BAB II
KAJIAN PUSTAKA

2.1 Teori Dasar

Deskripsi teori paling tidak berisi tentang penjelasan terhadap variabel-variabel yang diteliti melalui pendefinisian, dan uraian yang lengkap dan mendalam dari berbagai referensi, sehingga ruang lingkup, kedudukan dan prediksi terhadap hubungan antara variabel yang akan diteliti menjadi lebih jelas dan terarah (Sugiyono, 2014:58).

Di bab ini akan dijelaskan secara singkat beberapa teori dasar mengenai kecerdasan buatan atau *Artificial Intelligence (AI)* dan juga bagian-bagiannya seperti logika *fuzzy (fuzzy logic)*, jaringan syaraf tiruan *(artificial neural network)*, dan sistem pakar *(expert system)* dan juga metode-metodenya.

2.1.1 Kecerdasan Buatan atau *Artificial Intelligence* (AI)

Menurut (Budihartono & Suhartono, 2014:2-3) kecerdasan buatan atau *artificial intelligence (AI)* merupakan bidang ilmu komputer yang mempunyai peran penting di era kini dan masa akan datang. Mulai dari yang paling umum hingga yang khusus. Dari *learning* atau *perception* hingga pada permainan catur, pembuktian teori matematika, menulis puisi, mengemudikan mobil, dan melakukan diagnosis penyakit. Pada masa sekarang, perhatian difokuskan pada kemampuan

komputer untuk mengerjakan sesuatu yang dapat dilakukan oleh manusia. Dalam hal ini, komputer tersebut dapat meniru kemampuan kecerdasan dan perilaku manusia.

Biasanya konsep suatu kecerdasan buatan dilakukan dengan mengikuti/mencontoh karakteristik dan analogi berpikir dari kecerdasan manusia dan menerapkannya kedalam bentuk algoritma yang dapat dikenali oleh komputer. Untuk membuat suatu sistem menjadi cerdas yang berarti memiliki pengetahuan, pengalaman dan penalaran untuk membuat suatu keputusan dan kemudian mengambil tindakan maka sistem tersebut harus diberi kemampuan untuk menalar dan bekal pengetahuan. Semakin banyak bekal pengetahuan yang dimiliki oleh seseorang tentu saja diharapkan akan lebih mampu dalam menyelesaikan permasalahan.

2.1.1.1 Logika *Fuzzy (Fuzzy Logic)*

Logika *Fuzzy (Fuzzy Logic)* pertama kali diperkenalkan oleh Jan Lukasiewicz pada tahun 1920-an sebagai teori kemungkinan. Logika kemungkinan ini memperluas jangkauan dari nilai kebenaran untuk semua bilangan riil pada interval antara 0 dan 1. *Fuzzy logic* banyak digunakaan karena mirip dengan cara berpikir manusia. Sistem *fuzzy logic* dapat merepresentasikan pengetahuan manusia dalam bentuk matematis dengan menyerupai cara berpikir manusia (Budihartono & Suhartono, 2014:151).

Berdasarkan penelitian (Buana, 2014:138-139) Logika *fuzzy* merupakan suatu cara untuk memetakan suatu ruang masukan ke dalam suatu ruang keluaran. Dalam teori logika *fuzzy* dikenal himpunan *fuzzy* (*fuzzy set*). Merupakan pengelompokan sesuatu berdasarkan variabel bahasa yang dinyatakan dalam fungsi keanggotaan (*membership function*).

Menurut (Budihartono & Suhartono, 2014:152) Beberapa keuntungan yang dapat diambil dalam menggunakan logika *fuzzy* untuk memecahkan suatu masalah yaitu :

1. Konsep *fuzzy logic* mudah dimengerti
2. *Fuzzy logic* sangat fleksibel
3. *Fuzzy logic* memiliki toleransi terhadap data-data yang tidak tepat.
4. *Fuzzy logic* mampu memodelkan fungsi-fungsi non-linear yang sangat kompleks
5. *Fuzzy logic* dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung tanpa harus melalui pelatihan.
6. *Fuzzy logic* dapat bekerjasama dengan teknik-teknik kendali secara konvesional
7. *Fuzzy logic* didasarkan pada bahasa alami.

Beberapa metode yang digunakan dalam sistem inferensi *fuzzy* adalah (Buana, 2014:139):

1. Metode Tsukamoto

   Setiap konsekuen pada aturan yang berbentuk *if-then* harus direpresentasikan dengan suatu himpunan *fuzzy* dengan fungsi keanggotaan yang monoton.

2. Metode Mamdani

   Sering dikenal dengan nama Metode *Max-Min*.

3. Metode Sugeno

   Penalaran dengan metode sugeno hampir sama dengan penalaran mamdani, hanya saja output (konsekuen) sistem tidak berupa himpunan *fuzzy*.

4. Model Tahani

   Analisis data dilakukan untuk mengolah data yang telah di dapat dan mengelompokan data sesuai dengan kebutuhan perancangan.

2.1.1.2 Jaringan Saraf Tiruan *(Artifical Neural Network)*

Jaringan saraf tiruan merupakan merupakan salah satu upaya manusia untuk memodelkan cara kerja atau fungsi sistem syaraf manusia dalam melaksanakan tugas tertentu. Pemodelan ini didasari oleh kemampuan otak manusia dalam mengorganisasikan sel-sel penyusunnya yang disebut *neuron*, sehingga mampu melaksanakan tugas-tugas tertentu, khususnya pengenalan pola dengan efektifitas yang sangat tinggi (Suyanto, 2014:169-170).

Jaringan syaraf tiruan bisa digunakan untuk menyelesaikan berbagai masalah mulai dari klasifikasi, optimasi, kompresi, peramalan *(forecasting),* sistem kontrol, sistem pendeteksian kecurangan *(intrusion detection systems),* dan sebagainya. Pada umumnya jaringan syaraf tiruan sangat sesuai untuk permasalahan yang bernilai kontinyu, seperti pengenalan tulisan tangan, pengenalan wajah, pengenalan

suara, dan sebagainya. Tetapi kita bisa juga menggunakan jaringan syaraf tiruan unttuk masalah yang bernilai diskrit (Suyanto, 2014:189).

Berdasarkan penelitian (Sudarsono, 2016:62) jaringan syaraf tiruan terdiri dari beberapa *neuron*, dan terdapat hubungan antara *neuron-neuron* tersebut. Pada jaringan syaraf tiruan, *neuron-neuron* akan dikumpulkan dalam lapisan–lapisan yang disebut dengan lapisan *neuron*. Biasanya *neuron* pada satu lapisan akan dihubungkan dengan lapisan sebelum atau sesudahnya terkecuali lapisan masukkan dan lapisan keluaran. Informasi yang diberikan pada jaringan syaraf akan dirambatkan dari lapisan ke lapisan, melalui dari lapisan masukkan sampai lapisan keluaran melalui lapisan tersembunyi.

2.1.1.3 Sistem Pakar *(Expert System)*

Secara umum, sistem pakar ialah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer dan dirancang untuk memodelkan kemampuan menyelesaikan masalah layaknya seorang pakar pada bidang tertentu. Dengan menggunakan sistem pakar, orang awan pun dapat menyelesaikan atau mendapatkan informasi berkualitas yang sebenarnya hanya dapat diperoleh dengan bantuan para pakar dibidangnya.

Pengertian sistem pakar adalah program komputer yang menyimulasi penilaian dan perilaku manusia atau organisasi yang memiliki pengetahuan dan pengalaman ahli dalam bidang tertentu. Biasanya sistem seperti ini berisi basis pengetahuan yang berisi akumulasi pengalaman dan satu set aturan untuk

menerapkan pengetahuan dasar untuk setiap situasi tertentu. Sistem pakar yang canggih dapat ditingkatkan dengan cara menambahkan basis pengetahuan atau set aturan. Diantara banyak pakar yang ada, yang terkenal adalah aplikasi bermain catur dan sistem diagnosis medis (Budihartono & Suhartono, 2014:132).

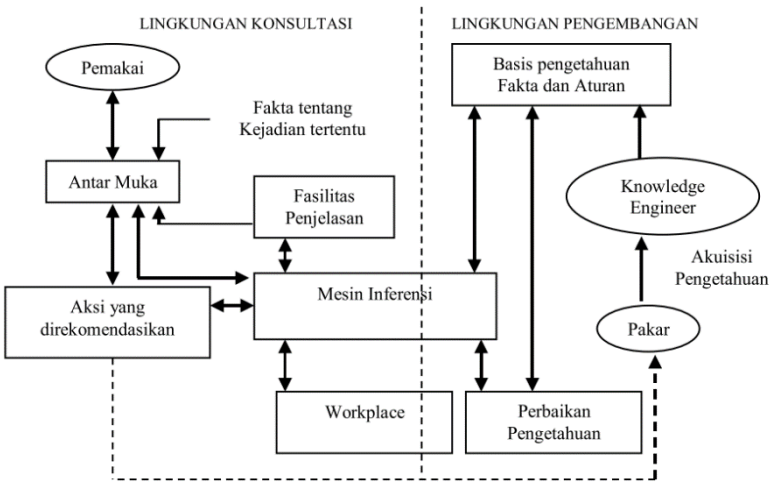
Menurut penelitian (Nurhayati, 2013:110) sistem pakar adalah sistem yang mampu menirukan penalaran seorang pakar agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli. Pengetahuan yang disimpan didalam sistem pakar umumnya diambil dari seorang manusia yang pakar dalam masalah tersebut. Peran penting seorang pakar dapat digantikan oleh program komputer yang pada prinsip kerjanya untuk memberikan solusi yang pasti seperti yang biasa dilakukan oleh pakar. Sistem pakar biasanya digunakan untuk konsultasi, analisis, diagnosis dan membantu mengambil keputusan.

Menurut (Budihartono & Suhartono, 2014:134) sistem pakar banyak digunakan pada aplikasi terkini dan kompleks karena:

1. Sistem pakar dapat bertindak sebagai konsultan, instruktur, atau pasangan/rekan.
2. Meningkatkan *availability* atau kepakaran tersedia pada semua perangkat komputer.
3. Mengurangi bahaya
4. Permanen
5. Pengetahuan dapat tidak lengkap, namun keahlian dapat diperluas sesuai kebutuhan. Program konvesional harus “lengkap” sebelum mereka dapat digunakan.

6. *Database* yang cerdas, sistem pakar dapat digunakan untuk mengakses *database* secara cerdas, misalnya data mining.

Secara umum ada dua bagian penting atau utama yang dibutuhkan dalam membuat aplikasi kecerdasan buatan yaitu basis pengetahuan *(knowlodge base)* dan mesin inferensi *(Inference Engine)*. Istilah sistem pakar berasal dari istilah *knowledge-based expert system*. Istilah ini muncul karena untuk memecahkan masalah, sistem pakar menggunakan pengetahuan seorang pakar yang dimasukkan ke dalam komputer. Seseorang yang bukan pakar menggunakan sistem pakar untuk meningkatkan kemampuan pemecahan masalah, sedangkan seorang pakar menggunakan sistem pakar untuk *knowledge assistant.* Tujuan utama sistem pakar bukan untuk menggantikan kedudukan sorang ahli maupun pakar, tetapi untuk memasyarakatkan pengetahuan dan pengalaman pakar-pakar yang ahli di bidangnya. Sistem pakar disusun oleh dua bagian utama, yaitu lingkungan pengembangan *(development environment)* dan lingkungan konsultasi *(consultation environment)* (Mukhtar & Samsudin, 2014:22).



Gambar 2.1 Struktur Sistem Pakar

Menurut (Hartati & Iswanti, 2008:3-6) sistem pakar sebagai suatu progam yang berfungsi meniru seorang pakar harus mampu melakukan hal-hal yang dapat dilakukan oleh seorang pakar. Menurut (Giarratano dan Riley, 2005) untuk membangun suatu sistem seperti itu maka dibutuhkan komponen-komponen yang harus dimiliki seperti:

1. Antar Muka Pengguna *(User Interface)*

Sistem pakar harus menyediakan pendukung yang diperlukan oleh pemakainya yang tidak mengerti masalah teknis agar sistem pakar dapat dijadikan sebagai pengganti seorang pakar dalam suatu kondisi tertentu. Antar muka ialah media komunikasi antara sistem dan pemakai sistem pakar, antar muka yang efektif dan ramah pengguna (*user-friendly*) merupakan hal yang penting terutama bagi pemakai yang tidak ahli dalam bidang yang diterapkan pada sistem pakar.

2. Basis Pengetahuan *(Knowledge Base)*

Basis pengetahuan adalah komponen yang berisi sekumpulan pengetahuan bidang tertentu pada tingkatan pakar dalam format tertentu. Basis pengetahuan bersifat dinamis yaitu selalu berkembang dari waktu ke waktu dikarenakan pengetahuan dapat bertambah dan *terupdate.* Pemisahan basis pengetahuan dan mesin inferensi dalam sistem pakar bertujuan agar perkembangan pada sistem pakar dapat secara leluasa disesuaikan dengan perkembangan pengetahuan pada suatu domain, dimana penambahan dan pengurangan pada basis pengetahuan dapat dilakukan tanpa menganggu mesin inferensi.

3. Mesin Inferensi *(Inference Machine)*

Mesin inferensi ialah otak dari sistem pakar yang berupa perangkat lunak yang melakukan tugas inferensi pada penalaran sistem pakar atau biasa juga disebut sebagai mesin pemikir (*Thinking Machine*). Pada prinsipnya mesin inferensi ialah mesin yang mencari solusi dari suatu permasalahan, adapun konsep-konsep yang biasa digunakan dalam mesin inferensi seperti runut-balik (*top-down*) yaitu proses penalaran yang berawal dari tujuan yang ingin dicapai lalu menelusuri fakta-fakta yang mendukung untuk mencapai tujuan dan runut muju (*bottom-up*) yaitu proses penalaran yang berawal dari kondisi yang diketahui menuju tujuan yang diinginkan.

4. Memori Kerja *(Working Memory)*

Memori kerja merupakan bagian dari sistem pakar yang bertugas menyimpan fakta-fakta yang diperoleh saat dilakukannya proses konsultasi. Fakta-fakta ini kemudian akan diolah oleh mesin inferensi berdasarkan pengetahuan yang disimpan dalam basis pengetahuan untuk menentukan suatu keputusan dalam pemecahan masalah. Konklusinya dapat berupa hasil diagnosa, tindakan ataupun akibat.

Menurut (Hartati & Iswanti, 2008:22) representasi pengetahuan dalam sistem pakar bertujuan untuk mengorganisasikan pengetahuan ke dalam bentuk dan format tertentu agar dapat dimengerti oleh komputer.

Pemilihan representasi pengetahuan yang tepat dapat membuat sistem pakar yang dibuat lebih efektif dikarenakan pemilihan representasi pengetahuan yang tepat akan membuat sistem pakar dapat mengakses basis pengetahuan dalam

keperluan pembuatan keputusan. Sistem pakar dalam penelitian ini menggunakan model representasi pengetahuan berbasis kaidah produksi *(Production Rule)*.

Menurut (Hartati & Iswanti, 2008:22-26) adapun beberapa model representasi pengetahuan yang penting:

1. Jaringan Semantik (*Semantic Nest*)

Jaringan semantik ialah teknik representasi pengetahuan yang digunakan untuk informasi yang proposional yang dimaksud proporsional disini ialah pernyataan yang mempunyai nilai benar atau salah. Contohnya: sebuah bujur sangkar mempunyai empat sisi. Informasi yang proporsional merupakan bahasa yang deklaratif karena menyatakan fakta.

2. Bingkai (*Frame*)

Teknik representasi bingkai berupa kumpulan slot-slot yang berisi atribut untuk mendeskripsikan pengetahuan, pengetahuan yang dimuat dalam slot dapat berupa kejadian, lokasi, situasi ataupun elemen-elemen lainnya. Representasi pengetahuan bingkai cocok untuk jenis pengetahuan yang memiliki subyek sempit, lebih bersifat pasti dan jarang berubah-ubah isinya kecuali terdapat kondisi khusus.

3. Kaidah Produksi (*Production Rule*)

Kaidah meyediakan cara formal dalam merepresentasikan rekomendasi, arahan atau strategi. Kaidah produksi ditulis dalam bentuk jika-maka (*if-then*). Kaidah jika-maka menghubungkan antesenden *(antecedent)* dengan konsekuensi yang diakibatkannya. Menurut (Adediji, 1992) adapun berbagai struktur kaidah *if-then* yang menhubungkan obyek atau atribut sebagai berikut:

a. *IF* premis *THEN* konklusi
b. *IF* masukan *THEN* keluaran
c. *IF* kondisi *THEN* tindakan
d. *IF* antesenden *THEN* konsekuen
e. *IF* data *THEN* hasil
f. *IF* aksi *THEN* reaksi
g. *IF* sebab *THEN* akibat
h. *IF* gejala *THEN* diagnosa

Premis mengacu pada fakta yang harus benar sebelum konklusi tertentu dapat diperoleh. Masukan mengacu pada data yang harus tersedia sebelum keluaran dapat diperoleh. Kondisi mengacu pada keadaan yang harus berlaku sebelum tindakan dapat diambil. Antesenden mengacu situasi yang terjadi sebelum konsekuensi dapat diamati. Data mengacu pada informasi yang harus tersedia sehingg sebuah hasil dapat diperoleh. Tindakan mengacu pada kegiatan yang harus dilakukan sebelum hasil dapat diharapkan. Aksi mengacu pada kegiatan yang menyebabkan munculnya efek dari tindakan tersebut. Gejala mengacu pada keadaan yang menyebabkan adanya kerusakan atau keadaan tertentu yang mendorong adanya pemeriksaan diagnosa.

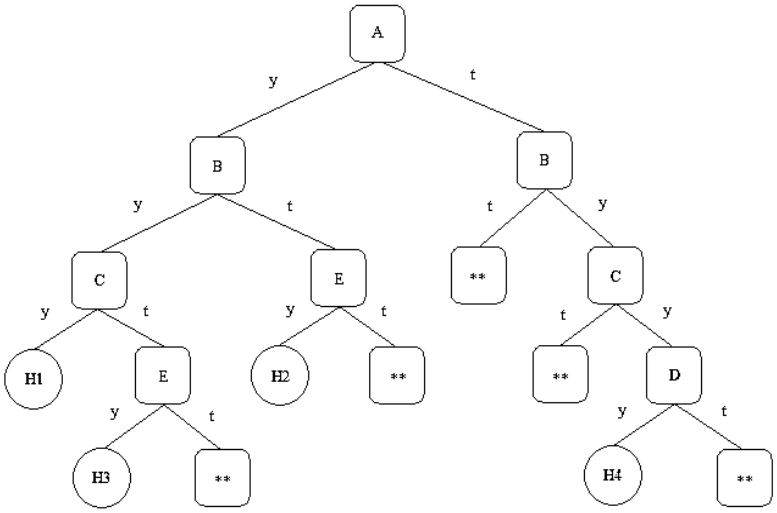
Sebelum sampai pada bentuk kaidah produksi, ada beberapa langkah yang harus ditempuh dari pengetahuan yang didapatkan dalam domain tertentu. Langkah-langkah tersebut ialah menyajikan pengetahuan yang berhasil didapatkan kedalam bentuk tabel keputusan (*decision table*) kemudian dari tabel keputusan dibuat pohon keputusan (*decision tree*).

Tabel 2.1 Tabel Keputusan

Hipotesa / ***Evidence***	**Hipotesa 1**	**Hipotesa 2**	**Hipotesa 3**	**Hipotesa 4**
Evidence A	ya	ya	ya	tidak
Evidence B	ya	tidak	ya	ya
Evidence C	ya	tidak	tidak	ya
Evidence D	tidak	tidak	tidak	ya
Evidence E	tidak	ya	ya	tidak

(Sumber: Hartati & Iswanti, 2008:32)

Mengacu pada tabel keputusan pada tabel 2.1 dapat dihasilkan pohon keputusan sebagai berikut:



Gambar 2.2 Pohon Keputusan

Keterangan:

A = *evidence* A, H1 = hipotesa 1, y = ya

B = *evidence* B, H2 = hipotesa 2, t = tidak

C = *evidence* C, H3 = hipotesa 3, ** = tidak menghasilkan hipotesa tertentu

D = *evidence* D, H4 = hipotesa 4

Dari gambar 2.2 dapat dilihat bahwa hipotesa H1 terpenuhi jika memenuhi *evidence* A, B, dan C. Hipotesa H2 terpenuhi jika memiliki *evidence* A dan *evidence* E. Hipotesa H3 akan terpenuhi jika memiliki *evidence* A, B, dan E. Hipotesa H4 akan dihasilkan jika memenuhi *evidence* B, C, dan D. Notasi "y" mengandung arti memenuhi *node* (*evidence*) di atasnya, notasi "t" artinya tidak memenuhi.

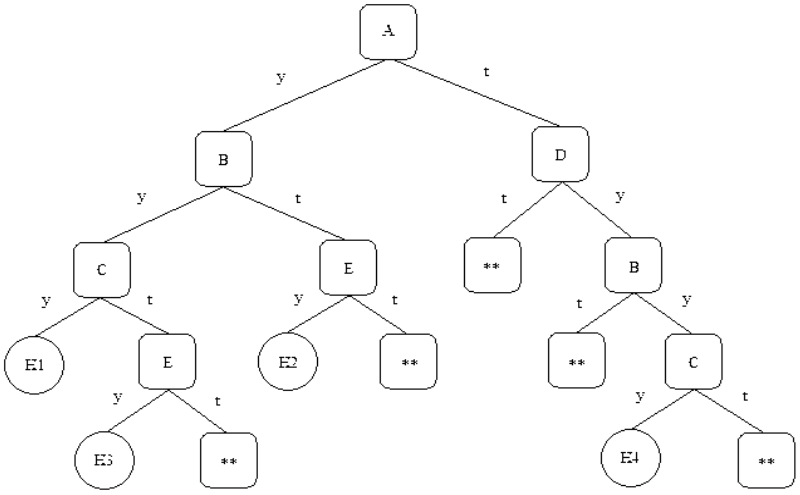
Dalam implementasi sitem pakar terutama dalam sesi konsultasi pada sistem pakar, *node-node* yang mewakili *evidence* biasanya akan menjadi pertanyaan yang diajukan oleh sistem. Dengan melihat pohon keputusan pada gambar 2.2 permasalahan dapat saja terjadi pada awal sesi konsultasi yaitu pada saat sistem pakar menanyakan "apakah memiliki *evidence* A?". Permasalahannya adalah apapun jawaban pengguna baik "ya" atau "tidak" maka sistem akan menanyakan *evidence* B. Ini berarti jawaban pengguna tidak akan mempengaruhi sistem. Salah satu cara untuk mengatasi hal ini adalah dengan mengubah urutan pada tabel keputusan seperti terlihat pada tabel 2.2.

Tabel 2.2 Alternatif Tabel Keputusan

Hipotesa / ***Evidence***	**Hipotesa 1**	**Hipotesa 2**	**Hipotesa 3**	**Hipotesa 4**
Evidence A	ya	ya	ya	tidak
Evidence D	tidak	tidak	tidak	ya
Evidence B	ya	tidak	ya	ya
Evidence C	ya	tidak	tidak	ya
Evidence E	Tidak	Ya	ya	tidak

(Sumber: Hartati & Iswanti, 2008:34)

Mengacu pada tabel keputusan pada tabel 2.2 dapat dihasilkan pohon keputusan sebagai berikut:



Gambar 2.3 Alternatif Pohon Keputusan

Keterangan:

A = *evidence* A,	H1 = hipotesa 1,	y	= ya
B = *evidence* B,	H2 = hipotesa 2,	t	= tidak
C = *evidence* C,	H3 = hipotesa 3,	**	= tidak menghasilkan hipotesa tertentu
D = *evidence* D,	H4 = hipotesa 4		

Dilihat dari gambar 2.3, masing-masing *node* yang mewakili *evidence* tertentu untuk kondisi “y” dan “t” sudah tidak mengarah pada *evidence* yang sama. Hal ini berarti jawaban pengguna yang berbeda akan mengarah pada pertanyaan yang berbeda pula.

Kaidah yang dapat dihasilkan berdasarkan pohon keputusan pada gambar 2.3 adalah sebagai berikut:

a. Kaidah 1:*IF* A *AND* B *AND* C *THEN* H1

b. Kaidah 2:*IF* A *AND* B *AND* E *THEN* H3

c. Kaidah 3:*IF* A *AND* E *THEN* H2

d. Kaidah 4:*IF* D *AND* B *AND* C *THEN* H4

Model representasi pengetahuan kaidah produksi banyak digunakan pada aplikasi sistem pakar karena model representasi ini mudah dipahami dan bersifat deklaratif sesuai dengan jalan pikiran manusia dalam menyelesaikan suatu masalah, dan mudah diinterpretasikan.

4. Logika Predikat *(Predicate Logic)*

Logika predikat berdasarkan pada kebenaran dan kaidah inferensi untuk merepresentasikan simbol-simbol dan hubungan antara satu dengan yang lain. Selain digunakan untuk menentukan kebenaran (*truthfulness*) atau kesalahan (*falsity*) sebuah pernyataan, logika predikat juga dapat digunakan untuk merepresentasikan pernyataan tentang obyek tertentu (Hartati & Iswanti, 2008:40).

Dalam melakukan proses inferensi, sistem pakar memerlukan adanya proses pengujian kaidah-kaidah dalam urutan tertentu untuk mencari suatu kondisi yang sesuai dengan kondisi awal atau untuk memastikan kondisi yang sedang berjalan sudah dimasukkan ke dalam basis data (*database).* Proses pengujian itu disebut dengan perunutan, yaitu proses pencocokan fakta atau kondisi tertentu yang tersimpan dalam basis pengetahuan maupun pada memori kerja dengan kondisi yang dinyatakan dalam premis atau bagian kondisi pada suatu kaidah atau aturan (Hartati & Iswanti, 2008:45).

Adapun beberapa konsep perunutan yang dapat digunakan oleh mesin inferensi yaitu:

a. Runut maju (*forward chaining*)

Konsep ini dapat juga disebut sebagai pencarian yang dimotori data (*data driven search*). Runut maju melakukan proses perunutan (penalaran) dimulai dari premis-premis atau informasi masukan (*IF*) terlebih dahulu kemudian menuju konklusi atau *derived information* (*THEN*). Konsep ini dapat dimodelkan sebagai berikut:

IF (informasi masukan)

THEN (konklusi)

Informasi masukan dapat berupa suatu pengamatan, data, bukti atau temuan. sedangkan konklusi dapat berupa diagnosis, tujuan, hipotesa atau penjelasan sehingga dapat dikatakan jalannya penalaran runut maju dimulai dari pengamatan menuju diagnosis. Pada metode ini, sistem tidak melakukan praduga apapun terhadap konklusi, namun sistem akan menerima semua gejala yang diberikan pengguna lalu sistem akan memeriksa gejala-gejala tersebut dan selanjutnya mencocokkan dengan konklusi yang sesuai (Hartati & Iswanti, 2008:45-46).

b. Runut balik (*backward chaining*)

Secara umum konsep ini diaplikasikan ketika tujuan ditentukan sebagai kondisi atau keadaan awal. Konsep ini disebut juga *goal-driven search*. Arah penalaran atau perunutan dalam konsep ini berlawanan dengan *forward chaining*. Konsep ini dapat dimodelkan sebagai berikut:

Tujuan,

IF (kondisi

Proses penalaran pada *backward chaining* dimulai dari tujuan kemudian merunut balik ke jalur yang mengarah ke tujuan tersebut, untuk membuktikan bahwa bagian kondisi pada kaidah atau aturan benar-benar terpenuhi. Proses *internal* selalu memeriksa konklusi (tujuan) terlebih dahulu sebagai praduga awal, kemudian memeriksa dan memastikan gejala-gejala (kondisi) telah terpenuhi dan selanjutnya mengeluarkan konklusi sebagai *output*. Jika sistem menemukan ada bagian kondisi yang tidak terpenuhi maka sistem akan memeriksa konklusi (tujuan) pada aturan atau kaidah berikutnya (Hartati & Iswanti, 2008:46-47).

2.1.2 Web

Menurut (Trimarsiah & Arafat, 2017:2) web atau *website* merupakan sebuah media informasi yang ada di internet. *Website* tidak hanya dapat digunakan untuk penyebaran infomasi saja melainkan bisa digunakan untuk membuat toko online. *Website* adalah kumpulan dari halaman-halaman situs, yang biasanya terangkum dalam sebuah domain atau subdomain, yang tempatnya berada di dalam *World Wide Web* (WWW) di Internet. Sebuah halaman web adalah dokumen yang ditulis dalam format HTML *(Hyper Text Markup Language),* yang hampir selalu bisa diakses melalui HTTP, yaitu protokol yang menyampaikan informasi dari server *website* untuk ditampilkan kepada para pemakai melalui web browser. Semua publikasi dari *website-website* tersebut dapat membentuk sebuah jaringan informasi yang sangat besar.

2.1.3 *Database* (basis data)

Menurut (A.S & Shalahuddin, 2015:43-44) sistem *database* adalah sistem terkomputerisasi yang tujuan utamanya adalan memelihara data atau informasi yang sudah diolah dan membuat informasi tersedia saat dibutuhkan. *Database* adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Kebutuhan basis data meliputi memasukkan, menyimpan, dan mengambil data serta membuat laporan berdasarkan data yang telah disimpan. Salah satu bentuk basis data yang dibutuhkan dalam sebuah sistem yaitu *Database Management System* (*DBMS*). *DBMS* adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data.

2.1.4 Validasi Sistem

Menurut (A.S & Shalahuddin, 2015:273-276) validasi mengacu pada sekumpulan aktifitas yang berbeda yang menjamin bahwa sistem atau perangkat lunak yang dibangun telah sesuai dengan yang diharapkan. Beberapa pendekatan dalam melakukan pengujian untuk validasi sistem antara lain:

1. *Black-Box Testing* (pengujian kotak hitam)

Pendekatan ini dilakukan dengan menguji sistem atau perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Tujuannya untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari sistem atau perangkat lunak sudah sesuai dengan spesifikasi yang dibutuhkan.

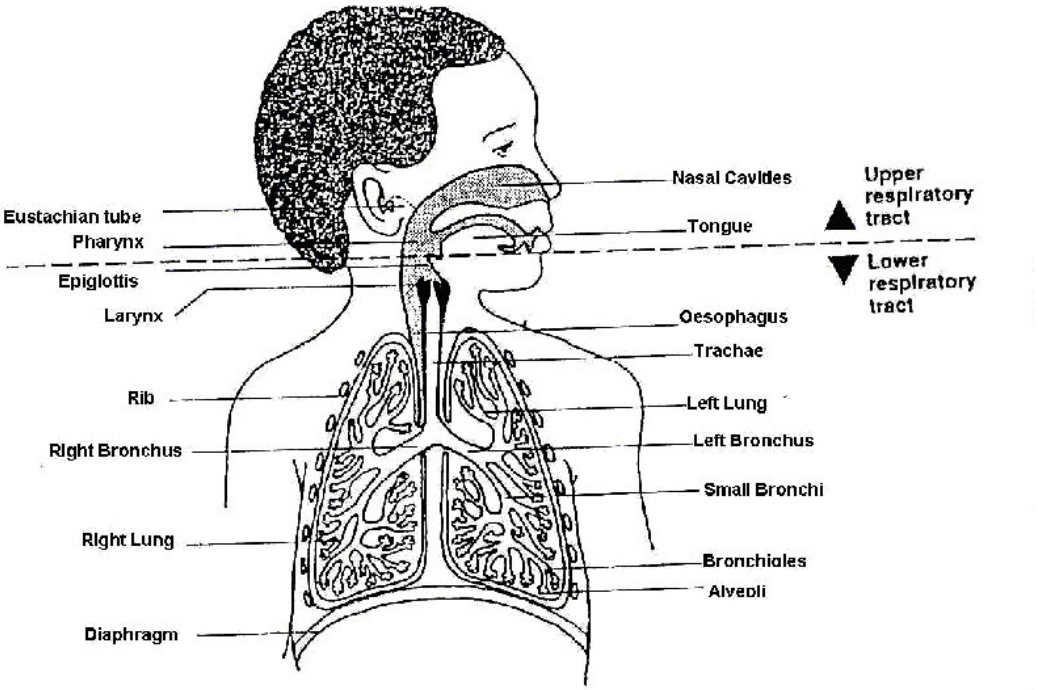
Pengujian dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan menggunakan sistem atau perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan *black-box testing* harus dibuat dengan kasus benar dan kasus salah.

2. *White-Box Testing* (pengujian kotak putih)

Pendekatan ini dilakukan dengan menguji sistem atau perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi yang dibutuhkan. *White-box testing* dilakukan dengan memeriksa logika dari kode program. Pembuatan kasus uji dapat mengikuti standar pengujian dari standar pemrograman yang ada.

2.1.5 Infeksi Saluran Pernafasan Akut

Infeksi saluran pernafaan akut (ISPA) secara anatomis mencakup saluran pernafasan bagian atas, saluran pernafasan bagian bawah (termasuk jaringan paru-paru) dan organ adneksa saluran pernafasan. Dengan batasan ini, jaringan paru termasuk dalam saluran pernafasan *(respiratory tract)*. Penyakit ISPA didiagnosa masyarakat awam berdasarkan ciri-ciri yang diketahuinya, sehingga manyarakat atau penderita sulit membedakan penyakit ISPA dengan jenis penyakit lainya akibatnya penyakit tersebut ditangani dengan cara yang sama (Nurhayati, 2013:110).



Gambar 2.4 Infeksi Saluran Pernafasan Akut

2.2 Variabel Penelitian

Variabel penelitian merupakan segala sesuatu yang berbentuk apa saja yang dimiliki orang, obyek atau kegiatan yang mempunyai variasi tertentu yang ditetapkan oleh peneliti untuk dipelajari sehingga diperoleh informasi tentang hal tersebut, kemudian ditarik kesimpulannya (Sugiyono, 2014:38). Pada penelitian ini yang menjadi variabel penelitian adalah penyakit infeksi saluran pernafasan akut (ISPA) khususnya Faringitis, Laringitis dan Tonsilitis.

2.2.1 Faringitis

Faringitis merupakan peradangan dinding faring yang dapat disebabkan oleh virus (40- 60%), bakteri (5-40%), alergi, trauma, toksin dan lain-lain. Virus dan bakteri melakukan invasi ke faring dan menimbulkan reaksi inflamasi lokal. Infeksi bakteri group A streptokokus ß hemolitikus dapat menyebabkan kerusakan jaringan yang hebat, karena bakteri ini melepaskan toksin ekstraseluler yang dapat menimbulkan demam reumatik, kerusakan katup jantung, glomerulonefritis akut karena fungsi glomerulus terganggu akibat terbentuknya kompleks antigen antibodi (Nadeak, 2017:57).

2.2.2 Laringitis

Laringitis adalah peradangan pada laring yang dapat disebabkan oleh virus, bakteri, atau jamur. Laringitis juga merupakan akibat dari penggunaan suara yang berlebihan, pajanan terhadap polutan eksogen, atau infeksi pada pita suara. Refluks gastroesofageal, bronkitis, dan pneumonia juga dapat menyebabkan laringitis. Laringitis pada anak sering diderita oleh anak usia 3 bulan hingga 3 tahun, dan biasanya disertai inflamasi pada trakea dan bronkus dan disebut sebagai penyakit croup. Penyakit ini seringkali disebabkan oleh virus, yaitu virus parainfluenza, adenovirus, virus influenza A dan B, RSV, dan virus campak (Zainuddin et al., 2016:351).

2.2.3 Tonsilitis

Tonsilitis adalah peradangan tonsil palatina yang merupakan bagian dari cincin Waldeyer. Cincin Waldeyer terdiri atas susunan jaringan limfoid yang terdapat di dalam rongga mulut yaitu: tonsil faringeal (adenoid), tonsil palatina (tonsil faucial), tonsil lingual (tonsil pangkal lidah), tonsil tuba *Eustachius* (*lateral band* dinding faring/ *Gerlach's tonsil*). Penyakit ini banyak diderita oleh anak-anak berusia 3 sampai 10 tahun (Zainuddin et al., 2016:355).

2.3 *Software* Pendukung

2.3.1 XAMPP

XAMPP adalah sebuah software *web server apache* yang didalamnya sudah tersedia *database server MySQL* dan dapat mendukung pemrograman *PHP*. *XAMPP* merupakan *software* yang mudah digunakan, gratis dan mendukung instalasi di *Linux* dan *Windows*. Keuntungan lainnya adalah cuma menginstal satu kali sudah tersedia *Apache Web Server, MySQL Database Server, PHP Support* (*PHP* 4 dan *PHP* 5) dan beberapa module lainnya (Februariyanti & Zuliarso, 2012:129).

2.3.2 phpMyAdmin

phpMyAdmin adalah perangkat lunak bebas yang ditulis dalam bahasa pemrograman *PHP* yang digunakan untuk menangani administrasi *MySQL* melalui Jejaring Jagat Jembar *(World Wide Web)*. phpMyAdmin mendukung berbagai operasi *MySQL*, diantaranya mengelola basis data, tabel- tabel, bidang *(fields)*, relasi *(relations)*, indeks, pengguna *(users)*, perijinan *(permissions)*, dan lain-lain. Pada dasarnya, mengelola basis data dengan *MySQL* harus dilakukan dengan cara mengetikkan baris-baris perintah yang sesuai *(command line)* untuk setiap maksud tertentu. Jika seseorang ingin membuat basis data *(database)*, ketikkan baris perintah yang sesuai untuk membuat basis data.

Hal tersebut tentu saja sangat menyulitkan karena seseorang harus hafal dan mengetikkan perintahnya satu per satu. Saat ini banyak sekali perangkat lunak yang dapat dimanfaatkan untuk mengelola basis data dalam *MySQL*, salah satunya adalah *phpMyAdmin*. Dengan *phpMyAdmin*, seseorang dapat membuat *database*, membuat tabel, mengisi data, dan lain-lain dengan mudah, tanpa harus menghafal baris perintahnya (Barri, Lumenta, & Wowor, 2015:25).

2.3.3 *Hypertext Preprocessor* (PHP)

PHP (Hypertext Preprocessor) merupakan bahasa pemrograman pada sisi *server* yang memperbolehkan *programmer* menyisipkan perintah-perintah perangkat lunak *web server* (*Apache*, *IIS*, atau apapun) akan dieksekusi sebelum

perintah itu dikirim oleh halaman ke *browser* yang me-*request*-nya, contohnya adalah bagaimana memungkinkannya memasukkan tanggal sekarang pada sebuah halaman web setiap kali tampilan tanggal dibutuhkan. Sesuai dengan fungsinya yang berjalan di sisi server maka *PHP* adalah bahasa pemrograman yang digunakan untuk membangun teknologi *web application*. *PHP* telah menjadi bahasa *scripting* untuk keperluan umum yang pada awalnya hanya digunakan untuk pembangunan web yang menghasilkan halaman web dinamis. Untuk tujuan ini, kode *PHP* tertanam ke dalam dokumen sumber *HTML* dan diinterpretasikan oleh *server web* dengan modul *PHP* prosesor, yang menghasilkan dokumen halaman web. Sebagai bahasa pemrograman untuk tujuan umum, kode *PHP* diproses oleh aplikasi penerjemah dalam modus baris-baris perintah modus dan melakukan operasi yang diinginkan sesuai sistem operasi untuk menghasilkan keluaran program di*channel output* standar. Hal ini juga dapat berfungsi sebagai aplikasi grafis. *PHP* tersedia sebagai prosesor untuk *server web* yang paling modern dan sebagai penerjemah mandiri pada sebagian besar sistem operasi dan komputer *platform* (Februariyanti & Zuliarso, 2012:128).

PHP merupakan salah satu bahasa pemrograman web *server-side* yang populer dan banyak digunakan sampai saat ini, *PHP* dirancang oleh Rasmus Ledrof pada tahun 1994. Awalnya *PHP* digunakan untuk mendeteksi *user* yang berkunjung pada situs (Utomo, 2014:1-2).

2.3.4 HTML *(Hyper Text Markup Language)*

HyperText Markup Language (HTML) adalah sebuah bahasa *markup* yang digunakan untuk membuat sebuah halaman web, menampilkan berbagai informasi di dalam sebuah penjelajah web internet dan *formating hypertext* sederhana yang ditulis kedalam berkas format *ASCII* agar dapat menghasilkan tampilan wujud yang terintegerasi. Dengan kata lain, berkas yang dibuat dalam perangkat lunak pengolah kata dan disimpan kedalam format *ASCII normal* sehingga menjadi *homepage* dengan perintah- perintah *HTML*. Bermula dari sebuah bahasa yang sebelumnya banyak digunakan di dunia penerbitan dan percetakan yang disebut dengan *SGML (Standard Generalized Markup Language), HTML* adalah sebuah standar yang digunakan secara luas untuk menampilkan halaman web. *HTML* saat ini merupakan standar internet yang didefinisikan dan dikendalikan penggunaannya oleh *World Wide Web Consortium(W3C). HTML* dibuat oleh kolaborasi Robert Cailliau dan Tim Berners-Lee ketika mereka bekerja di CERN pada tahun 1989 (CERN adalah lembaga penelitian fisika energi tinggi di Jenewa) (Harison & Syarif, 2016:43).

2.3.5 CSS *(Cascading Style Sheet)*

CSS (Cascading Style Sheet) adalah *stylesheet language* yang digunakan untuk mendeskripsikan penyajian dari dokumen yang dibuat dalam *mark up language*. *CSS* merupakan sebuah dokumen yang berguna untuk melakukan pengaturan pada komponen halaman web, inti dari dokumen ini adalah memformat

halaman web standar menjadi bentuk web yang memiliki kualitas yang lebih indah dan menarik (Binarso, Sarwoko, & Bahtiar, 2012:76).

2.3.6 JavaScript

JavaScript adalah bahasa *scripting* kecil, ringan, berorientasi objek yang ditempelkan pada kode *HTML* dan di proses di sisi *client*. *JavaScript* digunakan dalam pembuatan *website* agar lebih interaktif dengan memberikan kemampuan tambahan terhadap *HTML* melalui eksekusi perintah di sisi *browser*. *JavaScript* dapat merespon perintah *user* dengan cepat dan menjadikan halaman web menjadi responsif. *JavaScript* memiliki struktur sederhana, kodenya dapat disisipkan pada dokumen *HTML* atau berdiri sebagai satu kesatuan aplikasi (Yatini, 2014:2).

2.3.7 jQuery

jQuery adalah *JavaScript library* yang dirancang untuk meringkas kode-kode *JavaScript*, sehingga dapat menyederhanakan penulisan skrip program, sesuai dengan slogan "*write less, do more*". *jQuer*y pertama kali dirilis oleh John Resig pada tahun 2006, pada perkembangannya *jQuery* tidak hanya sebagai *framework JavaScript*, namun memiliki kelebihan antara lain:

1. Kemudahan mengakses dan memanipulasi elemen-elemen *HTML*.
2. Memanipulasi *CSS*.
3. Penanganan *event HTML*.

4. Efek-efek *JavaScript* dan animasi.

5. Memodifikasi elemen *HTML DOM*.

Sintak dasar *jQuery* $(selector).action(), tanda $ untuk mendefinisikan *jQuery*, *jQuery selector* digunakan untuk mendapatkan elemen *HTML*, *action* adalah tindakan yang dilakukan *jQuery* pada elemen () (Yatini, 2014:2).

2.3.8 MySQL dan SQL

Menurut (Saputra, 2012:77-78) *MySQL* ialah salah satu *database* kelas dunia yang sangat cocok jika dipadukan dengan bahasa pemrograman *PHP*. *MySQL* bekerja menggunakan bahasa *SQL (Structure Query Language)* yang merupakan bahasa standar yang berfungsi untuk manipulasi *database*. Ada beberapa alasan yang menjadikan *MySQL* sangat diminati oleh para *programmer*, diantaranya:

1. Bersifat *open-source*
2. Menggunakan bahasa *SQL (Structure Query Language)* yang merupakan standar bahasa dalam pengolahan data.
3. *Perfomance* dan *reliable*, pemrosesan *database*-nya sangat cepat dan stabil.
4. Sangat mudah dipelajari *(ease of use).*
5. Memiliki dukungan *(group)* pengguna *MySQL*.
6. Lintas *platform*, dapat digunakan pada berbagai sistem operasi berbeda.
7. *Multiuser*, dimana *MySQL* dapat digunakan oleh banyak *user* dalam waktu yang bersamaan tanpa mengalami konflik.
8. Dan masih banyak lagi.

2.3.9 Sublime Text

Sublime Text Editor adalah *editor* teks yang dapat digunakan untuk berbagai bahasa pemograman termasuk pemograman *PHP*. *Sublime Text Editor* merupakan *editor text* lintas-*platform* dengan *Python application programming interface* (*API*). *Sublime Text Editor* juga mendukung banyak bahasa pemrograman dan bahasa *markup*, dan fungsinya dapat ditambah dengan *plugin*, dan *Sublime Text Editor* tanpa lisensi perangkat lunak, *Text Editor* merupakan suatu kebutuhan wajib yang harus dimiliki dalam menulis suatu pemrograman terutama pemrograman web (Abdullah, 2017:4).

2.3.10 Laravel

Laravel merupakan salah satu *framework PHP* yang dapat digunakan secara gratis. *Laravel* dikembangkan oleh *programmer* asal Amerika yaitu Taylor Otwell pada tahun 2011. Secara sederhana *framework* dapat diartikan sebagai kumpulan kode-kode program yang akan selalu digunakan pada setiap pembuatan aplikasi. Dikarenakan sering atau selalu digunakan, maka kode-kode tersebut dikumpulkan dan disusun secara rapi pada folder-folder agar dapat mudah digunakan dan jadilah sebuah *framework, Laravel* telah menjadi salah satu *framework* terbaik dan paling banyak digunakan oleh *programmer* dunia (Abdullah, 2017:1-2).

2.3.11 Twitter Bootstrap

Twitter Bootstrap adalah *framework* yang kuat menyediakan *set* kelas *CSS* dan fungsi *JavaScript* untuk memudahkan proses pembangunan antarmuka halaman web. Mengaktifkan fitur desain responsif dukungan untuk menampilkan *desktop* maupun *mobile*. Situs dikembangkan dapat bekerja dengan baik pada *desktop* maupun *mobile*. *Developer* tidak harus bekerja dengan *CSS* untuk membuat *website* terlihat menarik atau mendukung prinsip desain *responsive*, kecuali diperlukan.

Twitter Bootstrap dapat diunduh secara gratis di *website* resminya, setelah itu tinggal memanggil *file CSS Twitter Bootstrap* pada *file project website* yang akan menggunakan *Twitter Bootstrap*. Begitu selesai memanggil *Bootstrap*, maka secara otomatis akan mengubah tampilan *website* tanpa harus melakukan pengetikan sintak-sintak *CSS* seperti biasa dilakukan (Rosid & Jakaria, 2016:130).

2.3.12 StarUML

StarUML merupakan suatu *CASE (Computer-Aided Software Engineering) tools* atau perangkat pembantu berbasis komputer untuk rekayasa perangkat lunak yang membantu dalam pembuatan suatu sistem perangkat lunak. *StarUML* termasuk salah satu *upper CASE tools* yang mendukung perencanaan strategis dan pembangunan perangkat lunak seperti membuat diagram ER (*Entity Relationship*),

DFD (*Data Flow Diagram*), dan diagram perencanaan atau desain perangkat lunak lainnya (A.S & Shalahuddin, 2015:122-123).

UML (*Unified Modeling Language*) adalah suatu standar bahasa yang banyak digunakan untuk mendefinisikan kebutuhan atau *requirement,* membuat analis dan desain, serta mengambarkan arsitektur dalam pemrograman berorientasi objek (OOP) dalam industri perangkat lunak (A.S & Shalahuddin, 2015:133).

Menurut (A.S & Shalahuddin, 2015:140-141) dalam *UML 2.3* terdapat 13 macam diagram yang dibagi menjadi 3 kelompok yaitu:

1. *Structure Diagrams*

   Dalam *Structure Diagrams* terdiri dari kumpulan diagram yang digunakan untuk mengambarkan suatu struktur statis dari sistem yang dimodelkan, Diagram yang termasuk dalam *Structure Diagrams* antara lain *class diagram, objet diagram, componet diagram, composite structure diagram, package diagram* dan *deployment diagram.*

2. *Behavior Diagrams*

   Dalam *Behavior Diagrams* terdiri dari kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sitem, yang termasuk dalam *Behavior Diagrams* antara lain *use case diagram, activity diagram* dan *state machine diagram.*

3. *Interaction Diagrams*

   Dalam *Interaction Diagrams* terdiri dari kumpulan diagram yang digunakan untuk menggambarkan interaksi sitem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem, yang termasuk dalam *Interaction Diagrams*

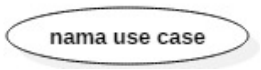
antara lain *sequence diagram, communication diagram, timing diagram* dan *interaction overview diagram.*

Menurut (A.S & Shalahuddin, 2015:18) *use case, class diagram* dan *sequence diagram* merupakan bagian dari desain sistem. Dalam penelitian ini, diagram yang akan digunakan untuk desain sistem yaitu:

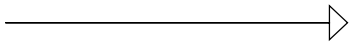
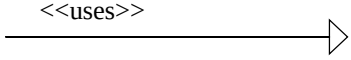
1. *Use case diagram*

Use case diagram merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case diagram* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Ada 2 hal utama yang terdapat pada *use case* yaitu aktor dan *use case*. Berikut ini adalah simbol-simbol yang digunakan dalam *use case diagram* (A.S & Shalahuddin, 2015:155-156).

Tabel 2.3 Simbol *Use Case Diagram*

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 di luar sistem informasi yang akan dibuat itu sendiri. Aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama actor

Tabel 2.3 Lanjutan

Asosiasi/association	Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan actor
Ekstensi/extend <<extend>> 	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walaupun tanpa use case tambahan itu. Arah panah mengarah pada use case yang ditambahkan.
Generalisasi/generalization 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara 2 buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari fungsi lainnya. Arah panah mengarah pada use case yang menjadi generalisasinya (umum)
Menggunakan/include/uses <<include>>  <<uses>> 	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya use case ini. Arah panah mengarah pada use case yang ditambahkan

(Sumber: A.S & Shalahuddin, 2015:156-158)

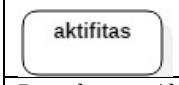
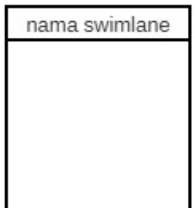
2. *Activity Diagram*

Menurut (A.S & Shalahuddin, 2015:161-163) *Activity Diagram* ialah diagram aktivitas yang mengambarkan aktivitas sistem bukan apa yang dilakukan oleh aktor, jadi aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Dibawah ini akan menjelaskan simbol-simbol *Activity* Diagram:

Tabel 2.4 Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktifitas memiliki sebuah status awal

Tabel 2.4 Lanjutan

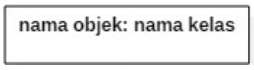
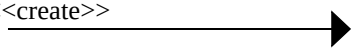
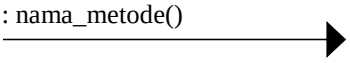
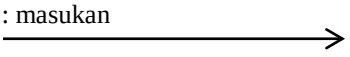
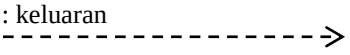
Aktifitas 	Aktifitas yang dilakukan sistem, aktifitas biasanya diawali dengan kata kerja
Percabangan/decision 	Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu
Penggabungan/join 	Asosiasi penggabungan dimana lebih dari satu aktifitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktifitas memiliki sebuah status akhir
Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi

(Sumber: A.S & Shalahuddin, 2015:162-163)

3. *Sequence Diagram*

Menurut (A.S & Shalahuddin, 2015:165:167) diagram sekuen mengambarakan kelakuaan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang akan dikirimkan dan diterima antar objek. Oleh karena itu objek-objek yang terlibat dalam *use case* beserta metode-metode yang dimiliki setiap kelas yang diinstansiasi menjadi objek harus diketahui untuk menggambar diagram sekuen. Dibawah ini akan menjelaskan simbol-simbol *Sequence Diagram* yaitu:

Tabel 2.5 Simbol *Sequence Diagram*

Simbol	Deskripsi
Aktor/*actor*  	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor
Garis hidup/*lifeline* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor
Objek  	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. Aktor tidak memiliki waktu aktif
Pesan tipe *create* <<create>> 	Menyatakan suatu objek membuat objek yang lain. Arah panah mengarah pada objek yang dibuat
pesan tipe call 1 : nama_metode() 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri. Arah panah mengarah pada objek yang memiliki operasi/metode.
Pesan tipe send 1 : masukan 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya. Arah panah mengarah pada objek yang dituju
pesan tipe return 1 : keluaran 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu. Arah panah mengarah pada objek penerima
Pesan tipe destroy  	Menyatakan suatu objek mengakhiri hidup objek lain. Arah panah mengarah pada objek yang diakhiri

(Sumber: A.S & Shalahuddin, 2015:165-167)

4. *Class Diagram*

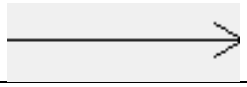
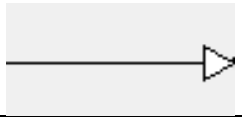
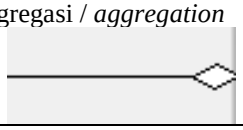
Menurut (A.S & Shalahuddin, 2015:141-148) *Class diagram* digunakan untuk melakukan visualisasi struktur kelas-kelas yang akan dibuat untuk memabangun suatu sistem *Class diagram* dibuat bertujuan agar pembuat program membuat kelas-kelas sesuai dengan rancangan dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Banyak berbagai kasus, perancangan kelas yang dibuat pada perangkat lunak, sehingga tidaklah berguna lagi sebuah perancangan karena apa yang dirancang dan hasil jadinya tidak sesuai atau sama. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

a. Kelas *main*
b. Kelas yang menangani tampilan sistem *(view)*
c. Kelas yang diambil dari pendefinisian *use-case (controller)*
d. Kelas yang diambil dari pendefinisian data *(model)*

Tabel 2.6 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas nama_kelas +atribut +operasi()	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Antarmuka / *interface* nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association*	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*

Tabel 2.6 Lanjutan

Asosiasi berarah / directed association 	Relasi antarakelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
generalisasi 	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus)
Kebergantungan / *dependency* 	Relasi antarkelas dengan makna kebergantungan antarkelas
Agregasi / *aggregation* 	Relasi antarkelas dengan makna semua-bagian (*whole-part*)

(Sumber: A.S & Shalahuddin, 2015:146-147)

2.4 Penelitian Terdahulu

Berikut peneliti mencatumkan beberapa penelitian terdahulu dibidang sistem pakar untuk mendukung teori yang berkaitan dengan penelitian.

Penelitian (Nurlaela, 2013:76) dengan judul "Sistem Pakar Untuk Mendiagnosa Penyakit Gigi Pada Manusia". Sistem pakar merupakan suatu sistem yang dirancang untuk membantu dalam mendeteksi penyakit dengan basis pengetahuan yang dinamis. Pengetahuan ini didapat dari pakar yaitu dokter gigi. Dalam sistem pakar ini menggunakan metode inferensi *forward chaining*. Dalam penelitian ini menggunakan metode wawancara dengan dokter gigi. Adapun untuk tujuan penelitian adalah menghasilkan suatu sistem pakar untuk membantu dokter gigi dalam mendokumentasikan ilmunya. Dan untuk manfaatnya dapat menerapkan ilmu yang telah didapatkan di perkuliahan untuk membantu

memberikan pelayanan kepada masyarakat. Hasil yang diperoleh dari penelitian ini adalah sistem pakar untuk mendeteksi penyakit gigi pada manusia dengan memanfaatkan komputer sebagai alat bantu untuk mengakses data.

Penelitian (Lisnawita, Lhaura Van & Lindra, 2016:95) dengan judul "Sistem Pakar Diagnosa Penyakit THT". Dalam mendiagnosa penyakit THT (Telinga, Hidung, tenggorokan), seorang dokter memerlukan data berupa gejala-gejala yang sedang dialami oleh si penderita. Gejala-gejala ini dapat di peroleh dari pemeriksaan fisik atau laboratorium. Sebagai seorang manusia yang memiliki keterbatasan, begitu juga dengan dokter THT tentunya memiliki kelemahan. Apabila terdapat penyakit THT baru atau lupa tentang jenis penyakit dan pengobatannya, maka seorang ahli THT mencari kembali buku-buku atau dokumen-dokumen yang membahas tentang penyakit THT tersebut. Cara ini tentu saja akan memakan waktu yang lama. Penelitian ini bertujuan untuk merancang sistem pakar untuk mendiagnosa penyakit THT. Perancangan ini di mulai dari pembuatan basis pengetahuan dilanjutkan dengan identifikasi *rule* dan perancangan *input-output* dengan metode *forward chaining*. Hasil rancangan pada penelitian ini diharapkan mampu membantu pekerjaan dokter menjadi lebih mudah sehingga memberikan hasil diagnosa penyakit THT menggunakan sistem pakar.

Penelitian (Syamsuddin & Ahyuna, 2014:64) dengan judul "Sistem Pakar untuk Diagnosa Penyakit yang Disebabkan oleh Nyamuk Berbasis Web". 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.

Penelitian (Pasalli, Poekoel, & Najoan, 2016:1) dengan judul "Sistem Pakar Diagnosa Penyakit Anak Menggunakan Metode *Forward Chaining* Berbasis *Mobile*". Tujuan Penelitian ini adalah membuat aplikasi sistem pakar berbasis *mobile* dalam mendiagnosis penyakit anak. Dengan menggunakan teknik penalaran *forward chaining* diagnosa dilakukan dengan memulai dari sekumpulan gejala-gejala, nantinya dapat melihat kesimpulan jenis penyakit pada anak. Metode yang digunakan sebagai tahapan penelitian ini adalah metode *Extreme Programming* (*XP*) yang merupakan metode rancang bangun perangkat lunak yang menekankan pada 4 tahap dalam pengembangan perangkat lunak.

Penelitian (Sinaga, 2014:101) dengan judul "Sistem Pakar Mendeteksi Penyakit Tanaman Terong Belanda dengan Menggunakan Metode *Forward Chaining*". Sistem pakar ini menggunakan mesin inferensi dengan metode *forward chaining*. Penalaran dilakukan berdasarkan dari gejala-gejala yang tampak secara fisik terhadap tanaman Terong Belanda. Berdasarkan gejala-gejala tersebut dibuatlah *rule-rule* yang kemudian menjadi knowledge base yang akan diterapkan ke dalam mesin inferensi untuk mengetahui penyakit apa yang dialami oleh tanaman Terong Belanda tersebut. Hasil program ini menunjukkan bahwa sistem pakar dapat digunakan sebagai suatu media yang dapat memberikan

informasi tentang tanaman Terong Belanda. Sistem pakar ini dapat digunakan untuk mempercepat pencarian dan pengaksesan terhadap pengetahuan oleh orang-orang yang membutuhkan informasi.

Penelitian (Ashari & Muniar, 2016:1) dengan judul "Penerapan Sistem Pakar Untuk Mendiagnosa Penyakit Pecernaan Dengan Pengobatan Alami". Penerapan sistem pakar untuk mendiagnosa penyakit pencernaan dengan pengobatan dari bahan alami menggunakan metode *forward chaining*. Metode *forward chaining* merupakan metode inferensi untuk penalaran dari suatu masalah dengan memberikan solusinya. Penelitian ini sebagai produk teknologi terapan yang diharapkan memberi manfaat sebagai media konsultasi atau instruktur bagi masyarakat pada umumnya, dan terkhusus bagi dokter dan paramedis pada klinik, puskesmas dan rumah sakit dalam memberikan alteratif pencegahan dan pengobatan secara alami. Perancangan sistem telah dilakukan melalui aktivitas pengumpulan data, perancangan rules, perancangan proses dan pengujian sistem. Tahun pertama menghasilkan produk aplikasi sistem pakar yang telah diimplementasikan pada sejumlah puskesmas dan pada tahun kedua dilakukan pengujian model integrasi sistem pakar terhadap berbagai gejala penyakit pencernaan yang terjadi dengan memberikan alternatif solusi pencegahan penyakit berdasarkan hasil inference yang ditemukan dengan pengobatan cara alami. Hasil pengujian sistem dinyatakan baik dengan tingkat akurasi 91,56%.

Penelitian (Supartini & Hindarto, 2016:147) dengan judul "Sistem Pakar Berbasis Web Dengan Metode *Forward Chaining* Dalam Mendiagnosis Dini Penyakit Tuberkulosis di JawaTimur". Tuberkulosis adalah suatu penyakit menular

berbahaya yang disebabkan oleh kelompok *Mycobacterium*, yaitu *Microbacterium* Tuberkulosis. Setiap pasien Tuberkulosis dapat menularkan penyakitnya pada orang lain yang berada di sekelilingnya dan atau yang berhubungan erat dengannya. Karena masih banyak orang yang tidak mengetahui gejala-gejala penyakit suatu sistem pakar mendiagnosis secara dini penyakit tuberkulosis menggunakan metode forward chaining berbasis web, dapat dikenali dengan melihat gejala-gejala dengan mendeteksi penyakit sejak dini, dilakukan pencegahan terhadap penyakit tuberkulosis. Diagnosis sistem pakar, memiliki nilai keakuratan 93,333 % dan nilai eror 6,667 % untuk uji coba pada 15 pasien. Sehingga dapat disimpulkan bahwa sistem pakar cukup layak untuk digunakan oleh pasien dalam mendiagnosis dini pada penyakit tuberkulosis.

Penelitian (Ali & Saudi, 2014:1) dengan judul *"An expert system for the diagnosis and management of ural ulcers"*. Kondisi-kondisi oral ulcer seperti *aphthous ulcers, chancre, traumatic ulcers, histoplasmosis ulcers, acute herpetic ulcers, burns, herpangina, tuberculosis, syphilis, gonorrhea, acute necrotizing ulcerative gingivitis, herpes labialis, herpes zoster, erythema multiform, Steven Johnson's syndrome.* Kondisinya juga termasuk alergi, sindrom *behcet*, neutropenia, sialometaplasia necrotizing, pemfigus, ulseratif lichen planus, defisiensi vitamin, anemia, lupus eritematosis, *epidermolisis bulosa, pemphigoid, karsinoma sel squaemous, gingivitis deskuamatif* dan lain-lain. Hasil menunjukkan bahwa rata-rata kualitas penilaian keseluruhan program adalah 3,0 ± 0,7 dengan tingkat keberhasilan 75%. Dapat disimpulkan bahwa sistem pakar yang

diperkenalkan sangat membantu bagi mahasiswa pascasarjana dalam diagnosis dan pengobatan ulkus oral yang paling umum.

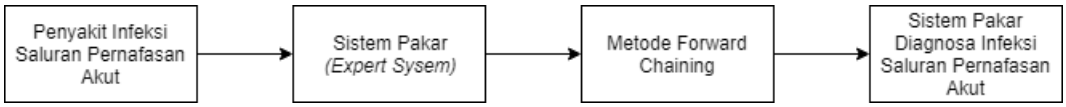
Penelitian (Singla, Grover, & Bhandari Asst, 2014:36) dengan judul *"Medical Expert Systems for Diagnosis of Various Diseases"*. Penyakit seharusnya dirawat dengan baik dan tepat waktu. Jika mereka tidak diobati tepat waktu, mereka dapat menyebabkan banyak masalah kesehatan dan masalah ini dapat menjadi penyebab kematian. Masalah-masalah ini menjadi lebih buruk karena kelangkaan spesialis, praktisi, dan fasilitas kesehatan. Dalam upaya untuk mengatasi masalah tersebut, penelitian ini dibuat sebagai upaya untuk merancang dan mengembangkan sistem pakar yang dapat memberikan saran bagi dokter dan pasien untuk memfasilitasi diagnosis dan merekomendasikan pengobatan pasien. Ulasan makalah ini menyajikan studi komprehensif sistem pakar medis untuk diagnosis berbagai penyakit. Ini memberikan gambaran singkat tentang sistem pakar diagnostik medis dan menyajikan analisis studi yang sudah ada.

2.5 Kerangka Pemikiran

Munurut (Sugiyono, 2014:60) kerangka pemikiran ini merupakan penjelasan sementara terhadap gejala-gejala yang menjadi obyek permasalahan dan juga merupakan sintesa tentang hubungan antar variabel yang disusun dari berbagai teori yang telah dideskripsikan.

Permasalahan seperti kurangnya penyebaran pengetahuan masyarakat mengenai infeksi saluran pernafasan akut dan masyarakat tidak dapat menanganin

penyakit infeksi saluran pernafasan akut dengan cepat yang berujung menyebabkan kematian merupakan permasalahan yang teridentifikasi dalam penelitian ini. Untuk menjawab permasalah tersebut berikut kerangka pemikiran pada penilitian ini:



Gambar 2.5 Kerangka Pemikiran

Data-data yang dibutuhkan berkaitan dengan penyakit infeksi saluran pernafasan akut dianalisis terlebih dahulu agar lebih sederhana atau mudah dilakukan proses pengolahan datanya. Data-datanya tersebut kemudian diolah menggunakan sistem pakar menggunakan metode *forward chaining*. Sistem pakar yang menggunakan metode *forward chaining* ini menggunakan bahasa pemograman *PHP* dan *database MySQL* yang dapat digunakan untuk mendiagnosa penyakit infeksi saluran pernafasan akut dan menghasilkan *output* (hasil diagnosa).