BAB II
LANDASAN TEORI

2.1 Kajian Pustaka

Untuk membangun sebuah aplikasi yang baik agar sesuai dengan kebutuhan, maka diperlukan referensi yang lain seperti berikut :

Implementasi Logika Fuzzy Untuk Sistem Pendukung Keputusan (Studi Kasus di Toko Komputer Mascom Sukoharjo) (Jatmiko et al, 2016). Aplikasi Sistem pendukung keputusan ini dibangun berbasis aplikasi desktop, yaitu dengan Visual Basic 6. Kekurangan dari aplikasi ini, penulis melihat hasil akhir dari yang ditampilkan masih kurang akurat, hal tersebut karena saat *user* memasukkan inputan untuk laptop kurang dari tiga juta, masih menampilkan laptop yang bernilai lebih dari tiga juta.

Sistem Pendukung Keputusan Pemilihan Guru Bidang Studi Komputer Menggunakan Metode Simple Additive Weighting (SAW) (Simarmata et al, 2018). Aplikasi Sistem pendukung keputusan ini dibangun menggunakan VB net 2008 dengan hasil akhir yang masih kurang bagus, yaitu hasil akhirnya tidak menampilkan hasil ranking, namun hanya daftar nama dan nilai yang dihasilkan sesuai urutan saat penginputpan.

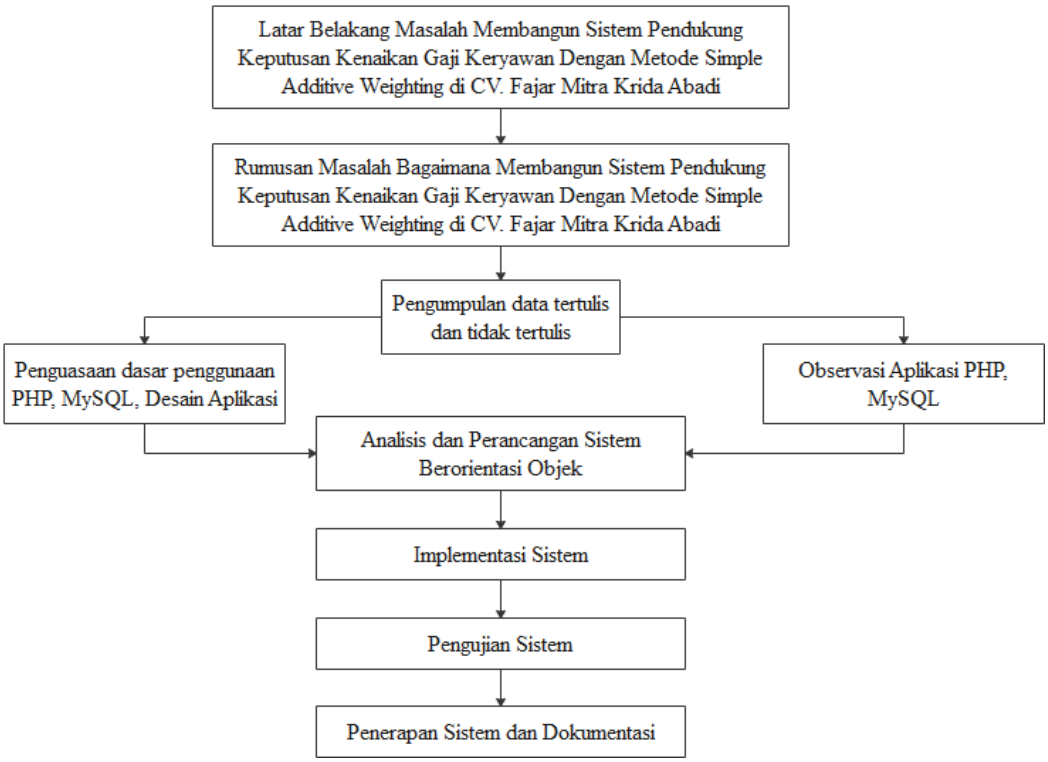
Sistem Pendukung Keputusan Kelayakan Sertifikasi Guru Menggunakan Metode Simple Additive Weighting (SAW) (Ishak et al, 2017). Aplikasi sistem pendukung keputusan ini dibangung menggunakan delphi. Keluaran yang dihasilkan dari sistem masih kurang bagus, hal tersebut dikarenakan hasil *ranking* yang dihitung manual dengan hasil *ranking* yang dibuat oleh sistem tidak menunjukkan kesamaan, dimana dari hitung manual, ahmad menjadi *ranking* pertama, sedangkan perhitungan dari aplikasi, arif menjadi *ranking* pertama.

Dari penelitian diatas yang sudah dilakukan sebelumnya, penulis akan mengembangkan sebuah penelitian sistem pendukung keputusan kenaikan gaji karyawan dengan metode simple additive weighting dengan penambahan kriteria-kriteria absolute, benefit dan cost dan penambahan modul *history* agar saat melakukan pengujian data baru, tidak harus menghapus data yang sudah diinputkan

sebelumnya. Penulis menggunakan *metode simple additive wighting* (SAW) karena metode tersebut sudah dibuktikan oleh penelitian sebelumnya dan metode tersebut lebih mudah dimengerti dan cocok untuk kriteria yang penulis gunakan untuk pendukung keputusan.

2.2 Kerangka Pemikiran

Kerangka pemikiran dari penelitian tugas akhir ini telah tercantum pada Gambar 2.1.



Gambar 2. 1 Kerangka Pemikiran

Keterangan kerangka pemikiran Tugas Akhir dijelaskan sebagai berikut :

1. Latar belakang masalah

   Pokok permasalahan yang mendasari perlunya adanya pembangunan sistem pendukung keputusan kenaikan gaji karyawan dengan metode *Simple Additive Weighting* di CV. Fajar Mitra Krida Abadi.

2. Perumusan masalah

   Perumusan masalah merupakan inti permasalahan dan jalan keluar menyelesaikan permasalahan.

3. Pengumpulan data tertulis dan tidak tertulis

   Pada penelitian dilakukan pengumpulan data secara tertulis dan tidak tertulis pada CV. Fajar Mitra Krida Abadi. Pengumpulan data penelitian ini menggunakan metode observasi, dokumentasi, dan wawancara.

4. Penguasaan dasar

   Pada penelitian dilakukan percobaan membuat sistem agar lebih menguasai.

5. Observasi sistem

   Pada penelitian dilakukan pengamatan pada sistem yang sudah ada agar dapat menjadi referensi dalam membangun aplikasi ini.

6. Analisis dan perancangan sistem

   Pada penelitian dilakukan menganalisa dan merancang bagaimana sistem nantinya akan dibuat untuk membantu memecahkan permasalahan yang ada.

7. Implementasi sistem

   Pada penelitian dilakukan implementasi apa yang sudah dirancang untuk pembangunan sistem pendukung keputusan kenaikan gaji karyawan dengan metode *Simple Additive Weighting* di CV. Fajar Mitra Krida Abadi.

8. Pengujian sistem

   Pada penelitian ini dilakukan uji coba aplikasi apakah masih terjadi kesalahan maupun kekurangan pada sistem.

9. Penerapan sistem dan Dokumentasi

   Sistem yang sudah diimplementasikan dan diuji coba kemudian diterapkan pada CV. Fajar Mitra Krida Abadi dan setelah itu dibuatnya dokumentasi dari keseluruhan kegiatan penyusunan Tugas Akhir.

2.3 Terori Pendukung

2.3.1 Pengertian Sistem Informasi

Sistem Informasi (SI) adalah kombinasi dari orang-orang, perangkat keras, perangkat lunak, jaringan komunikasi, sumber daya data, dan kebijakan serta prosedur dalam menyimpan, mendapatkan kembali, mengubah, dan menyebarkan informasi dalam suatu organisasi (O'Brien & Marakas, 2017).

2.3.2 Konsep Dasar Sistem Pendukung Keputusan

Menurut Sutabri, Sistem Pendukung Keputusan (SPK) dapat didefinisikan sebagai suatu sistem yang dapat menyediakan fungsi pengelolaan data berdasarkan model tertentu, sehingga pengguna sistem dapat memilih alternatif pengambilan keputusan yang terbaik. Harus ditekankan di sini bahwa DSS bukanlah alat pengambilan keputusan, tetapi alat bantu (Sutabri, 2005). Secara lebih rinci, ciri-ciri sistem yang dapat dikatakan DSS adalah sebagai berikut :

a. Berdasarkan pendekatan sistem yang luas yang dapat memberikan dukungan untuk proses pengambilan keputusan, sistem ini menekankan pada konsep manajemen perseptual.
b. Ada penerapan konsep manusia-mesin dimana manusia bertindak sebagai pengontrol mesin dan mesin bertindak sebagai sarana pendukung.
c. Kemampuan untuk mendukung proses pengambilan keputusan pada isu-isu semi terstruktur dan tidak terstruktur.
d. Memanfaatkan fungsi model selama analisis, baik model matematika, model statistik, atau jenis model lainnya.
e. Informasi yang diperlukan untuk mendukung fungsi interaktif dapat disediakan sehingga kita dapat dengan mudah memperoleh informasi yang diperlukan.
f. Terdapat subsistem terintegrasi yang dapat mendukung semua level manajemen.
g. Didukung oleh *database* yang lengkap.
h. Menerapkan sistem tampilan *easy to use*.

i. Dinamis dalam menghadapi masalah baru.

j. Pengambil keputusan memiliki kendali penuh atas semua langkah proses pengambilan keputusan. (Sutabri , 2005)

2.3.3 Algoritma *Simple Additive Weighting*

Metode *Simple Additive Weighting* (SAW) juga biasa disebut dengan metode penjumlahan terbobot. Konsep dasar metode *Simple Additive Weighting* adalah mencari jumlah terbobot dari peringkat kinerja setiap alternatif pada semua atribut (Munthe, 2013).

Kriteria penilaian dapat ditentukan sendiri dengan kebutuhan perusahaan (Turban, 2001).

$$R_{ij} \begin{cases} \dfrac{X_{ij}}{Max\ X_{ij}} & \text{Jika j adalah attribute benefit} \\ \dfrac{Min\ X_{ij}}{X_{ij}} & \text{Jika j adalah attribute cost} \end{cases}$$

Keterangan :

R ij = *Rating* kinerja ternormalisasi

Max X ij = Nilai maksimum dari setiap baris dan kolom

Min X ij = Nilai minimum dari setiap baris dan kolom

X ij = *Rating* kinerja dari alternatif Ai pada atribut Cj

$$V_i = \sum_{j=1}^{n} W_j R_{ij}$$

Keterangan :

Vi = Nilai akhir dari alternatif

Wi = Bobot yang telah ditentukan

Rij = *Rating* kinerja ternormalisasi

Menurut Munthe, Ada beberapa langkah dalam penyelesaian metode *Simple Additive Weighting* (SAW) adalah sebagai berikut :

1. Mengidentifikasi kriteria yang akan digunakan sebagai acuan dalam pendukung keputusan, yaitu Ci.

2. Menentukan tingkat kesesuaian untuk setiap alternatif berdasarkan kriteria masing-masing.
3. Membuat matriks keputusan berdasarkan kriteria (Ci).
4. Kemudian melakukan normalisasi matriks tersebut sesuai dengan persamaan yang menyesuaikan jenis atribut (atribut keuntungan dan atribut biaya) sehingga didapatkan matriks yang dinormalisasi R.
5. Hasil akhir diperoleh dari proses perangkingan yaitu penjumlahan dari perkalian matriks ternormalisasi R dengan vektor bobot sehingga diperoleh nilai. (Munthe, 2013)

2.3.4 Basis Data

Kumpulan dari kelompok data yang saling berhubungan yang diatur sedemikian rupa sehingga dapat digunakan kembali dengan cepat dan mudah adalah definisi dari *database*. Penggunaan *database* yang lebih banyak digunakan untuk memenuhi tujuan kecepatan, kenyamanan, efisiensi ruang penyimpanan, akurasi, ketersediaan, integritas, keamanan, dan penggunaan bersama (Hidayatullah et al, 2017).

Sedangkan menurut Rosa dan Salahuddin, "Basis Data adalah media di mana data disimpan sehingga dapat diakses dengan mudah dan cepat". Jadi, Basis Data adalah kumpulan data yang saling terkait yang dapat diakses oleh beberapa perangkat lunak (Rosa & Salahuddin, 2015).

2.3.5 PHP

Hidayatullah et al menyatakan bahwa "PHP *Hypertext Preprocessor* atau disingkat PHP adalah bahasa *scripting* yang khusus digunakan untuk pengembangan web. Karena sifatnya yang *server-side scripting*, untuk menjalankan PHP harus menggunakan *web server*" (Hidayatullah et al, 2017).

Hidayatullah et al menyatakan bahwa sebelum kita menggunakan PHP, kita akan mempelajari sintaks dasar PHP itu sendiri, ketika kita membuat *file* program php, kita harus memenuhi beberapa aturan sintaks (Hidayatullah et al, 2017). Di bawah ini adalah contoh sintaks dasar menggunakan PHP :

```
<!doctype html>
<HTML>
    <HEAD>
        <META charset=”utf-8”>
        <TITLE>Testing</TITLE>
    </HEAD>
    <?php
        Echo “Sintaks dasar php”;
    ?>
    <BODY>
    </BODY>
</HTML>
```

2.3.6 HTML

Sebuah bahasa *markup* untuk membuat halaman web, bahasa yang digunakan masih sangat standar, misalnya salah satu fungsinya untuk membuat tabel, menambahkan objek suara, video dan animasi adalah pengertian dari HTML (Hidayatullah et al, 2017).

Menurut Sulistiono, HTML (*Hypertext Markup Language*) adalah bahasa markup yang digunakan untuk membuat halaman web yang menampilkan berbagai informasi seperti gambar, teks, video, dan suara pada *web browser Internet*, ditulis dalam file format ASCII untuk menghasilkan tampilan komposit (Sulistiono, 2018).

Pengertian di atas dapat disimpulkan bahwa HTML adalah sebuah dokumen yang berisikan *tag*, beberapa elemen dan atribut untuk menampilkan halaman pada *web browser*.

2.3.7 CSS

Menurut Heru, CSS (*Cascading Style Sheets*) adalah aturan yang digunakan untuk mengontrol beberapa komponen di web agar lebih terstruktur dan terpadu. CSS bukanlah bahasa pemrograman. Sama seperti gaya dalam

aplikasi pengolah kata, Microsoft Word dapat mengatur berbagai gaya, seperti judul, subbagian, teks isi, *footer*, gambar, dan lainnya, untuk digunakan bersama dalam beberapa *file*. Umumnya, CSS digunakan untuk memformat tampilan halaman web yang dibuat dalam bahasa HTML dan XHTML (Sulistiono, 2018).

CSS dapat mengontrol parameter seperti ukuran gambar, warna teks, warna tabel, ukuran batas, warna batas, warna *hyperlink*, warna *mouseover*, spasi paragraf, spasi teks, kiri, kanan, atas, dan bawah. CSS adalah bahasa *style sheet* yang digunakan untuk mengontrol tampilan dokumen. Menggunakan CSS, memungkinkan kita untuk menampilkan halaman yang sama dalam format yang berbeda

2.3.8 Javascript

Javascript adalah bahasa pemprograman tingkat tinggi untuk membuat *website* dinamis, dan Jquery adalah *library* atau pustaka dari javascript yang dirancang untuk memudahkan penerapan *client side scripting* dan menyajikan sebuah paradigma baru pada penanganan *event* pada Javascript (Sulistiono, 2018)

2.3.9 Framework Codeigniter

Codeigniter adalah sebuah aplikasi *open source* berupa *framework* atau kerangka kerja untuk membangun *website* dengan menggunakan bahasa pemrograman PHP. Tujuannya adalah untuk memungkinkan pengembangan proyek lebih cepat daripada menulis kode dasar atau terstruktur dengan menyediakan banyak perpustakaan yang biasa digunakan dalam pengembangan. Antarmuka yang sederhana dan struktur logis untuk mengakses perpustakaan membuat CodeIgniter mudah digunakan dan dipelajari. Codelgniter ditulis atau dibuat oleh Ellis Lab dan dirilis pertama kali pada 28 Februari 2006 (Sulistiono, 2018).

2.3.9.1 MVC (*Model, View, Controller*)

MVC merupakan rangkaian logika yang disajikan dalam perangkat lunak terpisah, sehingga dalam implementasi ini meminimalkan pengkodean berulang.

Berikut pengertian dari Model. View, dan Controller :

1. *Model*

Model adalah sekumpulan logika yang mengimplementasikan manajemen struktur *database*, biasanya berhubungan langsung dengan *database* untuk memanipulasi data (menyisipkan, memperbarui, menghapus, mencari) dan menangani validasi dari bagian pengontrol, tetapi bukan bagian tampilan secara langsung.

2. *View*

View adalah substruktur yang menangani logika presentasi. Dalam aplikasi web, bagian ini biasanya berupa *file template* HTML yang dikelola oleh pengontrol. *View* digunakan untuk menerima data dan menyajikannya kepada pengguna. Bagian ini tidak memiliki akses langsung ke bagian *model*. *Folder views* berisi *file* atau *file* dengan ekstensi .php, biasanya *form*, tabel, paragraf, dll.

3. *Controller*

Controller adalah penghubung antara *model* dan *view*, menangani permintaan dari klien ke server dan memprosesnya sebagai HTTP *request* untuk dijadikan tampilan sebuah *website* (Sulistiono, 2018)

2.3.10 MySQL

Basis data adalah kumpulan data (*elementer*) yang secara logis terkait untuk mewakili fenomena/fakta secara terstruktur dalam *domain* tertentu untuk mendukung aplikasi pada beberapa sistem. Basis data adalah kumpulan data yang saling terkait yang mencerminkan fakta-fakta yang terkandung dalam suatu organisasi.

Sebuah basis data menggambarkan keadaan sebuah organisasi /perusahaan /sistem. Ketika suatu peristiwa terjadi di dunia nyata yang mengubah keadaan organisasi/perusahaan/sistem, perubahan harus dilakukan pada data yang disimpan dalam basis data. Basis data merupakan komponen utama dari sistem informasi karena semua informasi pengambilan keputusan berasal dari data dalam basis data. Manajemen basis data yang buruk dapat mengakibatkan hilangnya akses ke data

penting yang digunakan untuk menghasilkan informasi yang diperlukan untuk membuat keputusan (Hariyanto, 2004).

MySQL adalah sebuah perangkat lunak sistem manajemen basisdata SQL atau dikenal dengan DBMS (*Database Management System*), *database* ini *multithread*, *multi-user*. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL(*Sructured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleski dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis

Beberapa keunggulan dari *MySQL*:

1) *Portability*

Mysql dapat dijalankan stabil pada berbagai sistem operasi seperti Windows, Linux, freebsd dan masih banyak lagi.

2) Multiuser

Mysql dapat digunakan oleh beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

3) Security

Mysql memiliki beberapa lapisan sekuritas seperti *level subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta *password* terenkripsi.

4) *Scalability* dan *limits*

Mysql mampu menangani *database* dalam skala besar, dengan jumlah *record* lebih dari 50 juta dan 60 ribu tabel serta 5 milyard baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada setiap tabelnya (Huda & Bunafit Komputer, 2010).

2.3.11 Apache

Apache adalah server web yang berjalan di banyak sistem operasi (Unix, BSD, Linux, Microsoft Windows, dan Novell Netware, serta platform lain yang berguna untuk melayani dan menjalankan situs web). Protokol yang digunakan untuk melayani fasilitas web/www ini menggunakan HTTP Apache yang memiliki

fitur–fitur canggih seperti pesan error yang dapat dikonfigurasi, otentikasi berbasis database, dll. Apache juga didukung oleh sejumlah *Graphical User Interface* (GUI) yang memungkinkan penanganan server dengan mudah. Apache adalah perangkat lunak sumber terbuka yang dikembangkan oleh komunitas terbuka yang mencakup pengembang yang disponsori oleh Apache Software (Mulyadi, 2016).

2.3.12 *Unifed Modeling Language* (UML)

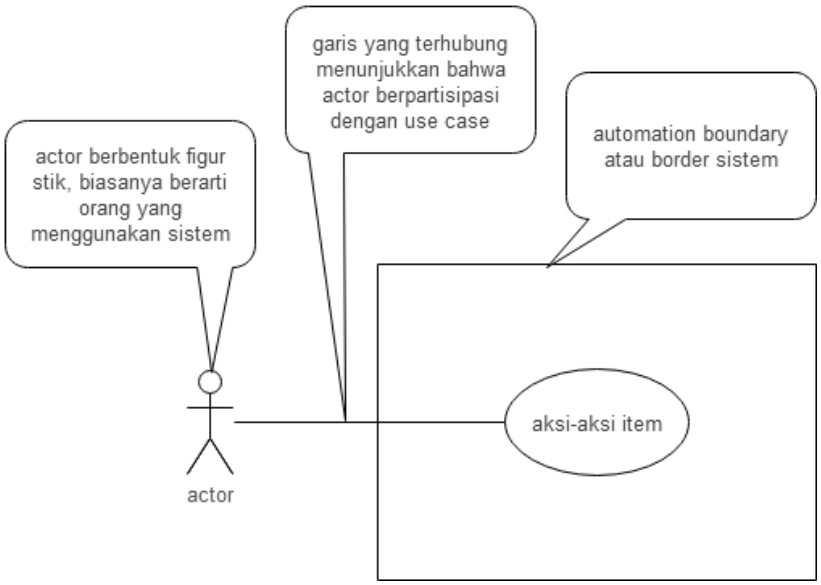
Unified modeling language (UML) adalah metode yang banyak digunakan untuk menggambarkan dan mendokumentasikan perangkat lunak dalam mendesain sistem (Shelly & Rosenblatt, 2012).

Unified Modeling Language (UML) adalah satu set standar pembuatan model dan notasi yang dikembangkan oleh *Object Management Group* (OMG), yang merupakan standar organisasi untuk pengembangan sistem (Satzinger et al, 2012).

Dengan demikian, kesimpulan dari konsep *Unified Modeling Language* (UML) adalah metode yang banyak digunakan untuk menggambarkan dan mendokumentasikan perangkat lunak dalam desain sistem dan merupakan dasar untuk pendekatan *object oriented*.

2.3.12.1 *Use Case Diagram*

Diagram *use case* adalah diagram yang digunakan untuk memodelkan proses bisnis dari perspektif pengguna sistem. Diagram ini menunjukkan kumpulan dari *use case* dan aktor-aktor (tipe kelas khusus). Diagram ini sangat penting untuk mengatur dan memodelkan perilaku sistem yang dibutuhkan dan diharapkan pengguna. Diagram *use case* terdiri dari batasan, use case, dan aktor. *Boundary* adalah diagram persegi panjang yang menggambarkan lingkungan komunikasi dalam sistem dengan penggunanya (yaitu peserta). Aktor menggambarkan pengguna yang berinteraksi dengan sistem aplikasi. Use case menggambarkan tindakan yang dapat dilakukan oleh aktor. *Use case* adalah elips dengan nama operasi di dalamnya. *Use case* dihubungkan dengan garis lurus untuk menunjukkan hubungan dengan aktor. Simbol-simbol dalam *Use Case Diagram* ditunjukan pada Gambar 2.2.

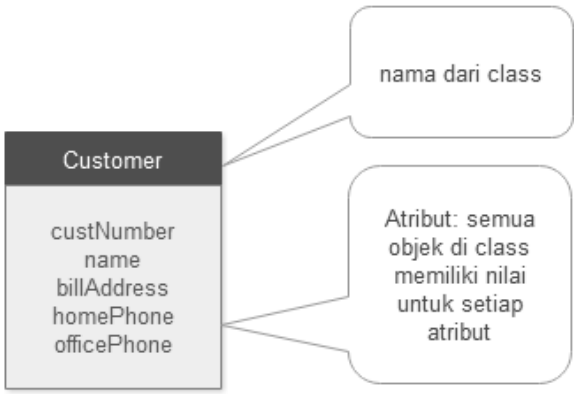


Gambar 2. 2 Simbol *Use Case Diagram*

(Sumber **:** Satzinger et al, 2012**)**

2.3.12.2 *Class Diagram*

Diagram kelas adalah diagram yang menggambarkan struktur suatu sistem, yang mendefinisikan kelas-kelas yang dibuat untuk membangun sistem. Sebuah kelas memiliki tiga bagian utama, properti, operasi, dan nama. Kelas-kelas yang ada dalam struktur sistem harus dapat menjalankan fungsi sesuai dengan kebutuhan sistem. Simbol- simbol *Class Diagram* ditunjukan pada Gambar 2.3.

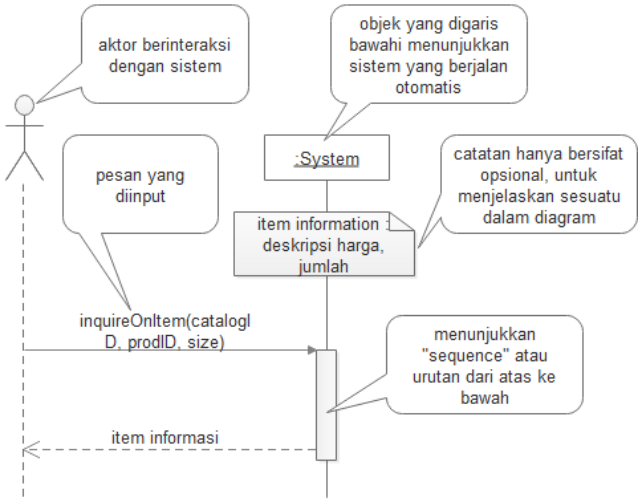


Gambar 2. 3 Simbol *Class Diagram*

(Sumber **:** Satzinger et al, 2012**)**

2.3.12.3 *Sequence Diagram*

Diagram ini memperlihatkan interaksi yang menekankan pada pengiriman pesan *(message)* dalam suatu waktu tertentu. Simbol-simbol *Sequence Diagram* ditunjukan pada Gambar 2.4.



Gambar 2. 4 Simbol *Sequence Diagram*

(Sumber **:** Satzinger et al, 2012**)**

2.3.12.4 *Activity Diagram*

Diagram ini memperlihatkan aliaran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi dalam suatu sistem dan memberi tekanan pada aliran kendali antar objek. Simbol-simbol *Activity Diagram* ditunjukan pada Tabel 2.1.

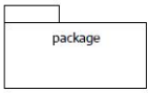
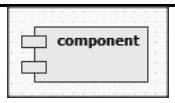
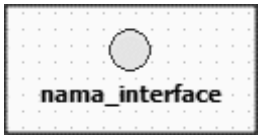
Tabel 2. 1 Simbol *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

2.3.12.5 *Component Diagram*

Component diagram adalah suatu bentuk implementasi dari jenis diagram yang menunjukan sistem arsitektur dan komponen logis secara keseluruhan di dalamnya (Satzinger et al., 2012).

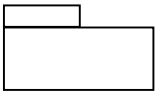
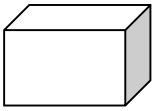
Tabel 2. 2 Simbol *Component* Diagram

No	GAMBAR	NAMA	KETERANGAN
1		*Package*	*Package* merupakan sebuah bungkusan dari satu atau lebih komponen.
2		Komponen	Komponen sistem.
3		Ketergantungan (*Dependency*)	Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai.
4		Antarmuka (*Interface*)	Sama dengan *interface* pada pemrograman berbasis objek, yaitu sebagai antarmuka komponen agar tidak mengakses langsung komponen.
5		*Link*	Relasi antar komponen.

2.3.12.6 *Deployment* Diagram

Deployment diagram merupakan gambaran proses-proses berbeda pada suatu sistem yang berjalan dan bagaimana relasi di dalamnya. Hal inilah yang mempermudah *user* dalam pemakaian sistem yang telah dibuat dan diagram tersebut merupakan diagram yang statis (Satzinger et al., 2012).

Tabel 2. 3 Simbol *Deployment* Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Package*	Package merupakan sebuah bungkusan dari satu atau lebih *node*
2		*Node*	Biasanya mengacu pada perangkat keras (*hardware*) . Perangkat lunak yang tidak biat sendri (*software*) jika didalam *node* disertakan komponen untuk mengkonsistenkan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelumnya pada diagram komponen.
3		*Depedency*	Kebergantungan antara *node*, arah panah mengarah pada *node* yang dipakai.
4		*Link*	Relasi antar *node*

2.3.13 *Blackbox Testing*

Pengujian *black box* merupakan pendekatan komplementer dari teknik *white box*, karena pengujian black box diharapkan mampu mengungkap kelas kesalahan yang lebih luas dibandingkan teknik *white box*. Pengujian *black box* berfokus pada pengujian persyaratan fungsional perangkat lunak, untuk mendapatkan serangkaian kondisi input yang sesuai dengan persyaratan fungsional suatu program (Asrul et al, 2022).

Pengujian *blackbox* memberikan pengembang *web* untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Pengujian *blackbox* bukan merupakan alternatif dari pengujian *whitebox*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *whitebox*.

Pengujian *blackbox* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

a) Fungsi-fungsi yang salah atau hilang
b) Kesalahan *interface*
c) Kesalahan dalam struktur data atau akses *database* eksternal
d) Kesalahan performa
e) kesalahan inisialisasi dan terminasi

Dengan mengaplikasikan pengujian *blackbox*, diharapkan dapat menghasilkan sekumpulan kasus uji yang memenuhi kriteria berikut:

a) Kasus uji yang berkurang, jika jumlahnya lebih dari 1, maka jumlah dari uji kasus tambahan harus didesain untuk mencapai ujicoba yang cukup beralasan.
b) Kasus uji yang memberitahukan sesuatu tentang keberadaan atau tidaknya suatu jenis kesalahan daripada kesalahan yang terhubung hanya dengan suatu pengujian yang spesifik.