BAB II

TINJAUAN PUSTAKA

2.1. Definisi Demam Berdarah *Dengue* (DBD)

Demam berdarah *Dengue* adalah penyakit menular yang disebabkan oleh virus *Dengue* dan ditularkan melalui gigitan nyamuk *Aedes aegypti*. Penyakit ini adalah penyakit demam akut yang disebabkan oleh serotipe virus *Dengue*, dan ditandai dengan empat gejala klinis utama yaitu demam yang tinggi, manifestasi perdarahan, *hepatomegali*, dan tanda-tanda kegagalan sirkulasi sampai timbulnya renjatan (sindrom renjatan *Dengue*) sebagai akibat dari kebocoran plasma yang dapat menyebabkan kematian (Departemen Kesehatan Republik Indonesia, 2010:2).

Penyakit Demam Berdarah *Dengue* (DBD) merupakan salah satu masalah kesehatan masyarakat di Indonesia yang jumlah penderitanya cenderung meningkat dan penyebarannya semakin luas dan penyakit ini merupakan penyakit menular yang terutama menyerang anak-anak (Widoyono, 2008).

Menurut WHO,1997, Demam Berdarah *Dengue* (DBD) dibagi kedalam 4 derajat setelah kriteria laboratoris terpenuhi, yaitu: (Herrygarna, 2012)

- Derajat I: demam mendadak 2-7 hari disertai gejala tidak khas dan satu-satunya manifestasi perdarahan yaitu tes *tourniquet* positif.
- Derajat II: derajat I disertai dengan perdarahan spontan di kulit atau perdarahan lainnya.

- Derajat III: derajat II ditambah kegagalan sirkulasi ringan, yaitu denyut nadi cepat, lemah, dan tekanan nadi yang menurun (20mmHg atau kurang) atau hipotensi, disertai dengan kulit yang dingin, lembab, dan penderita gelisah.
- Derajat IV: derajat III ditambah syok berat dengan nadi yang tidak teraba dan tekanan darah yang tidak terukur, dapat disertai penurunan kesadaran, sianosis, dan asidosis.

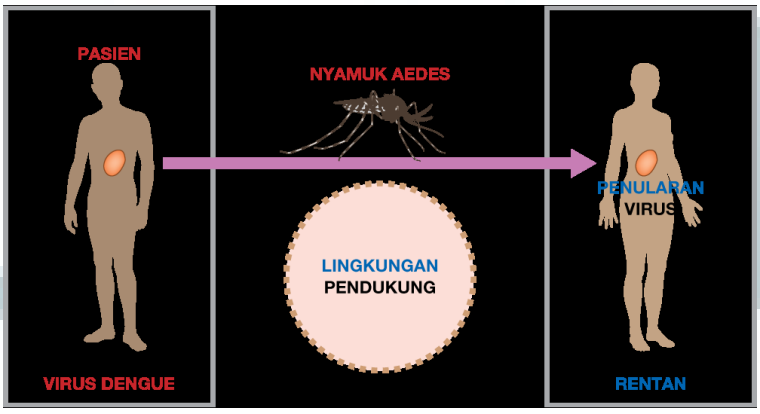
2.1.1. Penularan Virus *Dengue*

Penularan virus *dengue* dimulai saat nyamuk menghisap darah orang yang terinfeksi. Setelah melewati masa inkubasi ekstrinsik selama 4-7 hari di tubuh nyamuk, virus *dengue* dapat ditemukan pada cairan ludah nyamuk yang dikeluarkan melalui saluran pada hypofaring saat melakukan gigitan berikutnya. Orang sehat menerima virus *dengue* saat menerima gigitan infektif dari nyamuk yang terinfeksi tersebut. Virus yang masuk ke dalam aliran darah akan bereplikasi di dalam sel-sel makrofag jaringan dan darah, lalu selanjutnya keluar dari sel dan menginfeksi sel-sel berikutnya. Dibutuhkan waktu sekitar 5-7 hari untuk menghasilkan jumlah virus yang cukup agar dapat menyebabkan munculnya gejala. Periode sejak mendapat gigitan nyamuk terinfeksi sampai timbulnya gejala yang pertama dikenal sebagai masa inkubasi intrinsik di dalam tubuh manusia. Setelah timbul gejala inilah virus dapat terdeteksi di dalam darah, dan pasien menjadi infektif terhadap gigitan nyamuk berikutnya. Siklus penularan virus dari orang sehat ke nyamuk melewati masa inkubasi ekstrinsik di tubuh nyamuk.

Perpindahan dari nyamuk infektif ke orang sehat berikutnya, siklus intrinsik di tubuh pasien, sampai siap menginfeksi nyamuk berikutnya merupakan siklus perjalan virus yang perlu diperhatikan dalam hal pencegahan penularan virus *dengue* di masyarakat. Karena sebagian besar penanganan kasus penularan DBD dilakukan setelah pasien pergi berobat dan didiagnosis oleh dokter menderita DBD. Umumnya tindakan pencegahan penularan melalui vektor *control* menjadi terlambat karena nyamuk terinfeksi telah menggigit banyak orang selama masa inkubasi intrinsik pada manusia yang tanpa gejala sebelum datangnya upaya pemutusan rantai penularan / vektor control oleh petugas kesehatan. (Wahid, 2015)

2.1.2. Cara Penanganan dan Pencegahan

Prinsip penangan atau pencegahan penularan virus DBD sebenarnya meliputi pemotongan dari siklus virus pada tahap mana saja dalam siklus penularannya sebagaimana terlihat pada gambar 2.1.



Gambar 2.1. Skema Penularan Virus *Dengue*

Dalam gambar 2.1. tersebut, rantai penularan virus *dengue* dapat dilakukan pada 5 tempat: 1) mematikan virus pada saat masih di dalam tubuh penderita, namun sampai sekarang belum ada obat yang efektif terhadap virus *dengue*; 2) mencegah virus keluar dari tubuh orang sakit dan kontak dengan nyamuk sehingga penularan melalui gigitan nyamuk tidak terjadi. Dalam hal ini penderita yang infektif seharusnya ditempatkan di ruang isolasi yang bebas nyamuk, atau menggunakan perlindungan yang menghalangi gigitan nyamuk, misalnya penderita DBD harus memakai kelambu di dalam rumah sakit, atau menggunakan baju khusus yang menolak nyamuk; 3) melindungi orang sehat dari gigitan nyamuk yang infektif dengan cara menggunakan proteksi personal yang melindunginya dari gigitan nyamuk, misalnya penggunaan pakaian lengan panjang, penggunaan lotion penolak nyamuk, tidak mengunjungi pasien DBD yang tidak ditempatkan dalam ruang isolasi yang baik; 4) mencegah infeksi virus *dengue* walaupun tergigit nyamuk infektif. Hal ini dapat dilakukan jika dengan mengembangkan kekebalan yang efektif terhadap virus *dengue* dari serotipe tertentu melalui vaksinasi. Sayangnya, sampai saat ini belum ada vaksinasi terhadap virus *dengue* yang lengkap melindungi dari keempat serotipe yang beredar; dan 5) melakukan kegiatan vektor *control* yang mengurangi populasi nyamuk Aedes yang akan menularkan virus di daerah endemis. Kegiatan vektor control ini dapat langsung ditujukan kepada nyamuknya dengan menggunakan insektisida dan larvasida, juga dapat dilakukan dengan melakukan modifikasi lingkungan yang mengurangi tempat peridukan nyamuk dan perencanaan

arsitektur yang mengurangi struktur yang menyimpan air sebagai tempat perindukan nyamuk. (Wahid, 2015)

2.2. Sistem Pakar (*Expert System*)

Sistem pakar (*Expert System*) adalah sebuah sistem yang menggunakan pengetahuan manusia. pengetahuan tersebut dimasukan ke dalam sebuah komputer dan kemudian digunakan untuk menyelesaikan masalah-masalah yang biasanya membutuhkan kepakaran atau keahlian manusia (Sutojo, Mulyanto, dan Suhartono, 2010). Sistem pakar merupakan bidang yang dicirikan oleh sistem berbasis pengetahuan (*Knowledge Base System*), memungkinkan komputer dapat berfikir dan mengambil keputusan dari sekumpulan kaidah (Ignizio, 1991). Terdapat 4 komponen yang membentuk sistem pakar (Hu,1987), yaitu:

a) Basis Pengetahuan (*Knowledge Base*)

   Basis pengetahuan merupakan inti dari suatu sistem pakar, yaitu berupa representasi pengetahuan dari pakar. Basis pengetahuan tersusun atas fakta dan kaidah. Fakta adalah informasi tentang objek, peristiwa, atau situasi. Kaidah adalah cara untuk membangkitkan suatu fakta baru dari fakta yang sudah diketahui.

b) Basis Data (*Data Base*)

   Basis data terdiri atas semua fakta yang diperlukan, dimana fakta fakta tersebut digunakan untuk memenuhi kondisi dari kaidah-kaidah dalam sistem. Basis data menyimpan semua fakta, baik fakta awal pada saat sistem mulai beroperasi, maupun fakta-fakta yang diperoleh pada saat

proses penarikan kesimpulan sedang dilaksanakan. Basis data digunakan untuk menyimpan data hasil observasi dan data lain yang dibutuhkan selama pemrosesan.

c) Mesin Inferensi (*Inference Engine*)

Mesin inferensi berperan sebagai otak dari sistem pakar. Mesin inferensi berfungsi untuk memandu proses penalaran terhadap suatu kondisi, berdasarkan pada basis pengetahuan yang tersedia. Di dalam mesin inferensi terjadi proses untuk memanipulasi dan mengarahkan kaidah, model, dan fakta yang disimpan dalam basis pengetahuan dalam rangka mencapai solusi atau kesimpulan. Dalam prosesnya, mesin inferensi menggunakan strategi penalaran dan strategi pengendalian. Strategi penalaran terdiri dari strategi penalaran pasti (*Exact Reasoning*) dan strategi penalaran tak pasti (*Inexact Reasoning*). *Exact reasoning* akan dilakukan jika semua data yang dibutuhkan untuk menarik suatu kesimpulan tersedia, sedangkan *inexact reasoning* dilakukan pada keadaan sebaliknya. Strategi pengendalian berfungsi sebagai panduan arah dalam melakukan prose penalaran. Terdapat tiga teknik pengendalian yang sering digunakan, yaitu *forward chaining, backward chaining,* dan gabungan dari kedua teknik pengendalian tersebut.

d) Antarmuka Pengguna (*User Interface*)

Fasilitas ini digunakan sebagai perantara komunikasi antara pemakai dengan komputer.

Ada beberapa alasan mendasar adanya pengembangan sistem pakar untuk menggantikan seorang pakar, yaitu:

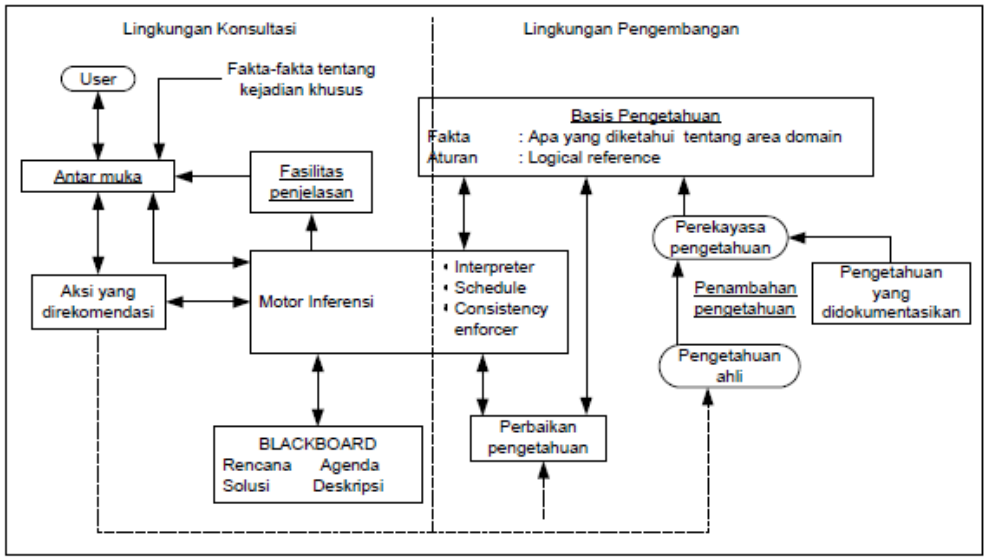
- Seorang pakar dapat pensiun atau pergi.
- Dapat menyediakan keahlian dari sebuah profesi setiap waktu dan di berbagai lokasi.
- Biaya untuk menggunakan tenaga seorang pakar sangat mahal, maka dari itu sistem pakar dapat mengurangi biaya atau beban untuk melakukan konsultasi.
- Secara otomatis mengerjakan tugas-tugas rutin yang membutuhkan seorang pakar.
- Kepakaran dibutuhkan juga pada lingkungan yang tidak bersahabat (*hostile environtment*).

2.2.1. Struktur Sistem Pakar

Sistem pakar terdiri dari dua bagian utama, yaitu bagian pengembangan dan konsultasi. Bagian pengembangan sistem pakar digunakan oleh penyusunnya untuk memasukkan pengetahuan dasar ke dalam lingkungan sistem informasi. Dalam hal ini operasionalisasi sistem pakar dibagi atas tiga modul, yaitu : Pengelolaan dialog (pengertian bahasa alamiah, konteks, dan lain - lain).

1) Pemecahan masalah (alasan, meta-logika, dan lain - lain).
2) Pengelolaan pengetahuan (penempatan fakta, aturan dan akses program secara algoritma klasik).
3) Struktur komunikasi antar tiga modul sebelumnya (butir 1-3).

Menurut Turban (1995) struktur skematis sebuah sistem pakar dapat dilihat pada Gambar 2.2. berikut :



Gambar 2.2. Struktur Sistem Pakar

Sistem pakar terdiri dari dua bagian pokok, yaitu: lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangandigunakan sebagai pembangun sistem pakar baik dari segi pembangun komponen maupun basis pengetahuan. Lingkungan konsultasi digunakan oleh seseorang yang bukan ahli untuk berkonsultasi. Komponen-komponen yang ada pada sistem pakar seperti pada Gambar 2.2 sebagai berikut :

1. Subsistem penambahan pengetahuan (Akuisisi Pengetahuan).

   Akusisi pengetahuan adalah akumulasi, transfer dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan ke dalam program komputer. Dalam tahap ini, perekayasa pengetahuan (*knowledge engineer*) berusaha menyerap pengetahuan untuk selanjutnya ditransfer ke dalam

basis pengetahuan. Pengetahuan diperoleh dari pakar, dilengkapi dengan buku, basis data, laporan penelitian dan pengalaman pemakai.

2. Basis pengetahuan (*Knowledge Base*)

   Berisi pengetahuan-pengetahuan yang dibutuhkan untuk memahami, memformulasikan dan menyelesaikan masalah. Basis pengetahuan merupakan bagian yang sangat penting dalam proses inferensi, yang di dalamnya menyimpan informasi dan aturan-aturan penyelesaian suatu pokok bahasan masalah beserta atributnya. Pada prinsipnya, basis pengetahuan mempunyai dua (2) komponen yaitu fakta-fakta dan aturan-aturan.

3. Mesin Inferensi (*Inference Engine*)

   Program yang berisi metodologi yang digunakan untuk melakukan penalaran terhadap informasi dalam basis pengetahuan dan blackboard, serta digunakan untuk memformulasikan konklusi.

4. *Workplace / Blackboard*

   Merupakan area dari sekumpulan memori kerja (*working memory*). *Workplace* digunakan untuk merekam kejadian yang sedang berlangsung termasuk keputusan sementara.

5. Antarmuka (*user interface*)

   Digunakan untuk media komunikasi antara *user* dan program. Menurut McLeod (1995), pada bagian ini terjadi dialog antara program dan pemakai, yang memungkinkan sistem pakar menerima instruksi dan

informasi (*input*) dari pemakai, juga memberikan informasi (output) kepada pemakai.

6. Subsistem penjelasan (*Explanation Facility*)

   Explanation Facility memungkinkan pengguna untuk mendapatkan penjelasan dari hasil konsultasi. Fasilitas penjelasan diberikan untuk menjelaskan bagaimana proses penarikan kesimpulan. Biasanya dengan cara memperlihatkan *rule* yang digunakan.

7. Perbaikan Pengetahuan (*Knowledge Refinement*)

   Sistem ini digunakan untuk mengevaluasi kinerja sistem pakar itu sendiri untuk melihat apakah pengetahuan-pengetahuan yang ada masih cocok untuk digunakan dimasa mendatang.

2.2.2. Ciri-ciri Sistem Pakar

Menurut Sri Kusumadewi (2003), sistem pakar yang baik harus memiliki ciri-ciri sebagai berikut:

a. Memiliki fasilitas informasi yang handal, baik dalam menampilkan langkah-langkah antara maupun dalam menjawab pentanyaan-pertanyaan tentang proses penyelesaian.
b. Mudah dimodifikasi, yaitu dengan menambah atau menghapus suatu kemampuan dari basis pengetahuannya.
c. Heuritik dalam menggunakan pengetahuan untuk mendapatkan penyelesaiannya.
d. Dapat digunakan dalam berbagai jenis komputer.
e. Memiliki kemampuan untuk beradaptasi.

2.2.3. Keuntungan dan Kelemahan Sistem Pakar

Menurut Kusrini (2006), ada beberapa keuntungan yang dapat diperoleh dengan mengembangkan sistem pakar antara lain:

1. Membuat seorang yang awam dapat bekerja seperti layaknya seorang pakar.
2. Dapat bekerja dengan informasi yang tidak lengkap atau tidak pasti.
3. Meningkatkan output dan produktivitas.
4. Meningkatkan kualitas.
5. Menyediakan nasihat atau solusi yang konsisten dan dapat mengurangi tingkat kesalahan.
6. Membuat peralatan yang kompleks dan mudah dioperasionalkan karena sistem pakar dapat melatih pekerja yang tidak berpengalaman.
7. Sistem tidak dapat lelah atau bosan.
8. Memungkinkan pemindahan pengetahuan ke lokasi yang jauh serta memperluas jangkauan seorang pakar, dapat diperoleh dan dipakai di mana saja.

Menurut Kusrini (2006), ada beberapa kelemahan yang dapat diperoleh dengan mengembangkan sistem pakar, antara lain:

1. Daya kerja dan produktivitas manusia menjadi berkurang karena semuanya dilakukan secara otomatis oleh sistem.

2. Pengembangan perangkat lunak sistem pakar lebih sulit dibandingkan dengan perangkat lunak konvensional.
3. Biaya pembuatannya mahal, karena seorang pakar membutuhkan pembuat aplikasi untuk membuat sistem pakar yang diinginkannya.

2.3. *Android*

Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. *Android* umum digunakan di *smartphone* dan juga *tablet* PC. Fungsinya sama seperti sistem operasi Symbian di Nokia, iOS di *Apple* dan *BlackBerry* OS. *Android* (Harahap, 2012:1). *Android* memiliki 4 (empat) karakteristik (Safaat,2011), yaitu:

1. Terbuka

   Android dibuat untuk dipakai siapa saja dan bersifat *open source.*

2. Semua aplikasi dibuat sama

   Android tidak membedakan antara aplikasi inti dengan aplikasi *third-party*. Semuanya memiliki akses yang sama terhadap kemampuan ponsel dalam menyediakan layanan dan aplikasi yang luas pada para pengguna.

3. Memecahkan batasan aplikasi

   Android memecahkan batasan untuk membangun aplikasi baru dan inovatif. Pengembang dapat menggabungkan informasi dari *web* dengan data yang ada pada ponsel individu.

4. Pengembangan aplikasi yang cepat dan mudah

   Android menyediakan akses ke berbagai *libraries* dan *tools* yang dapat digunakan untuk membangun aplikasi yang lebih kaya. Contohnya perangkat *Android* dapat berkomunikasi satu sama lain sehingga memungkinkan *peer-to-peer* aplikasi sosial.

2.4. Metode *Certainty factor*

2.4.1. Pengertian *Certainty factor*

Faktor kepastian (*Certainty factor*) diperkenalkan oleh Shortliffe Buchanan dalam pembuatan MYCIN *Certainty factor* (CF) merupakan nilai parameter klinis yang diberikan MYCIN untuk menunjukkan besarnya kepercayaan. *Certainty factor* menyatakan kepercayaan dalam sebuah kejadian (atau fakta atau hipotesis) berdasarkan bukti atau penilaian pakar (Turban, 2005). Metode ini sangat cocok untuk sistem pakar yang mendiagnosis sesuatu yang belum pasti. *Certainty factor* memberikan konsep pengenalan keyakinan dan ketidakyakinan yang diimplementasikan berdasarkan rumus sebagai berikut.

CF(H,E) = MB(H,E) – MD(H,E)……(2,1)

Keterangan:

CF(H,E) : *Certainty factor* dari hipotesis H yang dipengaruhi oleh gejala

(*evidence*) E. Besarnya CF berkisar antara –1 sampai dengan 1. Nilai –1 menunjukkan ketidak percayaan mutlak sedangkan nilai 1 menunjukkan kerpercayaan mutlak.

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

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

Metode MYCIN adalah metode yang digunakan untuk menggabungkan *evidence* pada *antecedent* sebuah aturan ditunjukkan oleh tabel 2.1 di bawah ini.

Tabel 2.1 Aturan MYCIN untuk mengkombinasikan *evidence antecedent*

***Evidence*, E**	***Antecedent* ketidakpastian**
E1 DAN E2	Min[CF(H,E1),CF(H,E2)]
E1 OR E2	Max[CF(H,E1),CF(H,E2)]
TIDAK E	-CF(H,E)

Bentuk dasar rumus *Certainty factor* sebuah aturan JIKA E MAKA H adalah sebagai berikut:

CF(H,e) = CF(E,e) * CF(H,E) ……(2,2)

dimana,

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

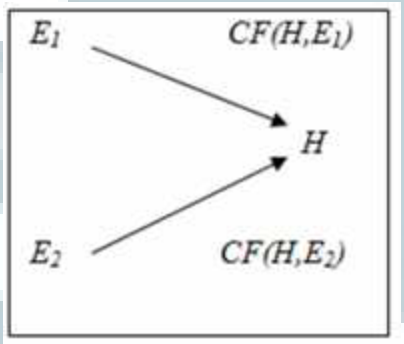
CF(H,E) : *Certainty factor* hipotesis dengan asumsi *evidence* diketahui dengan pasti, yaitu ketikaCF(E,e) = 1

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

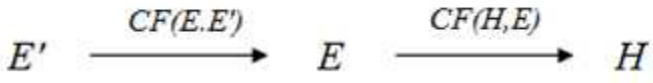
Jika semua *evidence* pada *antecedent* diketahui dengan pasti maka rumusnya akan menjadi:

CF(H,e) = CF(H,E) ……(2,3)

Ada dua macam kombinasi *Certainty factor* yaitu kombinasi paralel yang ditunjukkan oleh gambar 2.3 dan kombinasi sequensial yang ditunjukkan oleh gambar 2.4.



Gambar 2.3. Kombinasi Paralel *Certainty factor*



Gambar 2.4. Kombinasi Sequensial *Certainty factor*

Pada kondisi ini *evidence* E_1 dan E_2 mempengaruhi hipotesis yang sama yaitu H. Kedua *Certainty factor* CF(H, E_1) dan CF(H, E_2) dikombinasikan,

menghasilkan *Certainty factor* CF(H, E_1, E_2). Pada gamabr 2.5. menunjukkan fungsi kombinasi paralel tersebut didefinisikan sebagai berikut:

$$\begin{cases} x + y - xy & x,y \geq 0 \\ \dfrac{x + y}{1 - \min(|x|,|y|)} & = z \quad x,y \text{ berlawanan tanda} \\ x + y + xy & x,y < 0 \end{cases}$$

Gambar 2.5. Fungsi kombinasi paralel

Dimana x = CF(H,E_1), y = CF(H,E_2) dan z = CF(H,E_1E_2). *Certainty factor* kedua aturan dikombinasikan menghasilkan *certainty factor* CF(H,E'). Untuk menghitung kombinasi sequensial tersebut digunakan rumus berikut: CF(H,E') = CF(E, E') * CF(H, E) (Kusrini, 2006).

Logika metode *certainty factor* pada sesi konsultasi sistem, pengguna konsultasi diberi pilihan jawaban yang masing-masing memiliki bobot sebagai berikut (Suranti,2016):

Tabel 2.2. Bobot Nilai *User*

No.	**Keterangan**	**Nilai *User***
1.	Tidak	0
2.	Tidak Tahu	0.2
3.	Sedikit Yakin	0.4
4.	Cukup Yakin	0.6
5.	Yakin	0.8
6.	Sangat Yakin	1

Pada tabel 2.2. diatas nilai 0 menunjukkan bahwa pengguna konsultasi menginformasikan bahwa *user* tidak mengalami gejala yang ditanyakan oleh

sistem. Semakin pengguna konsultasi yakin ahwa gejala tersebut memang dialami, maka semakin tinggi hasil presentase keyakinan yang diperoleh. Proses penghitungan presentase keyakinan diawali dengan pemecahan sebuah kaidah yang memiliki premis majemuk , menjadi kaidah – kaidah yang memiiki premis tunggal. Kemudian masing-masing aturan baru dihitung nilai *certainty factor*-nya, sehinggal akan diperoleh nilai *certainty factor* untuk masing-masing aturan, kemudian nilai *certainty factor* tersebut dikombinasikan dan akan diperoleh persentase keyakinan pada penyakit yang diderita.

2.5. Metode Perancangan Aplikasi dengan *Rapid Application Development* (RAD)

2.5.1. Pengertian RAD

Rapid Application Development (RAD) merupakan salah satu metode yang digunakan untuk perancangan ataupun pengembangan suatu aplikasi. Berikut adalah beberapa pengertian *Rapid Application Development* (RAD) menurut para ahli:

1. "*Rapid Application Development* (RAD) adalah pengembangan dari beberapa metode atau teknik terstruktur (khususnya dalam pengolahan data untuk menghasilkan informasi), misalnya dengan mengintegrasikan metode *Prototyping*, metode SDLC dan teknik *Joint Apllication Development* untuk mempercepat pengembangan sistem informasi." (Susanto, 2004:353).

2. RAD adalah proses model perangkat lunak inkremental yang menekankan siklus pengembangan yang singkat. Model RAD adalah sebuah adaptasi "kecepatan tinggi" dari model *waterfall*, di mana perkembangan pesat dicapai dengan menggunakan pendekatan konstruksi berbasis komponen. Jika tiap-tiap kebutuhan dan batasan ruang lingkup proyek telah diketahui dengan baik, proses RAD memungkinkan tim pengembang untuk menciptakan sebuah "sistem yang berfungsi penuh" dalam jangka waktu yang sangat singkat (Roger,2012).
3. RAD adalah suatu pendekatan berorientasi objek terhadap pengembangan sistem yang mencakup suatu metode pengembangan serta perangkat-perangkat lunak. RAD bertujuan mempersingkat waktu yang biasanya diperlukan dalam siklus hidup pengembangan sistem tradisional antara perancangan dan penerapan suatu sistem informasi. Pada akhirnya, RAD sama-sama berusaha memenuhi syarat-syarat bisnis yang berubah secara cepat (Kendall,2010). Siklus RAD dapat dilihat pada Gambar 2.6. berikut:

Gambar 2.6. Siklus RAD Menurut (Kendall,2010)

Kelebihan metode RAD antara lain adalah meminimalkan kesalahan-kesalahan, keterlibatan *user* dan mempercepat waktu pengembangan sistem. Tujuan utama dari metode RAD adalah memberikan suatu sistem yang dapat memenuhi harapan dari para pemakai, akan tetapi sering kali di dalam melakukan pengembangan suatu sistem tidak melibatkan para pemakai sistem secara langsung, sehingga hal ini menyebabkan sistem informasi yang dibuat jauh dari harapan pemakai yang dapat berakibat sistem tersebut walaupun dapat diterima tetapi para pemakai enggan untuk menggunakannya atau bahkan para pemakai menolak untuk menggunakannya.

2.5.2. Tahapan-tahapan pada RAD

RAD digunakan pada pembuatan rancangan konstruksi aplikasi sistem, maka menekankan fase-fase. Ada tiga tahapan yang digunakan dalam RAD (Kendall, 2010) yaitu:

1. ***Requirements Planning*** **(Perencanaan Syarat-Syarat)**

   Pada tahap ini, pengguna dan penganalisis bertemu untuk mengidentifikasikan tujuan-tujuan aplikasi atau sistem serta untuk megidentifikasikan syarat-syarat informasi yang ditimbulkan dari tujuan-tujuan tersebut. Orientasi dalam fase ini adalah menyelesaikan masalah-masalah perusahaan. Meskipun teknologi informasi dan sistem bisa

mengarahkan sebagian dari sistem yang diajukan, fokusnya akan selalu tetap pada upaya pencapaian tujuan-tujuan perusahaan (Kendall, 2010).

2. *RAD Design Workshop* (*Workshop* Desain RAD)

Pada tahap ini berisi tahapan untuk merancang dan memperbaiki yang bisa digambarkan sebagai *workshop*. Penganalisis dan dan pemrogram dapat bekerja membangun dan menunjukkan representasi visual desain dan pola kerja kepada pengguna. *Workshop* desain ini dapat dilakukan selama beberapa hari tergantung dari ukuran aplikasi yang akan dikembangkan. Selama *workshop* desain RAD, pengguna merespon *prototipe* yang ada dan penganalisis memperbaiki modul-modul yang dirancang berdasarkan respon pengguna. Apabila sorang pengembangnya merupakan pengembang atau pengguna yang berpengalaman, Kendall menilai bahwa usaha kreatif ini dapat mendorong pengembangan sampai pada tingkat terakselerasi (Kendall, 2010).

3. *Implementation* (Implementasi)

Pada tahap implementasi ini, penganalisis bekerja dengan para pengguna secara intens selama *workshop* dan merancang aspek-aspek bisnis dan nonteknis perusahaan. Segera setelah aspek-aspek ini disetujui dan sistem-sistem dibangun dan disaring, sistem-sistem baru atau bagian dari sistem diuji coba dan kemudian diperkenalkan kepada organisasi (Kendall, 2010).

2.6. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk memvisualisasikan, menspesifikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan *software* berbasis *Object-Oriented*. UML juga memberikan standar penulisan sebuah sistem *blue print*, yang meliputi konsep proses bisnis, penulisan kelas-kelas dalam bahasa pemrograman yang spesifik, skema *database*, dan komponen-komponen yang diperlukan dalam sistem *software* (Wendi,2009).

Widodo dan Herlawati (2011:6-7), menjelaskan tentang kegunaan UML sebagai berikut :

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

Berikut adalah table tipe-tipe diagram UML:

Tabel 2.3. Tabel Tipe Diagram UML

No.	Diagram	Tujuan
1.	*Class*	Memperlihatkan himpunan kelas-kelas, antarmukaantarmuka, kolaborasi-kolaborasi, serta relasirelasi.
2.	*Package*	Memperlihatkan kumpulan kelas-kelas, merupakan dari diagram komponen.
3.	*Use Case*	Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari

		kelas).
4.	*Sequence*	Diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.
5.	*Communication*	Sebagai pengganti diagram kolaborasi UML 1.4 yang menekankan organisasi struktural dari obyek yang menerima serta mengirim pesan.
6.	*Statechart*	Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian atas aktivitas.
7.	*Activity*	Tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem.
8.	*Component*	Memperlihatkan organisasi serta kebergantungan sistem / perangkat lunak pada komponen komponen yang telah ada sebelumnya.
9.	*Deployment*	Memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*).

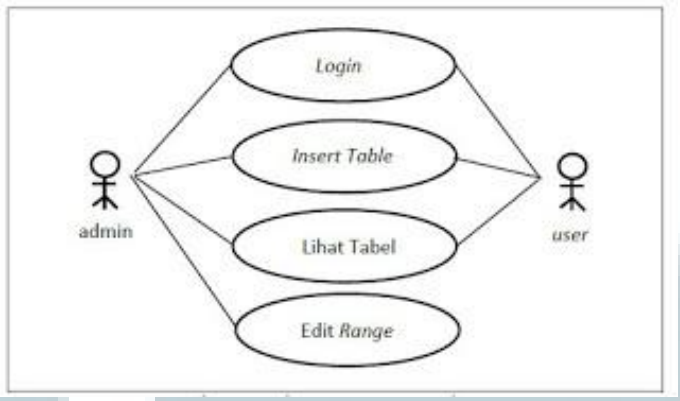
(Sumber: Widodo dan Herlawati (2011:10-12))

Menurut (Bentley, 2007: 371), *unified modeling language* (UML) merupakan kumpulan dari model yang akan digunakan untuk menjelaskan sistem dari perangkat lunak sebagai objek. Berikut adalah 8 jenis diagram yang menjadi bagian dari UML:

1. *Use Case* Diagram

   Use case menurut (Bentley, 2007:246) merupakan diagram yang menunjukkan interaksi antar sistem atau dengan pengguna dari sistem tersebut. Sehingga dapat diartikan bagaimana sistem ini akan bekerja dan apa yang di harapkan oleh pengguna ketika sedang berinteraksi dengan sistem. *Use case* dapat dinyatakan melalui *elips* yang berisi nama dari kegiatan yang dilakukan oleh sistem terhadap aktor-aktor yang bersangkutan dan hubungan antar *elips* yang mewakili satu tujuan dari

sistem dan menjelasan urutan langkah yang diambil untuk mencapai tujuan tersebut. Contoh gambar *use case* diagram bisa dilihat pada Gambar 2.7. berikut:

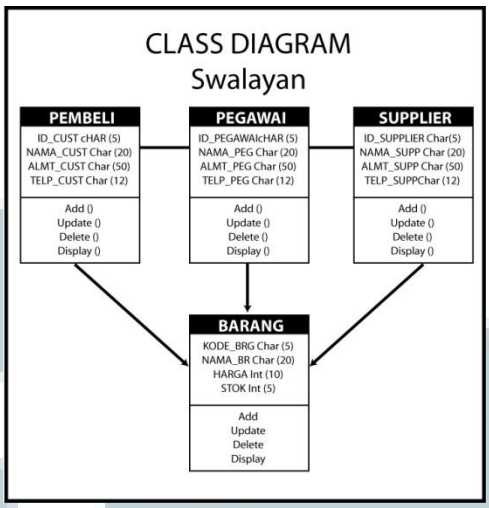


Gambar 2.7. Contoh Gambar *Use Case* Diagram

(Sumber: http://www.contohlengkap.com/2016/03/pengertian-dan-contoh-use-case-diagram.html)

2. *Class Diagram*

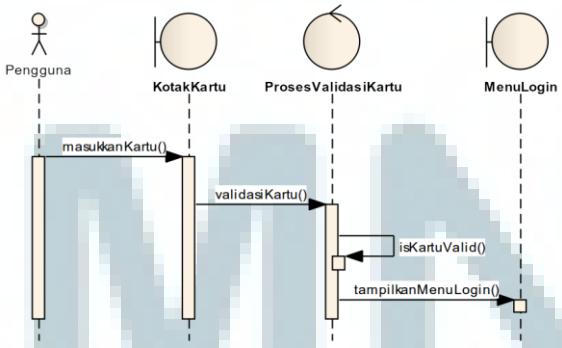
Menurut (Bentley, 2007:400) *Class diagram* adalah statik objek struktur dari sistem yang memperlihatkan kelas objek pada sistem yang saling berhubungan antara objek kelas. Berikut ini adalah contoh *class* diagram pada Gambar 2.8.



Gambar 2.8. Contoh Gambar *Class* Diagram

3. *Sequence Diagram*

Menurut (Bentley, 2007:394) *sequence* diagram adalah diagram yang digunakan untuk mendeskripsikan interaksi antara aktor dengan sistem dari *use case* skenario yang sudah dibuat. Berikut adalah contoh gambar *sequence diagram* dapat dilihat pada Gambar 2.9.:



Gambar 2.9. Contoh Gambar *Sequence Diagram*

2.7. *Smartphone*

Telepon pintar (*smartphone*) adalah telepon genggam yang mempunyai kemampuan tingkat tinggi, kadang-kadang dengan fungsi yang menyerupai *computer*. *Smartphone* merupakan teknologi *mobile phone* yang terus berkembang sejak awal kemunculannya dan akan terus mengalami berbagai inovasi untuk memenuhi kebutuhan komunikasi (Chuzaimah, Mabruroh, dan Dihan, 2010:315).

Tingkat penggunaan *smartphone* di zaman sekarang ini sangat tinggi, hal ini dikarenakan untuk memperlancar hubungan komunikasi bagi para kerabat atau keluarga yang jaraknya berjauhan. Fitur dan aplikasi tersebut sudah S*built-in* namun dapat ditambahkan secara gratis maupun berbayar melalui situs toko aplikasi yang disediakan oleh masing-masing *developer* sistem operasi *smartphone*, seperti Google *Play* (*Android*), APP *Store* (IOS), *Windows* Market *(Windows Phone)* dan *BlackBerry World (BlackBerry).*

2.8. *Android Studio*

Android Studio adalah sebuah *Integrated Development Environment* (IDE) untuk mengembangkan aplikasi pada *platform Android*. *Android Studio* merupakan pengembangkan dari *Eclipse* IDE, dan dibuat berdasarkan IDE *Java* populer, yaitu IntelliJ IDEA. *Android Studio* direncanakan untuk menggantikan *Eclipse* ke depannya sebagai IDE resmi untuk pengembangan aplikasi *Android*. (*Developer Android*, 2015).

2.9. *Java*

Java merupakan bahasa pemrograman berorientasi objek yang dapat berjalan pada *platform* yang berbeda baik *Windows*, *Linux*, serta *system* operasi lainnya. Jadi kita dapat membuat sebuah aplikasi dengan *java* pada *system* operasi *linux* dan selanjutnya menjalankan atau menginstal aplikasi tersebut pada *system* operasi *windows* dan juga sebaliknya tanpa mengalami masalah. Dengan menggunakan *java*, kita dapat mengembangkan banyak aplikasi yang dapat digunakan pada lingkungan yang berbeda, seperti pada: *Desktop*, *Mobile*, *Internet*, dan lain-lain (Supriyanto, 2010:10). Berikut adalah logo dari *Java* dapat dilihat pada Gambar 2.10

Gambar 2.10. Logo *Java*

Aplikasi *Android* akan ditulis dalam bahasa pemrograman *Java*, yaitu kode *Java* yang terkompilasi bersama-sama dengan data dan *file resources* yang dibutuhkan oleh aplikasi yang digabungkan oleh *aapt tools* menjadi paket *Android*, sebuah *file* yang ditandai dengan *suffix* .apk. *File* ini didistribusikan sebagai aplikasi dan diinstal pada perangkat *mobile* (Mulyadi, 2010).

Berikut adalah penjelasan mengenai paket aplikasi yang tersedia pada *Java* (Hakim dan Sutarto, 2009):

a. J2ME (*Java Micro Edition*)

Paket instalasi ini digunakan untuk mengembangkan *software* yang berjalan pada perangkat yang memiliki memori dan sumber daya yang kecil, seperti pada telepon selular, PDA, dan *smartcard*.

b. J2SE (*Java* 2 *Standard Edition*)

Paket instalasi ini digunakan untuk mengembangkan aplikasi desktop.

c. J2EE (*Java* 2 *Enterprise Edition*)

Paket instalasi ini digunakan untuk mengembangkan aplikasi pada lingkungan internet maupun aplikasi skala *enterprise*.

2.10. Teknik Pengumpulan Data

Teknik pengumpulan data merupakan cara yang digunakan peneliti untuk mendapatkan data dalam suatu penelitian. Sugiyono (2009:225) mengemukakan bahwa pengumpulan data dapat diperoleh dari hasil observasi, wawancara, dokumentasi, dan gabungan/triangulasi. Pada penelitian ini peneliti menggunakan teknik pengumpulan data dengan cara studi literatur, observasi dan wawancara:

2.10.1. Studi Literatur

Studi pustaka atau studi literatur adalah mengumpulkan informasi dan data dengan bantuan berbagai macam material yang ada di perpustakaan seperti dokumen, buku, catatan, majalah, kisah-kisah sejarah dan sebagainya (Mardalis:1999).

Definisi studi pustaka menurut Jonathan Sarwono (2006) adalah mempelajari berbagai buku referensi serta hasil penelitian sebelumnya yang sejenis yang berguna untuk mendapatkan landasan teori mengenai masalah yang akan diteliti.

2.10.2. Observasi

Observasi adalah suatu proses yang tersusun dari berbagai proses biologis dan psikologis, seperti proses pengamatan dan ingatan (Sugiyono, 2012:145). Observasi merupakan salah satu alat penilaian yang banyak digunakan dalam mengukur proses dan tingkah laku individu dalam sebuah kegiatan yang bisa diamati (Sudjana, 2011: 84).

2.10.3. Wawancara

Wawancara adalah proses pembekalan verbal, di mana dua orang atau lebih untuk menangani secara fisik, orang dapat melihat mukayang orang lain dan mendengarkan suara telinganya sendiri, ternyata informasi langsung alat pengumpulan pada beberapa jenis data sosial, baik yang tersembunyi (laten) atau manifest (Hadi, 1989:192).

Menurut Lexy J Moleong (1989), dengan metode wawancara peneliti dan responden/narasumber berhadapan langsung (tatap muka) untuk mendapatkan informasi secara lisan dengan mendapatkan data tujuan yang dapat menjelaskan masalah penelitian. Namun disini peneliti memilih melakukan wawancara mendalam, ini bertujuan untuk mengumpulkan informasi yang kompleks, yang

sebagian besar berisi pendapat, sikap, dan pengalaman pribadi, (Sulistyo dan Basuki, 2006:173).