

BAB II

KAJIAN PUSTAKA

2.1 Teori Dasar

Untuk menghasilkan penelitian yang baik dan terarah, maka diperlukan landasan berupa teori-teori yang telah ada. Landasan teori membahas teori-teori yang berhubungan dan berfungsi sebagai acuan agar penelitian tidak menyimpang dari tujuan awal yang telah ditetapkan.

Pada bab ini akan dijelaskan tentang beberapa teori antara lain Kecerdasan Buatan atau *Artificial Intelligence (AI)* dan beberapa subdisiplin ilmunya, *Web*, *Database*, *UML (Unified Modeling Language)*, variabel penelitian, penelitian-penelitian terdahulu, *Software* pendukung dan kerangka pemikiran yang digunakan dalam penelitian ini.

2.1.1 Kecerdasan Buatan (*Artificial Intelligence*)

Kecerdasan buatan berasal dari bahasa Inggris “*Artificial Intelligence*” atau disingkat *AI*, yaitu *intelligence* adalah kata sifat yang cerdas, sedangkan *artificial* artinya buatan. Kecerdasan buatan yang dimaksud disini merujuk pada mesin yang mampu berpikir, menimbang tindakan yang akan diambil, dan mampu mengambil keputusan seperti yang dilakukan oleh manusia (Sutojo, Mulyanto, dan Suhartono, 2011: 1).

Menurut Winston dan Prendergast (1984) *dalam* Sutojo, *dkk* (2011: 3), tujuan dari kecerdasan buatan adalah:

1. Membuat mesin menjadi lebih pintar (tujuan utama).
2. Memahami apa itu kecerdasan (tujuan ilmiah).
3. Membuat mesin lebih bermanfaat (tujuan *entrepreneurial*)

Berdasarkan definisi tersebut diatas, maka kecerdasan buatan menawarkan media maupun uji teori tentang kecerdasan. Teori-teori ini nantinya dapat dinyatakan dalam bahasa pemrograman dan eksekusinya dapat dibuktikan pada komputer nyata. Kecerdasan buatan memungkinkan komputer untuk berpikir atau menalar dan menirukan proses belajar manusia sehingga informasi baru dapat diserap sebagai pengetahuan, pengalaman, dan proses pembelajaran serta dapat digunakan sebagai acuan dimasa-masa yang akan datang (Sutojo, *dkk*, 2011: 3).

Menurut Budiharto dan Suhartono (2014: 3-4) sumbangan terbesar di bidang AI diawali oleh tulisan Alan Turing pada tahun 1950 yang mencoba menjawab pertanyaan, “Dapatkah komputer berpikir?” dengan menciptakan *Turing Machine*. Paper Alan Turing pada tahun 1950 berjudul *Computing Machine and Intelligence* mendiskusikan syarat sebuah mesin dianggap cerdas. Turing beranggapan bahwa jika mesin dapat dengan sukses berperilaku seperti manusia, maka mesin itu dapat dianggap cerdas.

Pada akhir 1955, Newell dan Simon mengembangkan *The Logic Theorist*, program AI pertama. Program ini merepresentasikan masalah sebagai model pohon, lalu penyelesaiannya dengan memilih cabang yang akan menghasilkan kesimpulan terbenar. Pada tahun 1956, John McCarthy dari *Massachusetts Institute of*

Technology yang dianggap sebagai Bapak *AI*, menyelenggarakan konferensi bertajuk *The Dartmouth Summer Research Project on Artificial Intelligence*. Konferensi ini bertujuan menarik bakat dan keahlian orang banyak untuk masuk dalam dunia kecerdasan buatan. Konferensi Dartmouth mempertemukan para pendiri dalam dunia *AI* dan bertugas meletakkan dasar bagi masa depan pengembangan serta penelitian *AI* (Budiharto dan Suhartono, 2014: 4).

Definisi *AI* pada awalnya diusulkan oleh John McCarthy pada tahun 1955, yaitu: *The goal of AI is to develop machines that behave as though they were intelligent*. McCarthy mendefinisikannya sebagai, “*AI* merupakan cabang dari ilmu komputer yang berfokus pada pengembangan komputer untuk dapat memiliki kemampuan dan berperilaku seperti manusia (Budiharto dan Suhartono, 2014: 4).

Subdisiplin ilmu dalam kecerdasan buatan antara lain (Sutojo, *dkk*, 2011: 12-17) :

1. Sistem Pakar (*Expert System*). Suatu sistem yang dirancang untuk dapat menirukan keahlian seorang pakar dalam menjawab pertanyaan dan memecahkan suatu masalah.
2. Pengolahan Bahasa Alami (*Natural Language Processing*). Pembuatan program yang memiliki kemampuan untuk memahami bahasa manusia.
3. Pengenalan Ucapan (*Speech Recognition*). Pengembangan teknik dan system yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan.
4. Robotika dan Sistem Sensor (*Robotics & Sensory Systems*). Robotika adalah ilmu pengetahuan dan teknologi rekayasa robot dan desain, manufaktur,

aplikasi, dan disposisi struktural. Untuk dapat menggunakan *software intelligence*, robot harus bisa menerima lingkungannya. Ini berarti robot harus menggunakan sensor untuk mendeteksi posisi yang ada disekitarnya.

5. *Computer Vision*. Salah satu cabang ilmu pengetahuan yang mempelajari bagaimana komputer dapat mengenali objek yang diamati atau diobservasi.
6. *Intelligent Computer-Aided Instruction*. Komputer dapat digunakan sebagai *tutor* yang dapat melatih dan mengajar.
7. *Game Playing*. *Game* adalah permainan, yaitu kegiatan yang kompleks yang didalamnya terdapat peraturan, *play* dan budaya.

Bidang ilmu lain yang termasuk dalam kecerdasan buatan diantaranya jaringan syaraf tiruan (*artificial neural network*) dan logika *fuzzy* (*fuzzy logic*).

2.1.2 Jaringan Saraf Tiruan (*Artificial Neural Network*)

Menurut Sutojo, dkk (2011: 283) Jaringan saraf tiruan adalah paradigma pengolahan informasi yang terinspirasi oleh sistem saraf secara biologis, seperti proses informasi pada otak manusia. Jaringan saraf tiruan mempunyai kemampuan luar biasa untuk mendapatkan informasi dari data yang rumit atau tidak tepat, mampu menyelesaikan permasalahan yang tidak terstruktur dan sulit didefinisikan, dapat belajar dari pengalaman, mampu mengakuisisi pengetahuan walaupun tidak ada kepastian, mampu melakukan generalisasi dan ekstraksi dari suatu pola data tertentu, dapat menciptakan suatu pola pengetahuan melalui pengaturan diri atau kemampuan belajar (*self organizing*), mampu memilih input data ke dalam kategori tertentu yang sudah ditetapkan (klasifikasi), mampu menggambarkan suatu objek

secara keseluruhan walaupun hanya diberikan sebagian data dari objek tersebut (asosiasi), mempunyai kemampuan mengolah data-data input tanpa harus mempunyai target, dan mampu menemukan jawaban terbaik sehingga mampu meminimalisasi fungsi biaya (optimasi) (Sutojo, *dkk*, 2011: 284).

Berdasarkan cara memodifikasi bobotnya, pelatihan jaringan saraf tiruan dibagi menjadi dua, yaitu (Sutojo, *dkk*, 2011: 301- 392):

1. Pelatihan dengan Supervisi (pembimbing)

Dalam pelatihan ini, jaringan dipandu oleh sejumlah pasangan data (masukan dan target) yang berfungsi sebagai pembimbing untuk melatih jaringan hingga diperoleh bobot yang terbaik. Algoritma yang termasuk dalam pelatihan dengan supervisi antara lain:

- a. *Hebb-Rule*

Pada tahun 1949, D.O. Hebb memperkenalkan cara menghitung bobot dan bias secara iteratif dengan memanfaatkan model pembelajaran dengan supervisi sehingga bobot dan bias dapat dihitung secara otomatis tanpa harus melakukan cara coba-coba. Arsitektur jaringan ini terdiri dari beberapa unit *input* dihubungkan langsung dengan sebuah unit *output*, ditambah dengan sebuah bias.

- b. *Perceptron*

Model ini ditemukan oleh Rosenblatt (1962) dan Minsky – Papert (1969). Model jaringan ini merupakan model yang terbaik pada saat itu. Algoritma pelatihan *perceptron* digunakan baik untuk *input* biner maupun bipolar, dengan θ tertentu, dan bias yang dapat diatur.

c. *Delta-Rule*

Selama pelatihan pola, *Delta-Rule* akan mengubah bobot dengan cara meminimalkan *error* antara *output* jaringan dengan target.

d. *Backpropagation*

Backpropagation adalah metode penurunan gradien untuk meminimalkan kuadrat *error* keluaran. Pelatihan jaringan ini terdiri dari 3 tahap, yaitu tahap perambatan maju (*forward propagation*), tahap perambatan balik, dan tahap perubahan bobot dan bias. Arsitektur jaringan ini terdiri dari *input layer*, *hidden layer*, dan *output layer*.

e. *Heteroassociative Memory*

Jaringan saraf *heteroassociative memory* adalah jaringan yang dapat menyimpan kumpulan pengelompokan pola dengan cara menentukan bobot-bobotnya sedemikian rupa. Algoritma pelatihan yang biasa digunakan adalah *Hebb-Rule*.

f. *Bidirectional Associative Memory (BAM)*

Bidirectional Associative Memory (BAM) adalah model jaringan saraf yang memiliki 2 lapisan, yaitu lapisan *input* dan lapisan *output* yang mempunyai hubungan timbal balik antara keduanya (bersifat *bidirectional*). Arsitektur jaringan ini terdiri dari 3 *neuron* pada lapisan *input* dan 2 *neuron* pada lapisan *output*. Model jaringan ini terbagi menjadi 2 jenis yaitu *BAM Diskrit* dan *BAM Kontinu*.

g. *Learning Vector Quantization (LVQ)*

Learning Vector Quantization (LVQ) adalah suatu model pelatihan pada lapisan kompetitif terawasi yang akan belajar secara otomatis untuk mengklasifikasikan vektor-vektor *input* ke dalam kelas-kelas tertentu. Kelas-kelas yang dihasilkan tergantung pada jarak antara vektor-vektor *input*.

2. Pelatihan tanpa Supervisi (Jaringan Kohonen)

Dalam pelatihan ini, tidak ada pembimbing yang digunakan untuk memandu proses pelatihan. Jaringan hanya diberi *input* tetapi tidak mendapatkan target yang diinginkan sehingga modifikasi bobot pada jaringan dilakukan menurut parameter tertentu. Prof. Teuvo Kohonen adalah orang pertama yang memperkenalkan jaringan kohonen. Pada jaringan ini, *neuron-neuron* pada suatu lapisan akan menyusun dirinya sendiri berdasarkan *input* nilai tertentu dalam suatu *cluster*. Dalam proses penyusunan diri, *cluster* yang dipilih sebagai pemenang adalah *cluster* yang mempunyai vektor bobot paling cocok dengan pola *input* (memiliki jarak yang paling dekat).

2.1.3 Logika Fuzzy (*Fuzzy Logic*)

Konsep tentang logika *fuzzy* diperkenalkan oleh Prof. Lotfi Astor Zadeh pada tahun 1962. Logika *fuzzy* adalah metode sistem kontrol pemecahan masalah, yang cocok untuk diimplementasikan pada sistem, mulai dari sistem yang sederhana, sistem kecil, *embedded system*, jaringan *personal computer*, *multi-channel* atau *workstation* berbasis akuisisi data, dan sistem control (Sutojo, dkk, 2011: 211).

Menurut Budiharto dan Suhartono (2014: 152) beberapa kelebihan logika *fuzzy* antara lain:

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

Menurut Budiharto dan Suhartono (2014: 152) logika *fuzzy* banyak digunakan di berbagai bidang, antara lain:

1. Pada bidang industri, logika *fuzzy* digunakan untuk menghasilkan *service robot* untuk melayani manusia.
2. Pada bidang bisnis, logika *fuzzy* digunakan untuk memperkirakan naik turunnya harga saham atau memperkirakan keuntungan penjualan selanjutnya.
3. Pada bidang manajemen, logika *fuzzy* juga dimanfaatkan untuk sistem penggajian karyawan.
4. Pada lingkungan sehari-hari, logika *fuzzy* juga banyak ditemukan pada mesin cuci dan pemanas ruangan.

5. Logika *fuzzy* juga telah masuk dalam bidang pertanian. Misalnya untuk meramal cuaca sebelum petani mulai menanam.

Menurut Sutojo, dkk (2011: 233-234) cara kerja logika *fuzzy* meliputi beberapa tahapan berikut:

1. *Fuzzyfikasi*

Fuzzyfikasi merupakan proses untuk mengubah *input fuzzy* yang mempunyai nilai tegas menjadi variabel linguistik menggunakan fungsi keanggotaan yang disimpan dalam basis pengetahuan.

2. Pembentukan basis pengetahuan *fuzzy*

Basis pengetahuan *fuzzy* berisikan kumpulan *rule-rule fuzzy* dalam bentuk pernyataan *IF...THEN*.

3. Mesin inferensi (fungsi implikasi *Max-Min* atau *Dot-Product*)

Mesin inferensi merupakan proses mengubah *input fuzzy* menjadi *output fuzzy* dengan cara mengikuti aturan-aturan (*IF-THEN rules*) yang telah ditetapkan pada basis pengetahuan *fuzzy*.

4. *Defuzzyfikasi*

Defuzzyfikasi adalah proses mengubah *output fuzzy* yang diperoleh dari mesin inferensi menjadi nilai tegas menggunakan fungsi keanggotaan yang sesuai dengan saat dilakukan *fuzzyfikasi*. Banyak cara untuk melakukan *defuzzyfikasi*, diantaranya metode berikut:

- a. Metode rata-rata (*Average*)
- b. Metode titik tengah (*Center of Area*)

Menurut Sutojo, *dkk* (2011: 233-238) metode yang digunakan dalam logika *fuzzy* terdiri dari:

1. Metode Tsukamoto

Dalam inferensinya, metode Tsukamoto menggunakan tahapan sebagai berikut:

- a. *Fuzzyfikasi*
- b. Pembentukan basis pengetahuan *fuzzy* (*rule* dalam bentuk *IF...THEN*)
- c. Mesin inferensi menggunakan fungsi implikasi *MIN* (*Minimum*)
- d. *Defuzzyfikasi* menggunakan metode Rata-rata (*Average*)

2. Metode Mamdani

Metode ini sering digunakan karena strukturnya yang sederhana. Pada metode ini, untuk mendapatkan *output* diperlukan 4 tahapan sebagai berikut:

- a. *Fuzzyfikasi*
- b. Pembentukan basis pengetahuan *fuzzy* (*rule* dalam bentuk *IF...THEN*)
- c. Aplikasi fungsi implikasi menggunakan fungsi *MIN* (*Minimum*) dan komposisi antar-*rule* menggunakan fungsi *MAX* (*Maximum*) dengan menghasilkan himpunan *fuzzy* baru
- d. *Defuzzyfikasi* menggunakan metode *Centroid* (Titik Tengah)

3. Metode Sugeno

Metode ini diperkenalkan oleh Takagi-Sugeno Kang pada tahun 1985. Dalam metode ini, *output* sistem berupa konstanta atau persamaan linier. Dalam inferensinya, metode Sugeno menggunakan tahapan sebagai berikut:

- a. *Fuzzyfikasi*

- b. Pembentukan basis pengetahuan *fuzzy* (*rule* dalam bentuk *IF...THEN*)
- c. Mesin inferensi menggunakan fungsi implikasi *MIN* (*Minimum*)
- d. *Defuzzyfikasi* menggunakan metode Rata-rata (*Average*)

2.1.4 Sistem Pakar (*Expert System*)

2.1.4.1 Pengertian Sistem Pakar

Menurut Sutojo, dkk (2011: 159-160) 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 Newel 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* & *XSEL*, untuk membantu konfigurasi sistem komputer besar, *SOPHIE* untuk analisis sirkuit elektronik, *Prospector* digunakan di bidang geologi untuk 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 Budiharto dan Suhartono (2014: 132) sistem pakar adalah program komputer yang menyimulasi penilaian dan perilaku manusia atau organisasi yang memiliki pengetahuan dan pengalaman ahli dalam bidang tertentu. Sistem pakar yang canggih dapat ditingkatkan dengan penambahan basis pengetahuan atau set aturan. Diantara banyak sistem pakar yang ada, yang terkenal adalah aplikasi

bermain catur dan sistem diagnosis medis. Sistem pakar dibuat hanya pada domain pengetahuan tertentu untuk suatu kepakaran tertentu yang mendekati kemampuan manusia di salah satu bidang saja. Sistem pakar mencoba mencari penyelesaian yang memuaskan, yaitu sebuah penyelesaian yang cukup bagus agar pekerjaan dapat berjalan walaupun bukan merupakan penyelesaian yang optimal (Hartati dan Iswanti, 2008: 2).

Istilah sistem pakar berasal dari istilah *knowledge-based expert system*. Istilah ini muncul karena untuk memecahkan masalah, sistem pakar menggunakan pengetahuan seorang pakar yang dimasukkan ke dalam komputer. Seorang yang bukan pakar menggunakan sistem pakar untuk meningkatkan kemampuan pemecahan masalah, sedangkan seorang pakar menggunakan sistem pakar untuk *knowledge assistant* (Sutojo, dkk, 2011: 160).

Berdasarkan penelitian Wahyudi dan Jumadi (2011: 198) disebutkan bahwa kekuatan sistem pakar terletak pada kemampuannya untuk memecahkan masalah-masalah praktis pada saat sang pakar berhalangan. Orientasi sistem pakar adalah konsultasi. Karena sistem pakar berfungsi sebagai konsultan, sehingga pemakai berkonsultasi dengan sistem pakar untuk mendapatkan nasihat. Dengan sistem pakar ini, pemakai yang tidak berpengalaman sekalipun bisa memecahkan suatu masalah yang bagaimanapun rumitnya dan bisa mengambil keputusan yang tepat dan akurat serta dapat membantu aktivitas para pakar sebagai asisten yang mempunyai pengetahuan yang dibutuhkan.

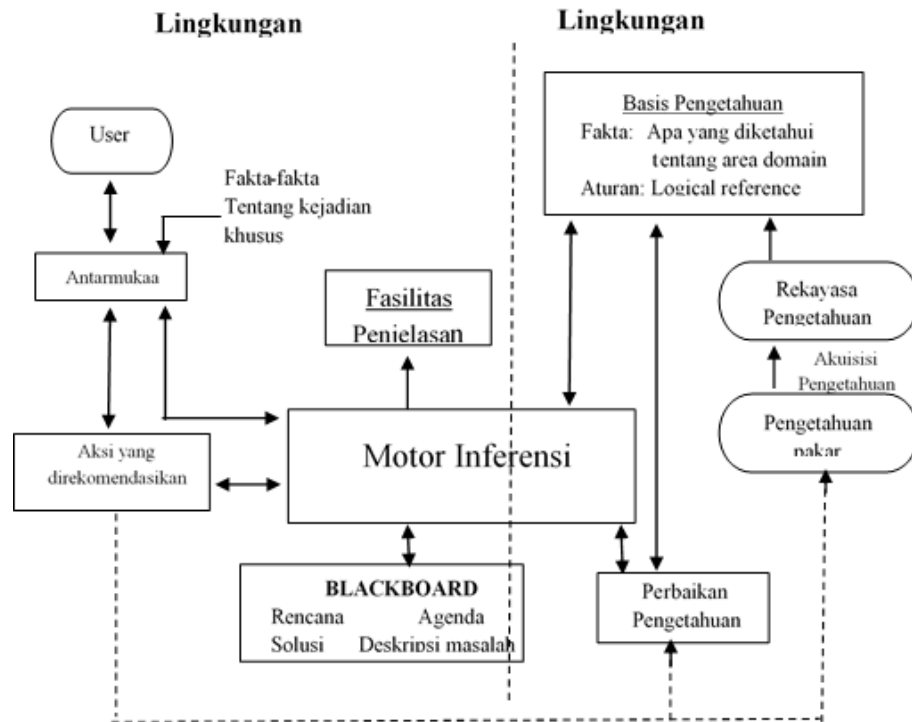
2.1.4.2 Ciri-Ciri Sistem Pakar

Menurut Sutojo, *dkk* (2011: 162) ciri-ciri dari sistem pakar adalah sebagai berikut:

1. Terbatas pada domain keahlian tertentu.
2. Dapat memberikan penalaran untuk data-data yang tidak lengkap atau tidak pasti.
3. Dapat menjelaskan alasan-alasan dengan cara yang dapat dipahami.
4. Bekerja berdasarkan kaidah/*rule* tertentu.
5. Mudah dimodifikasi.
6. Basis pengetahuan dan mekanisme inferensi terpisah.
7. Keluarannya bersifat anjuran.
8. Sistem dapat mengaktifkan kaidah secara searah yang sesuai, dituntun oleh dialog dengan pengguna.

2.1.4.3 Struktur Sistem Pakar

Sistem pakar terdiri dari dua bagian penting, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan digunakan untuk membangun komponen-komponennya dan memperkenalkan 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 seorang pakar (Sutojo, *dkk*, 2011: 166).



Gambar 2.1 Komponen-komponen Penting dalam Sistem Pakar
(Sumber: Sutojo, *dkk*, 2011: 167)

Keterangan:

1. Akuisisi Pengetahuan

Sub sistem ini 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. 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*)

Basis pengetahuan mengandung pengetahuan yang diperlukan untuk memahami, memformulasikan, dan menyelesaikan masalah. Basis pengetahuan terdiri dari dua elemen dasar, yaitu:

- a. Fakta, misalnya situasi, kondisi, atau permasalahan yang ada.
- b. *Rule* (aturan), untuk mengarahkan penggunaan pengetahuan dalam memecahkan masalah.

3. Mesin Inferensi (*Inference Engine*)

Mesin inferensi 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. Ada tiga teknik pengendalian yang digunakan, yaitu *forward chaining*, *backward chaining* dan gabungan dari kedua teknik tersebut.

4. Daerah Kerja (*Blackboard*)

Blackboard yaitu area pada memori yang berfungsi sebagai basis data. Tiga keputusan yang dapat di rekam 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 Pengguna (*User Interface*)

User interface digunakan sebagai media komunikasi antara pengguna dan sistem pakar. Komunikasi disajikan dalam Bahasa alami (*natural language*) dan dilengkapi dengan grafik, menu, dan formulir elektronik.

6. Subsistem Penjelasan (*Explanation Subsystem/Justifier*)

Subsistem penjelasan 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 diperlukan oleh program agar dapat menganalisis alasan-alasan kesuksesan dan kegagalannya dalam mengambil kesimpulan. Dengan cara ini basis pengetahuan yang lebih baik dan penalaran yang lebih efektif akan dihasilkan.

8. Pengguna

Pada umumnya pengguna sistem pakar bukanlah seorang pakar (*non-expert*) yang membutuhkan solusi, saran, atau pelatihan (*training*) dari berbagai permasalahan yang ada.

2.1.4.4 Kelebihan dan Kelemahan Sistem Pakar

Sistem pakar menjadi sangat populer karena sangat banyak kemampuan dan manfaat yang diberikannya, diantaranya (Sutojo, *dkk*, 2011: 160-161):

1. Meningkatkan produktivitas, karena sistem pakar dapat bekerja lebih cepat daripada manusia.
2. Membuat seorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kualitas, dengan memberi nasehat yang konsisten dan mengurangi kesalahan.
4. Mampu menangkap pengetahuan dan kepakaran seseorang.

5. Dapat beroperasi dilingkungan yang berbahaya.
6. Memudahkan akses pengetahuan seorang pakar.
7. Handal. Sistem pakar tidak pernah menjadi bosan dan kelelahan atau sakit.
8. Meningkatkan kapabilitas sistem komputer.
9. Mampu bekerja dengan informasi yang tidak lengkap atau tidak pasti.
10. Bisa digunakan sebagai media pelengkap dalam pelatihan.
11. Meningkatkan kemampuan untuk menyelesaikan masalah karena sistem pakar mengambil sumber pengetahuan dari banyak pakar.

Selain memiliki manfaat atau kelebihan, berikut adalah beberapa kelemahan dari Sistem pakar (Sutojo, *dkk*, 2011: 161):

1. Biaya yang sangat mahal untuk membuat dan memeliharanya.
2. Sulit dikembangkan karena keterbatasan keahlian dan ketersediaan pakar.
3. Sistem pakar tidak 100% (seratus persen) bernilai benar.

2.1.5 Forward Chaining

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 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 (Sutojo, *dkk*, 2011: 171).

Berdasarkan penelitian Listiyono (2008: 117) disebutkan bahwa pelacakan ke depan (*forward chaining*) adalah pendekatan yang dimotori data (*data driven*). Dalam pendekatan ini pelacakan dimulai dari informasi masukan, dan selanjutnya mencoba menggambarkan kesimpulan. Pelacakan ke depan (*forward chaining*) mencari fakta yang sesuai dengan bagian *IF* dari aturan *IF-THEN*.

Menurut Hartati dan Iswanti (2008: 10) Pengetahuan yang dimasukkan ke dalam sistem pakar disajikan dalam bentuk yang dapat dimengerti dan diterima oleh sistem pakar, salah satunya adalah kaidah produksi (*rule*). Kaidah produksi dituliskan dalam bentuk jika-maka (*IF-THEN*). Kaidah *IF-THEN* menghubungkan anteseden (*antecedent*) dengan konsekuensi yang diakibatkannya (Hartati dan Iswanti, 2008: 25). Berbagai struktur kaidah *IF-THEN* yang menghubungkan obyek atau atribut sebagai berikut (Adedeji, 1992 dalam Hartati dan Iswanti, 2008: 25):

1. *IF* premis *THEN* konklusi
2. *IF* masukan *THEN* keluaran
3. *IF* kondisi *THEN* tindakan
4. *IF* antesenden *THEN* konsekuen
5. *IF* data *THEN* hasil
6. *IF* tindakan *THEN* tujuan
7. *IF* aksi *THEN* reaksi
8. *IF* gejala *THEN* diagnosa

Menurut Hartati dan Iswanti (2008: 25) premis mengacu pada fakta yang harus benar sebelum konklusi tertentu dapat diperoleh. Masukan mengacu pada data yang harus tersedia sebelum keluaran dapat diperoleh. Kondisi mengacu pada

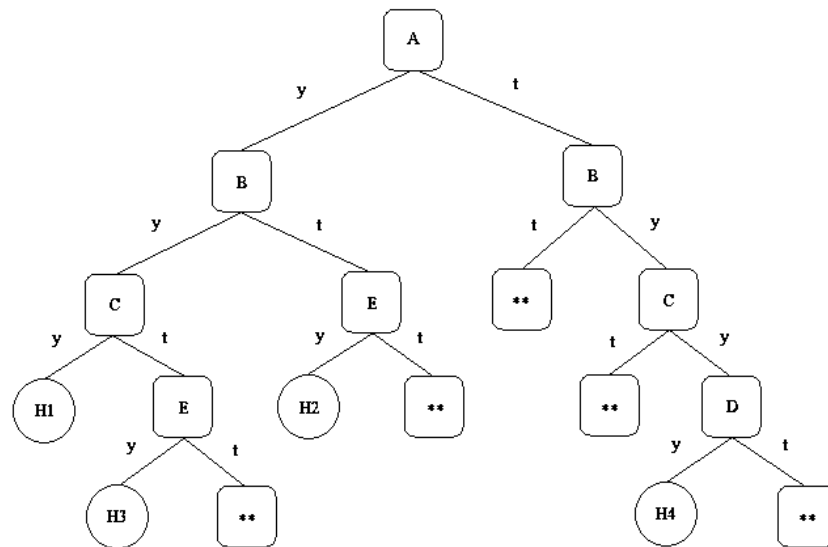
keadaan yang harus berlaku sebelum tindakan dapat diambil. Antesenden mengacu situasi yang terjadi sebelum konsekuensi dapat diamati. Data mengacu pada informasi yang harus tersedia sehingga sebuah hasil dapat diperoleh. Tindakan mengacu pada kegiatan yang harus dilakukan sebelum hasil dapat diharapkan. Aksi mengacu pada kegiatan yang menyebabkan munculnya efek dari tindakan tersebut. Gejala mengacu pada keadaan yang menyebabkan adanya kerusakan atau keadaan tertentu yang mendorong adanya pemeriksaan.

Sebelum sampai pada bentuk kaidah produksi atau *rule*, pengetahuan yang berhasil didapatkan dari domain tertentu disajikan dalam bentuk tabel keputusan (*decision table*) kemudian dari tabel keputusan dibuat pohon keputusan (*decision tree*). Berikut ini adalah contoh penyajian dalam bentuk tabel keputusan dan pohon keputusan (Hartati dan Iswanti, 2008: 26-39).

Tabel 2.1 Tabel Keputusan

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

Sumber: Hartati dan Iswanti (2008: 32)



Gambar 2.2 Pohon Keputusan
(Sumber: Hartati dan Iswanti, 2008: 32)

Keterangan:

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

Pada pohon keputusan dari gambar 2.2 dapat diketahui bahwa hipotesa H1 terpenuhi jika memenuhi *evidence* A, B, dan C. Hipotesa H2 terpenuhi jika memiliki *evidence* A dan *evidence* E. Hipotesa H3 akan terpenuhi jika memiliki *evidence* A, B, dan E. Hipotesa H4 akan dihasilkan jika memenuhi *evidence* B, C, dan D. Notasi “y” mengandung arti memenuhi *node (evidence)* di atasnya, notasi “t” artinya tidak memenuhi.

Dalam implementasi sistem pakar terutama dalam sesi konsultasi pada sistem pakar, *node-node* yang mewakili *evidence* biasanya akan menjadi pertanyaan yang diajukan oleh sistem. Mengacu pada pohon keputusan dari gambar 2.2 permasalahan dapat saja terjadi pada awal sesi konsultasi yaitu pada saat sistem

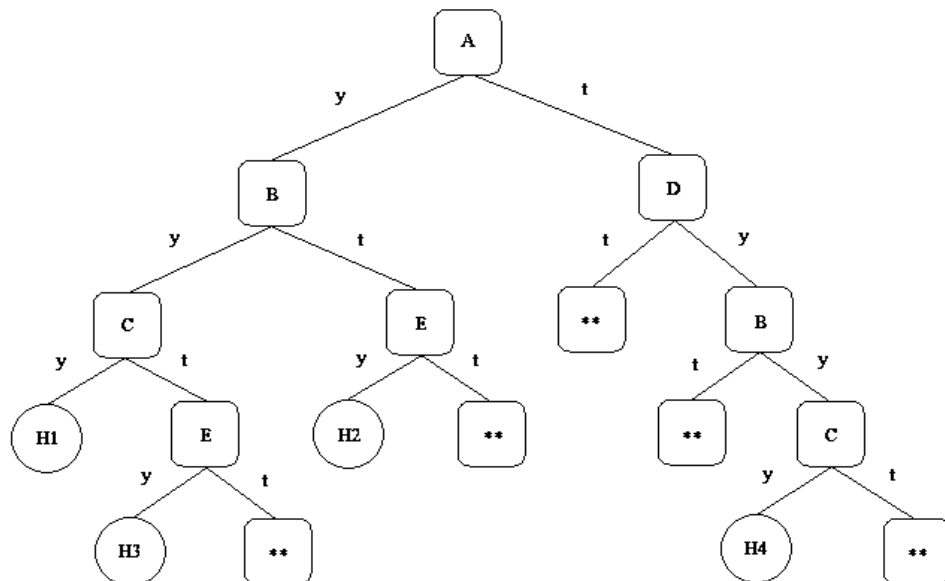
pakar menanyakan “apakah memiliki *evidence A*?”. Pada pohon keputusan, apapun jawaban pengguna baik “ya” atau “tidak” maka sistem akan menanyakan *evidence B*, dengan kata lain apapun jawaban pengguna maka tidak akan mempengaruhi sistem. Salah satu cara untuk mengatasi hal ini adalah dengan mengubah urutan pada tabel keputusan. Alternatif tabel keputusan seperti terlihat pada tabel 2.2.

Tabel 2.2 Alternatif Tabel Keputusan

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

Sumber: Hartati dan Iswanti (2008: 34)

Berdasarkan tabel 2.2 dapat dihasilkan pohon keputusan sebagai berikut:



Gambar 2.3 Alternatif Pohon Keputusan
(Sumber: Hartati dan Iswanti, 2008: 35)

Keterangan:

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

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

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

1. Kaidah 1: *IF A AND B AND C THEN H1*
2. Kaidah 2: *IF A AND B AND E THEN H3*
3. Kaidah 3: *IF A AND E THEN H2*
4. Kaidah 4: *IF D AND B AND C THEN H4*

2.1.6 Web

Menurut Sidik dan Pohan (2009: 1) WWW (*World Wide Web*) lebih dikenal dengan *web* merupakan salah satu layanan yang didapat oleh pengguna komputer yang terhubung ke *internet*. *Web* pada awalnya adalah ruang informasi dalam *internet*, dengan menggunakan teknologi *hypertext*. Pengguna dituntun untuk menemukan informasi dengan mengikuti *link* yang disediakan dalam dokumen *web* yang ditampilkan dalam *browser web*. Kepopuleran *web* sebagai standar *interface*

pada layanan-layanan yang ada di *internet* awalnya hanya sebagai penyedia informasi, sekarang *web* digunakan juga untuk komunikasi melalui *e-mail*, *chatting*, transaksi bisnis (*commerce*), pencarian informasi, dan sebagainya.

Web memudahkan pengguna komputer untuk berinteraksi dengan pelaku *internet* lainnya dan menelusuri informasi di *internet*. Selain itu, *web* telah diadopsi oleh perusahaan sebagai bagian dari strategi teknologi informasinya karena beberapa alasan yaitu (Sidik dan Pohan, 2009: 2):

1. Akses informasi yang mudah
2. *Setup server* lebih mudah
3. Informasi mudah didistribusikan, dan
4. Bebas *platform*, artinya informasi dapat disajikan oleh *browser web* pada sistem operasi apapun karena adanya standar dokumen berbagai tipe data yang disajikan.

2.1.7 Validitas Sistem

Menurut A.S. dan Shalahuddin (2013: 273) validasi mengacu pada sekumpulan aktifitas yang berbeda yang menjamin bahwa sistem atau perangkat lunak yang dibangun dapat ditelusuri sesuai dengan kebutuhan. Pengujian untuk validasi memiliki beberapa pendekatan sebagai berikut (A.S. dan Shalahuddin, 2013: 275-276):

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

Pendekatan ini dilakukan dengan menguji sistem atau perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian

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

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

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

2.2 Variabel

Variabel penelitian pada dasarnya adalah segala sesuatu berbentuk apa saja yang ditetapkan oleh peneliti untuk dipelajari sehingga diperoleh informasi tentang hal tersebut, kemudian ditarik kesimpulannya (Sugiyono, 2014: 38). Dalam penelitian ini yang menjadi variabel utama adalah mesin *fotocopy Ricoh MP 2000le* dan indikator-indikatornya adalah beberapa kerusakan komponen mesin *fotocopy Ricoh MP 2000le*.

2.2.1 Mesin *Fotocopy*

Berdasarkan penelitian Mulyadi, *dkk*, (2009: 60) disebutkan bahwa mesin *fotocopy* adalah alat yang digunakan untuk menyalin atau memindahkan data (*analog*) dalam bentuk teks atau gambar dari satu kertas ke kertas lain menggunakan media sinar laser. Pada awalnya mesin *fotocopy* hanya digunakan oleh perusahaan-perusahaan seperti percetakan dan penerbit buku untuk menggandakan dokumen misalnya dalam proses pembuatan buku.

Menurut Nagumo dan Takagi (1988) dalam Parker (2009: 7) pada mesin *fotocopy* terjadi metode perekaman gambar. Panas (*thermal*) pada sebuah *developing unit* di mesin *fotocopy* membentuk gambar yang tersembunyi (*latent*) pada bahan foto-sensitif yang kemudian ditekan antara *endless belt* dan pemanas *drum* yang harus menghasilkan panas. Untuk mempertahankan suhu yang tepat pada pemanas *drum*, *drum* dan *belt* berputar selama periode pemanasan atau setiap kali selembarnya bahan foto-sensitif digunakan. Alternatif lainnya, *endless belt* dapat dibuat dari bahan yang memiliki kapasitas panas rendah.

Dalam penelitian ini mesin *fotocopy* yang digunakan adalah mesin *fotocopy Ricoh MP 2000le*. Secara garis besar, mesin *fotocopy Ricoh MP 2000le* memiliki kecepatan *copy* sebesar 16/20 *copy* per menit, dengan maksimum ukuran kertas A3, kemampuan *zoom* 50-200%, dan daya listrik 220-240 V.



Gambar 2.4 Mesin *Fotocopy Ricoh MP 2000le*
(Sumber: Data Penelitian, 2017)

Berikut adalah beberapa komponen mesin *fotocopy Ricoh MP 2000le* yang menjadi indikator dalam penelitian ini:

1. *OPC Drum*

Drum adalah jantungnya mesin *fotocopy*. Ketika ada masalah dengan *drum*, maka akan berdampak pada setiap hasil *copy*. Sebagian besar *drum* tidak perlu servis, hanya perlu dibersihkan atau di ganti. *Drum* adalah silinder alumunium dengan dua ujung yang datar. *Drum* baik warna maupun ukurannya memiliki diameter berkisar 1 *inch* hingga 6 *inches*. Permukaan

drum sangat halus serta dilapisi dengan lapisan sangat tipis dari bahan fotokonduktif (Kuaimoku, 1994: 25).



Gambar 2.5 OPC Drum

(Sumber: <http://www.precisionroller.com/img/P8900.wh.jpg>)

2. *Hot Roll (Upper Fusing Roller)*

Hot Roll melelehkan *toner* pada kertas *fotocopy* saat melewati rol atas dan bawah, kemudian menekan *toner* ke dalam serat dari kertas *fotocopy*. *Hot Roll* dipanaskan oleh lampu pemanas dengan suhu tertentu yang cukup panas untuk melelehkan *toner*. *Hot Roll* biasanya terbuat dari aluminium atau baja dan dilapisi dengan bahan anti lengket, biasanya *teflon* atau silikon karet, untuk mencegah *toner* atau kertas *fotocopy* melekat pada *hot roll* (Kuaimoku, 1994: 99).



Gambar 2.6 Hot Roll

(Sumber: <http://www.precisionroller.com/img/R8160.wh.jpg>)

3. *Thermistor*

Thermistor adalah *resistor* yang perubahan pertahanannya sesuai dengan jumlah panas diterapkan. Secara bergiliran *thermistor* melakukan kontrol tegangan di sirkuit. *Thermistor* digunakan dalam sirkuit kontrol suhu untuk memantau suhu komponen lain. Di mesin *fotocopy*, *thermistor* digunakan untuk memantau suhu *fusing rollers (hot roll)* (Kuaimoku, 1994: 101).



Gambar 2.7 *Thermistor*

(Sumber: <http://www.precisionroller.com/img/A6448.jpg>)

4. *Heater Lamp* (Lampu Pemanas)

Secara umum pada mesin *fotocopy*, lampu pemanas terletak di dalam *hot roll*. Lampu pemanas memanaskan *hot roll* hingga suhu spesifik yang dapat mencairkan *toner* pada kertas *fotocopy*. Lampu pemanas berbentuk panjang dan tipis, biasanya lebih panjang dari *hot roll* sekitar 12 *inchies* dan berdiameter 1/8 *inch*. Lampu ini berisi gas seperti *halogen*, dan dapat mencapai suhu sekitar 200°C dalam operasi normal (Kuaimoku, 1994: 101-126).



Gambar 2.8 *Heat Lamp*

(Sumber: http://img.alicdn.com/imgextra/i4/350334905/TB2kfc2qFXXXXcxXpXXXXXXXXXX_!!350334905.jpg)

5. *Charge Roll*

Dalam gelap, *charge roll* memberikan muatan negatif ke *Organic Photo-Conductive (OPC) Drum*. *Charge Roll* tetap berada pada permukaan *drum* karena lapisan *OPC* memiliki hambatan listrik yang tinggi dalam gelap (Ricoh Corporation, 2007: 6-6).



Gambar 2.9 *Charge Roll*

(Sumber: <http://www.precisionroller.com/img/J3236.jpg>)

6. *BICU*

Berdasarkan hasil wawancara (2017), *BICU* berfungsi seperti *motherboard* jika di komputer, hanya saja dimesin *fotococopy Ricoh MP 2000le* ini

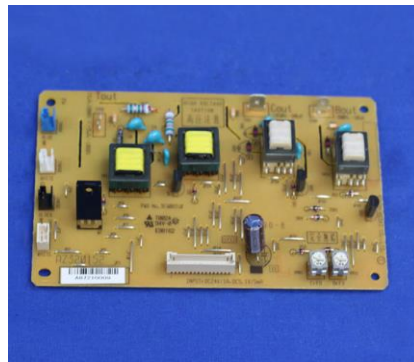
namanya *BICU* atau *Based Image Control Unit*. Fungsinya sesuai dengan namanya, sebagai otaknya mesin *fotocopy* yang mengatur semua fungsi-fungsi dari mesin. *BICU board* juga ini juga berperan sebagai *controller* yang mengatur lampu pemanasnya kapan hidup dan kapan mati.



Gambar 2.10 BICU
(Sumber: Data Penelitian, 2017)

7. *Powerpack*

Berdasarkan hasil wawancara (2017), *power pack* fungsinya untuk menaikkan tegangan yang dibutuhkan untuk *charge roll*. Sesuai namanya yaitu *power pack*, adalah suatu modul elektronik yang fungsinya untuk melipat gandakan tegangan yang dibutuhkan mesin.



Gambar 2.11 Powerpack
(Sumber: <http://www.iteminonline.com/mmITM/Images/AZ320152.jpg>)

8. *Cleaning Blade Drum*

Cleaning Blade terbuat dari bahan karet dan bekerja seperti *wiper* kaca depan mobil. Saat *drum* bekerja, *cleaning blade* membersihkan goresan *toner* dari permukaan *drum*, sehingga membuat *drum* tetap bersih dan bebas dari *toner* (Kuaimoku, 1994: 113).



Gambar 2.12 *Cleaning Blade Drum*
(Sumber: <http://www.precisionroller.com/img/C5440.jpg>)

2.3 Penelitian Terdahulu

Berikut ini merupakan penelitian-penelitian yang berhubungan dengan judul yang diangkat pada penelitian ini, yang digunakan untuk memperkuat dan menambah referensi penelitian di bidang sistem pakar dalam kategori diagnosis:

1. **Hersatoto Listiyono** (2008: 115-124), Jurnal Teknologi Informasi DINAMIK, Nomor 2, Volume 13, ISSN: 0854-9524, dengan judul **Merancang dan Membuat Sistem Pakar**. Suatu sistem pakar adalah suatu sistem komputer yang menyamai kemampuan pengambilan keputusan dari seorang pakar. Suatu emulsi jauh lebih kuat daripada suatu simulasi yang hanya membutuhkan sesuatu yang bersifat nyata dalam beberapa bidang atau hal. *Knowledge* dalam

sistem pakar mungkin saja seorang ahli, atau *knowledge* yang umumnya terdapat dalam buku, majalah dan orang yang mempunyai *knowledge* tentang suatu bidang. Istilah sistem pakar, sistem basis-pengetahuan (*knowledge-base*), atau sistem pakar basis-pengetahuan (*knowledge-base*), sering digunakan dalam arti yang sama.

2. **Jusuf Wahyudi dan Juju Jumadi** (2011: 196-224), Jurnal Media Infotama, Nomor 1, Volume 7, ISSN: 1858-2680, dengan judul **Sistem Pakar Kerusakan *Handphone* Nokia 5130 Xpressmusic Dengan Metode *Forward Chaining***. Metode inferensi yang digunakan pada penelitian ini adalah *forward chaining* dan bahasa pemrograman yang digunakan adalah *Visual Basic* 6.0. Representasi pengetahuan yang digunakan adalah sistem berbasis aturan (*rule based*) atau sistem produksi. Permasalahan kerusakan hanya membahas 10 (Sepuluh) kerusakan yang ditulis pada *rule*. Aplikasi ini dibuat untuk membantu proses diagnosa kerusakan *handphone* Nokia 5310 XpressMusic secara cepat pada PT. Trikonsel Oke, Tbk Cabang Bengkulu.
3. **Imam Gunawan** (2013: 125-139), Jurnal Teknologi Informasi dan Pendidikan, Nomor 2, Volume 6, ISSN: 2086-4981, dengan judul **Perancangan Sistem Pakar Untuk Diagnosis Kerusakan *Hardware Laptop***. Pada Penelitian ini dibuat sistem pakar untuk diagnosis kerusakan *hardware* laptop yang di desain untuk memodelkan atau mengemulasi kemampuan seorang pakar dalam memecahkan suatu masalah yang berbasiskan pada pengetahuan pakar itu sendiri. Metode inferensi yang digunakan pada penelitian ini adalah *forward chaining* dan bahasa pemrograman yang

digunakan adalah *Visual Basic*. Pengetahuan yang di dapat dari proses akuisisi pengetahuan direpresentasikan dengan menggunakan model kaidah produksi dan dalam mendeskripsikan alur program digunakan dua pendekatan, yaitu *flowmap* untuk mendeskripsikan mekanisme kerja aplikasi sistem pakar yang dirancang dan diagram aliran data (*DFD*) untuk mendeskripsikan proses aliran data yang ada dalam aplikasi sistem pakar yang dirancang.

4. **Taufiq dan Yulia Yudihartanti** (2014: 585-600), Jurnal JUTISI, Nomor 3, Volume 3, ISSN: 2089-3787, dengan judul **Model Sistem Pakar Untuk Menganalisa Penyebab Kerusakan *Printer Canon***. Penelitian ini dilakukan untuk membangun sebuah sistem pakar yang dapat mendeteksi kerusakan pada *printer Canon Ip Series Tipe Inkjet* dengan metode metode *forward chaining* menggunakan Pencarian mendalam atau *Depth First Search (DFS)*. Pada algoritma *DFS*, pencarian dilakukan pada satu *node* dalam setiap level dari yang paling kiri. Hasil penelitian ini dapat menganalisa penyebab kerusakan, perbaikan, serta pemeliharaan sehingga *printer* yang di gunakan dapat bekerja sebagaimana mestinya. Hal ini dapat dilihat pada pengujian validitas maupun reliabilitas menunjukkan bahwa pengujian *User Acceptance* ini telah menghasilkan data yang valid dengan nilai *Alpha Cronbach* adalah 0,811 dengan jumlah pertanyaan 5 buah. *Alpha Cronbach* = 0,811 terletak diantara 0,80 hingga 1,00 sehingga tingkat reliabilitasnya sangat reliabel.
5. **Rosmawati Tamin** (2015: 40-44), Jurnal Ilmu Komputer, Nomor 1, Volume 1, ISSN: 2442-4512, dengan judul **Sistem Pakar Diagnosa Kerusakan Pada *Printer Menggunakan Metode Forward Chaining***. Perencanaan sistem

dilakukan dengan membuat *knowledge base* menggunakan *decision tree* dan aturan *if-then* sebagai representasi pengetahuan. Sistem dibuat dengan menggunakan metode *forward chaining* dan bahasa pemrograman *Visual Basic*. Hasil penelitian ini mengungkapkan jenis-jenis kerusakan yang terjadi pada sebuah *printer* serta penanganan dari kerusakan tersebut. Pengujian aplikasi juga dilakukan untuk mengetahui akurasi dan variasi serta *user friendly* dan fleksibilitas sistem. Hasil dari keseluruhan pengujian ini dapat disimpulkan bahwa program sudah cukup baik walaupun jenis kerusakan yang dihasilkan belum lengkap karena pada sistem ini hanya mendeteksi 15 jenis kerusakan mesin secara umum.

6. **Kardiansyah dan Destiani** (2015: 1-6), Jurnal Algoritma Sekolah Tinggi Teknologi Garut, Nomor 1, Volume 12, ISSN: 2302-7339, dengan judul **Perancangan Sistem Pakar Identifikasi Kerusakan Mesin Jahit *Single Needle***. Metode yang digunakan dalam penelitian ini yaitu metode pengembangan sistem pakar dari Durkin (2002) yang terdiri dari *asessment* (penilaian), *knowledge acquisition* (akuisisi pengetahuan), *design* (desain), *test* (pengujian), dan *documentation* (dokumentasi). Sedangkan untuk mesin inferensinya menggunakan metode *forward chaining* (runut maju) dan *fuzzy* untuk pemilihan. Perangkat lunak yang digunakan dalam perancangan aplikasi sistem pakar ini menggunakan bahasa pemrograman *PHP*, *xampp* sebagai *software* pembuatan aplikasi. Pengujian terhadap fungsi-fungsi yang terdapat dalam sistem pakar menunjukan fasilitas *login*, konsultasi, dan pengolahan

basis data telah berjalan. Kesimpulan dari penelitian ini adalah bahwa perancangan sistem pakar ini telah berjalan.

2.4 Software Pendukung

Software pendukung merupakan beberapa perangkat lunak yang digunakan untuk mendukung pembuatan sistem pakar dalam penelitian ini. Perangkat lunak tersebut antara lain: *StarUML* yang dipergunakan untuk membuat *UML*, Bahasa Pemrograman yang digunakan, *Adobe Dreamweaver CS6*, *Web Server XAMPP*, *Database MySQL*, *PhpMyAdmin*, dan *Web Browser*.

2.4.1 *StarUML*

Berdasarkan penelitian Iswari (2015: 73) disebutkan bahwa *StarUML* merupakan proyek *open source* untuk mengembangkan *platform Unified Modeling Language (UML)* atau *Model Driven Architecture (MDA)* yang cepat, fleksibel, dapat diperluas, memiliki banyak fitur, dan tidak dipungut biaya. Tujuan dari proyek ini dalam untuk membangun sebuah perangkat lunak pemodelan dan sekaligus *platform* yang dapat menggantikan perangkat *UML* berbayar lain, seperti *Rational Rose*, *Together*, dan sebagainya. *StarUML* dikembangkan dalam Bahasa Pemrograman *Delphi*. Walaupun begitu, *StarUML* merupakan proyek yang *multi-lingual* dan tidak bergantung pada bahasa pemrograman yang spesifik, sehingga bahasa pemrograman apapun dapat digunakan untuk mengembangkan *StarUML*,

seperti *C/C++*, *Java*, *Visual Basic*, *Delphi*, *Jscript*, *VBScript*, *C#*, *VB.NET*, dan sebagainya.



Gambar 2.13 Logo *StarUML*

(Sumber: http://staruml.io/image/staruml_logo_text_small.png)

2.4.1.1 UML (Unified Modeling Language)

Menurut A.S. dan Shalahuddin (2014: 133) UML (*Unified Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.

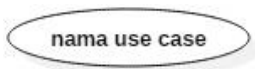
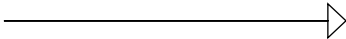
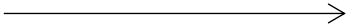
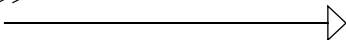
Berikut adalah penjelasan dari beberapa UML yang akan digunakan dalam penelitian ini yaitu *Use case diagram*, *Activity Diagram*, *Sequence Diagram*, dan *Class Diagram*:

1. *Use Case Diagram*

Use case atau diagram *use case* mendeksripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, sedangkan *use case* merupakan fungsionalitas yang disediakan sistem sebagai

unit-unit yang saling bertukar pesan antar unit atau aktor (A.S. dan Shalahuddin, 2014: 155). Berikut adalah simbol-simbol yang ada pada diagram *use case* (A.S. dan Shalahuddin, 2014: 156-158):

Tabel 2.3 Simbol *Use Case Diagram*

Simbol	Deskripsi
<i>Use case</i> 	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 <i>use case</i>
Aktor/actor 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor
asosiasi/association 	Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor
Ekstensi/extend <<extend>> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walaupun tanpa <i>use case</i> tambahan itu. Arah panah mengarah pada <i>use case</i> yang ditambahkan
generalisasi/generalization 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara 2 buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum dari fungsi lainnya. Arah panah mengarah pada <i>use case</i> yang menjadi generalisasinya (umum)
Menggunakan/include/uses <<include>>  <<uses>> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya <i>use case</i> ini. Arah panah mengarah pada <i>use case</i> yang ditambahkan

Sumber: A.S. dan Shalahuddin (2014: 156-158)

2. Activity Diagram

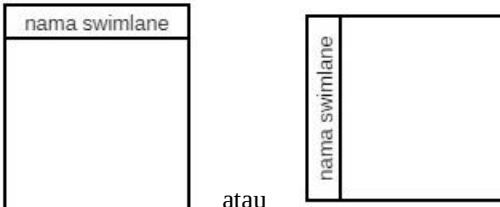
Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Jadi dapat disimpulkan bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor. Berikut adalah simbol-simbol yang terdapat dalam *activity diagram* (A.S. dan Shalahuddin: 2014: 162-163):

Tabel 2.4 Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktifitas memiliki sebuah status awal
Aktifitas 	Aktifitas yang dilakukan sistem, aktifitas biasanya diawali dengan kata kerja
Aktifitas 	Aktifitas yang dilakukan sistem, aktifitas biasanya diawali dengan kata kerja
Percabangan/ <i>decision</i> 	Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu
Penggabungan/ <i>join</i> 	Asosiasi penggabungan dimana lebih dari satu aktifitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktifitas memiliki sebuah status akhir

Sumber: A.S. dan Shalahuddin (2014: 162-163)

Tabel 2.4 Lanjutan

Simbol	Deskripsi
<i>Swimlane</i> 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi

Sumber: A.S. dan Shalahuddin (2014: 162-163)

3. *Sequence Diagram*

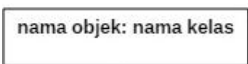
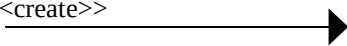
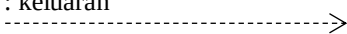
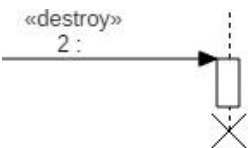
Diagram sekuen atau *sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima antar objek. Dapat disimpulkan bahwa untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek tersebut (A.S. dan Shalahuddin, 2014: 165). Berikut adalah simbol-simbol yang ada pada diagram *sequence diagram* (A.S. dan Shalahuddin, 2014: 165-167):

Tabel 2.5 Simbol *Sequence Diagram*

Simbol	Deskripsi
Aktor/actor 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi tersebut

Sumber: A.S. dan Shalahuddin (2014: 165- 167)

Tabel 2.5 Lanjutan

Simbol	Deskripsi
Garis hidup/<i>lifeline</i> 	Garis hidup menyatakan kehidupan suatu objek
Objek 	Objek menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. Aktor tidak memiliki waktu aktif
Pesan tipe <i>create</i> <<create>> 	Pesan tipe <i>create</i> menyatakan suatu objek membuat objek lain, arah panah mengarah pada objek yang dibuat
pesan tipe <i>return</i> 1 : keluaran 	Pesan tipe <i>return</i> menyatakan suatu objek telah menjalankan suatu operasi atau metode menghasilkan suatu keluaran ke objek tertentu, arah panah megarah pada objek yang menerima keluaran
Pesan tipe <i>destroy</i> <<destroy>> 2 : 	Pesan tipe <i>destroy</i> menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada <i>create</i> maka ada <i>destroy</i>

Sumber: A.S. dan Shalahuddin (2014: 165- 167)

4. Class Diagram

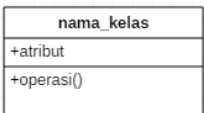
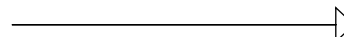
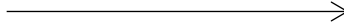
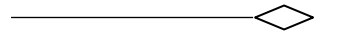
Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas dan operasi atau metode adalah fungsi-

fungsi yang dimiliki oleh suatu kelas (A.S. dan Shalahuddin 2014: 141-142).

Berikut adalah simbol-simbol yang ada pada diagram *class diagram* (A.S.

dan Shalahuddin, 2014: 146-147):

Tabel 2.6 Simbol Class Diagram

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka/interface 	Antarmuka/interface sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek
asosiasi/association 	Asosiasi merupakan relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
Asosiasi berarah/directed 	Asosiasi berarah merupakan relasi antarkelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
Generalisasi/generalization 	Generalisasi merupakan relasi antarkelas dengan makna generalisasi-spesialisasi (umum-khusus)
Kebergantungan/dependency 	Kebergantungan/dependency antarkelas dengan makna kebergantungan antar kelas
Agregasi/Agregation 	Agregasi merupakan relasi antar kelas dengan makna semua-bagian (<i>whole-part</i>)

Sumber: A.S. dan Shalahuddin (2014: 146- 147)

2.4.2 Bahasa Pemrograman

2.4.2.1 *HTML*

Menurut Sidik dan Pohan (2009: 9) *HTML* kependekan dari *Hyper Text Markup Language*. Dokumen *HTML* adalah *file* teks murni yang dapat dibuat dengan *editor* teks apapun. Dokumen ini dikenal sebagai *web page* dan disajikan dalam *browser* *web surfer*, serta umumnya berisi informasi atau *interface* aplikasi di dalam *internet*.

Menurut Saputra (2012: 4) sejak tahun 1990, bahasa pemrograman *HTML* mengalami perkembangan yang cukup pesat dan pada tahun 1996 *World Wide Web Consortium (W3C)* turut mengembangkan *HTML* dan mengeluarkan versi 3.2. Sejak saat itulah *HTML* digunakan sebagai bahasa standar *internet* yang kini dikendalikan oleh *W3C*.

Beberapa elemen wajib yang ada pada *file HTML* apabila akan membangun suatu pondasi kerangka *website* diantaranya (Saputra, 2012: 4-5):

1. Elemen *HTML*

Elemen *HTML* merupakan *tag* dasar apabila hendak memulai suatu dokumen *HTML*. Contoh *tag*-nya adalah **<html>** dan diakhiri dengan **</html>**.

2. Elemen *Head*

Head merupakan *tag* berikutnya setelah **<html>** yang berfungsi untuk menuliskan keterangan tentang dokumen *web* yang akan tampil. Formatnya sebagai berikut:

```

<html>
<head>
</head>
</html>

```

3. Elemen *Title*

Elemen *Title* merupakan suatu elemen yang harus dituliskan di dalam *head* yang digunakan untuk memberikan judul atau informasi pada *caption browser web* tentang topik (tema) atau judul dari suatu dokumen *web* yang ditampilkan pada *browser*. Struktur penggunaannya sebagai berikut:

```

<html>
<head>
<title> Tuliskan judul di sini </title>
</head>
</html>

```

4. Elemen *Body*

Elemen *body* merupakan bagian utama dalam dokumen *web*, berfungsi untuk menampilkan suatu teks atau informasi yang dikenal dengan sebutan konten.

Struktur elemennya adalah sebagai berikut:

```

<html>
<head>
<title> Tuliskan judul di sini </title>
</head>
<body>
Tuliskan konten di sini
</body>
</html>

```

HTML memiliki perintah-perintah dasar yang dapat digunakan untuk mengatur konten, diantaranya:

1. **P (Paragraph)**, berfungsi untuk mengganti paragraf yang diikuti dengan baris kosong di awal dan di akhir paragraf. Perintahnya adalah diawali dengan *tag* `<p>` dan diakhiri dengan *tag* `</p>`.

2. **BR (Line Break)**, berfungsi untuk mengganti baris.

Sintaks: `<br>`

3. **H1, H2, H3, H4, H5, H6 (Header)**, berfungsi untuk membuat *header* dengan ukuran enam jenis berbeda dan tercetak tebal.

Sintaks :

`<Hx>`

...

`</Hx>`

4. **B (Bold)**, berfungsi untuk membuat tampilan teks tercetak tebal.

Sintaks:

`<b> ... </b>`

5. **I (Italic)**, berfungsi untuk membuat tampilan teks tercetak miring.

Sintaks:

`<i> ... </i>`

6. **U (Underline)**, berfungsi untuk membuat tampilan teks tercetak garis bawah.

Sintaks:

`<u> ... </u>`

7. **PRE (Preformatted Text)**, berfungsi untuk menampilkan teks apa adanya.

Sintaks:

`<pre>`

...

`</pre>`

8. **Center**, berfungsi menampilkan teks dengan posisi horizontal tengah.

Sintaks:

```
<center>
...
</center>
```

9. **BaseFont**, berfungsi untuk merubah dasar ukuran huruf dari *web browser*.

Sintaks:

```
<basefont size="pixel">
```

10. **Font**, berfungsi untuk merubah jenis, ukuran, warna, dan tampilan huruf.

Sintaks:

```
<font color="warna" face="font" size="pixel">
...
</font>
```

11. **HR (*Horizontal Rule*)**, berfungsi untuk membuat garis bawah.

Sintaks:

```
<hr>
```

12. **OL (*Ordered List*)**, berfungsi untuk membua nomor daftar urut.

Sintaks:

```
<ol start="number" type="A"|"a"|"I"|"I"|"1">
...
</ol>
```

13. **UL (*Unordered List*)**, berfungsi untuk membuat daftar tanpa nomor urut
(dalam format *bullet*).

Sintaks:

```
<ul type="circle" | "square"|"disc">
...
</ul>
```

14. **LI (*List Item*)**, merupakan isi pada daftar.

Sintaks:

```
<li type="A"|"a"|"I"|"i"|"1"|"circle"|"square"|"disc">
...
```


Menurut Saputra (2012: 17-18) *HTML* 5 merupakan revisi ke-5 dari *HTML* dan masih dalam tahap pengembangan. *HTML* 5 akan menjadi standar pemrograman *web* menggantikan *HTML* versi terdahulu. *HTML* 5 juga dapat ditulis dengan cara *html* ataupun *xhtml*. Berikut adalah fitur-fitur terbaru dalam *HTML* 5:

1. Unsur *canvas* untuk gambar.
2. Bentuk control *form* seperti kalender, tanggal, waktu, *email*, *url*, dan *search*.
3. Elemen konten yang lebih spesifik, seperti artikel, *footer*, *header*, navigasi, dan *section*.
4. Dukungan yang lebih baik untuk penyimpanan secara *offline*.
5. Dan tentunya juga dukungan untuk pemutaran *video* dan *audio*.

Berikut adalah elemen-elemen baru yang tersedia dalam *HTML* 5 (Saputra, 2012: 18):

1. *Section*, serupa H1-H6.
2. *Article*, bisa berupa entri *blog* atau tulisan konten.
3. *Aside*, menyajikan konten pelengkap.
4. *Header*, digunakan untuk judul, deskripsi, bahkan nav untuk navigasi.
5. *Footer*, digunakan untuk bagian bawah *web* yang digunakan untuk menerangkan informasi *copyright* (hak cipta), perusahaan, nama pembuat, kontak, dan sebagainya.
6. *Dialog*, yang dikombinasikan dengan *dt* dan *dd*, digunakan untuk menyajikan percakapan.

7. Dan yang paling mencolok dalam *HTML* 5 adalah adanya penggunaan elemen *figure*, *video*, *audio*, *source*, *embed*, *canvas*, dan elemen-elemen lainnya yang berkaitan dengan *multimedia*.

Selain memberikan elemen-elemen baru, *HTML* 5 juga meniadakan elemen-elemen lama seperti *center*, *font*, *strike*, *u*, *big*, *frame*, *frameset*, dan sebagainya.

2.4.2.2 PHP

PHP atau *Hypertext Preprocessor* merupakan Bahasa pemrograman berbasis *server-side* yang dapat melakukan *parsing script php* menjadi *script web* sehingga dari sisi *client* menghasilkan suatu tampilan yang menarik. *PHP* merupakan pengembangan dari *FI* atau *Form Interface* yang dibuat oleh Rasmus Lerdoff pada tahun 1995. Kode *PHP* sering kali digabungkan dengan kode *HTML*. Untuk membedakannya dengan *HTML*, setiap kode *PHP* ditulis selalu diberi tag pembuka yaitu `<?php` dan pada akhir kode php di beri tag penutup yaitu `?>` (Ardhana, 2014: 65).



Gambar 2.14 Logo *Hypertext Preprocessor (PHP)*

(Sumber: <https://upload.wikimedia.org/wikipedia/commons/thumb/2/27/PHP-logo.svg/1280px-PHP-logo.svg.png>)

PHP memiliki manfaat yang sangat besar bagi pada *web programmer* dan *web developer* pada saat membuat *website* keren yang dinamis seperti membaca *file*, menulis *file*, menampilkan gambar, animasi atau *movie* dan yang paling penting adalah koneksi terhadap *database* seperti MySQL. Berikut adalah beberapa keunggulan dari Bahasa pemrograman PHP (Ardhana, 2014: 66):

1. *PHP* memiliki *native API* untuk koneksi ke berbagai *database*, sehingga secara otomatis dalam melakukan koneksi lebih cepat dibandingkan dengan *Open Database Connectivity (ODBC)*.
2. Eksekusi *scripting* dilakukan lebih cepat sehingga meningkatkan *throughout* pada *server*.
3. Penulisan program yang sederhana yang membuat *programmer newbie* (pemula) mudah dalam memahami *PHP*.
4. Dukungan koneksi yang hampir bisa dilakukan ke semua *database* seperti *MySQL, PostgreSQL, Sybase, Informix, Interbase, ORACLE, SQL Server*, dan lain-lain.
5. *PHP* dapat dijalankan dibeberapa *web server* seperti *PWS, IIS, Apache, Xitami, Netscape Enterprise, AOL Server* dan *Orelly Website Pro, CGI* dan *ISAPI*.
6. *PHP* juga dapat berjalan di berbagai *platform* seperti *UNIX* dan *Windows*.
7. *PHP* dapat didistribusikan kembali dibawah lisensi *GNU Public License (GPL)* karena bersifat *open source*.

2.4.2.3 Bootstrap Twitter

Menurut Cochran (2012: 2) *Twitter bootstrap* adalah *open-source front-end toolkit* yang dikembangkan oleh Mark Otto dan Jacob Thornton yang bekerja di perusahaan yang berkecimpung dalam bidang sosial media yaitu *Twitter* pada Agustus 2011. *Twitter Bootstrap* dikembangkan dengan tujuan untuk membantu desainer dan pengembang *website* dalam membangun *front-end* sebuah *website* secara cepat dan efisien.



Gambar 2.15 Logo *Bootstrap*

(Sumber: <https://v4-alpha.getbootstrap.com/assets/brand/bootstrap-social-logo.png>)

Pada penelitian ini *bootstrap* berperan seperti *CSS (Cascading Style Sheet)* untuk memberikan efek desain *web* yang lebih menarik, dengan kelebihan tanpa harus membuat kode programnya secara manual karena telah tersedia di <http://getbootstrap.com/> serta situs penyedia *template bootstrap* seperti <https://startbootstrap.com/>.

2.4.3 Adobe Dreamweaver CS6

Menurut Madcoms (2013: 1-2) *Adobe Dreamweaver CS6* adalah perangkat lunak terkemuka untuk desain *web* yang menyediakan kemampuan visual yang

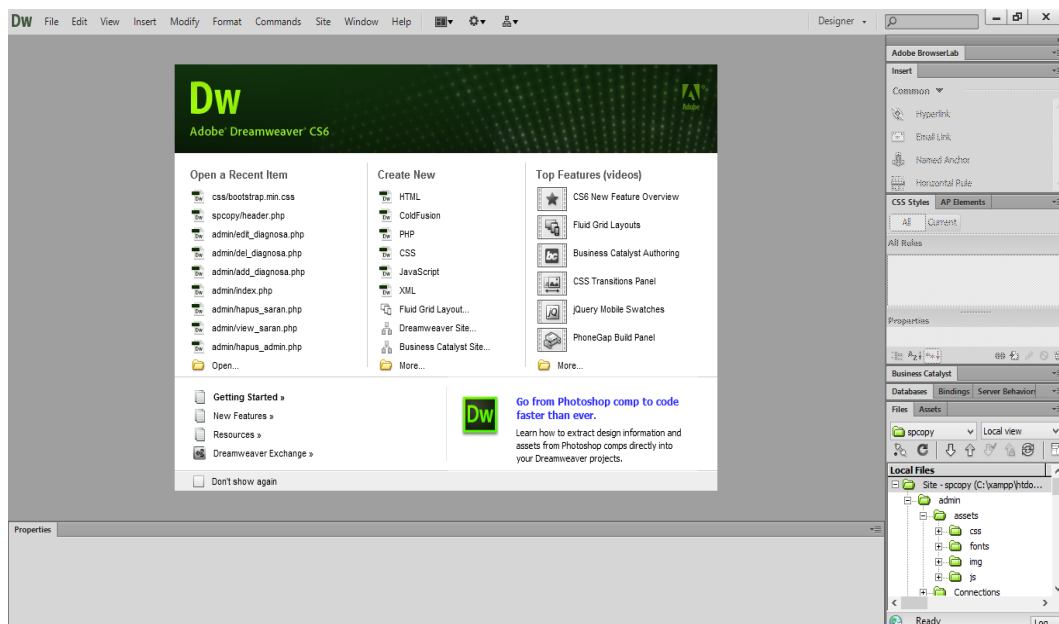
intuitif termasuk pada tingkat kode, yang dapat digunakan untuk membuat dan mengedit *website HTML* serta aplikasi *mobile* seperti *smartphone*, *tablet*, dan perangkat lainnya. Dengan adanya fitur *layout Fluid Grid* yang dirancang khusus untuk memungkinkan lintas *platform*, maka akan membuat *layout* adaptif atau dapat menyesuaikan dengan *browser* yang dipakai. *Dreamweaver* menjadi *web* desain standar dan alat pengembangan untuk banyak organisasi, dan *Dreamweaver CS6* dibutuhkan untuk merespon dengan cara *web* telah berubah, serta cara berubah.

Fitur-fitur dalam *Dreamweaver CS6* antara lain: *Layout fluid grid*, peningkatan kinerja *FTP* karena didukung oleh *FTPS* dan *FTPeS*, integrasi dengan *Adobe Business Catalyst*, peningkatan *support jQuery Mobile*, *support PhoneGap* diperbarui, transisi *CSS3* dan *HTML5*, fitur *Live View* diperbarui, panel *Multiscreen Preview* diperbarui, integrasi *Adobe BrowserLab*, mendukung integrasi *CMS*, isyarat kode yang lebih spesifik, integrasi *Adobe Creative Suite*, komunitas *Dreamweaver* yang semakin diperluas, didukung oleh teknologi terkemuka termasuk *HTML*, *XHTML*, *CSS*, *JavaScript*, *Ajax*, *PHP*, perangkat lunak *Adobe ColdFusion*, dan *ASP*; selalu menjadi yang terdepan dengan *World Wide Web Consortium (W3C)* validasi, mendukung *Subversion*, inspeksi atau pengecekan *CSS* dengan dukungan *CSS* yang komprehensif, bantuan pengkodean yang cerdas dengan isyarat kode *custom class PHP*, *setup* situs yang sederhana, *CSS stater pages*, dan terintegrasi dengan konten *FLV* (Madcoms, 2013: 4-11).



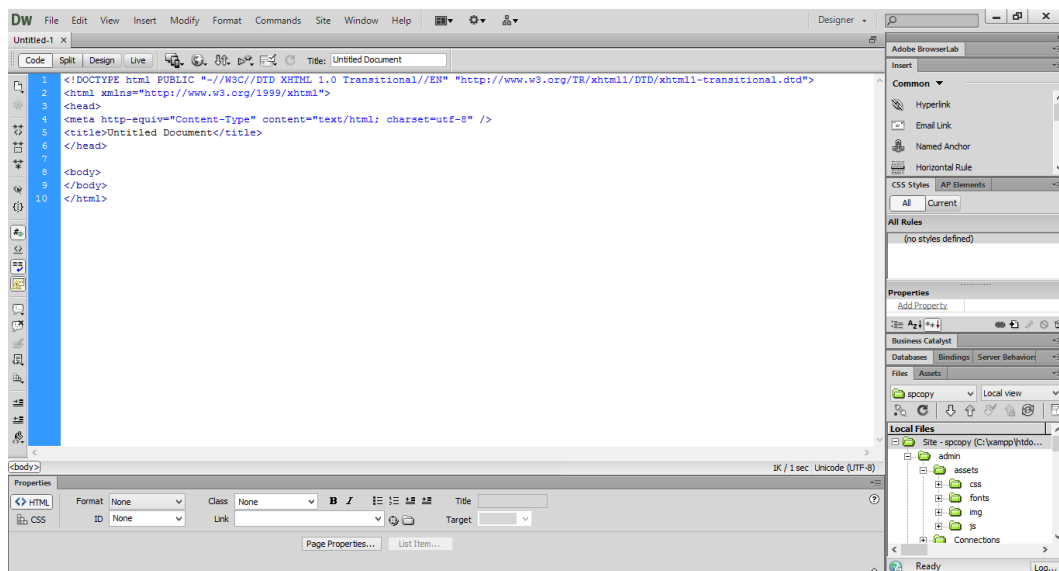
Gambar 2.16 Logo *Adobe Dreamweaver CS6*
(Sumber: https://upload.wikimedia.org/wikipedia/commons/7/72/Adobe_Dreamweaver_CS6_Icon.png)

Berikut adalah tampilan yang akan terlihat pertama kali ketika membuka *Adobe Dreamweaver CS6*:



Gambar 2.17 Halaman awal *Adobe Dreamweaver CS6*
(Sumber: Data Penelitian, 2017)

Berikut adalah tampilan halaman kerja (*worksheet*) dengan *mode designer* pada *Adobe Dreamweaver CS6*, ini merupakan tampilan halaman kerja yang digunakan untuk membuat kode program pada penelitian ini.



Gambar 2.18 Halaman kerja *Mode Designer* pada *Adobe Dreamweaver CS6*

(Sumber: Data Penelitian, 2017)

Tampilan halaman kerja pada *Adobe Dreamweaver CS6* dapat diubah sesuai keinginan pemrogram, dengan banyak pilihan *mode* seperti *APP Developer*, *APP Developer Plus*, *Business Catalyst*, *Clasic*, *Coder*, *Coder Plus*, *Designer*, *Designer Compact*, *Dual Screen*, *Fluid Layout*, dan *Mobile Application*.

2.4.4 XAMPP

Menurut Madcoms (2015: 269) *web server* adalah suatu program komputer yang mempunyai tanggung jawab atau tugas menerima *HTTP* dari komputer klien, yang dikenal dengan nama *web browser*, dan melayani klien dengan menyediakan respon *HTTP* berupa konten data, biasanya berupa halaman *web* yang terdiri dari dokumen *HTML*, dan objek terkait seperti gambar dan lain-lain. *XAMPP* adalah sebuah paket kumpulan *software* yang terdiri dari *Apache*, *MySQL*,

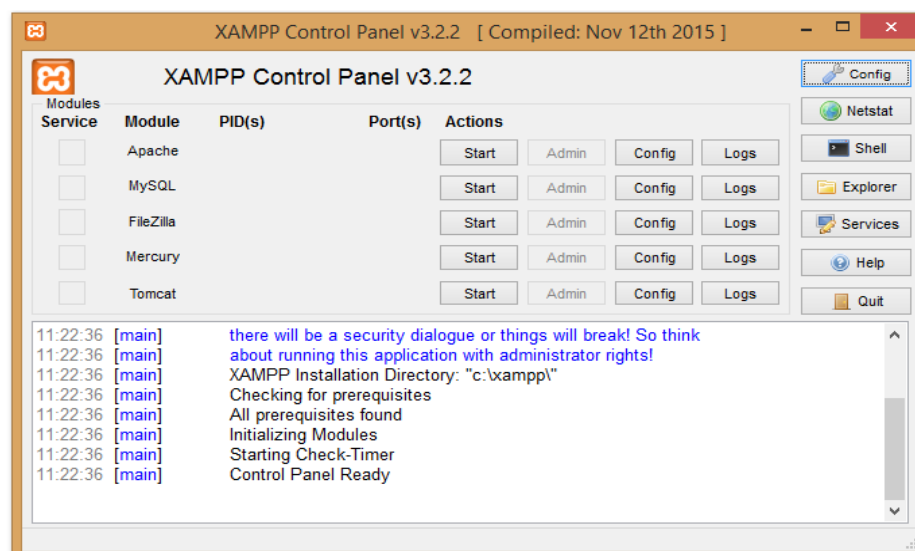
PhpMyAdmin, PHP, Perl, Filezilla dan lain-lain (Madcoms, 2016: 186). *XAMPP* berfungsi untuk memudahkan instalasi lingkungan *PHP*, di mana biasanya lingkungan pengembangan *web* memerlukan *PHP, Apache, MySQL*, dan *PhpMyAdmin* serta *software-software* yang terkait dengan pengembangan *web*. Sehingga tidak perlu menginstall aplikasi-aplikasi tersebut satu persatu.



Gambar 2.19 Logo *XAMPP*

(Sumber: <https://wiki.bitnami.com/@api/deki/files/527/=xampp-logo.jpg>)

Berikut adalah tampilan yang akan terlihat pertama kali ketika membuka *XAMPP*:



Gambar 2.20 *XAMPP Control Panel*

(Sumber: Data Penelitian, 2017)

Untuk menggunakan paket *software database* dapat dilakukan dengan mengaktifkan terlebih dahulu *Apache* dan *MySQL* (klik *start*) kemudian klik tombol *Admin* pada *MySQL* untuk mulai mengakses *database MySQL* dengan *PhpMyAdmin* pada *web server XAMPP*.

2.4.5 Basis data (*Database*)

Menurut A.S. dan Shalahuddin (2014: 43-44) sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Pengertian lain dari basis data (*database*) adalah sekumpulan data yang diorganisasikan untuk melayani berbagai aplikasi secara efisien dengan memusatkan data dan mengurangi penggandaan data (Laudon dan Laudon, 2015: 224).

Salah satu bentuk basis data yang dibutuhkan dalam sebuah sistem yaitu *Database Management System (DBMS)*. *DBMS* adalah perangkat lunak yang memungkinkan suatu organisasi memusatkan data, mengelola data secara efisien, dan menyediakan akses terhadap data yang disimpan oleh program aplikasi. *DBMS* bertindak sebagai antarmuka antara program aplikasi dan *file* data secara fisik (Laudon dan Laudon, 2015: 224). Syarat minimal dari *DBMS* antara lain (A.S. dan Shalahuddin, 2014: 44-45):

1. Menyediakan fasilitas untuk mengelola akses data
2. Mampu menangani integritas data

3. Mampu menangani akses data yang dilakukan secara bersamaan
4. Mampu menangani *backup* data

Beberapa DBMS yang paling banyak digunakan saat ini antara lain:

1. *DBMS* versi komersial, yaitu *Oracle*, *Microsoft SQL Server*, *IBM DB2*, dan *Microsoft Access*
2. *DBMS* versi *open source*, yaitu *MySQL*, *PostgreSQL*, *Firebird*, dan *SQLite*.

Basis data juga memiliki alur hidup atau *Database Life Cycle (DBLC)*. Fase-fase *DBLC* antara lain (A.S. dan Shalahuddin, 2014: 48-64):

1. Analisis kebutuhan (*requirement analysis*), yaitu tahap yang didefinisikan dengan mewawancarai produsen dan pemakai data, kemudian membuat kontrak spesifikasi basis data dan membuat *Entity Relationship Diagram (ERD)*. *ERD* dikembangkan berdasarkan teori himpunan dalam bidang matematika. *ERD* digunakan untuk pemodelan basis data relasional.
2. Desain logik basis data (*logical database design*), pada tahap ini harus dibuat rancangan logik basis data, seperti *Conceptual Data Model (CDM)*. *CDM* merupakan konsep yang berkaitan dengan pandangan pemakai terhadap data yang disimpan dalam basis data. *CDM* dibuat sudah dalam bentuk tabel-tabel tanpa tipe data yang menggambarkan relasi antar tabel untuk keperluan implementasi ke basis data.
3. Desain fisik basis data (*physical database design*), pada tahap ini harus dibuat rancangan fisik basis data, seperti *Physical Data Model (PDM)*. *PDM* adalah model yang menggunakan sejumlah tabel untuk menggambarkan data serta hubungan antar data. Setiap tabel mempunyai sejumlah kolom dan setiap

kolom memiliki nama yang unik beserta tipe datanya. *PDM* sudah merupakan bentuk fisik perancangan basis data yang sudah siap diimplementasikan ke dalam *DBMS* sehingga nama tabel juga sudah merupakan nama asli tabel yang akan diimplementasikan ke dalam *DBMS*.

4. Implementasi, yaitu tahap pembuatan *Quesry SQL* dan aplikasi ke *DBMS* atau *file*.

2.4.5.1 MySQL

Menurut Ardhana (2014: 45) *MySQL* dikembangkan oleh pengembang dan konsultan *database* bernama *MYSQL AB* sekitar tahun 1994 di Swedia. Tujuan dikembangkan *MySQL* yaitu untuk mengembangkan aplikasi berbasis *web* pada *client*. Michael Monty Widenius melakukan pengembangan *MySQL AB* yang memiliki sebuah aplikasi *UNIREG* dan *ISAM* buatannya yang pada waktu itu ingin membuat tampilan antar muka *user interface SQL (Structure Query Language)* yang cocok untuk diimplementasikan. Pada saat itu Monty telah menggunakan *user interface* buatan David Hughes pada aplikasi *SQL*-nya dengan nama *mini SQL* atau disingkat *mSQL*.

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 pengguna (*multi-user*) dan dapat melakukan suatu proses secara sinkron atau bersamaan (*multi-threaded*). *MySQL* banyak digunakan di berbagai kalangan untuk melakukan penyimpanan dan pengolahan data, mulai dari kalangan akademis sampai ke industri, baik industri

kecil, menengah, maupun besar. Lisensi *MySQL* terbagi menjadi dua, *MySQL* sebagai produk *open source* (gratis) di bawah *GNU (General Public License)* atau dapat membeli lisensi dari versi komersialnya (Raharjo, 2015:16).

Menurut Ardhana (2014: 46) beberapa keistimewaan yang dimiliki oleh *MySQL*, antara lain:

1. Portabilitas

MySQL dapat berjalan stabil pada berbagai sistem operasi seperti *Windows*, *Linux*, *FreeBSD*, *Mac Os X Server*, *Solaris*, *Amiga*, 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 pengguna dengan sistem perizinan yang mendetail serta 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 milyar 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 Socket (UNIX)* dan atau *Named Pipes (NT)*.

10. Lokalisasi

MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meskipun demikian, Bahasa Indonesia belum termasuk di dalamnya.

11. Antarmuka

MySQL memiliki antarmuka (*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* ataupun *Oracle*.

2.4.5.2 *PhpMyAdmin*

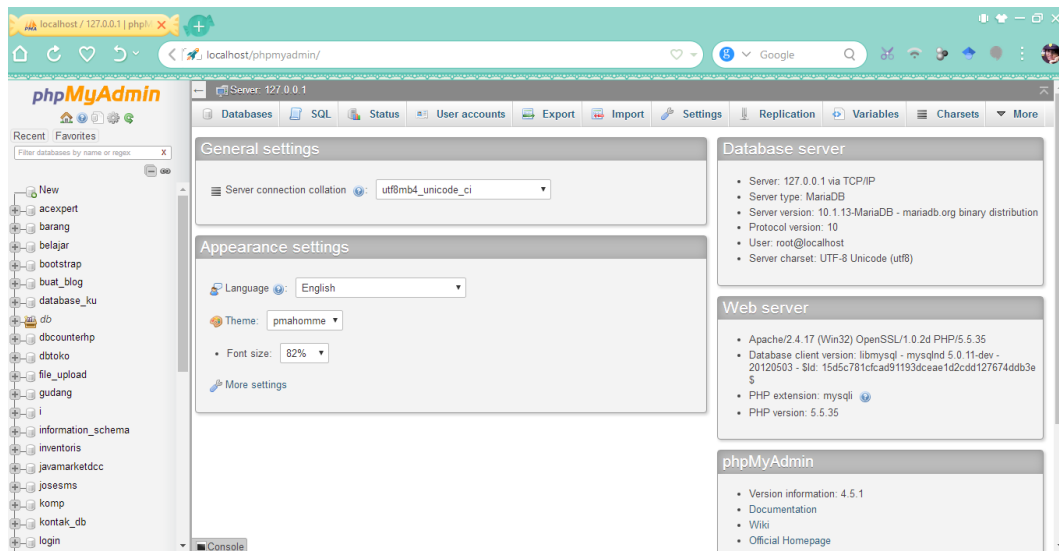
Penelitian ini menggunakan *PhpMyAdmin* yang terdapat dalam paket *XAMPP* untuk membuat *database MySQL*. Menurut Madcoms (2016: 186) *PhpMyAdmin* adalah sebuah aplikasi *open source* yang berfungsi untuk memudahkan manajemen *MySQL*. Dengan menggunakan *PhpMyAdmin*, pengguna (*user*) dapat membuat *database*, membuat tabel, meng-*insert*, menghapus dan meng-*update* data dengan *GUI* dan terasa lebih mudah, tanpa perlu mengetikkan perintah *SQL* secara manual.



Gambar 2.21 Logo *phpMyAdmin*

(Sumber: <https://www.phpmyadmin.net/static/images/logo-og.png>)

Berikut adalah tampilan yang akan terlihat pertama kali pada *web browser* saat mengakses *database MySQL* dengan *PhpMyAdmin* pada *web server XAMPP* dengan mengklik tombol *Admin*.



Gambar 2.22 Halaman Utama *phpMyAdmin*
(Sumber: Data Penelitian, 2017)

Selain klik tombol *Admin*, *PhpMyAdmin* juga dapat diakses langsung dengan mengetikkan *localhost/phpmyadmin* pada *Address bar web browser*.

2.4.6 Browser Web

Browser web adalah *software* yang digunakan untuk menampilkan informasi dari *server web*. *Software* ini dikembangkan dengan menggunakan *user interface* grafis, sehingga pemakai dapat dengan mudah melakukan ‘*point dan click*’ untuk pindah antar dokumen (Sidik dan Pohan, 2009: 5).

Beberapa jenis *browser web* yang umum digunakan adalah *Internet Explorer*, *Google Chrome*, *Mozilla Firefox*, *UC Browser* dan *safari*.

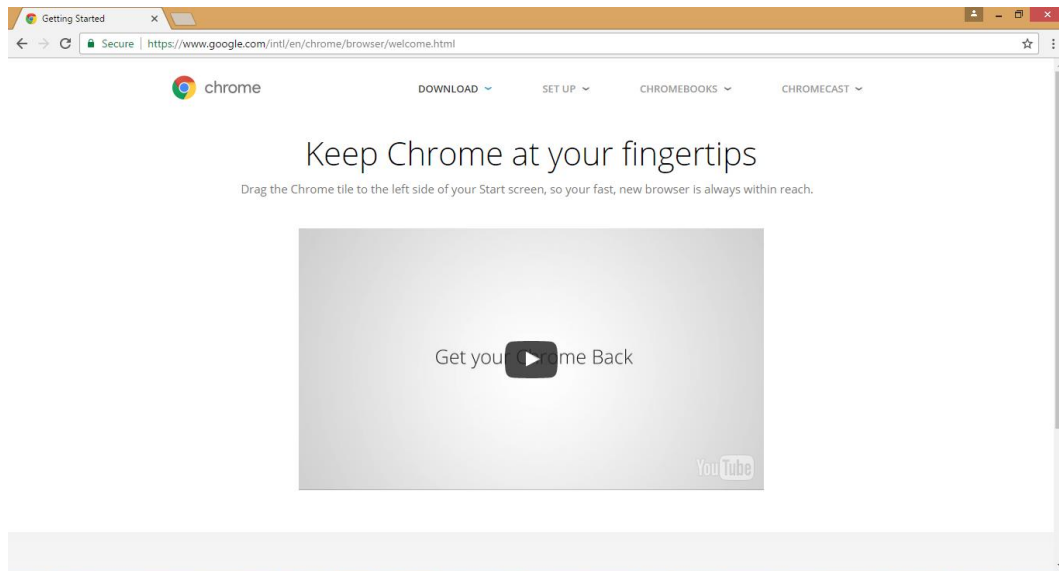
2.4.6.1 *Google Chrome*

Pada penelitian ini *browser web* yang digunakan adalah *Google Chrome*. Menurut Enterprise (2009: 1) *Google Chrome* adalah sebuah *web browser* yang dikembangkan oleh *Google* menggunakan mesin rendering *WebKit*. Mesin ini menerapkan sebuah kerangka kerja yang membuat para *programmer* mampu mendesain *browser* yang bekerja sama baiknya pada komputer pribadi maupun perangkat bergerak. *Google Chrome* pertama kali dirilis tanggal 2 September 2008 dalam versi beta untuk *Windows XP* dan *Vista*



Gambar 2.23 Logo *Google Chrome*
(Sumber: Enterprise, 2009: 3)

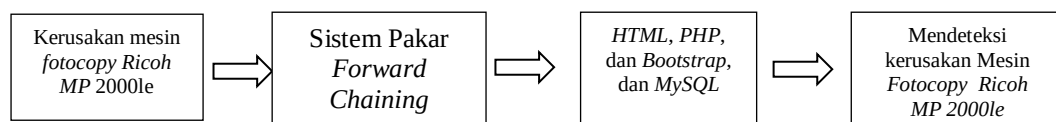
Berikut adalah tampilan yang akan terlihat pada halaman awal (*getting started*) *web browser Google Chrome*:



Gambar 2.24 Halaman *Getting Started Google Chrome*
(Sumber: Data Penelitian, 2017)

2.5 Kerangka Pemikiran

Kerangka pemikiran memuat pemikiran terhadap alur yang dipahami sebagai acuan dalam pemecahan masalah yang diteliti secara logis dan sistematis. Kerangka berfikir yang baik akan menjelaskan secara teoritis pertautan antar variabel yang diteliti (Sugiyono, 2014: 60). Berikut adalah kerangka berpikir dari “*Sistem Pakar Untuk Mendeteksi Kerusakan Mesin Fotocopy*”:



Gambar 2.25 Kerangka Pemikiran
(Sumber: Data Penelitian, 2017)

Data-data yang dibutuhkan berkaitan dengan kerusakan mesin *fotocopy Ricoh MP 2000le* dianalisis terlebih dahulu agar lebih sederhana dan mudah dilakukan

proses pengolahan datanya. Data-data tersebut kemudian diolah menggunakan metode sistem pakar *forward chaining* untuk membuat aturan (*rule*) yang akan digunakan. Sistem pakar dengan metode *forward chaining* dibuat menggunakan bahasa pemrograman *HTML*, *PHP* dan *Bootstrap* serta *database MySQL* sehingga menghasilkan sebuah sistem pakar untuk mendeteksi kerusakan mesin *fotocopy Ricoh MP 2000le* menggunakan metode *forward chaining* berbasis *web*.