BAB II
KAJIAN PUSTAKA

A. Kajian Teoritis

1. Sistem Pakar

Sistem pakar merupakan aplikasi perangkat lunak yang termasuk dalam bidang *artificial intelligence*, dirancang untuk meniru cara berpikir dan bernalar manusia, khususnya para ahli di bidang tertentu (Wahyuni & Nirsal, 2024:1). Dengan memanfaatkan basis data pengetahuan dan mesin penalaran, sistem mampu memberikan jawaban, saran, atau diagnosis yang hampir sama seperti yang diberikan oleh seorang pakar (ahli) (Rahayu et al., 2025:1). Struktur dasar sistem pakar terdiri dari dua elemen, yaitu basis pengetahuan yang menyimpan informasi, serta mesin penalaran yang digunakan untuk menyimpulkan (Tajrin et al., 2025:2).

Dalam perkembangannya, sistem pakar menggabungkan basis pengetahuan yang didasarkan pada bukti serta metode pencarian untuk memecahkan permasalahan yang memerlukan keahlian dari seseorang di bidang tertentu. Sistem pakar terus berubah mengikuti keperluan dan dapat memberikan bantuan kepada manusia dalam melakukan pekerjaan mirip dengan seorang profesional, tetapi tidak menggeser posisi seorang ahli (Mayatopani et al., 2022). Contoh penerapan sistem pakar mencakup bidang seperti diagnosis medis, perencanaan produksi, pengelolaan rantai pasok, peramalan cuaca, serta berbagai bidang lainnya dimana keputusan yang

diambil berdasarkan pada pengetahuan khusus yang sangat penting (Harianto & Azahari, 2023).

Kesimpulan berdasarkan dari kelima sumber yang telah disebutkan yaitu, sistem pakar merupakan jenis perangkat lunak kecerdasan buatan yang dirancang guna meniru cara berpikir dan kemampuan berpikir logis seorang ahli. Sistem ini memiliki dua elemen penting, yaitu basis pengetahuan dan mesin penelusuran untuk membuat diagnosis atau keputusan. Teknologi ini terus berkembang dan bisa digunakan di berbagai bidang seperti diagnosis medis hingga memperkirakan cuaca, sehingga membantu pekerjaan manusia tanpa menggantikan peran ahli.

2. Diabetes Melitus (DM)

Diabetes Melitus merupakan kondisi kesehatan yang menunjukkan gejala utama yaitu kadar gula dalam darah yang meningkat (*hiperglikemia*). Kadar gula darah yang tinggi terjadi karena produksi hormon insulin oleh kelenjar pankreas berkurang atau penurunan sensitivitas dari hormon insulin. Dampaknya, zat gula tersebut tidak bisa meresap ke jaringan tubuh dan tetap berada dalam darah (Murdiyanti et al., 2022:4). Berdasarkan Perhimpunan Endokrinologi Indonesia tahun 2021, hasil pemeriksaan glukosa darah dikelompokkan menjadi tiga status yaitu Non Diabetes (normal), Prediabetes, dan Diabetes (PERKENI, 2021:12). Seseorang yang menderita Diabetes Melitus biasanya mulai dari kondisi prediabetes, yaitu kondisi dimana kadar gula darahnya belum mencapai tingkat diabetes, tetapi sudah cukup tinggi dibandingkan batas non diabetes (normal) atau disebut juga sebagai *intermediate hyperglicemia* (Novriyanti et al., 2025).

Mendeteksi prediabetes serta elemen-elemen risiko yang terkait sangat krusial agar individu dapat mengubah pola hidup sebelum kondisi ini berkembang menjadi Diabetes Melitus atau masalah kardiovaskular yang lain (Kusuma et al., 2023). Jika dibiarkan secara terus menerus, diabetes bisa memicu masalah serius pada bagian ginjal, hati, dan organ tubuh yang lain, dan mengganggu sistem pencernaan, saraf, dan kandung kemih. Prevalensi diabetes terus meningkat hingga menyebabkan kenaikan mortalitas di kalangan lansia (Pratiwi et al., 2024:1).

Berdasarkan kelima sumber di atas dapat disimpulkan bahwa Diabetes Melitus merupakan suatu kondisi yang disebabkan oleh masalah pada hormon insulin yang dibagi menjadi tiga kategori yaitu Non Diabetes, Prediabetes, dan Diabetes. Kondisi ini biasannya dimulai dari tahap prediabetes, sehingga deteksi dini dan perubahan gaya hidup sangat diperlukan agar tidak berkembang menjadi diabetes. Jika dibiarkan, penyakit ini dapat menimbulkan komplikasi organ yang serius hingga menjadi salah satu penyebab kematian.

3. *Certainty Factor* (CF)

Certainty Factor (CF) dikemukakan oleh Shortliffe bersama Buchanan pada tahun 1975 agar bisa menangani ketidakyakinan dalam cara berpikir pakar. Seorang pakar, seperti seorang dokter, terkadang merasa ragu ketika menganalisis informasi yang ada dengan menggunakan kata-kata seperti "mungkin", "besar kemungkinannya", atau "hampir pasti". Untuk mengatasi masalah ini, digunakan metode Faktor Kepastian untuk menunjukkan tingkatan kepercayaan pakar pada permasalahan yang ada (Utari & Sudrajat, 2022). Keunggulan dari metode *Certainty Factor* yaitu kemampuannya untuk

mengevaluasi hal yang jelas maupun yang samar dalam proses pemilihan keputusan dalam sistem pakar untuk diagnosis penyakit (Cantika et al., 2023).

Menurut (Pamungkas et al., 2023) *Certainty Factor* bisa didefinisikan sebagai berikut:

$$CF(H,E) = MB(H,E) - MD(H,E) \quad (1)$$

Keterangan :

MB : Ukuran keyakinan adalah indikator peningkatan kepercayaan terhadap hipotesis H yang dipengaruhi oleh gejala E.

MD : Ukuran keraguan adalah indikator peningkatan ketidakpercayaan terhadap hipotesis H yang dipengaruhi oleh gejala E.

E : *Evidence* (peristiwa atau data).

H : Hipotesis (dugaan).

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

$$CF(H,E)1 = CF(H) * CF(E) \quad (2)$$

Keterangan:

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

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

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

Setelah nilai CF(H,E) dihitung, jika ada lebih dari satu gejala yang terdeteksi, nilai-nilai CF(H,E) ini akan digabungkan dengan nilai CF yang telah dihitung sebelumnya menggunakan persamaan berikut (Ayal & Susilawati, 2024):

$$CFcombine = CF(H,E)1 + CF(H,E)2 * [1-CF(H,E)1] \quad (3)$$

$$CFcombine = CFold + CF(H,E)3 * [1-CFold] \quad (4)$$

Keterangan:

CFcombine : Hasil penghitungan penggabungan korelasi.

CF(H,E) : Hasil penghitungan dari H dan E (CF gejala).

CFold : Hasil penghitungan CFcombine yang telah dilakukan sebelumnya.

Sebagai langkah akhir dalam menentukan tingkat keyakinan sistem terhadap hasil diagnosis, nilai CFcombine yang telah dihitung akan dikonversikan ke dalam bentuk persentase menggunakan rumus berikut:

$$CFpersentase = CFcombine * 100 \quad (5)$$

Keterangan:

CFpersentase : Hasil pembulatan dari CFcombine.

CFcombine : Hasil perhitungan penggabungan korelasi.

Kesimpulan berdasarkan dari kelima sumber yang telah disebutkan yaitu Metode *Certainty Factor* (CF) digunakan dalam mengatasi ketidakpastian dalam sistem pakar. Pada tahap awal dilakukan dengan mencari nilai CF(HE) yang diperoleh dari hasil perkalian antara nilai *Certainty Factor* hipotesis dari pakar dan nilai *Certainty Factor evidence* dari pengguna. Nilai tersebut kemudian dijumlahkan menggunakan rumus CF combine apabila terdapat banyak gejala, lalu dikalikan 100 untuk mendapatkan persentase tingkat keyakinan akhir.

4. Website

Website merupakan sekumpulan halaman yang terdapat di sebuah domain di internet, dirancang dengan maksud tertentu, saling berkaitan, dan dapat diakses oleh banyak orang melalui halaman utama menggunakan browser dengan alamat URL website (Wahyuningtyas & Chusnah, 2021:7). Halaman-halaman tersebut umumnya dibuat menggunakan HTML dan dapat digabungkan dengan CSS serta JavaScript agar tampilannya lebih bagus dan interaktif. Website berperan sebagi sarana untuk menyimpan informasi, berkomunikasi, serta menyediakan layanan digital yang dapat digunakan oleh pengguna dari berbagai tempat (Santoso, 2026:1). Website sendiri adalah sekumpulan halaman yang menyajikan berbagai informasi dalam bentuk tulisan, gambar, video, dan lainnya yang dapat diakses kapan saja, dari mana saja, dan oleh siapa saja asalkan terhubung dengan internet (Rahmi et al., 2023).

Website dibagi menjadi dua kategori, yaitu website statis dan website dinamis. Website statis adalah jenis website dimana jika ingin mengubah isi dalam website tersebut, harus diubah secara manual dengan mengedit kodenya. Umumnya, halaman-halaman dari website statis masih memakai HTML, dan data tidak disimpan dalam suatu *database*. Di sisi lain, website dinamis memungkinkan pengubahan kontennya dengan mudah tanpa harus mengakses kode sumber, serta bisa di *update* secara berkala (Widia & Asriningtias, 2021:3). Kelebihan website dinamis adalah interaktif, konten bisa diperbarui dengan mudah (hanya perlu mengubah data di *database*, tanpa harus mengedit kode), dan bisa disesuaikan berdasarkan kebutuhan pengguna masing-masing.

Contohnya mencakup *platform* belanja *online* (*e-commerce*), situs berita, media sosial (Facebook, Instagram), aplikasi perbankan yang dapat diakses secara *online*, serta sistem pendidikan berbasis daring (LMS) (Rianto, 2025:5).

Kesimpulan berdasarkan dari kelima sumber yang telah disebutkan yaitu, website merupakan sekumpulan halaman internet yang memuat informasi interaktif dan saling terhubung, serta bisa diakses oleh siapa pun melalui URL. Secara fungsi, website berperan sebagai tempat menyimpan data, komunikasi, serta fasilitas layanan digital yang fleksibel bagi pengguna. Berdasarkan sifat kontennya, website dibagi menjadi dua macam yaitu website statis dimana perubahan harus dilakukan secara manual melalui *source code* dan website dinamis yang dapat diperbarui dengan mudah menggunakan *database*.

5. Laragon

Laragon merupakan *software* yang dirancang untuk mengembangkan program atau aplikasi di lingkungan server lokal, yaitu pada komputer. Salah satu fitur unggulan Laragon adalah kelengkapan fiturnya yang melebihi aplikasi serupa, serta kemudahan dalam penggunaannya karena menawarkan antarmuka berbasis GUI. Dengan Laragon, dapat membangun aplikasi dengan Git dan mengelola basis data melalui php MyAdmin. Selain itu, Laragon mendukung pengembangan aplikasi yang menggunakan Node. js, MongoDB, Django, Flask, PostgreSQL, Ruby, Java, atau Go. Laragon dapat dioperasikan di berbagai sistem operasi seperti Windows, MacOS, dan Linux (Sitorus et al., 2024:1).

Laragon adalah suatu *universal development environment* yang fungsinya hampir sama dengan XAMPP, tetapi lebih cepat, ringan, dan mudah digunakan

(Hanief et al., 2022:8). Laragon memiliki kemampuan untuk menjadikan komputer sebagai server lokal untuk tujuan pengembangan website (Hendrawansyah et al., 2026). Software ini cocok untuk pemula karena proses instalasi dan pengaturan folder proyek menjadi lebih mudah (Yel, 2025:6).

Kesimpulan dari keempat sumber diatas yaitu, Laragon adalah sebuah perangkat lunak *development environment universal* yang berfungsi mengubah komputer menjadi server lokal untuk pengembangan website. Aplikasi ini lebih cepat, ringan, mudah digunakan melalui tampilan GUI, dan sangat cocok untuk pemula. Selain itu, Laragon juga mendukung pengelolaan *database* serta berbagai teknologi pemrograman yang banyak digunakan seperti Node.js, Python, dan Git yang bisa digunakan di berbagai sitem operasi.

6. MySQL

MySQL merupakan Sistem Manajemen Basis Data Relasional (RDBMS) yang bersifat terbuka dan sering digunakan untuk mengorganisir dan menyimpan informasi (Hendrawansyah et al., 2026). MySQL beroperasi dengan menggunakan bahasa SQL (*Structural Query Language*). Ini menunjukkan bahwa MySQL merupakan acuan utama dalam pemanfaatan *database* di seluruh dunia untuk mengelola dan memproses data. Secara umum, instruksi yang paling sering diterapkan dalam MySQL meliputi *Select, Insert, Update*, dan *Delete*. Di samping itu, MySQL juga menawarkan instruksi untuk membangun *database* atau *index*, serta menambah atau menghapus data yang dapat dijalankan secara langsung dalam sistem operasi (Prahasti et al., 2022).

MySQL diciptakan pada awal tahun 1990-an dan telah terpasang lebih dari 10 juta kali. MySQL adalah sistem pengelola basis data yang paling umum

digunakan untuk server web. Saat ini, MySQL adalah teknologi yang sudah matang dan mendukung berbagai situs internet yang paling banyak dikunjungi (Silalahi, 2022:1). MySQL tersedia dalam dua jenis lisensi, yaitu *Free Software* dan *Shareware*. Versi yang umum digunakan dari MySQL adalah MySQL *Free Software* yang diatur oleh Lisensi GNU/GPL (*General Public License*) (Primadewi & Nugroho, 2022:4). MySQL sering digunakan pada aplikasi web, khususnya dalam pembuatan aplikasi yang menggunakan bahasa pemrograman PHP dan Apache. MySQL banyak digunakan oleh para programmer karena kualitasnya yang baik dan dapat digunakan secara gratis (Aziz et al., 2025).

Kesimpulan dari kelima sumber diatas menunjukkan bahwa, MySQL merupakan sistem manajemen basis data relasional yang menggunakan SQL dan merupakan *software* sumber terbuka yang banyak dipakai untuk mengatur data di server. Perangkat lunak yang tersedia tanpa biaya ini memfasilitasi pengolahan data dengan mudah melalui perintah-perintah standar seperti *select, insert, update*, dan *delete*. Karena keandalannya yang sudah teruji dan kualitas yang baik, MySQL menjadi pilihan utama bagi para pengembang dalam menciptakan berbagai aplikasi web.

7. PHP

PHP merupakan singkatan dari "*Hypertext Preprocessor*," adalah jenis bahasa pemrograman yang digunakan untuk membangun website dinamis. PHP dikenal sebagai bahasa *script* berbasis server, yang berarti bahwa instruksi yang ditulis akan sepenuhnya dieksekusi oleh server, tetapi bisa menyertakannya dalam bahasa HTML biasa. Program ini bisa berjalan sendiri atau diintegrasikan sintaks HTML, sehingga dapat ditampilkan secara langsung

melalui sintaks HTML tersebut (Zakir & Maiyana, 2023:1). PHP pada awalnya merupakan proyek kecil yang akhirnya berkembang menjadi bahasa yang sangat populer untuk membuat aplikasi web. Popularitasnya meningkat karena terbukti sangat mudah sangat mudah digunakan. Rasmus Lerdof adalah orang yang menciptakan PHP pada tahun 1994 (Singh et al., 2024:1).

PHP bersifat gratis dan *open source*. Secara umum, PHP digunakan untuk membuat bagian logika *backend* dalam pengembangan website. Di samping itu, PHP juga dapat digunakan untuk menciptakan aplikasi antarmuka pengguna (GUI) menggunakan PHP-GTK. Selain itu, PHP saat ini digunakan untuk memproses algoritma pembelajaran mesin. Saat ini, PHP tetap menjadi bahasa pemrograman berbasis server yang paling banyak digunakan dibandingkan dengan bahasa pemrograman lainnya (Sholihin & Nurjaya, 2024:1). PHP merupakan sebuah bahasa pemrograman yang berfungsi untuk menangani basis data dan konten website, sehingga menjadikannya lebih dinamis. Selain itu, PHP dapat dipadukan dengan HTML untuk menghasilkan halaman website yang lebih interaktif (Arafat et al., 2022). PHP adalah bahasa pemrograman yang digunakan di sisi server dan sering dipakai untuk membuat website yang bisa berubah-ubah sesuai kebutuhan pengguna (Fadilah et al., 2026:18).

Berdasarkan kelima sumber diatas dapat disimpulkan bahwa, PHP merupakan bahasa pemrograman berbasis server dan terbuka yang digunakan untuk membuat website dinamis. Perintah didalamnya dijalankan sepenuhnya oleh server dan dapat diintegrasikan dengan kode HTML untuk membuat halaman yang lebih interaktif. Selain mengelola logika *backend* dan basis data,

bahasa ini juga digunakan untuk membuat tampilan antarmuka pengguna hingga memproses algoritma pembelajaran mesin.

8. Laravel

Laravel merupakan sebuah kerangka kerja PHP yang gratis digunakan, diciptakan oleh Taylor Otwell dan pertama kali dikeluarkan pada tahun 2011 menggunakan lisensi MIT (Melyanti et al., 2023:2). Sejak pertama kali diluncurkan, Laravel berhasil mendapat banyak perhatian dari para pengembang *software* dan dengan cepat menjadi salah satu kerangka kerja PHP yang sangat terkenal. Laravel terkenal karena menyediakan banyak fitur yang berguna dalam pengembangan aplikasi, seperti sistem *routing* yang bisa diatur dengan fleksible, ORM yang kuat untuk mengelola *database*, serta *engine templating* yang mudah digunakan (Zaidan et al., 2023:1).

Laravel dibuat dengan menggunakan pola *Model-View-Controller* (MVC) yang memudahkan dalam membuat aplikasi web, karena memisahkan bagian logika aplikasi dari bagian tampilannya (Febriansyah et al., 2023:8). MVC adalah pendekatan dalam pembuatan *software* yang memisahkan bagian yang mengatur logika aplikasi dari bagian yang menampilkan tampilan aplikasi. MVC membagi aplikasi menjadi bagian-bagian seperti pengolahan data, *controller*, dan *user interface* (Sweetania & Herawati, 2022). Laravel juga menyediakan berbagai fitur yang tidak selalu tersedia dalam semua kerangka kerja. Sebagai sebuah kerangka kerja modern, Laravel memudahkan pembuatan situs web dengan berbagai fitur terkini, seperti proses otentikasi mutakhir (Pratiwi et al., 2024:5).

Berdasarkan kelima sumber diatas dapat disimpulkan bahwa, Laravel adalah *framework* PHP *open-source* yang banyak digunakan, yang diciptakan oleh Taylor Otwell pada tahun 2011. Kerangka kerja modern ini menggunakan pola arsitektur *Model-View-Controller* (MVC) agar logika aplikasi terpisah dengan tampilan antarmuka. Popularitasnya didukung oleh fitur-fitur modern seperti sistem *routing* yang fleksibel, ORM yang sangat efektif, dan proses autentikasi yang aman.

9. *Black Box Testing*

Black Box Testing merupakan pengujian sistem atau aplikasi bekerja dari perspektif pengguna, tanpa harus memahami bagian dalam atau cara pembuatan sistem tersebut. Secara sederhana, *Black Box Testing* hanya mengevaluasi apakah sistem mampu memberikan *output* atau hasil setelah menerima *input* dalam bentuk informasi atau instruksi (Hozairi et al., 2024:9). Keuntungan dari *Black Box Testing* yaitu tidak membutuhkan kode, jadi tidak membutuhkan akses kebagian dalam program (Fikri & Voutama, 2023).

Dalam *Black Box Testing*, berbagai teknik dan pendekatan agar *software* bekerja sesuai dengan apa yang diharapkan, tanpa perlu melihat bagian dalamnya. Teknik ini mengarahkan pada pemeriksaan *input* dan *output* sistem untuk mengidentifikasi kesalahan fungsional, validasi data, serta pengalaman pengguna. Dengan memahami metode yang tepat, pengujian dapat dilakukan dengan cara yang lebih efektif untuk mendeteksi *bug* dan memastikan perangkat lunak berjalan baik sesuai yang direncanakan (Fauzi & Yunial, 2025:118). *Black Box Testing* merupakan metode yang digunakan untuk

menguji hasil desain sistem, dengan perhatian utama pada bagian fungsi sistem tersebut (Hendrawansyah et al., 2026).

Berdasarkan keempat sumber diatas dapat disimpulkan bahwa, *Black Box Testing* merupakan cara menguji sistem yang memfokuskan pada bagaimana sistem berfungsi dari perspektif pengguna. Pengujian ini memeriksa apakah *input* dan *output* untuk menemukan kesalahana serta memastikan data benar tanpa perlu mengakses *source code* atau struktur program. Dengan berbagai teknik dan pendekatan yang tepat, metode ini sangat efektif dalam menemukan kesalahan dan memastikan aplikasi berjalan sesuai dengan spesifikasi yang diharapkan.

10. *System Usability Scale* (SUS)

System Usability Scale (SUS) merupakan cara untuk mengetahui seberapa mudah dan nyaman pengguna merasa saat menggunakan sistem atau aplikasi tertentu. SUS dibuat oleh John Brooke pada tahun 1986 dan sudah digunakan dalam banyak penelitian serta penilaian produk. SUS berisi 10 pernyataan yang meliputi berbagai aspek mengenai kegunaan, dan responden diminta untuk memberikan penilaian dengan menggunakan skala likert. SUS menghasilkan skor total dari 0 hingga 100 yang memudahkan dalam memahami tingkat kegunaannya (Sa'adah et al., 2024). Kelebihan dari SUS salah satunya yaitu, proses penilaian lebih sederhana dan mudah dipahami oleh orang yang menjawab, bisa menunjukkan hasil yang maksimal bahkan dengan sampel yang tidak terlalu banyak, serta mampu membedakan baik atau tidak sebuah aplikasi ketika digunakan. SUS memiliki metode perhitungan yang jelas dalam proses evaluasi, sehingga hasil evaluasi yang di

dapat diharapkan memiliki tingkat ketepatan yang tetap terjaga (Fatmawati, 2021).

Usability Testing adalah cara untuk mengevaluasi sejauh mana pengguna dapat dengan mudah mengakses dan menggunakan *software*. Tujuannya agar aplikasi atau produk menjadi lebih mudah digunakan dengan cara menganalisis, mengenali masalah, mengukur seberapa efisien, mudahnya penggunaan, serta tingkat kepuasan pengguna (Candana et al., 2024). Metode SUS memiliki langkah-langkah yang digunakan untuk mengevaluasi sistem. Langkah pertama yaitu menyusun pertanyaan yang berisi 10 pertanyaan dan 5 jawaban. Jawaban bisa berupa setuju hingga tidak setuju. Kemudian langkah selanjutnya adalah mengirimkan daftar tersebut kepada para responden (Susila & Arsa, 2023).

Kesimpulan dari keempat sumber diatas yaitu, *System Usability Scale* (SUS) merupakan alat pengujian yang digunakan untuk menilai sejauh mana pengguna merasa nyaman, efisien, dan mudah dalam menggunakan perangkat lunak. Pengujian ini dilakukan dengan memberikan kuesioner yang berisi 10 pertanyaan menggunakan skala likert untuk mengevaluasi berbagai aspek kegunaan sistem secara teratur dan terorganisir. Cara ini membuat penilaian menjadi lebih gampang dan tepat, metode ini memberikan skor akhir berupa angka 0 sampai 100 yang memudahkan dalam memahami tingkat kelayakan suatu aplikasi.

11. *Flowchart*

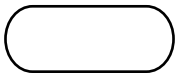
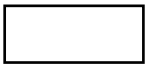
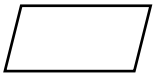
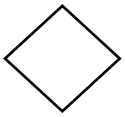
Flowchart adalah gambaran visual yang menunjukkan langkah-langkah serta urutan cara kerja dari suatu program. Umumnya membantu

menyelesaikan masalah yang khususnya memerlukan pembelajaran dan evaluasi lebih lanjut (Tuasamu et al., 2023). Bagan alir program adalah gambar yang menunjukkan dengan jelas setiap tahap dalam proses kerja program. Bagan alir program dibuat berdasarkan pengembangan dari bagan alir sistem (Smrti et al., 2023).

Diagram alir, juga dikenal sebagai *flowchart* adalah cara untuk menggambarkan sebuah proses, sistem, atau alur kerja dengan menggunakan berbagai simbol yang dihubungkan oleh garis dan tanda panah. Diagram alir dapat membuat pemahaman tentang suatu proses lebih mudah, karena menunjukkan setiap tahapan yang dilakukan, serta membantu menjelaskan informasi yang rumit agar orang lain bisa memahami dengan lebih baik (Suwanda et al., 2025:82). Tujuan utama membuat *flowchart* adalah untuk menampilkan urutan langkah-langkah yang menjelaskan cara menyelesaikan suatu masalah dengan menggunakan operasi aritmatika dan/atau logika yang dapat diinstruksikan kepada komputer (Khairunnisa et al., 2023:89).

Berdasarkan keempat sumber diatas dapat disimpulkan bahwa, *flowchart* atau diagram alir merupakan gambar visual yang menjelaskan tahapan dan proses kerja dalam suatu program dengan menggunakan simbol-simbol yang menunjukkan arah alir. Metode ini membantu membuat alur kerja yang rumit lebih mudah dipahami dan mempermudah evaluasi masalah dengan cara yang lebih mendalam. Tujuan utamanya adalah menjelaskan langkah-langkah penyelesaian masalah menggunakan logika dan perhitungan matematika yang dapat diinstruksikan langsung kepada komputer.

Tabel 2. 1 Simbol *Flowchart*

No	Gambar	Simbol	Keterangan
1.		Terminal	Simbol yang berfungsi untuk menunjukkan awal dan akhir dari serangkaian proses yang berkaitan dengan komputer.
2.		Garis Alir	Simbol yang berfungsi untuk menghubungkan antar simbol
3.		Pemrosesan Komputer	Simbol yang berfungsi untuk menggambarkan suatu proses yang dilakukan oleh sistem komputer.
4.		*Input-Output*	Simbol yang berfungsi untuk menggambarkan operasi *input/output*
5.		Keputusan	Simbol yang berfungsi untuk menggambarkan posisi di mana keputusan harus diambil untuk menentukan tindakan selanjutnya.
6		*Input/Output* Dokumen	Simbol yang berfungsi ketika masukan berasal dari dokumen dan keluaran menuju dokumen.

Sumber : (Khairunnisa et al., 2023:90).

12. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) digunakan untuk merancang dan memodelkan sistem perangkat lunak, termasuk proses bisnis. UML menggabungkan berbagai teknik pemodelan yang sudah dipakai sebelumnya kedalam satu sistem kerja yang konsisten dan sesuai standar (Wicaksono, 2023:21). Pada tahun 1997, versi pertama UML secara resmi diperkenalkan kepada masyarakat umum. Metodologi ini memiliki standar pada setiap diagramnya. Setelah itu, pada versi UML 2.0 tahun 2005 terjadi perubahan yang menekankan pendekatan berorientasi objek. Hingga tahun 2017, versi

UML 2.5 terus diperbarui agar bisa menyesuaikan diri dengan berbagai masalah dan perkembangan teknologi terkini (Andriyanto, 2022:2).

UML merupakan sebuah bahasa yang menggunakan gambar untuk menggambarkan, menentukan, membuat, dan mendokumentasikan sebuah sistem pengembangan perangkat lunak berbasis pendekatan berorientasi objek (Sumirat et al., 2023:73). Pemodelan perangkat lunak dapat membantu menyederhanakan langkah selanjutnya dalam pengembangan sistem informasi agar lebih teratur dan terencana (Suwanda et al., 2025:106). UML tidak hanya berupa bahasa pemrograman visual, tetapi dapat digunakan langsung dengan berbagai bahasa pemrograman lainnya, seperti Java dan C++. *Visual Basic* bahkan dapat digunakan langsung dalam basis data berorientasi objek (Saputra et al., 2023:140).

Berdasarkan kelima sumber diatas dapat disimpulkan bahwa, *Unified Modeling Language* (UML) merupakan bahasa pemodelan visual standar yang digunakan untuk membangun dan menjelaskan pengembangan *software* berbasis objek. Metodologi yang terus diperbarui sesuai dengan perkembangan teknologi ini membantu menyederhanakan proses pembuatan sistem sehingga lebih teratur dan terencana. Keunggulan utamanya adalah kemampuannya dapat dihubungkan langsung ke berbagai bahasa pemrograman dan basis data berorientasi objek.

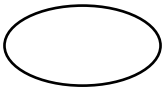
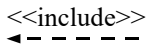
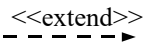
a. *Use Case Diagram*

Use Case Diagram berfungsi untuk menunjukkan bagaimana pengguna atau aktor berinteraksi dengan sistem yang sedang dianalisis. Diagram ini dapat memudahkan dalam mengenali fitur-fitur penting sistem

dan cara pengguna berinteraksi dengan sistem tersebut (Wicaksono, 2023:50). Diagram ini berfokus pada "*use case*" atau kasus penggunaan, yang menggambarkan fungsi-fungsi atau aksi-aksi spesifik yang akan dilakukan oleh aktor dalam sistem (Putra et al., 2024:51). Ada dua jenis diagram *use case*, yaitu *use case primer* dan *use case* narasi. *Use case* primer digunakan untuk menjelaskan secara umum bagaimana sistem bekerja, sehingga proses setiap aktor dan hubungan antar *use case* bisa terlihat dengan jelas. Sementara itu, *use case* narasi dibuat untuk menggambarkan proses yang terjadi dalam sebuah antarmuka pengguna grafis (GUI) (Sumirat et al., 2023:83). Diagram *use case* dekat kaitannya dengan kejadian-kejadian. Kejadian atau skenario tersebut adalah contoh yang menunjukkan bagaimana seseorang berinteraksi dengan sistem (Saputra et al., 2023:145).

Berdasarkan keempat sumber diatas dapat disimpulkan bahwa, diagram *use case* berfungsi untuk menjelaskan cara aktor berinteraksi dengan berbagai fungsi dalam sistem. Diagram ini menunjukkan bagaimana fitur utama dan situasi yang terjadi ketika pengguna menggunakan aplikasi tersebut. Berdasarkan jenisnya, diagram ini dibagi menjadi dua yaitu *use case primer* yang menggambarkan alur sistem secara keseluruhan dan *use case* narasi yang menjelaskan proses pada antarmuka grafis (GUI).

Tabel 2. 2 Simbol *Use Case Diagram*

No	Simbol	Nama	Keterangan
1.		*Actor/Role*	Simbol yang memnunjukkan peran sistem lain, orang, atau alat ketika berhubungan dalam *use case.*
2.		*Use Case*	Simbol yang menunjukkan abstraksi dan cara berinteraksi antara *actor* dengan sistem.
3.		*Generalization*	Simbol yang menggambarkan interaksi aturan-aturan dengan elemen lain.
4.		*Include*	Simbol yang digunakan untuk memasukkan sebuah *use case* ke dalam *use case* lainnya.
5.		*Extend*	Simbol yang digunakan untuk memperluas *use case* agar dapat memasukkan perilaku yang tidak wajib.

Sumber : (Nurelasari, 2021:19)

b. *Activity Diagram*

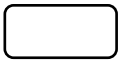
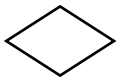
Diagram Aktivitas adalah salah satu jenis diagram perilaku dalam UML yang berfungsi untuk menampilkan bagaimana suatu aktivitas atau kasus penggunaan bekerja atau berlangsung. *Activity Diagram* menunjukkan langkah-langkah yang dijalani oleh suatu objek atau kelas untuk mencapai tujuan tertentu, termasuk pilihan-pilihan dan jalur-jalur percabangan yang mungkin muncul dalam alur proses tersebut (Putra et al., 2024:49). *Activity Diagram* berfokus pada aktivitas-aktivitas yang berlangsung dan berkaitan dalam satu proses yang sama. Dengan kata lain, diagram tersebut menggambarkan saling ketergantungan antar aktivitas yang ada (Saputra et al., 2023:156).

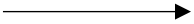
Activity Diagram digunakan untuk menunjukkan aliran proses bisnis serta cara aktivitas satu dengan lainnya saling terkait. Diagram ini

membantu mengidentifikasi tahapan dalam proses bisnis serta bagaimana setiap tahapan tersebut memiliki keterkaitan satu sama lain (Wicaksono, 2023:51). Diagram aktivitas ini dirancang untuk menjelaskan proses sistem yang sedang berjalan, alih-alih berfokus pada tindakan yang dilakukan oleh aktor. Secara umum, penekanan diagram ini lebih tertuju pada alur dan proses aktivitas secara umum dari tingkat yang lebih tinggi (Andriyanto, 2022:21).

Berdasarkan keempat sumber diatas dapat disimpulkan bahwa *activity diagram* merupakan salah satu jenis diagram perilaku dalam UML yang digunakan dalam menampilkan alur proses bisnis dan hubungan antar aktivitas. Fokus utama diagram ini adalah menunjukkan tahapan aktivitas dari sistem secara umum pada tingkat yang lebih tinggi, bukan tindakan yang dilakukan oleh aktor. Dengan bantuan visualisasi, diagram ini mempermudah proses identifikasi langkah-langkah, pilihan, hingga jalur percabangan yang dilalui objek agar mencapai tujuan tertentu.

Tabel 2. 3 Simbol *Activity Diagram*

No	Simbol	Nama	Keterangan
1.		Status Awal / *Initial Node*	Simbol yang menunjukkan awal dari sejumlah tindakan atau kegiatan.
2.		Activity	Simbol yang menggambarkan sekumpulan tindakan (*action*).
3.		*Decision Node*	Simbol yang berfungsi untuk menunjukkan kondisi pengujian agar aliran kontrol atau aliran objek hanya bergerak melalui satu jalur saja. Dilambangkan dengan kriteria keputusan untuk meneruskan ke jalur tertentu.

No	Simbol	Nama	Keterangan
4.		Status Akhir / *Final Node*	Simbol yang berfungsi untuk menghentikan semua aliran kontrol dan aliran objek dalam sebuah aktivitas.
5.		*Control Flow*	Simbol yang berfungsi untuk menunjukkan urutan eksekusi.
6.		*Swimlane*	Simbol yang dipakai untuk membagi diagram aktivitas menjadi baris dan kolom agar kegiatan tertentu dapat diberikan kepada individu atau objek yang bertanggung jawab melaksanakannya.

Sumber : (Nurelasari, 2021:21)

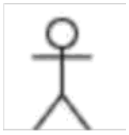
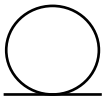
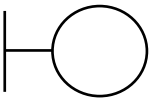
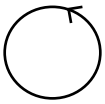
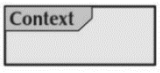
c. *Sequence Diagram*

Sequence diagram merupakan alat bantu visual yang berfungsi untuk memodelkan sistem, yang menunjukkan cara berbagai objek dan sistem berinteraksi dalam skenario tertentu, sehingga memudahkan dalam mengevaluasi bagaimana sistem berjalan (Albaqir & Winarti, 2025:11). *Sequence Diagram* menunjukkan bagaimana objek-objek saling berinteraksi satu sama lain dalam sebuah urutan waktu tertentu. Diagram ini menunjukkan bagaimana objek-objek saling berinteraksi dengan menampilkan pesan-pesan yang dikirimkan antar objek tersebut. Pesan-pesan ini diatur berdasarkan urutan waktu, sehingga dapat melihat bagaimana pesan dikirim dan diterima dari satu objek ke objek lainnya. Diagram urutan sangat berguna untuk menggambarkana jalur eksekusi dari suatu proses atau fungsi dalam sistem (Putra et al., 2024:64).

Diagram sequence digunakan untuk menunjukkan cara objek dalam sistem berinteraksi serta bagaimana pesan dikirim dan diterima seiring berjalannya waktu. Diagram ini memudahkan mengenali cara berkomunikasi dan kolaborasi antar objek dalam sistem (Wicaksono, 2023:52). Tujuan membuat *diagram sequence* adalah agar bisa memahami urutan kejadian-kejadian yang diperlukan untuk menghasilkan keluaran sesuai harapan. Selain itu, tujuan dari *sequence diagram* hampir sama dengan *activity diagram*, yaitu untuk menunjukkan proses kerja dari suatu aktivitas, serta mampu menampilkan aliran data secara lebih terperinci, termasuk data atau tindakan yang diterima atau dikirimkan (Sumirat et al., 2023:87).

Berdasarkan keempat sumber tersebut, dapat disimpulkan bahwa *diagram sequence* adalah alat bantu visual yang digunakan untuk menggambarkan cara objek dalam sistem berinteraksi dan bekerja sama berdasarkan urutan waktu. Diagram ini berfungsi dengan menampilkan pesan yang dikirimkan dan diterima antar objek agar menjelaskan cara suatu fungsi dijalankan secara rinci. Tujuan utamanya adalah memudahkan pemahaman urutan kejadian dan aliran data yang kompleks, sehingga bisa menghasilkan *output* sistem yang diharapkan.

Tabel 2. 4 Simbol *Sequence Diagram*

No	Gambar	Nama	Keterangan
1.		*Actor*	Simbol yang menggambarkan entitas, baik berupa manusia maupun sistem, yang terlibat dalam suatu alur dengan mengirimkan dan/atau menerima pesan.
2.		*Entity Class*	Simbol yang mewakili *Class.*
1.		*Boundary Class*	Simbol yang menggambarkan tampilan dari sistem
2.		*Control Class*	Simbol yang menggambarkan *controller* dari sistem
3.		*Message*	Simbol yang digunakan untuk memberi tahu informasi dari satu benda ke benda yang lainnya.
4.		*Life Line*	Simbol yang menunjukkan adanya kehidupan pada suatu objek dalam urutan tertentu.
5.		*Execution Occurrence*	Simbol yang menggambarkan momen ketika sebuah objek melakukan aktivitas pengiriman atau penerimaan pesan.
6.		*Frame*	Simbol yang menunjukkan konteks *sequence diagram.*

Sumber : (Nurelasari, 2021:30)

d. *Class Diagram*

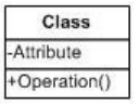
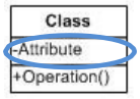
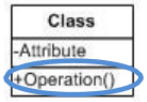
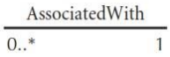
Diagram class merupakan salah satu jenis diagram struktur yang digunakan dalam proses pembuatan model perangkat lunak dan pengembangan sistem. Diagram ini masuk ke dalam kategori diagram yang

umum digunakan dalam UML. Diagram kelas menunjukkan struktur berbagai kelas dalam sistem serta cara kelas-kelas tersebut saling terhubung (Putra et al., 2024:54). Diagram kelas memberikan gambaran umum tentang suatu sistem dengan menampilkan kelas-kelas yang ada dan keterkaitan di antara mereka. Diagram kelas memiliki sifat yang statis, ia menggambarkan bagaimana hubungan yang ada, bukan kondisi yang terjadi saat hubungan tersebut terbentuk (Saputra et al., 2023:146).

Diagram kelas berfungsi untuk menunjukkan struktur sistem yang mencakup kelas, sifat, serta hubungan antara satu kelas dengan kelas lainnya. Diagram ini memudahkan pemahaman tentang komponen-komponen sistem serta cara mereka saling terhubung (Wicaksono, 2023:51). Desain model dalam diagram kelas terdiri dari dua bagian. Bagian pertama yaitu menjelaskan *database*. Bagian kedua merupakan bagian dari modul MVC yang terdiri dari kelas *interface*, kelas *control*, dan kelas *entity* (Sumirat et al., 2023:89).

Berdasarkan keempat sumber diatas dapat disimpulkan bahwa *class diagram* merupakan jenis diagram struktur dalam UML yang digunakan untuk menampilkan struktur kelas, sifat, serta hubungan antar objek dalam suatu sistem. Diagram ini membantu memudahkan pemahaman tentang komponen-komponen dalam sistem dan cara mereka saling terhubung. Dalam penerapannya, desain model diagram ini dibagi menjadi dua bagian utama, yaitu penjelasan tentang basis data dan bagian modul MVC yang terdiri dari antarmuka, kontrol, dan entitas.

Tabel 2. 5 Simbol *Class Diagram*

No	Gambar	Nama	Keterangan
1.		*Generalization*	Simbol yang mewakili jenis hubungan antara beberapa kelas.
2.		*Class*	Simbol ini mewakili entitas seperti individu, lokasi, atau objek tertentu yang datanya perlu direkam dan disimpan oleh sistem.
3		*Attribute*	Simbol yang mewakili properti yang menjelaskan kondisi sebuah objek.
4		*Operation*	Simbol yang menggambarkan tindakan atau fungsi spesifik yang dapat dijalankan oleh suatu kelas.
5		*Association*	Simbol yang mewakili keterkaitan antar kelas maupun hubungan sebuah kelas dengan dirinya sendiri.

Sumber : (Santoso & Migunani, 2021:187)

B. Kajian Empiris

Penelitian yang dilakukan oleh Banurea et al., tahun 2024, berfokus dalam menyelesaikan suatu permasalahan dengan membuat sistem pakar yang akan digunakan oleh asisten dokter dalam proses diagnosis penyakit pankreatitis menggunakan metode *Certainty Factor*. Hasil penelitian ini adalah terciptanya sebuah sistem pakar yang mampu memberikan hasil akhir berupa tingkat kemungkinan seseorang mengalami pankreatitis serta solusi yang dapat dilakukan, sehingga dapat membantu rumah sakit dalam mendiagnosis pankreatitis pada pasien.

Sementara itu, penelitian yang dilakukan oleh Hafiz et al., tahun 2022 berfokus dalam membangun sistem pakar untuk mendiagnosis penyakit ibu hamil menggunakan metode *Certainty Factor*. Tujuan dari penelitian ini adalah untuk mengurangi risiko kematian pada ibu hamil berbasis android. Penelitian ini dimulai dengan pengamatan untuk mengumpulkan data keluhan yang dialami ibu hamil, yang akan menjadi dasar dalam merancang sistem pakar. Selanjutnya dilakukan wawancara dengan dokter atau bidan yang bertugas untuk mengumpulkan informasi yang akan digunakan sebagai data gejala dan penyakit. Sistem diuji menggunakan metode *blackbox testing* dan hasilnya menunjukkan bahwa sistem berjalan dengan baik. Dengan aplikasi ini, ibu hamil nantinya bisa dengan mudah memantau kondisi kesehatannya.

Penelitian yang dilakukan oleh Maulana et al., tahun 2023 bertujuan untuk membuat aplikasi sistem pakar yang bisa digunakan dalam mendeteksi dan mendiagnosis penyakit ginjal. Penelitian ini menggunakan metode *Certainty Factor* untuk menunjukkan persentase tingkat keyakinan dan kemungkinan seseorang terkena penyakit ginjal. Metode yang digunakan untuk mengembangkan sistem pakar adalah metode siklus pengembangan sistem pakar (ESDLC). Aplikasi sistem pakar ini juga bisa membantu pengguna dalam memahami informasi mengenai penyakit ginjal, serta bisa menjadi pilihan alternatif bagi masyarakat untuk mendeteksi gejala-gejala penyakit ginjal secara dini yang dialami oleh pengguna.

Penelitian yang dilakukan oleh Putra et al., pada tahun 2021 bertujuan membuat sistem pakar untuk membantu mendiagnosis mioma uterus, dengan menggunakan metode *Certainty Factor*. Sistem pakar yang menggunakan metode

Certainty Factor diuji pada pasien dengan cara memasukkan gejala-gejala yang dialami oleh pasien tersebut. Penelitian ini menunjukkan bahwa sistem pakar yang menggunakan metode *Certainty Factor* bisa membantu dalam mendeteksi mioma rahim lebih awal.

Penelitian yang dilakukan oleh Ayal & Susilawati tahun 2024 bertujuan membuat sistem pakar untuk mendiagnosis cedera pada bahu, siku, dan pergelangan tangan. Sistem ini menggunakan metode *Certainty Factor* yang dapat membantu pengguna mengenali gejala awal dan mendiagnosis cedera yang mungkin terjadi di area tersebut. Selain itu, sistem ini juga memberikan solusi atau saran penanganan yang tepat. Penelitian ini menyarankan agar sistem pakar diperluas dengan menambahkan fitur-fitur untuk meningkatkan keakuratan dalam proses diagnosis.

Penelitian yang dilakukan oleh Pamungkas et al., pada tahun 2023 bertujuan untuk membuat sistem pakar yang bisa membantu mendiagnosis penyakit pada lambung dengan menggunakan metode *Certainty Factor*. Dalam tahap desain, sistem dirancang menggunakan *flowchart*, *unified modeling language*, serta pembuatan basis data. Sistem ini menggunakan metode *Certainty Factor* untuk menentukan diagnosis. Desain basis data dilakukan dengan menggunakan MySQL, sedangkan pembuatan website dilakukan dengan PHP, CSS, dan JavaScript. Penelitian ini menunjukkan bahwa sistem pakar mampu memberikan diagnosis yang benar berdasarkan aturan yang ditetapkan oleh ahli dan metode yang digunakan. Sistem ini berjalan dengan baik dan bisa memberikan hasil diagnosis yang sesuai dengan gejala-gejala yang dirasakan oleh pengguna yang sedang mengalami sakit lambung.

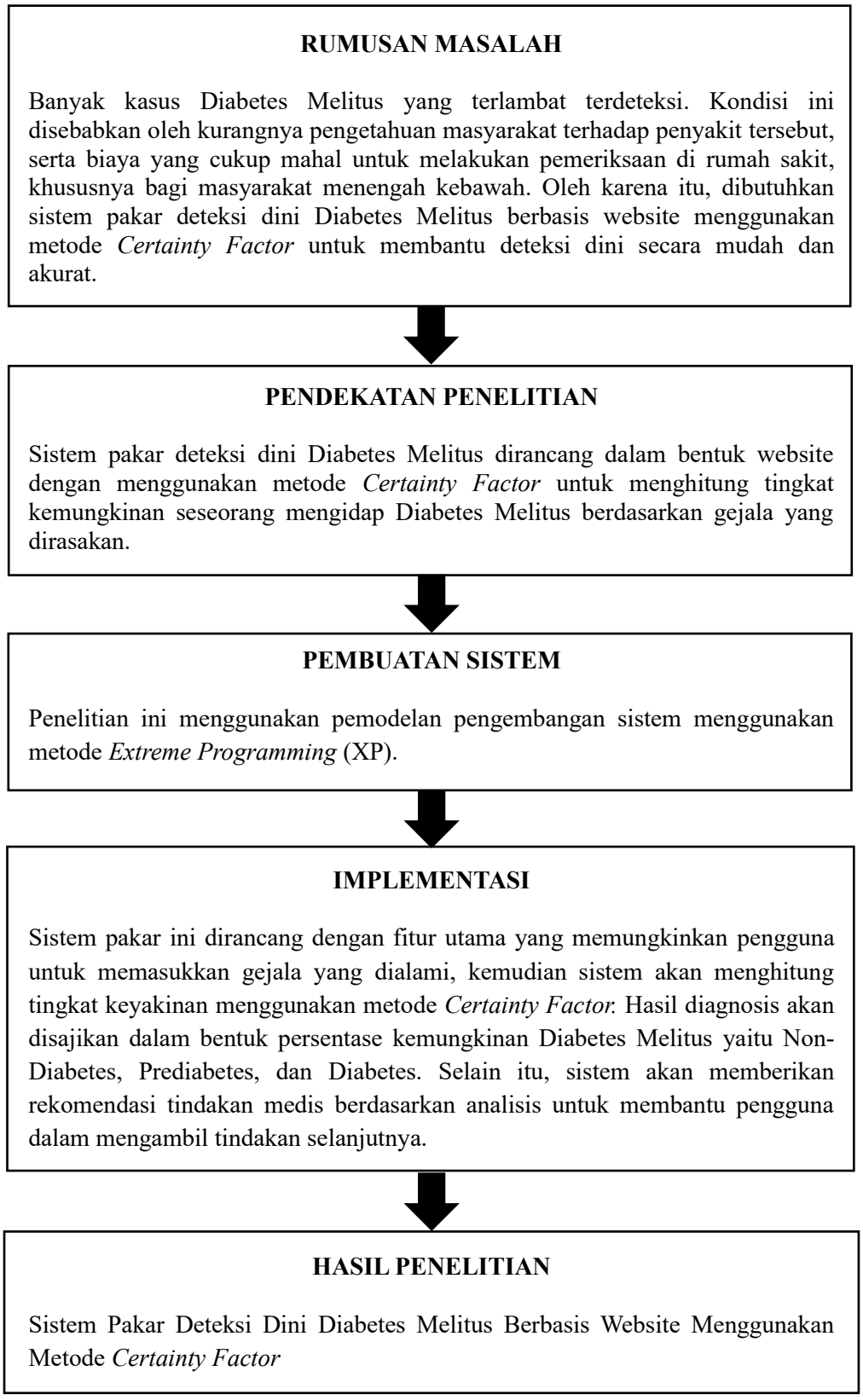
Penelitian yang akan dilakukan bertujuan untuk merancang dan membangun Sistem Pakar Deteksi Dini Diabetes Melitus menggunakan metode *Certainty Factor*. Sistem ini dikembangkan untuk mengatasi keterbatasan pada penelitian sebelumnya, dimana penelitian Banurea et al., (2024) berfokus pada diagnosis pankreatitis tanpa mendeteksi diabetes. Selain itu penelitian Hafiz et al., (2022) yang hanya dapat mendeteksi penyakit diabetes pada ibu hamil. Penelitian ini mengacu pada metode *certainty factor* yang digunakan dalam penelitian Maulana et al., (2023) namun, menggunakan metode pengembangan sistem *Extreme Programming.*

Penelitian ini juga mengacu pada metode penelitian yang digunakan Putra et al., (2021), Ayal & Susilawati (2024) dan Pamungkas et al., (2023) yaitu menggunakan metode *Certainty Factor* yang dapat memberikan hasil diagnosis yang sesuai dengan aturan yang telah ditetapkan pakar dan juga metode yang dipakai. Dengan demikian, penelitian ini diharapkan dapat menjadi solusi sistem pakar yang dapat mendiagnosa Diabetes Melitus dengan efisien dan akurat.

C. Kerangka Berpikir

Kerangka berpikir ini dibuat untuk menunjukkan langkah-langkah dalam merancang dan membangun sistem pakar deteksi dini Diabetes Melitus berbasis website menggunakan metode *Certainty Factor*. Tujuan dari penelitian ini adalah untuk membantu masyarakat mengenali kemungkinan gejala Diabetes Melitus sejak dini secara mandiri. Masalah utama yang diangkat adalah masih banyaknya kasus Diabetes yang terlambat terdeteksi karena kurangnya pengetahuan masyarakat terhadap penyakit Diabetes dan biaya yang cukup mahal khususnya bagi masyarakat menengah kebawah.

Dalam penelitian ini, digunakan pendekatan metode *Extreme Programming* yang dimulai dari perencanaan, perancangan, penulisan kode program, dan pengujian. Sistem akan bekerja dengan meminta pengguna memilih gejala yang dialami, lalu menghitung kemungkinan terkena Diabetes Melitus berdasarkan gejala tersebut menggunakan metode *Certainty Factor*. Hasil dari sistem ini adalah informasi tingkat kemungkinan dalam persen, serta saran dan rekomendasi untuk langkah selanjutnya. Berikut adalah kerangka berpikir pada penelitian ini dapat dilihat pada Gambar 2.1 :



Gambar 2. 1 Kerangka Berpikir