BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Sistem Pakar

Mnurut Buku (Rika Rosnelly ; 2012 : 2,3) sistem pakar adalah sistem komputer yang ditujukan untuk meniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*decision making*) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah. Pakar atau ahli (*expert*) didefenisikan sebagai seseorang yang memiliki pengetahuan tau keahlian khusus yang tidak dimiliki oleh kebanyakan orang. Seorang pakar dapat memecahkan masalah yang tidak mampu dipecahkan kebanyakan orang. Dengan kata lain, dapat memecahkan suatu masalah dengan lebih efisien namun bukan berarti lebih murah. Pengetahuan yang dimuat dalam sistem bukan berarti lebih murah. Pengetahuan yang dimuat ke dalam sistem pakar dapat berasal dari buku, jurnal, majalah dan dekumentasi yang dipublikasikan lainnya, serta orang yang memiliki pengetahuan meskipun bukan ahli. Istilah sistem pakar (*expert system*) sering di sinonimkan dengan sistem berbasis pengetahuan (*knowladge-based system*) atau sistem pakar berbasis pengetahuan (*knowladge-based expert system*).

Munurut Jurnal (Muhammad Dahria, Dkk ; 2013 : 2) Sistem Pakar (*Expert System*) adalah program yang berbasis pengetahuan yang menyediakan solusi-solusi dengan kualitas pakar untuk problema-problema dalam suatu domain yang spesifik. Sistem pakar merupakan program computer yang menuru proses

pemikiran dan pengetahuan pakar dalam menyelesaikan suatu masalah tertentu. Implementasi sistem pakar dapat digunakan dalam bidang kesehatan karena sistem pakar dipandang cara penyimpanan pengetahuan pakar pada bidang tertentu dalam program computer sehinggga keputusan dapat diberikan dalammelakukan penalaran secara cerdas.

II.1.1. Konsep Dasar Sistem Pakar

Menurut E-Book Ilmu computer No. 2 (2003-2010) konsep dasar sistem pakar mengandung : keahlian, ahli, pengalihan keahlian, inferensi, aturan dan kemampuan menjelaskan. Keahlian adalah suatu kelebihan penguasaan pengetahuan di bidang tertentu yang diperoleh dari pelatihan, membaca atau pengalaman. Contoh bentuk pengetahuan yang termasuk keahlian adalah:

a. Fakta-fakta pada lingkup permasalahan tertentu.
b. Teori-teori pada lingkungan permasalahn tertentu.
c. Prosedur-prosedur dan aturan-aturan berkenaan dengan lingkup permasalahan tertentu.
d. Strategi-strategi global untuk menyelesaikan masalah.
e. Meta-knowladge (pengetahuan tentang pengetahuan).

II.1.2. Struktur Sistem Pakar

Sistem pakar terdidri dari 2 bagian pokok, yaitu : lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan digunakan sebagai

pembangunan sistem pakar dari segi pembangunan komponen maupun basis pengetahuan. Lingkungan konsultasi digunakan oleh seorang yang ahli untuk berkonsultasi. (Ari Fadli 2010, 5)

1. Basis Pengetahuan (*Knowledge Base*)

Basis pengetahuan berisi pengetahuan-pengetahuan dalam penyelesaian masalah, tentu saja didalam domain tertentu. Ada 2 bentuk pendekatan basis pengetahuan yang sangat umum digunakan, yaitu :

a. Penalaran Berbasis Aturan (*Rule Based Reasoning*)

Pada penalaran berbasis aturan, pengetahuan dipresentasikan dengan menggunakan aturan 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.

b. 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 digunkan apabila kita telah memiliki sejumlah situasi atau kasus tertentu dalam basis pengetahuan.

Menurut Jurnal Muhammad dahria, Dkk. ; 2013 : 3) Struktur Sistem Pakar disusunoleh dua bagian utama, yaitu lingkungan pengembangan (*development*

environment) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan sistim pakar digunakan untuk memesukan ilmu pengetahuan sistim pakar ke dalam lingkungan sistim pakar. Sedangkan lingkungan konsultasi digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar.

II.2. Teori Dempster Shafer

Menurut jurnal (Iswari Nur Hidayati ; 2010 : 57) Teori Dempster Shafer adalah teori yang mampu menangani berbagai kemungkinan yang mengkombinasikan satu kemungkinan dengan fakta yang ada. Dalam *demster shafer theory* (DST) ada berbagai konflik yang dipersatukan untuk mengkombinasikan dari berbagai informasiyang ada. Kumpiulan informasi yang bersifat berbeda dan menyeluruh dalam teori ini dikenal dengan *frame discernment* (ɵ). Bagian dari himpunan bagian (*sub set*) ɵ juga merupakan *hipotesis*. *Belief* dapat diberikan kedalam semua kemungkinan yang ada dalam setiap *subset* ɵ, yang dinotasikan menjadi 2 ɵ. Hal ini menggambarkan bahwa himpunan ɵ adalah suatu himpunan yang elemenya merupakan semua himpunan bagian dari ɵ, termasuk himpunan kosong dan himpunan ɵ sendiri. *Belief* merupakan unsur yang disebut dengan "kepercayaan" yang diberikan sampai label piksel itu hamper tidak mempunyai perbedaan yang signifikan. Sebagai contoh jika ukuran satu set adalah n, maka akan mempunyai 2^{n} *subset* (himpunan bagian). Artinya bahwa jika salah satu piksel mempunyai nilai n maka label piksel itu akan mempunyai 2^{n} kemungkinan.

Menurut Jurnal (Muhammad dahria, DKK. ; 2013 : 8) teori *demster shafer* adalah suatu teori metematika untuk pembuktian berdasarkan *belief functions* and *plausible reasoning* (fungsi kepercayaan dan pemikiran yang masuk akal), yang digunakan untuk mengkombinasikan potongan informasi yang terpisah (bukti) untuk mengkalkulasikan kemungkinan dari suatu peristiwa. Rumus dari *demster shafer* :

$$M3(Z) = \frac{\sum x \cap y = z\, m1(x)\, m2(y)}{1 - \Sigma x \cap y = 0\; m1(x)\, m2(y)}$$

dalam menghadapi suatu permasalahan sering ditemukan jawaban yang tidak memiliki berupa penuh. Ketidakpastian ini dapat berupa hasil suatu kejadian. Hasil yang tidak pasti disebabkan oleh beberapa faktor, yaitu aturan yang tidak pasti dan jawaban pengguna yang tidak pasti atas suatu pertanyaan yang diajukan oleh sistem. Hal ini sangat diagnosis gangguan, dimana pakar tidak dapat mendefenisikan hubungan antara gejala dengan penyebabnya secara pasti dan pasien tidak dapat merasakan suatu gejala dengan pasti pula. Pada akhirnya akan ditemukan banyak kemungkinan diagnosis. *Demster Shafer* merupakan nilai parameter klinis yang diberikan untuk menunjukkan besarnya kepercayaaan. Dimana nilai bel(m) suatu gejala yang diinput antara (0-09).

II.3. Pengertian PHP

PHP merupakan bahasa pemograman yang digunakan untuk pengembangan web. Pemrograman ini bersifat server side yang di-embed dalam HTML. Dalam arti, suatu dokumen HTML dapat dimasuki script PHP.

Bahasa pemrograman ini memiliki fitur struk *control*, *operator*, *type variable*, *deklarasi fungsi*, *deklarasi class/object*. PHP merupakan bahasa yang berbasis pemrograman berorientasi objek dan procedural. Tag PHP yang digunakan dalam HTML menggunakan tag berbentuk tanda Tanya (<?...?>).

```
<html>
<head>
<title>script PHP</title>
</head>
<body>
<?php
Echo “menggunakan kode PHP”;
?>
</body>
</html>
```

(Benedicta Rini W ; 2012 : 210)

II.4. Sejarah MYSQL

MySQL pertama kali dirintis oleh seorang programmer database bernama Michael Widenius, yang dapat anda hubungi di-emailnya monthy@analytikerna.

MySQL database server adalah RDBMS (*Relasional Database Management System*) yang dapat menangani data yang bervolume besar. Meskipun begitu, tidak menuntut *resource* yang besar. MySQL adalah database yang paling popular diantara databast-database yang lain.

MySQL adalah program database yang mampu mengirim dan menerima data dengan sangat cepat dan multi-user.MySQL memiliki dua bentuk lisensi, yaitu *free software* dan *shareware*. Penulis sendiri dalam menjelaskan buku ini menggunakan MySQL yang free software karena bebas menggunakan database ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensi yang ada dibawah lisensi GNU/GPL (*general public license*), yang dapat anda downloadpada alamat resminya http://www.mysql.com MySQL sudah cukup lama dikembangkan, beberapa fase penting dalampengembangan MySQL adalah sebagai berikut :

a. MySQL dirilis pertama kali secara internal pada 23 Mei 1995.
b. Versi Windows dirilis pada 8 Januari 1998 untuk Windows 95 dan Windows NT.
c. Versi 3.23 : beta dari Juni 2000 Dan dirilis pada January 2001.
d. Versi 4.0 : beta dari Agustus2000 dan dirilis pada maret 2003 (unlons).
e. Versi 4.1: beta dari bulan Juni 2004 dirilis pada bulan Oktober 2004 (R-trees dan B-trees, subqueries, prepared statement.
f. Versi 5.0: beta dari bulan Maret 2005 dirilis pada Oktober 2005. (cursor, stored procedure, trigger, views XA transaction).
g. Suri Microsystem membeli MySQL AB pada tanggal 26 Februari 2008.

h. Versi 5.1: dirilis 27 November 2008 (event schedure, partitioning, plug-in API, row-based replication, server loq table).

(Wahana Komputer ; 2010 : 19)

II.5. Pengertian UML

UML singkatan dari *Unifield Modelling Language* yang berarti bahasa pemodelan standart. (Chonoles, 2003 : bab 1) mengatakan sebagai bahasa , berarti UML memiliki sintaks dan sistematik. Ketika kita membuat model menggunakan konsep UML ada aturan-aturan yang harus diikuti. Bagaimanaelemen pada model-model yang kita buat berhubungan satu dengan yang lainnya harus mengikuti stansar yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. Ketika pelanggan memesan sesuatu dari system, bagaimana transaksinya ? bagaimana system mengatasi error yang terjadi ? bagaimana keamanan terhadap system yang kita buat ?dan sebagainya dapat dijawab dengan UML.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

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

UML telah diaplikasikan dalam bidang investasi perbankan, lembaga kesehatan, departemen pertahanan, system terdistribusi, system pendukung alat kerja, retail, sales dan supplier.

Blok pembangunan utama UML adalah diagram. Beberapa diagram ada yang rinci(jenis timing diagram) danlainnya ada yang bersifat umum (misalnya diagram kelas). Para pengembangan system berorientasi objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan system yang mereka rancang. UML memungkin kan para anggota team untuk bekerja sama dengan bahasa model yang sama dalam mengaplikasikan beragam system. Intinya, UML merupakan alat komunikasi yang konsisten dalam mensuport para pengembangan system saat ini.sebagai perancang system, mau tidak mau pasti akan menjumpai UML, baik kita sendiri yang membuat atau sekedar membaca diagram UML buatan orang lain (Pilone,2005 : bab 1). (prabowo Pudjo widodo, Herlawati ; 2011 : 8,9)

II.5.1. Diagram-Diagram UML

Beberapa literature menyebutkan bahwa UML menyediakan Sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa diagram yang digabung, misalnya diagram komunikasi, diagram urutan dan diagram pewaktuan diagram menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokkan berdasarka sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain :

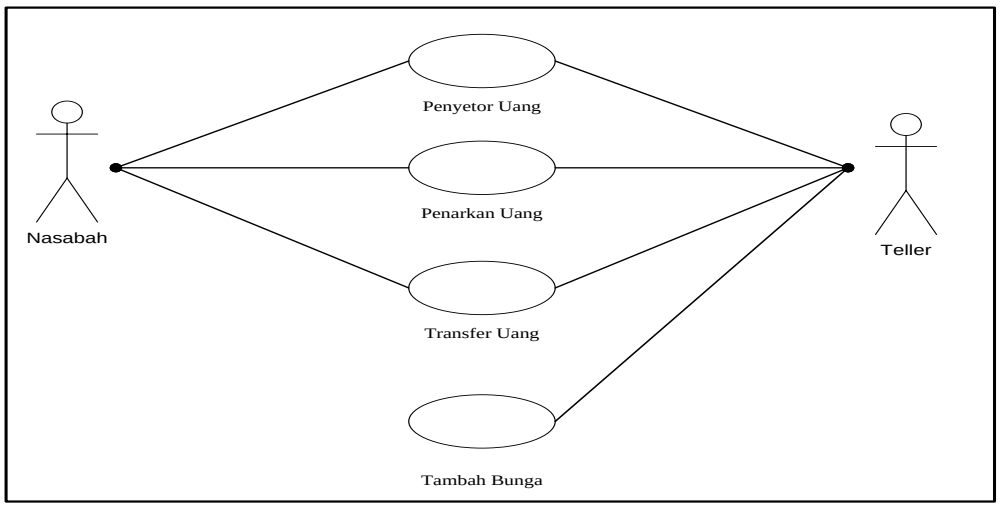
1. Diagram kelas. Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, Antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan system berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif.
2. Diagram paket (*Packet Diagram*). Bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas merupakan bagian dari diagram komponen.
3. Diagram *Use-Case*. Berifat statis. Diagram ini memperlitkan himpunan use-case dan actor-aktor (suatu jenis khusus dari kela). Diagram ini terutamasangat penting untuk mengorganisasi dan memodelkan perilaku suatu system yang dibutuhkan serta diharapkan pengguna.
4. Diagram interaksi dan *Sequence* (urutan). Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekan pada pengiriman pesan dalam suatu waktu tertentu.
5. Diagram komunikasi (*Communication Diagram*). Bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi UML 1.4 yang menekan organisasi structural dari objek-objek yang menerima serta mengirim pesan.
6. Diagram *Statechart* Diagram. bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada system, membuat status (*state*), transisi, kejadian serta aktifitas. Diagram ini terutama penting untuk memperlihatkan sifat dinamis dari antarmuka *(interface)*, kelas, kolaborasi dan terutama penting pada pemodelan system-sietem yang reaktif.
7. Diagaram aktivitas (*Activity Diagram*). Bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu

aktivitas ke aktivitas lainnya dalam suatu system. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu system dan memberi tekanan pada aliran kendali antar objek.

8. Diagram komponen (*Component diagram*). Bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan system/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan kedalam satu atau lebih kelas-kelas, antarmuk- antarmuka serta kolaborasi-kolaborasi.
9. Diagram Deployment (*Deployment Diagram*). Bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*). Memuat simpul-simpul beserta komponen- komponen yang ada didalamnya. *Diagram deployment* berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasikita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*).

Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan. Pada UML dimungkinkan kita menggunakan diagram-diagram lainnya (misalanya *Data Flow Diagram*, *Entity Relationship Diagram* dan sebagainya).

(Prabowo Pudjo Widodo Herlawati ; 2011 : 10,11,12).



Gambar II.1 Diagram Use case

Sumber : prabowo Pudjo widodo, Herlawati (2011 : 17)

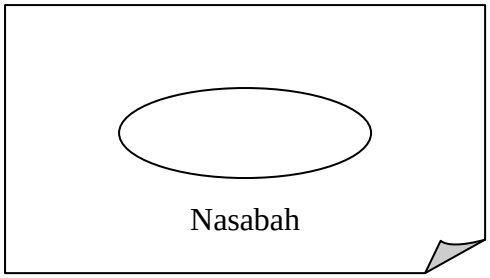
a) Use Case

Menurut (Pilone, 2005: 21) use case menggambarkan fungsi tertentu dalam suatu system berupa komponen, kejadian atau kelas. Sedangkan whitten, 2004: 258) mengartikan *use case* sebagai urutan langkah-langkah yang secara tindakan saling terkait (scenario), baik terotomatisasi maupun secara manual, untuk tujuan berhubungan dengan supplier dan pelanggan yang bersifat eksternal harus diperhatikan. (prabowo Pudjo widodo, Herlawati ; 2011 : 22)

Komponen pembentukan diagram *use case* adalah :

1. Aktor (*actor*), menggambarkan pihak-pihak yang berperan dalam system.
2. *Use case*, aktivitas/saran yang disiapkan oleh bisnis/system.
3. Hubungan (link), aktor mana saja yang terlibat dalam use case ini

Gambar dibawah ini merupakan salah satu contoh bentuk diagram use case.



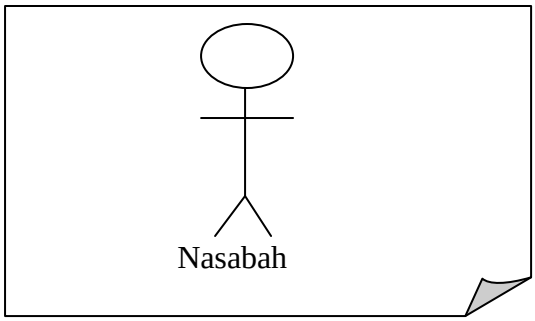
Gambar II.2 Simbol Use case

Sumber : prabowo Pudjo widodo, Herlawati (2011: 22)

b) Aktor

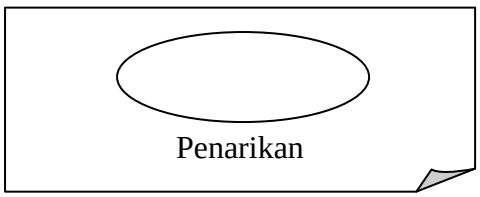
memperlihatkan diagram *use case* dengan dua aktor (nasabah dan teller) dan empat *use case* (Penyetor Uang, Penarikan Uang, Transfer Uang dan Tambah Bunga). Simbol aktor adalah orang. (Chonoles, 2003 : bab 8) menyarankan sebelum membuat use case dan menentukan aktornya, agar mengidentifikasi siapa saja pihak yang terlibat dalam sistem kita. Pihak yang terlibat biasanya dinamakan stakeholder. Langkah digambarkan dalam bentuk ellips/oval.

Gambar II.3



Gambar II.3 Aktor

Sumber : prabowo Pudjo widodo, Herlawati (2011 : 17)



Gambar II.4. Simbol Use case

Sumber : prabowo Pudjo widodo, Herlawati (2011: 22)

Use case sangat menentukan karakteristik system yang kita buat, oleh karena itu (Chonoles, 2003: bab 8) menawarkan cara untuk menghasilkan *use case* yang baik, yakni :

1. Pilihlah nama yang baik. *Use case* adalah sebuah behavior (perilaku), jadi seharusnya dalam frase kata kerja. Untuk membuat namanya lebih detail, tambahkan kata benda yang mengindikasikan dampak aksinya terhadap suatu kelas objek. Oleh karena itu diagram *use case* seharusnya berhubungan dengan diagram kelas.
2. Ilustrasikan perilaku dengan lengkap. *Use case* dimulai dari inisiasi oleh aktor primer dan berakhir pada actor dan menghasilkan tujuan. Jangan membuat *use case* kecuali anda mengetahui tujuannya. Sebagai cintoh, memilih jenis tempat tidur (*king size*, *queen size* atau dobel) saat tamu memesan tidak dapat case memesan kamar dan tidak dapat berdiri sendiri (tidak mungkin tamu memesan kamar tidur jenis king tapi tidak memesan kamar hotel).
3. Identifikasi perilaku dengan lengkap. Untuk mencapai tujuan dan menghasilkan nilai tertentu dari aktor, *use case* harus lengkap . ketika memberi nama pada *use case*, pilihlah frasa kata kerja yang

implikasinyahingga selesai. Misalnya gunakan *frasa reserve a room* (pemesanan kamar) dan jangan reserving a room (memesan kamar) karena memesan menggambarkan perilaku yang belum selesai.

4. Menyediakan use case lawan (inverse). Kita biasanya membutuhkan use case yang membatalkan tujuan, misalnya pada use case pemesanan kamar, dibutuhkan pula use case pembatalan pemesanan kamar .
5. Batasi use case hingga satu perilaku saja. Kadang kita cendrung membuat use case yang menghasilakn lebih dari satu tujuan aktivitas. Guna menghindari keracunan, jagalah *use case* kita hanya focus pada satu hal. Misalnya, penggunaan *use case Check-in* dan *Check-out* dalam satu *use case* menghasilkan ketidak fokusan, karena memiliki dua perilaku yang berbeda.

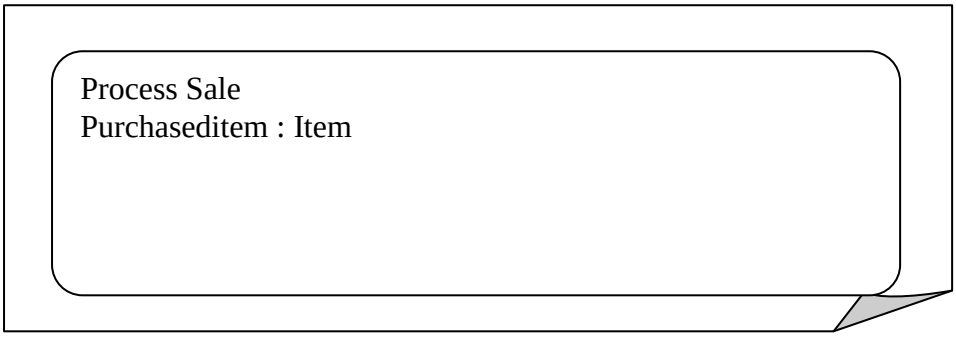
c) Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagian sistem itu dirakit. Diagram ini tidak hanya memodelkan software melainkan memodelkan model bisnis juga. Diagram aktivitas menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi. Aktivitas merupakan kumpulan aksi-aksi melakukan langkah sekali saja tidak boleh dipecah menjadi beberapa langkah lagi. Contoh aksi yaitu : Prabowo Pudjo Widodo Herlawati ; 2011 : 32,33,34)

1) Fungsi matematika
2) Pemanggilan perilaku
3) Pemrosesan data

Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu classifier, classifier dikatakan kontek dari aktivitas. Aktivitas dapat mengakses atribut dan operasi classfier, tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan untuk model proses bisnis, informasi itu biasaanya disebut process-relevant data. Aktivitas deharapkan dapat digunakan ulang dalam suatu aplikasi, sedangkan aksi biasanya spesifik dan digunakan hanya untuk aktivitas tertentu.

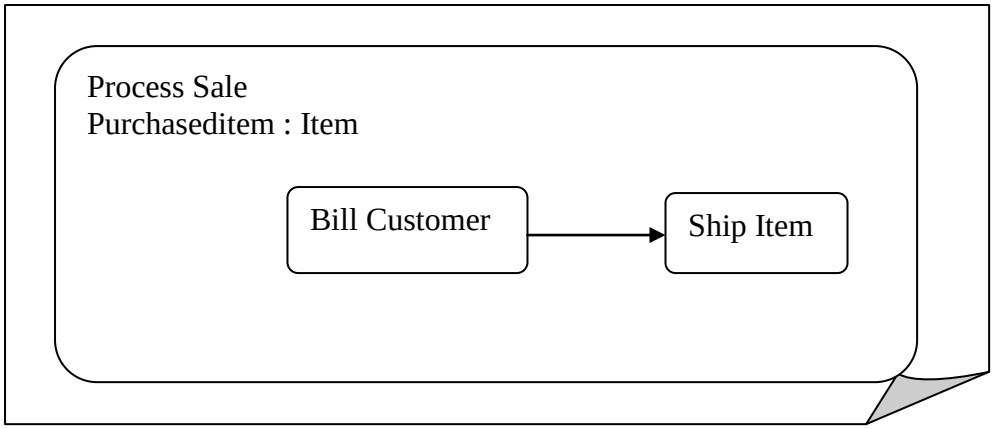
Aktivitas igambarkan dengan persegi panjang tumpul. Namanya ditulis dikiri atas. Parameter yang terlibat dalam aktivitas ditulis dibawahnya.



Gambar II.5. Aktivitas sederhana tanpa rincian

Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 145)

Detil aktivitas dapat dimasukkan didalam kotak. Aksi diperlihatkan denagn symbol yang sama dengan aktivitas dan namanya diletakkan di dalam persegi panjang.



Gambar II.6. Aktivitas dengan detail sederhana.

Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 145)

II.6. ERD (*Entity Relationship Diagram*)

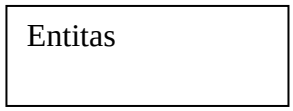
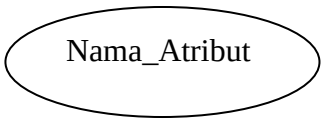
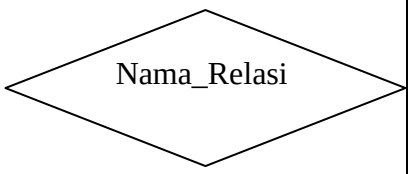
ERD (*Entity Relationship Diagram*) data model didasarkan pada persepsi terhadap dunia nyata yang tersusun atas kumpulan objek-objek dasar yang disebut entitas dan hubungan antar objek. Entitas adalah sesuatu atau objek dalam dunia nyata yang dapat dibedakan dari objek lain.

Sebagai tambahan, model ER menyajikan pula batasan isi basis dta harus menyesuiakan dengan batasan. Salah satu batasan yang penting adalah pemetaan kardinitas (*mapping cardinalities*), yang menggambarkan jumlah.

II.7.1. Simbol-simbol ERD (*Entity Relationship Diagram*)

Adapun simbol-sismbol ERD (*Entity Relationship Diagram*) ditunjukkan pada table II.1.

Tabel : II.1. simbol-sismbol ERD (*Entity Relationship Diagram*)

Simbol	Deskripsi
Entitas / entity 	Entitas merupakan data inti yang akan disimpan; bakal table pada basis data
Atribut 	Fiel atau kolom data yang perlu disimpan dalam suatu entitas
Relasi 	Relasi yang menghubungkan antara entitas; relasi biasanya diawali dengan kata kerja
Asosiasi / Association **1** **0..***	Penghubung antar relasi dan entitas diman kedua ujungnya memiliki multiplicity kemungkinan jumlah pemakaian

Sumber : (Janner Si ; 2010 : 59-60)

II.7. Notepad

Notepad adalah sebuah penyunting teks dan penyunting kode sumber yang berjalan di sistem operasi windows. Notepad menggunakan komponen Scintilla untuk dapat menampilkan dan menyunting teks dan berkas kode sumber berbagai

bahasa pemograman. Notepad didistribusikan sebagai perangkat lunak bebas. Proyek ini dilayani oleh Sourceforge.net dengan telah di unduh lebih dari 27 juta kali dan dua kali memenangkan penghargaan SourceForge Community Choice Award for Best develover Tool.

Perangkat lunak computer ini memiliki kelebihan pada peningkatan kemampuan sebuah program texs editor. Lebih dari sekedar program notpad windows. Notepad bias mengenal tag dan kode dalam berbagai bahasa pemograman. Fitur pencarian tingkat lanjut dan pengeditan teks yang tersedia juga cukup ampuh, sangat membantu tugas seorang programmer atau developer dalam menyelesaikan skrip kode programnya.

(Bunafit Nugroho ; 2009 : 2)

II.8. Pengertian Penyakit Lupus

Penyakit lupus adalah peradangan kronis yang terjadi ketika sistem imun tubuh menyerang organ dan jaringan tubuh. Peradangan yang disebabkan oleh lupus dapat berefek pada berbagai sistem di dalam tubuh, antara lain sendi, kulit, ginjal, sel darah, jantung dan paru-paru. Lupus lebih sering terjadi pada wanita, meskipun tidak jelas alasannya. Ada empat jenis lupus -systemic lupus erythematosus, discoid lupus erythematosus, drug-induced lupus erythematosus dan neonatal lupus. Diantaranya, systemic lupus erythematosus adalah yang paling umum dan paling serius. (Keneth.J.Leveno, Dkk ; 2009 : 59)

II.8.1. Penyebab Penyakit Lupus

Lupus adalah penyakit autoimun yang muncul ketika tubuh terkena zat asing tertentu. Seperti bakteri dan virus. Kemudian sistem imun tersebut juga menyerang jaringan tubuh yang sehat. Hal ini menyebabkan peradangan dan kerusakan berbagai bagian tubuh, antara lain sendi, kulit, ginjal, jantung, paru-paru, pembuluh darah dan otak.

Dokter tidak mengetahui apa yang menyebabkanpenyakit ini. Lupus seperti merupakan kombinasi factor genetic dan lingkungan. Banyak dari mereka dengan kecendrungan turunan mengalami lupus ketika mereka terkena sesuatu di dalam lingkungan yang dapat memicu lupus, seperti obat atau virus. (Keneth.J.Leveno, Dkk ; 2009 : 60)

II.8.2. Ciri-ciri Penyakit Lupus

Salah satu ciri-ciri penyakit lupus secara umum adalah perasaan lelah dan letih yang berkepanjangan yang tak dapat diatasi hanya dengan beristirahat saja. Kendati telah tidur dan beristirahat cukup, penderita akan tetap merasa kelelahan pada pagi hari, disertai juga dengan linu, pegal-pegal dan sakit nyerti pada otot juga sendi.

Ciri-ciri penyakit lupus lainnya adalah terjadi kerontokan pada rambut si penderita lupus. Selain kedua gejala tersebut, gejala paling umum yang paling mudah untuk memastikan bahwa seseorang terkena penyakit lupus adalah munculnya bercak – bercak merah berpola yang menyerupai ruam / penyakit kulit pada bagian tubuh, tangan atau pipi wajah.

Bentuk pola ruam yang kerap muncul biasanya berpola seperti kupu – kupu (lihat gambar) dengan bentuk memanjang dari pipi kiri melintas hingga ke pipi kanan. Akan tetapi ruam berpola kupu – kupu (malar rash) ini tak mutlak selalu muncul pada semua penderita. Ada pula yang membentuk lingkaran atau areal pada bagian-bagian lain seperti leher, dada, punggung atau tangan. (Keneth.J.Leveno, Dkk ; 2009 : 61)