

BAB II

KAJIAN PUSTAKA

A. Kajian Teoritis

1. Rancang Bangun

Dalam membuat sebuah *project* atau produk tentunya harus memiliki gambaran yang jelas mengenai *project* tersebut. Biasanya dibuat sebuah rancangan yang berisi gambaran awal atau *draft* dari suatu konsep desain sebelum dilakukan pengembangan lebih lanjut. Menurut Rauf & Prastowo, (2021) rancang bangun merupakan pembuatan sketsa dengan mengubah hasil analisis perangkat lunak dan memperbaiki sistem yang sudah ada menjadi sistem baru. Perangkat lunak awalnya terdiri dari sejumlah elemen-elemen yang terpisah kemudian dibentuk menjadi satu unit yang utuh serta memiliki fungsi yang dapat membantu aktivitas manusia.

Menurut Gusti Putri & Setiawan, (2020) mengemukakan, rancang bangun adalah proses untuk menerjemahkan hasil analisis sistem ke dalam bahasa pemrograman yang bertujuan untuk menggambarkan secara detail bagaimana komponen yang ada akan direalisasikan. Rancangan sistem mencakup semua fitur utama yang akan diimplementasikan oleh pembuatnya. Dengan adanya rancangan ini seseorang akan memiliki gambaran atau *visualisasi* awal dari ide dan konsep yang dimiliki. Rancang bangun sendiri juga dapat diartikan sebagai sebuah proses pengembangan sistem untuk membuat sistem yang baru atau

melakukan perbaikan pada sistem yang sudah ada baik secara keseluruhan ataupun sebagian (Ziliwu et al., 2021).

Berdasarkan penjelasan diatas, disimpulkan bahwa rancang bangun adalah proses pengembangan sistem yang melibatkan pembuatan sketsa dan penerjemahan hasil analisis perangkat lunak ke dalam bahasa pemrograman. Proses ini digunakan untuk menciptakan, mengganti, atau memperbaiki sistem yang sudah ada secara keseluruhan atau sebagian.

2. Sistem Pakar

Kecerdasan yang sudah diterapkan dalam dunia bisnis berupa sistem pakar, jaringan syaraf tiruan, algoritma genetika, dan agen cerdas. Komputer tidak dapat menggantikan peran pakar, melainkan berfungsi sebagai asistensi manusia dalam mencari solusi untuk masalah yang sedang dihadapi. Kumarahadi et al., (2020) menjelaskan bahwa sistem pakar merupakan sistem yang berusaha menerapkan pengetahuan manusia pada komputer sehingga komputer dapat memecahkan masalah yang biasa diselesaikan oleh seorang pakar.

Pada dasarnya, sistem pakar adalah sebuah sistem yang bertindak sebagai pakar serta menggunakan pengetahuan seorang pakar (Pebrianto et al., 2020). Sistem pakar dapat mensimulasikan proses berpikir dan keahlian seorang pakar untuk memecahkan suatu masalah tertentu (Syahwal Alam & Wantoro, 2022). Komponen yang diperlukan agar terbentuknya sistem pakar meliputi memori kerja, antarmuka pengguna, mesin inferensi serta basis pengetahuan (Z. E. Fitri et al., 2022).

Maka dapat disimpulkan bahwa sistem pakar merupakan program komputer yang dibuat untuk menduplikat pengetahuan dan kemampuan pemecahan masalah pakar. Sistem ini berfungsi mengelola konsultasi dengan mensimulasikan proses berpikir dan keahlian pakar dalam memecahkan masalah tertentu.

3. Demam Berdarah Dengue (DBD)

Demam berdarah dengue disebabkan oleh virus *dengue* dan ditularkan melalui gigitan nyamuk *Aedes Albopictus* atau *Aedes Aegypti*. Selama musim hujan, penyakit ini sering muncul terutama di genangan air yang lembab. Perilaku hidup masyarakat dan kondisi lingkungan juga menjadi faktor adanya sarang nyamuk ini. Penerapan hidup bersih dan sehat dapat mencegah munculnya nyamuk demam berdarah *dengue* sehingga perkembangannya dapat diminimalisasikan. Menurut Anggraini et al., (2021), demam berdarah *dengue* merupakan kompilasi dari demam *dengue* (*Dengue Fever*) yang semakin parah. Penyakit ini dapat menular dan ditandai dengan demam disertai pendarahan. Demam mendadak yang berlangsung selama 2 sampai 7 hari tanpa sebab yang jelas, mudah lelah, dan rasa mulas disertai pendarahan diarea kulit kulit berupa bintik-bintik, memar, atau ruam, mimisan dan darah merupakan beberapa gejala terjadinya demam berdarah *dengue* (Mubarak et al., 2022). Penyakit demam berdarah *dengue* yaitu penyakit yang menyebabkan kerusakan kaplier dan sistem pembekuan darah sehingga mengakibatkan pendarahan bahkan kematian (Misnadiarly, 2009).

Berdasarkan penjelasan diatas, dapat disimpulkan bahwa demam berdarah *dengue* adalah penyakit serius yang disebabkan gigitan nyamuk *Aedes Albopictus* atau *Aedes Aegypti* dan mengakibatkan kerusakan pada kaplier serta pembekuan darah bahkan dapat menyebabkan kematian.

Pada buku karya Frida (2019) mengemukakan bahwa *World Health Organization* (WHO) menyebutkan beberapa gejala klinis penyakit demam berdarah *dengue*, antara lain demam tinggi dengan mendadak dan terus-menerus selama 2-7 hari, manifestasi pendarahan, pembesaran hati, *trombositopeni* (kekurangan trombosit dalam darah, 1.000.000/ul atau kurang), *hemokonsentrasi* (pembesaran plasma).

4. Forward Chaining

Menurut Yansyah & Sumijan, (2021) menjelaskan bahwa *forward chaining* merupakan teknik penelusuran pengetahuan dimulai dengan suatu situasi atau fakta kemudian menarik kesimpulan berdasarkan fakta tersebut. *Forward Chaining* juga dapat digambarkan sebagai teknik penalaran dimulai dengan serangkaian fakta yang diketahui untuk menemukan solusi yang sedang dicari. Metode ini mengeksplorasi solusi berdasarkan permasalahan yang ada, atau dengan mempertimbangkan fakta dan memberikan kesimpulan berdasarkan fakta yang ada (Viviliani & Tanone, 2019:3).

Menurut Wadi (2018:1), *forward chaining* adalah suatu teknik penalaran pada mesin inferensi, dimulai dengan data dan alasan yang mengarah pada sebuah kesimpulan. Logikanya, metode ini juga disebut sebagai iterasi logika modus ponens. Mesin inferensi dengan metode *forward chaining* akan

mencari kaidah sampai ditemukan satu *antecedent* atau *klausa if* yang bernilai benar. Mesin inferensi dapat menarik garis besar dari data setelah ditemukannya aturan. Jika klausa premis sesuai keadaan dan nilainya benar, maka proses akan membuat suatu kesimpulan (Bahroni et al., 2022).

Berdasarkan uraian diatas, dapat disimpulkan bahwa metode *forward chaining* adalah teknik penelusuran dan penalaran dalam sistem pakar dan mesin inferensi yang dimulai dengan serangkaian fakta atau data yang sudah diketahui.

5. Website

Menurut Elgamar (2020), *website* adalah media yang terdiri dari beberapa halaman yang terhubung, serta berisi informasi seperti teks, gambar, video, suara, animasi atau gabungan dari semua. *Website* sendiri juga dapat diartikan sebagai kumpulan elemen halaman *web* berupa teks dengan format HTML (*Hypertext Markup Language*) (Widia & Asriningtias, 2021).

Berdasarkan sudut pandang yang telah dijelaskan, media digital yang berisi beberapa halaman saling terhubung dan memberikan informasi dengan berbagai format serta dapat diakses melalui koneksi internet disebut *website*.

6. Flowchart




Suatu algoritma dapat digambarkan dengan *flowchart*, terutama yang disusun berbentuk gambar sehingga lebih mudah dipahami orang lain. *Flowchart* adalah teknik yang menggambarkan urutan logik proses pemecahan masalah (Hanief & Jepriana, 2020).

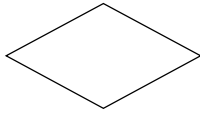









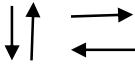
Menurut Huda et al (2021), *flowchart* merupakan diagram dengan simbol-simbol tertentu yang memberikan gambaran mendalam langkah proses secara rinci dan bagaimana satu proses berhubungan dengan proses lain dalam sebuah program. Diagram ini berisi langkah-langkah pembuatan sebuah *website* atau aplikasi yang dihubungkan dengan sebuah garis atau arah panah. *Flowchart* dapat diartikan sebagai *chart* yang menunjukkan urutan prosedur sistem terutama untuk membantu komunikasi dan dokumentasi program agar mudah dipahami oleh programmer (Kusrini, 2007).

Berdasarkan penjelasan diatas, *flowchart* adalah teknik visual yang efektif untuk menggambarkan urutan logika dalam proses pemecahan masalah yang menggunakan simbol tertentu untuk menunjukkan hubungan rinci antara satu proses dengan proses lainnya dalam sebuah sistem.

Pada buku karya Hanief & Jepriana, (2020) terdapat simbol untuk menggambarkan algoritma dalam bentuk diagram alir beserta dengan fungsinya. Simbol tersebut meliputi:

Tabel 2. 1 Simbol-Simbol *Flowchart*

Simbol	Nama	Fungsi
	<i>Terminal</i>	Permulaan atau akhir dari tahapan yang disajikan flowchart
	<i>Data</i>	Proses masukan input atau output data
	<i>Process</i>	Suatu tindakan atau aksi yang dilakukan

	<i>Decision</i>	Pengujian terhadap kondisi tertentu yang menghasilkan dua jawaban ya/tidak
	<i>Connector</i>	Suatu hubungan dari suatu proses ke proses yang lain pada halaman yang sama
	<i>Offline Connector/Off-page Connector</i>	Hubungan suatu proses ke proses yang lain pada halaman berbeda
	<i>Preparation/Predefined Process</i>	Penyediaan tempat penyimpanan suatu pengolahan untuk memberi nilai awal
	<i>Punched Card</i>	Menyatakan suatu masukan atau keluaran dari kartu
	<i>Punch Tape</i>	Masukan dan keluaran punched card digunakan untuk satu alat
	<i>Document</i>	Mencetak data yang dapat dibaca oleh orang lain berupa keluaran dalam bentuk dokumen
	<i>Manual Input</i>	Proses yang dilakukan secara manual oleh manusia
	<i>Subroutine</i>	Proses yang telah didefinisikan sebelumnya
	<i>Stored Data</i>	Bentuk tempat penyimpanan data
	<i>Flow</i>	Jalannya arus suatu proses

Sumber : (Hanief & Jepriana, 2020:9)

7. Database

Menurut Setyawati et al., (2020), database adalah tabel data berisi informasi yang saling terhubung serta terdiri dari satu atau lebih tabel. Database dapat diartikan sebagai sistem yang memungkinkan mudah mengatur, menyimpan dan mengambil data (Ramadhan & Mukhaiyar, 2020). Pengertian database menurut Sidharta & Wibowo (2020), adalah media penyimpanan data yang dapat memudahkan pencarian, penyimpanan, dan modifikasi data dalam jumlah besar. Dengan database, penyimpanan data skala besar dapat dipermudah melalui fitur sistem manajemen database untuk meminimalisir duplikasi data. Dengan demikian, database dapat didefinisikan sebagai sebuah sistem terdiri dari satu atau lebih tabel dan berisi informasi yang saling berhubungan.

8. PHP

PHP adalah singkatan dari (*Hypertext Perprocessor*) berarti bahasa pemrograman berbasis *web* yang mampu menangani dan mengolah data dengan berbagai cara (Fadila et al., 2021). PHP adalah *server-side embedded script language*, yang berarti bahwa setiap sintaks dieksekusi sepenuhnya oleh *server* tetapi bisa dimasukkan pada halaman HTML. Menurut Murni et al.,(2023), PHP merupakan bahasa *script* untuk membuat aplikasi *web* dan menghubungkannya ke *server*. PHP berjalan di server dan bekerja dengan HTML, serta dapat membuat berbagai halaman web seperti *Dynamic Server Pages* (ASP), dan *Java Server Pages* (JSP).

Berdasarkan sudut pandang yang telah dijelaskan, disimpulkan bahwa PHP adalah bahasa pemrograman berbasis *web* fleksibel dalam menangani dan mengolah data. Selain itu, PHP dapat dipakai untuk membuat aplikasi *web* dan menghubungkannya ke *server*.

9. MySQL

Menurut Noviana (2022), *MySQL* adalah *query* basis data tertentu dimana *sub* bahasa bisa membuat dan mengubah data pada basis data. *MySQL* bertujuan untuk melakukan berbagai tugas seperti memperbarui database yang mengacu pada konsep *Relational Database Management System* (RDBMS). Selain itu, *MySQL* merupakan mesin *server* database yang mendukung bahasa database *SQL* sebagai bahasa manajemen data interaktif (Sitanggung Rianto et al., 2022). Pengertian lain menurut Fitri (2020), *MySQL* berarti DBMS *open source* dengan dua bentuk lisensi yaitu *Free Software* dan *Shareware*. *MySQL* merupakan *server* gratis dengan lisensi GNU yang digunakan untuk keperluan pribadi atau komersial tanpa harus membayar untuk *server* database yang sudah ada.

Berdasarkan pemahaman diatas, dapat disimpulkan bahwa *MySQL* adalah *Database Management System* (DBMS) *open-source* yang mendukung bahasa *SQL* untuk manajemen data interaktif dan berfungsi untuk memudahkan pembuatan dan manipulasi data dalam basis data.

10. XAMPP

Menurut Haqi (2019), *Xampp* adalah program bebas terdiri dari berbagai program, dan mendukung banyak sistem operasi. *Tool* ini sangat populer untuk membantu pengembangan perangkat lunak sederhana hingga tahap professional dan sering digunakan baik oleh programmer pemula maupun berpengalaman. Selain itu, *Xampp* merupakan alat yang menawarkan paket perangkat lunak dalam satu paket, yang memungkinkan instalasi instan untuk *Apache*, PHP, *MySQL* serta mempermudah proses instalasi dari ketiga produk tersebut (Sitanggang Rianto et al., 2022). Untuk mengembangkan perangkat lunak atau menampilkan *website* dengan cepat, mudah, dan terorganisir, *xampp* digunakan sebagai server mandiri atau dikenal dengan *localhost* (Noviana, 2022).

Berdasarkan penjelasan diatas, *Xampp* adalah *software* bebas yang mendukung banyak sistem operasi dan berfungsi sebagai *server* yang berdiri sendiri (*localhost*) serta memfasilitasi proses pengeditan, desain, dan pengembangan aplikasi.

11. HTML

Pengguna menggunakan *Hypertext Markup Language* (HTML) untuk membuat dan mengatur paragraf, heading, tautan, dan kutipan situs web dan aplikasi (Anamisa & Mufarroha, 2020). HTML sendiri bukanlah bahasa pemrograman, melainkan bahasa *markup* yang berisi perintah dalam format tertentu untuk menunjukkan tampilan khusus. Dalam dunia pemrograman *website*, HTML berperan sebagai pondasi dasar pada halaman *website* terdiri

dari *head*, *body*, *tag*, serta *attribute* yang memiliki fungsi masing-masing. Banyak *script* HTML disimpan dalam format file *.html*. Untuk mengimport *script* HTML dapat menggunakan *text editor* khusus seperti *Notepad++*, *Sublime text*, *Visual Studio Code*, dan aplikasi lainnya. HTML memiliki beberapa fungsi, menurut (Mariko, 2019) fungsi dari HTML meliputi mengatur dan mendesain konten *website*, membuat tabel pada halaman *web*, mempublikasikan halaman dalam baris-baris *website*, membuat formulir entri data, mengelola registrasi dan transaksi melalui *website*, serta menampilkan kotak gambar di *browser*.

Berdasarkan penjelasan diatas, dapat disimpulkan bahwa HTML adalah bahasa *Markup* yang digunakan pengguna untuk membuat dan mengatur paragraf, judul, tautan, kutipan untuk situs *web* dan aplikasi dengan menggunakan ekstensi *.html*.

12. CSS

Menurut Marlina et al (2021), CSS atau *Cascading Style Sheet* merupakan ketentuan yang mengatur tampilan dan nuansa suatu *website* agar *website* lebih terstruktur. CSS sendiri bukanlah sebuah bahasa pemrograman, melainkan konfigurasi untuk menampilkan tag pada suatu halaman *web*. CSS dapat mengubah teks, warna, latarbelakang, dan posisi sebuah tag. Saat mendesign halaman *website*, CSS menggunakan penanda berupa *id* dan *class*. CSS dapat mengatur ukuran gambar, warna konten teks, warna tabel, ukuran batas, warna batas, warna hyperlink, warna gerakan *mouse*, spasi paragraf,

spasi teks, margin kiri, kanan, atas, bawah dan pengaturan lainnya (Hilabi, 2017).

Berdasarkan uraian diatas, maka CSS (*Cascading Style Sheet*) yaitu sebuah alat pengembangan *web* yang digunakan untuk mengatur tampilan dan nuansa suatu halaman *website*. Meskipun bukan bahasa pemrograman, CSS berfungsi sebagai konfigurasi untuk menampilkan elemen HTML secara terstruktur dan estetis.

13. Bootstrap

Bootstrap adalah *framework front-end* yang efisien dan mendukung tampilan *mobile device* sehingga membuat pengembangan situs *web* menjadi lebih cepat dan sederhana (Suprayogi & Rahmanesa, 2019). *Bootsrtap* merupakan kerangka kerja dalam pembuatan desain *web responsive*. Fitur ini dapat diaktifkan atau dinonaktifkan sesuai dengan keinginan, jadi sebuah *website* dapat diatur untuk diperlihatkan sesuai dengan ukuran layar. Menurut Mandasari (2020), *bootstrap* merupakan kerangka CSS yang populer serta didukung oleh banyak artikel, tutorial, serta *plugin*, ekstensi pihak ketiga, pembuat tema, dan masih banyak lagi. Hal ini menjadi alasan mengapa banyak orang menggunakan *bootstrap*. *Bootsrap* mendukung banyak *platform* tampilan dan *browser web* yang berbeda dibandingkan dengan *framework* CSS *responsive* lainnya, sehingga *bootstrap* banyak digunakan oleh desainer dan pengembang *web*.

Dengan mempertimbangkan penjelasan diatas, disimpulkan bahwa, *bootstrap* adalah *framework front-end* yang sangat efisien dan populer serta

mendukung tampilan *responsive* untuk berbagai perangkat. Selain itu, *bootstrap* juga memudahkan dan mempercepat pengembangan situs *web*.

14. Unified Modelling Language (UML)

Menurut Abdillah (2021), UML adalah bahasa pemodelan perangkat lunak standar yang mendukung penulisan rencana perangkat lunak. Arsitek perangkat lunak dapat membuat diagram UML untuk membantu pemrogram atau pengembang membuat perangkat lunak, sama seperti arsitek membuat dokumen skematik yang digunakan perusahaan konstruksi membangun bangunan. Diagram UML juga dapat digunakan untuk memvisualisasikan, menentukan, membangun, dan mendokumentasikan bagian-bagian tertentu dari sistem yang terdapat dalam perangkat lunak.

Menurut Fu'adi & Prianggono (2022), UML merupakan bahasa pemodelan grafis untuk mendeskripsikan, membangun dan mendokumentasikan artefak sistem perangkat lunak yang digunakan sebagai standar pembuatan model sistem, mencakup hal-hal konseptual. UML dapat digunakan sebagai alat bantu untuk mentransfer pengetahuan mengenai sistem yang akan disampaikan dari satu pengembang ke pengembang lainnya.

Berdasarkan informasi diatas, disimpulkan bahwa UML berarti bahasa pemodelan umum yang digunakan dalam pengembangan perangkat lunak untuk menentukan, memvisualisasi, membuat, serta, mendokumentasikan komponen sistem perangkat lunak.


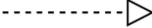

Untuk membuat konteks sistem agar mudah dipahami dan dicerna oleh programmer atau orang awam yang membacanya, UML harus

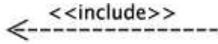
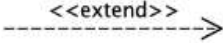





menjelaskan alur atau konteks diagram. Diagram UML terdiri beberapa pemodelan, diantaranya

a. *Use Case Diagram*

Use case diagram adalah model untuk menunjukkan perilaku sistem yang akan dibuat dengan menggambarkan interaksi antara user dan bagaimana peran user tersebut dalam menggunakan sebuah sistem. Selain itu diagram *use case* digunakan untuk mengetahui fungsi-fungsi apa saja yang ada pada sistem dan siapa saja yang berhak menggunakan sistem tersebut. Tujuan *use case* yaitu untuk memetakan kebutuhan sistem, mewakili interaksi pengguna dengan sistem, serta mengetahui atau mengeksplorasi kebutuhan diluar sistem (Habibi & Aprilian, 2019). Pada buku karya (Sa'ad, 2020:47) tapat simbol-simbol dari *use case diagram*, antara lain:

Tabel 2. 2 *Use Case Diagram*

Simbol	Nama	Fungsi
	<i>Actor</i>	Spesifikasi serangkaian peran pengguna saat berinteraksi dengan <i>use case</i>
	<i>Dependency</i>	Hubungan elemen <i>dependen</i> dalam elemen <i>non independent</i> dipengaruhi oleh perubahan pada elemen <i>independent</i>
	<i>Generalization</i>	Hubungan objek anak (<i>descendent</i>) berbagi perilaku dan struktur objek induk (<i>ancestor</i>)



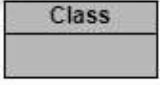
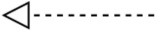
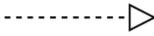
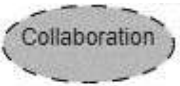

	<i>Include</i>	<i>Use case</i> sumber bersifat <i>eksplisit</i>
	<i>Extend</i>	<i>Use case</i> target harus memperluas perilaku dari <i>use case</i> sumber pada waktu tertentu
	<i>Association</i>	Menghubungkan objek satu dengan objek lainnya
	<i>System</i>	Menentukan paket yang menampilkan sistem secara terbatas
	<i>Use case</i>	Urutan tindakan sistem untuk mendapatkan hasil terukur bagi agen
	<i>Collaboration</i>	Aturan interaksi dan elemen lainnya bekerja sama memberikan perilaku yang lebih besar dari jumlah bagian-bagiannya
	<i>Note</i>	Elemen fisik yang ada ketika aplikasi dijalankan dan mewakili sumber daya komputasi

Sumber : (Sa'ad, 2020:47)

b. *Class Diagram*

Class diagram adalah diagram yang menunjukkan hubungan antar *class* dalam sistem yang sedang dibuat dan menjelaskan bagaimana mereka bekerja sama. Diagram ini bersifat statis dan menunjukkan himpunan kelas, antarmuka, kolaborasi serta relasi yang bisa ditemukan dalam pemodelan sistem berorientasi objek (Muslihudin & Oktafianto, 2016). Pada buku karya (Sa'ad, 2020), terdapat simbol-simbol *class diagram*:

Tabel 2. 3 *Class Diagram*

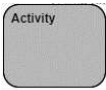



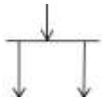
Simbol	Nama	Fungsi
	<i>Generalization</i>	Hubungan objek anak (<i>descendent</i>) berbagi perilaku dan struktur objek induk (<i>ancestor</i>)
	<i>Nary Association</i>	Upaya untuk menghindari pengikatan lebih dari 2 objek
	<i>Class</i>	Kumpulan objek yang memiliki atribut dan operasi yang sama
	<i>Realization</i>	Operasi sebenarnya yang dilakukan oleh sebuah objek
	<i>Dependency</i>	Hubungan perubahan yang terjadi pada elemen <i>independent</i> mempengaruhi elemen <i>dependen</i> yang terkandung dalam elemen <i>no-independent</i>
	<i>Collaboration</i>	Aturan interaksi dan elemen lainnya bekerja sama untuk memberikan perilaku yang lebih besar dari jumlah bagian-bagiannya
	<i>Association</i>	Apa yang menghubungkan objek satu dengan objek lainnya

Sumber : (Sa'ad, 2020:50)

c. *Activity Diagram*

Activity diagram digunakan untuk mendokumentasikan alur kerja dalam suatu sistem, dari tampilan tingkat bisnis (*business level*) hingga ke tingkat operasional (*operational level*) (Sari & Utami, 2021). Diagram ini menunjukkan aliran dari satu aktivitas ke aktivitas lainnya dalam sistem. *Activity diagram* menangani tampilan dinamis sistem dan penting untuk memodelkan fungsionalitas sistem dan menekan aliran kendali antar objek (I. Sari, 2021). Pada buku karya (Sa'ad, 2020), terdapat beberapa simbol dari *class diagram*, diantaranya:

Tabel 2. 4 *Activity Diagram*






Simbol	Nama	Fungsi
	<i>Activity</i>	Menunjukkan kelas antarmuka berinteraksi satu sama lain
	<i>Action</i>	Keadaan sistem yang mencerminkan eksekusi sesuatu
	<i>Initial Node</i>	Objek dibuat dibentuk dan diawali
	<i>Activity Final Node</i>	Objek dibuat dan dihilangkan
	<i>Fork Node</i>	Sebuah aliran di tahapan tertentu berubah menjadi beberapa aliran

Sumber : (Sa'ad, 2020:51)

d. *Sequence Diagram*

Sequence diagram mencatat interaksi antar *class*. Diagram ini menunjukkan beberapa objek dan pesan yang ditempatkan diantara objek di dalam *use case*. *Sequence diagram* adalah jenis diagram dalam UML yang dapat menjelaskan urutan temporal dari pemrosesan sistem (Abdillah, 2021). Pada buku karya (Sa'ad, 2020), terdapat simbol dari *sequence diagram* yang meliputi:

Tabel 2. 5 Sequence Diagram

Simbol	Nama	Fungsi
	<i>Object</i>	Mewakili objek, pesan, atau contoh dari sebuah <i>class</i> dengan nama objek dimulai dari titik koma
	<i>Actor</i>	Spesifikasi peran yang diambil pengguna saat berinteraksi dengan <i>use case</i>
	<i>Lifeline</i>	Kehidupan suatu objek atau menyatakan keberadaan sebuah objek. <i>Lifeline</i> biasanya berupa garis putus <i>vertical</i> yang ditarik dari sebuah objek
	<i>Activation</i>	Objek aktif dan berinteraksi pesan, digambarkan segi empat yang disimpan di <i>lifeline</i> yang menggambarkan suatu objek yang akan melakukan aksi
	<i>Message</i>	Menyatakan komunikasi antar objek

Sumber : (Sa'ad, 2020:53)

15. *Blackbox Testing*

Blackbox testing digunakan untuk menjaga mutu perangkat lunak dalam tahap pengembangannya guna memastikan sebuah sistem tidak terjadi eror atau *bug* (Rahadi & Vikasari, 2020). Pengujian ini dapat menemukan kesalahan fungsional seperti struktur data, akses basis data, antarmuka, serta inisialisasi dan terminasi. Menurut Shadiq et al., (2021), *blackbox testing* adalah metode yang digunakan untuk memastikan bahwa perangkat lunak memenuhi spesifikasi fungsional dan kebutuhan suatu Perusahaan tanpa harus memahani detail internal atau kode program.

Menurut Praniffa et al., (2023), *Blackbox Testing* sangat berguna untuk memastikan perangkat lunak sesuai dengan kebutuhan dan spesifikasi yang telah ditetapkan, meskipun penguji tidak memiliki pengetahuan mendalam tentang struktur internal perangkat lunak. Di sisi lain, keterbatasan pengetahuan penguji dapat menjadi hambatan dalam mendeteksi semua kemungkinan kesalahan atau *bug* yang ada dalam perangkat lunak.

Berdasarkan penjelasan diatas, dapat disimpulkan bahwa *blackbox testing* adalah metode untuk memastikan perangkat lunak memenuhi spesifikasi fungsional dan kebutuhan suatu perusahaan. Metode ini dapat menemukan berbagai kesalahan fungsional tanpa memerlukan pengetahuan tentang struktur internal perangkat lunak. Namun, keterbatasan ini juga menjadi kelemahan, karena penguji mungkin tidak mampu mendeteksi semua kesalahan atau *bug* yang ada.

16. *System Usability Scale (SUS)*

Menurut Fatmawati (2021), *System Usability Scale (SUS)* adalah salah satu metode dalam melakukan analisa atau pengujian *usability* dengan melibatkan pengguna akhir atau *end user* dalam proses pengerjaannya. Metode SUS ini terdiri dari 10 pertanyaan dalam kuisisioner yang digunakan untuk mengevaluasi sebuah produk dan berisi pernyataan bernilai positif dan negative (Wardani et al., 2023). SUS dinilai efektif dan sederhana untuk mengukur kebergunaan suatu sistem dari perspektif pengguna. Dengan melibatkan pengguna yang telah berinteraksi dengan sistem, SUS memberikan wawasan yang berharga tentang pengalaman pengguna dan membantu dalam mengidentifikasi area yang memerlukan perbaikan (Defriani et al., 2021).

Berdasarkan uraian diatas, maka *System Usability Scale (SUS)* adalah metode untuk mengukur kebergunaan sistem dari prespektif pengguna menggunakan kuesioner guna membantu mengidentifikasi area yang memerlukan perbaikan sehingga dapat meningkatkan kualitas dan kebergunaan produk secara keseluruhan.

B. Kajian Empiris

Penelitian yang membahas tentang sistem pakar diagnosis penyakit demam berdarah *dengue* berbasis *website* telah diterapkan dalam berbagai bahan referensi. Pengembangan ini memerlukan penelitian yang tepat dari berbagai sumber penelitian yang relevan.

Penelitian yang dilakukan oleh Fuad et al., (2023), menunjukkan bahwa masyarakat masih mengalami kesulitan dalam membedakan penyakit demam berdarah *dengue* dengan demam biasa karena adanya kemiripan gejala antara kedua penyakit tersebut sehingga menyebabkan keterlambatan dalam penanganannya, melalui metode *certainty factor* membantu proses diagnosis, karena dapat menunjukkan ketidakpastian dari tanggapan seorang pakar tentang tata cara menyelesaikan masalah.

Penelitian Puja (2023), menjelaskan bahwa sulitnya orang awam untuk mengetahui penyakit demam berdarah *dengue* dengan tifus menimbulkan kebingungan dan kesalahan diagnosis sehingga menyebabkan kesalahan diagnosis serta penyalahgunaan obat, dengan metode *forward chaining* dapat mengumpulkan informasi tentang suatu tujuan dan menggabungkan aturan untuk mencapai kesimpulan atau tujuan.

Berdasarkan penelitian Shofia et al., (2017), menjelaskan penyakit demam berdarah *dengue*, malaria dan tifoid memiliki kemiripan pada gejalanya, sehingga banyak tenaga medis dan dokter internship seringkali melakukan kesalahan dalam mendiagnosis, metode *certainty factor* dan *K-Nearest Neighbor* digunakan karena metode ini dapat mengatasi permasalahan ketidakpastian dalam pengambilan keputusan serta klasifikasi terhadap objek yang memiliki jarak terdekat dengan objek tersebut.

Menurut penelitian Yulisman & Monalisa (2019), minimnya pengetahuan pasien terhadap gejala penyakit demam berdarah *dengue* mengakibatkan kesalahan dan keterlambatan dalam penanganannya, penggunaan metode *konvensional*

membutuhkan waktu yang lama sehingga banyak pasien mengantri untuk berkonsultasi atau berobat kepada dokter penggunaan metode *forward chaining* menjadi solusi dalam membantu proses diagnosis ini, karena bersifat runtut maju berdasarkan premis-premis dalam aturan dan fakta yang diberikan oleh *user*.

Penelitian Pribadi et al., (2018), mengemukakan bahwa ketidaktahuan masyarakat mengenai demam berdarah dan keterbatasan dalam pengobatannya seringkali menyebabkan terlambatnya diagnosis sehingga memerlukan tindakan dini untuk mencegah penularan demam berdarah *dengue*, melalui metode *ID3* membantu permasalahan diagnosis karena menyertakan algoritma pembelajaran pohon keputusan untuk melakukan pencarian menyeluruh atas kemungkinan pohon keputusan, serta penggunaan analisis data *mining classifier* memungkinkan meningkatnya tingkat akurasi diagnosis.

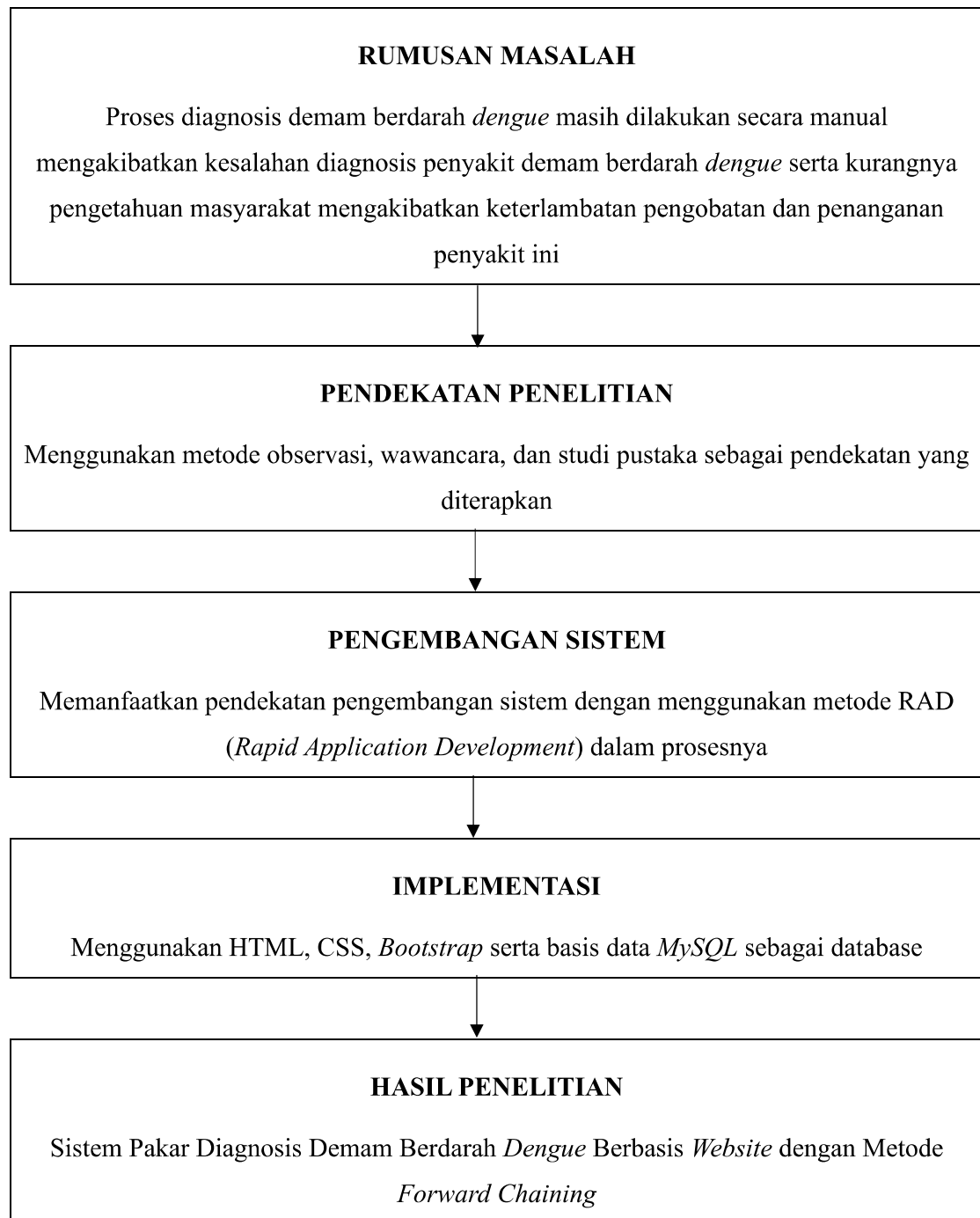
Penelitian sebelumnya terkait sistem pakar diagnosis penyakit demam berdarah *dengue*, berdasarkan kelima referensi dari penelitian terdahulu terdapat kesamaan fungsi utilitasnya, bahwa masyarakat masih mengalami kesulitan dalam membedakan gejala penyakit demam berdarah *dengue* dengan demam biasa atau penyakit lainnya. Perbedaannya terletak pada metode dalam mendiagnosis gejala demam berdarah *dengue*. Penelitian sebelumnya menggunakan metode seperti, *certainty factor*, *forward chaining*, *K-Nearest Neighbor*, serta *ID3* yang membantu proses diagnosis dan meningkatkan tingkat akurasi diagnosis. Namun, pada penelitian ini, peneliti menggunakan metode *forward chaining* karena pendekatan sistem dimulai dengan mengevaluasi informasi yang ada secara bertahap berdasarkan aturan yang telah ditetapkan untuk mendapatkan diagnosis yang tepat.

C. Kerangka Berpikir

Permasalahan pada Praktik dokter umum dr. Saiful saat ini proses diagnosis penyakit demam berdarah *dengue* masih dilakukan secara manual. Oleh sebab itu, dikhawatirkan adanya kesalahan diagnosis penyakit demam berdarah *dengue* dan membeludaknya antrian konsultasi, kurangnya pengetahuan masyarakat pada gejala demam berdarah *dengue*, juga menjadi alasan dalam pembuatan sistem ini.

Dalam penelitian ini, metode yang digunakan dalam pengembangan sistem adalah metode RAD (*Rapid Application Development*). Selain itu dilakukan proses pembuatan desain rancangan kasar untuk mengetahui bagaimana gambaran dari sistem yang akan dibuat. *Website* ini dirancang dengan menggunakan bahasa pemrograman PHP, HTML, CSS, *Bootstrap*, serta menggunakan basis data *MySQL*.

Hasil penelitian berupa sistem pakar diagnosis penyakit demam berdarah *dengue* berbasis *website* yang digunakan untuk membantu dokter dalam proses diagnosis penyakit. Selain itu, sistem ini juga dapat memberi pengetahuan kepada masyarakat mengenai gejala demam berdarah *dengue* sehingga mendapatkan penanganan yang tepat. Berdasarkan penjelasan di atas, digambarkan bagan kerangka berpikir dari penelitian ini sebagai berikut:



Gambar 2. 1 Kerangka Berpikir