BAB II

KAJIAN PUSTAKA

A. Kajian Teoritis

1. Sistem Pakar

Sistem pakar dalam kecerdasan buatan dirancang untuk menyelesaikan masalah yang biasanya membutuhkan tenaga manusia dengan menerapkan pengetahuan dalam bidang tertentu. Sistem pakar biasa dikenal dengan subbidang *Artificial Intelligence* (AI) yang memanfaatkan spesialis tertentu untuk mengatasi masalah. Pakar sendiri merupakan individu yang mempunyai pengetahuan pada bidang profesi tertentu, khususnya orang yang mempunyai keterampilan atau pengetahuan khusus di bidang yang tidak dimiliki oleh kebanyakan orang (Tajrin et al., 2025:1).

Menurut Aldo et al., (2022:15), sistem pakar merupakan sistem yang meniru wawasan pakar dan kemampuan penalaran manusia ke dalam komputer sehingga dapat menangani permasalahan seperti seorang ahli. Bagian utama dari sistem pakar adalah mesin inferensi (*Inferensi Engine*) yang juga sebagai penerjemah aturan (*Rule Interpreter*) dalam proses pengambilan keputusan. Jadi dapat disimpulkan sistem pakar diartikan sebagai teknologi yang mampu memprediksi gejala atau indikasi suatu kondisi berdasarkan pengetahuan yang diperoleh dari pakar di bidang tertentu.

Komputer tidak dapat memperoleh pengetahuan secara mandiri melainkan bergantung pada inputan dari seorang pakar manusia.

Pengetahuan yang dimasukkan dalam sistem mempunyai batasan. Setelah pengetahuan berhasil dibentuk, teknik kecerdasan buatan dapat dimanfaatkan untuk memberikan kemampuan pada komputer dalam berfikir, bernalar, serta mengambil keputusan berdasarkan pengalaman dan pertimbangan yang di dasari oleh fakta (Panggabean & Wijaya, 2022:13).

2. Kanker Serviks

Kanker serviks merupakan kondisi dimana sel-sel ganas tumbuh tidak normal pada serviks. Kasus kanker serviks yang disebabkan oleh sel epitel pipih sekitar 90%, sedangkan 10% diantaranya dari jaringan endoserviks. Rentang usia penderita kanker serviks berkisar 40-50 tahun, tetapi kini sering ditemukan pada penderita usia muda sekitar 25-30 tahun (Digambiro, 2024:1).

Faktor risiko kanker serviks berkaitan dengan gaya hidup, terutama perilaku seksual. Salah satu faktornya adalah kebiasaan berganti pasangan, merokok, infeksi menular seperti virus herpes simpleks dan *Chlamydia* serta kurangnya kebersihan organ reproduksi. Faktor lain yang juga berpengaruh adalah riwayat keluarga dengan kanker serviks serta gaya hidup yang tidak sehat (Azlina & Firdausi, 2025:7).

Infeksi HPV (*Human Papilloma Virus*) pada fase awal biasanya tidak memiliki gejala yang spesifik, sehingga penderita tidak menyadari keberadaan kanker serviks dan tetap menjalani kehidupan normal. Namun, melalui tes skrining, dokter dapat menemukan sel-sel serviks abnormal yang disebut sebagai lesi prakanker. Bila tidak ditangani dan telah mencapai tahap

lanjut, kanker serviks dapat menimbulkan perdarahan setelah melakukan hubungan seksual, menstruasi tidak teratur, keputihan bercampur darah dan berbau, serta rasa nyeri di panggul dan saat buang air kecil (Banjarnahor et al., 2024:4).

Untuk mengurangi dampak dan risiko kematian akibat kanker serviks dapat dilakukan pencegahan yang meliputi pencegahan primer dengan mengurangi atau menghindari karsinogen (zat yang menyebabkan kanker), skrining dan deteksi dini untuk menemukan adanya kelainan termasuk dalam pencegahan sekunder, sehingga peluang untuk sembuh meningkat. Untuk pencegahan tersier berupa pengobatan dan melakukan pemeriksaan IVA Tes dan *Pap Smear* untuk mendeteksi dini kanker serviks (Jalilah & Prapitasari, 2021:134).

3. Certainty Factor (CF)

CF adalah metode untuk mengukur tingkat kepastian dalam sistem berbasis aturan. Nilai CF dihitung dengan mengurangi besarnya kepercayaan yang diperoleh dari suatu gejala (MB) dengan besarnya keraguan yang menyartainya (MD). Rumus *Certainty Factor* dapat dirumuskan seperti berikut:

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

Dalam hal ini, ukuran kepercayaan menunjukkan seberapa besar suatu bukti (*evidence*) mendukung hipotesis, sedangkan ukuran keraguan merepresentasikan sebaliknya (Ohyver et al., 2024:59).

Untuk *certainty factor* dengan premis tunggal, perhitungnnya sebagai berikut: $CF[H,E]_1 = CF[H] * CF[E]$. Sedangkan untuk CF berdasarkan aturan dengan kesimpulan yang serupa (*similarly concluded rules*) digunakan rumus berikut:

$$CF_{combine}[H,E]_{1,2} = CF[H,E]_1 + CF[H,E]_2 * [1 - CF[H,E]_1]$$

$$CF_{combine}[H,E]_{old3} = CF[H,E]_{old} + CF[H,E]_3 * [1 - CF[H,E]_{old}]$$

Dengan adanya rumus-rumus ini, metode CF dapat digunakan untuk menghitung tingkat keyakinan terhadap suatu hipotesis berdasarkan berbagai data pendukung yang diberikan (Pratama & Prasetyaningrum, 2024:159).

4. XAMPP

XAMPP merupakan aplikasi bebas yang dirancang untuk dapat berjalan di berbagai platform sistem operasi. Aplikasi ini mengintegrasikan beberapa program penting untuk menyediakan layanan server lokal. *XAMPP* adalah singkatan dari empat komponen sistem yang mencakup *Apache* sebagai web server, *MySQL* sebagai sistem manajemen basis data, serta interpreter untuk bahasa pemrograman *PHP* dan *Perl*. Program ini tersedia bebas dalam lisensi GNU (*General Public License*) dan berfungsi sebagai web server untuk menampilkan konten dinamis, untuk mendapatkan tampilan tersebut dapat diunduh langsung melalui web resminya (Suharyadi et al., 2023:9).

Komponen penting penyusun *XAMPP* meliputi *control panel* yang digunakan untuk mengelola *XAMPP* mulai dari mengelola DBMS

(*Database Manajement System*), menyisipkan file dan melakukan konfigurasi projek website dan fungsionalitas fitur lainnya. Folder *HTDocs* digunakan untuk menampung file dan dokumen yang akan diakses melalui browser dan ditampilkan dalam bentuk laman web dengan kapasitas penyimpanan menyesuaikan *hardisk storage*, serta *PhpMyAdmin* sebagai pengatur proses konfigurasi pada *MySQL* (Irwan et al., 2024:24).

5. MySQL

MySQL merupakan perangkat lunak manajemen basis data yang bersifat *open source* dan menawarkan dua pilihan versi, yaitu versi bebas pakai dan versi terbatas. Melalui lisensi *General Public License* (GPL), *MySQL* dapat digunakan untuk kebutuhan pribadi maupun komersial tanpa memerlukan biaya lisensi. Selain itu, *MySQL* termasuk dalam kategori *Relational Database Management System* (RDBMS), data disusun dalam bentuk tabel yang terdiri dari kolom dan baris (Namruddin et al., 2023:2).

MySQL adalah salah satu basis data yang paling populer digunakan sebagai pengembang aplikasi web yang membutuhkan pengelolaan data. Peran *MySQL* dapat dianggap sebagai pelaksana *query*, karena setiap perintah *SQL* yang dijalankan harus dituliskan melalui fungsi ini. Tanpa bantuan *MySQL*, *SQL* tidak dapat dioperasikan. *MySQL* sendiri termasuk dalam jenis sistem manajemen basis data relasional (RDBMS). Oleh karena itu, istilah seperti tabel, baris, dan kolom tetap digunakan dalam struktur MySQL (Kurniawan et al., 2024:19).

MySQL menyediakan fitur penting dalam pengolahan data, termasuk sistem *query* yang memungkinkan pengguna melakukan permintaan data menggunakan bahasa *SQL* (*Structured Query Language*). Dengan *SQL*, pengguna dapat mengambil, memperbarui, menghapus, atau menambahkan data ke dalam tabel dengan sintaks yang konsisten. *MySQL* juga menawarkan *scalability*, memungkinkan pengelolaan data dalam skala besar dengan respon cepat dan integrasi dengan sistem terdistribusi. Fitur keamanan *MySQL* mencakup autentikasi pengguna, enkripsi data, dan dukungan SSL/TLS untuk melindungi komunikasi antara aplikasi dan *database*. Selain itu, *MySQL* mendukung transaksi yang aman dan dapat dipulihkan, menjaga konsistensi data meskipun terjadi kegagalan. *MySQL* juga menyediakan replikasi data secara *real-time* ke server lain untuk mendukung *backup*, *load balancing*, dan ketersediaan tinggi dalam pemulihan bencana (Febriansyah & Awangga, 2023:28).

6. PHP

PHP adalah bahasa pemrograman yang berperan sebagai server terpusat, dimanfaatkan untuk merancang aplikasi berbasis website. Website yang dirancang sebaiknya memiliki karakteristik dinamis dimana memungkinkan tampilan konten berubah sesuai kondisi tertentu, misalnya menampilkan produk yang disesuaikan dengan riwayat pengguna yang dibuat oleh sistem dan karakteristik interaktif yang memungkinkan website memberikan respon terhadap perintah pengguna misalnya menampilkan hasil pencarian produk (Enterprise, 2022:2).

PHP atau *Hypertext Pre-Processor* adalah bahasa yang digunakan untuk menulis kode dalam pembuatan website, baik bersifat statis maupun dinamis. Sebagai bahasa *interpreted language*, *PHP* tidak memerlukan *compiler* untuk dieksekusi. Agar dapat dijalankan *PHP* membutuhkan web server yang telah terinstal. Selai itu, *PHP* merupakan bahas pemrogram yang dieksekusi di sisi server dimana hasil eksekusinya akan dikirimkan ke browser dalam bentuk *HTML*. Keunggulan lainnya, *PHP* bersifat gratis dan bebas, sehingga banyak digunakan dalam pengembangan website (Khozaimi, 2021:19).

7. CodeIgniter

CodeIgniter merupakan *framework PHP* yang digunakan untuk merancang aplikasi web dinamis yang bersifat bebas. *CodeIgniter* adalah *framework* yang menerapkan model *MVC* (*Model, View, Controller*) di dalamnya, untuk mempercepat proses pengembangan aplikasi website dinamis. *Framework* ini juga ringan dan cepat saat digunakan karena disertai dokumentasi lengkap dengan contoh implementasinya (Isnardi et al., 2021:25).

Framework CodeIgniter memiliki manfaat seperti mendukung pola *MVC* yang mencakup elemen seperti *routing*, *controller*, *model*, dan *filter*. Keuntungan dari penggunaan *framework* ini selain dapat mempercepat dan mempermudah proses pembangunan sistem, juga membantu dalam proses pemeliharaan sistem karena telah memiliki pola kerja tertentu yang terstruktur dan telah dilengkapi dengan fitur seperti validasi, *ORM*,

pagination, dukungan terhadap banyak basis data, *scaffolding*, manajemen sesi, hingga penanganan kesalahan, sehingga tidak perlu dibuat dari awal. Dibandingkan dengan *CMS*, *framework* memberikan fleksibilitas yang lebih tinggi dalam proses pengembangan karena tidak terlalu terikat pada struktur yang telah ditentukan sebelumnya (Taufani & Sujono, 2021:5).

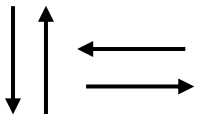
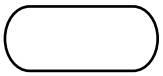
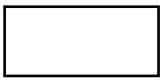
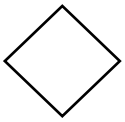
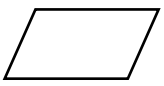
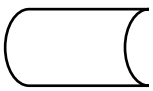
8. Flowchart

Flowchart merupakan bentuk penyajian visual untuk menggambarkan suatu proses atau alur kerja secara grafis. *Flowchart* biasa digunakan dalam proses perancangan atau analisa proses bisnis, algoritma maupun program komputer. Diagram ini tersusun dari simbol geometris seperti kotak, lingkaran, dan panah yang digunakan merepresentasikan langkah-langkah serta keputusan dalam suatu proses (Rusli & Sany, 2023:12).

Tujuan utama dari *flowchart* adalah membantu mendokumentasikan dan menyampaikan proses secara visual agar lebih mudah dipahami. *Flowchart* dianggap sebagai media interaksi yang efektif dalam tim pengembang dan pengguna karena dapat memperjelas alur proses, serta mendukung analisis dan pemecahan masalah dengan cara memvisualisasikan proses tersebut sehingga memudahkan identifikasi kendala dan pengembangan solusi (Samodra et al., 2025:39).

Setiap simbol *flowchart* memiliki makna yang berbeda dan digunakan sesuai fungsinya. Berikut adalah beberapa simbol *flowchart* yang sering digunakan.

Tabel 2.1 Simbol *Flowchart*

Simbol	Nama	Arti Simbol
	*Flow*	Untuk menghubungkan simbol. Simbol ini biasa disebut dengan *Connecting Line*.
	*On-Page Reference*	Untuk menyambungkan alur *flowchart* pada bagian berbeda dalam halaman yang sama.
	*Off-Page Reference*	Untuk menyambungkan bagian diagram alur yang terpisah halaman.
	Terminal	Untuk penanda dimulainya dan berakhirnya suatu rangkaian aktivitas.
	Process	Digunakan sebagai petunjuk pemrosesan apapun yang dilakukan oleh sistem.
	*Decision*	Digunakan untuk menandai proses pengambilan keputusan, dan sebagai penentu arah selanjutnya yang dilakukan oleh sistem.
	*Input/Output*	Digunakan untuk menunjukkan adanya masukan dan keluaran.
	*Disk Storage*	Digunakan untuk menyatakan input yang berasal dari disk atau disimpan ke disk.
	*Print/Document*	Untuk menggambarkan pemasukan berasal dari dokumen atau *output* yang perlu dicetak.

Sumber: (Kadang, 2021)

9. UML

a) Pengertian UML

UML atau *Unified Modeling Language* merupakan bahasa pemodelan yang umum digunakan dalam rekayasa perangkat lunak untuk mendokumentasikan, menspesifikasikan, serta membangun sistem. Dalam penggunaannya, UML menggabungkan berbagai praktik terbaik dari diagram hubungan entitas (ERD), bisnis model (*workflow*/alur kerja), pemodelan objek (*object modeling*) dan pemodelan komponen, sehingga dapat diterapkan pada seluruh tahapan siklus hidup perangkat lunak (Sari & Utami, 2021:107).

Diagram UML dapat dikelompokkan menjadi dua kategori utama, yaitu diagram struktur (statis) dan diagram perilaku (dinamis). Diagram struktur digunakan untuk menggambarkan bentuk statis dari *class* atau objek dalam suatu sistem, seperti *use case*, *class*, dan profil diagram. Sementara itu, diagram perilaku menunjukkan interaksi antarobjek atau fungsi dinamis dalam sistem, contohnya *sequence* dan *interaction overview diagram*. Dari berbagai jenis diagram yang tersedia, *use case*, *class*, *sequence*, dan *activity diagram* merupakan yang paling sering digunakan (Ardhana et al., 2025:73).

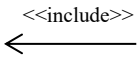
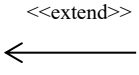
b) Diagram Model UML

1) *Use Case Diagram*

Menurut Suryantara (2024:23), diagram *use case* dalam UML digunakan untuk menggambarkan bagaimana pengguna atau aktor berkomunikasi dengan sistem dalam berbagai skenario dunia

nyata. Diagram ini berfungsi untuk memodelkan fungsionalitas sistem serta menunjukkan cara sistem digunakan dalam situasi tertentu. Simbol yang digunakan dalam *use case diagram* dapat dilihat seperti tabel berikut.

Tabel 2.2 Simbol *Use Case Diagram*

Simbol	Nama Simbol	Keterangan
	Aktor	Representasi dari pengguna, sistem, atau alat bantu ketika berkomunikasi.
	*Use case*	Penggambaran serta pola interaksi yang terjadi antara aktor dan sistem.
	Association	Gambaran mengenai hubungan antara aktor dan fungsi dalam *use case*.
	Generalization	Aktor turunan yang mewarisi semua interaksi dengan *use case*.
	*Include*	Menggambarkan bahwa *use case* hanya dapat berjalan jika fungsi dari *use case* lain dijalankan.
	*Extend*	Menggambarkan proses penambahan fungsi pada *use case* utama, jika diperlukan.

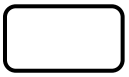
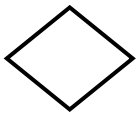
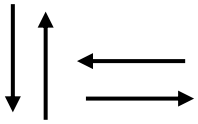
Sumber: Hanifa et al. (2024)

2) *Activity Diagram*

Menurut Permana & Purnomo (2021:65), *activity diagram* adalah suatu bentuk yang digunakan dalam UML untuk menjelaskan jalannnya aktivitas serta alur kerja dalam sebuah

sistem, mulai dari kondisi awal hingga keputusan yang mungkin terjadi pada kondisi akhir. Diagram ini membantu memvisualisasikan proses bisnis atau urutan aktivitas dalam suatu sistem secara lebih jelas. Selain itu, *activity diagram* merupakan bentuk khusus dari *state diagram*, di mana transisi antar *state* dipicu oleh selesainya suatu proses sebelumnya. Diagram ini tidak hanya menggambarkan interaksi internal dalam sistem, tetapi juga menunjukkan jalur aktivitas secara umum. *Activity diagram* memiliki struktur yang hampir sama dengan *flowchart* atau *Data Flow Diagram* dalam perancangan terstruktur dan sangat membantu dalam memahami keseluruhan proses sebelum diimplementasikan. Simbol yang digunakan pada *activity diagram* dijelaskan pada tabel berikut.

Tabel 2.3 Simbol *Activity Diagram*

Simbol	Nama Simbol	Keterangan
	Status awal	Menunjukkan titik awal dari suatu aktivitas.
	Aktivitas	Mewakili tindakan atau proses yang dilakukan.
	Percabangan / *decision*	Percabangan dalam alur proses berdasarkan kondisi tertentu (*if-else*).
	*Line Connector*	Untuk menghubungkan simbol.

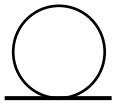
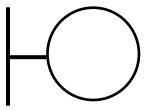
	Status akhir	Menunjukkan akhir dari suatu proses.

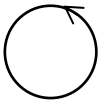
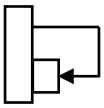
Sumber: (Rasiban et al., 2024)

3) Sequence Diagram

Menurut Sari (2021:176), *sequence diagram* menggambarkan hubungan antar objek dalam suatu sistem berdasarkan urutan waktu. Diagram ini memiliki dua arah, yaitu arah vertikal untuk menunjukkan waktu dan arah horizontal untuk objek yang saling berkomunikasi. Biasanya, *sequence diagram* digunakan untuk merepresentasikan rangkaian langkah-langkah yang terjadi sebagai respon terhadap suatu peristiwa hingga menghasilkan output tertentu. Diagram ini menunjukkan pemicu aktivitas, proses yang terjadi, serta perubahan dan output yang dihasilkan, dengan setiap objek memiliki lifeline vertikal. Daftar simbol pada *sequence diagram* dijelaskan pada tabel dibawah.

Tabel 2.4 Simbol *Sequence Diagram*

Simbol	Nama Simbol	Keterangan
	Kelas Entitas	Digunakan untuk menunjukkan struktur data yang berhubungan langsung dengan basis data.
	Kelas Antarmuka	Menunjukkan hubungan antara pengguna dan sistem. Tugasnya tidak menyimpan data namun hanya menampilkan dan berinteraksi.

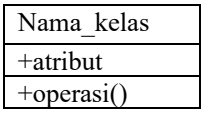
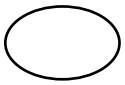
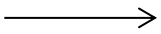
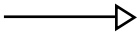
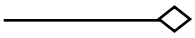
	Kelas Proses	Digunakan untuk mengatur alur proses dalam sistem. Simbol ini digunakan sebagai penghubung antara tampilan dan data.
	Rekursif	Menggambarkan suatu proses yang memanggil dirinya sendiri untuk menyelesaikan tugas hingga selesai.
	Aktivasi	Menggambarkan periode waktu ketika sebuah objek sedang menjalankan proses dan menunjukkan bahwa objek sedang bekerja.
	Garis hidup	Menunjukkan suatu objek sedang aktif dan berinteraksi dengan sistem.

Sumber: (Rasiban et al., 2024)

4) *Class Diagram*

Menurut Widiyawati et al. (2022:179), diagram kelas digunakan untuk menggambarkan susunan sistem melalui kelas, atribut, metode, dan hubungan antar kelas. Bersifat statis, diagram ini merepresentasikan entitas dalam domain bisnis maupun aspek teknis. Selain menunjukkan kumpulan kelas, *class diagram* juga mengilustrasikan hubungan seperti asosiasi, pewarisan (*inheritance*), serta atribut dan operasi dalam tiap kelas. Berikut adalah daftar simbol pada *class diagram*.

Tabel 2.5 Simbol *Class Diagram*

Simbol	Nama Simbol	Keterangan
	Kelas	Gambaran dari struktur sistem yang berisi atribut dan operasi.
	Antarmuka	Dalam PBO, konsep ini sama dengan antarmuka sistem.
	Asosiasi	Hubungan umum antar kelas yang biasanya disertai dengan konsep yang menjelaskan jumlah objek yang terlibat.
	Asosiasi berarah	Relasi antar kelas di mana satu kelas digunakan oleh kelas lain, sering kali juga mencantumkan jumlah hubungan.
	Generalisasi	Relasi antara kelas induk dan turunannya, serta sering digunakan saat aktivitas telah berakhir.
	Agregasi	Relasi yang menunjukkan bahwa suatu kelas terdiri dari bagian kelas lainnya, tetapi bagian tersebut bisa berdiri tanpa bergantung pada kelas utama.

Sumber: (Nurlita & Anggraini, 2023)

B. Kajian Empiris

Adapun penelitian sebelumnya yang relevan dengan pengembangan sistem pakar berbasis *certainty factor* antara lain sebagai berikut:

Penelitian Azzahra & Desiani (2023), merupakan penelitian yang berfokus pada pengembangan sistem pakar untuk mendiagnosis penyakit ginekologi menggunakan metode *Certainty Factor*. Sistem ini dirancang untuk

membantu pengguna mengenali kemungkinan penyakit berdasarkan gejala yang diinput, dengan menampilkan tingkat keyakinan dalam bentuk persentase. Penelitian ini menguji diagnosis terhadap lima jenis kanker ginekologi, yaitu kanker serviks, endometrium, vulva, tuba fallopi, dan ovarium, dengan akurasi bervariasi pada masing-masing jenis. Hasil penelitian menunjukkan bahwa sistem pakar dapat menjadi alternatif bagi individu yang ingin mendapatkan informasi awal sebelum berkonsultasi dengan tenaga medis.

Penelitian Prasetya et al. (2022) merupakan penelitian yang berfokus pada pengembangan sistem pakar untuk deteksi dini preeklamsia pada ibu hamil. Sistem ini membantu tenaga medis dan ibu hamil mendeteksi kemungkinan preeklamsia berdasarkan gejala yang dialami, menggunakan basis pengetahuan dari pakar dan menghitung tingkat kepastian diagnosis. Hasil pengujian menunjukkan akurasi 90%. Kemudian menguji sisi usabilitas menggunakan metode SUS yang menghasilkan skor 55 pada komponen antarmuka dan interpretasi hasil. Penelitian ini merekomendasikan pengembangan lebih lanjut dan integrasi dengan rekam medis elektronik untuk meningkatkan pengalaman pengguna serta mengintegrasikan sistem dengan rekam medis elektronik.

Penelitian Hardiyansa (2024) mengembangkan sistem pakar diagnosis penyakit kucing menggunakan metode *Certainty Factor* (CF). Sistem ini bertujuan membantu pemilik kucing dan dokter hewan dalam mendiagnosis penyakit secara cepat dan akurat tanpa perlu pertemuan langsung. Hasil pengujian menunjukkan sistem berfungsi dengan baik, memungkinkan pengguna mendapatkan diagnosis instan berdasarkan gejala yang dimasukkan.

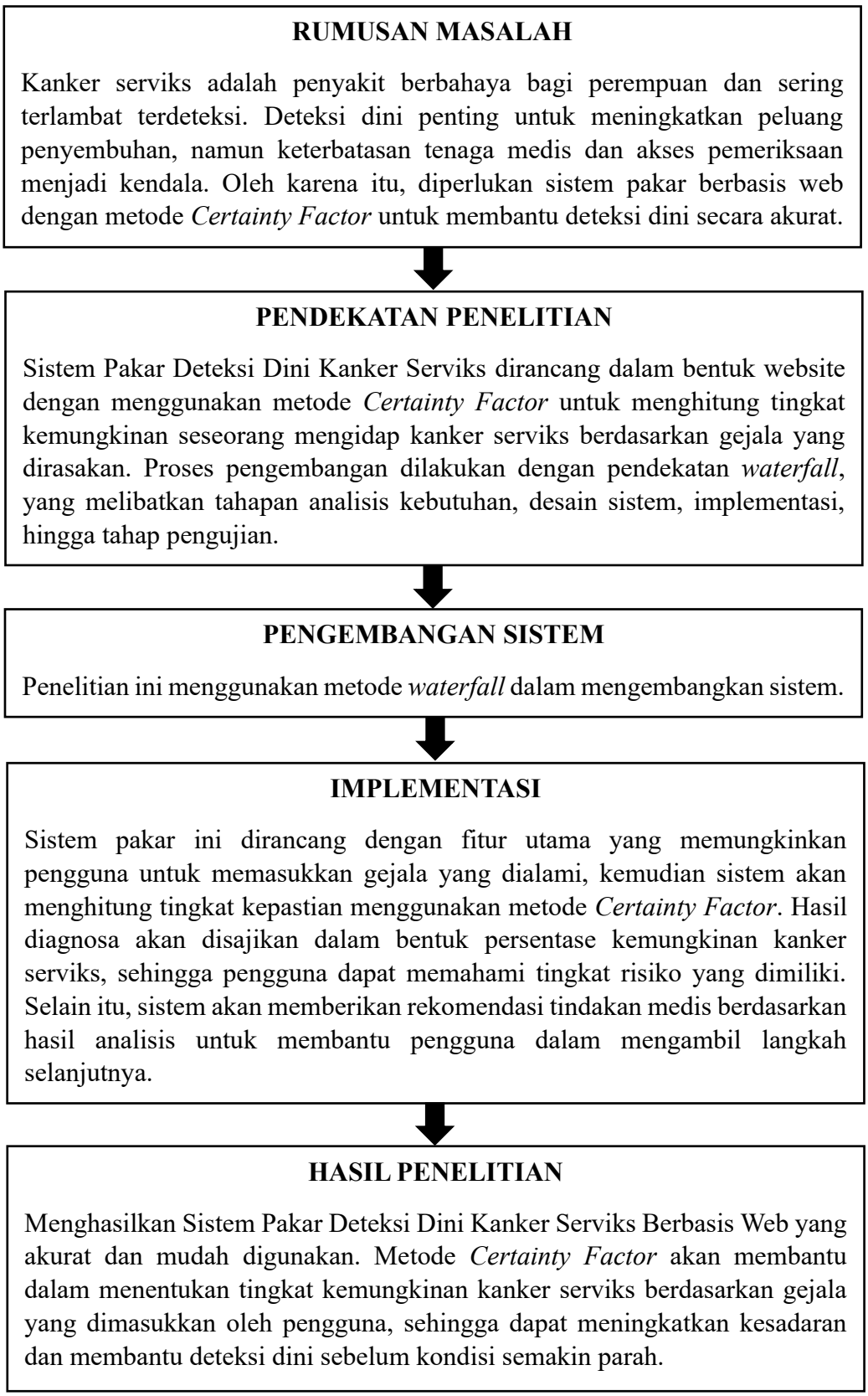
Namun, sistem masih terbatas pada penyakit kucing dan belum mendukung integrasi dengan rekam medis elektronik. Penelitian ini merekomendasikan pengembangan berbasis *mobile*, peningkatan antarmuka pengguna, serta integrasi dengan sistem rekam medis untuk meningkatkan pengalaman dan keakuratan diagnosis.

Penelitian Wati et al. (2023) mengembangkan sistem pakar berbasis web untuk diagnosis penyakit kehamilan menggunakan metode *Certainty Factor* (CF). Sistem ini bertujuan membantu ibu hamil dalam mendeteksi kemungkinan penyakit berdasarkan gejala yang dialami, sehingga diagnosis awal dapat dilakukan tanpa harus langsung berkonsultasi dengan tenaga medis. Hasil pengujian menunjukkan bahwa sistem memiliki tingkat kesesuaian 62,5% dengan hasil diagnosis pakar. Namun, masih terdapat 37,5% ketidaksesuaian, yang menunjukkan perlunya peningkatan dalam akurasi sistem. Penelitian ini merekomendasikan pengembangan lebih lanjut, seperti peningkatan basis pengetahuan, optimasi perhitungan CF, serta integrasi dengan sistem rekam medis elektronik untuk meningkatkan keakuratan dan kemudahan akses bagi pengguna.

C. Kerangka Berpikir

Kerangka berpikir ini dibuat untuk menunjukkan langkah-langkah dalam merancang dan membangun sistem pakar deteksi dini kanker serviks berbasis web dengan menggunakan metode *Certainty Factor*. Tujuan dari penelitian ini adalah untuk membantu pengguna mengenali kemungkinan gejala kanker serviks sejak dini secara mandiri. Masalah utama yang diangkat adalah

masih banyaknya kasus kanker serviks yang terlambat terdeteksi karena kurangnya informasi dan keterbatasan tenaga medis. Oleh karena itu, dibutuhkan sebuah sistem yang bisa membantu masyarakat melakukan deteksi awal dengan mudah dan cepat. Penelitian ini menggunakan metode pengembangan *waterfall*, yaitu dimulai dari analisis kebutuhan, perancangan sistem, pembuatan program (implementasi), hingga tahap pengujian. Sistem akan bekerja dengan meminta pengguna memilih gejala yang dialami, lalu sistem menghitung kemungkinan terkena kanker serviks berdasarkan gejala tersebut menggunakan metode *Certainty Factor*. Hasil dari sistem ini adalah informasi tingkat kemungkinan (dalam persen) serta saran atau rekomendasi untuk langkah selanjutnya. Dengan begitu, pengguna bisa lebih waspada dan segera melakukan pemeriksaan ke tenaga medis jika diperlukan. Berikut adalah kerangka berpikir pada penelitian ini:



Gambar 2.1 Kerangka Berpikir