BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Tinjauan pustaka dari penelitian yang dilakukan sebelumnya digunakan untuk mendukung penelitian yang sedang dilakukan. Berikut ini merupakan tinjauan studi yang disajikan dalam tabel 2.1

Tabel 2.1 Tabel Literatur

No Literatur	Penulis	Tahun	Judul
Literatur 01	Rachmat Agusli, Sutarman, Suhendri	2017	Sistem Pakar Identifikasi Tipe Kepribadian Karyawan Menggunakan metode *Certainty Factor*
Literatur 02	Puji Sari Ramadhan, Usti Fatimah Sitorus Pane	2018	Sistem *E-Healthcare* Untuk Mendiagnosa Penyakit Inflamasi Dermatitis Imun Anak dengan Menggunakan metode *Certainty Factor*
Literatur 03	Ria Sartika	2017	Aplikasi Sistem Pakar Diagnosa Kepribadian Anak Dengan Solusi Pola Asuh Menggunakan metode *Certainty Factor*
Literatur 04	Heni Sulistyani, Kurnia Muludi	2018	Penerapan Metode *Certainty Factor* dalam Mendeteksi Penyakit Tanaman Karet
Literatur 05	Sam'ani, M. Haris Qamaruzzaman	2018	Sistem Pakar Pendeteksi Kerusakan *Notebook* menggunakan metode *Certainty Factor*

2.1.1 Tinjauan Literatur 01

Setiap karyawan memiliki kepribadian yang unik, tidak mudah dalam menilai kepribadian seseorang, disini pengurus perlu adanya menerapkan sebuah sistem yang dapat mendeskripsikan kepribadian setiap karyawan, dimana isi dari sistem tersebut bersumber dari para pakar dibidang kepribadian manusia oleh sebab itu dibutuhkan sistem pakar yang dapat membantu manajer untuk dapat mengidentifikasi kepribadian karyawan sehingga manajer dapat memposisikan karyawan sesuai bidangnya. Sistem tersebut dibangun menggunakan metode *Certainty Factor*. Dalam membuat sistem yang bersumber dari pakar tidaklah mudah, kita sering menghadapi suatu masalah yaitu ditemukan jawaban yang tidak memiliki kepastian penuh, maka sistem yang dibangun harus menggunakan metode *Certainty Factor* untuk meminimalisir ketidakpastian tersebut.

Dengan menggunakan sistem pakar identifikasi tipe kepribadian karyawan ini manajer dipermudah dalam mengidentikasi kepribadian karyawan, masing-masing karyawan melakukan tes kepribadian menggunakan sistem yang telah disediakan dengan mendaftar terlebih dahulu sebagai *user* kemudian *login* dan menjawab setiap pertanyaan dengan jujur sesuai realita ketika sudah selesai simpan data dan keluar dari sistem, hasil dari jawaban akan tersimpan di sistem secara otomatis dan manajer dapat melihat hasilnya untuk dijadikan referensi atau gambaran kepribadian karyawan baru yang sebelumnya belum mengetahui kepribadian karyawan tersebut baik itu sanguinis, korelis, phlegmatis atau melankolis dengan menjawab masing-masing 40 pertanyaan dari setiap kepribadian. Sistem tersebut dibangun menggunakan metode *Certainty Factor, Certainty Factor* adalah teori yang digunakan untuk mengakomodasi ketidakpastian pemikiran (*inexact reasoning*) seorang pakar. Hasil dari penelitian ini adalah dengan adanya sistem pakar ini manajer dapat melakukan identifikasi kepribadian karyawan secara efektif dan efisien, oleh karena itu sangat dibutuhkan sistem ini untuk mengidentifikasi tipe kepribadian karyawan menggunakan metode *Certainty Factor*.

2.1.2 Tinjauan Literatur 02

Kulit adalah organ penting pada tubuh yang luasnya sekitar $2m^2$ dan juga memiliki saraf peraba, selain itu kulit juga berfungsi sebagai pelindung pertahanan tubuh awal dari berbagai gangguan yang datang dari luar seperti virus berbahaya dan kuman serta bakteri penyebab penyakit. Penyakit kulit yang disebabkan oleh gangguan sistem imun adalah Inflamasi Dermatitis Imun, penyakit ini pada umumnya menyerang anak-anak dikarenakan sistem kekebalan tubuh pada anak masih lemah serta memiliki sensitifitas yang tinggi terhadap infeksi virus, lingkungan, udara dan bakteri. Kurangnya pengetahuan masyarakat serta tidak tercukupi tenaga ahli spesialis dibidang imunologi, mengakibatkan sulitnya penanganan terhadap pasien anak yang menderita penyakit Inflamasi Dermatitis Imun.

Melihat fenomena yang terjadi maka diperlukan sebuah sistem *E-Healthcare* yang mampu menerapkan metode *certainty factor* untuk mendiagnosa jenis penyakit Inflamasi Dermatitis Imun pada anak berdasarkan gejala klinis yang terjadi. Proses penerapannya dengan terlebih dahulu mengumpulkan basis pengetahuan, kemudian melakukan penelusuran inferensi *forward chaining* terhadap *rule-rule* yang ada dan selanjutnya melakukan proses perhitungan metode *certainty factor* untuk mengetahui probabilitas dan jenis penyakit Inflamasi Dermatitis Imun pada anak.

Hasil dari penelitian ini adalah sistem *E-Healthcare* yang digunakan berbasis web dengan menerapkan metode *certainty factor* dalam proses pendiagnosaan penyakit Inflamasi Dermatitis Imun pada anak sehingga sistem yang dirancang dapat diimplementasikan oelh seluruh masyarakat luas sebagai sarana konsultasi dan pengambilan diagnosa awal

2.1.2 Tinjauan Literatur 03

Berbagai macam jenis kepribadian anak haruslah diketahui oleh orangtua, hal ini dikarenakan masing-masing anak mempunyai karakteristik yang memang sudah dibawa semenjak dia lahir. Sebagai orangtua yang baik haruslah mengetahui bahwa anaknya mempunyai jenis kepribadian apa sehingga tidak akan terjadi salah pola

asuh. Setiap pola asuh berbeda-beda cara mengimplementasikannya dan itu tidak dapat disamakan atau diterapkan kesemua jenis kepribadian anak.

Fasilitas yang diberikan untuk *user* dan *administrator* memungkinkan baik *user* maupun *administrator* untuk menggunakan sistem ini sesuai kebutuhannya masing-masing. *User* diberi kemudahan dalam mengetahui informasi berbagai gangguan autisme dan gangguan psikologis anak dengan gejala-gejala klinisnya, serta konsultasi layaknya dengan seorang psikolog melalui beberapa pertanyaan yang harus dijawab *user* untuk mengetahui hasil diagnosanya. Sedangkan *administrator* dimudahkan dalam memanajemen sistem baik proses tambah, hapus maupun *update* data terbaru.

Oleh karena itu dengan adanya penelitian dari sistem pakar ini dapat membantu orangtua mengetahui jenis kepribadian anak tersebut sesuai dengan layaknya seorang pakar, dan pembangunan sebuah sistem berbasis pengetahuan dalam mendiagnosa autisme dan gangguan psikologis lainnya pada usia anak-anak yang dapat diakses melalui web. Hasil dari penelitian ini adalah dengan adanya aplikasi ini *user* dapat mengetahui jenis kepribadian anak berdasarkan gejala yang dialami. Berdasarkan jenis kepribadian anak yang terdiagnosa orangtua akan mendapatkan solusi pola asuh yang tepat untuk diterapkan dalam pola asuhnya.

2.1.3 Tinjauan Literatur 04

Karet (*Hevea brasiliensis*) termasuk dalam *Genus Hevea* dari familia *Euphorbiaceae*, merupakan pohon kayu tropis yang berasal dari hutan Amazon. Banyak artikel da penelitian yang menyebutkan bahwa hingga saat ini indonesia telah memiliki luas areal perkebunan karet terluas namun tidak didukung dengan produktivitas yang tinggi. Hal tersebut tentu saja akan menimbulkan kerugian ekonomi. Kerugian ekonomi dari budidaya karet disebabkan adanya serangan penyakit. Pengetahuan para petani tentang penanganan penyakit pada tanaman karet juga masih rendah. Namun, seiring dengan kemajuan teknologi dalam berbagi aspek kehidupan saat ini menjadikan para pengembang teknologi untuk membuat aplikasi baru yang lebih memudahkan masyarakat dalam memperoleh informasi.

Maka dari itu, perlu dibangun sebuah alat atau sistem yang praktis dan memiliki kemampuan layaknya seorang pakar dalam mendeteksi penyakit pada tanaman karet. Sistem tersebut adalah sistem pakar yang mengadopsi pengetahuan manusia ke dalam komputer agar agar dapat menyelesaikan masalah. Metode *Certainty Factor* sering digunakan dalam sistem pakar agar terlihat lebih natural. Penelitian ini menekankan pada dua aspek utama yaitu gejala-gejala dan jenis penyakit pada pohon karet. Hasil penelitian menunjukkan bahwa penerapan metode *Certainty Factor* pada penelitian ini memiliki akurasi sebesar 100% dari hasil diagnosa sistem pakar. Dengan adanya sistem pakar ini, diharapkan dapat membantu para petani untuk mengidentifikasi penyakit pada pohon karet.

2.1.5 Tinjauan Literatur 05

Pengguanaan *notebook* semakin marak dan dimensinya yang keil membuatnya mudah dibawa kemana-mana serta kinerjanya yang sudah sama dengan komputer *dekstop* membuat banyak orang lebih memilih *notebook* daripada komputer dekstop. Tidak jarang saat *notebook* mengalami masalah maka kegiatan yang dilakukan akan terhambat dan untuk mengatasinya harus memanggil teknisi atau membawa *notebook* tersebut ketempat perbaikan yang pasti akan memakan biaya dan waktu yang cukup banyak walaupun hanya untuk masalah sebenarnya bisa ditangani sendiri oleh karena itu diperlukan sebuah sistem pakar dapat menjadi alternatif dalam mengatasi permasalah tersebut. Karena hal itulah maka jika *notebook* mengalami kerusakan maka akan mengganggu setiap kegiatan menggunakan *notebook*.

Permasalahan dari penelitian ini adalah bagaimana membangun sebuah sistem pakar pendekteksi kerusakan *notebook* berbasis *mobile android* menggunakan metode *certainty factor*. Sistem pakar ini bertujuan untuk membantu mendiagnosa dan memecahkan masalah umum yang basa terjadi pada *notebook*. Sistem pakar ini dibuat menggunakan *metode certainty factor* dan *forward chaning*.

Hasil dari penelitian ini adalah sebuah sistem pakar berbasis *mobile* yang dapat memberikan informasi mengenai masalah, penyebab dan solusi yang dapat dilakukan untuk menangani kerusakan *notebook*. Sistem pakar telah diuji

menggunakan metode pengujian *blackbox testing* dan semua komponen sistem pakar dapat berjalan dengan baik. Setelah dilakukan validasi kepada pakar, sistem ini mendapat nilai validitas sebesar 90% dan dinyatakan layak untuk digunakan.

2.2 Pakar

Pakar merupakan orang yang memiliki pengetahuan, fakta dan teknik penalaran dalam memecahkan masalah yang biasanya hanya dapat dipecahkan oleh seorang pakar dalam bidang tertentu.

2.3 Sistem Pakar

Sistem pakar pertama kali dikembangkan oleh komunitas AI pada pertengahan tahun 1960. Sistem pakar yang muncul pertama kali adalah *General Purpose Problem Solver* (GPS). Sistem Pakar adalah suatu program komputer yang mensimulasikan penilaian dan perilaku manusia atau organisasi yang memiliki pengetahuan dan pengalaman ahli dalam bidang tertentu. Biasanya sistem itu mengandung basis pengetahuan, akumulasi pengalaman dan perangkat aturan untuk menerapkan kondisi setiap suatu situasi tertentu yang dijelaskan dalam suatu program. Sistem pakar yang canggih dapat ditingkatkan dengan penambahan basis pengetahuan atau seperangkat aturan. Dengan kata lain, ini adalah sistem berbasis software yang membuat atau mengevaluasi keputusan berdasarkan aturan yang ditetapkan dalam perangkat lunak (Josephine and Jeyabalaraja, 2012).

Pengetahuan yang digunakan dalam sistem pakar merupakan serangkaian informasi mengenai gejala-diagnosa, sebab-akibat, aksi-reaksi atau domain tertentu (misalnya, domain diagnosa medis). Secara umum, definisi tradisional sebuah program komputer biasa :

Algoritma + Struktur data = PROGRAM

Dalam sistem pakar, definisi berubah menjadi.

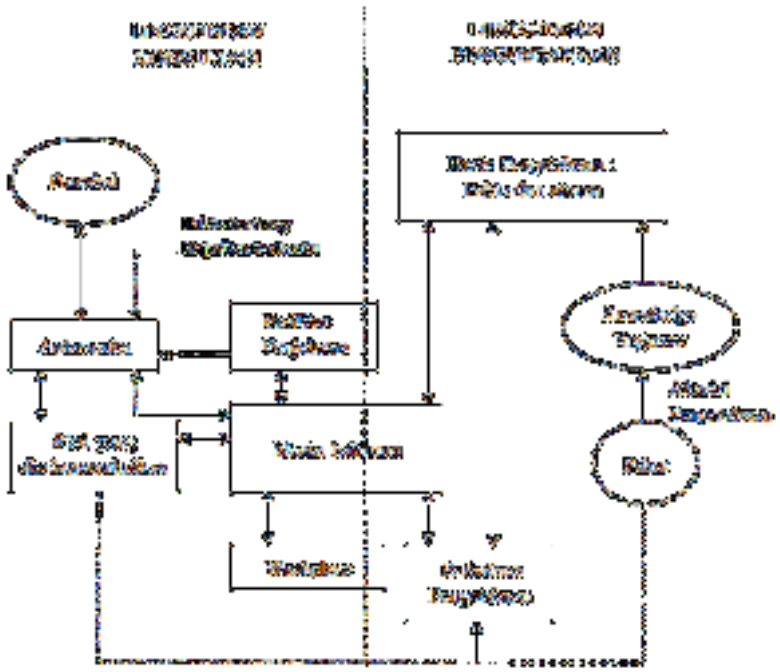
Mesin inferensi + Pengetahuan = Sistem Pakar

Dengan sistem pakar, masalah yang seharusnya hanya dapat diselesaikan oleh pakar/ahli dapat diselesaikan oleh orang biasa/awam. Sedangkan para ahli sistem

pakar membantu aktifitas mereka sebagai asisten yang seolah-olah sudah mempunyai banyak pengalaman.

2.3.1 Struktur Sistem Pakar

Sistem pakar disusun oleh dua bagian utama, yaitu lingkungan pengembangan(*development environment*) dan lingkungan konsultasi (*consultasion environment*) (Rohman and Fauzijah, 2018). Lingkungan pengembangan sistem pakar digunakan untuk memasukan pengetahuan pakar ke dalam lingkungan sistem pakar, sedangkan lingkungan konsultasi digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar. Komponen-komponen tersebut dapat dilihat pada gambar 2.1



Gambar 2.1 Arsitektur sistem pakar

Sumber: Turban (2011)

Komponen utama pada struktur sistem pakar menurut (Turban, 2011) meliputi:

1. Basis Pengetahuan (*Knowledge Base*)

Basis pengetahuan merupakan inti dari suatu sistem pakar, yaitu berupa representasi pengetahuan dari pakar. Basis pengetahuan tersusun atas fakta dan kaidah. Fakta adalah informasi tentang objek, peristiwa, atau situasi. Kaidah adalah cara untuk membangkitkan suatu fakta baru dari fakta yang sudah diketahui.

2. Mesin Inferensi (*Inference Engine*)

Mesin inferensi berperan sebagai otak dari sistem pakar, mesin inferensi berfungsi untuk memandu proses penalaran terhadap suatu kondisi, berdasarkan pada basis pengetahuan yang tersedia. Di dalam mesin inferensi terjadi proses untuk memanipulasi dan mengarahkan kaidah, model, dan fakta yang disimpan dalam basis pengetahuan dalam rangka mencapai solusi atau kesimpulan. Dalam prosesnya, mesin inferensi menggunakan strategi penalaran dan strategi pengendalian. Strategi penalaran terdiri dari strategi penalaran pasti (*Exact Reasoning*) dan strategi penalaran tak pasti (*Inexact Reasoning*). *Exact reasoning* akan dilakukan jika semua data yang dibutuhkan untuk menarik suatu kesimpulan tersedia, sedangkan *inexact reasoning* dilakukan pada keadaan sebaliknya. Strategi pengendalian berfungsi sebagai panduan arah dalam melakukan proses penalaran. Terdapat tiga teknik pengendalian yang sering digunakan, yaitu *forward chaining, backward chaining*, dan gabungan dari kedua teknik pengendalian tersebut.

3. Basis Data (*Data Base*)

Basis data terdiri atas semua fakta yang diperlukan, dimana fakta-fakta tersebut digunakan untuk memenuhi kondisi dari kaidah-kaidah dalam sistem. Basis data menyimpan semua fakta, baik fakta awal pada saat sistem mulai beroperasi, maupun fakta-fakta yang diperoleh pada saat proses penarikan kesimpulan sedang dilaksanakan. Basis data digunakan untuk menyimpan data hasil observasi dan data lain yang dibutuhkan selama pemprosesan.

4. Antarmuka Pemakai (*User Interface*)

Fasilitas ini digunakan sebagai perantara komunikasi antara pemakai dengan komputer. Teknik Representasi Pengetahuan adalah suatu teknik untuk merepresentasikan basis pengetahuan yang diperoleh ke dalam suatu skema/diagram tertentu sehingga dapat diketahui relasi/keterhubungan antara suatu data dengan data

yang lain. Teknik ini membantu *knowledge engineer* dalam memahami struktur pengetahuan yang akan dibuat sistem pakarnya. Terdapat beberapa teknik representasi pengetahuan yang biasa digunakan dalam pengembangan suatu sistem pakar, yaitu :

1. *Rule-Based Knowledge*

   Pengetahuan direpresentasikan dalam suatu bentuk fakta (*facts*) dan aturan (*rules*). Bentuk representasi ini terdiri atas premise dan kesimpulan.

2. *Frame-Based Knowledge*

   Pengetahuan direpresentasikan dalam suatu bentuk hirarki atau jaringan *frame.*

3. *Object-Based Knowledge*

   Pengetahuan direpresentasikan sebagai jaringan dari obyek-obyek. Obyek adalah elemen data yang terdiri dari data dan metoda (proses).

4. *Case-Base Reasoning*

   Pengetahuan direpresentasikan dalam bentuk kesimpulan kasus (*cases*). *Inferencing* dengan *Rule* : *Forward* dan *Backward Chaining* Inferensi dengan *rules* merupakan implementasi dari modus ponen, yang direfleksikan dalam mekanisme *search* (pencarian). Dapat pula mengecek semua *rule* pada *knowledge base* dalam arah *forward* maupun *backward*. Proses pencarian berlanjut sampai tidak ada *rule* yang dapat digunakan atau sampai sebuah tujuan (*goal*) tercapai. Ada dua metode *inferencing* dengan *rules*, yaitu *forward chaining* atau data-*driven* dan *backward chaining* atau *goal-driven*.

   1. *Backward chaining*

      Menggunakan pendekatan *goal-driven*, dimulai dari ekspektasi apa yang diinginkan terjadi (hipotesis), kemudian mengecek pada sebab-sebab yang mendukung (ataupun kontradiktif) dari ekspektasi tersebut. Jika suatu aplikasi menghasilkan *tree* yang sempit dan cukup dalam, maka gunakan *backward chaining*.

   2. *Forward chaining*

Forward chaining merupakan grup dari multiple inferensi yang melakukan pencarian dari suatu masalah kepada solusinya. Jika klausa premis sesuai dengan situasi (bernilai *TRUE*), maka proses akan meng-assert konklusi. *Forward chaining* adalah data-*driven* karena inferensi dimulai dengan informasi yang tersedia dan baru konklusi diperoleh.

Jika suatu aplikasi menghasilkan *tree* yang lebar dan tidak dalam, maka gunakan *forward chaining*.

2.3.2 Kelebihan dan Kelemahan Sistem pakar

Sistem pakar (*expert system*) merupakan paket perangkat lunak atau paket program komputer yang ditujukan sebagai penyedia nasihat dan sarana bantu dalam memecahkan masalah di bidang-bidang spesialisasi tertentu seperti sains, perekayasaan, matematika, kedokteran, pendidikan dan sebagainya.

Ada beberapa kelebihan sistem pakar, diantaranya:

1. Menghimpun data dalam jumlah yang sangat besar
2. Menyimpan data tersebut untuk jangka waktu yang panjang dalam suatu bentuk tertentu
3. Mengerjakan perhitungan secara cepat dan tepat dan tanpa jemu mencari kembali data yang tersimpan dengan kecepatan tinggi.

Selain memiliki kelebihan, sistem pakar juga memiliki kelemahan diantaranya adalah:

1. Masalah dalam mendapatkan pengetahuan, dimana pengetahuan tidak selalu bisa didapatkan dengan mudah, karena kadangkala pakar dari masalah yang kita buat tidak ada, dan kalaupun ada kadang-kadang pendekatan yang dimiliki oleh pakar berbeda-beda.
2. Untuk membuat suatu sistem pakar yang benar-benar berkualitas tinggi sangatlah sulit dan memerlukan biaya yang sangat besar untuk pengembangan dan

3. Sistem pakar tidaklah 100% bernilai benar. Oleh karena itu perlu diuji ulang secara teliti sebelum digunakan. Dalam hal ini peran manusia tetap merupakan faktor dominan.

2.4 Kepribadian

2.4.1 Pengertian Kepribadian

Kepribadian merupakan kesuluruhan perasaan, ekpresi, tempramen, ciri-ciri khas dan perilaku seseorang. Sikap perasaan ekspresi dan temperamen ini akan terwujud dalam tindakan seseorang jika dihadapkan pada situasi tertentu. Kepribadian berperan penting dalam kehidupan yaitu menggambarkan perilaku, watak, atau pribadi seseorang. Kepribadian mencakup gaya, sikap yang berperan aktif dalam menentukan tingkah laku yang menyebabkan seseorang memiliki suatu perilaku konsisten (Gortap dan Mufria, 2020). Terjadinya Interaksi psiko-fisik mengarahkan tingkah laku manusia. Maksud dinamis pada pengertian tersebut adalah perilaku mungkin saja berubah-ubah melalui proses pembelajaran atau melalui pengalaman-pengalaman, *reward, punishment*, pendidikan dsb. Kepribadian adalah ciri, karakteristik, gaya atau sifat-sifat yang memang khas dikaitkan dengan diri kita. Dapat dikatakan bahwa kepribadian itu bersumber dari bentukan-bentukan yang kita terima dari lingkungan, misalnya bentukan dari keluarga pada masa kecil kita dan juga bawaan-bawaan yang dibawa sejak lahir. Jadi yang disebut kepribadian itu sebetulnya adalah campuran dari hal-hal yang bersifat psikologis, kejiwaan dan juga yang bersifat fisik.

2.4.2 Penggolongan Manusia Berdasarkan Kepribadiannya

Penggolongan manusia berdasarkan beberapa kriteria tertentu sangatlah sulit, kendalanya terletak pada heterogenitas dan keunikan sifat manusia. Tidak ada satu manusiapun yang dapat dianggap memiliki sifat yang sama kemudian dikelompokkan berdasarkan sifat itu. Ilmu pengetahuan hanya bisa melakukan pendekatan agar beberapa ciri yang agak mirip dikelompokkan menjadi beberapa kelompok kepribadian(Gortap dan Mufria, 2020). Kepribadian adalah ciri, karakteristik, gaya atau sifat-sifat yang memang khas dikaitkan dengan diri kita. Dapat dikatakan bahwa

kepribadian itu bersumber dari bentukan-bentukan yang kita terima dari lingkungan, misalnya bentukan dari keluarga pada masa kecil kita dan juga bawaan-bawaan yang dibawa sejak lahir. Jadi yang disebut kepribadian itu sebetulnya adalah campuran dari hal-hal yang bersifat psikologis, kejiwaan dan juga yang bersifat fisik. Dalam ilmu keperawatan hal ini dikenal dengan istilah *holistic* (*Biopsikososiospiritual*).

Berdasarkan aspek biologis Hipocrates membagi kepribadian menjadi 4 kelompok besar yaitu :

a. Sanguinis

Sanguinis adalah orang yang gembira, yang senang hatinya, mudah untuk membuat orang tertawa, dan bisa memberi semangat pada orang lain. Tapi kelemahannya adalah dia cenderung impulsive, yaitu orang yang bertindak sesuai emosi atau keinginannya.

b. Phlegmatis

Tipe phlegmatis adalah orang yang cenderung tenang, dari luar cenderung tidak beremosi, tidak menampakkan perasaan sedih atau senang. Naik turun emosinya itu tidak nampak dengan jelas. Orang ini memang cenderung bisa menguasai dirinya dengan cukup baik, ia intorspektif sekali, memikirkan ke dalam, bisa melihat, menatap dan memikirkan masalah-masalah yang terjadi di sekitarnya. Kelemahan orang phlegmatis adalah ia cenderung mau ambil mudahnya, tidak mau susah, sehingga suka mengambil jalan pintas yang paling mudah dan gampang.

c. Melankolis

Tipe melankolis adalah orang yang terobsesi dengan karya yang paling bagus, yang paling sempurna dan dia memang adalah seseorang yang mengerti estetika keindahan hidup ini. Perasaannya sangat kuat, sangat sensitif maka kita bisa menyimpulkan bahwa cukup banyak seniman yang memang berdarah melankolis. Kelemahan orang melankolis, ia mudah sekali dikuasai oleh perasaan dan cukup sering perasaan yang mendasari hidupnya sehari-hari adalah perasaan murung.

d. Koleris

Seseorang yang koleris adalah seseorang yang dikatakan berorientasi pada pekerjaan dan tugas, dia adalah seseorang yang mempunyai disiplin kerja yang sangat tinggi. Kelebihannya adalah dia bisa melaksanakan tugas dengan setia dan akan bertanggung jawab dengan tugas yang diembannya. Kelemahan orang yang berciri koleris adalah kurangnya kemampuan untuk bisa merasakan perasaan orang lain (empati), belas kasihannya terhadap penderitaan orang lain juga agak minim, karena perasaannya kurang bermain.

Faktor-faktor yang mempengaruhi kepribadian :

1. Faktor genetik

   Dari beberapa penelitian bayi-bayi baru lahir mempunyai temperamen yang berbeda, Perbedaan ini lebih jelas terlihat pada usia 3 bulan. Perbedaan meliputi: tingkat aktivitas, rentang atensi, adaptabilitas pada perubahan lingkungan. Sedangkan menurut hasil riset tahun 2007 kazuo Murakami di Jepang menunjukan bahwa gen Dorman bisa distimulasi dan diaktivasi pada diri seseorang dalam bentuk potensi baik dan potensi buruk.

2. Faktor lingkungan

   Perlekatan (*attachment*): kecenderungan bayi untuk mencari kedekatan dengan pengasuhnya dan untuk merasa lebih aman dengan kehadiran pengasuhnya dapat mempengaruhi kepribadian.

3. Faktor stimulasi gen dan cara berpikir

   Kepribadian sepenuhnya dikendalikan oleh gen yang bersipat Dorman (tidur) atau tidak aktip dan yang bersipat aktip. Bila kita sering menyalakan gen yang tidur dengan cara positif *thinking* maka kepribadian dan nasib kita akan lebih baik. Jadi genetik bukan sesuatu yang kaku, permanen dan tidak dapat dirubah.

2.5 Pengertian *Certainty Factor*

Certainty Factor (CF) adalah metode untuk mengamsumsikan derajat keyakinan seorang pakar terhadap suatu data. *Certainty Factor* memperkenalkan konsep keyakinan dan ketidak yakinan (Suwarno, Husin and Zenni, 2019). Seorang pakar (misalnya dokter) sering menganalisis informasi yang ada dengan ungkapan dengan ketidakpastian, untuk mengakomodasi hal ini kita menggunakan CF guna menggambarkan tingkat keyakinan pakar terhadap masalah yang sedang dihadapi. Dalam mengekspresikan derajat kepastian, CF untuk mengasumsikan derajat kepastian seorang pakar terhadap suatu data (Sutojo, Mulyanto, and Suhartono, 2011).

Menurut Giarrantano dan Riley metode CF dapat didefinisikan sebagai berikut :

$$CF[H.E] = MB[H.E] - MD[H.E]$$

$$CF_{gejala} = CF_{[user]} * CF_{[pakar]}$$

Keterangan

CF = *Certainty Factor* (faktor kepastian) dalam hipotesis H yang dipengaruhi oleh fakta E.

MB = *Measure of Belief* (tingkat keyakinan), adalah ukuran kenaikan dari kepercayaan hipotesis H dipengaruhi oleh fakta E.

MD = *Measure of Disbelief* (tingkat tidak keyakinan), adalah kenaikan dari ketidak percayaan hipotesis H terhadap E.

E = *Evidence* (pristiwa atau fakta)

H = Hipotesis (dugaan)

$$CF_{combine} = CF_1 + CF_2 * (1 - CF_1)$$

Dimana CF_1 dan CF_2 memiliki hipotesis yang sama:

CF_1 = nilai *Certainty Factor evidence* 1 terhadap hipotesis

CF_2 = nilai *Certainty Factor evidence* 2 terhadap hipotesis

Tabel 2.2 Representasi Nilai CF

Uncertain Term	CF
Pasti Tidak	-1.0
Hampir Pasti Tidak	-0.8
Kemungkinan Besar Tidak	-0.6
Mungkin Tidak	-0.4
Tidak Tahu	-0.2 sampai 0.2
Mungkin	0.4
Kemungkinan Besar	0.6
Hampir Pasti	0.8
Pasti	1.0

Sumber: Puspita sari (2012)

2.6 Kelebihan dan Kekurangan Metode *Certainty Factor*

Menurut Sutojo, Mulyanto and Suhartono (2011) terdapat kelebihan dan kelemahan pada *Certainty Factor,* yaitu

2.6.1 Kelebihan metode *Certainty Factor*

1. Metode ini cocok dipakai pada sistem pakar yang mengandung ketidakpastian.
2. Metode ini dapat menjaga keakuratan data karena sistem perhitungan metode ini hanya dapat mengolah dua data dalam satu kali perhitungan.

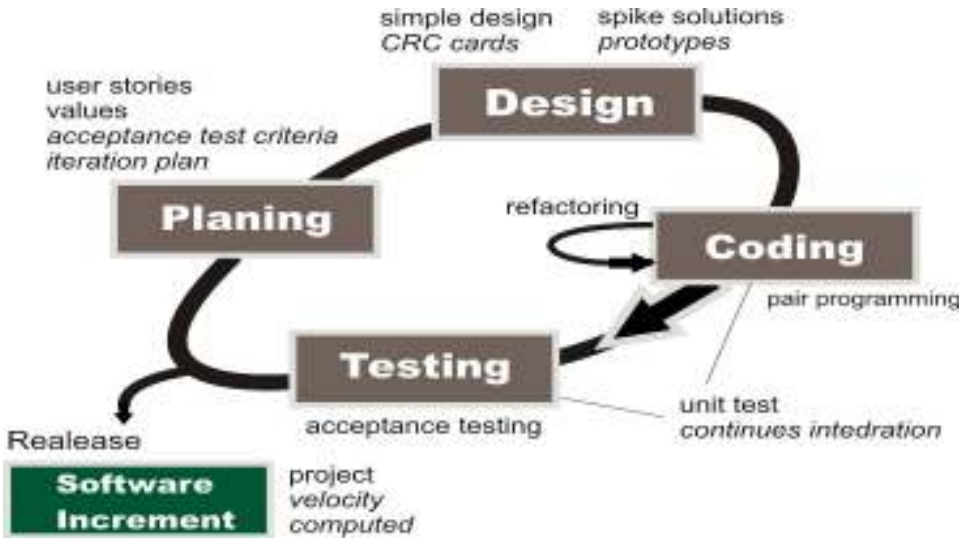
2.6.2 Kekurangan metode *Certainty Factor*

1. Pemodelan ketidakpastian yang menggunakan perhitungan metode *Certainty Factor* biasanya masih diperdebatkan.
2. Metode *Certainty Factor* hanya dapat mengolah dua data saja, apabila ada data yang lebih dari dua maka perlu beberapa kali pengolahan data.

2.7 Metode Pengembangan Sistem

Pengembangan sistem berarti menyusun sistem baru untuk mengganti sistem lama secara keseluruhan atau memperbaiki bagian-bagian tertentu dalam sistem lama. Metode yang digunakan dalam pengembangan sistem yaitu dengan siklus klasik atau air terjun dengan tahapan - tahapan yang terdiri dari survei sistem, analisis sistem, pembuatan sistem, implementasi sistem, pengujian dan pemeliharaan sistem. Dalam metode air terjun setiap tahun harus diselesaikan terlebih dahulu secara penuh sebelum diteruskan ke tahap berikutnya untuk menghindari pengulangan tahapan (Pressman, 2012).

ExtremeProgramming(XP) adalah metodologi pengembangan perangkat lunak yang ditujukan untuk meningkatkan kualitas perangkat lunak dan tanggap terhadap perubahan kebutuhan pelanggan. Jenis pengembangan perangkat lunak semacam ini dimaksudkan untuk meningkatkan produktivitas dan memperkenalkan pos pemeriksaan dimana persyaratan pelanggan baru dapat diadopsi. Tahapan-tahapan dari *Extreme Programming* terdiri dari *planning* seperti memahami kriteria pengguna dan perencanaan pengembangan, *designing* seperti perancangan *prototype* dan tampilan, *coding* termasuk pengintegrasian, dan yang terakhir adalah *testing* (Pressman, 2012)



Gambar 2.2 Model *Extreme Programming (XP)*

Sumber : (Pressman, 2012)

Dibawah ini adalah penjelasan tahapan *Extreme Programming* yaitu :

1. *Planning* (Perencanaan)

   Kegiatan Perencanaan (disebut juga *planning game*) biasanya dimulai dengan mendengarkan suatu kegiatan yang bertujuan mengumpulkan kebutuhan-kebutuhan untuk memahami konteks bisnis dan perlunya keluaran-keluaran (*output*), fungsi utama, dan *fungsionalitas.*

   Pada perencanaan terdapat *user stories values* yaitu story dengan *value* tertinggi akan dipindahkan dari jadwal dan diimplementasikan pertama, *acceptance test criteria interation plan* melakukan perhitungan kecepatan project selama *development, customer* dapat menambah *story*, merubah *value*, membagi *story* atau menghapusnya.

2. *Design* (Perancangan)

   Perancangan yang simple, menarik, dan sederhana selalu memberikan hasil yang lebih disukai daripada gambaran-gambaran yang lebih kompleks. Perancangan XP memberikan panduan implementasi untuk suatu cerita ketika ditulis, tidak kurang, tidak lebih.

   Terdapat *simple design CRC Cards* untuk mengenali dan mengatur *object oriented class* sesuai dengan *software increment* dan *spike solutions prototypes* melakukan spesifikasi solusi dari *object oriented class.*

3. *Coding* (Pengkodean)

   Pengkodean ini dilanjutkan setelah cerita yang telah dikembangkan dan rancangan yang telah dilakukan oleh tim perangkat lunak. Pengkodean ini tidak langsung mengarah ke kode-kode program. Tim akan mengembangkan serangkaian unit pengujian lalu beralih ke pengkodean.

   Pada tahapan *pair programming* melakukan kerja sama untuk membuat code dari satu story. Dan *refactoring* adalah proses restrukturisasi kode program komputer yang ada tanpa mengubah perilaku eksternalnya.

4. *Pengujian* (Pengujian)

   Unit pengujian yang harus dibuat dan kemudian dijalankan menggunakan kerangka kerja yang memungkinkan mereka untuk diotomatisasi sehingga dapat

dijialankan dengan mudah dan dapat dijalankan berulang kali.

Pada tahapan pengujian yaitu *unit test continuos integration* yaitu tahapan pengujian kode yang diintegrasikan dengan kerja lainnya dengan pengujian yang dilakukan oleh *customer* dan *focus* pada keseluruhan dan fungsional sistem, dan *acceptance testing* yaitu pengujian yang dilakukan *customer* stories yang akan diimplementasikan sebagai bagian dari *software realease.*

Selanjutnya terdapat tahapan *software increment project velocity computed* yaitu tahapan yang telah diimplementasikan dari *software realease* yang nantinya akan diterapkan dalam suatu sistem.

2.8 *Unified Modeling Language* (UML)

Menurut (Rosa and Shalahudin, 2018)"UML *(Unified Modeling Language)* adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk medefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemograman berorientasi objek".

UML dikelompokkan menjadi 3 kategori, struktur *diagrams, behaviour diagrams, dan Interaction diagrams*. Struktur *diagrams* merupakan kumpulan dari diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. *Behaviour diagrams* digunakan untuk menggambarkan perilaku sistem atau rangkaian perubahan yang terjadi pada sistem. Dan *Interaction diagrams* menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

Tabel 2.3 Ilustrasi Pembagian Kategori dalam UML

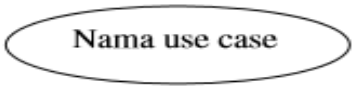
UML Category	Spesific UML Detail
Structure Diagram	*Class Diagram*
	Object Diagram
	Component Diagram
	Composite Structure Diagram
	Package Diagram

UML Category	Spesific UML Detail
	Deployment Diagram
Behavior Diagram	*Use Case Diagram*
	Activity Diagram
Interaction Diagram	*State Machine Diagram*
	Squence Diagram
	Communication Diagram
	Timming Diagram
	Interaction Overview Diagram

2.8.1 *Use Case* Diagram

Merupakan permodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut (Rosa and Shalahudin, 2018).

Tabel 2.4 Simbol Use Case Diagram

No	Simbol	Keterangan
1	*Use Case* 	Fungsionalitas yang disediakan sistem sebagai unit- unit yang saling bertukar pesan antar unit atau *aktor*, biasanya dinyatakan dengan menggunakan kata kerja diawali frase nama *use case*.
2	Aktor/ *actor*	Orang, proses, atau sistem lain yang berinteraksi

		dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda diawali frase nama aktor.
3	Assosiasi/ *association* 	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4	Exstensi/ *extend* <<extend>> 	Relasi *use case* tambahan kesebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan misal:

		 Arah panah mengarah pada *use case* yang ditambahkan, biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
5	Generalisasi/ *generalization* 	Hubungan generalisasi dan spesialisasi (umum- khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:  Arah panah mengarah pada

		use case yang menjadi generalisasinya (Umum).
6	Menggunakan/ *include*/ *uses*   	Relasi *use case* tambahan sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini. Ada dua sudut pandang cukup besar mengenai include di *use case*: - *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat use case tambahan dijalankan, misalnya pada kasus berikut:  - *Include* berarti *use case* yang ditambahkan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan di jalankan, misal pada kasus berikut:

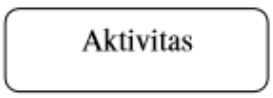
		 Kedua interpretasi diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangandan interpretasiyang dibutuhkan.

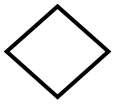
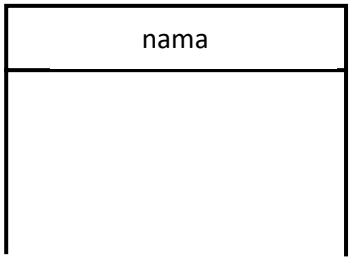
Sumber: (Rosa and Shalahudin, 2018)

2.8.2 *Activity Diagram*

Activity diagram menggambarkan *Work Flow*(Aliran kerja) aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem (Rosa and Shalahudin, 2018). Berikut ini adalah simbol- simbol yang ada pada diagram aktivitas:

Tabel 2.5 Simbol *Activity Diagram*

Simbol	Keterangan
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas yang memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas diawali dengan kata kerja.

Simbol	Keterangan
Percabangan/ *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan/ *join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Sumber: (A.S dan Shalahuddin, 2018)

2.8.3 *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas- kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi (A.S dan Shalahuddin, 2018). Sebuah kelas diagram terdiri dari sebuah kelas yang dihubungkan dengan garis yang menunjukkan hubungan antar kelas.

Tabel 2.6 Simbol C*lass Diagram*

No.	Simbol	Keterangan
1	Kelas/ *Class* **Nama_kelas** +atribut +operasi()	Kelas pada struktur sistem.
2	Antar muka/ *Interface* **Nama_interface**	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3	Asosiasi/ *Association*	Realasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
4	Asosiasi	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
5	*Generalisasi*	Relasi antar kelas dengan makna generalisasi- spesialisasi (Umum khusus).
6	Kebergantungan atau *Dependensy*	Relasi antar kelas dengan makna kebergantungan antar kelas.
7	Agregasi/ *Agregation*	Relasi antar kelas dengan makna semua bagian (*Whole-*

No.	Simbol	Keterangan
		part).

Sumber: (Rosa and Shalahudin, 2018)

2.9. XAMPP

Menurut Nugroho (2015), "XAMPP adalah paket program web lengkap yang dapat Anda pakai untuk belajar pemrograman web, khususnya PHP dan MySQL".

Menurut Buana (2014), "XAMPP adalah perangkat lunak *opensource* yang diunggah secara geratis dan bisa dijalankan di semua semua operasi seperti *windows, linux, solaris, dan mac"*.

2.10. *Sublime Text*

Sublime Text adalah aplikasi editor untuk kode dan teks yang dapat berjalan diberbagai platform *operating system* dengan menggunakan teknologi Phyton API. Terciptanya aplikasi ini terinspirasi dari aplikasi Vim, Aplikasi ini sangatlah fleksibel dan *powerfull*. Fungsionalitas dari aplikasi ini dapat dikembangkan dengan menggunakan *sublime-packages*. *Sublime Text* bukanlah aplikasi *open source* dan juga aplikasi yang dapat digunakan dan didapatkan secara gratis, akan tetapi beberapa fitur pengembangan fungsionalitas (*packages*) dari aplikasi ini merupakan hasil dari temuan dan mendapat dukungan penuh dari komunitas serta memiliki lisensi aplikasi gratis.

Sublime Text mendukung berbagai bahasa pemrograman dan mampu menyajikan fitur *syntax highlight* hampir di semua bahasa pemrogramman yang didukung ataupun dikembangkan oleh komunitas seperti; C, C++, C#, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, MATLAB, OCaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textile and XML. Biasanya bagi bahasa pemrograman yang didukung ataupun belum terdukung secara *default* dapat lebih dimaksimalkan atau didukung dengan menggunakan *add-ons* yang bisa di*download* sesuai kebutuhan *user*(Sadeli, 2014).

2.11. PHPMyAdmin

PhpMyAdmin adalah *tools* yang dapat digunakan dengan mudah untuk memanajemen database MySQL secara visual dan Server MySQl, sehingga kita tidak perlu lagi harus menulis query SQL setiap akan melakukan perintah operasi database". *Tools* ini cukup populer, Anda dapat mendapatkan fasilitas ini ketika menginstal paket triad phpMyAdmin, karena termasuk dalam xampp yang sudah di instal (Nugroho, 2015)

PhpMyAdmin adalah salah satu aplikasi yang digunakan untuk memudahkan dalam melakukan pengelolaan *database* MySQL. phpMyAdmin merupakan aplikasi web yang bersifat *opensource* (Buana, 2014).

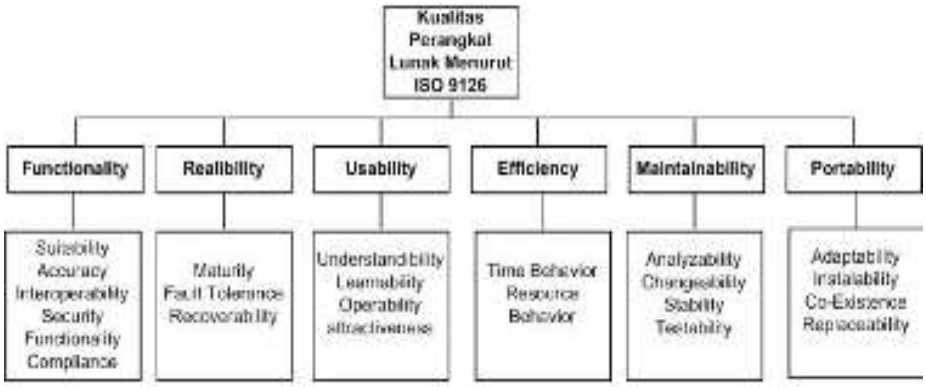
2.12. ISO 9126

Salah satu tolak ukur kualitas perangkat lunak adalah ISO 9126, yang dibuat oleh *International Organization for Standardization (ISO)* dan *International Electrotechnical Commission (IEC)*. ISO 9126 mendefinisikan kualitas produk perangkat lunak, model, karakteristik mutu, dan metrik terkait yang digunakan untuk mengevaluasi dan menetapkan kualitas sebuah produk software. Standar ISO 9126 telah dikembangkan dalam usaha untuk mengidentifikasi atribut-atribut kunci kualitas untuk perangkat lunak komputer. Menurut (Abran *et al.*, 2018), ISO 9126 adalah standar internasional yang diterbitkan oleh ISO untuk evaluasi kualitas perangkat lunak dan merupakan pengembangan dari ISO 9001. Faktor kualitas menurut ISO 9126 meliputi enam karakteristik kualitas sebagai berikut:

1) *Functionalityity* (Fungsionalitas). Kemampuan perangkat lunak untuk menyediakan fungsi sesuai kebutuhan pengguna, ketika digunakan dalam kondisi tertentu.
2) *Reliability* (Kehandalan). Kemampuan perangkat lunak untuk mempertahankan tingkat kinerja tertentu, ketika digunakan dalam kondisi tertentu.
3) *Usability* (Kebergunaan). Kemampuan perangkat lunak untuk dipahami, dipelajari, digunakan, dan menarik bagi pengguna, ketika digunakan dalam kondisi tertentu.

4) *Efficiency* (Efisiensi). Kemampuan perangkat lunak untuk memberikan kinerja yang sesuai dan relatif terhadap jumlah sumber daya yang digunakan pada saat keadaan tersebut.
5) *Maintainability* (Pemeliharaan). Kemampuan perangkat lunak untuk dimodifikasi. Modifikasi meliputi koreksi, perbaikan atau adaptasi terhadap perubahan lingkungan, persyaratan, dan spesifikasi fungsional.
6) *Portability* (Portabilitas). Kemampuan perangkat lunak untuk ditransfer dari satu lingkungan ke lingkungan lain.

Karakteristik Kualitas Perangkat Lunak Model ISO 9126



Gambar 2.3 Model Kualitas Perangkat Lunak Model ISO 9126
Sumber: Al-Qutaish (2010)

Masing-masing karakteristik kualitas perangkat lunak model ISO 9126 dibagi menjadi beberapa sub-karakteristik kualitas. Tabel karakteristik Kualitas Perangkat Lunak Model ISO 9126 dapat dilihat pada tabel 2.7.

Tabel 2.7 Karakteristik ISO 9126

Karakteristik	Sub Karakteristik	Deskripsi
Functionality	*Suitability*	Kemampuan perangkat lunak untuk menyediakan serangkaian fungsi yang sesuai untuk tugas-tugas tertentu dan tujuan pengguna.
	Accuracy	Kemampuan perangkat lunak dalam memberikan hasil yang presisi dan benar sesuai dengan kebutuhan.
	Security	Kemampuan perangkat lunak untuk mencegah *Access* yang tidak diinginkan, menghadapi penyusup (*hacker*) maupun otorisasi dalam modifikasi data.
	Interoperabiy	Kemampuan perangkat lunak untuk berinteraksi dengan satu atau lebih sistem tertentu.
	Compliance	Kemampuan perangkat lunak dalam memenuhi standar dan kebutuhan sesuai peraturan yang berlaku.
Reliability	*Maturity*	Kemampuan perangkat lunak untuk menghindari kegagalan sebagai akibat dari kesalahan dalam perangkat lunak.
	Fault tolerance	Kemampuan perangkat lunak untuk mempertahankan kinerjanya jika terjadi kesalahan perangkat lunak
	Recoverabiliy	Kemampuan perangkat lunak untuk

		membangun kembali tingkat kinerja ketika terjadi kegagalan sistem, termasuk data dan koneksi jaringan.
Usability	*Understandibility*	Kemampuan perangkat lunak dalam kemudahan untuk dipahami.
	Learnability	Kemampuan perangkat lunak dalam kemudahan untuk dipelajari.
	Operability	Kemampuan perangkat lunak dalam kemudahan untuk dioperasikan.
	Attractiveness	Kemampuan perangkat lunak dalam menarik pengguna.
Efficiency	*Time behavior*	Kemampuan perangkat lunak dalam memberikan respon dan waktu pengolahan yang sesuai saat melakukan fungsinya.
	Resource behavior	Kemampuan perangkat lunak dalam menggunakan sumber daya yang dimilikinya ketika melakukan fungsi yang ditentukan.
Maintainabiliy	*Analyzability*	Kemampuan perangkat lunak dalam mendiagnosis kekurangan atau penyebab kegagalan.
	Changeabiliy	Kemampuan perangkat lunak untuk dimodifikasi tertentu.
	Stability	Kemampuan perangkat lunak untuk meminimalkan efek tak terduga dari modifikasi perangkat lunak.
	Testability	Kemampuan perangkat lunak untuk dimodifikasi dan divalidasi

		perangkat lunak lain.
Portability	*Adaptability*	Kemampuan perangkat lunak untuk diadaptasikan pada lingkungan yang berbeda-beda.
	Instalability	Kemampuan perangkat lunak untuk diinstal dalam lingkungan yang berbeda-beda.
	Coexistence	Kemampuan perangkat lunak untuk berdampingan dengan perangkat lunak lainnya dalam satu lingkungan dengan berbagi sumber daya.
	Replaceabiliy	Kemampuan perangkat lunak untuk digunakan sebagai sebagai pengganti perangkat lunak lainnya.

Sumber: (Al-Qutaish 2010, 172-173)

Adapun alasan penggunaan ISO 9126 karena ISO sudah berstandar *International Organization for Standardization (ISO)* dan *International Electrotechnical Commission (IEC)*. Kualitas produk perangkat lunak ISO 9126 memiliki enam karakteristik pendukung yang dapat digunakan sebagai acuan dalam menilai maupun memberikan masukan terhadap kualitas perangkat lunak yang akan dibangun yang akan menghasilkan nilai uji yang terukur.

Dalam pengujian perangkat lunak ini penulis menggunakan suatu metode pengujian yang berfokus pada persyaratan fungsional dan kegunaan perangkat lunak dari sistem yang akan dibangun. Pengujian sistem ini akan diuji oleh pelanggan, admin perusahaan, dan bagian marketing dengan menggunakan metode yang diambil yaitu metode pengujian *ISO 9*126 berdasarkan *Functionality, Usability*. Dan *Effeciency.*

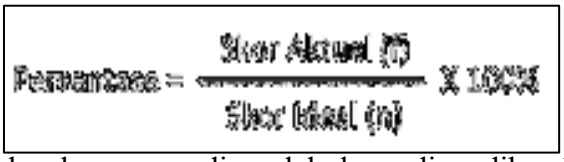
2.13. Skala Pengukuran

Skala pengukuran yang digunakan adalah skala Likert, skala yang didasarkan pada penjumlahan sikap responden dalam merespon pernyataan berkaitan indikator-indikator suatu konsep atau variabel yang sedang diukur(Sugiyono, 2017). Skala Likert umumnya menggunakan lima titik denganlabel netral pada posisi tengah (ketiga). Skala Likert apat dilihat pada Tabel 2.8

Tabel 2.8 Skala Likert

Jawaban	Skor
Sangat Setuju	5
Setuju	4
Netral	3
Tidak Setuju	2
Sangat Tidak Setuju	1

Hasil penilaian responden akan dihitung *persentase* kelayakannya dengan menggunakan perhitungan (Sugiyono, 2017), dapat dilihat dibawah ini :



Persentase kelayakan yang diperoleh kemudian dibandingkan dengan Tabel konversi yang berpedoman pada acuan konversi nilai (Sugiyono, 2017), dapat dilihat pada Tabel 2.9

Tabel 2.9 Skala Konversi Nilai

Persentase Pencapaian (%)	Interpretasi
$90 \leq x$	Sangat Baik
$80 \leq x < 90$	Baik
$70 \leq x < 80$	Cukup
$60 \leq x < 70$	Kurang
$X < 60$	Sangat Kurang

Keterangan: x = persentase hasil pengujian.