BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Penelitian yang dilakukan oleh Hariyanto dan Leidiyana (2020) mengenai Sistem Pakar untuk Mendiagnosa Penyakit Persendian Menggunakan Metode *Certainty Factor*, penelitian Hariyanto dan Leidiyana menggunakan metode *certainty factor* untuk diagnosa penyakit persendian, sedangkan penelitian ini menggunakan metode *certainty factor* untuk diagnosa penyakit pada hati.

Penelitian yang dilakukan oleh Hidayat dan Mulyanto (2021) mengenai Sistem Pakar Diagnosa Penyakit Ayam Petelur Menggunakan Metode *Certainty Factor* Berbasis *Web*, penelitian Hidayat dan Mulyanto menggunakan metode *certainty factor* untuk diagnosa penyakit ayam petelur, sedangkan penelitian ini menggunakan metode *certainty factor* untuk diagnosa penyakit pada hati.

Penelitian yang dilakukan oleh Prebiana dan Astuti (2020) mengenai Penerapan Metode *Certainty Factor* (CF) Dalam Pembuatan Sistem Pakar Diagnosis Penyakit Tumor Otak, penelitian Prebiana dan Astuti menggunakan metode *certainty factor* untuk diagnosa penyakit tumor otak, sedangkan penelitian ini menggunakan metode *certainty factor* untuk diagnosa penyakit pada hati.

Penelitian yang dilakukan oleh Putri (2020) mengenai Perancangan Aplikasi Sistem Pakar Penyakit *Roseola* Dengan Menggunakan Metode *Certainty Factor*,

penelitian Putri menggunakan metode *certainty factor* untuk diagnosa penyakit *roseola*, sedangkan penelitian ini menggunakan metode *certainty factor* untuk diagnosa penyakit pada hati.

Penelitian yang dilakukan oleh Wahyuningsih dan Zuhriyah (2021) mengenai Sistem Pakar Diagnosa Penyakit Campak *Rubella* Pada Anak Menggunakan Metode *Certainty Factor* Berbasis *Website*, penelitian Wahyuningsih dan Zuhriyah menggunakan metode *certainty factor* untuk diagnosa penyakit campak *rubella*, sedangkan penelitian ini menggunakan metode *certainty factor* untuk diagnosa penyakit pada hati.

II.2. Landasan Teori

Landasan teori dikutip dari beberapa jurnal penelitian terdahulu yang berkaitan dengan penelitian ini.

II.2.1. Sistem Pakar

Sistem Pakar (*Expert System*) adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang dilakukan oleh para ahli, dan sistem pakar yang baik dirancang agar dapat menyelesaikan suatu permasalahan tertentu dengan meniru kerja dari para ahli. Pada dasarnya sistem pakar diterapkan untuk mendukung aktivitas pemecahan masalah. Beberapa aktivitas pemecahan masalah yang dimaksud antara lain: pembuatan keputusan (*decision making*), panduan pengetahuan (*knowlegde fusing*),

pembuatan desain (*forecasting*), pengaturan (*regulating*), pengendalian (*controlling*), diagnosis (*diagnosing*), perumusan (*prescribing*), penjelasan (*explaining*), pemberian nasihan (*advising*), dan pelatihan (*tutoring*).

Komponen-komponen yang terdapat dalam arsitektur atau struktur pakar adalah sebagai berikut:

1. Antarmuka Pengguna (*User Interface*)
2. Basis Pengetahuan (*Knowledge Base*)
3. Basis Data (*Data Base*)
4. Mesin Inferensi (*Inference Engine*). (Hidayat dan Mulyanto, 2021: 149).

II.2.2. Diagnosa

Diagnosa merupakan tahapan dan hasil dari diagnosis suatu penyakit yang diderita oleh penderita. Menurut Kamus Besar Bahasa Indonesia (KBBI), Diagnosis adalah penentuan jenis penyakit dengan cara memeriksa atau meneliti gejala-gejalanya. Secara umum diagnosa adalah upaya untuk mengetahui atau mengindentifikasi suatu penyakit atau masalah kesehatan yang diderita atau dialami seseorang pasien atau penderita. Diagnosis adalah menentukan sebab malfungsi dalam situasi kompleks yang didasarkan pada gejala-gejala yang teramati, diantaranya medis, elektronis, mekanis, dan diagnosis perangkat lunak. (Ayudia, 2020: 383).

II.2.3. Penyakit

Penyakit adalah suatu keadaan tidak normal dari tubuh maupun psikis yang sifatnya objektif karena masing-masing memiliki parameter tertentu. Penyakit adalah istilah medis yang digambarkan sebagai gangguan dalam fungsi tubuh yang menghasilkan berkurangnya kapasitas. Penyakit terjadi ketika keseimbangan dalam tubuh tidak dapat dipertahankan. Keadaan sakit terjadi pada saat seseorang tidak lagi berada dalam kondisi sehat yang normal. (Sukorini, 2017: 9).

II.2.4. Hati

Hati adalah organ yang paling besar dan penting bagi tubuh kita. Kita tidak bisa hidup tanpa hati. Penyakit pada hati merupakan peradangan yang disebabkan oleh infeksi virus, bakteri atau bahan-bahan beracun sehingga hati tidak dapat melakukan fungsinya dengan baik serta tidak mudah ditemukan dalam tahap awal dalam mendiagnosis penyakit pada hati. Penanganan pasien dengan penyakit pada hati pada tahap awal akan memperpanjang hidup pasien, namun sayangnya semua ahli kesehatan tidak memiliki keahlian khusus dalam mendiagnosis medis, sehingga sistem diagnosis otomatis secara medis mungkin akan sangat bermanfaat dan membantu dengan menggunakan *machine learning* telah banyak digunakan dalam bidang medis. (Musyaffa dan Rifai, 2018: 190).

II.2.5. Penyakit Pada Hati

Hati merupakan organ yang sangat penting dalam pengaturan homeostasis tubuh meliputi *metabolisme, biotransformasi*, *sintesis*, penyimpanan dan *imunologi*. Sel-sel hati (*hepatosit*) mempunyai kemampuan regenerasi yang cepat. Oleh karena itu sampai batas tertentu, hati dapat mempertahankan fungsinya bila terjadi gangguan ringan. Pada gangguan yang lebih berat, terjadi gangguan fungsi yang serius dan akan berakibat fatal. Jenis-jenis penyakit pada hati antara lain yaitu, abses hati, hepatitis, kanker hati, sirosis hati, liver, hemokromatis, kista hati, autoimun dan lemak hati. Penyakit-penyakit pada hati akut akan banyak mempengaruhi fungsi-fungsi hati, penyakit tesebut dapat diketahui dari gejala klinis maupun fisik yang timbul pada diri pasien, gejala klinis dapat diketahui dari apa yang dirasakan oleh pasien, sedangkan gejala fisik dapat diketahui dari keadaan tubuh pasien. Penyebab penyakit pada hati bervariasi, sebagian besar disebabkan oleh virus yang menular secara fekal-oral, parenteral, seksual, perinatal dan sebagainya. Penyebab lain dari penyakit pada hati adalah akibat efek toxic dari obat-obatan, alkohol, racun, jamur dan lain-lain. Di samping itu juga terdapat beberapa penyakit pada hati yang belum diketahui pasti penyebabnya. (Rafsanjani, dkk, 2018: 4479).

Penyakit pada hati atau liver memiliki berbagai macam jenis, salah satunya yang sangat berbahaya dan sering terjadi adalah hepatitis B. Menurut Yayasan Lembaga Konsumen Indonesia, penyakit hepatitis B termasuk dalam ancaman kesehatan tertinggi di Indonesia. Tercatat dari Riset Kesehatan Dasar (Riskesdas) tahun 2013 mencatat, secara nasional prevalensi penyakit hepatitis B mencapai 21,8

persen, atau menduduki peringkat tertinggi ketiga di Indonesia. Dengan daerah prevalensi tertinggi di provinsi Bangka Belitung (48,2%), Maluku (47,6%) dan DKI Jakarta (37,7%). Tingginya tingkat presentase yang muncul di Indonesia terjadi karena beberapa faktor, seperti faktor lingkungan, faktor pelayanan kesehatan, dan faktor dari masyarakat sendiri. Faktor dari masyarakat terjadi diantaranya karena kurangnya pengetahuan masyarakat tentang penyakit pada hati, meliputi gejala-gejala dan jenis penyakit pada hati serta cara pencegahannya. (Prayoga, dkk, 2018: 2667).

II.2.6. Metode *Certainty Factor*

Metode *Certainty Factor* adalah metode yang digunakan untuk menyatakan kepercayaan dalam sebuah kejadian berdasarkan hasil penelitian atau penilaian dari para pakar. Metode *certainty factor* digunakan ketika menghadapi suatu masalah yang jawabannya tidak pasti. Ketidakpastian ini dapat berupa probabilitas. Metode *certainty factor* dibagi beberapa tahap yang direlasikan dengan data latih yaitu data rekam medik yang sebelumnya sudah dihitung terlebih dahulu. Metode ini sangat cocok untuk sistem pakar untuk mendiagnosa hal-hal yang belum pasti. Faktor kepastian (*Certainty Factor*) dikemukakan oleh Shortliffe Buchanan dalam pembuatan MYCIN (sistem pakar yang dikembangkan pada awal tahun 1970 di Stanford University). *Certainty Factor* (CF) merupakan nilai parameter klinis yang diberikan MYCIN untuk menunjukan besar kepercayaan. (Hariyanti dan Leidiyana, 2020: 28).

Certainty Factor didefinisikan sebagai persamaan berikut:

CF (H, E) = MB (H, E)-MD (H, E)).....(1)

CF (H, E) : *Certainty Factor* dari hipotesis H yang dipengaruhi oleh gejala (*evidence*) E. Besarnya CF berkisar antara -1 sampai 1. Nilai -1 menunjukkan ketidak percayaan mutlak sedangkan nilai 1 menunjukkan kepercayaan mutlak.

MB (H, E): ukuran kenaikan kepercayaan (*measure of increased belief*) terhadap hipotesis H yang dipengaruhi oleh gejala E.

MD (H, E): ukuran kenaikan ketidakpercayaan (*measure of increased disbelief*) terhadap hipotesis H yang dipengaruhi oleh gejala E.

Bentuk dasar rumus *certainty factor*, adalah sebuah aturan jika E maka H seperti ditunjukkan pada persamaan II.1. berikut:

CF (H, e) = CF (E, e) * CF (H, E).....(2)

Dimana:

CF (H, e) : *Certainty Factor* hipotesis yang dipengaruhi oleh *evidence* e.

CF (E, e) : *Certainty Factor evidence* E yang dipengaruhi oleh *evidence* e.

CF (H, E): *Certainty Factor* hipotesis dengan asumsi *evidence* diketahui dengan pasti, yaitu ketika CF (E, e) = 1.

Jika semua *evidence* pada *antecedent* diketahui dengan pasti maka persamaannya akan menjadi:

CF (E, e) = CF (H, E)

Dalam aplikasinya, CF (H,E) merupakan nilai kepastian yang diberikan oleh pakar terhadap suatu aturan, sedangkan CF (E,e) merupakan nilai kerpercayaan yang diberikan oleh pengguna terhadap gejala yang dialaminya. (Hariyanti dan Leidiyana, 2020: 28).

II.2.7. *Web*

Situs *web* merupakan salah satu media yang dapat digunakan untuk memberikan layanan informasi. *Web Browser* adalah aplikasi perangkat lunak yang digunakan untuk mengambil dan menyajikan sumber informasi *web*. Sumber informasi *web* di identifikasikan dengan *Uniform Resource Identufuer* (URL) yang dapat terdiri dari halaman *web*, *video*, gambar, ataupun konten lainnya. (Pakpahan dan Halawa, 2020: 111).

II.2.8. *Hyper Text Markup Language* (HTML)

HTML adalah singkatan dari *Hypertext Markup Language*. HTML memungkinkan seorang *user* untuk membuat dan menyusun bagian paragraf, *heading*, *link* atau tautan, dan *blockquote* untuk halaman *web* dan aplikasi. HTML bukanlah bahasa pemrograman, dan itu berarti HTML tidak punya kemampuan untuk membuat fungsionalitas yang dinamis. Sebagai gantinya, HTML memungkinkan *user* untuk mengorganisir dan memformat dokumen, sama seperti Microsoft Word. (Sugijanto, dkk, 2020: 2).

HTML ialah kepanjangan dari *Hypertext Markup Language*. Definisi HTML adalah bahasa yang digunakan untuk menulis halaman *web*. Fungsi utama HTML ialah memberi perintah pada *browser* untuk melakukan manipulasi tampilan melalui *tag-tag* yang ditulis dalam HTML. (Rahmasari, 2019: 415).

II.2.9. *Hypertext Preprocessor* (PHP)

PHP merupakan singkatan dari "*Hypertext Preprocessor*". PHP adalah sebuah bahasa *scripting* yang terpasang pada HTML. Sebagian besar sintaknya mirip dengan bahasa pemrogramn C, Java, ASP dan Perl ditambah beberapa fungsi PHP yang spesifik dan mudah dimengerti. PHP digunakan untuk membuat tampilan *web* menjadi lebih dinamis, dengan PHP anda bisa menampilkan atau menjalankan beberapa *file* dalam 1 *file* dengan cara di *include* dan *require*. PHP itu sendiri sudah dapat berinteraksi dengan beberpa *database* walaupun dengan kelengkapan yang berbeda yaitu seperti DBM, MySQL, Oracle. (Rahmasari, 2019: 415).

PHP adalah bahasa pemrograman yang sering disisipkan ke dalam HTML. PHP sendiri berasal dari kata *Hypertext Preprocessor*. Sejarah PHP pada awalnya merupakan kependekan dari *Personal Home Page* (Situs personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari *web*. Bahasa pemrograman ini menggunakan sistem *server-side*. *Server-side programming* adalah jenis bahasa pemrograman yang nantinya *script*/program tersebut akan dijalankan/diproses oleh *server*. Kelebihannya

adalah mudah digunakan, sederhana, dan mudah untuk dimengerti dan dipelajari. (Sugijanto, dkk, 2020: 2).

II.2.9. *My Structure Query Language* (MySQL)

Definisi MySQL merupakan *software* RDMS (*Relational Database Management System*) yang dapat mengelola *database* dengan sangat cepat, dapat menampung data dalam jumlah sangat besar, dapat diakses oleh banyak pengguna dan dapat melakukan suatu proses secara sinkron atau bersamaan. (Rahmasari, 2019: 415).

MySQL merupakan *database engine* atau *server database* yang mendukung bahasa *database* pencarian SQL. MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS yang *multithread, multi-user*. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. (Sugijanto, dkk, 2020: 2).

II.2.8. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) yaitu satu kumpulan konvensi permodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem perangkat lunak yang terkait dengan objek. UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan system yang besar dan kompleks.

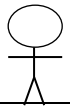
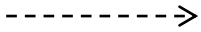
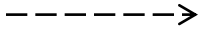
UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan. (Andikos, 2019: 39).

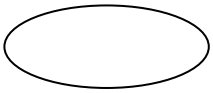
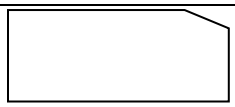
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut:

1. *Use Case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case diagram* dapat digambarkan dengan sumber-sumber pada Tabel II.1.

Tabel II.1. Simbol *Use Case*

Gambar	Nama	Keterangan
	*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
	*Depedency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Include*	Menspesifikasikan bahwa *use case* sumber secara eksplisit.
	*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
	Association	Apa yang mnghubungkan antara

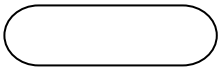
		objek satu dengan objek lainnya.
	*System*	Menspesifikan paket yang menampilkan sistem secara terbatas.
	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*.
	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

(Sumber: Andikos, 2019: 39)

2. Diagram Aktivitas (*Activity Diagram*)

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* dapat digambarkan dengan simbol-simbol seperti pada tabel II.2.

Tabel II.2. Simbol *Activity Diagram*

Gambar	Nama	Keterangan
	*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
	*Initial Node*	Bagaimana objek dibentuk atau diawali.

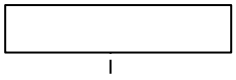
	*Activity Final*	Bagaimana objek dibentuk dan dihancurkan
	*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

(Sumber: Andikos, 2019: 39)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence Diagram* dapat digambarkan dengan simbol-simbol seperti pada Tabel II.3.

Tabel II.3. Simbol *Sequence Diagram*

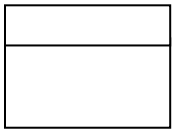
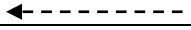
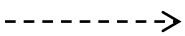
Gambar	Nama	Keterangan
	*Lifeline*	Objek *entity*, antarmuka yang saling berinteraksi.
	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktivitas yang terjadi.
	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktivitas yang terjadi.

(Sumber: Andikos, 2019: 39)

4. *Class Diagram* (Diagram Kelas)

Class Diagram adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. *Class diagram* dapat digambarkan dengan simbol-simbol seperti pada Tabel II.4.

Tabel II.4. Simbol *Class Diagram*

Gambar	Nama	Keterangan
	Generalization	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
	*Depedency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
	Assocation	Apa yang menghubungkan antara objek satu dengan objek lainnya.

(Sumber: Andikos, 2019: 39)