BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Untuk mendukung keberhasilan penelitian ini, penyusun melakukan pendekatan teoritis melalui beberapa literatur yang berhubungan dengan penelitian yang dilakukan. Beberapa uraian penelitian terdahulu yang menjadi acuan dalam penelitian ini yaitu :

1. Fristy Riandari, 2017, dengan judul "Sistem Pendukung Keputusan Penentuan Menggunakan Metode TOPSIS Dalam Memilih Departemen Pada Kantor Balai Wilayah Sungai Sumatera II Medan)" Penelitian ini menghasilkan suatu sistem pemilihan calon kepala departemen pada kantor balai wilayah sungai sumatera II medan.
2. Diana Fatmawati, dkk, 2017, dengan judul "Sistem Pengambilan Keputusan kelayakan Bagi Calon Penerima Dana Bantuan Masyarakat Miskin Menggunakan Metode TOPSIS Berbasis Web" Penelitian ini menghasilkan sebuah sistem pendukung keputusan yang mampu menampilkan hasil dari keputusan penilaian yang lebih tepat karena didasarkan pada nilai kriteria dan bobot yang sudah ditentukan sehingga akan mendapatkan hasil yang lebih akurat.
3. Mia Purnama, dkk, 2019, dengan judul "Sistem Pendukung Keputusan Penentuan Kelompok Uang Kuliah Tunggal Menggunakan Metode TOPSIS Dengan Pembobotan Metode Rangking Pada Universitas Tanjungpura.

Pontianak Morawa”Penelitian ini menghasilkan sebuah sistem pendukung keputusan yang mampu membantu mahasiswa dalam melakukan pembayaran uang kuliah.

Dalam penelitian ini penulis akan mengembangkan suatu Sistem Pendukung Keputusan dengan menggunakan metode TOPSIS dan menambahkan Pembobotan ROC untuk menentukan Sarung Tangan Layak Jual pada PT. Intan Hevea Industri.

II.2. Landasan Teoritis

II.2.1. Sistem Pendukung Keputusan

Sistem Pendukung Keputusan (SPK) atau *Decision Support System* (DSS) mulai dikembangkan pada tahun 1970-an oleh Michael S.Cott Morton dengan istilah *Management Decision System* (Turban dkk, 2005). Sistem tersebut adalah sistem berbasis komputer yang ditujukan untuk membantu pengambilan keputusan dalam memanfaatkan data dan model tertentu untuk memecahkan bebagai persoalan yang tidak terstruktur. Menurut Kendal, 2002, *Decision Support System* (DSS) atau sistem pendukung keputusan hampir sama dengan sistem informasi manajemen tradisional karena keduanya sama-sama tergantung pada basis data sebagai sumber data dimana DSS menekankan pada fungsi pendukung pembuatan keputusan diseluruh tahap-tahapnya, walaupun keputusan aktual masih tetap wewenang eksekutif sebagai pembuat keputusan.

II.2.2. *Technique For Order PreferenceBy Similarity To IdealSolution (TOPSIS)*

Technique For Order Preference By Similarity To Ideal Solution (TOPSIS) didasarkan pada konsep dimana alternatif terpilih yang terbaik tidak hanya memiliki jarak terpendek dari solusi ideal positif, namun juga memiliki jarak terpanjang dari solusi ideal negatif (Mia Purnama dkk, 2019). Secara umum, prosedur TOPSIS mengikuti langkah-langkah sebagai berikut:

a. Menentukan matriks keputusan yang ternormalisasi
b. Menghitung matriks keputusan ternormalisasi yang terbobot
c. Menghitung matriks solusi ideal positif dan matriks solusi ideal negatif
d. Menghitung jarak antara nilai setiap alternatif dengan matriks solusi ideal positif dan matriks solusi ideal negatif
e. Menghitung nilai preferensi untuk setiap alternatif.

Dalam penelitian ini menggunakan metode TOPSIS. Adapun langkah-langkahnya adalah:

a. Membangun sebuah matriks keputusan

Matriks keputusan X dapat dilihat sebagai berikut:

$$rij = \frac{xij}{\sqrt{\sum_{i=1}^{m} x\frac{2}{ij}}} \qquad \text{(II.1)}$$

Keterangan:

i = 1,2,…,m; dan j = 1,2,…,n,

rij = matriks keputusan ternormalisasi.

xij = bobot kriteria ke j pada alternatif ke i.

b. Membuat matriks keputusan yang ternormalisasi terbobot. Matriks keputusan ternormalisasi terbobot didapatkan dari perkalian matriks R dengan bobot preferensi (30%; 25%; 20%; 15%; 10%) didapat:

$$y_{ij} = w_{ij.} r_{ij} \quad \text{(II.2)}$$

yij = matriksternormalisasiterbobot [i][j]

wi = vectorbobot[i]

rij = matriks keputusan ternormalisasi.

c. Menentukan matriks solusi ideal positif dan matriks solusi ideal negatif

$$A^+ = (\text{y1}^+, \quad \text{y2}^+, \ldots \quad \text{yi}^+) \quad \text{(II.3)}$$

$$A^- = (\text{y1}^-, \quad \text{y2}^-, \ldots \quad \text{yi}^-) \quad \text{(II.4)}$$

Keterangan:

yj+ = max yij, jika j adalahatributkeuntungan min yij, jika j adalahatributbiaya

yj- = min yij, jika j adalahatributkeuntunganmax yij, jika j adalahatributbiaya

d. Menentukan jarak antara nilai setiap alternatif dengan matriks solusi ideal positif dan negatif.

$$D^+ = \sqrt{\sum_{i=1}^{m} (yi^2 - \quad yij^+)\text{^}2} \quad \text{(II.5)}$$

Keterangan :

Di+ = jarakalternatif Ai dengansolusi ideal positif

yi+ = solusi ideal positif[i]

yij = matriksnormalisasiterbobot[i][j]

e. Jarakantaraalternatif *Ai* dengansolusi ideal negatif dirumuskansebagai:

$$D^- = \sqrt{\sum_{i=1}^{m}(yi^2 - \quad yij^-)\text{^}2} \qquad \text{(II.6)}$$

Keterangan :

Di- = jarak alternatif Ai dengan solusi ideal negatif

yi- = solusi ideal positif[i]

yij = matriksnormalisasiterbobot[i][j]

f. Nilaipreferensiuntuksetiapalternatif *(Vi)* diberikansebagai:

$$Vi = \frac{D_i^-}{D_i^- + \quad D_i^+} \qquad \text{(II.7)}$$

Dimana :

Vi = nilai referensi untuk setiap alternatif

Di+ = jarakantaraalternatif dengansolusi ideal positif

Di- = jarakantaraalternativedengansolusi ideal negatif

NilaiVi yang lebihbesarmenunjukkanbahwaalternatif Ai lebihdipilih.

II.2.3. Metode Pembobotan (*Rank Order Centroid*)

Menurut Alfa Saleh (2017) Teknik ROC memberikan bobot pada setiap kriteria sesuai dengan ranking yang dinilai berdasarkan tingkat prioritas. Biasanya dibentuk dengan pernyataan “Kriteria 1 lebih penting dari kriteria 2, yang lebih penting dari kriteria 3” dan seterusnya hingga kriteria ke n, ditulisCr1 ≥Cr2≥Cr3 ≥…≥ Crn . Untuk menentukan bobotnya, diberikan aturan yang sama yaitu W1 ≥ W2≥ Wr3 ≥ … ≥Wndimana W1 merupakan bobot untuk kriteria C1. Secara umum pembobotan ROC untuk setiap criteria dapat dirumuskan pada persamaan 3 berikut :

$$W_{k} = \frac{1}{k} \sum_{i=1}^{k} \left(\frac{1}{i}\right) \qquad \text{(II.8)}$$

Keterangan:

W= Nilai pembobotan kriteria

K= Jumlah kriteria

i= Nilai alternatif

II.2.4.UML

Unified Modeling Language (UML) adalah bahasa spesifik standar yang dipergunakan untuk mendokumentasikan, menspesifikan dan membangun perangkat lunak.UML merupakan metodologi dalam mengembangkan system berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem.UML saat ini sangat banyak digunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industry perangkat lunak dan pengembangan sistem. Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML yaitu *Use Case Diagram, Activity Diagram, Class Diagram* dan *Sequence Diagram*. (Urva dan Siregar, 2012 : 95).

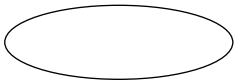
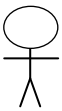
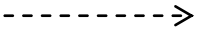
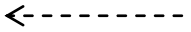
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut:

1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.

Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.1 dibawah ini :

Tabel II.1. Simbol *Use Case*

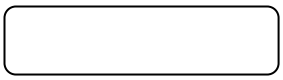
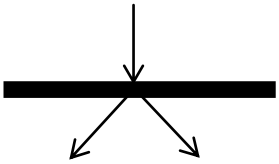
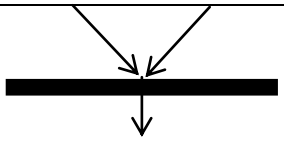
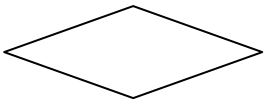
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	*Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar : 2015: 94)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2 berikut ini:

Tabel II.2. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

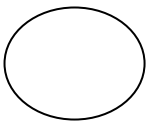
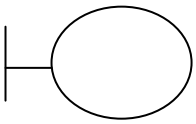
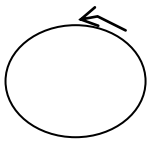
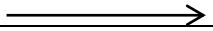
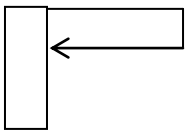
(Sumber : Gellysa Urva dan Helmi Fauzi Siregar : 2015: 94)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan

diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada label II.3. berikut ini :

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class,* berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message,* simbol mengirim pesan antar *class*.
	*Recursive,* menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

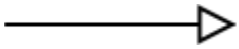
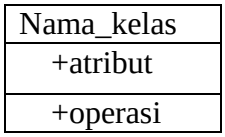
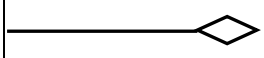
(Sumber : Gellysa Urva dan Helmi Fauzi Siregar:2015: 95)

4. Diagram Kelas (*Class Diagram)*

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan.

Class diagram secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/ Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti (Ade Hendini, 2016 : 111). Simbol *class diagram* dan*multiplicity class diagram* dapat dilihat pada Tabel II.4 dan Tabel II.5. dibawah ini :

Tabel II.4. Simbol *Class Diagram*

Gambar	Keterangan
	*Generalization*, untuk menghubungkan antar kelas dengan arti umum-khusus. Jadi jika ada kelas dengan arti umum-khusus. Jadi jika ada kelas bermakna umum dan kelas bermakna khusus dapat menggunakan simbol ini.
	*Class*, untuk sebuah kelas pada struktur sistem. Penulisan tidak boleh menggunakan spasi. Simbol ini memiliki 3 susunan, yaitu kotak pertama adalah nama kelas, kedua atribut dan ketiga operasi.
	*Interface*, untuk simbol *interface* atau dalam bahasa indonesianya antar muka. Konsep yang digunakan pun sama dengan pemrograman berorientasi object (OOP).
	*Association,*digunakan untuk menghubungkan atau merelasikan kelas satu dengan kelas yang lainnya dengan makna umum.
	*Directed Association*, adalah relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain.
	*Aggregation*, adalah relasi antar kelas dengan makna semua bagian.
	Dependency, adalah relasi antar kelas dengan makna kebergantungan antar kelas.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar : 2015 : 95

Tabel II.5. *Multiplicity Class Diagram*

Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar : 2015 : 95)

II.2.5. *Visual Studio 2010*

Menurut Ninuk Wiliani, Syadid Zambi (2018) Visual Studio 2010 pada dasarnya adalah sebuah bahasa pemrograman komputer. Dimana pengertian dari bahasa pemrograman itu adalah perintah-perintah atau intruksi yang dimengerti oleh computer untuk melakukan tugas-tugas tertentu. Visual Studio 2010 (yang sering juga disebut VB Net 2010) selain disebut dengan bahasa pemrograman, juga sering disebut sebagai sarana *(tool)* untuk menghasilkan program-program aplikasi berbasiskan windows. Visual basic adalah sebuah bahasa pemrograman yang berpusat pada object *(Object Oriented Programming)* digunakan dalam pembuatan aplikasi *windows* yang berbasis Graphical User Interface, hal ini menjadikan Visual basic menjadi bahasa pemrograman yang wajib diketahui dan dikuasai oleh setiap programmer. Beberapa karakteristik obyek tidak dapat dilakukan oleh *Visual Basic* misalnya seperti *Inheritance* tidak bisa module dan *Polymorphism* secara terbatas biasa dilakukan dengan deklarasi *class module* yang mempunyai *Interface* tertentu.

Beberapa kemampuan atau manfaat dari *Visual Studio* 2010 diantaranya seperti :

1. Untuk membuat program aplikasi berbasiskan *windows*
2. Untuk membuat objek-objek pembantu program seperti, misalnya :control *ActiveX, file Help,* aplikasi *Internet* dan sebagainya.
3. Menguji program (*debugging*) dan menghasilkan program berakhiran *EXE* yang bersifat *executable* atau dapat langsung dijalankan.

II.2.6. SQL SERVER 2008

Microsoft SQL Server merupakan produk RDBMS (Relational Database Management System) yang dibuat oleh Microsoft. Orang sering menyebutnya dengan SQL Server saja. Microsoft SQL Server juga mendukung SQL sebagai bahasa untuk memproses query ke dalam database. Mirosoft SQL Server banyak digunakan pada dunia bisnis, pendidikan atau juga pemerintahan sebagai solusi database atau penyimpanan data (Setiyadi dan Herlawati : 2019).