BAB II
KAJIAN PUSTAKA

2.1 Teori Dasar

2.1.1 *Artificial Intelliegence*

Artificial intelligence atau kecerdasan buatan dapat diketahui sebagai salah satu cabang ilmu pengetahuan yang mencoba membuat komputer dapat berfikir dan mengambil keputusan layaknya manusia.

Menurut Hartati dan Iswanti (2008: 1) *Artificial Intelligence* atau kecerdasan buatan adalah salah satu bidang ilmu komputer yang mendayagunakan komputer sehingga dapat berperilaku cerdas seperti manusia. Ilmu komputer tersebut mengembangkan perangkat lunak dan perangkat keras untuk meniru tindakan manusia. Aktifitas manusia yang ditirukan seperti penalaran, penglihatan, pembelajaran, pemecahan masalah, pemahaman bahasa alami, dan sebagainya. Sesuai dengan teknologi tersebut, maka teknologi kecerdasan buatan dipelajari dalam bidang-bidang seperti: Robotika (*Robotics*), Penglihatan Komputer (*Computer vision*), Pengolahan Bahasa Alami (*Natural Lenguage Processing*), Pengenalan Pola (*Pattern Recognition*), Sistem Syaraf Buatan (*Artificial Neural System*), Pengenalan Suara (*Speech Recognition*), dan Sistem Pakar (*Expert System*).

2.1.2 Sistem Pakar

Sistem Pakar atau *expert system* dapat diketahui sebagai suatu sistem yang berusaha memindahkan pengetahuan seorang pakar atau ahli ke dalam komputer agar dapat menyelesaikan permasalahan tertentu layaknya seorang pakar.

Menurut Kusrini (2008: 3) Sistem pakar adalah aplikasi berbasis komputer yang digunakan untuk menyelesaikan masalah sebagaimana yang dipikirkan oleh pakar. Pakar yang dimaksud di sini adalah orang yang mempunyai keahlian khusus yang dapat menyelesaikan masalah yang tidak dapat diselesaikan oleh orang awam. Sebagai contoh, dokter adalah seorang pakar yang mampu mendiagnosis penyakit yang diderita pasien, serta dapat memberikan penatalaksanaan terhadap penyakit tersebut. Tidak semua orang dapat mengambil keputusan mengenai diagnosis dan memberikan penatalaksanaan suatu penyakit. Contoh yang lain, montir adalah seorang yang punya keahlian dan pengalaman dalam menyelesaikan kerusakan mesin motor/mobil, psikolog adalah orang yang ahli dalam memahami kepribadian seseorang, dan lain-lain.

Sistem pakar yang mencoba memecahkan masalah yang biasanya hanya bisa dipecahkan oleh seorang pakar, dipandang berhasil ketika mampu mengambil keputusan seperti yang dilakukan oleh pakar aslinya baik dari sisi proses pengambilan keputusannya maupun hasil keputusan yang diperoleh. Sebuah sistem pakar memiliki 2 komponen utama yaitu basis pengetahuan dan mesin inferensi. Basis pengetahuan merupakan tempat penyimpanan pengetahuan dalam memori komputer, di mana pengetahuan diambil dari pengetahuan pakar.

Mesin inferensi merupakan otak dari aplikasi sistem pakar. Bagian inilah yang menuntun *user* untuk memasukkan fakta sehingga diperoleh suatu kesimpulan. Apa yang dilakukan oleh mesin inferensi ini didasarkan pada pengetahuan yang ada dalam basis pengetahuan.

Menurut Andi (2009: 3-4) terdapat banyak manfaat yang dapat diperoleh dengan mengembangkan sistem pakar. Antara lain:

1. Masyarakat awam dapat memanfaatkan keahlian di dalam bidang tertentu tanpa kehadiran langsung seorang pakar.
2. Meningkatkan produktifitas kerja, yaitu bertambah efiseiensi pekerjaan tertentu serta hasil solusi kerja.
3. Penghematan waktu dalam meneyelesaikan masalah yang kompleks.
4. Memberikan penyederhanaan solusi untuk kasus-kasus yang kompleks dan berulang.
5. Pengetahuan dari seorang pakar dapat didokumentasikan tanpa ada batasan waktu.
6. Memungkinkan penggabungan berbagai bidang pengetahuan dari berbagai pakar untuk dikombinasikan.

Selain banyak manfaat yang diperoleh, ada juga kelemahan pengembangan sistem pakar, yaitu:

1. Daya kerja dan produktivitas manusia menjadi berkurang karena semuanya dilakukan secara otomatis oleh sistem.
2. Pengembangan perangkat lunak sistem pakar lebih sulit dibandingkan perangkat lunak konvensional.

2.1.3 Runut Maju (*Forward Chaining*)

Metode *Forward Chaining* atau runut maju dapat diketahui sebagai teknik penalaran dalam pengambilan keputusan yang dimulai dari fakta-fakta terlebih dahulu untuk menentukan kesimpulan.

Menurut Wilson (1998) *dalam* Kusrini (2008: 8-11) runut maju berarti menggunakan himpunan aturan kondisi-aksi. Dalam metode ini, data digunakan untuk menentukan aturan mana yang akan dijalankan, kemudian aturan tersebut dijalankan. Memungkinkan proses menambahkan data ke memori kerja. Proses diulang sampai ditemukan suatu hasil.

Metode inferensi runut maju cocok digunakan untuk menangani masalah pengendalian (*controlling*) dan peramalan (*prognosis*) (Giarattano dan Riley, 1994 *dalam* Kusrini, 2008: 8). Untuk memudahkan pemahaman mengenai metode ini, akan diberikan ilustrasi kasus pembuatan sistem pakar sebagai berikut: ingin diperoleh konklusi dari daftar konklusi yang ada berdasarkan *premis-premis* dalam aturan fakta yang diberikan oleh *user*. Berikut adalah daftar aturannya:

Aturan 9:

Jika premis 1

Dan premis 2

Dan premis 3

Maka konklusi 1

Aturan 10:

Jika premis 1

Dan premis 3

Dan premis 4

Maka konklusi 2

Aturan 11:

Jika premis 2

Dan premis 3

Dan premis 5

Maka konklusi 3

Aturan 12:

Jika premis 1

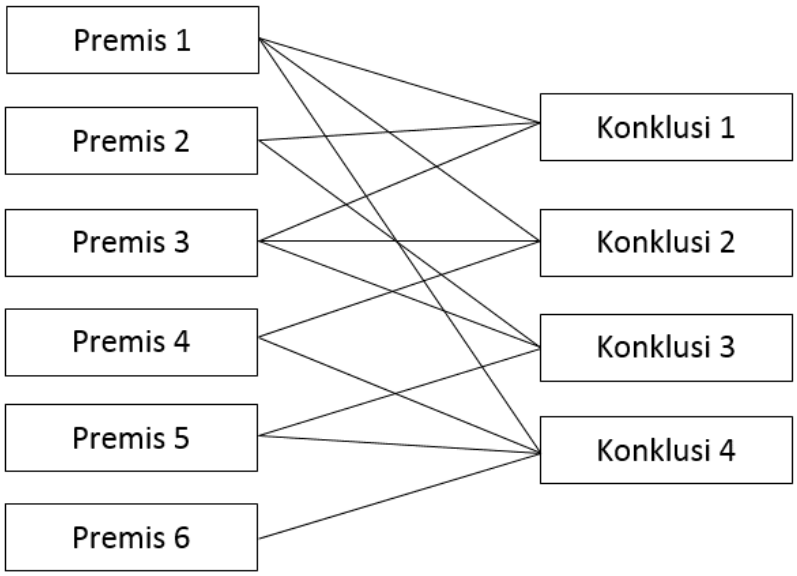
Dan premis 4

Dan premis 5

Dan premis 6

Maka konklusi 4

Penelusuran maju pada kasus ini adalah untuk mengetahui apakah suatu fakta yang dialami oleh pengguna itu termasuk konklusi 1, konklusi 2, konklusi 3, konklusi 4, atau bahkan bukan salah satu dari konklusi tersebut yang artinya sistem belum mampu mengambil keputusan karena keterbatasan aturan. Jika aturan ini kita gambarkan sebagai sebuah *graph* yang memetakan antara *premis-premis* dan konklusi-konklusi akan tampak seperti gambar 2.1



Gambar 2.1 *Graph* Pengetahuan
Sumber: Kusrini (2008)

Dalam penalaran ini, *user* diminta untuk memasukkan *premis-premis* yang dialami. Untuk memudahkan pengguna, sistem dapat memunculkan daftar *premis* yang mungkin sehingga *user* dapat memberikan umpan balik *premis* mana yang dialami dengan memilih 1 atau beberapa dari daftar *premis* yang tersedia. Berarti daftar *premis*nya adalah:

premis 1, premis 2, premis 3, premis 4, premis 5, dan premis 6

Berdasarkan *premis-premis* yang dipilih, maka sistem akan mencari aturan yang sesuai, sehingga akan diperoleh konklusinya. Seandainya *user* memilih *premis* 1, *premis* 2, dan *premis* 3 maka aturan yang terpilih adalah aturan 1 dengan konklusinya adalah konklusi 1. Seandainya *user* memilih *premis* 1 dan *premis* 6, maka sistem akan mengarah pada aturan 4 dengan konklusinya adalah konklusi 4, tetapi karena aturan tersebut *premis*nya adalah *premis* 1, *premis* 4, *premis* 5 dan

premis 6, maka *premis-premis* yang dipilih oleh *user* tidak cukup untuk mengambil kesimpulan konklusi 4 sebagai konklusi terpilih.

2.1.4 Runut Balik (*Backward Chaining*)

Metode *Backward Chaining* atau runut balik dapat diketahui sebagai kebalikan dari metode *Forward Chaining*. Penalaran dimulai dari tujuan sebagai kesimpulan dari permasalahan yang dihadapi, baru ke fakta-fakta atau penyebab yang ada.

Menurut Giarattano dan Riley (1994) *dalam* Kusrini (2008: 11-12) runut balik merupakan metode penalaran kebalikan dari runut maju. Dalam runut balik penalaran dimulai dengan tujuan tersebut. Runut balik disebut juga sebagai *goal-driven reasoning*, merupakan cara yang efisien untuk memecahkan masalah yang dimodelkan sebagai masalah pemilihan terstruktur. Tujuan inferensi adalah mengambil pilihan terbaik dari banyak kemungkinan. Metode inferensi runut balik ini cocok digunakan untuk memecahkan masalah diagnosis.

Dengan menggunakan kasus yang sama pada proses penalaran runut maju, yang ingin didapatkan pada penalaran ini juga sama yaitu salah satu konklusi dari konklusi 1, konklusi 2, konklusi 3, konklusi 4 atau bahkan tidak dari keempat konklusi tersebut. Penelusuran didasarkan pada suatu keyakinan bahwa ada kemungkinan konklusi dari daftar konklusi merupakan salah satu tujuan atau konklusi terpilih berdasarkan fakta yang diberikan oleh *user*. Sistem dengan urutan tertentu akan mengambil sebuah konklusi sebagai calon konklusinya. Misal urutannya adalah sesuai dengan urutan konklusi awalnya sistem akan mengambil hipotesis bahwa konklusinya adalah konklusi 1. Untuk membuktikan hipotesisnya,

sistem akan menncari *premis-premis* aturan yang mengandung konklusi 1. Setelah itu, sistem akan meminta umpan balik kepada *user* mengenai *premis-premis* yang ditemukan tersebut. Untuk konklusi 1 *premis*nya adalah *premis* 1, *premis* 2 dan *premis* 3, maka sistem akan mencari tahu apakah *user* memilih *premis-premis* tersebut.

Cara untuk mengambil umpan balik dari *user* bisa dilakukan dengan mencari dari daftar *premis* yang dipilih oleh *user* atau dengan menanyakan satu persatu *premis-premis* yang seharusnya dipilih. Jika ternyata ada *premis* yang tidak terpilih oleh *user* maka hipotesis terhadap konklusi tersebut gugur, yang artinya fakta yang dimasukkan *user* konklusinya bukan konklusi 1. Oleh karena itu, sistem akan melanjutkan hipotesis ke konklusi berikutnya. Demikian seterusnya sampai ditemukan konklusi yang semua *premis* dalam aturannya terpilih. Jika sampai akhir konklusi yang mungkin tidak ada *premis* yang terpenuhi, maka sistem akan mengambil kesimpulan bahwa konklusinya adalah diluar pengetahuannya, yang artinya sistem tidak akan menemukan solusi untuk *premis-premis* pilihan *user*.

2.2 Demam Berdarah *Dengue*

Penyakit demam berdarah *dengue* (DBD) dapat diketahui sebagai penyakit berbahaya di Indonesia, tidak sedikit jumlah korban jiwa akibat penyakit ini. Penyakit DBD disebabkan oleh virus dan ditularkan melalui nyamuk *aedes aegypti* yang biasanya berkembang biak di genangan-genangan air.

Menurut Nadesul (2016: 1) virus *dengue* penyebab DBD memerlukan bantuan nyamuk untuk berpindah ke tubuh manusia. Nyamuk sendiri harus jenis

nyamuk belang-belang hitam-putih *aedes*, dan bukan oleh nyamuk lainnya. Nyamuk rumah, nyamuk malaria, dan jemis nyamuk lainnya tidak dapat membawa virus *dengue*, sehingga bukan nyamuk penularnya.

Menurut Nadesul (2016:3-4) tubuh untuk pertama kali dimasuki virus *dengue*, akan terserang penyakit demam *dengue*. Di Indonesia dikenal sebagai "demam 5 hari". Dokter inggris menjulukinya "demam pelana kuda". Demam *dengue* bersifat khas, yakni 3 hari pertama demam tinggi (39-40 derajat *celcius*), kemudian demam mereda pada hari keempat, lalu demam bangkit kembali setelah hari kelima. Jadi, kalau digambarkan grafik demamnya akan menyerupai pelana kuda.

Setelah serangan virus *dengue* untuk pertama kali, tubuh akan membentuk kekebalan spesifik untuk *dengue* namun tidak bersifat absolut. Artinya masih mungkin diserang untuk kedua kalinya, atau lebih. Oleh karena ada lebih dari 1 virus *dengue*, maka seorang dapat terserang virus *dengue* lebih dari 1 kali.

Apabila tubuh yang sama terserang virus *dengue* tipe yang berbeda, maka muncullah penyakit DBD. Berbeda dengan demam *dengue*, selain demam yang khas *dengue*, disertai pula dengan gejala pendarahan, selain kemungkinan munculnya *shock*.

Pada infeksi virus *dengue* ulangan, terjadi reaksi imun yang lebih lambat di dalam tubuh. Respon tubuh yang sudah pernah diduduki virus *dengue* sebelumnya, akan lebih sengit sehingga gejala penyakitnya lebih hebat.

Pada tubuh yang kecukupan gizi, respon tubuh lebih sengit dibanding pada tubuh yang kekurangan gizi. DBD lebih hebat pada tubuh yang tidak kurang gizi, dan jarang pada tubuh yang kurang gizi.

Menurut Hastuti (2008: 12-13) tanda dan gejala penyakit demam berdarah adalah tidak khas, bervariasi pada tiap penderita berdasarkan derajat yang dialaminya. Umumnya penderita akan mengalami tanda dan gejala-gejala berikut:

1. Demam.
2. Perdarahan/bintik-bintik merah pada kulit.
3. Perdarahan lain: mimisan, perdarahan gusi.
4. Keluhan pada saluran pernapasan: batuk, pilek.
5. Keluhan pada saluran pencernaan ataupun sakit waktu menelan.
6. Keluhan pada bagian tubuh yang lain: nyeri/sakit kepala; nyeri pada otot, tulang, sendi, dan ulu hati; pegal-pegal pada seluruh tubuh.
7. Dapat juga dijumpai adanya pembesaran hati, limpa, dan kelenjar getah bening, yang akan kembali normal pada masa penyembuhan.
8. Pada keadaan yang berat, penderita akan jatuh pada keadaan renjatan/*shock*, yang dikenal dengan DSS (*Dengue Shock Sindrome*), dengan tanda-tanda sebagai berikut:

a. Kulit terasa lembap dan dingin.
b. Tekanan darah menurun, nadi cepat dan lemah.
c. Nyeri perut yang hebat.
d. Terjadi perdarahan, baik dari mulut, hidung, maupun anus yang terlihat seperti tinja hitam.

e. Lemah, mengantuk, terjadi penurunan tingkat kesadaran.

f. Gelisah.

g. Tarnpak kebiru-biruan pada sekitar rnulut, hidung, dan ujung-ujung jari.

h. Tidak buang air kecil selama 4-6 jam.

2.3 *Software* Pendukung

2.3.1 *Unified Modeling Language* (UML)

Unified Modelling Language atau UML diketahui sebagai suatu bahasa yang sudah menjadi standar pada visualisasi, perancangan dan juga pendokumentasian sistem *software*. Saat ini UML sudah menjadi bahasa standar dalam penulisan *blue print software*.

Menurut Rosa dan Shalahuddin (2013: 137-138) UML muncul karena adanya kebutuhan akan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari suatu perangkat lunak. UML merupakan bahasa visual untuk permodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan permodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek. Perkembangan penggunaan UML tergantung pada level abstraksi penggunaannya. Jadi belum tentu pandangan yang berbeda dalam penggunaan UML adalah salah, jika ada banyak perbedaan dan interpretasi dalam bidang informasi merupakan hal yang sangat wajar.

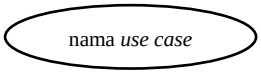
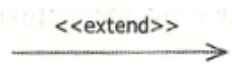
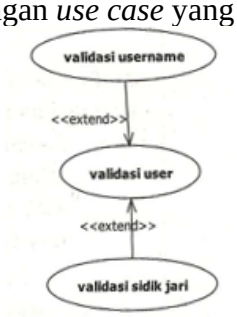
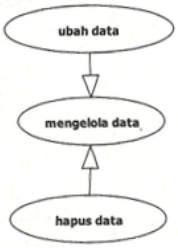
2.3.1.1 *Use Case Diagram*

Use case diagram diketahui sebagai salah satu jenis diagram pada UML yang menggambarkan interaksi antara sistem dan aktor, *use case diagram* juga dapat mendeskripsikan tipe interaksi antara si pemakai sistem dengan sistemnya

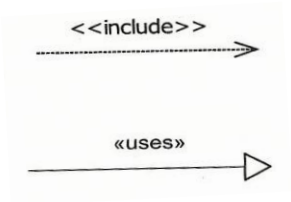
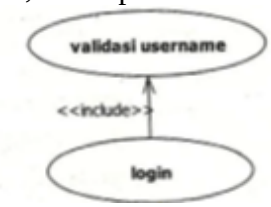
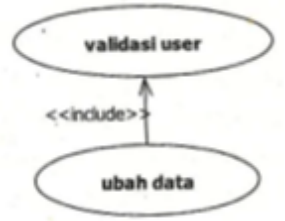
Menurut Rosa, *dkk.* (2013: 155-158) *use case* atau *use case diagram* merupakan permodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *use case* mendeskripsikan sebuah interaksi satu atau lebih aktor dengan sistem informasi yang akan dibuat. secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada 2 hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol aktor adalah gambar orang, tepi aktor belum tentu merupakan orang. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit atau aktor. Berikut simbol-simbol yang ada pada diagram *use case*:

Tabel 2.1 Simbol *use case*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor/*actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan dengan kata benda di awal frase nama aktor.
Asosiasi/*association* 	Komunikasi antara aktor dan *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi/*extend* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu; mirip dengan konsep *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misal:  Arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara 2 buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:  Arah panah mengarah ke *use case* yang menjadi generalisasinya (umum)

Tabel 2.1 Lanjutan simbol *use case*

Simbol	Deskripsi
Menggunakan/*include*/*uses* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerluklan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. Ada 2 sudut pandang yang cukup besar mengenai include di *use case*: 1. *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, misal pada kasus berikut:  2. *Include* berarti *use case* tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut:  Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangn dan interpretasi yang dibutuhkan.

Sumber: Rosa, *dkk.* (2013: 156-158)

2.3.1.2 *Activity Diagram*

Activity diagram atau diagram aktivitas diketahui sebagai salah satu jenis diagram pada UML yang dapat memodelkan proses-proses apa saja yang terjadi pada sistem.

Menurut Rosa, *dkk.* (2013: 161-162) diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau

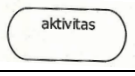
proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

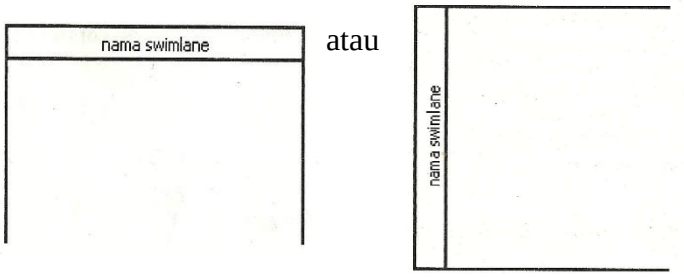
1. Rancangan proses bisnis dimana setiap urusan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut adalah simbol-simbol yang ada pada *activity diagram* (diagram aktivitas):

Tabel 2.2 Simbol *activity diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari 1.
Penggabungan/*join*	Asosiasi penggabungan dimana lebih dari 1 aktivitas digabungkan menjadi 1.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

Tabel 2.2 Lanjutan simbol *activity diagram*

Simbol	Deskripsi
Swimlane	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi. 

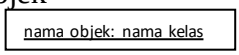
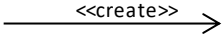
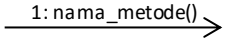
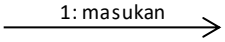
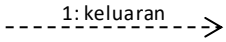
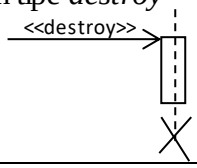
Sumber: Rosa, *dkk.* (2013: 162-163)

2.3.1.3 *Sequence Diagram*

Menurut Rosa, *dkk.* (2011: 137-138) *sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu, untuk menggambarkan *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Banyaknya *sequence diagram* yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada *sequence* diagram sehingga semakin banyak *use case* yang didefinisikan maka *sequence diagram* yang harus dibuat juga semakin banyak. Berikut adalah simbol-simbol yang terdapat pada *sequence diagram*:

Tabel 2.3 Simbol *Sequence diagram*

Simbol	Deskripsi
Aktor 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor asalah gambar orang, aktor belum tentu merupakan orang.
Garis hidup/*lifetime*	Menyatakan garis hidup suatu objek.
Objek 	Menyatakan objek yang berinteraksi pesan.
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.
Pesan tipe *create* 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek yang lain, atau dirinya sendiri.
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada ibjek yang dikirimi.
Pesan tipe *return* 	Menyatakan bahwa suatu objek telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek lain, arah panah mengarah pada objek yang diakhiri.

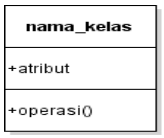
Sumber: Rosa, *dkk*. (2011: 138-139)

2.3.1.4 *Class Diagram*

Menurut Rosa, *dkk.* (2011: 122-123) *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki kelas.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem. Berikut adalah simbol-simbol yang ada pada *class diagram*:

Tabel 2.4 Simbol *class diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka/*interface*  nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi/*association*	Relasi antar kelas dengan makna umum
Asosiasi berarah/ *direrected association*	Relasi antar kelas dengan makna yang satu digunakan oleh kelas yang lain
Generalisasi	Relasi antar kelas dengan makna generalisai-spesialisasi (umum-khusus)
Kebergantungan/ *dependency*	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi/*aggregation*	Rrelasi antar kelas dengan makna semua bagian (*whole-part*)

Sumber: Rosa, *dkk.* (2011: 123-124)

2.3.2 *Hyper Text Markup Language* (HTML)

HTML diketahui sebagai singkatan dari *HyperText Markup Language* diketahui sebagai *script* untuk menyusun dokumen-dokumen *web*. Mendesain HTML berarti melakukan suatu tindakan pemrograman.

Menurut Saputra (2012: 1) HTML merupakan singkatan dari *Hyper Text Markup Language*. HTML bisa disebut sebagai bahasa paling dasar dan penting yang digunakan untuk menampilkan dan mengelola tampilan pada halaman *website*. HTML digunakan untuk menampilkan berbagai informasi didalam sebuah penjelajah *web* internet dan *formatting hypertext* sederhana yang ditulis ke dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegrasi. HTML menggunakan 2 macam ekstensi *file* yaitu .htm dan .html.

Menurut Saputra (2012: 17-18) saat ini, HTML versi 5 yang paling marak dibincangkan didunia maya, HTML 5 memiliki keunggulan mampu menyederhanakan kode-kode HTML terdahulu menjadi lebih ringkas. HTML 5 juga dapat ditulis dengan cara html ataupun xhtml. Berikut adalah fitur-fitur terbaru HTML 5:

1. Unsur canvas untuk gambar.
2. Bentuk kontrol *form* seperti kalender, tanggal, waktu, email, url, dan *search*.
3. Elemen konten yang lebih spesifik, seperti artikel, *footer*, *header*, navigasi, dan *section*.
4. Dukungan yang lebih baik untuk penyimpanan secara *offline.*
5. Dukungan untuk pemutaran *video* dan *audio*.

2.3.3 *Hypertext Preprocessor* (PHP)

PHP dapat digunakan untuk membuat tampilan *web* menjadi dinamis. *Website* dinamis yang dapat dibuat menggunakan PHP adalah situs *web* yang dapat menyesuaikan tampilan konten tergantung situasi. Untuk membuat halaman *web*, sebenarnya PHP bukanlah bahasa pemrograman yang wajib digunakan. Kita dapat saja membuat *website* hanya menggunakan HTML saja. *Web* yang dihasilkan dengan HTML (dan CSS) ini dikenal dengan *website* statis, dimana konten dan halaman *web* bersifat tetap.

Menurut Wahana (2013:309) PHP merupakan singkatan dari *hypertext preprocessor*. PHP digunakan untuk membuat tampilan *web* menjadi lebih dinamis. Dengan PHP kita bisa menampilkan atau menjalankan beberapa *file* dalam 1 *file* dengan cara di *include* atau *require*. PHP itu sendiri sudah dapat berinteraksi dengan kelengkapan yang berbeda, yaitu seperti: *DBM, FilePro (Personic, Inc)*, *Informix, Ingres, Interbase, Microsoft Access, MSSQL, MySQL, Oracle, PostgrSQL,* dan *Sybase*.

Menurut Westriningsih (2012: 76) PHP merupakan bahasa pemrograman berbasis *web* yang memiliki kemampuan untuk memproses dan mengolah data secara dinamis. PHP dapat dikatakan sebagai sebuah *server-side embedded script lenguage*, artinya sintak-sintak dan perintah program yang ditulis akan sepenuhnya dijalankan oleh server tetapi dapat disertakan pada halaman HTML biasa. Aplikasi-aplikasi yang dibangun menggunakan PHP umumnya akan memberikan hasil pada *web browser* tetapi prosesnya secara keseluruhan dijalankan pada *server*.

2.3.4 XAMPP

XAMPP diketahui sebagai perangkat lunak bebas (*free software*), yang mendukung untuk banyak sistem operasi, yang merupakan kompilasi dari beberapa program.

Menurut Wicaksono dan Community (2008: 7) XAMPP adalah sebuah *sofware* yang berfungsi untuk menjalankan *website* berbasis PHP dan menggunakan pengolah data MySQL di komputer lokal. XAMPP berperan sebagai *web server* pada komputer. XAMPP juga dapat disebut sebuah *CPanel server virtual*, yang dapat membantu membuat *preview* sehingga dapat memodifikasi *website* tanpa harus *online* atau terakses dengan *internet.*

Menurut Wahana (2013: 311) *web server* adalah suatu program komputer yang mempunyai tanggung jawab atau tugas menerima permintaan HTTP dari komputer klien, yang dikenal dengan nama *web browser*, dan melayani mereka dengan menyediakan respon HTTP berupa konten data, biasanya berupa halaman *web* yang terdiri dari dokumen HTML, dan obyek terkait seperti gambar, dan lain-lain.

2.3.5 MySQL *Database*

MySQL diketahui sebagai sistem manajemen *database* SQL yang bersifat *Open Source* dan paling populer saat ini. *Database* ini dibuat untuk keperluan sistem *database* yang cepat, handal dan mudah digunakan.

Menurut Saputra (2012: 77-78) MySQL merupakan salah satu *database* kelas dunia yang sangat cocok bila dipadukan dengan bahasa pemrograman PHP. MySQL bekerja menggunakan bahasa SQL (*Structure Query Lenguage*) yang merupakan bahasa standar yang digunakan untuk memanipulasi *database*. Perintah yang paling sering digunakan dalam MySQL adalah *SELECT* (mengambil), *INSERT* (menambah), *UPDATE* (mengubah), dan *DELETE* (menghapus). Selain itu, SQL juga menyediakan perintah untuk membuat *database*, *field*, ataupun *index* untuk menambah atau menghapus data.

Beberapa alasan yang menjadikan *database* MySQL sangat diminati oleh para programmer, yaitu:

1. Bersifat *open source*.
2. Menggunakn bahasa SQL, yang merupakan standar bahasa dalam pengolahan data.
3. *Performance* dan *reliable*, pemrosesan *database*-nya sangat cepat dan stabil.
4. Sangat mudah dipelajari.
5. Memiliki dukungan (*group*) pengguna MySQL.
6. Lintas *platform*, dapat digunakan pada berbagai sistem operasi yang berbeda.
7. Dapat digunakan oleh banyak *user* dalam waktu yang bersamaan tanpa mengalami konflik.

2.4 Penelitian Terdahulu

Berikut beberapa penelitian yang penulis jadikan referensi untuk penelitian sistem pakar ini:

1. **Sadly Syamsuddin dan Ahyuna** (2014) Sistem Pakar untuk Diagnosa Penyakit yang Disebabkan oleh Nyamuk Berbasis *WEB*, diperoleh fakta: perkembangan yang pesat pada ilmu kedokteran dan penemuan obat-obatan serta metode penyembuhan melahirkan suatu ketergantungan satu sama lain antara dokter dan pasiennya. Masalah akan timbul apabila dokter tidak dapat mengidentifikasi dengan tepat penyakit menurut gejala-gejala yang diderita oleh pasiennya, begitu pula dengan pasien yang mungkin sulit untuk memberi pemaparan yang jelas mengenai gejala-gejala yang mereka derita kepada dokternya. Oleh karena itu dibutuhkan suatu sistem pakar yang mampu mengadopsi pengetahuan manusia atau dokter ke komputer yang dirancang untuk memodelkan kemampuan manusia atau dokter tersebut menyelesaikan masalah seperti layaknya seorang pakar. Penelitian ini merancang suatu sistem pakar dimana sistem pakar ini bertujuan untuk mendiagnosa penyakit yang disebabkan oleh nyamuk. Sistem pakar ini dirancangan dengan alat pemodelan sistem *Unified Modeling Language* (UML), MySQL sebagai sistem manajemen basis data SQL dan PHP sebagai bahasa pemrograman. Setelah sistem dapat diimplementasikan maka dilakukanlah pengujian sistem dengan metode *Black Box*. Hasil dari sistem yang dibangun adalah sebuah sistem pakar yang mampu melakukan diagnosa penyakit yang disebabkan oleh nyamuk dengan tingkat akurasi yang baik dan hampir tidak ditemukan kesalahan yang ada pada tiap *form* komponen yang diuji.
2. **Muhammad Silmi, Eko Adi Sarwoko, dan Kushartantya** (2013) Sistem Pakar Berbasis *Web* dan *Mobile Web* untuk Mendiagnosis Penyakit Darah pada

Manusia dengan Menggunakan Metode inferensi *Forward Chaining*, diperoleh fakta: sistem pakar (*expert system*) secara umum adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer yang dirancang dan diimplementasikan dengan bahasa pemrograman tertentu agar komputer dapat menyelesaikan masalah seperti yang dilakukan oleh para ahli. Saat ini sistem pakar telah banyak dikembangkan dalam berbagai macam bidang, salah satunya adalah bidang kesehatan. Sistem pakar dalam bidang kesehatan banyak dikembangkan untuk mendeteksi berbagai macam penyakit dengan menggunakan berbagai macam metode, salah satunya menggunakan metode inferensi *Forward Chaining*. Metode inferensi *Forward Chaining* merupakan metode inferensi penelusuran ke depan yang dibuat dengan perancangan yang mudah dan sesuai dengan aturan yang ada. Masyarakat memandang kesehatan sebagai sutau hal yang penting, salah satunya berkaitan dengan penyakit darah. Pengetahuan masyarakat umum tentang penyakit darah masih minim, masyarakat masih belum dapat mengetahui ataupun menentukan penyakit yang dideritanya. Penelitian ini, bertujuan untuk merancang sistem pakar yang dapat mendiagnosis penyakit darah menggunakan inferensi *Forward Chaining*. Aplikasi ini diharapkan dapat membantu masyarakat dalam mendiagnosa penyakit darah. Implementasi sistem pakar penyakit ini diharapkan memberikan kemudahan aksses bagi penggunanya, melalui penggunaan media sarana berbasis *web* dan *mobile web*.

3. **Siska Iriani** (2015) Penerapan Metode *Backward Chaining* pada Sistem Pakar Diagnosa Penyakit Tulang Manusia, diperoleh fakta: sistem pakar untuk

mendiagnosa penyakit tulang pada manusia merupakan aplikasi yang berguna untuk mengetahui jenis penyakit pada tulang manusia, beserta gejala yang dialami pemakai. Pembahasan utama dalam sistem ini adalah perancangan dan pembuatan sistem pakar untuk melakukan diagnosa dan selanjutnya memberikan informasi–informasi mengenai penyakit tulang, gejala-gejala pada penyakit tersebut serta cara pencegahan, pengobatan dan penyebabnya. Model inferensi yang digunakan dalam pembuatan sistem pakar ini adalah penalaran mundur *(Backward Chaining)* sedangkan teknik pencarian menggunakan *Depth First Search*. Penentuan diagnosa dalam sistem pakar ini dilakukan melalui proses konsultasi antara sistem dan pemakai. Jawaban disesuaikan dengan aturan yang berada di dalam sistem, jika jawaban yang dimasukkan sesuai dengan aturan yang berlaku, maka sistem ini akan memberikan hasil diagnosa berupa informasi penyakit. Diharapkan dengan dibuatnya Sistem Pakar Untuk Mendiagnosa Penyakit Tulang Pada Manusia ini dapat memberikan hasil diagnosa, penyebab, pengobatan, serta pencegahan terhadap suatu penyakit. Sistem ini disebut dengan Sistem Pakar (*Expert Sistem).*

4. **Benny Wijaya dan Maria Irmina Prasetiyowati** (2012) Rancang Bangun Sistem Pakar Pendiagnosa Penyakit Demam *Typhoid* dan Demam Berdarah *Dengue* dengan Metode *Forward Chaining*, diperoleh fakta: penyakit demam *typhoid* dan demam berdarah *dengue* merupakan penyakit yang umum di Indonesia. Kedua penyakit ini memiliki gejala yang hampir sama. Apabila pada saat menangani pasien, dokter salah mengetahui jenis penyakit yang diderita, hal ini dapat menyebabkan kematian. Oleh karena itu, dibuatlah Sistem pakar

pendiagnosa penyakit demam *typhoid* dan demam berdarah *dengue*. Sistem pakar ini dibangun menggunakan metode inferensi *Forward Chaining*. Metode inferensi *Forward Chaining* ini diimplementasikan dengan menggunakan bahasa pemrograman C#. Sistem pakar yang dirancang dalam skripsi ini merupakan *rule-based expert system*. Dari hasil uji coba sistem dapat disimpulkan bahwa tingkat keakuratan sistem adalah 93,33%, rata-rata waktu yang dibutuhkan untuk mendiagnosa penyakit menggunakan sistem ini adalah 3,16 menit. Tingkat keakuratan sistem bergantung pada *knowledge base* yang disimpan dalam *database*.

5. **Nur Anjas Sari** (2013) Sistem Pakar Mendiagnosa Penyakit Demam Berdarah Menggunakan Metode *Certainty Factor*, diperoleh fakta: penyakit demam berdarah merupakan penyakit infeksi yang disebabkan oleh virus *dengue* dan ditularkan melalui gigitan nyamuk *Aedes aegepty* dan *Aedes albopictus*. Demam berdarah *dangue* merupakan salah satu penyakit menular yang sering menimbulkan wabah dan menyebabkan kematian. Seringkali penyakit demam berdarah terlambat didiagnosa. Pada penelitian ini penulis membuat suatu penerapan metode *certainty factor* agar masyarakat dapat mengenali dan menanggulangi penyakit yang dideritanya. Sistem pakar untuk diagnosa penyakit demam berdarah ini merupakan suatu sistem pakar yang dirancang sebagai alat bantu untuk mendiagnosa penyakit demam berdarah dengan basis pengetahuan yang dinamis. Dimana sistem pakar merupakan sistem komputer yang dapat melakukan penalaran seorang pakar dengan keahlian pada suatu keahlian tertentu. Sistem pakar dapat menggantikan peran seorang pakar yang

prinsip kerjanya dapat memberikan hasil yang pasti, seperti yang dilakukan oleh seorang pakar. Metode sistem pakar yang dipakai adalah *certainty factor*. Sistem pakar ini akan menampilkan pilihan gejala yang dapat dipilih oleh *user,* dimana setiap pilihan gejala akan membawa *user* kepada pilihan gejala selanjutnya sampai mendapatkan hasil akhir. Pada hasil akhir, sistem akan menampilkan pilihan gejala *user*, dan penyakit yang diderita. Sistem tersebut memberikan hasil berupa kemungkinan penyakit yang dialami, persentase keyakinan, serta nilai keyakinan yang diberikan oleh pengguna dalam menjawab pertanyaan selama sesi konsultasi ketika menggunakan sistem ini.

2.5 Kerangka Pemikiran

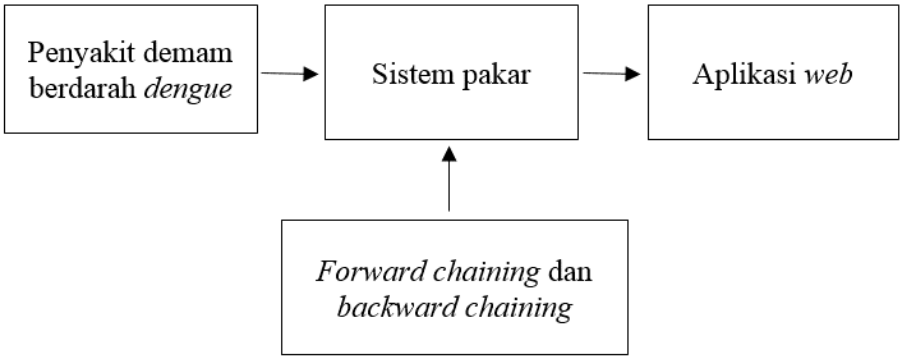
Kerangka pemikiran diketahui sebagai suatu gambaran yang menjelaskan secara garis besar alur sebuah penelitian.

Menurut Sekaran *dalam* Sugiyono (2012: 60) Kerangka berpikir merupakan model konseptual tentang bagaimana teori berhubungan dengan berbagai faktor yang telah diidentifikasi sebagai hal yang penting. Kerangka berfikir yang baik akan menjelaskan secara teoritis pertautan antar variabel yang akan diteliti.

Identifikasi masalah pada penelitian ini adalah: Penyakit DBD dapat menyerang semua kelompok usia, penyakit ini ditandai dengan demam mendadak tinggi selama 2 sampai 5 hari, yang disertai sakit kepala, nyeri otot, nyeri sendi, ruam (kemerahan pada muka/tubuh), lemah, mual, nyeri perut, serta dapat disertai gejala perdarahan yang bersifat ringan (perdarahan kulit atau bintik merah dibawah kulit, perdarahan gusi, dan perdarahan dari hidung seperti mimisan) sampai

perdarahan yang berat (pup hitam), dan karena gejala awal DBD adalah demam, masyarakat sering menyalah artikan sebagai demam biasa.

Berdasarkan identifikasi masalah diatas, maka kerangka pemikiran dari penelitian ini adalah sebagai berikut:



Gambar 2.2 Kerangka pemikiran
Sumber: Data penelitian (2017)

Pada gambar 2.2 diatas, peneliti meneliti mengenai penyakit demam berdarah *dengue* yang kemudian diimplementasikan ke dalam sistem pakar dengan metode *Forward Chaining* dan *Backward Chaining*. Hasil penelitian berupa aplikasi sistem pakar berbasis *web*.