BAB II

LANDASAN TEORI

2.1 Penilaian Kinerja

Kamus Besar Bahasa Indonesia (1995 : 503, 690) menjelaskan tentang penilaian dan kinerja sebagai berikut;

Penilaian yaitu mempunyai pengertian proses atau cara menilai. Dalam bahasa Inggris sering diartikan dengan kata measurement yang berarti sistem pengukuran. Sedangkan kinerja mempunyai arti kemampuan kerja. Dalam bahasa Inggris kinerja sering diartikan dengan kata performance yang mempunyai arti pelaksanaan. Jadi pengertian penilaian kinerja (*performance measurement*) mengandung makna suatu proses atau sistem pengukuran mengenai pelaksanaan kemampuan kerja suatu organisasi.

Penilaian kinerja menurut Mulyadi (2001 : 353) penentuan secara periodik efektivitas operasional suatu organisasi, bagian organisasi, dan personelnya, berdasarkan sasaran, standar dan kriteria yang telah ditetapkan sebelumnya.

Siegel dan Macroni (dalam Hessel N. S. Tangkilisan, 2003 : 107) memperjelas bahwa penilaian kinerja adalah penentuan secara periodik efektivitas operasional dan karyawannya berdasarkan saran, standar dan kriteia yang telah ditetapkan sebelumnya.

Pada dasarnya penilaian kinerja merupakan penilaian perilaku personal dalam melaksanakan tugasnya untuk tujuan organisasi. Secara umum pengukuran kinerja merupakan suaru cara untuk mengukur arah dan kecepatan perubahan. Pengukuran kinerja sangat berperan nantinya dalam proses evaluasi kinerja

perusahaan. Evaluasi kinerja adalah proses membandingkan antara kinerja aktual dan target yang telah direncanakan oleh manajemen, untuk mengidentifikasikan tindakn-tindakan perbaikan yang perlu dilakukan untuk menjamin tercapainya tujuan perusahaan dan untuk mengkomunikasikannya kepada pihak-pihak yang berwenang.

Berdasarkan uraian diatas, ukuran kinerja mempunyai peranan yang sangat penting dalam efektivitas proses evaluasi kinerja. Hal ini dikarenakan informasi yang diperoleh dalam proses evaluasi kinerja sangat bergantung pada ukuran kinerja yang digunakan.

2.2 Promosi Karyawan

Promosi jabatan adalah perpindahan dari suatu jabatan ke jabatan lain yang mempunyai status dan tanggung jawab yang lebih tinggi (Martoyo, 2007 : 71).

Sedangkan menurut Hasibuan (2008 : 108) promosi jabatan adalah perpindahan yang memperbesar *authority* dan *responsibility* karyawan ke jabatan yang lebih tinggi di dalam suatu organisasi sehingga kewajiban hak, status dan penghasilannya semakin besar.

Lain halnya menurut Tohardi yang dikutip dari Flippo (2002 : 382) bahwa promosi jabatan adalah merupakan perubahan dari pekerjaan yang satu ke yang lain yang mempunyai syarat-syarat lebih baik dalam hal kedudukan dan tanggung jawab.

Dari definisi tersebut di atas dapat disimpulkan bahwa promosi jabatan karyawan mempunyai arti yang penting bagi perusahaan, sebab dengan promosi berarti kestabilan perusahaan dan moral karyawan yang akan lebih terjamin.

Promosi akan selalu diikuti oleh tugas, tanggung jawab yang lebih tinggi daripada jabatan yang diduduki sebelumnya. Pada umumnya promosi juga diikuti dengan peningkatan pendapatan serta fasilitas yang lain. Namun, promosi ini sendiri sebenarnya mempunyai nilai karena merupakan bukti pengakuan, antara lain terhadap prestasinya.

2.3 Sistem Pendukung Keputusan

Sistem Pendukung Keputusan (SPK) atau *Decision Support System* (DSS) adalah sebuah sistem yang mampu memberikan kemampuan pemecahan masalah maupun kemampuan pengkomunikasian untuk masalah dengan kondisi semi terstruktur dan tak terstruktur. Sistem ini digunakan untuk membantu pengambilan keputusan dalam situasi semi terstruktur dan situasi yang tidak terstruktur, dimana tak seorangpun tahu secara pasti bagaimana keputusan seharusnya dibuat (Turban, 2001).

Sprague dan Watson mendefinisikan Sistem Pendukung Keputusan (SPK) sebagai sistem yang memiliki lima karakteristik utama yaitu (Sprague et.al, 1993):

1. Sistem yang berbasis computer,
2. Dipergunakan untuk membantu para pengambil keputusan,
3. Untuk memecahkan masalah-masalah rumit yang mustahil dilakukan dengan kalkulasi manual,
4. Melalui cara simulasi yang interaktif ,
5. Dimana data dan model analisis sebaai komponen utama.

2.4 Multi Criteria Decision Making

Multi Criteria Decision Making adalah salah satu metode yang membantu proses pengambilan keputusan yang memiliki banyak kriteria. Menurut Mulliner, Malys, dan Maliene (2016), *Multi Criteria Decision Making* adalah seperangkat metode yang berhubungan dengan evaluasi serangkaian alternatif yang banyak, sering bertentangan, dan berbagai kriteria. Tujuan dari *Multi Criteria Decision Making* adalah untuk memberikan pilihan, peringkat, deskripsi, klasifikasi, pengelompokan, dan untuk mengurutkan alternatif dari yang paling disukai hingga opsi yang paling tidak disukai.

Menurut Asadabadi (2018) metode-metode *Multi Criteria Decision Making* pada saat ini sudah banyak dikembangkan untuk memfasilitasi penyeleksian terhadap alternatif yang memiliki banyak kriteria. Di antaranya terdapat beberapa metode *MCDM* yang telah banyak digunakan seperti berikut:

- *Analytical Hierarchy Process (AHP)*
- *Analytical Network Process (ANP)*
- *Preference Ranking Organization Method for Enrichment of Evaluations (PROMETHEE)*
- *Vlsekriterijumska Optimizacija I Kompromisno Resenje (Multi-criteria optimation and compromise solution or VIKOR)*
- *Elimination Et Choix Traduisant la Realite (Elemination and Choise Expressing Reality or ELECTRE)*
- *Best Worst Method (BWM)*
- *Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)*
- *Decision Making Trial and Evaluation Laboratory (DEMATEL).*

2.5 TOPSIS (Technique For Others Reference by Similarity to Ideal Solution)

Menurut Ridaini (2014 : 34) "*TOPSIS* menggunakan prinsip bahwa alternatif yang terpilih harus mempunyai jarak terdekat dari solusi ideal positif dan terjauh dari solusi ideal negatif dari sudut pandang geometris dengan menggunakan jarak *euclidean* untuk menentukan kedekatan relatif dari suatu alternatif dengan solusi ideal, metode yang digunakan dalam sistem pendukung keputusan skripsi ini adalah *TOPSIS* meskipun dengan alur algoritma yang sederhana tetapi dapat menjadi bahan solusi terhadap permasalahan dalam menentukan objek lokasi".

TOPSIS (*Technique For Others Reference by Similarity to Ideal Solution*) adalah salah satu metode pengambilan keputusan multikriteria yang pertama kali diperkenalkan oleh Yoon dan Hwang (1981). *TOPSIS* menggunakan prinsip bahwa alternatif yang terpilih harus mempunyai jarak terdekat dari solusi ideal positif dan terjauh dari solusi ideal negatif dari sudut pandang geometris dengan menggunakan jarak euclidean untuk menentukan kedekatan relatif dari suatu alternatif dengan solusi optimal. Solusi ideal positif didefinisikan sebagai jumlah dari seluruh nilai terbaik yang dapat dicapai untuk setiap atribut, sedangkan solusi negatif-ideal terdiri dari seluruh nilai terburuk yang dicapai untuk setiap atribut.

Adapun algoritma penyelesaian dalam Metode *TOPSIS* (*Technique For Others Reference by Similarity to Ideal Solution*) yaitu sebagai berikut:

Langkah 1 : Menentukan data alternatif, kriteria dan sifat.

Langkah 2 : Menentukan nilai alternatif.

Langkah 3 : Membuat matriks keputusan ternomalisasi.

TOPSIS membutuhkan ranking kinerja setiap alternatif Ai pada setiap kriteria Cj yang ternormalisasi yaitu : $r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}}$

dengan i=1,2,...,m ; nilai m menunjukkan jumlah alternatif yang dievaluasi, dan nilai Xij menunjukkan nilai rating kecocokan alternatif ke-i terhadap kriteria ke-j

Langkah 4 : Membuat matriks keputusan ternomalisasi terbobot.

$$y_{ij} = w_j . r_{ij}$$

Nilai dari masing-masing data ternormalisasi (R) kemudian dikalikan dengan bobot (W) untuk mendapatkan matriks keputusan ternormalisasi terbobot (Y). Dengan wj adalah pangkat bernilai positif untuk atribut keuntungan (*benefit*), dan bernilai negatif untuk atribut biaya (*cost*). Nilai wj menunjukkan nilai bobot dari kriteria C yang ke-j.

Langkah 5 : Menentukan matriks solusi ideal positif dan matriks solusi ideal negative. $A^+ = (y_1^+, y_2^+, y_3^+, \dots, y_n^+)$

$$A^- = (y_1^-, y_2^-, y_3^-, \dots, y_n^-)$$

Solusi ideal positif A+ dan solusi ideal negatif A- dapat ditentukan berdasarkan ranking bobot ternormalisasi (yij).

Langkah 6 : Menentukan jarak antara nilai setiap alternatif dengan matriks solusi ideal positif dan negatif.

$$D_i^+ = \sqrt{\sum_{j=1}^{n} (y_i^+ - y_{ij})^2} \quad D_i^- = \sqrt{\sum_{j=1}^{n} (y_{ij} - y_j^-)^2}$$

Langkah 7 : Menentukan nilai preferensi untuk setiap alternatif.

$$V_i = \frac{D_i^-}{D_i^- + D_i^+}$$

2.6 SMART (Simple Multi Attribute Rating Technique)

Menurut Pratiwi (2016 : 141) "*Simple Multi Attribute Rating Technique (SMART)* merupakan suatu teknik atau metode yang multiattribut dalam sistem pengambilan keputusan."

Metode *SMART* dikembangkan pada tahun 1977 oleh Edward. Teknik pembuatan keputusan multiatribut ini dapat digunakan untuk mendukung pembuat keputusan dalam memilih antara beberapa alternatif, setiap pembuat keputusan harus memilih sebuah alternatif yang sesuai dengan tujuan yang telah dirumuskan. Setiap alternatif terdiri dari sekumpulan atribut dan setiap atribut mempunyai nilai - nilai. Nilai ini di rata - rata dengan skala tertentu.

Setiap atribut mempunyai bobot yang menggambarkan seberapa penting dibandingkan dengan atribut lain. Pembobotan dan pemberian peringkat digunakan untuk menilai setiap alternatif agar diperoleh alternatif terbaik.

Adapun algoritma penyelesaian dalam Metode *SMART (Simple Multi Atribute Rating Technique)* yaitu sebagai berikut :

Langkah 1 : Menentukan jumlah kriteria dari keputusan yang akan di ambil.

Langkah 2 : Menentukan bobot kriteria menggunakan interval 0 - 100 berdasarkan prioritas terpenting dari masing - masing kriteria.

Langkah 3 : Melakukan normalisasi bobot kriteria.

Menghitung normalisasi bobot dari setiap kriteria dengan membandingkan nilai bobot kriteria dengan jumlah bobot kriteria, menggunakan persamaan: $w_i = \frac{w_i'}{\sum_{j=1}^{m} w_j}$

Langkah 4 : Memberikan nilai parameter untuk tiap kriteria.

Memberikan nilai kriteria untuk setiap alternatif, nilai kriteria untuk setiap alternatif ini dapat berbentuk data kuantitatif (angka) ataupun berbentuk data kualitatif, misalkan nilai untuk kriteria harga sudah dapat dipastikan berbentuk kuantitatif sedangkan nilai untuk kriteria fasilitas bisa jadi berbentuk kualitatif (sangat lengkap, lengkap, kurang lengkap). Apabila nilai kriteria berbentuk kualitatif maka kita perlu mengubah ke data kuantitatif dengan membuat parameter nilai kriteria, misalkan sangat lengkap artinya 3, lengkap artinya 2 dan tidak lengkap artinya 1.

Langkah 5 : Menentukan nilai utility.

Menentukan nilai utility dengan mengkonversikan nilai kriteria pada masing-masing kriteria menjadi nilai kriteria data baku. Nilai utility ini tergantung pada sifat kriteria itu sendiri. Terdapat 2 kriteria yaitu kriteria biaya & kriteria keuntungan.

Kriteria Biaya (Cost Criteria) :

Kriteria yang bersifat "lebih diinginkan nilai yang lebih kecil" kriteria seperti ini biasanya dalam bentuk biaya yang harus dikeluarkan (misalkan kriteria harga, kriteria penggunaan bahan bakar per kilometer untuk pembelian mobil, periode pengembalian modal dalam suatu usaha, kriteria

waktu pengiriman) dapat dihitung dengan menggunakan persamaan :

$$u_i(a_i) = \frac{(c_{max} - c_{out})}{(c_{max} - c_{min})}$$

Keterangan

Ui(ai) : nilai utility kriteria ke-i untuk alternatif ke-i

cmax : nilai kriteria maksimal

cmin : nilai kriteria minimal

cout : nilai kriteria ke-i

Kriteria Keuntungan (Benefit Kriteria) :

Kriteria yang bersifat "lebih diinginkan nilai yang lebih besar", kriteria seperti ini biasanya dalam bentuk keuntungan (misalkan kriteria kapasitas tangki untuk pembelian mobil, kriteria kualitas dan lainnya). Persamaan yang digunakan untuk menentukan nilai utility jenis ini adalah :

$$u_i(a_i) = \frac{(c_{out} - c_{min})}{(c_{max} - c_{min})}$$

Keterangan

uj(ai) : nilai utility kriteria ke-j untuk alternatif ke-i

cmax : nilai kriteria maksimal

cmin : nilai kriteria minimal

cout : nilai kriteria alternatif ke-i

Langkah 6 : Menentukan nilai akhir.

Menentukan nilai akhir dari masing-masing dengan mengalikan nilai yang didapat dari normalisasi nilai kriteria data baku dengan nilai normalisasi bobot kriteria. Kemudian jumlahkan nilai dari perkalian tersebut.

$$u(a_i) = \sum_{j=1}^{m} w_j * u_j(a_i)$$

Langkah 7 : Perangkingan.

Hasil dari perhitungan Nilai akhir kemudian diurutkan dari nilai yang terbesar hingga yang terkecil, alternatif dengan nilai akhir yang terbesar menunjukkan alternatif yang terbaik.

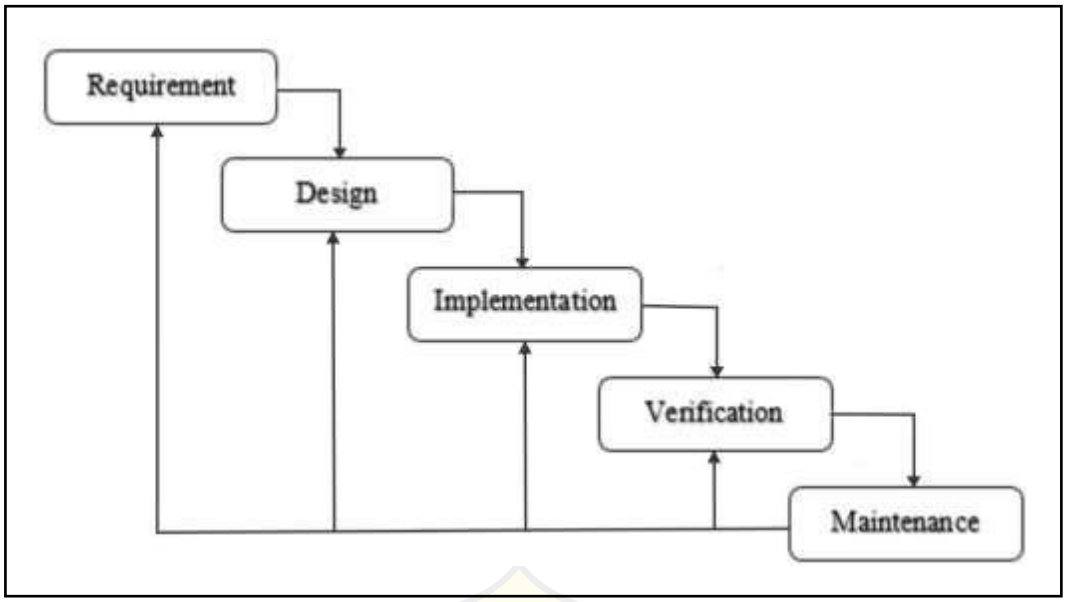
2.7 Metode Pengembangan

Dalam perancangan sistem ini, metode pengembangan yang digunakan adalah metode *system development life cycle* dengan Model *Waterfall*.

2.7.1 Model Waterfall

Menurut Pressman (2015 : 42) "*waterfall* model merupakan pola model tertua pada pengembangan software yang bertujuan untuk memberikan saran yang sistematis kepada pengembang software yang diawali dari kebutuhan pelanggan yang kemudian diteruskan hingga berpuncak pada dukungan untuk penyelesaian *software*. Proses pengembangan yang sangat terstruktur ini dapat membuat potensi kerugian akibat kesalahan pada proses sebelumnya sangat besar dan acap kali mahal karena membengkaknya biaya pengembangan ulang."

Model waterfall melakukan pendekatan secara sistematis dan terurut. Sesuai dengan namanya yaitu *waterfall*, maka tahapan dalam model ini disusun bertingkat. setiap tahap dalam model ini dilakukan berurutan, satu sebelum yang lainnya. Model ini biasanya digunakan untuk membuat sebuah *software* dalam skala besar dan yang akan dipakai dalam waktu yang lama. Berikut fase dari metode *waterfall* :



Gambar 2. 1 Model *Waterfall* (Presman, 2015)

Adapun penjelasan urutan tahapan - tahapan yang dimiliki oleh waterfall adalah sebagai berikut :

2.7.1.1 Requirement Analysis

Menurut Sugiyono (2015 : 335) “Analisis adalah sebuah kegiatan untuk mencari suatu pola. selain itu analisis juga merupakan cara berpikir yang berkaitan dengan pengujian secara sistematis terhadap sesuatu untuk menentukan bagian hubungan antar bagian dan hubungannya dengan keseluruhan.”

Seluruh kebutuhan *software* didapatkan pada fase ini. Pada tahap ini informasi dapat didapatkan melalui wawancara, survei atau diskusi yang kemudian informasi tersebut dianalisis untuk mendapatkan dokumentasi kebutuhan pengguna untuk digunakan pada tahapan selanjutnya.

2.7.1.2 Design

Tahap ini adalah tahapan sebelum memulai pemrograman. Tahap ini bertujuan untuk memberikan gambaran tentang apa yang harus dikerjakan dan bagaimana tampilannya. Tahap ini membantu dalam menspesifikasikan kebutuhan hardware dan sistem serta mendefinisikan arsitektur sistem secara keseluruhan.

2.7.1.2.1 UML (Unified Modelling Language)

UML merupakan singkatan dari Unified Modeling Language yang bisa berarti bahasa pemodelan standar. Menurut Mulyani (2016:48) *UML (Unified Modeling Language)* merupakan "Sebuah teknik pada pengembangan sistem yang menggunakan bahasa grafis sebagai alat untuk mendokumentasikan dan melakukan spesifikasi pada sistem."

Menurut A.s & Shalahuddin (2016:133) mendefinisikan bahwa "*UML (Unified Modeling Language)* adalah suatu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek". Berikut design sistem yang penulis butuhkan :

2.7.1.2.1.1 Use Case Diagram

Menurut Rachmaniah (2018 : 75) "*Use Case Diagram* berisi kebutuhan bisnis dari sistem dan juga dapat mengilustrasikan interaksi antara sistem dengan lingkungannya. *Use Case Diagram* juga merupakan ikhtisiar dalam bentuk grafis dari aktor - aktor yang terlibat dalam sistem."

Menurut A.s & Shalahuddin (2016 : 155) mendefinisikan bahwa "*Use case* atau *diagram use case* adalah pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat".

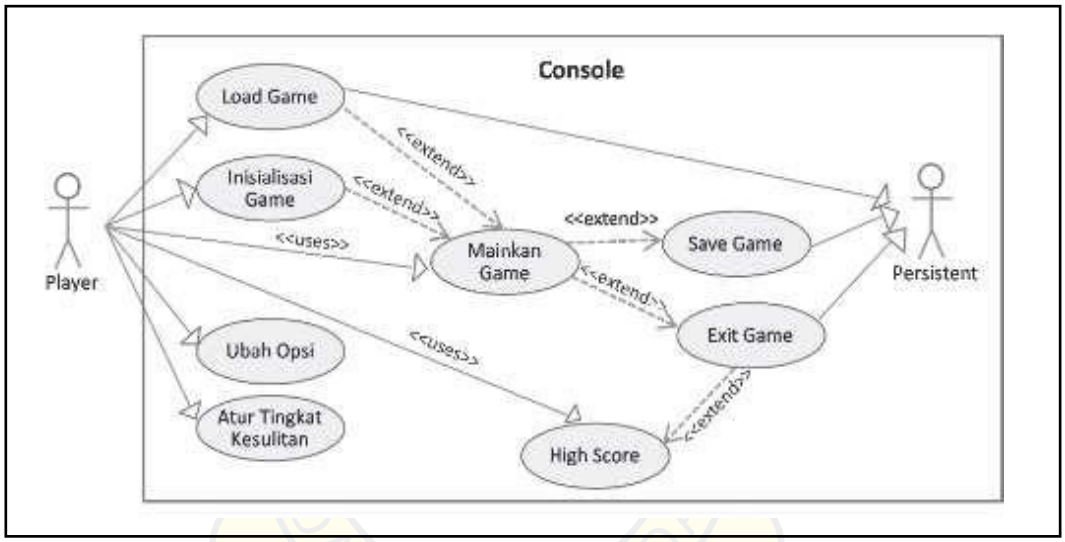
Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Berikut ini merupakan komponen - komponen *use case diagram*.

Tabel 2. 1 Sintaks Elemen - Elemen *Use Case Diagram*

Terminologi atau Definisi	Simbol
Aktor • Orang atau sistem yang memperoleh manfaat dari sistem dan bersifat eksternal dari sistem. • Diberi label sesuai perannya. • Bisa berasosiasi dengan aktor lain dengan cara *specialization* / *association,* dicirikan dengan tanda panah ujung yang kosong (*bollow arrow head).* • Ditempatkan di luar system boundary.	
Use Case • Dapat mempresentasikan suatu fungsional dari sistem. • Bisa diperluas ke use case lain.	

Association Relationship • Dapat menghubungkan aktor dengan *use case* di mana aktor tersebut berinteraksi dengan *use case* tersebut.	

Contoh *Use Case Diagram* seperti pada gambar di bawah ini :



Gambar 2. 2 Contoh *Use Case Diagram* (Rachmaniah, 2018)

2.7.1.2.1.2 Activity Diagram

Menurut Rachmaniah (2018 : 81) "*Activity Diagram* yaitu *workflow* secara grafis untuk mengilustrasikan sebuah alur bisnis atau *workflow* operasional dari komponen yang terdapat pada sistem."

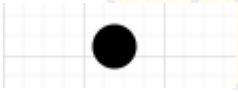
Sedangkan menurut A.s & Shalahuddin (2016 : 161) mendefiniskan bahwa "Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. perlu diperhatikan, diagram aktivitas menggambarkan aktivitas

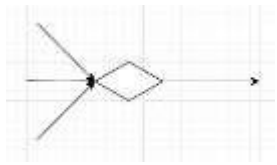
sistem bukan apa yang dilakukan aktor, jadi aktivitas apa saja yang dapat dilakukan oleh sistem."

Sedangkan menurut Rahayu, dkk (2015 : 8) *Activity Diagram* merupakan "bentuk visual dari alur kerja yang berisi aktivitas dan tindakan, yang juga dapat berisi pilihan, atau pengulangan."

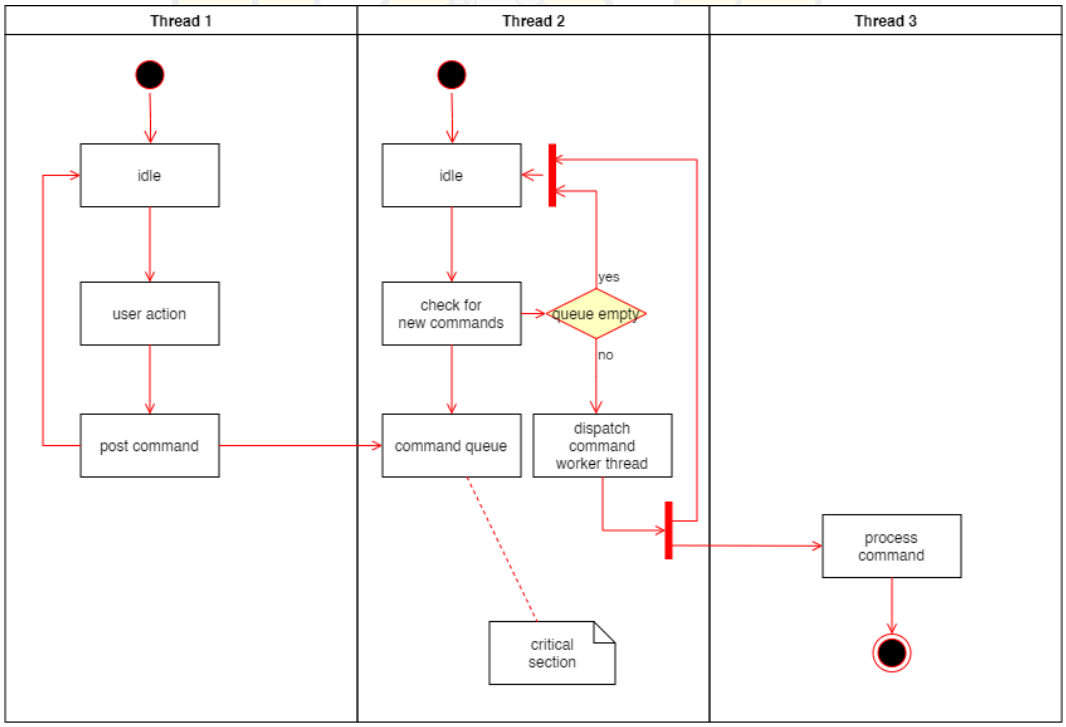
Diagram aktivitas memiliki komponen - komponen dengan bentuk yang tertentu, dihubungan dengan tanda panah. Panah tersebut juga mengarahkan urutan aktivitas yang terjadi, dari awal sampai akhir. Yang perlu diperhatikan adalah diagram aktivitas bukan menggambarkan aktivitas - aktivitas sistem yang dilakukan oleh aktor, tetapi menggambarkan aktivitas apa saja yang dapat dilakukan oleh sistem. Berikut ini merupakan komponen - komponen pada *Activity Diagram*.

Tabel 2. 2 Komponen - Komponen *Activity Diagram*

No	**Gambar**	**Nama**	**Keterangan**
1		*Start Point*	*Start Poin / Initial State* adalah sebuah lingkaran hitam kecil, yang dapat menandakan bahwa titik awal aktivitas untuk setiap diagram aktivitas.

2	Activity	*Activity*	*Activity* menunjukan aktivitas yang sedang dilakukan / sedang terjadi dalam *activity diagram*.
3		*Action / Flow*	*Action Flow* digunakan untuk melakukan transisi dari suatu tindakan ke tindakan yang lain.
4		*Merge Event*	*Merge Event* digunakan untuk menggabungkan *flow / action* yang dipecah oleh *decission*.
5		*Final State*	*Final State* menunjukkan bagian akhir dari aktivitas.

Contoh *Activity Diagram* seperti pada gambar di bawah ini :



Gambar 2. 3 Contoh *Activity Diagram*

2.7.1.2.1.3 Sequence Diagram

Menurut Rahmaniah (2018 : 82) "*Sequence Diagram* memperlihatkan bagaimana sebuah objek berinteraksi satu dengan lainnya disertai urutan terjadinya interaksi tersebut serta difokuskan pada sebuah message interchange antar *life line* (objek)."

Sedangkan Menurut A.s & Shalahuddin (2016 : 165) yaitu "*Diagram sequence* menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup sebuah objek dan message yang dikirimkan dan diterima antar objek". *Sequence Diagram* dapat menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Diagram sequence merupakan salah satu yang menjelaskan bagaimana suatu operasi itu dapat dilakukan, message (pesan) apa saja yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Objek-objek yang saling berkaitan dengan proses berjalannya operasi dapat diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.

2.7.1.3 Implementation dan Testing

Pada tahap ini, sistem pertama kali dikembangkan di program kecil yang disebut unit, yang terintegrasi dalam tahap selanjutnya. Setiap unit dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai unit testing. Selain itu pada tahap ini juga dilakukan pemeriksaan terhadap modul yang dibuat apakah sudah memenuhi fungsi yang diinginkan atau belum.

2.7.1.3.1 Website

Menurut Abdulloh (2018 : 1) "*Website* dapat diartikan sebagai sebuah kumpulan halaman yang berisi informasi data digital baik berupa teks, gambar, animasi, suara, video atau gabungan dari semuanya yang disediakan melalui jalur koneksi internet sehingga dapat diakses serta dilihat oleh semua orang di seluruh dunia."

Secara umum, *website* dapat dibagi menjadi 3 jenis, yaitu website statis, dinamis, dan interaktif. *Website* statis adalah suatu jenis website yang isinya tidak diperbaharui secara berkala, sehingga isinya dari waktu ke waktu akan selalu tetap. *Website* dinamis yaitu jenis *website* yang isinya terus diperbaharui secara berkala oleh pengelola *web* atau pemilik *website*. *Website* interaktif yaitu isi informasinya tidak hanya diubah oleh pengelola *website* tetapi juga lebih banyak dilakukan oleh para pengguna website itu sendiri.

2.7.1.3.2 HTML

Menurut Abdulloh (2018 : 7) "*HTML* singkatan dari *Hypertext Markup Language* yaitu suatu bahasa standar web yang dikelola oleh *W3C (World Wide Web Consortium)* berupa tag - tag yang menyusun setiap elemen dari *website*."

HTML terdiri atas unsur - unsur yang membentuk struktur skrip *HTML*, yaitu tag, atribut, dan elemen. Tag merupakan symbol khusus (*markup*) berupa dua karakter "<" dan ">" yang mengapit suatu teks sebagai nama tag. Atribut adalah property yang mengatur bagaiman sebuah elemen dari suatu tag akan ditampilkan. Elemen adalah bagian dari skrip *HTML* yang terdiri dari tag pembuka, isi elemen, dan tag penutup.

2.7.1.3.3 CSS

Menurut Pamungkas (2017 : 32) "*Cascanding Style Sheet (CSS)* adalah suatu file yang berisi rangkaian intruksi untuk mengatur komponen dalam sebuah halaman web sehingga akan lebih terstruktur dan rapi."

CSS dapat mengendalikan ukuran gambar, warna bagian tubuh pada teks, warna tabel, ukuran *border*, warna *border*, warna *hyperlink*, warna *mouse over*, spasi antar paragraf, spasi antar teks, margin kiri, kanan, atas, bawah, dan parameter lainnya.

Cara penulisan *CSS* dibedakan menjadi 3 macam yaitu inline, internal, dan external. Ketiganya dapat digunakan sesuai dengan kebutuhan. Inline *CSS* yaitu menuliskan *CSS* dengan menggunakan atribut style yang langsung dituliskan di dalam tag HTML. Internal *CSS* yaitu menuliskan *CSS* menggunakan tag <style> … </style>. External *CSS* yaitu menuliskan *CSS* dimana skrip *CSS* disimpan dalam file tersendiri dengan ekstensi *CSS* dan terpisah dengan file *HTML*.

2.7.1.3.4 JavaScript

Menurut Agusriandi (2018 : 128) "*JavaScript* adalah bahasa *script* yang disisipkan pada kode *HTML* dan diproses pada sisi klien yaitu yang sering disebut client side. Bahasa ini menjadikan dokumen *HTML* menjadi semakin luas. Sebagai contoh, dengan menggunakan *JavaScript* dimungkinkan untuk mem-validasi data - data yang dimasukan pada formulis sebelum data tersebut dikirimkan ke *server*."

2.7.1.3.5 PHP

Menurut Abdulloh (2018 : 127) "*PHP* merupakan singkatan *Hypertext Preprocessor* yaitu sebuah bahasa pemrograman *web* yang dapat disisipkan dalam

skrip *HTML* dan bekerja di sisi *server*. Tujuan dari bahasa ini adalah membantu para pengembangan web untuk membuat web dinamis dengan cepat."

Skrip *PHP* dituliskan di antara tanda <?php dan ?> yang memisahkan skrip *PHP* dengan skrip lainnya. Satu file *PHP* dapat berisi full skrip *PHP* atau dapat disisipkan diantara skrip lain seperti *HTML*, *CSS* maupun *JavaScript*. File yang berisi skrip *PHP* wajib disimpan dengan ekstensi *.php dan disimpan pada server (folder htdocs atau www). Jika disimpan dengan ekstensi *HTML* atau disimpan sembarangn tempat maka skrip php tidak diproses sebagaimana mestinya.

Setiap baris skrip *PHP* harus diakhiri dengan tanda semicolon yaitu (;). Jika tidak, maka akan menampilkan pesan *error*.

2.7.1.3.6 Bootstrap

Menurut Abdulloh (2018 : 261) "*Bootstrap* yaitu salah satu *framework CSS* paling popular dari sekian banyak framework *css* yang ada. *Bootstrap* juga memungkinkan desain sebuah *web* menjadi *responsive* sehingga dapat dilihat dari berbagai macam ukuran *devices* dengan tampilan tetap menarik. *Bootstrap* juga membuat proses pada pengaturan desain menjadi lebih cepat karena tidak perlu lagi banyak menulis skrip *CSS*, bahkan hampir tidak perlu kecuali jika memerlukan pengaturan desain yang berbeda dengan *style Bootstrap*. *Bootstrap* telah didukung oleh hampir semua *browser* baik pada *desktop* maupun *mobile*."

2.7.1.3.7 Jquery

Menurut Setiawan (2017 : 210) "*Jquery* merupakan sebuah *library* pada *javascript*. Dalam dunia pemrograman, *library* adalah kumpulan dari berbagai jenis fungsi yang 'siap pakai' untuk memudahkan pembuatan sebuah aplikasi. Dengan

demikian, *jquery* adalah kumpulan fungsi - fungsi javascript yang memudahkan penulisan kode *javascript*."

2.7.1.3.8 Ajax

Asynchronous JavaScript And XML, atau disingkat *Ajax*, adalah suatu teknik pada pemrograman berbasis web untuk menciptakan aplikasi *web* interaktif. Tujuannya adalah untuk memindahkan sebagian besar interaksi pada komputer web *server*, melakukan pertukaran data dengan *server* di belakang layar, sehingga pada halaman *web* tidak harus dibaca berulang - ulang secara keseluruhan setiap kali seorang pengguna melakukan perubahan. Hal ini akan meningkatkan interaktivitas, kecepatan, serta *usability*.

2.7.1.3.9 Database

Menurut Mandar (2017 : 25) "*Database* atau basis data merupakan suatu kumpulan data yang terhubung dan disimpan secara bersama - sama pada suatu media tanpa adanya suatu kerangkapan data, sehingga mudah saat digunakan kembali serta tidak mengalami suatu ketergantungan pada program yang akan menggunakannya dan dapat diakses oleh satu atau lebih program aplikasi secara optimal."

2.7.1.3.10 MySQL

Menurut Penerbit Andi (2016 : 2) "*MySQL* adalah sistem manajemen database *SQL* yang bersifat *Open Source* serta paling popular saat ini. Sistem *Database MySQL* mendukung beberapa fitur seperti *multithreaded*, *multi-user* dan *SQL database management system (DBMS)*. *Database* ini dibuat untuk keperluan sistem database yang cepat, handal, dan mudah digunakan."

2.7.1.3.11 PhpMyAdmin

Menurut Mandar (2017 : 133) "*PhpMyAdmin* merupakan sebuah software manajemen database berbasis antar muka sehingga tidak perlu lagi mengetikkan perintah - perintah *sql* dalam mengolah *database*."

PhpMyAdmin menjadi salah satu pilihan utama bagi para pengembang *web* sebab sudah tersedia dalam paket *WampServer*, *XAMPP*, atau aplikasi *server local* lainnya sehingga tidak perlu lagi menginstallnya secara terpisah.

2.7.1.3.12 Xampp

Menurut Aryanto (2016 : 5) "*XAMPP* adalah sebuah aplikasi perangkat lunak pemrograman dan database yang di dalamnya terdapat berbagai macam aplikasi pemrograman seperti; *Apache HTTP Server*, *MySql Database*, bahasa pemrograman *PHP* dan *Perl*."

2.7.1.4 Verification (Verifikasi)

Menurut Pressman (2015 : 67) "Tahapan *Verification* yaitu meliputi pengintegrasian sistem dan juga melakukan *testing* terhadap aplikasi yang telah dibuat. Sistem akan diverifikasi untuk diuji sejauh mana kelayakannya."

Dalam tahapan ini semua modul yang dikerjakan oleh programmer yang berbeda akan digabungkan kemudian diuji apakah telah sesuai dengan spesifikasi yang ditetapkan atau terdapat kesalahan / *error* dalam sistem sebelum kemudian diperbaiki ulang.

2.7.1.5 Maintenance (Pemeliharaan)

Menurut Pressman (2015 : 67) "Tahap ini merupakan tahap terakhir pada *waterfall model*. *Software* yang telah dibuat akan dilakukan maintenance.

Maintenance termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya."

Sedangkan menurut Sugiyono (2015 : 338) "Pemeliharaan merupakan sebuah operasi atau aktivitas yang harus dilakukan secara berkala dengan tujuan untuk mempercepat pergantian kerusakan peralatan dengan *resources* yang ada. Perawatan juga ditujukan untuk mengembalikan suatu sistem pada kondisinya agar dapat berfungsi, memperpanjang usia kegunaan mesin dan menekan *failure* sekecil mungkin."