakan pada sistem ini yaitu metodologi *Agile Software Development.* Hasil penggunaan metodologi

Agile menghasilkan promosi jasa menjahit berbasis website dengan fitur *live chat* [4].

Penelitian serupa dilakukan oleh Elsa (2020) dengan judul "Pengembangan Sistem Informasi Industri Jasa Menjahit Online Berbasis Web Menggunakan Metodologi Waterfall". Pengembangan sistem informasi ini yaitu meningkatkan kualitas pelayanan penjualan dengan memberikan kemudahan kepada konsumen dalam pemesanan dan pembayaran yang dapat dilakukan dimanapun dan kapanpun. Metodologi pengembangan sistem ini menggunakan metodologi Waterfall. Hasil penelitian dan pembangunan sistem informasi ini yaitu sebuah aplikasi berbasis website yang memudahkan konsumen dalam pemesanan pakaian yang telah jadi atau pakaian yang menyesuaikan keinginan konsumen dengan mengirimkan data ukuran pakaian ataupun model pakaian yang diinginkan konsumen [5].

Pada penelitian ini, penulis bermaksud mengembangkan sebuah sistem informasi untuk membantu Cahaya Vermak meningkatkan kualitas pelayanan kepada konsumen dan memberikan kepuasan konsumen terhadap Cahaya Vermak. Pelayanan yang diberikan adalah dengan menampilkan progres jahitan kepada konsumen. Selain itu sistem ini juga dilengkapi pengajuan komplain jahitan dan jasa retur perbaikan jahitan. Pembangunan sistem informasi ini nantinya akan menggunakan metodologi *prototyping*. Metodologi *prototyping* dipilih karena metodologi ini dinilai dapat menghasilkan sebuah sistem informasi yang sesuai dengan kebutuhan yang ada di Cahaya Vermak. Selain itu, metodologi ini terbuka untuk melakukan perbaikan terhadap sistem apabila ditemukan fitur yang tidak sesuai dengan kebutuhan yang ada.

2.2 Landasan Teori

Dalam penelitian ini perlu adanya teori-teori yang mendasar untuk menunjang proses penelitian ini, berikut teori-teorinya :

2.2.1 Sistem Informasi

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan terkait laporan-laporan yang diperlukan. Sistem informasi dibangun oleh komponen-komponen yang disebut blok bangunan (*building block*) yang terdiri dari blok masukan, blok model, blok keluaran, blok teknologi, blok basis data dan blok kendali. Penerapan sistem informasi yang efektif dan efisien

diperlukan perencanaan, pelaksanaan, pengaturan dan evaluasi. Oleh karena itu, bahan perencanaan sistem informasi terdiri dari empat tingkatan yaitu ide, desain, pelaksaaan dan evaluasi.

Pengelolaan sistem informasi memerlukan unsur manusia sebagai manajer yang menentukan tingkah laku suatu organisasi dalam mencapai suatu tujuan dalam sistem. Selain itu pengendalian sistem informasi merupakan tahapan penting dalam proses pengelolaan informasi. Pengelola sistem informasi perlu memahami dan memiliki keterampilan manajerial dalam melaksanakan kegiatan pengendalian sistem informasi yaitu pertama, kemampuan mengendalikan kegiatan perencanaan informasi. Kedua, kemampuan mengendalikan proses transformasi informasi. Ketiga, kemampuan mengendalikan organisasi pelaksanaan sistem informasi. Terakhir, kemampuan melaksanakan kegiatan koordinasi. Pengendalian sistem informasi ditujukan untuk menjamin kelancaran pelaksanaan pengelolaan.

Komponen penilaian merupakan komponen terakhir dan komponen yang strategis. Produk yang dicapai oleh sistem memiliki mutu dan jumlah yang tinggi. Fungsi utama dari penilaian informasi adalah ketersediaan informasi sebagai bahan pertimbangan dalam membuat keputusan [6].

2.2.2 Konveksi

Konveksi adalah industri kecil rumahan yang bekerja di bidang pembuatan pakaian jadi secara massal. Konveksi digolongkan sebagai usaha kecil menengah (UKM) [7]. Pakaian jadi yang dijahit yaitu kemeja, kaos, polo shirt, jacket, celana dan seragam.

Konveksi digolongkan dalam banyak jenis seperti besar kecilnya perusahan, jenis layanan yang tersedia. Kategori yang termasuk ke dalam besar kecilnya usaha, yaitu :

a. Konveksi Rumah adalah konveksi yang menghasilkan produksi rendah. Pakaian yang diproduksi minimal kurang dari enam potong. Harga layanan pada koveksi rumahan terhitung lebih mahal dibandingkan dengan jasa konveksi umum dan lebih murah dari jasa penjahit rumahan.

b. Konveksi Pabrik adalah jasa konveksi pakaian dalam bentuk pabrik. Memiliki skala produksi yang besar hingga ribuan potong jahitan.

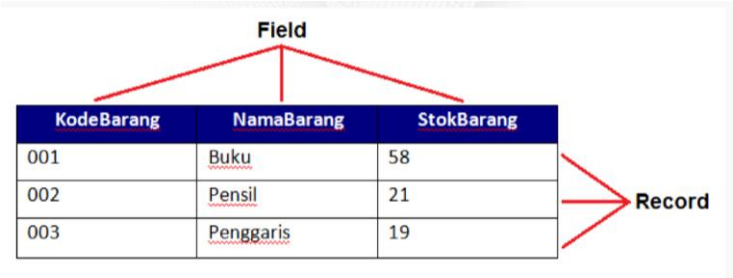
Adapun kategori yang termasuk dalam jenis konveksi berdasarkan layanannya, yaitu :

a. Jasa Pola adalah sebuah layanan pembuatan pola. Pola yang disediakan berupa pola pakaian seragam sesuai standar konveksi atau pola yang dibuat berdasarkan permintaan konsumen.
b. Jasa Potong adalah sebuah layanan yang paling banyak disediakan pada konveksi. Jasa ini menggunakan alat atau mesin dalam pemotongan kainnya.
c. Jasa Bordir adalah sebuah layanan jasa yang hanya disediakan oleh konveksi umum. Pola yang dibuat biasanya mengikuti permintaan konsumen.
d. Jasa Packing adalah sebuah layanan berupa pelipatan pakaian atau perapihan pakaian untuk dikemas sesuai dengan persetujuan sebelumnya.
e. Jasa Vermak adalah sebuah layanan berupa memodifikasi atau memperbaiki pakaian yang sudah jadi. Jasa ini biasanya digunakan apabila pakaian yang dibeli konsumen kurang sesuai atau rusak [8].

2.2.3 Basis Data

Basis data terdiri dari dua kata yaitu basis dan data. Basis adalah gudang. Data adalah catatan atas kumpulan informasi seperti manusia, barang, hewan, konsep dan peristiwa yang diwujudkan dalam bentuk angka, huruf, simbol, gambar, teks, bunyi atau kombinasi. Basis data adalah himpunan kelompok data yang saling terhubung dan diorganisasikan yang tersimpan di dalam media penyimpanan elektronis.

Basis data memiliki 8 operasi dasar yaitu *create database, drop database, create table, drop table, insert, read, update dan delete.* Sebuah data didalam database memiliki baris dan kolom. Setiap baris yang memiliki data disebut *record* dan setiap kolom untuk menyimpan karakteristik semua baris disebut *fields*. Adapun contoh penerapan kolom dan baris pada basis data dapat dilihat pada Gambar 2.1.



Gambar 2. 1 Contoh Penerapan Baris dan Kolom Pada Basis Data

Komponen basis data terdiri dari *hardware* atau perangkat keras. *Hardware* berfungsi sebagai pendukung operasi pengolahana data yang terdiri dari CPU, disk, terminal dan memori. Komponen selanjutnya yaitu *software operating system* (OS) atau sistem operasi perangkat lunak. *Software* OS merupakan perangkat yang memfungsikan, mengendalikan seluruh sumber daya serta melakukan operasi dasar pada sistem komputer. *Software* OS memiliki beragam jenisnya seperti linux, unix, windows, ubuntu, apple dan sebagainya. *Software* OS harus kompatibel atau sesuai dengan software pengelola basis data yang digunakan. Komponen selanjutnya yaitu *database management system* (DBMS) atau sistem manajemen basis data. DBMS ini digunakan oleh user untuk mengontrol, memelihara dan mengakses data secara praktis dan efisien. Bahasa pemrograman yang disediakan oleh DBMS terdiri dari *data definition language* (DDL) yaitu digunakan untuk membuat tabel baru, memuat indeks atau mengubah tabel. *Data Manipulation Language* (DML) yaitu digunakan untuk memanipulasi dan pengambilan data dari *database* seperti menambahkan data baru, menghapus data atau mengubah data. Komponen terakhir yaitu *software* program aplikasi merupakan aplikasi yang digunakan untuk pembangunan *database*. Aplikasi yang digunakan dalam pembangunan *database* adalah mySQL, Microsoft SQL server dan oracle [9].

2.2.4 Pemrograman Berbasis Orientasi

Pemrograman Berbasis Objek (PBO) adalah pemrograman procedural yang membangi sebuah aplikasi berdasarkan fungsi yang ada di dalamnya. PBO memecah komponen-komponennya menjadi objek

yang saling berinteraksi. Keuntungan menggunakan PBO, sebagai berikut :

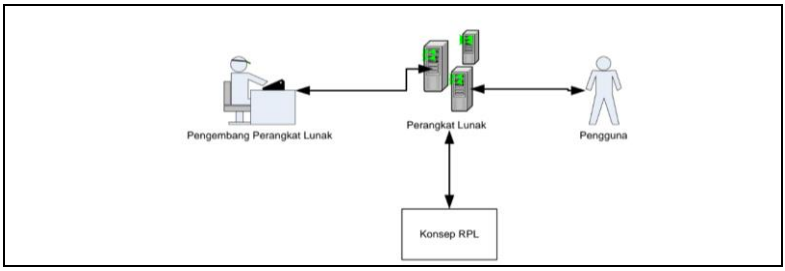
1. *Real World Programming* yaitu sebuah program disusun oleh objek yang masing-masing memiliki fungsi sesuai dengan peran dan kebutuhan interaksinya.
2. *Reusability of Code* yaitu kelas yang telah dibuat dalam pemrograman dapat digunakan oleh program lain.
3. *Resilience to Change* yaitu program yang memodelkan dunia nyata yang bersifat dinamis.
4. *Information Hiding* yaitu menyembunyikan objek dari luar kelas yang ada. Tujuannya mengamankan data agar pihak luar menerima data yag dibutuhkan saja.
5. *Modularity of Code* yaitu objek yang terbentuk dikelola secara terpisah dengan objek yang lain.

Fitur yang terdapat dalam PBO, sebagai berikut :

1. *Encaptulation* atau enkapsulasi adalah cara menyembunyikan implementasi detail dari sebuah kelas.
2. *Abstraction* atau abstraksi adalah atribut dari sebuah objek.
3. *Inheritance* atau pewarisan adalah pembuatan hierarki dari sebuah kelas dan membantu pendefinisian atribut dan *method* bagi kelas keturunannya. Sebuah kelas yang disebut *superclass* menurunkan atribut dan *method*nya kepada lainnya yang disebut *subclass.*
4. *Polymorphism* atau polimorfisme adalah pemberian arti dan fungsi yang berbeda dalam penggunaan sebuah entitas yang memiliki indentitas yang sama [10].

2.2.5 Rekayasa Perangkat Lunak

Rekaya Perangkat Lunak (RPL) adalah sebuah disiplin ilmu yang mencakup segala hal yang berhubungan dengan proses pengembangan perangkat lunak sehingga siklus hidup perangkat lunak dapat berlangsung secara efisien dan terukur. RPL memiliki unsur yang terdiri dari *software engineers* atau pengembang perangkat lunak, *software* atau perangkat lunak dan *user* atau pengguna. Adapun hubungan unsur RPL dengan RPL dapat dilihat pada Gambar 2.2.



Gambar 2. 2 Hubungan unsur RPL dengan RPL

Rekayasa perangkat lunak memiliki beberapa model proses yang dipilih berdasarkan pada sifat proyek dan aplikasi, metode dan alat-alat yang akan digunakan, dan control dan kiriman yang diperlukan, sebagai berikut :

a. *Waterfall Model* (Linear Sequential Model) adalah model yang paling banyak digunakan dan paling tua.
b. *Prototype Models* adalah pelanggan dan pengembang bertemu untuk menentukan tujuan keseluruhan untuk perangkat lunak dan mengidentifikasi persyaratan yang berlaku. Prototipe diuji oleh pengguna untuk memperbaiki persyaratan untuk perangkat lunak yang akan dikembangkan.
c. *RAD (Rapid Application Development) Models* adalah model pengembangan perangkat lunak yang menekankan siklus perkembangan yang sangat pendek. RAD merupakan adaptasi dari siklus *waterfall models* [11].

2.2.6 Prototipe

Prototipe adalah sebuah model pengembangan sistem yang dikembangan dengan cepat untuk memeriksa persyaratan atau kelayakan dari beberapa keputusan desain yang diminta klien. Dalam penggunaan model ini memungkinkan pengguna untuk berinteraksi dan bereksperimen dengan model kerja dari sistem. Prototipe memberikan ruang bagi klien untuk merasakan kondisi sebenarnya dari sistem yang akan dikembangkan.

Prototipe dikembangkan dengan harapan dapat melakukan evolusi prototipe ke dalam sistem kerja, sebagai berikut :

a. Dalam rekayasa persyaratan yaitu prototipe dapat membantu dengan elisitasi dan validasi persyaratan sistem. Hal ini bertujuan

untuk mengungkapkan kesalahan dalam persyaratan. Spesifikasi kemudian dimodifikasi untuk mencerminkan perubahan.

b. Dalam desain sistem yaitu melakukan percobaan yang sesuai untuk memeriksa kelayakan desain yang diusulkan.

Sehingga dengan harapan yang telah diuraikan, model prototipe dibuat sebagai keperluan untuk komunikasi dengan pengguna *(listen to costumer)* agar antara pengembang dengan pengguna memiliki komunikasi lebih banyak terkait perancangannya.

Kelebihan menggunakan model prototipe adalah :

a. Meningkatkan kepercayaan pengguna.
b. Pengembang mendapatkan pengalaman dan wawasan lebih luas terkait pengimplementasian persyaratan yang lebih baik.
c. Model prototipe digunakan untuk memperjelas persyaratan yang tidak jelas.
d. Mencegah dan mengurangi resiko yang dapat terjadi.

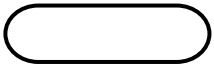
Kekurangan menggunakan model prototipe adalah :

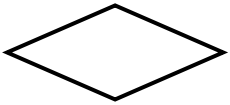
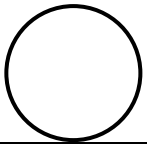
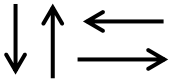
a. Selalu terjadi pengembangan model prototipe jika terjadi ketidakpuasan dari pelanggan.
b. Pengembang kehilangan focus tujuan prototipe yang sesungguhnya [12].

2.2.7 *Flowchart*

Flowchart adalah sekumpulan gambar yang menunjukkan setiap tindakan yang terjadi dalam program. Simbol-simbol *flowchart* standar yang dikeluarkan oleh *American National Standards Institue* atau ANSI dan *The International Organization for Standardization* atau ISO dapat dilihat pada Tabel 2.1 [13].

Tabel 2. 1 Simbol Flowchart

No.	Simbol	Nama	Fungsi
1.		Terminal	Menyatakan permulaan atau akhir suatu program.
2.		*Input/Output*	Menyatakan proses masuk atau keluar tanpa tergantung jenis peralatannya.

3.		Proses	Menyatakan suatu Tindakan (proses) yang dilakukan oleh komputer.
4.		*Keputusan*	Menunjukkan suatu kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban : ya/tidak.
5.		*Connector*	Menyatakan sambungan dari proses ke proses lainnta dalam halaman yang sama
6.		*Offline Connector*	Menyatakan sambungan dari proses ke proses lainnya dalam halaman yang berbeda.
7.		*Predefned Process*	Menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal.
8.		*Punched Card*	Menyatakan masukan berasal dari kartu atau keluaran ditulis di kartu
9.		*Document*	Mencetak keluaran dalam bentuk dokumen (melalui printer).
10.		*Flow*	Menyatakan jalannya arus suatu proses.

2.2.8 Unified Modelling Language (UML)

Unified modelling language (UML) adalah metodologi pemodelan visual yang digunakan sebagai sarana perancangan sistem berorietasi objek. Tujuan dan fungsi diperlukan adanya UML sebagai berikut :

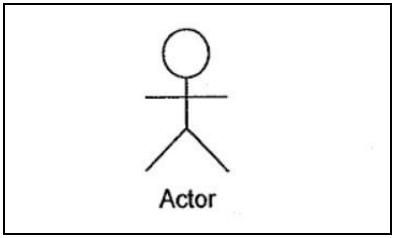
a. Dapat memberikan bahasa pemodelan visual atau gambar kepada para pengguna dari berbagai macam pemrograman maupun proses rekayasa.
b. Menyatukan informasi dalam pemodelan.
c. Memberikan gambaran model.
d. Mempermudah pengguna membaca suatu sistem.
e. Sebagai *blueprint* [14].

Adapun contoh diagram UML, yaitu :

a. *Usecase Diagram*

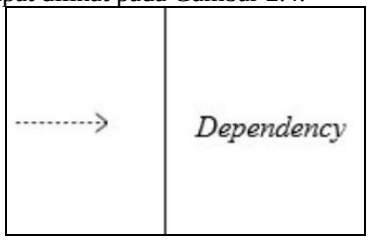
Adalah salah satu jenis dari UML yang menggambarkan interaksi antara sistem dan aktor. Adapun simbol-simbol usecase diagram, sebagai berikut :

1. Aktor sebagai pemeran yang berinteraksi dengan sistem. Simbol aktor dapat dilihat pada Gambar 2.3.



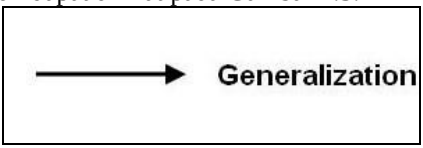
Gambar 2. 3 Aktor di dalam *UseCase Diagram*

2. *Dependecy* adalah proses ketika hubungan suatu elemen mandiri dan tidak dapat dipengaruhi oleh elemen lain. Simbol *dependency* dapat dilihat pada Gambar 2.4.



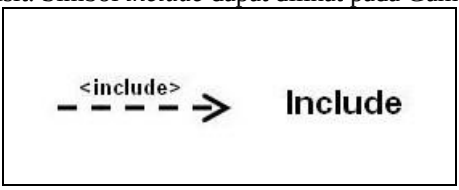
Gambar 2. 4 *Dependecy* di dalam *Usecase Diagram*

3. *Generalization* adalah objek anak membagikan struktur data dan perilakunya dari objek induk, sedangkan objek induk ini berarti objek yang berada diatasnya atau *ancestor*. Simbol *generalization* dapat dilihat pada Gambar 2.5.



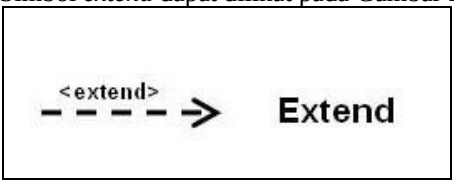
Gambar 2. 5 *Generalization* di dalam *Usecase Diagram*

4. *Include* berfungsi mengaktegorikan usecase sumber dengan cara eksplisit. Simbol *include* dapat dilihat pada Gambar 2.6.



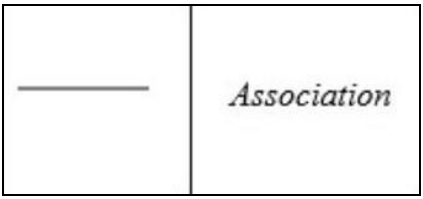
Gambar 2. 6 *Include* di dalam *Usecase Diagram*

5. *Extend* berguna mengkategorikan apabila usecase melakukan perluasan perilaku dari sumber ke suatu titik yang telah diberikan. Simbol *extend* dapat dilihat pada Gambar 2.7.



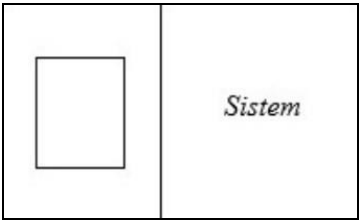
Gambar 2. 7 *Extend* di dalam *Usecase Diagram*

6. *Association* adalah suatu garis yang menghubungkan suatu objek dengan objek yang lain. Simbol *associaton* dapat dilihat pada Gambar 2.8.



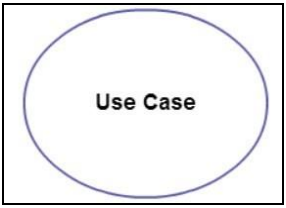
Gambar 2. 8 Association di dalam *Usecase Diagram*

7. Sistem digunakan untuk menspesifikasi paket dengan menunjukkan sistem secara terbatas. Simbol sistem dapat dilihat pada Gambar 2.9.



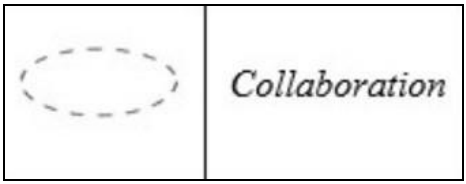
Gambar 2. 9 Sistem di dalam *Usecase Diagram*

8. *Usecase* berbentuk elips, digunakan untuk menunjukkan sebuah keterangan atas urutan aksi yang ditampilkan oleh sistem, menghasilkan aktor lain yang lebih terstruktur. Simbol *usecase* dapat dilihat pada Gambar 2.10.



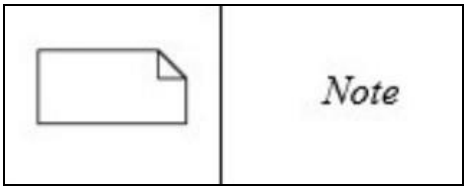
Gambar 2. 10 Simbol *Usecase* di dalam *Usecase Diagram*

9. *Collaboration* merupakan berbagai aturan dan elemen yang bekerja guna menyediakan aksi yang lebih besar. Simbol *collaboration* dapat dilihat pada Gambar 2.11.



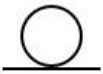
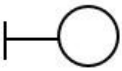
Gambar 2. 11 *Collaboration* di dalam *Usecase Diagram*

10. *Note* adalah elemen untuk menggambarkan suatu daya komputasi[18]. Simbol *note* dapat dilihat pada Gambar 2.12 [15].



Gambar 2. 12 *Note* di dalam *Usecase Diagram*

b. *Sequence Diagram*
Adalah sebuah diagram yang menjelaskan tentang interaksi objek berdasarkan urutan waktu. Adapun simbol yang tedapat pada *sequence diagram* dapat dilihat pada Gambar 2.13 [14].

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menggambar orang yang sedang berinteraksi dengan sisitem.
2		*Entity Class*	Menggambarkan hubungan yang akan dilakukan
3		*Boundary Class*	Menggambarkan sebuah gambbaran dari foem
4		*Control Class*	Menggambarkan penghubung antara boundary dengan tabel
5		*A focus of Control & A Life Line*	Menggambarkan tempat mulai dan berakhirnya massage
6		*A massage*	Menggambarkan Pengiriman Pesan

Gambar 2. 13 Simbol *Sequence Diagram*

c. *Class Diagram*
Adalah alur diagram yang memetakan suatu struktur tertentu dengan memodelkan kelas, atribut, operasi dan hubungan antar objek satu sama lain. Adapun simbol yang terdapat di dalam *class diagram* dapat dilihat pada Gambar 2.14 [16].

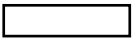
NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

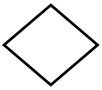
Gambar 2. 14 Simbol *Class Diagram*

d. *ERD* (Entity Relationship Diagram)

ERD adalah alat yang digunakan untuk memodelkan struktur data dengan menggambarkan entitas dan hubungan antar entitas secara abstrak. ERD memiliki beberapa notasi dan simbol. Adapun simbol ERD dapat dilihat pada Tabel 2.2 [17].

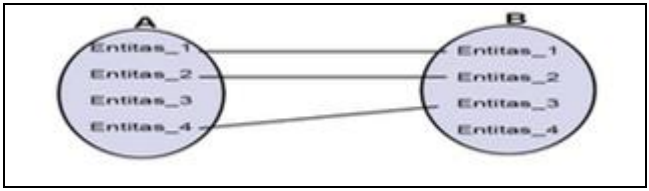
Tabel 2. 2 **Simbol ERD**

No.	Simbol	Nama	Keterangan
1.		Entitas/Entity	Entitas merupakan data inti yang akan disimpan
2.		Atribut	Kolom data yang disimpan dalam suatu entitas

3.		Atribut kunci primer	Kolom data dalam suatu entitas yang menjadi kunci akses *record* yang diinginkan. Biasanya berupa id. Dalam penamaan data tersebut menggunakan garis bawah.
4.		Atribut multi nilai/ values	Kolom data yang disimpan dalam suatu entitas yang dapat memiliki nilai lebih
5.		Relasi	Relasi atau hubungan yang menghubungkan antar entitas, biasanya diawali dengan kata kerja
6.		Asosiasi	Penghubung antara relasi dengan entitas di kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian. Kemungkinan jumlah maksimum keterhubungan antara entitas satu dengan yang lain disebut kardinalitas. Misalkan ada kardinalitas 1 ke N atau sering disebut dengan one to many yang menghubungkan entitas A dan entitas B.

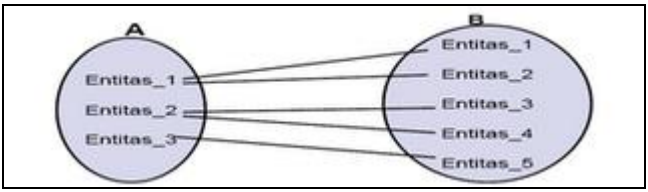
Kardinalitas adalah hubungan yang terjadi antara satu atau lebih entitas. Macam-macam kardinalitas sebagai berikut :

1. *One to One* (1 to 1) yaitu setiap anggota pada entitas A hanya boleh berelasi tepat satu dengan anggota B dan sebaliknya. Kardinalitas *one to one* dapat dilihat pada Gambar 2.15.



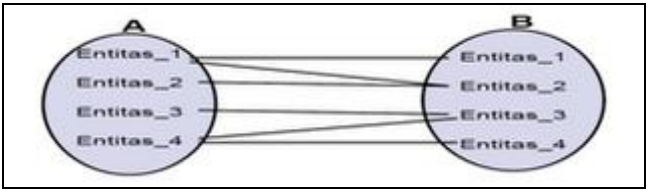
Gambar 2. 15 Kardinalitas *One to One*

2. *One to Many* (1 to N) yaitu setiap anggota entitas A boleh berelasi lebih dari satu anggota entitas B dan tidak berlaku untuk sebaliknya. Kardinalitas *one to many* dapat dilihat pada Gambar 2.16.



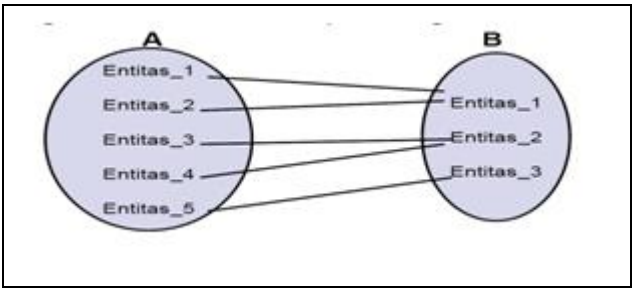
Gambar 2. 16 Kardinalitas *One to Many*

3. *Many to Many* (N to N) yaitu setiap anggita entitas A boleh berelasi lebih dari satu anggota entitas B dan berlaku sebaliknya. Kardinalitas *many to many* dapat dilihat pada Gambar 2.17.



Gambar 2. 17 Kardinalitas *Many to Many*

4. *Many to One* (N to 1) yaitu setiap anggota A berelasi tepat satu anggota B tetapi entitas B berelasi lebih dari satu anggota A. Kardinalitas *many to one* dapat dilihat pada Gambar 2.18 [18].



Gambar 2. 18 Kardinalitas *Many to One*