BAB II

TINJAUAN PUSTAKA

II.1 Sistem Pakar

Sistem pakar (*expert system*) adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli. Sistem pakar dirancang agar dapat menyelesaikan masalah tertentu dengan meniru kerja dari para ahli. Dengan sistem pakar ini, orang awam pun dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli.

Sistem pakar adalah salah satu cabang dari AI yang membuat pengguna secara luas *knowledge* yang khusus untuk penyelesaian masalah tingkat manusia yang pakar. Seorang pakar adalah orang yang mempunyai keahlian dalam bidang tertentu, yaitu pakar yang mempunyai *knowledge* atau kemampuan khusus yang orang lain tidak mengetahui atau mampu dalam bidang yang dimilikinya. Ketika sistem pakar dikembangkan pertama kali sekitar tahun 70-an, sistem pakar hanya berisi *knowledge* yang ekslusif. Namun demikian sekarang ini istilah sistem pakar sudah digunakan untuk berbagai macam sistem yang menggunakan teknologi sistem pakar itu.

Walaupun tujuan umum penyelesaian masalah masih jauh dari apa yang diharapkan namun sistem pakar berfungsi sangat baik dalam batasan domainnya. Hal ini dapat dibuktikan bahwa sistem pakar telah banyak diaplikasikan dalam

berbagai bidang yang *trend* saat ini, seperti bisnis, kedokteran, ilmu pengetahuan, dan teknik. (Jurnal SAINTIKOM Vol.10 No.3 September 2011).

II.1.1 Keuntungan Sistem Pakar

Sistem pakar mampu memecahkan masalah tanpa dipengaruhi oleh faktor dari luar seperti intimidasi, paksaan kejiwaan, faktor ekonomi ataupun perasaan. Berikut ini beberapa keuntungan dari sistem pakar :

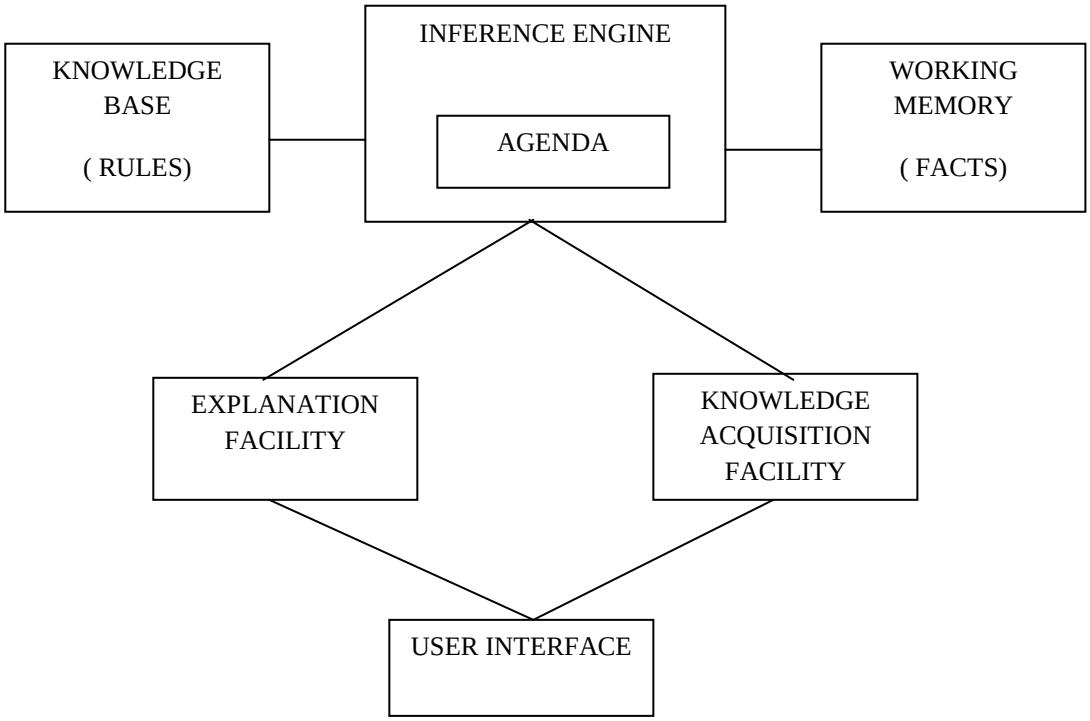
1. Memungkinkan orang awam bisa mengerjakan pekerjaan para ahli.
2. Bisa melakukan proses secara berulang secara otomatis.
3. Menyimpan pengetahuan dan keahlian para pakar.
4. Mampu mengambil dan melestarikan keahlian para pakar (terutama yang termasuk keahlian langka).
5. Mampu beroperasi dilingkungan yang berbahaya.
6. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian.
7. Tidak memerlukan biaya saat tidak digunakan.
8. Dapat digandakan (diperbanyak) sesuai kebutuhan dengan waktu yang minimal dan sedikit biaya.
9. Menghemat biaya dalam pengambilan keputusan.
10. Meningkatkan kualitas dan produktivitas.

Selain keuntungan-keuntungan yang telah disebutkan di atas, sistem pakar juga memiliki kelemahan diantaranya adalah :

1. Biaya yang diperlukan untuk membuat dan memeliharanya sangat mahal.
2. Sulit dikembangkan. Hal ini tentu saja erat kaitannya dengan ketersediaan pakar di bidangnya.
3. Sistem pakar tidak 100% bernilai benar.

II.1.2 Struktur Sistem Pakar

Adapun struktur sistem pakar dapat dilhat pada gambar II.1



Gambar II.1 Struktur Sistem Pakar

Sumber : (Rika Rosnelly, 2012:13)

Komponen yang terdapat dalam struktur sistem pakar ini adalah *knowledge base (rules), inference engine, working memory, explanation facility, knowledge acquisition facility, user interface.*

1. *Knowledge Base* (Basis Pengetahuan)

Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi dan penyelesaian masalah. Komponen sistem pakar disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang objek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

2. *Inference Engine* (Mesin Inferensi)

Mesin inferensi merupakan otak dari sebuah sistem pakar dan dikenal juga dengan sebuat *control structure* (struktur kontrol) atau *rule interpreter* (dalam sistem pakar berbasis kaidah). Mesin inferensi disini adalah *processor* pada sistem pakar yang mencocokkan bagian kondisi dari *rule* yang tersimpan di dalam *knowledge base* dengan fakta yang tersimpan di *working memory*.

3. Working Memory

Berguna untuk menyimpan fakta yang dihasilkan oleh *inference engine* dengan penambahan parameter berupa derajat kepercayaan atau dapat juga dikatakan sebagai *global database* dari fakta yang digunakan oleh *rule* yang ada.

4. Explanation Facility

Menyediakan kebenaran dari solusi yang dihasilkan kepada *user* (*reasoning chain*)

5. *Knowledge Acquisition Facility*

Meliputi proses pengumpulan, pemindahan dan perubahan dari kemampuan pemecahan masalah seorang pakar atau sumber pengetahuan terdokumentasi ke *program* komputer, yang bertujuan untuk memperbaiki atau mengembangkan basis pengetahuan.

6. *User Interface*

Mekanisme untuk memberikan kesempatan kepada *user* dan sistem pakar untuk berkomunikasi. Antar muka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai. (Rika Rosnelly, 2012:14-15).

II.1.3 Konsep Dasar Sistem Pakar

Konsep dasar sistem pakar mengandung keahlian, ahli/pakar, pengalihan keahlian, mengambil keputusan, aturan, kemampuan menjelaskan.

1. Keahlian

Keahlian adalah suatu kelebihan penguasaan pengetahuan dibidang tertentu yang diperoleh dari pelatihan, membaca, atau dari pengalaman.

Bentuk pengetahuan yang termasuk keahlian :

a. Fakta-fakta pada lingkup permasalahan tertentu.
b. Teori-teori pada lingkup permasalahan tertentu.
c. Aturan-aturan berkenaan dengan lingkup permasalahan tertentu.
d. *Meta-knowledge* (pengetahuan tentang pengetahuan).

2. Ahli/Pakar

Seorang ahli adalah seorang yang mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan, menyusun kembali pengetahuan jika dipandang perlu, memecahkan masalah dengan cepat dan tepat.

3. Pengalihan Keahlian

Tujuan dari sistem pakar adalah untuk men*transfer* keahlian dari seorang pakar ke dalam komputer kemudian ke masyarakat. Proses ini meliputi 4 kegiatan, yaitu perolehan pengetahuan (dari para ahli atau sumber-sumber lainnya), representasi keahlian ke komputer, kesimpulan dari pengetahuan dan pengalihan pengetahuan ke pengguna.

4. Mengambil Keputusan

Hal yang unik dari sistem pakar adalah kemampuan untuk menjelaskan dimana keahlian tersimpan dalam basis pengetahuan. Kemampuan komputer untuk mengambil komponen yang dikenal dengan mesin inferensi yaitu meliputi prosedur tentang pemecahan masalah.

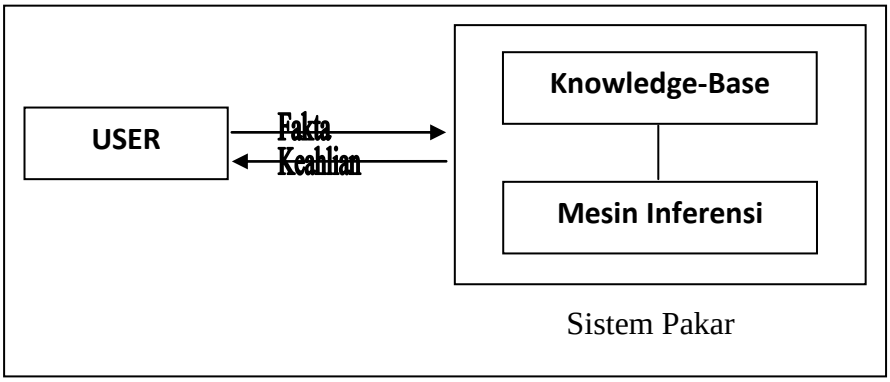
5. Aturan (*Rule)*

Sistem pakar yang dibuat merupakan sistem yang berdsarkan pada aturan-aturan dimana *program* tersimpan dalam bentuk aturan-aturan sebagai prosedur pemecahan masalah. Aturan tersebut biasa berbentuk IF – THEN.

6. Kemampuan Menjelaskan

Keunikan lain dari sistem pakar adalah kemampuan dalam menjelaskan atau memberi saran/rekomendasi serta juga menjelaskan mengapa beberapa tindakan/saran tidak direkomendasikan.

Gambar II.2 menggambarkan konsep dasar suatu sistem pakar *knowledge-base*. Pengguna menyampaikan fakta atau informasi untuk sistem pakar dan kemudian menerima saran dari pakar atau jawaban ahlinya. Bagian dalam sistem pakar terdiri dari dua komponen utama, yaitu *knowledge base* yang berisi *knowledge* dan mesin inferensi yang menggambarkan kesimpulan. Kesimpulan tersebut merupakan respon dari sistem pakar atas permintaan pengguna.



Gambar II.2 Konsep Dasar Fungsi Sistem Pakar

Sumber : (Muhammad Dahria, 2011:200-201).

II.1.4 Tujuan Sistem Pakar

Pengalihan keahlian dari para ahli ke komputer untuk kemudian dialihkan ke orang lain yang bukan ahli. Seorang ahli adalah seorang yang mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan, menyusun kembali pengetahuan jika dipandang perlu, memecahkan aturan-aturan jika dibutuhkan dan menentukan relevan tidaknya keahlian mereka.

Proses pengalihan pengetahuan membutuhkan 4 aktivitas, yaitu:

1. Tambahkan pengetahuan (dari para ahli atau sumber-sumber lainnya).
2. Representasikan pengetahuan (ke komputer).
3. Lakukan inferensi pengetahuan.
4. Pengalihan pengetahuan ke *user*.

Pengetahuan yang disimpan ke dalam komputer disebut basis pengetahuan. Ada 2 tipe pengetahuan yaitu : fakta dan prosedur (biasanya berupa aturan).

II.1.5 Basis Pengetahuan Pakar

Basis pengetahuan berisi pengetahuan-pengetahuan dalam penyelesaian masalah. Ada 2 bentuk pendekatan basis pengetahuan.

1. Penalaran berbasis aturan (*rule-based reasoning)*

Pada penalaran berbasis aturan, pengetahuan berbentuk IF-THEN. Bentuk ini digunakan apabila kita memiliki sejumlah pengetahuan pakar pada suatu permasalahan tertentu dan si pakar dapat menyelesaikan masalah tersebut secara berurutan. Disamping itu, bentuk ini juga digunakan apabila dibutuhkan penjelasan tentang jejak (langkah-langkah) pencapaian solusi.

Contoh : aturan identifikasi hewan

Rule 1 : IF hewan berambut dan menyusui THEN hewan mamalia

Rule 2 : hewan mempunyai sayap dan bertelur THEN hewan jenis burung

Rule 3 : hewan mamalia dan memakan daging THEN hewan karnovora

Dan seterusnya…….

2. Penalaran berbasis kasus (*case-based reasoning*)

Pada penalaran berbasis kasus, basis pengetahuan akan berisi solusi-solusi yang telah dicapai sebelumnya, kemudian akan diturunkan suatu solusi untuk keadaan yang terjadi sekarang (fakta yang ada). Bentuk ini digunakan apabila *user* menginginkan untuk tahu lebih banyak lagi pada kasus-kasus yang hampir sama (mirip). Selain itu bentuk ini juga digunakan bila kita telah memiliki sejumlah situasi atau kasus tertentu dalam basis pengetahuan. (Jurnal SAINTIKOM Vol.10 No.3 September 2011).

II.1.6 Motor Inferensi (*Inference Engine*)

Mesin inferensi (*inference engine*) merupakan bagian yang bertindak sebagai pencari solusi dari suatu permasalahan berdsarkan pada kaidah-kaidah yang ada dalam basis pengetahuan sistem pakar. Selama proses inferensi, mesin inferensi memerikasa status dari basis pengetahuan dan memori kerja (*working memory*) untuk menentukan fakta apa saja yang diketahui dan untuk menambah fakta baru ke dalam memori kerja tersebut. Fakta-fakta yang merupakan hasil dari proses inferensi disimpan dalam memori kerja.

Ada dua strategi pencarian dasar yang bisa digunakan oleh mesin inferensi dalam mencari kesimpulan untuk mendapatkan solusi bagi permasalahan yang dihadapi sistem pakar, yaitu runut maju (*forward chaining*) dan runut balik (*backward chaining).*

1. ***Forward Chaining*** **(Runut Maju)**

Runut maju merupakan strategi pencarian yang memulai proses pencarian dari sekumpulan data atau fakta, dari data-data tersebut dicari suatu kesimpulan yang menjadi solusi dari permasalahan yang dihadapi. Mesin inferensi mencari kaidah-kaidah dalam basis pengetahuan yang premisnya sesuai dengan data-data tersebut, kemudian dari kaidah-kaidah tersebut diperoleh suatu kesimpulan. Runut maju memulai proses pencarian dengan data sehingga strategi ini juga diisebut dengan *data-driven.*

2. ***Backward Chaining*** **(Runut Balik)**

Runut balik merupakan strategi pencarian yang arahnya kebalikan dari runut maju. Proses pencarian dimulai dari tujuan, yaitu kesimpulan yang menjadi solusi permasalahan yang dihadapi. Mesin inferensi mencari kaidah-kaidah dalam basis pengetahuan yang kesimpulannya merupakan solusi yang ingin dicapai, kemudian dari kaidah-kaidah yang dipeoleh, masing-masing kesimpulan di runut balik jalur yang mengarah ke kesimpulan tersebut. Jika informasi-informasi atau nilai dari *atribut-atribut* yang mengarah ke kesimpulan tersebut sesuai dengan data yang diberikan maka kesimpulan tersebut merupakan solusi yang dicari, jika tidak sesuai maka kesimpulan tersebut bukan merupakan solusi yang dicari. Runut balik memulai proses pencari dengan suatu tujuan sehingga strategi ini disebut juga dengan *goal-driven*. (Jurnal SAINTIKOM Vol.10 No.3 September 2011).

Kedua metode inferensi tersebut dipengaruhi oleh tiga macam penelusuran, yaitu *Depth First Search, Breadth First Search* dan *Best First Seacrh.*

1. ***Depth-first search,*** melakukan penelusuran kaidah secara mendalam dari simpul akar bergerak menurun ke tingkat dalam yang berurutan.
2. ***Breadth first search,*** bergerak dari simpul ke akar, simpul yang ada pada setiap tingkat diuji sebelum pindah ke tingkat selanjutnya.
3. ***Best first search,*** bekerja berdasarkan kombinasi kedua metode sebelumnya. (Rika Rosnelly, 2012:17)

II.2 Gangguan Genital dan Saluran Kencing

Infeksi saluran kencing (kemih) adalah kolonisasi bakteri di berbagai segmen disaluran kemih. Jumlah *organisme* dalam urine lebih besar dari yang dapat ditampung. Kriteria diagnostik paling umum untuk ISK adalah terdapat minimal 100.000 koloni bakteri dalam 1 ml urine pancar tengah pada dua biakan berturut-turut. Adanya urine dan kotoran disekeliling meatus urinarius memungkinkan bakteri berproliferasi dan naik kedalam ureta. Anak-anak yang beresiko adalah mereka yang mengalami defek pada system perkemihan, penyakit kronis, dan kelainan neulogis. (Cecily Lynn Betz, Linda A. Sowden, 2009:680)

II.2.1 *Enuresis*

Ngompol (*enuresis*) adalah hal biasa. Umur saat anak tidak kencing di tempat tidur pada malam hari sangat bervariasi. Tak banyak anak yang dapat mengontrol fungsi kandung kemih mereka sebelum mencapai usia 3 tahun. Kontrol kandung kemih yang baik pada siang dan malam hari dapat diperoleh kapan saja antara usia 3 sampai 7 tahun. Umumnya tidak ada yang perlu

dikhawatirkan kecuali anak Anda masih ngompol sampai usia 7 tahun atau mulai ngompol setelah lama tak ngompol (6 sampai 12 bulan).

II.2.1.1 Gejala Gangguan *Enuresis*

Gejala *enuresis* terlihat ketika anak ngompol pada saat tidur, baik kadang-kadang ngompol ataupun rutin setiap malam, padahal pada pemeriksaan kencing dan fisik tidak didapatkan kelainan. Pada kasus yang disebabkan oleh faktor kejiwaan juga didapat gejala ngompol di siang hari. Biasanya tak ada rasa sakit saat kencing tapi sering kencing dan harus buru-buru bila sudah terdesak atau kebelet kencing. Disamping itu pada banyak kasus pemeriksaan radiologi tak ada kelainan.

II.2.1.2 Penyebab Gangguan *Enuresis*

Pada sebagian besar kasus, *enuresis* kemungkinan akibat terlambatnya kematangan bagian-bagian sistem syaraf yang mengendalikan kandung kemih. Enuresis juga dapat muncul jika seorang anak mengalami Infeksi Saluran Kencing atau sedang merasa cemas. Meski jarang, kerusakan bawaan pada sistem saluran kencing atau Diabetes Mellitus dapat menjadi penyebabnya.

II.2.1.3 Solusi Gangguan *Enuresis*

Jika penyebab fisik *enuresis* ditemukan, beri antibiotik untuk mengobati infeksi sistem saluran kencing. Jika tak ada penyebab fisik ditemukan, maka jangan langsung menghukum anak anda jika ia mengompol, kemungkinan anda membuatnya semakin cemas dan *problem* akan memburuk. Sebaiknya, puji anak Anda jika ia tidak mengompol. Membuat grafik agar anak anda dapat

mencoretkan tanda bintang pada grafik itu tiap kali ia tidak ngompol dapat member motivasi.

II.2.2 Infeksi Sistem Saluran Kencing

Infeksi saluran kencing sering terjadi pada anak-anak, khususnya pada anak putri. Namun, bayi laki-laki yang baru lahir lebih rentan terhadap infeksi saluran kencing dibanding bayi perempuan. Infeksi dapat menyerang saluran kencing (*urethritis*), kandung kemih (*cystitis*), dan atau ginjal (*pyelonepbritis*). Perawatan yang tepat terhadap infeksi saluran kencing penting untuk mencegah goresan pada ginjal, yang paling cenderung muncul pada anak BALITA.

II.2.2.1 Gejala Infeksi Sistem Saluran Kencing

Anak di bawah usia 2 tahun cenderung mengalami gejala infeksi umum, yang berupa : demam, diare, muntah ,kurang energy atau gelisah.

Pada anak usia lebih tua, gejala umumnya lebih spesifik, dapat berupa : rasa panas saat mengeluarkan urin, semakin sering mengeluarkan urin, sakit yang menyerang punggung bagian bawah atau salah satu sisi perut, ngompol, meski sebelumnya tidak pernah lagi ngompol, urin berwarna merah, merah muda atau berkabut, karena terdapat darah, demam.

II.2.2.2 Penyebab Infeksi Sistem Saluran Kencing

Infeksi saluran kencing paling sering disebabkan oleh bakteri dari dubur yang memasuki saluran kencing dalam kemaluan. Bakteri terkadang menyebar sampai saluran kencing melalui aliran darah. Anak penderita reflux saluran kencing lebih rentan terhadap infeksi. Dalam kondisi bawaan ini, saat kandung

kemih kosong, ada urin yang lewat kearah belakang menuju ginjal. Anak yang memiliki cacat bawaan pada saluran kencing, menderita Sembelit kronis atau ginjalnya tergores akibat infeksi sebelumnya juga lebih rentan terhadap infeksi.

II.2.2.3 Solusi Infeksi Sistem Saluran Kencing

Jika ditemukan adanya infeksi, dokter dapat memberikan *antibiotic oral*. Anak yang infeksinya serius dapat dirawat dirumah sakit dengan *antibiotic* yang disuntikkan melalui pembuluh darah. *Antibiotic* disuntikan sampai seminggu. Untuk pencegahan, anak sebaiknya mengeluarkan urin setidaknya tiap 4 jam atau sebelum makan dan sebelum tidur. Setelah menggunakan toilet, sebaiknya seka bokong anak anda dari arah depan ke belakang. Anak anda sebaiknya mandi tiap hari, menghindari penggunaan bahan penyebab iritasi, seperti sabun beraroma atau *bubble bath.*

II.2.3 *Sindrom Nephrotic*

Sindrom Nephrotic merupakan gangguan yang jarang ditemui, dan terutama menyerang anak usia antara 1 sampai 6 tahun.

II.2.3.1 Gejala Gangguan *Sindrom Nephrotic*

Gejala utamanya adalah :

1. Pembengkakan bagian tubuh, umumnya terjadi secara perlahan selama beberapa minggu.
2. Pengurangan jumlah urin.
3. Berat badan bertambah.
4. Terkadang, diare, hilang nafsu makan, dan kelelahan yang tidak biasa.

Anak yang menderita *sindrom nephritic* rentan terhadap infeksi dan pembentukan darah beku dalam urat darah halus.

II.2.3.2 Penyebab Gangguan *Sindrom Nephrotic*

Pada kondisi ini, sejumlah besar protein hilang dari aliran darah melalui ginjal, masuk ke dalam urin. Berkurangnya tingkat protein dalam darah menimbulkan *oedema* (penimbunan kelebihan cairan dalam jaringan tubuh).

II.2.3.3 Solusi Gangguan *Sindrom Nephrotic*

Anak Anda sebaiknya diperiksa dokter dalam 24 jam jika ada bagian tubuhnya yang membengkak. Dokter akan memeriksa anak anda dan menguji urinnya untuk mengetahui adanya protein. Jika hasil menunjukan adanya *sindrom nephritic*, anak anda akan dirujuk kerumah sakit untuk menjalani pemeriksaan lebih lanjut. Jika diagnosa ditegaskan, anak anda akan dirawat dirumah sakit dengan *corticosteroid*.

Dalam 10 hari, setelah kelebihan cairan dalam tubuh dikeluarkan oleh ginjal, terjadi kemajuan dalam *oedema*, bersamaan dengan turunnya berat badan secara drastis. Anak anda kemungkinan akan tetap mondok di rumah sakit sampai tes menunjukkan sudah ada penurunan berarti dalam jumlah protein yang hilang adalah urin.

II.2.4 Tumor Wilms

Tumor ginjal yang berbahaya (bersifat kanker) dan jarang ini dapat muncul saat kelahiran atau pada saat 4 tahun pertama usia anak. Dalam sebagian

besar kasus, tumor Wilms menyerang satu ginjal jarang sekali tumor ini menyerang dua ginjal.

II.2.4.1 Gejala Tumor Wilms

Gejala utama tumor ini adalah :

1. Pembengkakan perut.
2. Sakit pada perut.
3. Urin merah, merah muda, atau nerkabut, karena adanya darah.

II.2.4.2 Penyebab Tumor Wilms

Penyebab tumor wilms' hingga kini belum diketahui secara pasti, dipercaya kanker ini timbul saat anak masih dalam rahim. Tumor wilms' berasal dari kegagalan beberapa sel untuk membentuk ginjal, sebaliknya mereka berkembang biak dalam keadaan primitif mereka menjadi tumor yang puncaknya pada waktu anak berusia sekitar 3 atau 4 tahun. Beberapa kanker, seperti tumor Wilms', terjadi ketika perubahan (mutasi) gen pada anak. Dalam beberapa kasus, kanker ini disebabkan oleh cacat genetik yang diturunkan dari orang tua kepada anak. Sebagai contoh, beberapa kasus tumor wilms' berkaitan dengan cacat pada salah satu dari dua gen – baik Wilms' tumor 1 (WT1) atau wilms' tumor 2 (WT2). Para ilmuwan percaya bahwa mutasi pada kromosom lain juga mungkin memainkan peran dalam terjadinya tumor Wilms'.

II.2.4.3 Solusi Tumor Wilms

Anak anda sebaiknya diperiksa dokter dalam waktu 24 jam jika daerah perutnya bengkak dan/atau jika terdapat darah dalam urin. Jika dokter menduga anak Anda terkena tumor, ia akan dikirim ke rumah sakit untuk menjalani

pemeriksaan. *Ultrasound scanning* dan teknik *imaging* lainnya akan digunakan untuk menegaskan diagnosa dan member informasi mengenai sifat tumor.

Ginjal yang mengandung kanker akan diangkat melalui pembedahan. Kemoterapi dan, pada beberapa kasus, radioterapi dapat digunakan sebelum dilakukan operasi untuk mengurangi ukuran tumor; metode ini juga dapat digunakan setelah operasi untuk membunuh sel kanker yang masih ada. Anak Anda akan sembuh kembali asal ginjal yang terinfeksi sudah diangkat sebelum tumor membesar. Ginjal yang tertinggal mampu melaksanakan fungsi kedua ginjal.

II.2.5 *Vulvovaginitis*

Peradangan vulva dan vagina disebut *vulvovaginitis* masalah ini kerap kali terjadi pada remaja putri dan umumnya bersifat minor. *Vulvovaginitis* dapat disebabkan oleh iritasi jaringan genital yang lembut (misalnya, karena *bubble bath*) dan pada kasus ini, umumnya sembuh dengan perawatan pertolongan mandiri. Terkadang, penyebabnya adalah bakteri atau infeksi ragi dan diperlukan perawatan medis. Namun, seringkali *vulvovaginitis* tak memiliki penyebab pasti.

II.2.5.1 Gejala *Vulvovaginitis*

Gejala *vulvovaginitis* adalah :

1. Radang, sakit, dan gatal yang menyerang area genital.
2. Terasa sakit saat mengeluarkan urin.

3. Cairan vagina kehijauan atau kuning keabuan jika infeksi bakteri adalah penyebabnya. Kotoran dapat berbau jika infeksi disebabkan oleh benda asing dalam vagina.
4. Cairan vagina berwarna putih, kental, jika sariawan merupakan sebabnya.

II.2.5.2 Penyebab *Vulvovaginitis*

Penyebab umum pada iritasi ini antara lain kurangya kebersihan, pakaian ketat, *bubble bath,* atau sabun beraroma. Pada beberapa anak putri, mungkin tidak ada penyebab nyata pada gejala; lapisan vulva dan vagina hanya terasa sensitive. Kemungkinan penyebab pada remaja putri dapat berupa suatu serangan Cacing Benang. Setelah masa pubertas, sariawan merupakan penyebab umum.

II.2.5.3 Solusi *Vulvovaginitis*

Vulvovaginitis yang tidak disebabkan oleh suatu infeksi umumnya hilang melalui perawatan mandiri. Dua kali sehari selama seminggu, anak anda sebaiknya membersihkan area tersebut tanpa menggunakan sabun beraroma atau *bubble bath*. Setelah itu, oleskan krim *barrier* (misalnya krim *zinc*). Anak anda sebaiknya mengenakan pakaian dalam katun longgar, yang diganti setiap hari. Jika mungkin, usahakan agar area genital terkena udara meski sebentar, namun tiap hari.

II.3 Aplikasi *Web*

Kepopuleran Internet di seluruh penjuru dunia mendorong aplikasi *web* semakin diminati. Dengan menggunakan aplikasi *web*, Anda hanya perlu menempatkan aplikasi dalam sebuah *server* dan dengan sendirinya aplikasi

tersebut dapat diakses dari manapun, sepanjang pemakai dapat mengakses *web server* nya. Aplikasi *web* yang paling dasar ditulis dengan menggunakan HTML. Sebagaimana diketahui, HTML (*HyperText Markup Language*) adalah bahasa standar untuk membuat halaman-halaman *web*. (Abdul Kadir, 2009:2).

II.4 *Database*

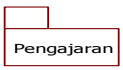
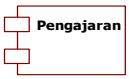
Database sering didefinisikan sebagai kumpulan data yang terkait. Secara teknis, yang berada dalam sebuah *database* adalah sekumpulan tabel objek lain (*indeks, view,* dan lain-lain). Tujuan utama pembuatan *database* adalah untuk memudahkan dalam mengakses data. Data dapat ditambahkan, diubah, dihapus, atau dibaca dengan *relative* mudah dan cepat. (Abdul Khadir, 2009:14).

II.5 UML *(Unified Modelling Language)*

Unified Modelling language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi merancang dan mendokumentasikan sistem piranti lunak.UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML dapat dibuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut bisa berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemprograman apapun. (Jurnal Teknologi Informasi DINAMIK Volume XIV, No.1, Januari 2009:23-29).

Ada beberapa pengklasifikasi (*Classifer*) dan notasi dalam UML (Adi Nugroho, 2010:16) yang ditunjukkan pada Tabel II.1.

Tabel II.1. Pangklaisfikasi dan Notasi UML

Pengklasifikasi	**Kegunaan**	**Notasi**
Actor	Menggambarkan semua objek diluar sistem yang berinteraksi dengan sistem yang dikembangkan.	
Use Case	Menggambarkan fungsionalistis yang dimiliki sistem.	
Kelas (*Class*)	Menggambarkan konsep dasar pemodelan sistem.	
Subsistem (*Subsystem*)	Menggambarkan paket spesifikasi serta implementasi.	
Komponen (*Component*)	Menggambarkan bagian-bagian fisik sistem/perangkat lunak yang dikembangkan.	
Antarmuka (*Interface*)	Menggambarkan antarmuka pengiriman pesan (*message*) antar pengklasifikasi.	
Simpul (*Node*)	Menggambarkan sumber daya komputasional yang digunakan oleh sistem.	

(Sumber : Adi Nugroho, 2010:16)

Relasi-relasi antar pengkalsifikasi yang dikenali UML adalah asosiasi, generalisasi, aliran, dan berbagai jenis kebergantungan termasuk didalamnya realisasi dan penggunaan. Untuk lebih jelasnya dapat dilihat pada Tabel II.2.

Tabel II.2. Relasi-relasi dalam UML

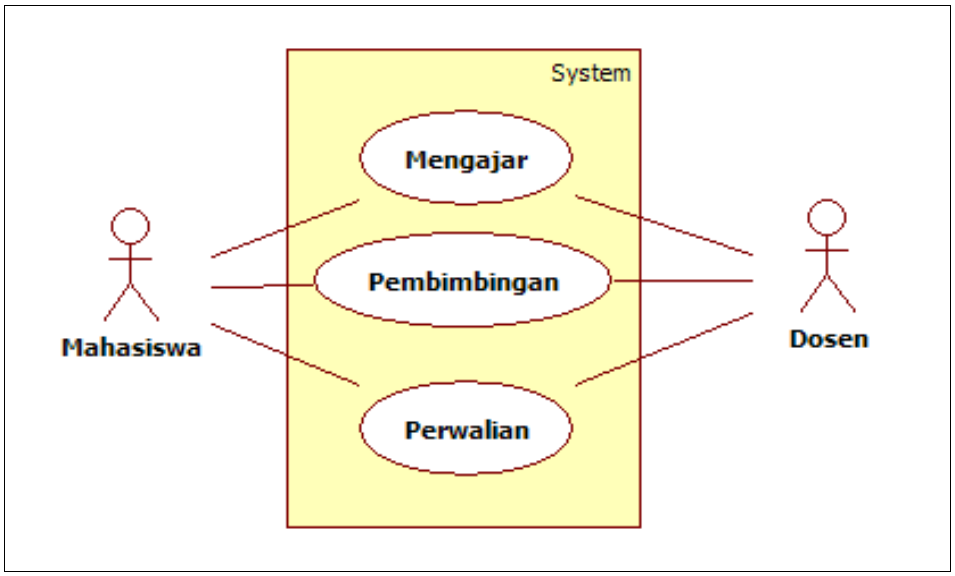
Relasi	Fungsi	Notasi
Asosiasi (*Association*)	Mendeskripsikan hubugan antar instance suatu kelas.	————————
Kerbergantungan (*Dependency*)	Relasi antar dua elemen model.	- - - - - - - - - ->
Aliran (*Flow*)	Relasi antar dua versi suatu model.	- - - - - - - - - ->
Generalisasi (*Generalization*)	Relasi antar pengkalisifikasi yang memiliki deskripsi yang bersifat lebih umum dengan berbagai pengklasifikasi yang lebih spesifik, digunakan dalam struktur pewarisan.	————————→
Realisasi (*Realization*)	Relasi antara spesifikasi dan implementasinya.	- - - - - - - - - -→
Penggunaan (*Usage*)	Situasi di mana salah satu elemen membutuhkan elemen yang lainnya agar dapat berfungsi dengan baik.	- - - - - - - - - ->

(Sumber : Adi Nugroho, 2010:23)

II.5.1 *Use Case Diagram*

Diagram ini bersifat statis. Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.*Use Case Diagram* adalah fungsionalitas atau persyaratan – persyaratan sistem yang harus dipenuhi oleh sistem yang akan dikembangkan tersebut menurut pandangan pemakai sistem (Sholiq, 2010:21). *Use case* diagram juga dapat diartikan sebagai urutan langkah-langkah yang secara tindakan saling terkait (skenario), baik terotomatisasi maupun secara

manual, untuk tujuan melengkapi satu tugas bisnis tunggal. *Use case* digambarkan dalam bentuk *ellips/oval*.

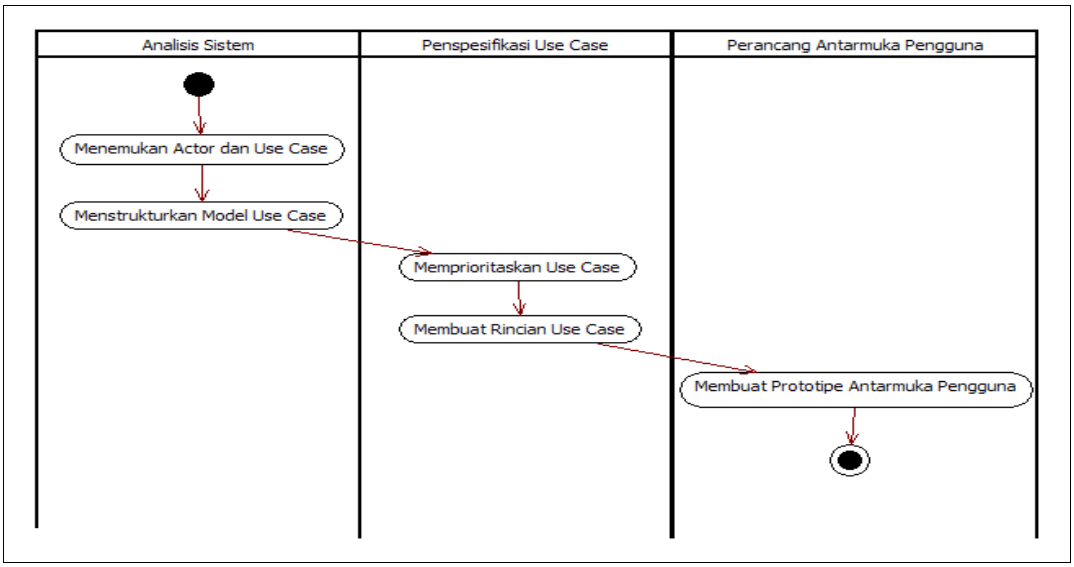


Gambar II.3. Contoh *Use Case Diagram*

(Sumber : Adi Nugroho, 2010:34)

II.5.2 *Activity Diagram*

Acitivity diagram memodelkan alur kerja (*workflow*) sebuah proses bisnis dan urutan aktivitas dalam suatu proses (Sulistyorini, 2009:27). Diagram ini sangat mirip dengan sebuah *flowchart* karena dapat dimodelkan sebuah alur kerja dari satu aktivitas ke aktivitas lainnya atau dari satu aktivitas ke dalam keadaan sesaat (*state*). Seringkali bermanfaat bila dibuat sebuah*activity* terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. *Activity diagram* juga sangat berguna ketika ingin menggambarkan perilaku paralel atau menjelaskan bagaimana perilaku dalam berbagai *use case* berinteraksi.



Gambar II.4. Contoh *Activity Diagram*

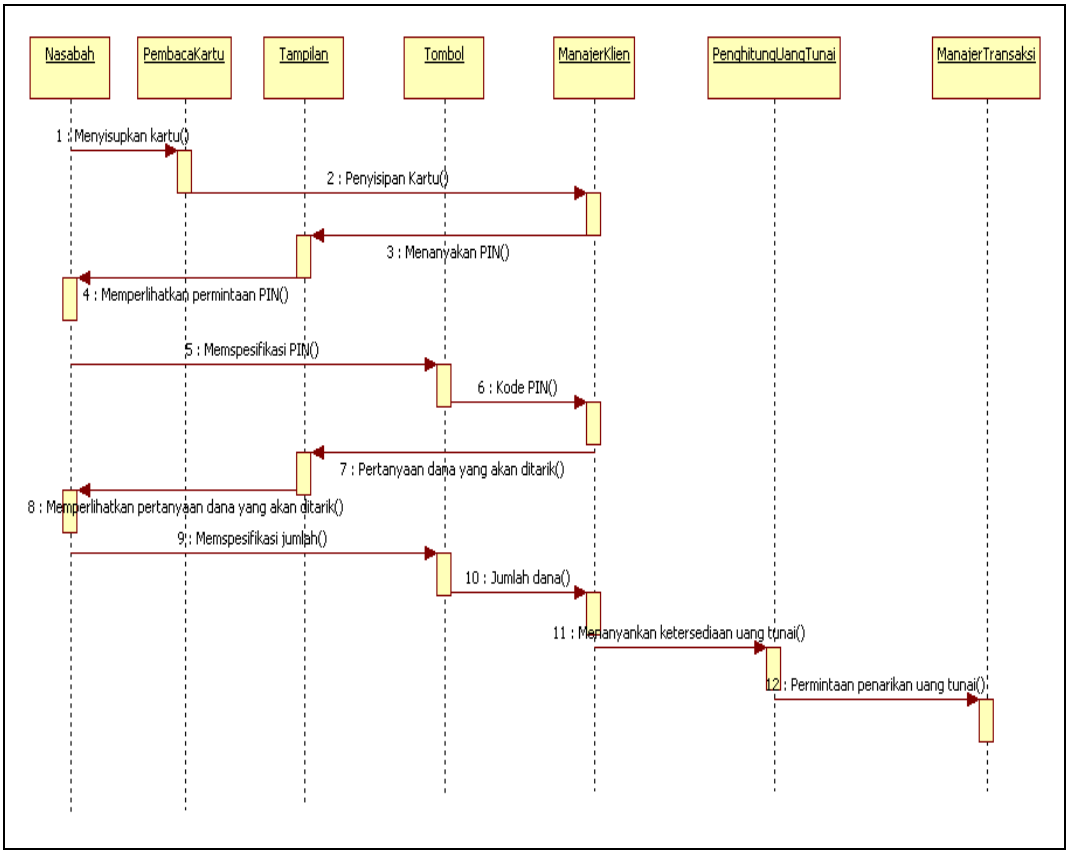
(Sumber : Adi Nugroho, 2010:141)

Dapat digunakan *statechart* diagram untuk memodelkan perilaku dinamis satu kelas atau objek. *Statechart* diagram memperlihatkan urutan keadaan sesaat (*state*) yang dilalui sebuah objek, kejadian yang menyebabkan sebuah transisi dari satu state atau aktivitas ke *state* atau aktivitas lainnya, dan aksi yang menyebabkan perubahan satu state lainnya, dan aksi yang menyebabkan perubahan satu state atau aktivitas. Diagram aktivitas paling cocok digunakan untuk memodelkan aktivitas dalam suatu proses.

II.5.3 *Sequence Diagram*

Sequence Diagram memeperlihatkan interaksi sebagai diagram dua mantra (dimensi). Mantra vertikal adalah sumbu waktu; waktu bertambah dari atas ke bawah sedangkan mantra horizontal memperlihatkan pengklasifikasi yang mempresentasikan objek-objek mandiri yang terlibat dalam kolaborasi. *Diagramsequence* merupakan diagram interaksi yang menekankan pada

pengiriman pesan (*message*) dalam suatu waktu tertentu (Sulistyorini, 2009:24). *Sequence Diagram* digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah contoh objek dan *messege* (pesan) yang diletakkan diantara objek-objek ini dalam *use case*.



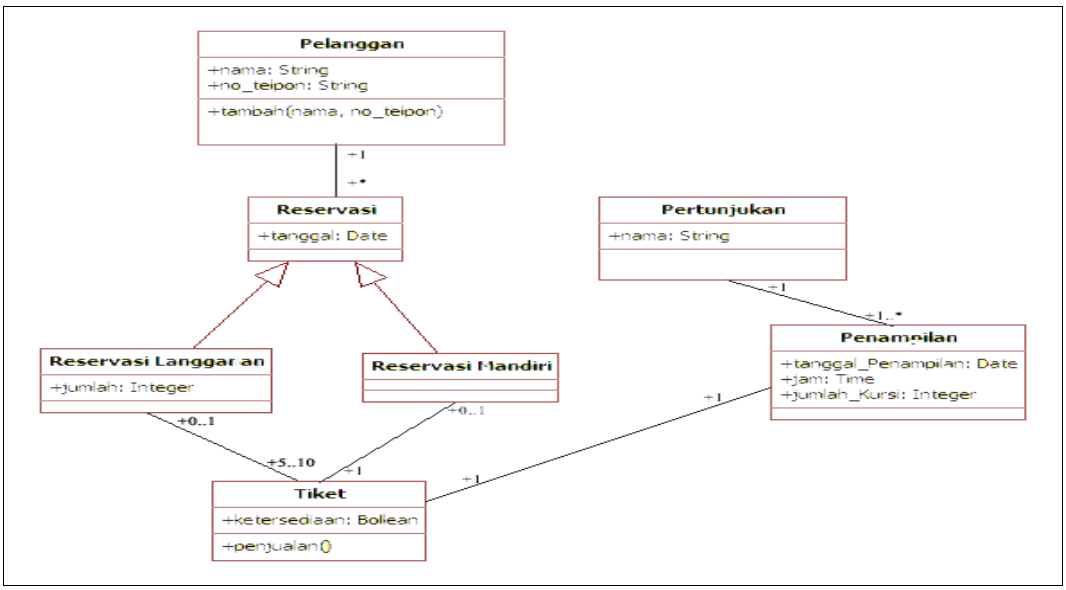
Gambar II.5. Contoh *Sequence Diagram*

(Sumber : Adi Nugroho, 2010:109)

II.5.4 *Class Diagram*

Class Diagram menunjukkan interaksi antar kelas – kelas dalam sistem. Sebuah kelas mengandung informasi dan tingkah laku (*behavior*) yang berkaitan dengan informasi tersebut.*Class diagram* sesungguhnya merupakan deskripsi dari konsep yang datang dari ranah aplikasi atau solusi aplikasi. (Adi Nugroho : 2010

:11). Oleh karena itu pengertian kelas sangat penting sebelum merancang diagram kelas. Kelas diagram sebagai satu *set* objek yang memiliki atribut dan prilaku yang sama.*Class diagram* membantu dalam visualisasi struktur kelas – kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak. *Class diagram* memperlihatkan hubungan antar kelas dan penjelasan detail tiap – tiap kelas di dalam model desain (dalam *logical view*) dari suatu sistem. Selama proses analisi, *class diagram* memperlihatkan aturan – aturan dan tanggung jawab entitas yang menentukan perilaku sistem.



Gambar II.6. *Contoh Class Diagram*

(Sumber : Adi Nugroho, 2010:12)

II.6 ERD (*Entity Relationship Diagram)*

Pada dasarnya ERD (*Entity Relationship Diagram*) adalah sebuah diagram yang secara konseptual memetakan hubungan antar penyimpanan. ERD ini digunakan untuk melakukan pemodelan terhadap *struktur* data dan hubungannya.

Penggunaan ERD ini dilakukan untuk mengurangi tingkat kerumitan penyusunan sebauh *database* yang baik.(WAHANA KOMPUTER, 2010:30).

Model *entity relationship* terbentuk karena didukung oleh beberapa komponen yang saling berhubungan satu sama lain. Komponen-komponen yang terdapat pada model *entity relationship* adalah sebagai berikut :

1. *Entity*

   Entity adalah suatu yang dapat dibedakan dalam dunia nyata dimana informasi yang berkaitan dengannya dikumpulkan. Simbol *entity* adalah persegi panjang.

2. *Relationship*

   Merupakan hubungan yang terjadi antara satu atau lebih *entity*.

3. *Attribute*

   Attribute merupakan karakteristik dari *entity* atau *relationship* yang menyediakan penjelasan detail tentang hal tersebut. Nilai *attribute* adalah suatu data yang aktual.

4. Indikator Tipe

   Indikator tipe ada dua, yakni : indikator tipe *asscociative object* dan indikator tipe supertipe.

5. *Cardinality Rasio*

   Menjelaskan hubungan batasan jumlah keterhubungan satu *entity* dengan *entity* lainnya atau banyaknya *entity* yang bersesuaian dengan *entity* yang lain melalui *relationship*.

6. Derajat *Relationship*

   Derajat *Relationship* menyatakan jumlah entity yang berpatisipasi di dalam suatu *relationship*.

7. *Participantion Contraint*

   Participantion Contraint, menjelaskan apakah keberadaan suatu *entity* tergantung pada hubungannya dengan *entity* yang lain.

8. Representasi dari *Entity Set*

   Entity set dipresentasikan dalam bentuk tabel dan nama yang *unique*. Setiap tabel terdiri dari sejumlah kolom dan diberi nama yang *unique*.

II.6.1. Normalisasi

Normalisasi data adalah proses dimana tabel-tabel dalam database dites dalam hal kesalingtergantungan diantara *field-field* pada sebuah tabel. (Wahana Komputer, 2010:32). Ada beberapa bentuk dalam *normalisasi* tabel, yaitu :

1. Bentuk Tidak *Normal* (*Unnormalized Form*)

   Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan untuk mengikuti suatu *format* tertentu. Dapat saja data tidak lengkap atau terduplikasi. Data tersebut dikumpulkan apa adanya.

2. Bentuk *Normal* Pertama (1NF)

   Suatu relasi 1NF jika dan hanya jika sifat dan setiap relasi atributnya bersifat atomik. Atomik bermaksud tidak berkepunyaan untuk berada dalam keadaan satu bagian. Ciri-ciri bentuk *normal* pertama, yaitu :

   a. Setiap data dibentuk dalam *flat file*.

b. Tidak ada *set atribut* yang berulang atau bernilai ganda.

c. Tiap *field* hanya satu pengertian.

3. Bentuk *Normal* Kedua (2NF)

   Bentuk *normal* kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk *normal* pertama. *Atribut* bukan kunci haruslah bergantung secara fungsi pada *primary key*. Peraturan ini menentukan kebergantungan sepenuhnya. Beberapa sumber teks menjelaskan sebagai kebergantungan secara fungsi dan transitif.

4. Bentuk *Normal* Ketiga (3NF)

   Satu hubungan dikatakan dalam bentuk 3NF jika dan hanya jika ia dalam bentuk 2NF dan setiap *atribut* tanpa kekunci pula bergantungan secara tidak transitif dengan kunci *primer*.

II.6.2. Kamus Data

Kamus data (*data dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data (Raymond dan George, 2008:171). Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi *field*, definisi tabel, relasi tabel dan hal-hal lainnya.

Kamus data dibuat dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. Pada tahap analisis kamus data digunakan sebagai alat komunikasi antara sistem analis dengan *user* tentang data yang mengalir pada sistem tersebut serta informasi yang dibutuhkan oleh pemakai sistem. Karena

kamus data merupakan suatu *file* yang terpisah dalam suatu basis data, sehingga kamus data menyimpan informasi seperti :

1. Nama setiap *item*/jenis atau kolom data.
2. Struktur data untuk setiap *item*.
3. *Program* yang menggunakan tiap *item*.
4. Ukuran *item* pada setiap tipe data.

Kamus data berguna khusus bagi perlindungan timbulnya kelebihan data. Tanpa kamus data, pemakai data lain bagian mungkin menyimpan versi identik dari *item* data yang sama pada beberapa lokasi, di mana masing-masing *item* data mempunyai nama yang berbeda.

II.7 PHP

PHP singkatan dari PHP *Hypertext Preprocessor* yaitu bahasa pemprograman *web server-side* yang bersifat *open-source.* PHP merupakan *script* yang ter*integrasi* dengan HTML dan berada pada *server (server side* HTML *embedded scripting)*. PHP adalah *script* yang digunakan untuk membuat halaman *website* yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halama itu diminta oleh *client*. Mekanisme ini meyebabkan informasi yang diterima *client* selalu yang terbaru/*up to date*. Semua *script* PHP dieksekusi pada *server* dimana *script* tersebut dijalankan.(Anhar, ST, 2010:3);

II.8 MySQL

MySQL (baca: mai-se-kyu-el) merupakan *software* yang tergolong *database server* dan bersifat *Open Source. Open Source* menyatakan bahwa *software* ini dilengkapi dengan *source code* (kode yang dipakai untuk membuat MySQL), selain tentu saja bentuk *executable*-nya atau kode yang dapat dijalankan secara langsung dalam sistem operasi. Hal menarik lainnya adalah MySQL juga bersifat *multiplatform.* MySQL dapat dijalankan pada berbagai sistem operasi.

Pengaksesan data dalam *database* dapat dilakukan dengan mudah melalui SQL (*Structured Query Language).* Data dalam *database* bisa diakses melalui aplikasi non-*Web* (misalnya dengan Visual Basic) maupun aplikasi *Web* (misalnya dengan PHP). (Abdul Khadir, 2009:15).

II.9 Adobe Dreamweaver CS4

Adobe Dreamweaver CS4 merupakan salah satu *program* aplikasi yang digunakan untuk membangun sebuah *website*, baik secara grafis maupun dengan menuliskan kode sumber secara langsung.

Adobe Dreamweaver CS4 memudahkan pengembang *website* untuk mengelola halaman-halaman website dan aset-asetnya, baik gambar (*image*), animasi flash, video, suara dan lain sebagainya. Selain itu, Adobe Dreamweaver CS4 juga menyediakan fasilitas untuk melakukan pemprograman scripting, baik ASP (*Active Server Page*), JSp (*Java Server Page*), PHP (PHP *Hypertext Preprocessor*), JavaScript (js), Cold Fusion, CSS (*Cascading Style Sheet*), XML (*Extensible Markup Language*) dan lainnya. (WAHANA KOMPUTER, 2010:2).