BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Penelitian yang telah ada digunakan sebagai sumber referensi dan bahan acuan baik kelebihan ataupun kekurangan dari sisi sistem. Beberapa penelitian terkait yang membahas sistem pakar dengan metode *Certainty Factor* (CF) adalah sebagai berikut :

Penelitian terdahulu metode Certainty Factor menurut Stephanie Halim , Seng Hansun pada jurnal yang berjudul "Penerapan Metode *Certainty Factor* dalam Sistem Pakar Pendeteksi Resiko Osteoporosis dan Osteoarthritis" Menyediakan sebuah aplikasi sistem pakar mendeteksi resiko penyakit osteoporosis dan osteoarthritis. Dengan presentasi keakuratan 80% menjadi bukti nyata bahwa diagnosa gejala setiap pakar mempengaruhi tingkat keakuratan sistem [6].

Penelitian lainnya dilakukan oleh Mohammad Arifin, Slamin, Windi Eka Yulia Retnani dengan judul "Penerapan *Metode Certainty Factor* Untuk Sistem Pakar Diagnosis Hama Dan Penyakit Pada Tanaman Tembakau". Website ini dapat mendiagnosis hama dan penyakit pada tanaman tembakau dengan Proses konsultasi yang dilakukan user untuk mendapatkan hasil data hama atau penyakit dalam persentasenya, nilai tertinggi yang dicapai ialah 99.985729744% menjadi bukti bahwa metode certainty factor dalam penerapan sistem pakar masih sangat cocok [3].

Penelitian yang serupa juga dilakukan oleh Khairina Eka Setyaputri , Abdul Fadlil , dan Sunardi dengan judul "Analisis Metode *Certainty Factor* pada Sistem Pakar Diagnosa Penyakit THT". Sistem pakar dapat membantu Tim Medis dalam mendiagnosa suatu penyakit, khususnya penyakit THT berdasarkan gejala-gejala yang dikeluhkan [7].

Penelitian lainnya dilakukan oleh Rizal Rachman, Amirul Mukminin dengan judul "Penerapan Metode *Certainty Factor* pada Sistem Pakar Penentuan Minat dan Bakat Siswa SD". Aplikasi ini mampu menentukan minat dan bakat siswa. Aplikasi ini juga memberikan informasi tentang kecerdasan, minat dan bakat baik itu jenis, ciri-ciri atau pun stimulasi minat dan bakat dengan cara yang mudah [8].

Sistem pakar yang akan saya kembangkan, dapat mendiagnosis penyakit jantung dilihat dari gejala yang dipilih pengguna dan dapat memberikan solusi untuk mengatasi penyakit tersebut. Solusi yang diberikan mengenai pola hidup sehat hal ini menjadi pembeda, dibanding sistem pakar lain.

Sistem ini dikembangkan dengan media penyimpanan menggunakan XAMPP v3.2.3 dengan format data SQL. Untuk memisahkan variabel data yang akan dirubah dalam format SQL digunakan bahasa pemrograman *Hypertext Prepocessor* (PHP) sebagai lapisan *query*, *framework Codeigniter* versi 3, teks editor *Sublime Text* 3, dan *phpmyadmin* sebagai *database*.

2.2 Landasan Teori

2.2.1 Rekayasa Web

Web Engineering atau sering dikenal dengan Rekayasa Web adalah disiplin ilmu yang mempelajari proses yang digunakan untuk menciptakan aplikasi web yang berkualitas tinggi [9]. Mengadaptasi rekayasa perangkat lunak dalam hal konsep dasar yang menekankan pada aktifitas teknis dan manajemen, tapi dengan perubahan dan penyesuaian. Selain itu *Web Engineering* merupakan gabungan antara *web publishing* (suatu konsep yang berasal dari *printed publishing*) dan aktifitas rekayasa perangkat lunak karena desain sebuah aplikasi web menekankan pada desain grafis, desain informasi, teori *hypertext*, desain sistem dan pemrograman [10].

Metode ini memerlukan pendekatan yang sistematik dan sekuensial yang mulai pada tingkat dan kemajuan sistem pada setiap tahapan [11]. Metode *web engineering* terdapat 5 (lima) tahapan untuk dapat mengembangkan suatu perangkat lunak yaitu [12]:

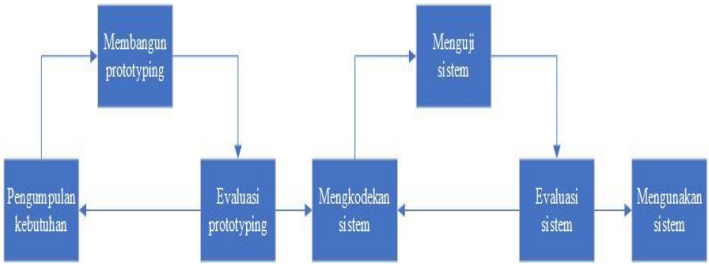
1. *Customer Communication*
   Komunikasi dalam hal ini terutama terkonsentrasi pada dua hal, analisa bisnis dan perumusan. Analisa bisnis akan mendefinisikan hal-hal apa saja yang akan termuat di dalam aplikasi web, misalnya pengguna web yang akan dibangun, perubahan potensial dalam lingkungan bisnis, integrasi antara web yang akan dibangun dengan situasi bisnis perusahaan, maupun *database* perusahaan.
2. *Planning*
   Perencanaan proyek pengembangan aplikasi web kemudian ditentukan perencanaan akan terdiri dari pendefinisian pekerjaan dan target waktu atas pekerjaan maupun sub pekerjaan yang ditentukan tersebut.
3. *Modeling*
   Tujuan dari aktivitas ini adalah untuk menjelaskan hal-hal apa saja yang memang diperlukan/ dibutuhkan pada aplikasi yang akan

dibangun dan solusi yang ditawarkan yang diharapkan dapat menjawab apa yang tersirat dari hasil-hasil analisa dan pengumpulan data.

4. *Construction*
   Pembangunan aplikasi web memadukan antara perkembangan teknologi dengan tools pengembangan web yang telah ada, artinya memilih *tools* yang efektif namun tetap dapat menyesuaikan dengan teknologi yang berkembang saat ini.
5. *Deployment*
   Aplikasi web diciptakan untuk dapat berguna bagi kebutuhan pekerjaan, dapat dioperasikan oleh *end-user*, dan kemudian dilakukan evaluasi secara berkala, memberi masukan-masukan kepada team pengembang dan apabila diperlukan akan dilakukan modifikasi pada aplikasi web tersebut.

1. Metode *Prototype*

Metode pengembangan sistem yang digunakan adalah model *Prototype*. *Prototype* merupakan metode pengembangan perangat lunak, yang berupa model fisik kerja sistem dan berfungsi sebagai versi awal dari sistem. Dengan metode *prototyping* ini akan dihasilkan *prototype* sistem sebagai perantara pengembang dan pengguna agar dapat berinteraksi dalam proses kegiatan pengembangan sistem informasi [4]. Metode ini dipilih karena dapat ditambah maupun dikurangi sesuai berjalannya proses pengembangan. Kemajuan tahap demi tahap dapat diikuti langsung oleh pengguna. Disamping itu, tahapan pada *prototype* dapat menghemat sumberdaya dan waktu dalam menghasilkan produk yang lebih baik dan tepat guna bagi pengguna. Berikut ini tahapan-tahapan utama dari metode *prototype* [5] :



Gambar 2. 1 Metode *Prototype*

1. Pengumpulan kebutuhan
   Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.
2. Membangun *prototyping*
   Membangun *prototyping* dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misalnya dengan membuat input dan format output).
3. Evaluasi *protoptyping*
   Evaluasi ini dilakukan oleh pelanggan apakah *prototyping* yang sudah dibangun sudah sesuai dengan keinginann pelanggan. Jika sudah sesuai maka langkah 4 akan diambil. Jika tidak *prototyping* direvisi dengan mengulang langkah 1, 2 , dan 3.
4. Mengkodekan sistem
   Dalam tahap ini *prototyping* yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.
5. Menguji sistem
   Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini dilakukan dengan *White Box, Black Box, Basis Path*, pengujian arsitektur dan lain-lain
6. Evaluasi Sistem
   Pelanggan mengevaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan . Jika ya, langkah 7 dilakukan, jika tidak, ulangi langkah 4 dan 5.
7. Menggunakan sistem
   Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan.

2. UML (*Unified Modelling Language*)

UML (*Unified Modelling Language*) adalah sebuah bahasa untuk menentukan, visualisasi, konstruksi, dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perabfgkat lunak. *Artifact* dapat berupa model deskripsi atau perangkat lunak dari sistem perangkat lunak, seperti pada pemodelan bisnis dan system non perangkat lunak lainnya. UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan system yang besar dan kompleks. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan [13]. Adapun diagram yang kali ini digunakan:

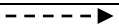
a. Use Case

Use Case atau diagram *Use Case* merupakan permodelan untuk kelakukan (behavior) sistem informasi yang akan dibuat. Use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut [14]. Syarat penamaan pada *Use Case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Terdapat dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case* [15]. Penjelasan symbol-simbol dalam perancangan *Use Case diagram* dapat dilihat pada Tabel 2.1 [16].

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Oleh karena itu walaupun simbol dari aktor ada adalah gambar orang, tetapi belum tentu hal tersebut merupakan orang [15].
2. *Use Case* merupakan fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau actor [15].

Tabel 2. 1 Simbol *Use Case*

No.	Gambar	Nama	Keterangan
1.		Aktor	Mewakili peran orang, sistem yang lain atau alat ketika berkomunikasi dengan use case

2.		Association	Apa yang menghubungkan antara objek satu dengan objek lainnya
3.		Include	Menspesifikasikan bahwa use case sumber secara eksplisit
4.		Extend	Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan
5.		System	Menspesifikasikan paket yang menampilkan sistem secara terbatas
6.		Use Case	Deskripsi dari urutan aksi – aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor

b. Sequence Diagram

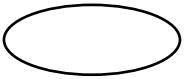
Sequence Diagram menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek [14]. Oleh karena itu untuk menggambar diagram sequence maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu [17]. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan

interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak use case yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak [18].

c. Entity Relation Diagram (ERD)

Entity Relation Diagram adalah suatu model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak [19]. Tujuan dari ERD ini adalah untuk menunjukan objek dan *relationship* yang ada pada objek tersebut, disamping itu ERD bersama-sama dengan detail pendukung merupakan model data yang pada gilirannya digunakan sebagai spesifikasi untuk database [20]. Beberapa symbol yang dapat digunkan dalam pembuatan ERD beserta fungsinya dapat dilihat pada Tabel 2.2 [16].

Tabel 2. 2 Tabel Simbol ERD

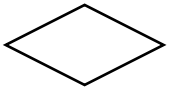
No.	Simbol	Nama dan Fungsi
1.		Entitas adalah sesuatu apa saja yang ada dalam sistem, nyata maupun abstrak dimana data disimpan atau dimana terdapat data.
2.		Atribut Adalah sifat atau karakteristik dari tiaptiap entitas dan relasi atau elemen data dari entitas dan relasi. Atribut ini digunakan untuk penamaan dari bagian-bagian yang terdapat dalam entitas
3.		Relasi, Digambarkan dengan sebuah bentuk belah ketupat. Relasi adalah hubungan alamiah yang terjadi antara entitas.
4.		Garis lurus, Menghubungkan antara entitas satu dengan entitas yang lainnya.

d. *Flowchart*

Flowchart merupakan penggambaran secara grafik dari langkah-langkah dan urutan prosedur suatu program[21]. Biasanya mempengaruhi

penyelesaian masalah yang khusunya perlu dipelajari dan dievaluasi lebih lanjut [22]. Penjelasan symbol-simbol dalam flowchart dapat dilihat pada Tabel 2.3

Tabel 2. 3 Tabel simbol *flowchart*

No.	Simbol	Nama	Fungsi
1.		Terminator	Awal atau akhir paragraf
2.		Flow	Arah aliran program
3.		Preparation	Inisialisasi/ pemberian nilai awal
4.		Proses	Proses / pengolahan data
5.		Input / output data	Simbol yang menyatakan proses input atau output tanpa tergantung jenis peralatannya.
6.		Sub Program	Sub program
7.		Decision	Seleksi atau kondisi

8.		On page conector	Penghubung bagian – bagian flowchart pada halaman yang sama.
9.		Of page conector	Penghubung bagian – bagian flowchart pada halaman yang berbeda.

3. Blackbox Testing

Pengujian *blackbox* atau disebut uji fungsional adalah pengujian yang mengabaikan mekanisme internal sistem atau komponen dan hanya berfokus pada output yang dihasilkan dalam menanggapi input yang dipilih dan kondisi eksekusi. Sehingga dapat disimpukan bahwa *blackbox testing* merupakan pengujian yang berorientasi pada fungsionalitas yaitu perilaku dari perangkat lunak atas input yang diberikan pengguna sehingga mendapatkan/ menghasilkan output yang diinginkan tanpa melihat proses internal atau kode program yang dieksekusi oleh perangkat lunak.

Blackbox Testing dibagi menjadi dua jenis, yaitu [23]:

a) Functional Testing

Dimana jenis ini berkaitan dengan persyaratan fungsional atau spesifikasi aplikasi. Beberapa jenis Pengujian Fungsional utama adalah:

1.Smoke Testing
2.Sanity Testing
3.Integration Testing
4.System Testing
5.Regression Testing
6.User Acceptance Testing

b) Non-Functional Testing

Dimana terlepas dari fungsionalitas persyaratan, ada beberapa aspek non-fungsional yang perlu diuji untuk meningkatkan kualitas dan kinerja aplikasi, yang meliputi:

1.Usability Testing
2.Load Testing
3.Performance Testing

4.Compatibility Testing
5.Stress Testing
6.Scalability Testing

2.2.2 Basis Data

Basis data terdiri dari 2 kata, yaitu Basis dan Data. Basis dapat diartikan sebagai mrkas atau gudang diman atempat bersarang/berkumpul. Sedangkan data adalah representasi fakta dunai nyata yang mewakili suatu objek seperti manusia, barang, hewan, peristiwa konsep dan sebagainya, yang akan direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi atau kombinasinya [23]. Basis data adalah kumpulan file-file yang mempunyai itan antara satu file denga file yang lain sehingga membentuk suatu bangunan data untuk menginformasikan suatu perusahaan atau instansi dalam batasan tertentu. Adapun tujuan dari basis data adalah sebagai berikut:

1. Mengatur data sehingga diperoleh kemudahan, ketepatan dan kecepatan dalam penggunaan data tersebut
2. Tidak adanya redundansi dan menjaga konsistensi data
3. Pengaturan dalam pemilhana data sesuai dengan fungsi dan jenisnya.

a. Mysql

Mysql merupakan sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis. Setiap pengguna dapat secara bebas menggunkan Mysql tetapi dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya. SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basisdata transaksional maupun operasi basisdata non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak

peladen basisdata kompetitor lainnya. Namun pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi *blogging* berbasis web, CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional [24].

2.2.3 PBO

A. Pengertian PBO

Metodologi berorientasi objek merupakan salah satu strategi pembangunan perangkat luank yang mengorganisasikan *software* sebagai kelompok objek yang berisi operasi dan data yang digunakan terhadapnya. Metode berorientasi objek didasari pada penggunaan kaidah pengelolaan kompleksitas. Berikut ini konsep dasar pada metodologi berorientasi objek:

1. Kelas (*class*)
   Kelas merupakan kumpulan objek-objek dengan sifat yang sama. Kelas adalah definisi statik dan himpunan objek yang sama yang mungkin diciptakan dari kelas tersebut.
2. Objek (*object*)
   Objek adalah suatu entitas yang dapat menyimpan informasi dan memiliki operasi (kelakuan) yang dapat berpengaruh pada satus objeknya.
3. Metode (*method*)
   Operasi merupakan fungsi yang mampu dilakukan teperasi merupakan fungsi yang mampu dilakukan terhadap objek atau dilakukan oleh objek.
4. Atribut (*atribut)*
   Atribut dari sebuah kelas adalah variable global u=yang mempunyai sebuah kelas. Atribut bisa berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek.
5. Abstraksi (*abstraction*)
   Prinsip untuk mewakili dunia nyata yang kompleks menjadi satu bentuk model yang sederhania dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan persoalan.
6. Enkapsulasi (*encapsulation*)

Sebuah mekanisme yang digunakan untuk mneyembunyikan atau melindungi proses dari gangguan atau penyalahgunaan dari luar sistem, sekaligus menyederhanakan penggunaan sistem.

7. Pewarisan (*inheritance)*

   Konsep mewarisi atribut dan metode yang dimiliki kelas tertentu ke kelas turunannya. Dengan konsep ini, kelas yang dibuat dapat dengan mudah mendefinisikan atribut dan metode spesifiknya, atribut dan metjode yang lebih umum akan diperoleh dari kelas sebagai kelas induk.

2.2.4 CodeIgniter

Menurut Budi Raharjo (2015:3), "*CodeIgniter* adalah *framework web* untuk bahasa pemrograman PHP yang dibuat oleh Rick Ellis pada tahun 2006, penemu dan pendiri EllisLab. *CodeIgniter* memiliki banyak fitur (fasilitas) yang membantu para pengembang (*developer*) PHP untuk dapat membuat aplikasi web secara mudah dan cepat. Dibandingkan dengan *framework web* PHP lainnya, harus diakui bahwa *CodeIgniter* memiliki desain yang lebih sederhana dan bersifat fleksibel (tidak kaku). *CodeIgniter* mengizinkan para pengembang untuk menggunakan *framework* secara parsial atau secara keseluruhan.

CodeIgniter merupakan sebuah *toolkit* yang ditujukan untuk orang yang ingin membangun aplikasi web dalam bahasa pemrograman PHP. Beberapa keunggulan yang ditawarkan oleh *CodeIgniter* adalah sebagai berikut:

1. *CodeIgniter* adalah *framework* yang bersifat *free* dan *open-source.*
2. *CodeIgniter* memiliki ukuran yang kecil dibandingkan dengan framework lain. Setelah proses instalasi, *framework CodeIgniter* hanya berukuran kurang lebih 2MB (tanpa dokumentasi atau jika direktori *user guide* dihapus). Dokumentasi *CodeIgniter* memiliki ukuran sekitar 6MB.
3. Aplikasi yang dibuat menggunakan *CodeIgniter* bisa berjalan cepat. 22
4. *CodeIgniter* menggunakan pola desain *Model-View-Controller* (MVC) sehingga satu file tidak terlalu berisi banyak kode. Hal ini menjadikan kode lebih mudah dibaca, dipahami, dan dipelihara di kemudian hari.
5. *CodeIgniter* dapat diperluas sesuai dengan kebutuhan.

6. *CodeIgniter* terdokumentasi dengan baik. Informasi tentang pustaka kelas dan fungsi yang disediakan oleh *CodeIgniter* dapat diperoleh melalui dokumentasi yang disertakan di dalam paket distribusinya [13].

2.2.5 Kecerdasan Buatan

Kecerdasan buatan adalah salah satu cabnag ilmu pengetahuan yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara yang lebih manusiawi. Hal Ini biasanya dilakukan dengan mengikuti/mencontoh karakteristik dan analogi berpikir dari kecerdasan/Inteligensia manusia, dan menerapkannya sebagai algoritma yang dikenal oleh komputer. Dengan suatu pendekatan yang kurang lebih fleksibel dan efisien dapat diambil tergantung dari keperluan, yang mempengaruhi bagaimana wujud dari perilaku kecerdasan buatan. AI biasanya dihubungkan dengan Ilmu Komputer, akan tetapi juga terkait erat dengan bidang lain seperti Matematika, Psikologi, Pengamatan, Biologi, Filosofi, dan yang lainnya. Kemampuan untuk mengkombinasikan pengetahuan dari semua bidang ini pada akhirnya akan bermanfaat bagi kemajuan dalam upaya menciptakan suatu kecerdasan buatan [25].

Pengembangan lain kecerdasan buatan adalah sistem pakar yang menggabungkan pengetahuan dan penelusuran data untuk memastikan masalah yang secara normal memerlukan keahlian manusia [26].

2.2.6 Sistem Pakar

Sistem pakar (*expert system*) adalah sistem yang berusaha mengadopsi pengetahuan manusia ke kompuetr, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli [27]. Dengan sistem pakar ini orang awam dapat menyelesaikan msalahnya atau hanya sekerdar mencari suatu informasi berkualitas yang sebenarnya hanya dapat diperoleh dengan bantuan para ahli dibidangnya. Sistem pakar ini juga akan dapat membantu aktivitas para pakar sebagai asisten yang berpengalaman dan mempunyai asisten yang berpengalaman dan mempunyai pengetahuan yang dibutuhkan.

Dalam penyusunannya, sistem pakar mengkombinasikan kaidah-kaidah penarikan kesimpulan (*inference rules*) dengan basis pengetahuan tertentu yang diberikan oleh satu atau lebih pakar dalam bidang tertentu.

Kombinasi dari kedua hal tersebut disimpan dalam komputer, yang selanjutnya digunakan dalam proses pengambilan keputusan untuk penyelesaian masalah tertentu. Ada berbagai ciri dan karakteristik yang membedakan Sistem Pakar dengan sistem yang lain. Ciri dan karakteristik ini menjadi pedoman utama dalam pengembangan Sistem Pakar. Ciri dan karakteristik yang dimaksud antara lain Arhami (2005) [28]:

a. Pengetahuan Sistem Pakar merupakan suatu konsep, bukan berbentuk numeris. Hal ini dikarenakan komputer melakukan proses pengolahan data secara numerik sedangkan keahlian dari seorang pakar adalah fakta dan aturan-aturan, bukan numerik.
b. Informasi dalam Sistem Pakar tidak selalu lengkap, subyektif, tidak konsisten, subyek terus berubah dan tergantung pada kondisi lingkungan sehingga keputusan yang diambil bersifat tidak pasti dan tidak mutlak "ya" atau "tidak" akan tetapi menurut ukuran kebenaran tertentu. Oleh karena itu dibutuhkan kemampuan sistem untuk belajar secara mandiri dalam menyelesaikan masalah-masalah dengan pertimbangan-pertimbangan khusus.
c. Kemungkinan solusi Sistem Pakar terhadap suatu permasalahan adalah bervariasi dan mempunyai banyak pilihan jawaban yang dapat diterima, semua faktor yang ditelusuri memiliki ruang masalah yang luas dan tidak pasti. Oleh karena itu diperlukan sistem yang fleksibel dalam menangani kemungkinan solusi dari berbagai permasalahan.
d. Perubahan atau pengembangan pengetahuan dalam Sistem Pakar dapat terjadi setiap saat bahkan sepanjang waktu sehingga diperlukan kemudahan dalam modifikasi sistem untuk menampung jumlah pengetahuan yang semakin besar dan semakin bervariasi.
e. Pandangan dan pendapat setiap pakar tidaklah selalu sama, yang oleh karena itu tidak ada jaminan bahwa solusi Sistem Pakar merupakan jawaban yang pasti benar. Setiap pakar akan memberikan pertimbangan-pertimbangan berdasarkan faktor subyektif.
f. Keputusan merupakan bagian terpenting dari Sistem Pakar. Sistem harus memberikan solusi yang akurat berdasarkan masukan pengetahuan meskipun solusinya sulit sehingga fasilitas informasi sistem harus selalu diperlukan.

2.2.7 Certainty Factor

Menurut Rizal Rachman dan Amirul Mukminin metode *Certainty Factor* merupakan suatu metode untuk membuktikan apakah suatu fakta itu pasti ataukah tidak pasti yang berbentuk metric yang biasanya digunakan dalam sistem pakar [8]. Menurut Sutojo, dkk (2010:194) awal mula Teori *certainty factor* (CF) diusulkan oleh Shortlife dan Buchanan pada 1975 untuk mengakomodasi ketidakpastian pemikiran seorang pakar. Seorang pakar/ahli dalam hal ini biasanya dokter sering kali menganalisis informasi yang ada dengan ungkapan seperti "mungkin", "kemungkinan besar", "hampir pasti". Untuk mengakomodasi hal ini kita menggunakan certainty factor guna menggambarkan tingkat keyakinan pakar terhadap masalah yang sedang dihadapi [28]. Alasan penggunaan metode ini karena dapat memberikan hasil yang akurat yang didapatkan dari perhitungan berdasarkan bobot gejala yang dipilih pengguna, mampu memberikan jawaban pada permasalahan yang tidak pasti kebenarannya seperti masalah diagnosa resiko penyakit, dan dengan metode ini pakar menggambarkan keyakinan seorang pakar dengan memberikan bobot keyakinan sesuai dengan pengetahuan pakar terkait [6].

Nilai kepercayaan diperoleh dari interpretasi seorang pakar yang dikonversi menjadi nilai kepercayaan dengan ketentuan seperti pada Tabel 2.4 [29].

Tabel 2. 4 Ketentuan Nilai Kepercayan (CF)

Uncertainty Term	CF
Pasti Tidak	0,2 - 0
Hampir pasti tidak	0,3
Kemungkinan besar tidak	0,4
Kemungkinan tidak	0,5
Tidak tahu	0,6
Kemungkinan	0,7
Kemungkinan besar	0,8
Hampi pasti	0,9
Pasti	1

Certainty Factor memperkenalkan konsep keyakinan dan ketidakyakinan yang kemudian diformulakan dalam rumusan dasar sebagai berikut [30]:

CF [P,E] = MB [P,E] – MD [P,E] …… **(Persamaan 1)**

Keterangan :

CF *: Certainty Factor*
MB *: Measure of Belief*
MD *: Measure of Disbelief*
P *: Probability*
E *: Evidence* (Peristiwa/Fakta)

Berikut ini adalah deskripsi beberapa kombinasi *Certainty Factor* terhadap berbagai kondisi :

Certainty Factor untuk kaidah dengan premis tunggal *(single premis rules)*:
CF(H,E) = CF(E)*CF(*rule*)
= CF(user)*CF(pakar))……...**(Persamaan 2)**

Certainty Factor untuk kaidah dengan premis majemuk *(multiple premis rules)*:
CF *(A AND B)* = *Minimum* (CF (a),CF (b)) * CF *(rule)*
CF (A OR B) = Maximum (CF (a),CF (b)) * CF (rule)….**(Persamaan 3)**

Certainty Factor untuk kaidah dengan kesimpulan yang serupa *(similarly concluded rules)* :
CFCOMBINE (CF1, CF2) = CF1 + CF2*(1-CF1)……**(Persamaan 4)**

2.2.8 Penyakit Jantung

A. Definisi Penyakit Jantung

Penyakit kardiovaskular atau yang biasa disebut penyakit jantung umumnya mengacu pada kondisi yang melibatkan penyempitan atau pemblokiran pembuluh darah yang bisa menyebabkan serangan jantung, nyeri dada (angina) atau stroke. Kondisi jantung lainnya yang mempengaruhi otot jantung, katup atau ritme, juga dianggap bentuk penyakit jantung (*American Heart Association*, 2017) [28].

B. Penyakit Jantung

Berikut macam macam penyakit pada jantung yang dicantumkan dibatasan masalah beserta gejala dan solusi yang diperoleh dari narasumber dan berbagai informasi dari sumber pustaka yang kemudain dijadikan data dalam pemnuatan sistem dapat dilihat pada Tabel 2.5.

Tabel 2. 4 Tabel penyakit pada jantung beserta gejala

No	Penyakit	Gejala
1.	Gagal Jantung	- Sesak napas saat aktivitas - Sesak napas saat posisi berbaring - Riwayat terbangun malam hari karena sesak napas - Penurunan toleransi aktivitas fisik - Kelelahan atau keletihan - Membutuhkan waktu yang lebih lama untuk pemulihan setelah aktivitas fisik - Kaki bengkak - Batuk pada malam hari - Napas mengi - Perasaan penuh di perut - Penurunan nafsu makan - Linglung (terutama pada usia tua) - Depresi - Berdebar-debar - Pingsan - Pusing

		- BB naik lebih dari 2 kg/minggu - Kulit terasa dingin
2.	Serangan Jantung	- Nyeri dada tidak dapat dilokalisir - Nyeri dada dipicu oleh aktivitas fisik - Nyeri dada dipicu oleh stres emosional - Nyeri dada berkurang saat istirahat - Nyeri dada berkurang saat meletakkan obat di bawah lidah - Nyeri dada disertai keringat dingin hingga baju basah, mual, m - Nyeri dada menjalar ke lengan kiri, bahu, punggung, *epigastrium*, leher rasa tercekik atau rahang bawah - Nyeri dada terasa di bagian tengah dada atau sebelah kiri - Nyeri dada dengan durasi lebih dari 20 menit - Nyeri dada rasanya seperti ditindih beban berat, diremas, rasa terbakar - Nyeri dada terasa seperti ditusuk-tusuk jarum - Nyeri dada terasa hilang dan timbul, serta berlangsung dalam hitungan detik - Nyeri dada terasa tanpa henti sepanjang hari dengan intensitas sama - Nyeri dada berkaitan dengan posisi tubuh miring atau bungkuk - Nyeri dada terpusat pada satu lokasi di dada - Nyeri dada dapat ditunjuk dengan jari

		- Nyeri dada terasa saat menarik napas dalam-dalam 35
3.	Aritmia	- Detak jantung berdetak lebih cepat (lebih dari 100 kali/menit) - Detak jantung berdetak lebih lambat (kurang dari 60 kali/menit) - Detak jantung terasa tidak teratur - Dada terasa berdebar-debar - Pingsan - Pandangan mata gelap hingga mau jatuh - Pusing - Badan lemas