BAB II
TINJAUAN PUSTAKA

2.1 Teori Dasar

Menurut Sudaryono (2015:13) teori adalah generalisasi atau kumpulan generalisasi yang dapat digunakan untuk menjelaskan berbagai fenomena secara sistematis. Generalisasi adalah proses penalaran yang bertolak dari fenomena individual menuju kesimpulan umum.

2.1.1 Kecerdasan Buatan

Menurut Sutojo, dkk. (2011:1) kecerdasan buatan yang dimaksud di sini merujuk pada mesin yang mampu berpikir, menimbang tindakan yang akan diambil, dan mampu mengambil keputusan seperti yang dilakukan oleh manusia.

Herbert Alexander Simon (June 15, 1916-February 9, 2001): "Kecerdasan buatan (*Artificial Intelligence)* merupakan kawasan penelitian, aplikasi, dan instruksi yang terkait dengan pemrograman komputer untuk melakukan sesuatu hal yang dalam pandangan manusia adalah cerdas" (Sutojo, dkk., 2011:2).

Bidang - bidang yang menggunakan kecerdasan buatan antara lain sebagai berikut.

1. Jaringan Saraf Tiruan (JST)

   Menurut Sutojo, dkk. (2011: 396), jaringan saraf tiruan adalah paradigma pengolahan informasi yang terinspirasi oleh sistem saraf secara biologis,

seperti proses informasi pada otak manusia. Cara kerja JST seperti cara kerja manusia, yaitu belajar melalui contoh.

2. Sistem Pakar (*Expert System*)

Menurut Sutojo, dkk. (2011: 204) sistem pakar adalah sebuah sistem yang menggunakan pengetahuan manusia di mana pengetahuan tersebut dimasukkan ke dalam sebuah computer dan kemudian digunakan untuk menyelesaikan masalah – masalah yang biasanya membutuhkan kepakaran. Bebarapa metode dalam sistem pakar adalah sebagai berikut.

a. *Forward Chaining*

Forward Chaining adalah teknik pencarian yang dimulai dengan fakta yang diketahui, kemudian mencoccokkan fakta-fakta tersebut dengan bagian *IF* dari *rules IF-THEN*.

b. *Backward Chaining*

Backward Chaining adalah metode inferensi yang bekerja mundur ke arah kondisi awal. Proses diawali dari *Goal* (yang berada dibagian *THEN* dari *rule IF-THEN*), kemudian pencarian mulai dijalankan untuk mencocokkan apakah fakta-fakta yang ada cocok dengan premis-premis di bagian *IF*.

3. Logika *Fuzzy* (*Fuzzy Logic*)

Logika *fuzzy* adalah metodologi sistem kontrol pemecahan masalah, yang cocok untuk diimplementasikan pada sistem, mulai dari sistem yang sederhana, sistem kecil, *embedded system*, jaringan PC, *multi-channel* atau *workstation* berbasis akuisisi data, dan sistem kontrol (Sutojo, dkk., 2011: 276).

Menurut Sutojo, dkk. (2011: 233) terdapat beberapa metode dalam logika fuzzy antara lain:

a. Metode *Tsukamoto*

Secara umum bentuk model *fuzzy Tsukamoto* adalah:

If (X IS A) and (Y IS B) Then (Z IS C)

Di mana A, B dan C adalah himpunan *fuzzy*.

b. Metode *Mamdani*

Metode Mamdani paling sering digunakan dalam aplikasi-aplikasi karena strukturnya yang sederhana, yaitu menggunakan operasi *MIN-MAX* atau *MAX-PRODUCT*.

c. Metode *Sugeno*

Dalam metode *Sugeno*, output sistem berupa konstanta atau persamaan linear. Secara umum bentuk model *fuzzy Sugeno* adalah:

IF (x_1 is A_1)[][]•…..•(x_N is A_N) THEN $z = f(x,y)$

2.1.2 Sistem Pakar

Menurut Sutojo, dkk. (2011:159) Sistem pakar merupakan cabang dari *Artificial Intelligence* (AI) yang cukup tua karena sistem ini mulai dikembangkan pada pertengahan 1960. Sistem pakar yang muncul pertama kali adalah *General-purpose problem solver* (GPS) yang dikembangkan oleh Newell dan Simon. Sampai saat ini sudah banyak sistem pakar yang dibuat, seperti MYCIN untuk diagnosis penyakit, DENDRAL untuk mengidentifikasi struktur molekul campuran yang tak dikenal, XCON dan XSEL untuk membantu kongfigurasi

sistem komputer besar, SOPHIE untuk analisis sirkuit elektronik, Prospector digunakan di bidang geologi untuk membantu mencari dan menemukan deposit, FOLIO digunakan untuk membantu memberikan keputusan bagi seorang manager dalam stok dan investasi, DELTA dipakai untuk pemeliharaan lokomotif listrik diesel dan sebagainya.

Menurut Hayadi (2016:1) Sistem pakar atau *Expert System* biasa disebut juga dengan *Knowledge Based System* yaitu suatu aplikasi komputer yang ditujukan untuk membantu pengambilan keputusan atau pemecahan persoalan dalam bidang yang spesifik. Sistem ini bekerja dengan menggunakan pengetahuan dan metode analisis yang telah didefinisikan terlebih dahulu oleh pakar yang sesuai dengan bidang keahliannya.

Menurut Jackson (1999, p3) "Sistem pakar adalah program komputer yang merepresentasikan dan melakukan penalaran dengan pengetahuan beberapa pakar untuk memecahkan masalah atau memberikan saran" (Sutojo, dkk., 2011:160).

Konsep dasar sistem pakar meliputi enam hal berikut (Sutojo, dkk., 2011:163).

1 . Kepakaran (*Expertise*)

Kepakaran merupakan suatu pengetahuan yang diperoleh dari pelatihan, membaca, dan pengalaman. kepakaran inilah yang memungkinkan para ahli dapat mengambil keputusan lebih cepat dan lebih baik daripada seseorang yang bukan pakar. Kepakaran itu sendiri meliputi pengetahuan tentang

a. Fakta-fakta tentang bidang permasalahan tertentu

b. Teori-teori tentang bidang permasalahan tertentu

c. Aturan *heuristic* yang harus dikerjakan dalam suatu situasi tertentu

d. Aturan-aturan dan prosedur-prosedur menurut bidang permasalahan umumnya

e. Strategi global untuk memecahkan permasalahan

f. Pengetahuan tentang pengetahuan (*meta knowledge*).

2. Pakar (*Expert*)

Pakar adalah seseorang yang mempunyai pengetahuan, pengalaman, dan metode khusus, serta mampu menerapkan untuk memecahkan masalah atau memberi nasihat. Seorang pakar harus mampu menjelaskan dan mempelajari hal-hal baru yang berkaitan topik permasalahan, jika perlu harus mampu menyusun kembali pengetahuan-pengetahuan yang didapatkan dan dapat memecahkan aturan-aturan serta menentukan relevansi kepakarannya. Jadi, seorang pakar harus mampu melakukan kegiatan-kegiatan berikut:

a. Mengenali dan memformulasikan masalah.

b. Memecahlan masalah dengan cepat dan tegas.

c. Menerangkan pemecahannya.

d. Belajar dari pengalaman.

e. Merekstrukturisasi pengetahuan.

f. Memecahkan aturan-aturan.

g. Menentukan relevansi

3. Pemindahan Kepakaran (*Transfering Expertise*)

Tujuan dari sistem pakar adalah memindahkan kepakaran dari seorang pakar kedalam komputer, kemudian ditransfer kepada orang lain yang bukan pakar. Proses ini melibatkan empat kegiatan, yaitu:

a. Akusisi pengetahuan (dari pakar atau sumber lain)

b. Representasi pengetahuan (pada komputer)

c. Inferensi pengetahuan

d. Pemindahan pengetahuan ke pengguna

4. Inferensi (*Inferencing*)

Inferensi adalah sebuah prosedur (program) yang mempunyai kemampuan dalam melakukan penalaran. Inferensi ditampilkan pada suatu komponen yang disebut mesin inferensi yang mencakup prosedur-prosedur mengenai pemecahan masalah. Semua pengetahuan yang dimiliki oleh seorang pakar disimpan pada basis pengetahuan oleh sistem pakar. Tugas mesin inferensi adalah mengambil kesimpulan berdasarkan basis pengetahuan yang dimilikinya.

5. Aturan-aturan (*Rule*)

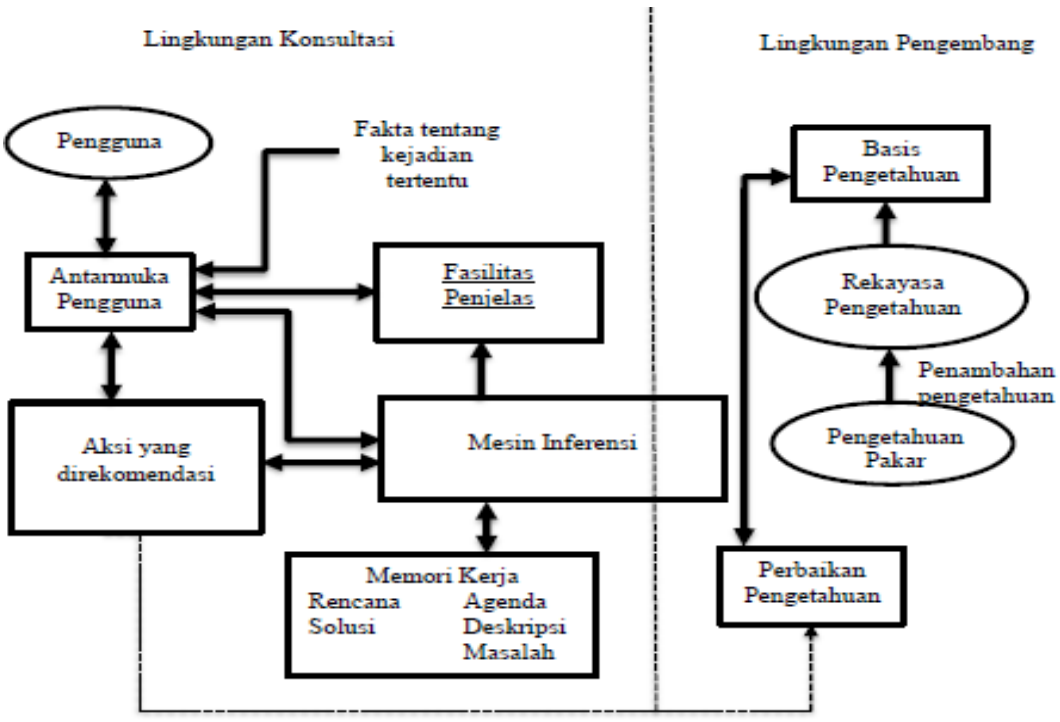
Kebanyakan *software* sistem pakar komersial adalah sistem yang berbasis *rule* (*rule based systems*), yaitu pengetahuan disimpan terutama dalam bentuk rule, sebagai prosedur-prosedur pemecahan masalah.

6. Kemampuan menjelaskan (*Explanation Capability*)

Fasilitas lain dari sistem pakar adalah kemampuannya untuk menjelaskan saran atau rekomendasi yang diberikan oleh sistem pakar. Penjelasan dilakukan

dalam subsistem yang disebut subsistem pejelasan (*explanation*). Bagian dari sistem ini memungkinkan sistem untuk memeriksa penalaran yang dibuatnya sendiri dan menjelaskan operasi operasinya.

Ada dua bagian penting dari sistem pakar, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan digunakan oleh pembuat sistem pakar untuk membangun komponen-komponenya dan memasukkan pengetahuan ke dalam *knowledge base* (basis pengetahuan). Lingkungan konsultasi digunakan oleh pengguna untuk berkonsultasi sehingga pengguna mendapatkan pengetahuan dan nasihat dari sistem pakar layaknya berkonsultasi dengan sistem pakar (Sutojo, dkk., 2011:166). Gambar berikut menunjukan komponen-komponen yang penting dalam sebuah sistem pakar:



Gambar 2.1 Komponen-komponen penting dalam sebuah sistem pakar
(Sumber: Sutojo, dkk., 2011:167)

Keterangan :

1. Akusisi pengetahuan: digunakan untuk memasukkan pengetahuan dari seorang pakar dengan cara merekayasa pengetahuan agar bisa diproses oleh komputer dan menaruhnya ke dalam basis pengetahuan dengan format tertentu (dalam bentuk representasi pengetahuan). Sumber-sumber pengetahuan bisa diperoleh dari pakar, buku, dokumen multimedia, basis data, laporan riset khusus, dan informasi yang terdapat di *web*.
2. Basis pengetahuan (*knowledge base*): mengandung pengetahuan yang diperlukan untuk memahami, merumuskan dan menyelesaikan malasah. Basis pengetahuan terdiri dari dua elemen dasar, yaitu:
   a. Fakta, misalnya situasi, kondisi atau permasalahan yang ada.
   b. Aturan (*rule*), untuk mengarahkan penggunaan pengetahuan dalam memecahkan masalah.
3. Mesin Inferensi (*Inference Engine*): adalah sebuah program yang berfungsi untuk memandu proses penalaran terhadap suatu kondisi berdasarkan pada basis pengetahuan yang ada, memanipulasi dan mengarahkan kaidah, model dan fakta yang disimpan dalam basis pengetahuan untuk mencapai solusi atau kesimpulan. Dalam prosesnya, mesin inferensi menggunakan strategi pengendalian, yaitu strategi yang berfungsi sebagai panduan arah dalam melakukan proses penalaran. Ada tiga teknik pengendalian yang digunakan, yaitu *forward chaining*, *backward chaining*, dan gabungan dari kedua teknik tersebut.

4. Daerah kerja (*blackboard*): untuk merekam hasil sementara yang akan dijadikan sebagai keputusan dan untuk menjelaskan sebuah masalah yang sedang terjadi, sistem pakar membutuhkan *blackboard*, yaitu area pada memori yang berfungsi sebagai basis data. Tiga tipe keputusan yang dapat direkam pada *blackboard* yaitu:

   a. Rencana : bagaimana menghadapi masalah

   b. Agenda : aksi-aksi potensial yang sedang menunggu untuk dieksekusi

   c. Solusi : calon aksi yang akan dibangkitkan

5. Antarmuka pemakai (*user interface*): digunakan sebagai media komunikasi antara pengguna dan sistem pakar. Komunikasi ini paling bagus bila disajikan dalam bahasa alami (*natural language*) dan dilengkapi dengan grafik, menu dan formulir elektronik. Pada bagian ini akan terjadi dialog antara sistem pakar dan pengguna.

6. Sistem perbaikan pengetahuan (*knowledge refining system*): Subsistem penjelas berfungsi memberi penjelasan kepada pengguna, bagaimana suatu kesimpulan dapat diambil. Kemampuan seperti ini sangat penting bagi pengguna untuk mengetahui proses pemindahan keahlian pakar maupun dalam pemecahan masalah.

7. Sistem perbaikan pengetahuan (*knowledge refining system*): Kemampuan memperbaiki pengetahuan (*knowledge refining system*) dari seorang pakar diperlukan untuk menganalisis pengetahuan belajar dari kesalahan masa lalu, kemudian memperbaiki pengetahuannya sehingga dapat digunakan di masa mendatang. Kemampuan evaluasi diri seperti itu diperlukan oleh program

agar dapat menganalisis alasan-alasan kesuksesan dan kegagalannya dalam mengambil kesimpulan. Cara ini dapat menghasilkan basis pengetahuan yang lebih baik dan penalaran yang lebih efektif.

8. Pengguna (*User*): pada umumnya pengguna sistem pakar bukanlah seorang pakar yang membutuhkan solusi, saran, atau pelatihan (*training*) dan berbagai permasalahan yang ada.

2.1.3 Forward Chaining

Menurut Sutojo, dkk. (2011:171) Forward Chaining adalah teknik pencarian yang dimulai dengan fakta yang diketahui, kemudian mencocokkan fakta-fakta tersebut dengan bagian IF dari rules IF-THEN. Bila ada fakta yang cocok dengan bagian IF, maka rule tersebut dieksekusi. Bila sebuah rule dieksekusi, maka sebuah fakta baru (bagian THEN) ditambahkan ke dalam database. Setiap kali pencocokan, dimulai dari rule teratas. Setiap rule hanya boleh dieksekusi sekali saja. Proses pencocokan berhenti bila tidak ada lagi rule yang bisa dieksekusi.

Menurut Russel S,Norvig P, 2003 Metode Forward Chaining adalah metode pencarian atau teknik pelacakan ke depan yang dimulai dengan informasi yang ada dan penggabungan rule untuk menghasilkan suatu kesimpulan atau tujuan (Hayadi 2016:9).

Menurut Hayadi (2016:11) langkah-langkah yang harus dilakukan dalam membuat sistem forward *chaining* berbasis aturan, yaitu:

1. Pendefinisian Masalah

Tahap ini meliputi pemilihan domain masalah dan akusisi pengetahuan.

2. Pendefinisian Data Input

Sistem *forward chaining* memerlukan data awal untuk melakukan inferensi.

3. Pendefinisian Struktur Pengendalian Data

Aplikasi yang kompleks memerlukan premis tambahan untuk membantu mengendalikan pengaktifan suatu aturan.

4. Penulisan Kode Awal

Tahap ini berguna untuk menentukan apakah sistem telah menangkap domain pengetahuan secara efektif dalam struktur aturan yang baik.

5. Pengujian Sistem

Pengujian sistem dilakukan dengan beberapa aturan untuk menguji sejauh mana sistem berjalan dengan benar.

6. Perancangan Antarmuka

Antarmuka adalah suatu komponen penting dari suatu sistem. Perancangan antarmuka dibuat bersama-sama dengan pembuatan basis pengetahuan.

7. Pengembangan Sistem

Pengembangan sistem meliputi penambahan antarmuka dan pengetahuan sesuai dengan prototipe sistem.

8. Evaluasi Sistem

Pada tahap ini dilakukan pengujian sistem dengan masalah yang sebenarnya. Jika sistem belum berjalan dengan baik maka akan dilakukan pengembangan kembali.

2.2 Variabel

Menurut Sugiyono (2012:38) variabel penelitian adalah suatu atribut atau sifat atau nilai dari orang, obyek atau kegiatan yang mempunyai variasi tertentu yang diterapkan oleh peneliti untuk dipelajari dan kemudian ditarik kesimpulannya.

Menurut Sudaryono (2015:16) variabel penelitian pada dasarnya adalah segala sesuatu yang berbentuk apa saja yang ditetapkan oleh peneliti untuk dipelajari sehingga diperoleh informasi dan kesimpulannya.

Berdasarkan Keputusan Menteri Kesehatan Republik Indonesia Nomor : 1995/Menkes/SK/XII/2010 tentang Standar Antropometri Penilaian Status Gizi Anak, indeks yang digunakan sebagai penentu kategori status gizi sebagai berikut:

2.2.1 Umur

Umur dihitung dalam bulan penuh. Contoh : umur 2 bulan 29 hari dihitung sebagai umur 2 bulan (Kemenkes, 2010:3).

2.2.2 *Length/height-for-age*

Ukuran panjang badan digunakan untuk anak umur 0 sampai 24 bulan yang diukur telentang. Bila anak umur 0 sampai 24 bulan diukur berdiri, maka hasil pengukurannya dikoreksi dengan menambahkan 0.7 cm (Kemenkes, 2010:3).

Standar untuk pertumbuhan linear memiliki bagian berdasarkan panjang (panjang-untuk-usia, 0 sampai 24 bulan) dan satu lagi di ketinggian (tinggi-untuk-

umur, 2 sampai 5 tahun). Dua bagian dibangun menggunakan model yang sama tetapi kurva akhir mencerminkan perbedaan rata-rata antara panjang berbaring dan berdiri tinggi. Dengan desain, anak-anak antara 18 dan 30 bulan dalam komponen *cross-sectional* dari MGRS (*Multicentre Growth Reference Study)* mengambil pengukuran panjang dan tinggi. Rata-rata perbedaan antara dua pengukuran pada 1625 anak-anak adalah 0,73 cm. Untuk cocok dengan model tunggal untuk rentang usia utuh, 0,7 cm itu ditambahkan ke nilai ketinggian penampang sebelum penggabungan dengan data panjang. Setelah model disesuaikan, kurva median dialihkan kembali ke bawah 0,7 cm untuk usia di atas dua tahun, dan koefisien kurva variasi disesuaikan dengan nilai-nilai median baru untuk membangun kurva pertumbuhan tinggi-untuk-usia. Daya transformasi pada usia diterapkan untuk meregangkan skala usia untuk masing-masing jenis kelamin sebelum disesuaikan dengan splines kubik untuk menghasilkan kurva pertumbuhan masing-masing. Kurva anak laki-laki diperlukan model dengan tingkat kebebasan yang lebih untuk menyesuaikan kedua median dan koefisien kurva variasi. Data untuk kedua jenis kelamin mengikuti distribusi normal (Onis, *et al.*, 2008:xviii).

Pengukuran tinggi anak dibagi menjadi dua, yaitu Panjang Badan (PB) dan Tinggi Badan (TB). Panjang Badan digunakan untuk anak umut 0 sampai 24 bulan yang diukur telentang. Bila anak umur 0 sampai 24 bulan diukur berdiri, maka hasil pengukurannya dikoreksi dengan menambahkan 0,7 cm. Ukuran Tinggi badan (TB) digunakan untuk anak umur di atas 24 bulan yang diukur berdiri. Bila anak umur 24 bulan diukur telentang, maka hasil pengukurannya dikoreksi dengan mengurangkan 0,7 cm (Kemenkes, 2010:3).

2.2.3 *Weight-for-age*

Bobot dari sampel longitudinal dan *cross-sectional* digabung tanpa penyesuaian dan model tunggal disesuaikan untuk menghasilkan satu set kurva yang berdasarkan jenis kelamin masing-masing terhadap standar berat-untuk-usia. Daya transformasi yang sama diterapkan pada usia anak laki-laki dan perempuan sebelum disesuaikan dengan model konstruksi kurva. Data berat badan untuk kedua jenis kelamin yang miring, sehingga dalam menentukan model, parameter yang terkait dengan kemiringan dipasang di samping median dan koefisien perkiraan variasi. Dalam pemodelan kemiringan kurva gadis-gadis diperlukan tingkat kebebasan untuk menyesuaikan kurva untuk parameter ini (Onis, *et al*., 2008:xviii).

2.3 Software Pendukung

2.3.1 Java

Java, dalam ilmu computer, merupakan bahasa pemrograman berorientasi objek yang diperkenalkan pada tahun 1995 oleh Sun Microsystems, Inc., yang saat Java diciptakan, dipimpin oleh James Gosling. Java memfasilitasi penyebaran baik data maupun program aplikasi kecil, yang dinamakan *applet*, lewat internet. Aplikasi-aplikasi Java tidak berinteraksi dengan CPU (*Central Processing Unit)* atau sistem operasi computer yang digunakan sehingga ia bisa bersifat mandiri

terhadap platform komputer, baik platform perangkat lunak maupun perangkat keras (Nugroho, 2008:4).

Keunggulan dari Java adalah dapat berjalan di banyak platform perangkat keras dan perangkat lunak sehingga pengembang aplikasi dan pemrogram dapat menuliskan program dengan Java (sekali dan hanya sekali saja) kemudian dengan relative mudah bisa menjalankannya di mana saja; di platform sistem operasi apapun serta di kebanyakan computer yang ada saat ini. Istilah populernya adalah "*Write Once, Run Anywhere"* (Nugroho, 2008:4).

Kepopuleran bahasa pemrograman Java juga dapat dimengerti karena Java memiliki bentuk dan sifat yang dimiliki bahasa lain yang terkemuka, seperti C, C++, Objective-C, Smalltalk serta Common Lisp. Dari Smalltalk, Java meminjam konsep model ekstensibilitas saat eksekusi, menajemen memori dinamis serta pengeksekusian program secara bersamaan (*multi-threading)*. Dari C/C++, Java meminjam konsep-konsep operator yang sama sehingga mempermudah pemrogram C/C++ beralih ke Java (Nugroho, 2008:5).

Dalam bahasa pemrograman Java, kelas serta metode merupakan blok bangunan dasar. Jadi dengan bahasa pemrograman Java kita dapat menerapkan konsep pemrograman berorientasi objek dengan tuntas. Apalagi Java dilengkapi dengan pustaka kelas (*library)* yang beragam. Dalam Java, alokasi memori dinamis diatur secara otomatis oleh bagian kompiler Java yang dinamakan JVM (*Java Virtual Machine)* sehingga pemrograman agak dimudahkan. Keuntungan lain, karena ukuran JVM ini relative kecil maka kita tidak akan terlalu dipusingkan oleh batasan memori komputer yang kita miliki (Nugroho, 2008:6-7).

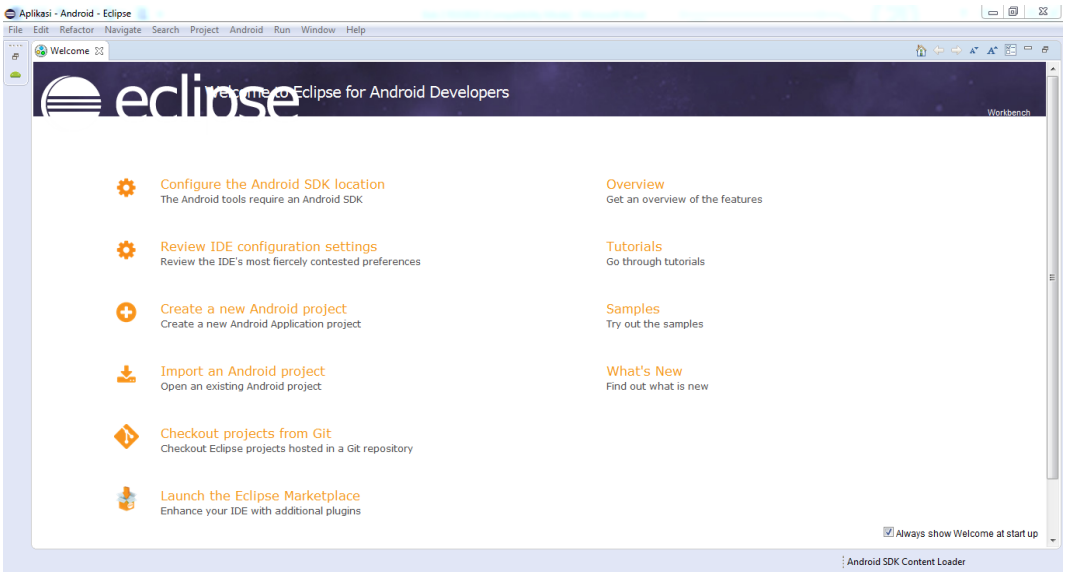
2.3.2 Eclipse

EMS (2015:35) mengatakana untuk IDE (*Integrated Development Environment)* yang dipakai dengan pemrograman Android, baik berbasis Java atau pemaketan dengan HTML dengan Cordova, Anda bisa menggunakan Eclipse. Sebenarnya *tool* yang bisa dipakai tidak hanya Eclipse, tapi ada juga *tool* lainnya seperti Android Studio. Tapi yang paling terkenal dan mudah konfigurasinya adalah Eclipse.

Menurut Safaat H (2015:4-5) Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (*platform-independent*). Berikut ini adalah sifat dari Eclipse:

1. **Multi-platform**: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. **Multi-language**: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.
3. **Multi-role**: Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.

Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in.*



Gambar 2.2 Tampilan Utama Aplikasi Eclipse

(Sumber: Data Penelitian, 2016)

2.3.3 Android

Menurut EMS (2015:1) Android secara sederhana bisa diartikan sebagai sebuah software yang digunakan pada perangkat mobile yang mencakup sistem operasi, *middleware*, dan aplikasi kunci yang dirilis oleh Google. Sehingga Android mencakup keseluruhan sebuah aplikasi, mulai dari sistem operasi sampai pada pengembangan aplikasi itu sendiri. Pengembangan aplikasi pada platform

Android ini menggunakan dasar bahasa pemrograman Java. Tapi secara sempit, Android biasanya mengacu pada sistem operasinya saja .

Platform pengembangan aplikasi Android ini bersifat *open-source* atau terbuka, sehingga Anda dapat mengembangkan kemampuan untuk membangun aplikasi yang kaya dan inovatif. Bahkan seorang pengembang (*developer*) Android dapat membuat aplikasi yang bervariasi (EMS, 2015:2).

Android Lollipop (Hilmi Masruri & Creativity, 2015:20) adalah versi stabil terbaru dari sistem operasi Android yang dikembangkan oleh Google, yang pada saat ini mencakup versi antara 5.0 dan 5.1. Fitur yang di tawrkan Android Lollipop 5.0, antara lain :

1. Desain Material

Desain antarmuka yang berani, penuh warna, dan responsif untuk pengalaman yang konsisten dan intuitif di semua perangkat Anda. Responsif, gerak alami, pencahayaan dan bayangan realistis, dan elemen visual yang sudah familier memudahkan Anda untuk menavigasi perangkat. Warna baru yang terang, tipografi, dan gambar dari tepi ke tepi membantu memfokuskan perhatian Anda.

2. Notifikasi

Cara baru untuk mengontrol waktu dan cara menerima pesan, hanya pada saat yang Anda inginkan. Lihat dan balas pesan langsung dari layar kunci. Anda juga dapat menyembunyikan konten sensitif untuk notifikasi tersebut. Untuk meminimalkan gangguan, aktifkan mode Prioritas melalui tombol volume perangkat agar hanya orang dan notifikasi tertentu yang dapat masuk. Atau

jadwalkan masa tidak aktif secara berulang, misalnya dari jam 10 malam hingga 8 pagi, agar notifikasi Prioritas saja yang dapat masuk. Dengan Lollipop, panggilan telepon masuk tidak akan mengganggu film yang sedang Anda tonton atau game yang Anda mainkan. Anda dapat memilih untuk menjawab panggilan atau terus melakukan aktivitas. Kontrol notifikasi aplikasi, sembunyikan konten sensitif, dan prioritaskan atau nonaktifkan notifikasi aplikasi. Peringkat notifikasi yang lebih cerdas berdasarkan pengirimnya dan jenis komunikasi. Lihat semua notifikasi di satu tempat dengan mengetuk bagian atas layar.

3. Baterai

Daya untuk aktivitas sepanjang hari. Fitur penghemat baterai yang memperpanjang masa pakai perangkat hingga 90 menit. Perkiraan sisa waktu sampai baterai terisi penuh ditampilkan saat perangkat dicolokkan. Perkiraan sisa waktu sebelum mengisi daya perangkat kini dapat dilihat di setelan baterai.

4. Keamanan

Jaga agar file Anda tetap aman. Penerapan SELinux pada semua aplikasi untuk perlindungan yang lebih baik terhadap kerentanan dan perangkat lunak perusak. Gunakan Android Smart Lock untuk mengamankan ponsel atau tablet dengan menyandingkannya dengan perangkat tepercaya seperti Android Wear atau bahkan mobil.

5. Berbagi Perangkat

Berbagi dengan keluarga dan teman jadi lebih fleksibel. Fitur banyak pengguna untuk ponsel. Jika kelupaan ponsel, Anda tetap dapat menelepon teman

(atau mengakses pesan, foto, dll) dengan masuk ke ponsel Android lain yang menjalankan Lollipop. Fitur ini juga cocok untuk anggota keluarga yang ingin menggunakan ponsel bersama-sama tanpa berbagi informasi pribadi. Melalui fitur pengguna tamu di ponsel dan tablet, Anda dapat meminjamkan perangkat dan menjaga informasi pribadi tetap aman. Pin ke layar: pasang pin agar pengguna lain hanya dapat mengakses konten tertentu tanpa mengotak-atik item lainnya.

6. Setelan Cepat Baru

Kontrol baru yang praktis seperti senter, hotspot, pemutaran layar, dan layar transmisi.

7. Konektivitas

Penyempurnaan handoff jaringan yang mengakibatkan terbatasnya gangguan konektivitas. Misalnya, Anda dapat melanjutkan obrolan video atau panggilan VoIP tanpa gangguan saat keluar rumah dan beralih dari Wi-Fi rumah ke seluler.

8. Waktu Proses dan Performa

ART (*Android Run Time)*, waktu proses Android yang benar-benar baru, menyempurnakan performa dan daya respons aplikasi. Memadatkan aplikasi dan layanan latar belakang agar Anda dapat melakukan lebih banyak hal sekaligus. Dukungan untuk perangkat 64 bit, seperti Nexus 9, menghadirkan CPU kelas desktop ke Android.

9. Media

Masukan audio latensi rendah memastikan aplikasi musik dan komunikasi dengan persyaratan penundaan yang ketat memberikan pengalaman langsung yang menakjubkan. Dengan campuran aliran audio multi-saluran, aplikasi audio profesional kini dapat mencampur hingga delapan saluran, termasuk saluran 5.1 dan 7.1. Dengan dukungan Audio USB, Anda dapat mencolokkan mikrofon atau pengeras suara USB, dan banyak perangkat audio USB lainnya seperti amplifier dan mixer ke perangkat Android. Teknologi video yang canggih dengan dukungan HEVC untuk pemutaran video UHD 4K, video yang disalurkan untuk pemutaran video berkualitas tinggi di Android TV, dan penyempurnaan dukungan HLS untuk streaming.

10. Aksesibilitas

Penyempurnaan kemampuan untuk pengguna yang mengalami gangguan penglihatan dan buta warna. Meningkatkan kontras teks atau inversi warna untuk meningkatkan keterbacaan. Sesuaikan tampilan untuk meningkatkan diferensiasi warna.

Gambar 2.3 Logo Android Versi 5.1
(Sumber : https://goo.gl/images/tf1LrT)

2.3.4 *XAMPP (Apache PHP)*

XAMPP adalah sebuah paket *web server* yang mengandung *Apache* dan *PHP*, serta *MySQL*. *XAMPP* tersedia untuk multisystem operasi, seperti *Windows* dan *Linux* (EMS, 2016:11).

Gambar 2.4 Logo *XAMPP*
(Sumber : https://goo.gl/images/9UJ9Vk)

PHP adalah singkatan dari *PHP Hypertext Preprocessing*. Merupakan bahasa *scripting* untuk *web* dimana bisa membuat *web* dinamis dengan menyelipkan *script* kode-kode *HTML* yang merupakan bahasa *markup* standar untuk dunia *web*. *PHP* 5.0 memasukkan model pemrograman berorientasi objek ke dalam *PHP* untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek (EMS, 2016:1 - 4).

Gambar 2.5 Logo *PHP*
(Sumber : https://goo.gl/images/zSfClB)

Apache digunakan sebagai *web server* dari aplikasi yang akan kita buat. *Apache* berorientasi *web* sehingga sangat cocok untuk dunia *web*, kecepatan responnya tinggi. Komunikasi *PHP* dan *MySQL* sangat baik serta kompabilitasnya tinggi. Bisa dikostumisasi karena kode sumber keduanya terbuka (EMS, 2016:10 - 11).

Gambar 2.6 Logo *Apache*
(Sumber : https://goo.gl/images/3zjlDv)

2.3.5 MySQL

Menurut Raharjo (2011:3) *database* didefinisikan sebagai kumpulan data yang terintegrasi dan diatur sedemikian rupa sehingga data tersebut dapat dimanipulasi, diambil, dan dicari secara cepat.

Menurut Raharjo (2011:21) MySQL merupakan *software* RDBMS (atau *server database)* yang dapat mengelola *database* dengan sangat cepat, dapat menampung data dalam jumlah sangat besar, dapat diakses oleh banyak *user (multi-user),* dan dapat melakukan suatu proses secara sinkron atau berbarengan (*multi-threaded).*

Beberapa alasan menggunakan MySQL sebagai server database untuk aplikasi-aplikasi yang dikembangan (Raharjo, 2011:22) :

1 . Fleksibel

MySQL dapat digunakan untuk mengembangkan aplikasi *desktop* maupun aplikasi web dengan menggunakan teknologi yang bervariasi. Ini berarti bahwa MySQL memiliki fleksibilitas terhadap teknologi yang akan digunakan sebagai pengembang aplikasi, apakah itu PHP, JSP, Jav, Delphi, C++, maupun yang lainnya dengan cara menyediakan *plug-in* dan *driver* yang spesifik untuk masing-masing teknologi tersebut. Dalam database MySQL juga memiliki dukungan terhadap *stored procedure,* fungsi, *trigger, view,*SQL standar ANSI, dll, yang akan mempermudah dan mempercepat proses pengembangan aplikasi.

2. Performa Tinggi

MySQL memiliki mesin *query* dengan performa tinggi, dengan demikian proses transaksional dapat dilakukan dengan sangat cepat. Hal ini terbukti dengan digunakannya MySQL sebagai *database* dari beberapa aplikasi *web* yang memiliki *traffic* (lalu lintas) sangat tinggi.

3. Lintas Platform

MySQL dapat digunakan pada *platform* atau lingkungan (dalam hal ini Sistem Operasi) yang beragam, bisa Microsoft Windows, Linux, atau UNIX. Ini menyebabkan proses migrasi data (bila dibutuhkan) antarsistem operasi dapat dilakukan secara lebih mudah.

4. Gratis

MySQL dapat digunakan secara gratis.

5. Proteksi Data yang Handal

Perlindungan terhadap keamanan data merupakan hal nomor satu yang dilakukan oleh para professional di bidang *database*. MySQL menyediakan mekanisme yang *powerfull* untuk menangani hal tesebut, yaitu dengan menyediakan fasilitas menajemen *user*, enkripsi data, dan lain sebagainya.

6. Komunitas Luas

Karena penggunanya banyak maka MySQL memiliki komunitas yang luas. Hal ini berguna jika kita menemui suatu permasalahan dalam proses pengolahan data menggunakan MySQL.

Menurut Hendry (2015: 7) MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional yang didistribusikan secara gratis di bawah lisensi GPL (General Public License). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya; SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basis data, terutama untuk pemilihan atau seleksi dan pemasukkan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

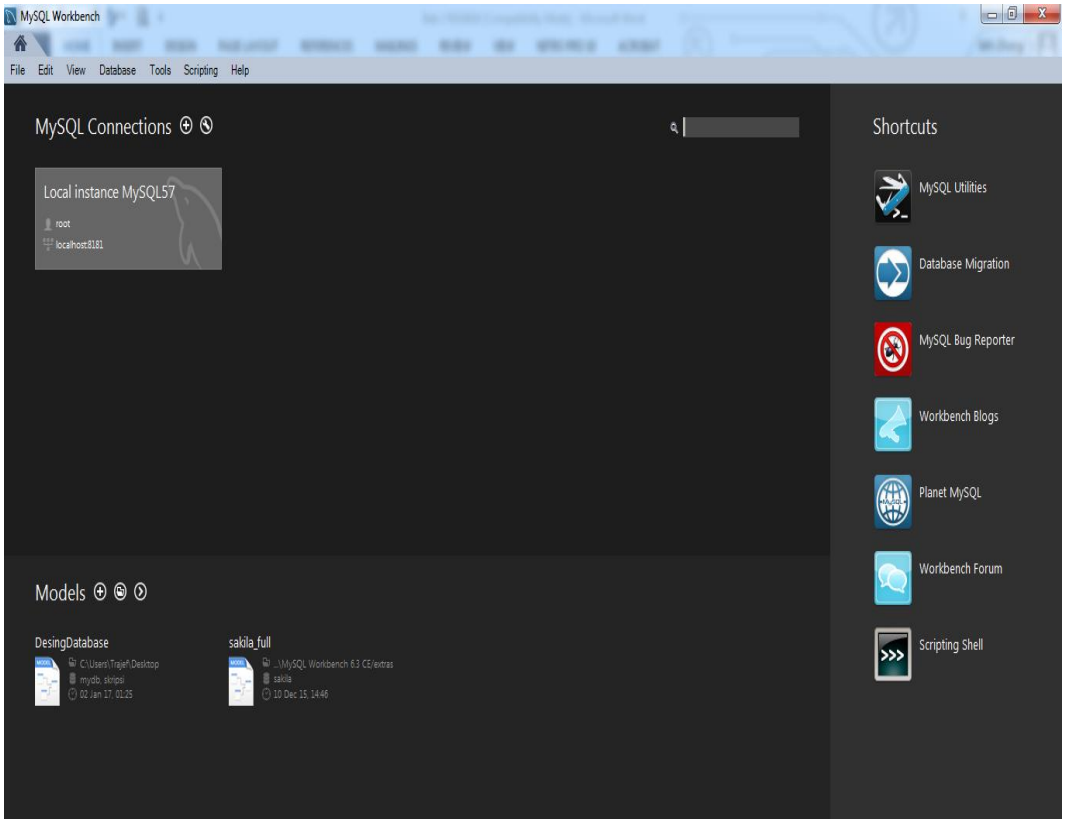
Menurut Hendry (2015: 7) MySQL memiliki beberapa keistimewaan, antara lain :

1. **Portabilitas**. MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, Mac Os X Server, dan masih banyak lagi.
2. **Perangkat lunak sumber terbuka**. MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.
3. **Multi-user**. MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. ***Performance tuning***, MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.

5. **Ragam tipe data**. MySQL memiliki ragam tipe data yang sangat kaya, seperti signed / unsigned integer, float, double, char, text, date, timestamp, dan lain-lain.
6. **Perintah dan Fungsi**. MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah Select dan Where dalam perintah (query).
7. **Keamanan**. MySQL memiliki beberapa lapisan keamanan seperti level subnetmask, nama host, dan izin akses user dengan sistem perizinan yang mendetail serta sandi terenkripsi.
8. **Skalabilitas dan Pembatasan**. MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (records) lebih dari 50 juta dan 60 ribu tabel serta 5 miliar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.
9. **Konektivitas**. MySQL dapat melakukan koneksi dengan klien menggunakan protokol TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).
10. **Lokalisasi.** MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11. **Antar Muka.** MySQL memiliki antar muka (interface) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (Application Programming Interface).

12. **Klien dan Peralatan**. MySQL dilengkapi dengan berbagai peralatan (tool) yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.

13. **Struktur tabel.** MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam PostgreSQL ataupunOracle.



Gambar 2.7 Tampilan Utama Aplikasi MySQL Workbench 6.3 CE
(Sumber: Data Penelitian, 2016)

2.3.5.1 *Entity Relationship Diagram*

Entity Relationship Diagram (ERD) merupakan salah satu alat bantu (berupa gambar) dalam model *database* relasional yang berguna untuk

menjelaskan hubungan atau relasi antar tabel yang terdapat di dalam *database.* Dalam ERD kita juga dapat melihat daftar kolom yang menyusun masing-masing tabel (Raharjo, 2011:57).

2.3.6 *Unified Modeling Language* (UML)

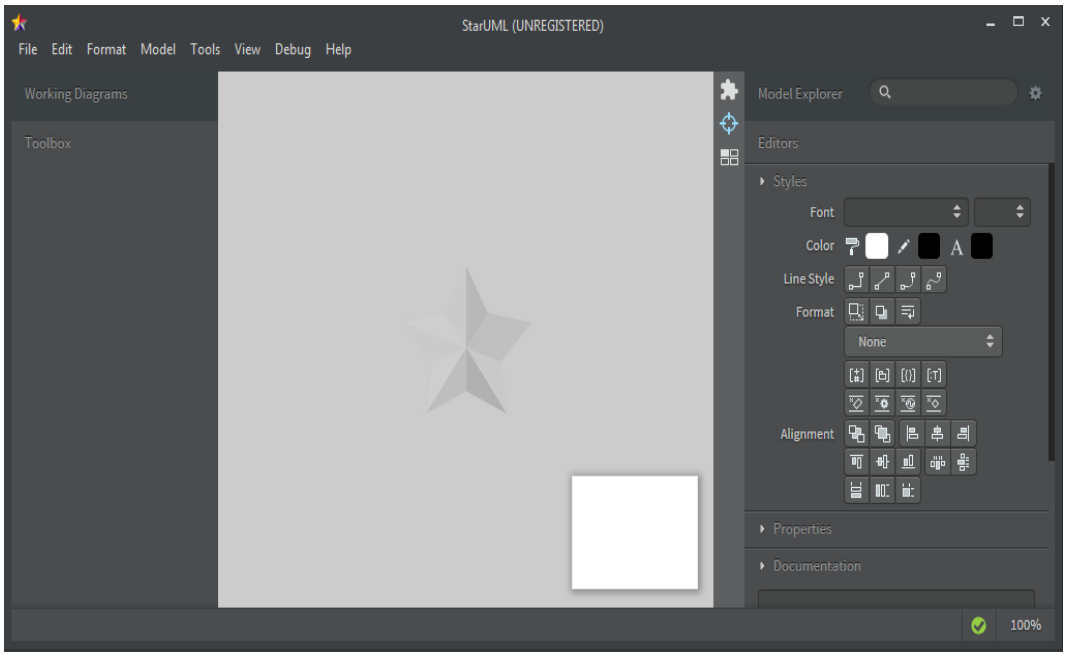
Menurut A.S dan Shalahuddin (2016:136) perangkat pemodelan adalah suatu model yang digunakan untuk menguraikan sistem menjadi bagian-bagian yang dapat diatur dan mengomunikasikan ciri konseptual dan fungsional kepada pengamat. Peran perangkat pemodelan :

1. Perangkat pemodelan bisa digunakan sebagai alat komunikasi antara pemakai dengan analis sistem maupun *developer* dalam pengembangan sistem.
2. Sebuah eksperimentasi yang bersifat ”*trial and error*”.
3. Model yang dibuat bisa meramalkan bagaimana suatu sistem akan bekerja.

Menurut A.S dan Shalahuddin (2016:133) UML (*Unified Modelling Language*) adalah salah satu standar bahasa ynag digunakan di dunia industri untk mendefinisikan *requirement,* membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.

Perangkat lunak yang membantu dalam mengrekayasa *UML* yang dinamakan CASE (*Computer-Aided Software Engineering*) tools. CASE *tools* mulai digunkana oleh pelaku rekayasa perangkat lunak sejak tahun 1980-an dalam hal ini mereka focus pada tahap awal pengembangan. Salah satu *tools* atau

perangkat lunak tersebut adalah StarUML. StarUML digolongkan kedalam kelompor *upper* Case *tools* yang mendukung perencanaan strategis dan pembangunan perangkat lunak (A.S dan Shalahuddin, 2016:122-123).



Gambar 2.8 Tampilan Utama Aplikasi StarUML
(Sumber: Data Penelitian, 2016)

2.3.6.1 *Class Diagram*

Class diagram berada paga bagian struktur diagram yang berguna menggambarkan kelas-kelas yang akan dibuat untuk membangun sebuah sistem. Sebuah kelas terdiri dari atribut dan operasi atau metode. Atribut adalah variabel-variabel atau kolom yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas (A.S dan Shalahuddin, 2016:141-142).

Menurut A.S dan Shalahuddin (2016:142) susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1. Kelas main

   Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan

2. Kelas yang menangani tampilan sistem (*View*)

   Kelas yang mendefinisikan dan mengatur tampilan ke pemakai

3. Kelas yang diambil dari pendefinisian *use case* (*controller*)

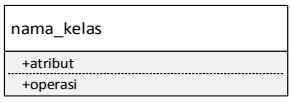
   Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*.

4. Kelas yang diambil dari pendefinisian data (*model*)

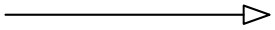
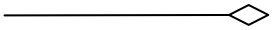
   Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Berikut adalah simbol-simbol yang ada pada diagram kelas, bisa di lihat pada tabel di bawah ini :

Tabel 2.1 Simbol pada *Class diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
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 2.1 Lanjutan

Asosiasi berarah / *direct association* 	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus)
Kebergantungan / *dependency*	Relasi antar kelas dengan makna kebergantungan kelas
Agresi / aggregation 	Relasi antar kelas dengan makna semua-bagian (*whole-part*)

Sumber: A.S dan Shalahuddin (2016:146-147)

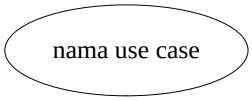
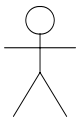
2.3.6.2 *Use Case Diagram*

Use case atau diagram *use case* berfungsi untuk menjelaskan fungsi yang ada pada suatu sistem dan siapa yang berhak menggunakan fungsi tersebut. *Use case* diagram merupakan bagian dari diagram kelakuan (*behavior*) (A.S dan Shalahuddin, 2016:155).

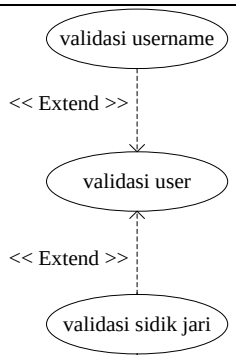
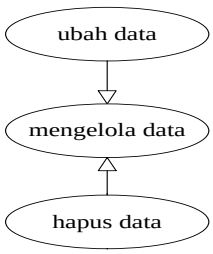
Menurut A.S dan Shalahuddin (2016:155) ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case.*

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi, jadi walaupun symbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

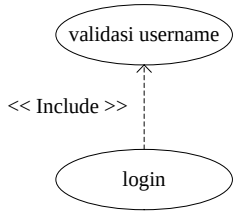
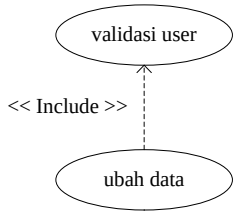
Tabel 2.2 Simbol pada *Use case diagram*

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 / a*ctor*  Nama aktor	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi.
Asosiasi / *association* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor
Ekstensi / *extend*	Relasi *use case* tambahan kesebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misalnya:

Tabel 2.2 Lanjutan

<< Extend >>	 arah panah mengarah pada *use case* yang ditambahkan, biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
Generalisasi / *generalization*	Hubungan generalisasi dan spesialisasi (umum - khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:  arah panah mengarah pada *use case* yang menjadi generalisasainya (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 diljalankan *use case* ini. Ada dua sudut pandang yang cukup besar mengenai *include* di *use case*: *Include* berarti *use case* yang ditambah-kan akan selalu dipanggil saat *use case* tambahan dijalankan, misal pada kasus berikut:

Tabel 2.2 Lanjutan

	 *1.Include* berarti *use case* yang ditambah-kan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan, missal pada kasus berikut:  Kedua sudut pandang diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan kebutuhan

Sumber: A.S dan Shalahuddin (2016:156-158)

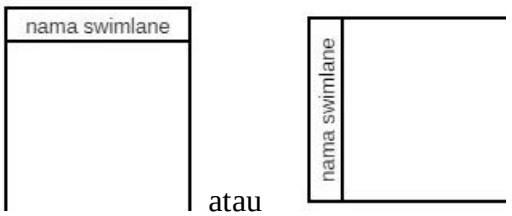
2.3.6.3 *Activity Diagram*

Activity diagram juga merupakan bagian dari diagram kelakuan (*behavior*). *Activity diagram* berguna untuk mendeskripsikan *workflow* (aliran kerja) dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. *Activity* diagram berbeda dengan *use case* diagram karna dia tidak menggambarkan apa yang dilakukan aktor, tetapi menggambarkan aktivitas yang dilakukan sistem (A.S dan Shalahuddin, 2016:161).

Tabel 2.3 Simbol pada *Activity diagram*

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

Tabel 2.3 Lanjutan

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

Sumber: A.S dan Shalahuddin (2016:162-163)

2.3.6.4 *Sequence Diagram*

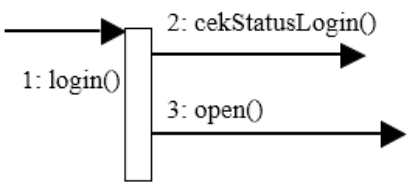
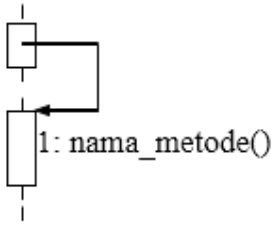
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripWsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Diagram sekuen yang sudah dibuat bisa digunakan untuk melihat

scenario yang mungkin terjadi pada *use case.* Sehingga jumlah diagram sekuen akan banyak jika jumlah diagram *use case* banyak (A.S dan Shalahuddin, 2016:165).

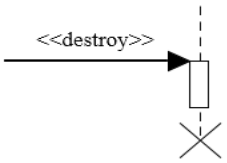
Tabel 2.4 Simbol pada *Sequence diagram*

Simbol	Deskripsi
Aktor/actor  atau nama aktor Tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi. 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 nama objek : nama kelas	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, misalnya

Tabel 2.4 Lanjutan

	 Maka cekStatusLogin() dan open() dilakukan di dalam metode login() Aktor tidak memiliki waktu aktif.
Pesan tipe create <<create>> pesan tipe call 1 : nama_metode()	Menyatakan suatu objek membuat objek yang lain. Arah panah mengarah pada objek yang dibuat Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri  arah panah mengarah pada objek yang memiliki operasi/metode, maka operasi/ metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi

Tabel 2.4 Lanjutan

Pesan tipe send 1 : masukan →	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirmi.
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 yang menerima kembalian.
Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada destroy

Sumber: A.S dan Shalahuddin (2016:165-167)

2.4 Penelitian Terdahulu

Dalam penelitian ini penulis memaparkan beberapa penelitian terdahulu yang relevan dengan permasalahan yang akan diteliti tentang analisis sistem pakar pertumbuhan anak standar who berbasis android dengan metode forward chaining.

Fidiantoro dan Setiadi (2013:367), *Model Penentuan Status Gizi Balita Di Puskesmas*, status gizi balita merupakan faktor penting yang harus diperhatikan karena masa balita merupakan periode perkembangan yang rentan dengan gizi. Upaya pemerintah dalam perbaikan gizi dengan memantau status gizi balita di

setiap wilayah kerja Puskesmas. Hasil penelitian ini berupa aplikasi model penentuan status gizi balita di Puskesmas yang hasilnya dapat membantu petugas gizi untuk menentukan status gizi balita di Puskesmas agar lebih efektif dan akurat.

Zamroni, dkk. (2013:507) pada jurnalnya yang berjudul "*Sistem Pakar Perkembangan Anak Usia 0-12 Bulan Berbasis Web Dengan Metode Forward Chaining*", masalah yang dihadapi adalah banyaknya bayi yang lahir pada saat ini. Tetapi dengan bertambahnya tersebut masih ada saja orang tua yang belum faham atau kurang mengertinya tentang perkembangan anak secara baik dan sesuai dengan usia yang benar. Salah satu tujuan penelitian yaitu menyediakan media informasi yang mudah pada masyarakat dalam mengambil keputusan sehingga dapat segera mengetahui perkembangan anak tergolong yang seperti apa. Salah satu manfaat penelitian adalah masyarakat *non-pakar* dapat memanfaatkan untuk mengetahui perkembangan anak secara baik dan benar. Pada sistem pakar ini digunakan metode Forward Chainning, dimana proses dimulai dengan memilih fakta-fakta yang telah disediakan selanjutnya akan ditentukan seberapa jauh perkembangan anak tersebut. Saran yang di berikan yaitu "Perlu diadakan penambahan data untuk jenis perkembangan usia selanjutnya sehingga informasi yang dimiliki akan semakin luas dan banyak."

Penelitian Ernawati, dkk. (2014:109), *Hubungan Panjang Badan Lahir Terhadap Perkembangan Anak Usia 12 Bulan*, menyatakan hasil penelitian menunjukkan sebanyak 9,5 persen bayi dengan berat badan lahir rendah (BBLR) dan 22 persen stunting. Nilai z-skor panjang badan terhadap umur pada bayi baru

lahir berkolerasi dengan perkembangan motorik dan sosial emosi sejak bayi berumur nol bulan, yaitu rho=0,33; p=0,004 untuk motorik dan rho=0,244 dengan p=0,036 untuk sosial emosi. Sedangkan korelasi terhadap perkembangan bahasa baru tampak pada saat bayi berumur satu bulan yaitu rho=0,29 dengan p=0,031 dan korelasi terhadap perkembangan kognitif terjadi pada usia dua bulan rho=0,318 dengan p=0,011. Pada anak lahir stunting median perkembangan bahasa lebih rendah dibandingkan kelompok yang normal. Pendek (*stunting)* adalah gangguan pertumbuhan pada anak balita di Indonesia yang perlu mendapat perhatian khusus. Salah satu dampaknya *stunting*, terutama pada anak usia kurang dua tahun yang mengakibatkan penurunan tingkat kecerdasan.

Penelitian Shaid, dkk. (2015:37), *Sistem Pakar Pertumbuhan Balita Berbasis Web Dengan Metode Case Based Reasoning*, menyatakan pertumbuhan balita bisa terjadi berdasarkan beberapa factor, yaitu berdasarkan kelahirannya dan pertumbuhan gizi yang dikonsumsi. Dengan memanfaatkan metode Case-base Reasoning, dapat dihasilkan suatu aplikasi untuk mengidentifikasi pertumbuhan balita. Dengan harapan sistem ini nantinya dapat digunakan sebagai sarana atau sebagai pengetahuan dalam menjaga kestabilan pertumbuhan balita dan membantu anda untuk mendapatkan hasil yang optimal dalam menjaga pertumbuhan setiap balita. Metode Case Based Reasoning (CBR) digunakan dalam aplikasi Pertumbuhan Balita dengan menggunakan Perhitungan Nearest Neighbor, Dimana data kasus baru akan dibandingkan perhitungannya dengan data kasus lama yang ada di database, dan kemudian dihitung kriteria kemiripannya berdasarkan rumus atau ketentuan yang berlaku. Dimana Shaid, dkk.

(2015: 44) menyarankan "Program ini dapat lebih disempurnakan, khususnya pada bagian proses pencarian data kasus. Hal ini akan memberikan hasil konsultasi yang diberikan perangkat lunak lebih baik."

Nugraha, dkk. (2015:11), *Analisis Sistem Pakar Cara Diet Berdasarkan Golongan Darah*, saat ini perkembangan teknologi begitu pesat, begitu pesatnya hingga muncul kecerdasan buatan yang memiliki sistem pakar. Dari hal tersebut kami melakukan analisis tentang sistem pakar yang menyangkut cara diet berdasarkan golongan darah. Dari hal itu adapula penelitian yang dilakukan dari sebuah buku untuk mengambil data dan mencari sampel untuk uji coba program, maka dibuatlah suatu program yang berbasis web dari cara diet berdasarkan golongan darah.

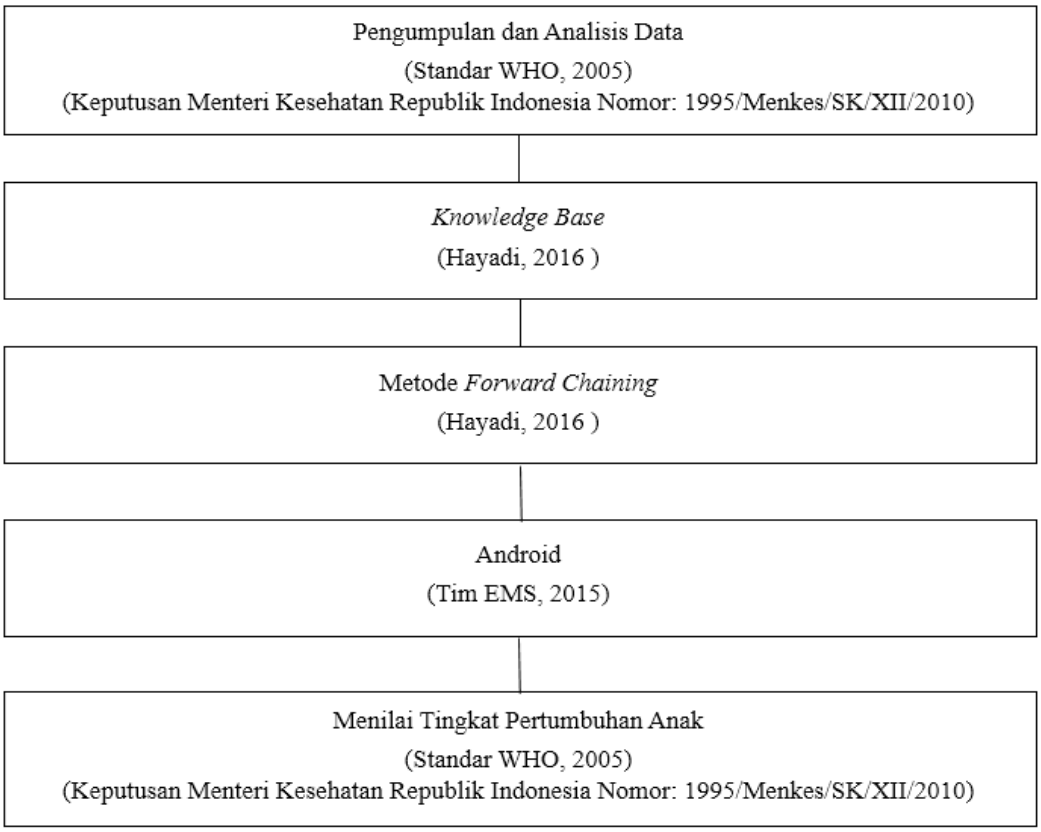
Rengganis (2016:20), *Perancangan Sistem Pakar Klasifikasi Status Gizi Balita Berdasarkan Indeks Antropometri Berat Badan Terhadap Umur (BB/U) menggunakan Metode Forward Chaining,* anak berusia dibawah lima tahun (balita) merupakan golongan usia yang sangat rentan akan kebutuhan gizi dan masalah kesehatan. Namun kurangnya kesadaran masyarakat akan gizi dan kurangnya sarana informasi penunjang kebutuhan gizi menyebabkan masalah ini kurang mendapat penanganan secara tepat dan cepat. Peneliti menyarankan bahwa penelitian lanjutan dengan menambahkan indeks antropometri lainnya.

Purwati (2016:12), *Klasifikasi Status Gizi Balita Berdasarkan Indeks Antropometri Bb/U Dan Bb/Tb Menggunakan Jaringan Saraf Tiruan*, status gizi balita dapat ditentukan berdasarkan indeks antropometri BB/U dan BB/TB dengan menggunakan standar baku WHO-NCHS. Penelitian ini bertujuan untuk

mengklasifikasikan status gizi menggunakan Jarigan Saraf Tiruan *Backpropagation* berdasarkan indeks antropometri BB/U yang akan menghasilkan status gizi kedalam gizi buruk, kurang, baik, dan lebih, serta mengklasifikasikan status gizi berdasarkan indeks antropometri BB/TB yang akan menghasilkan status sangat kurus, kurus, normal, dan gemuk. Variabel yang digunakan dalam penelitian ini yaitu jenis kelamin, umur, berat badan, tinggi badan, dan status ekonomi.

2.5 Kerangka Pemikiran

Kerangka Berpikir adalah sebuah gambaran yang menghubungkan variabel-variabel untuk mendeskripsikan dan menjelaskan masalah serta menghasilkan sebuah hipotesis atau solusi. Jadi, kerangka berpikir adalah sintesis tentang hubungan antar variabel yang disusun dari berbagai teori yang dideskripsikan (Sudaryono, 2015:21). Kerangka berfikir pada penelitian ini dapat dilihat pada gambar 2.9.



Gambar 2.9 Kerangka Pemikir
(Sumber: Data Penelitian, 2016)

Mengumpulkan data-data yang berkaitan dengan penilaian tingkat pertumbuhan balita standar WHO 2005. Melakukan analisis pada data ynag didapatkan untuk menghasilkan sebuah basis pengetahuan (*knowledge base).* Sehingga bisa digunakan dalam perancangan sebuah sistem pakar dengan menggunakan metode *forward chaining*. Sistem pakar ini akan dirancang menjadi sebuah aplikasi Android. Dimaksudkan agar aplikasi android ini dapat menguraikan penilaian tingkat pertumbuhan balita dan dapat digunakan dengan mudah oleh masyarakat *non-pakar*.