BAB II
TINJAUAN PUSTAKA

2.1. Teori Dasar

2.1.1. Kecerdasan Buatan (Artificial Intelligence)

Menurut (Budiharto & Suhartono, 2014:2) kecerdasan buatan atau *Artificial Intelligence* (AI) merupakan bidang ilmu komputer yang mempunyai peran penting di era kini dan masa akan datang. Bidang ini telah berkembang sangat pesat di 20 tahun terakhir seiring dengan pertumbuhan kebutuhan akan perangkat cerdas pada industri dan rumah tangga. Buku ini akan memaparkan berbagai pandangan modern dan hasil riset terkini yang perlu dikuasai oleh para akademis, mahasiswa, pelajar, dan praktisi.

Tujuan dari kecerdasan buatan adalah membuat mesin menjadi pintar (tujuan utama), memahami apa itu kecerdasan (tujuan ilmiah) dan membuat mesin lebih bermanfaat (tujuan entrepreneurial) (Sutojo, Mulyanto, & Suhartono, 2011:3). Berdasarkan definisi ini, 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.

Menurut Turban dalam (Sutojo et al., 2011:10.11) jika dibanding dengan kecerdasan buatan memiliki keuntungan komersial, antara lain:

1. Kecerdasan buatan lebih bersifat permanen. Kecerdasan alami akan cepat mengalami perubahan. Kemampuan kecerdasan buatan tidak akan pernah berubah selama programnya tidak diubah oleh *programmer*. Berbeda dengan kecerdasan alami. Karena sifat manusia yang subjektif, pelupa dan makin lama makin tua hingga kemampuan berpikirnya berkurang seiring bertambahnya waktu, kemampuan kecerdasan alami cenderung tidak permanen.
2. Kecerdasan buatan lebih mudah diduplikasi dan disebarkan. Misalnya saja pemerintah membutuhkan 10.000 orang pakar penyakit jantung untuk ditempatkan di seluruh Indonesia. Bayangkan kalau pemerintah harus menyekolahkan anak bangsa sejumlah 10.000 orang, mulai dari SD sampai lulus sarjana kedokteran spesialis penyakit jantung. Waktu yang dibutuhkan minimal 20 tahun. Jika biaya pendidikan 1 orang Rp100 juta, maka untuk 10.000 orang biaya yang diperlukan adalah Rp1 triliun. Belum lagi kendala jika orangnya meninggal sebelum menjadi sarjana. Kalau sudah jadi sarjana, pemerintah akan kesulitan untuk mendistribusikan mereka ke pelosok-pelosok Indonesia. Hal ini sangat tidak efisien, Sementara itu, untuk kecerdasan buatan, pemerintah cukup membuat 1 sistem pakar penyakit jantung dengan waktu yang relatif lebih cepat dan biaya yang jauh lebih murah. Proses duplikasi dan pendistribusiannya ke seluruh pelosok tanah air pun sangat mudah.
3. Kecerdasan buatan lebih murah dibanding kecerdasan alami. Hal ini tergambar seperti dijelaskan dalam nomor 2.

4. Kecerdasan buatan bersifat konsisten. Misalnya saja telah dibuat sistem pakar hakim pengadilan untuk mengadili kasus-kasus pidana di Indonesia. Untuk kasus yang sama solusi dan keputusan yang dibuat oleh kecerdasan buatan tidak pernah berubah. Berbeda dengan kecerdasan alami. Arti kata hakim bisa berubah menjadi “Hubungi Aku Kalau Ingin Menang”. Untuk kasus yang sama, solusi dan keputusan yang dibuat oleh kecerdasan alami bisa berubah-ubah tergantung orang yang terkena kasus telah menghubungi hakim atau tidak.
5. Kecerdasan buatan dapat didokumentasi. Solusi dan keputusan yang dibuat oleh kecerdasan buatan dapat didokumentasi dengan mudah karena disimpan dalam *hard disk* dan pencarian datanya relatif lebih mudah dilacak. Sedangkan untuk kecerdasan alami, hal ini sangat sulit dilakukan.
6. Kecerdasan buatan dapat mengerjakan pekerjaan lebih cepat dibanding dengan kecerdasan alami. Tentu saja karena kecepatan berpikir dari sebuah prosesor jauh lebih cepat dibanding kecepatan berpikir dari otak manusia.
7. Kecerdasan buatan dapat mengerjakan pekerjaan lebih baik dibanding dengan kecerdasan alami.

Sementara itu, kecerdasan alami memberikan keuntungan sebagai berikut.

1. Kreatif. Pengetahuan seorang manusia selalu bertambah seiring dengan perkembangan waktu. Sifat bosan manusia pun mengakibatkan ia harus berpikir kreatif untuk mencari solusi-solusi terbaru. Berbeda dengan kecerdasan buatan, penambahan pengetahuan harus dilakukan pada sistem yang telah dibangun.

2. Kecerdasan alami memungkinkan orang menggunakan pengalaman secara langsung. Sedangkan pada kecerdasan buatan harus bekerja dengan input-input simbolik.
3. Pemikiran manusia dapat digunakan secara luas, sedangkan kecerdasan buatan sangat terbatas.

2.1.2. Sistem Pakar (*Expert System*)

Menurut (Budiharto & Suhartono, 2014:132.133) sistem pakar adalah program komputer yang menyimulasi penilaian dan perilaku manusia atau organisasi yang memiliki pengetahuan dan pengalaman ahil dalam bidang tertentu. Biasanya, sistem seperti ini berisi basis pengetahuan yang berisi akumulasi pengalaman dan satu set aturan untuk menerapkan pengetahuan dasar untuk setiap situasi tertentu. Sistem pakar yang canggih dapat ditingkatkan dengan penambahan basis pengetahuan atau set aturan. Di antara banyak sistem pakar yang ada, yang terkenal adalah aplikasi bermain catur dan sistem diagnosis medis. Definisi sistem pakar yang paling dikenal adalah:

a. Menurut Durkin : Sistem pakar adalah suatu program komputer yang dirancang untuk memodelkan kemampuan penyelesaian masalah yang dilakukan seorang pakar.

b. Menurut Ignizio : Sistem pakar adalah suatu model dan prosedur yang berkaitan, dalam suatu domain tertentu, yang mana tingkat keahliannya dapat dibandingkan dengan keahlian seorang pakar.

c. Menurut Giarratano dan Riley: Sistem pakar adalah suatu sistem komputer yang bisa menyamai atau meniru kemampuan seorang pakar

2.1.2.1. Konsep Sistem Pakar

Sistem pakar memiliki 6 konsep dasar yaitu (Sutojo, 2011: 163):

1. Kepakaran (*Expertise*)

   Kepakaran merupakan suatu pengetahuan yang di peroleh dari pelatihan, membaca, dan pengalaman. Kepakaran inilah yang memungkinkan para ahli dapat mengambil keputusan lebih cepat dari dapat mengambil keputusan lebih cepat daripada seorang yang bukan pakar.

2. Pakar (*Expert*)

   Pakar adalah seseorang yang mempunyai pengetahuan, pengalaman, dan metode khusus serta mampu menerapkannya untuk memecahkan masalah atau memberi nasehat.

3. Pemindahan Kepakaran (*Transferring Expertise*)

   Tujuan sistem pakar adalah memindahkan kepakaran dari seorang pakar ke dalam komputer, kemudian di transfer kepada orang lain yang bukan pakar.

4. Inferensi (*Inferencing*)

   Infrensi adalah sebuah prosedur (*program*) yang mempunyai kemampuan dalam melakukan penalaran. Inferensi di tampilkan pada suatu komponen di sebut mesin inferensi yang mencakup prosedur-prosedur mengenai pemecahan masalah.

5. Aturan-aturan (*Rule*)

   Kebanyakan *software* sistem pakar komersial adalaah sistem berbasis *rule* (*rule-based system*), yaitu pengetahuan disimpan terutama dalam bentuk *rule*, sebagai prosedur-prosedur pemechaan masalah

6. Kemampuan menjelaskan (*Explanation Capability*)

   Fasilitas lain dari sistem pakar adalah kemampuannya untuk menjelaskan saran atau rekomendasi yang berikannya. Penjelasan ini di lakukan dalam subsistem yang di sebut subsistem penjelasan

2.1.2.2. Keuntungan Sistem Pakar

Secara garis besar, banyak manfaat yang dapat di ambil dengan adanya sistem pakar, antara lain (Kusumadewi, 2003:110):

1. Memungkinkan orang awam bisa mengerjakan pekerjaan para ahli.
2. Bisa melakukan proses secara berulang secara otomatis.
3. Menyimpan pengetahuan dan keahlian pakar.
4. Meningkatkan output dan produktivitas.
5. Meningkatkan kualitas.
6. Mampu mengambil keputusan dan melestarikan keahlian para pakar (terutama yang termasuk keahlian langka).
7. Mampu beroperasi dalam lingkungan yang berbahaya.
8. Memiliki kemampuan untuk mengakses pengetahuan.
9. Memiliki realiabilitas.
10. Meningkatkan kapabilitas sistem komputer.

11. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian
12. Sebagai media pelengkap dalam pelatihan

2.1.2.3. Kelemahan Sistem

Di samping memiliki beberapa keuntungan, sistem pakar juga memiliki beberapa kelemahan, antara lain (Kusumadewi, 2003:111):

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

2.1.2.4. Bentuk Sistem Pakar

Ada 4 bentuk sistem pakar, yaitu (Kusumadewi, 2003:113);

1. Berdiri sendiri. Sistem pakar jenis ini merupakan *software* yang berdiri-sendiri tidak tergabung dengan *software* yang lainnya. Semua contoh sistem pakar pada Tabel 2.1 merupakan sistem pakar jenis ini.
2. Tergabung. Sistem pakar jenis ini merupakan bagian program yang terkandung di dalam suatu algoritma (konvensional), atau merupaka program dimana di dalamnya memanggil algoritma subrutin lain (konvensional)
3. Menghubungkan ke *software* Lain, bentuk ini biasanya merupakan sistem pakar yang menghubungkan ke suatu paket program tertentu, misalnya dengan DBMS

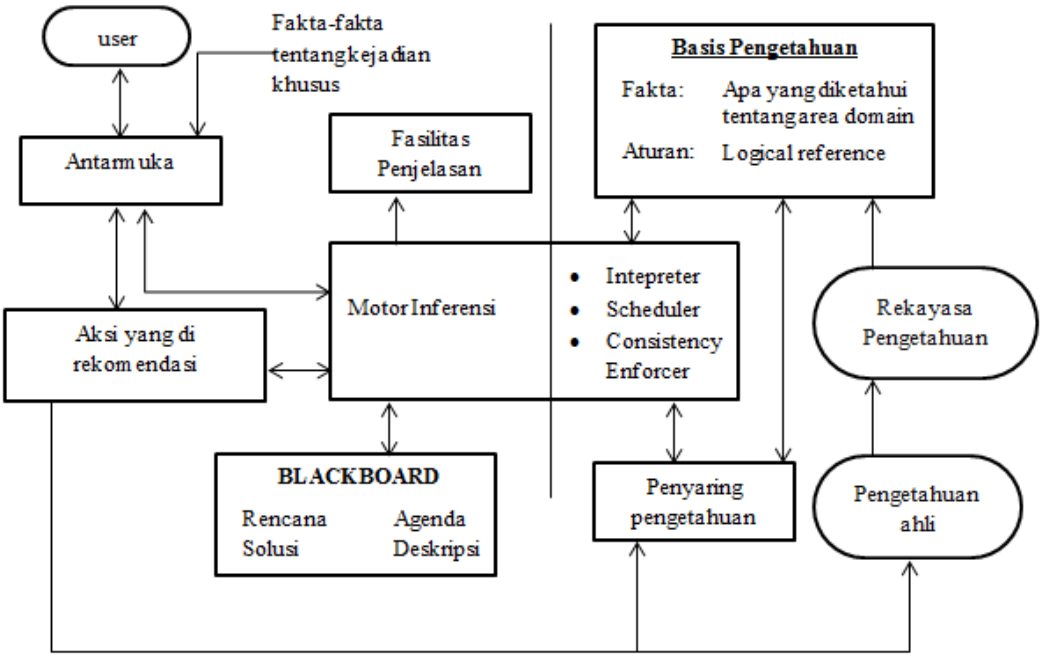
4. Sistem Mengabdi. Sistem pakar merupakan bagian dari komputer khusus yang di hubungkan dengan suatu fungsi tertentu. Misalnya sistem pakar yang di gunakan untuk membantu menganalisis data radar

Tabel 2.1 Sistem Pakar yang terkenal

Sistem Pakar	Kegunaan
MYCIN	Diagnosa Penyakit
DENDRAL	Mengidentifikasi struktur molekular campuran yang tak di kenal
SOPHIE	Analisis sirkut elektronik
Prospector	Digunakan dialam geologi untuk membantu mencari dan menemukan deposit
XCON & XSEL	Membantu konfigurasi sistem komputer besar
FOLIO	Membantu memberikan kuputusan bagi seorang manajer dalam hal stok broker dan investasi
DELTA	Pemeliharaan lokomotif listrik disel

2.1.2.5. Struktur Sistem Pakar

Struktur sistem pakar terdiri dari 2 bagian pokok, yaitu: lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan di gunaka sebagai pembangun sistem pakar baik dari segi pembangun komponen maupun basis pengetahuan. Lingkungan konsultasi di gunakan oleh seorang yang bukan ahli konsultasi (Kusumadewi, 2003:113).



Gambar 2.1 Struktur Sistem Pakar

Komponen-komponen yang ada pada sistem pakar adalah gambar 2.1 (Kusumadewi, 2003:113):

1. Subsistem penambahan pengetahuan. Bagian ini di gunakan untuk memasukan pengetahuan, mengkonstruksi atau memperluas pengetahuan dalam basis pengetahuan. Pengetahuan itu berasal dari ahli, buku, basis data, peneltian dan gambar.
2. Basis pengetahuan. Berisi pengetahuan-pengetahuan yang di butuhkan untuk memahami, memformulasikan dan menyelesaikan masalah.
3. Motor inferensi (*inference engine*). Program yang berisi metodologi yang di gunakan untuk melakukan penalaran terhadap informasi-informasi dalam basis pengetahuan dan *blackboard,* serta di gunakan untuk memformulasikan konklusi, ada 3 elemen utama dalam motor inferensi, yaitu:

- *Intepreter:* mengeksekusi item-item agenda yang terpilih dengan menggunakan aturan-aturan dalam basis pengetahuan yang sesuai
- *Scheduler:* akan mengontrol agenda
- *Consistency enforcer:* akan berusaha memelihara kekonsistenan dalam mempresentasikan solusi yang bersifat darurat

4. *Blackboard*. Merupakan area dalam memori yang di gunakan untuk merekam kejadian yang sedang berlangsung termasuk keputusan sementara. Ada 3 tipe keputusan yang dapat di rekam, yaitu
   - Rencana: bagaimana menghadapi masalah
   - Agenda: aksi-aksi yang potensial yang sedang menunggu di eksekusi
   - Solusi: calon aksi yang akan di bangkitkan
5. Antarmuka, di gunakan untuk media komunikasi antara user dan program
6. Subsistem penjelasan. Di gunakan untuk melacak respon dan memberikan penjelasan tentang kelakuan sistem pakar secara interaktif melalui pertanyaan:
   - Mengapa suatu pertanyaan di tanyakan oleh sistem pakar?
   - Bagaimana konklusi di capai?
   - Mengapa ada alternatif yang di batalkan?
   - Rencana apa yang di gunakan untuk mendapatkan solusi?
7. Sistem penyaring pengetahuan. Sistem ini di gunakan untuk mengevaluasi kinerja sistem pakar itu sendiri untuk melihat apakah pengetahuan-pengetahuan yang ada masih cocok untuk di gunakan di masa mendatang.

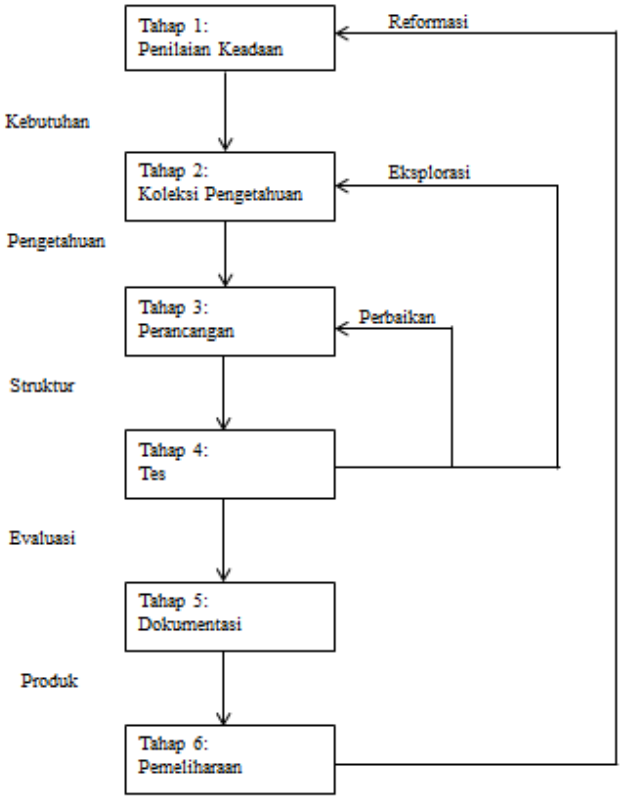
2.1.2.6. Mengembangkan Sistem pakar

Secara garis besar pengembangan sistem pakar pada gambar adalah sebagai berikut (Kusumadewi, 2003:122):

1. Mengidentifikasikan masalah dan kebutuhan. Mengkaji situasi dan memutuskan dengan kepastian tentang masalah yang akan di komputerisasi dan apakah dengan sistem pakar bisa lebih membantu atau tidak.
2. Menentukan masalah yang cocok, ada beberapa syarat yang harus di penuhi agar sistem pakar dapat bekerja dengan baik.
3. Mempertimbangkan alternatif. Dalam hal ini ada 2 alternatif yaitu menggunakan sistem pakar atau komputer tradisional.
4. Menghitung pengembalian investasi, termasuk diantaranya: biaya pembuatan sistem pakar, dan biaya training.
5. Memilih alat pengembangan, bisa di gunaka *software* pembuat sistem pakar atau dirancang dengan bahasa pemograman sendiri.
6. Rekayasa Pengetahuan. Perlu di lakukan dengan penyempurnaan terhadap aturan-aturan yang sesuai.
7. Merancang sistem. Bagian ini termasuk pembuatan *protoype*, serta menterjemahkan pengetahuan menjadi aturan-aturan.
8. Melengkapi pengembangan, termasuk pengembangan *protoype* apabila sistem yang telah ada sudah sesuai dengan keinginan.
9. Menguji dan mecari kesalahan sistem.
10. Memelihara sistem. Dalam hal ini harus di lakukan: memperbaharui pengetahuan, mengganti penggetahuan, mengganti pengetahuan yang sudah

ketinggalan, dan meluweskan sistem agar bisa lebih baik lagi dalam menyelesaikan masalah.

Seperti dari penjelasan pengembangan perangkat lunak di atas, terdapat gambar tahap-tahap pengembangan sistem pakar seperti gambar 2.2



Gambar 2.2 Tahap-Tahap Pengembangan sistem pakar

2.1.2.7. Ciri-Ciri Sistem Pakar

Menurut (Sutojo et al., 2011:162) Ciri-ciri sistem pakar adalah sebagai berikut.

1. Terbatas pada dominan keahlian tertentu.
2. ada dominan keahlian tertentu.
3. Dapat memberikan penalaran untuk data-data yang tidak lengkap atau pasti.
4. Dapat menjelaskan alasan – alasan dengan cara yang dapat di pahami.
5. Bekerja berdasarkan kaidah/ rule tertentu.
6. Mudah di modifikasi.
7. Basis pengetahuan dan mekanisme inferensi terpisah
8. Keluarannya bersifat anjuran.
9. Sistem dapat mengaktifkan kaidah
10. secara searah yang sesuai, di tuntun oleh dialog dengan pengguna.

2.1.2.8. Teknik inferensi *foward chaining* dan *backward chaining*

Menurut (Sutojo et al., 2011:171) Pada sistem pakar berbasis rule, domain pengetahuan di representasikan dalam sebuah kumpulan rule berbentuk *IF-THEN*, sedangakan data direpresentasikan dalam sebuah kumpulan fakta-fakta tentang kejadian saat ini. Mesin inferensi membandingkan masing-masing rule yang tersimpan dalam basis pengetahuan dengan fakta-fakta yang terdapat dalam database. Jika bagian *IF* (kondisi) dari *rule* cocok dengan fakta, maka rule dieksekusi dan bagian *THEN* (aksi) di letakkan oleh dalam database sebagai fakta baru yang di tambahkan.

1. *Forward Chaining*

   Forward chaining adalah teknik pencarian yang di mulai dengan fakta yang di ketahui.kemudian mencocokan fakta-fakta tersebut dengan bagian *if* dari *rule IF-THEN*. Bila ada fakta yang cocok dengan bagian *if*, make *rule* tersebut di eksekusi. Bila sebuah rule di eksekusi, maka sebuah fakta baru (bagian *THEN*) di tambahkan ke dalam database. Setiap kali pencocokan, di mulai dari *rule* teratas. Setiap rule hanya boleh dieksekusi sekali saja, proses pencocokan berhenti bila tidak ada lagi rule yang bisa di eksekusi. Metode pencarian yang di gunakan adalah *Depth-First Search (DFS), Breadth-First Search (BFS) atau Best First Search.*

2. *Backward Chaining*

   Backward chaining adalah metode inferensi yang bekerja mundur ke arah kondisi awal. Proses di awali dari Goal (yang berada di bagian THEN dari rule IF-THEN), kemudian pencarian mulai di jalankan untuk mencocokan apakah fakta-fakta yang ad cocok dengan premis-premis di bagian IF. Jika cocok, rule di eksekusi, kemudian di hipotesis di bagian THEN di tempatkan di basis data sebagai fakta baru. Jika tidak cocok, simpan premis dibagian IF ke dalam stack sebagai subGoal. Proese berakhir jika Goal di temukan atau tidak ada rule yang bisa membuktikan kebenaran dari subGoal atau Goal.

2.1.2.9. Strategi Penyelesaian Konflik

Menurut (Sutojo et al., 2011:186) Apabila ada suatu fakta yang dapat memicu lebih dari satu rule, maka ada kemungkinan akan terjadi konflik dalam memori kerja. Artinya rule mana yang harus di pilih oleh sistem pakar. Sistem pakar tidak bisa memilih semau rule sekaligus. Ia harus memutuskan untuk memilih satu rule. Dalam melakukan pemilihan, sistem pakar menggunakan cara-cara untuk memilih rule-rule yang akan di terapkan apabila terdapat lebih dari satu rule yang cocok dengan fakta yang terdapat dalam memori kerja, diantaranya adalah:

1. *No Duplication*. Tidak boleh memicu sebuah rule dua kali menggunakan fakta yang sama agar tidak ada fakta yang di tambahkan kedalam memori kerja lebih dari sekali
2. *Regency*. Fakta yang paling baru dalam memori kerja harus di gunakan dalam memilih *rule*.
3. *Specificity*. Fakta yang lebih spesifik (khusus) harus di gunakan dalam memicu *rule*.
4. *Operation priority*. Pilih aturan dengan prioritas yang lebih tinggi.

2.1.3. Jaringan Syaraf Tiruan

Menurut (Sutojo, 2011:283) Jarigan Syaraf Tiruan adalah paradigma pengelolahan informasi yang terinspirasi oleh sistem saraf secara biologis, Seperti Proses Informasi pada otak manusia. Cara kerja sistem saraf tiruan seperti cara kerja manusia yaitu belajar melalui contoh.

Kelebihan-kelebihan yang di berikan JST antara lain:

a) *Belajar Adaptive*: Kemampuan untuk mempelajari bagaimana melakukan pekerjaan berdasarkan data yang di berika untuk pelatihan atau pengalaman awal.
b) *Self-Organisation*: sebuah JST dapat membuat organisasi sendiri atau representasi dari informasi yang di terimanya selama waktu belajar
c) *Real Time Protection*: Perhitungan JST dapat di lakukan secara pararel sehingga perangkat keras yang di rancang dan di produksi secara khusus dapat mengambil keuntungan dari kemampuan ini.

Selain mempunyai kelebihan-kelebihan tersebut, JST juga mempunyai kelemahan-kelemahan berikut.

a) Tidak efekif jika digunakan untuk melakikan operasi-operasi numerik dengan presisi tinggi.
b) Tidak efesien jika digunakan untuk melakukan operasi-operasi numerik dengan presisi tinggi.
c) Untuk beroperasi JST butuh pelatihan sehingga bila jumlah datanya besar, waktu yang di gunakan untuk proses pelatihan sangat lama.

2.1.4. Fuzzy Logic

Menurut (Sutojo et al., 2011:211) 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.

Metodologi ini dapat diterapkan pada perangkat keras, perangkat lunak, atau kombinasi keduanya. Dalam logika klasik dinyatakan bahwa segala sesuatu bersifat biner, yang artinya adalah hanya mempunyai dua kemungkinan, “Ya atau Tidak”, “Benar atau Salah”, “Baik atau Buruk”, dan lain-lain.

Ada 3 metode dalam logika *fuzzy*, yaitu:

1. Metode Tsukamoto

Menurut (Kusumadewi, 2013) pada Metode Tsukamoto, Setiap Konsukuen pada aturan yang berbentuk *IF-Then* harus di presentasikan dengan suatu himpunan fuzzy dengan fungsi keanggotaan yang menonton Sebagai hasilnya, output hasil referensi dari tiap-tiap aturan yang di berikan secara tegas berdasarkan α-predikat. Hasil akhirnya di peroleh dengan menggunakan rata-rata terbobot

2. Metode Mamdani

Menurut (Kusumadewi, 2013) Metode Mamdani sering di kenal sebagai *Max-Min.* Metode ini di perkenalkan oleh Ebrahim Mamdani pada tahun 1975. Untuk mendapatkan output, di perlukan 4 tahapan:

a) Pembentukan himpunan *fuzzy*

Pada metode mamdani, baik variabel *input* maupun variabel *output* di bagi menjadi satu atau lebih himpunan *fuzzy*.

b) Aplikasi fungsi implikasi

Pada metode mamdani, fungsi implikasi yang di gunakan adalah Min.

c) Komposisi aturan

Tidak seperti penalaran menonton, apabila sistem terdiri dari beberapa aturan, maka inferensi di peroleh dari kumpulan dan kolerasi antar aturan.

Ada 3 metode yang di gunakan dalam melakukan inferensi sisem fuzzy, yaitu: *max additive* dan probabilistik OR (probor)

d) Penegasan (*defuzzy*)

Input dari proses defuzzifikasi adalah suatu himpunan *fuzzy* yang diperoleh dari komposisi aturan-aturan *fuzzy*, sedangkan *output* yang dihasilkan merupakan suatu bilangan pada domain himpunan *fuzzy* tersebut. Sehingga diberikan suatu *fuzzy* dalam *range* tertentu, maka harus dapat diambil suatu nilai crisp tertentu sebagai *output*.

3. Metode Sugeno

Menurut (Kusumadewi, 2013) penalaran metode SUGENO hampir sama dengan penalaran MAMDANI, hanya saja *output* (Konsukuen) sistem tidak dapat berupa himpunan *fuzzy*; melainkan berupa konstanta atau persamaan linear. Metode ini di perkenalkan oleh Takagi-Sugeno Kang pada tahun 1985, sehinggan metode ini sering juga dinamakan metode TSK.

2.2. SIFAT

Pada hakikatnya pola pikir seorang entreprenuer hendaknya di mulai dengan berpikir tentang keinginan menjadi pribadi yang sukses. Yakni hakikat entrepreneur adalah orang-orang yang memiliki kemampuan dalam melihat dan menilai kesempatan-kesempatan bisnis, mengumpulkan sumber daya - sumber daya yang di butuhkan untuk mengambil tindakan yang tepat, mengambil keuntungan serta memiliki ciri, sifat, watak dan kemauan untuk mewujudkan gagasan inovatif kedalam dunia nyata secara kreatif dalam rangka sukses meningkatkan pendapatan. (Tontowi, 2016:11)

Berikut adalah sifat-sifat entreprenuer yang di katakan sebagai sukses:

1. Percaya diri

   Sifat utama dari percaya diri dimulai dari pribadi yang mantap, tidak mudah terambang-ambing oleh pendapat dan saran orang lain melainkan menggunakan sebagian saran tersebut sebagai masukan.

2. Berorientasi pada tugas dan hasil

   Sifat seorang entreprenuer tidak mengutamakan *prestige* dahulu melainkan fokus kepada prestasi yang ingin di capai

3. Pengambilan resiko.

   Ciri pengambilan resiko berpengaruh penting dalam dunia wirausaha yang penuh resiko dan tantangan. Hal penting yang harus diperhatikan adalah bahwa bagaimana seorang *entreprenuer* mengambil sebuah risiko penuh pertimbangan

4. kepemimpinan.

   Dalam diri seorang *entreprenuer* mutlak memiliki jiwa kepemimpinan. Seorang pemimpin yang baik harus mendengar saran dan kritik dari bawahanya demi kemajuan kinerja perusahaan.

5. Kreativitas

   Seorang entreprenuer tidak hanya mengekor pada orang lain, tetapi memilik pendapat sendiri, ide yang orisinil dan mampu merealisasikan ide tersebut.

6. Berorientasi pada masa depan

   Seorang entreprenuer harus perspektif, mempunyai visi ke depan. Sebab sebuah usaha bukan didirikan untuk sementara tetapi untuk selamanya. Untuk

menyiapkan visi yang jauh ke depan, entreprenuer perlu menyusun perencanaan dan strategi yang matang

7. Jujur dan tekun

   Kejujuran merupakan landasan moral yang terkadang dilupakan oleh seorang wirausahawan. Kejujuran dalam berperilaku bersifat komplek

2.3. Software Pendukung

2.3.1 *UML (Unified Modeling Language)*

"*The UML defines a diagrammatic notation for describing the artefacts of an OOAD. Through the UML we can visualize, specify, construct and document our software application*". (Noni Rahmawati; Herry Mulyono, 2016:106-107)

UML adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membantu analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. Banyak orang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan pemrograman procedural atau structural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time* (waktu nyata).

Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorintasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan model

visual untuk menspesifikasikan, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.(Rosa A.S; M.Shalahuddin, 2011 :113).

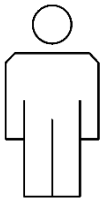
2.3.1.1. Use Case Diagram

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Ada duahal utama pada *use case* yaitu pendefinisian apa yang disebut *actor* dan *use case.*(Rosa A.S; M.Shalahuddin, 2011)

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

Tabel 2.2 Simbol-simbol pada *Use Case*

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 di awal frase nama *use case*
Aktor / *actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi / a*ssociation* ————————	Komunikasi antara aktor dan *use case* yang berpatisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi / *extend* – – – –>	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
Generalisasi / *generalization* ————▷	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Menggunakan / *include*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini

– – - –>	untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

Sumber: (Rosa A.S; M.Shalahuddin, 2011)

2.3.1.2. *Activity Diagram*

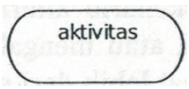
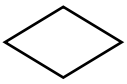
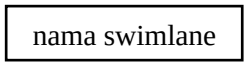
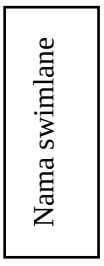
Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor*, jadi aktivitas yang dapat dilakukan oleh sistem. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut: (Rosa A.S; M.Shalahuddin, 2011)

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem/*userinterface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut adalah simbol-simbol yang ada pada diagram aktivitas:

Tabel 2.3 Simbol-simbol pada *activity diagram*

Simbol	Deskripsi

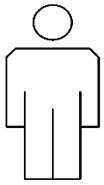
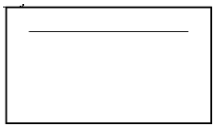
Status awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / *join*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

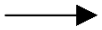
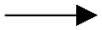
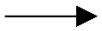
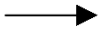
Sumber: (Rosa A.S; M.Shalahuddin, 2011)

2.3.1.3. Sequence Diagram

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan watu hidup objek dan massage yang dikirimkan dan diterima antara objek. Membuat diagram sekuen juga dibutuhkan untuk menlihat skenario yang ada pada *use case*. (Rosa A.S; M.Shalahuddin, 2011)

Tabel 2.4 Simbol-simbol pada *Sequence diagaram*

Simbol	**Deskripsi**
Aktor 	Orang, proses, atau sistem lain yang beinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang. Biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Garis hidup / *lifeline*	Menyatakan kehidupan suatu objek.
Objek 	Menyatakan objek yang beinteraksi pesan.
Waktu aktif	Menyatakan objek dalam keadaaan aktif dan beinteraksi pesan.

Pesan tipe *create* <<create>> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
Pesan tipe *call* 1 : nama_metode() 	Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri.
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainnya, arah panah mengarahkan pada objek yang dikirimi.
Pesan tipe *return*	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan sesuatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
Pesan tipe *destroy* <<destroy>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*.

Sumber: (Rosa A.S; M.Shalahuddin, 2011)

2.3.1.4. *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga pembuat perangkat lunak atau *programmer* dapat membuat kelas-kelas di dalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut: (Rosa A.S; M.Shalahuddin, 2011)

1. Kelas main

   Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.
2. Kelas yang menangani tampilan sistem (*View*)

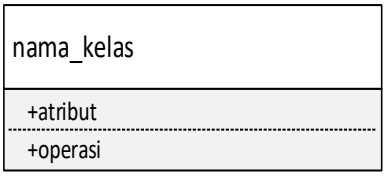
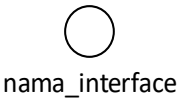
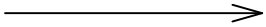
   Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
3. Kelas yang diambil dari pendefinisian *use case* (*controller*)

   Kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
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:

Tabel 2.5 Simbol-simbol pada *class diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka / *interface* 	Sama dengan konsep interface dalam pemrograman berorientasi objek
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: (Rosa A.S; M.Shalahuddin, 2011)

2.3.2. *Xampp*

Xampp merupakan *software* bundel yang terdiri dari kumpulan paket program yang berhubungan dengan *database server*, *web server*, dan lain sebagainya. Didalam *xampp* terdapat *Apache* sebagai *server web*, *MySQL* sebagai *server* basis data, *Filezila* sebagai *FTP server*, dan beberapa fitur tambahan seperti *Mercury* dan *Tomcat*. Aplikasi ini dapat di unduh di alamat "*apachefriends.org*". (Angga Reza Palevi; Krisnawati, 2013:4)

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

2.3.2.1. *PHP*

PHP stands for Hypertext Preprocessor. It is a server-side scripting language that powers some of the most popular websites in the world, including WordPress and Facebook. It is open source, relatively easy to learn, and works perfectly with MySQL, making it a popular choice for web developers. (Er. Saurabh Walia; Er. Satinderjit Kaur Gill, 2014:27)

PHP (*Hypertext Preprocessor*) adalah bahsa *server-sidescripting* yang menyatu dengan HTML untuk membuat halaman web yang dinamis. Karena *PHP* merupakan *server-sidescripting* maka sintaks dan perintah-perintah *PHP* akan

dieksekusi di *server* kemudian hasilnya akan dikirim ke *browser*. (Arip Aryanto; Tri Irianto Tjendrowasono; Berliana Kusuma Riasti, 2012).

PHP (Hypertext Preprocessor) merupakan bahasa pemrograman berbasis *web* yang memiliki kemampuan untuk memproses dan mengolah data secara dinamis. *PHP* dapat dikatakan sebagai sebuah *server-sideembedded script language*, artinya semua sintaks dan pemerintah program yang anda tulis akan sepenuhnya dijalankan oleh *server*, tetapi dapat disertakan pada halaman HTML biasa. Pada umumnya, semua aplikasi yang dibangun menggunakan *PHP* akan memberikan hasil *web browser*, tetapi prosesnya secara seluruhan dijalankan pada *server*. (Wahana Komputer, 2009 :3)

PHP mempunyai beberapa kemampuan yang merupakan salah satu kelebihan *PHP*. Kemampuan tersebut antara lain: (Wahana Komputer, 2009 :10)

1. Cara koneksi dan *query database* yang sederhana.
2. Dapat bekerja pada sistem operasi berbasis *Windows, Linux, Mac OS,* dan kebanyakan varian *UNIX*.
3. Biaya yang dibutukan untuk menggunakan *PHP* tidak mahal, atau bahkan gratis.
4. Mudah digunakan karena memiliki fitur dan fungsi khusus untuk membuat *web* dinamis. Bahasa pemrograman *PHP* dirancang untuk dapat dimasukkan dalam *HTML (embedded script).*
5. *Security* sistem yang cukup tinggi.

6. Waktu eksekusi yang lebih cepat dibandingkan dengan bahasa pemrograman *web* lainnya berorientasi pada *serverside scripting*.

7. Akses ke sistem *database* yang lebih fleksibel dan mudah, seperti *MySQL*.

Selain memiliki berbagai keunggulan, *PHP* juga memiliki beberapa kekurangan. Dari segi bahasa *PHP* bukanlah bahasa yang cocok untuk pengembangan berskala besar. Kekurangan yang utama adalah tidak adanya *namespace*. *Namespace* adalah sebuah cara untuk mengelompokkan fungsi atau nama variable dalam susunan hierarki. (Wahana Komputer, 2009 :12)

2.3.2.2. *MySQL*

MYSQL adalah multi user database yang menggunakan bahasa *SQL* (*Structured Query Language*). *SQL* adalah bahasa standar yang digunakan untuk mengakses *database server*.

Dalam konteks bahasa *SQL*, pada umumnya informsi tersimpan dalam tabel-tabel yang secara logika merupakan struktur dua dimensi yang terdiri atas baris-baris data yang berada dalam satu atau lebih kolom. Baris pada tabel sering disebut sebagai *instance* dari data, sedangkan kolom sering disebut sebagai *attribute* atau *field*. Keseluruhan tabel tersebut dihimpun dalam satu kesatuan yang disebut *database*. (Veronica Lusiana, 2009:28)

MySQL (My Structured Query Language) atau yang biasa dibaca mai-sekuel adalah sebuah program pembuat dan pengelolah *database* atau yang sering disebut denagn *DBMS* (*Database Management System*), sifat dari *DBMS* ini adalah *Open Source*. *MySQL* sebenarnya produk yang berjalan pada *platform Linux*, dengan

adanya perkembangan dan banyaknya pengguna, serta lisensi dari database ini adalah *Open Source*, maka para pengembang kemudian merilisi versi *windows*.

Selain itu *MySQL* juga merupakan program pengakses *database* yang bersifat jaringan, sehingga dapat digunakan untuk aplikasi *Multi User* (Banyak Pengguna). Kelebihan lain dari *MySQL* adalah menggunakan bahasa *query* (permintaan) standar *SQL* (Structures Query Language). *SQL* adalah suatu bahasa permintaan yang terstruktur, *SQL* telah distandartkan untuk semua program pengakses *database* seperti *Oracle, PosgreSQL, SQL server* dan lain-lain.

Sebuah program penghasil *database*, *MySQL* tidak mungkin berjalan sendiri tanpa adanya sebuah aplikasi pengguna (*interface*) yang berguna sebagai program aplikasi pengakses *database* yang dihasilkan. *MySQL* dapat didukung oleh hamper semua program aplikasi baik yang *Open Source* seperti *PHP* maupun yang tidak *OpenSource* yang ada pada platform *windows* seperti *visual basic*, *Delphi* dan lainnya. (Bunafit Nugroho, 2008 :91)

Pada umumnya, perintah yang paling sering digunakan dalam *MySQL* adalah *SELECT* (mengambil), *INSERT* (menambah), *UPDATE* (mengubah), dan *DELETE* (menghapus). Selain itu, *SQL* juga menyediakan perintah untuk membuat *database*, *field*, ataupun *index* untuk menambah atau menghapus data.

Ada beberapa alasan yang menjadikan *database MySQL* sangat diminati oleh para programmer, diantaranya:

1. Bersifat *open source*.

2. Menggunakan bahasa *SQL* (*Structure Query Language*), yang merupakan standar bahasa dalam pengolahan data.

3. *Performance* dan *reliable*, pemrosesan database-nya sangat cepat dan stabil.

4. Sangat mudah dipelajari (*ease of use*).

5. Memiliki dukungan (*group*) pengguna *MySQL*.

6. Lintas *Platform*, dapat digunakan pada berbagai Sistem Operasi berbeda.

7. *Multiuser*, dimana *MySQL* dapat digunakan oleh banyak user dalam waktu yang bersamaan tanpa mengalami konflik.

2.4. Penelitian Terdahulu

Sebagai bahan pertimbangan dalam penelitian ini, maka penulis mencantumkan beberapa penelitian yang di ambil beberapa jurnal ilmiah, yaitu:

1. Penelitian yang di lakukan oleh (Iskandar, Sholeh, & Iswahyudi, 2015), **SISTEM PAKAR UNTUK MENENTUKAN KEPRIBADIAN SESEORANG BERDASARKAN TES PERSONALITAS *FLORENCE* LITTAUER BERBASIS *WEB***, dalam penelitian ini dapat di simpulkan bahwa Sistem Pakar tes kepribadian dapat membantu user mendiagnosa tipe kepribadian diri sendiri, memberikan rekomendasi jenis pekerjaan yang sesuai dengan tipe kepribadian dan memberikan pengetahuan tentang kelebihan dan kelemahan dari setiap tipe kepribadian. Sistem ini dibangun untuk menyimpan pengetahuan keahlian seorang pakar psikologi khususnya psikologi kepribadian. Pembangunan sistem dirancang sedemikian rupa

sehingga dapat mengadopsi perkembangan profil kepribadian yang digunakan berbasis aturan tes personalitas *Florence* Littauer dan metode inferensi forward chaining. Implementasi sistem pakar dalam bentuk *web* sangat membantu memberikan kemudahan bagi user dalam mengaksesnya. Sistem ini juga masih sebatas simulasi dan belum dikonsultasikan terhadap ahli psikologi.

2. (Mahmudi & Fatkhudin, 2013), **PENENTUAN BANTUAN MODAL WIRAUSAHA BARU MENGGUNAKAN METODE FUZZY INFERENSI SYSTEM (FIS) MAMDANI** Berdasarkan kegiatan selama perancangan dan implementasi pada proses pembuatan aplikasi Fuzzy Inference System Untuk Penentuan Bantuan Modal Wirausaha menggunakan Metode Mamdani maka dapat di ambil kesimpulan bahwa Aplikasi Penentuan Bantuan Modal Wirausaha yang dikembangkan dapat membantu para pengambil kebijakan dalam hal ini Dinsosnakertrans dalam memilih para pengusaha mikro yang mengajukan permohonan bantuan dengan kemungkinan hasil yang terbaik karena setiap perhitungan diperoleh dari kriteria – kriteria yang dikehendaki oleh pengambil kebijakan.
3. (Ade Setiadi, 2015), **ANALISA SISTEM PAKAR UNTUK IDENTIFIKASI KEPRIBADIAN SISWA MENGGUNAKAN ALGORITMA FUZZY PADA SISWA SLTA**, Berdasarkan hasil penelitian dan pembahasan yang penulis lakukan mengenai “ Analisa Sistem Pakar Untuk Identifikasi Kepribadian Siswa Menggunakan Algoritma Fuzzy”. Maka penulis pada akhirnya mengambil beberapa

kesimpulan dalam merepresentasikan pengetahuan dengan kaidah produksi pada sistem pakar yang diterapkan untuk mendiagnosa yaitu Dengan adanya sistem pakar ini masalah kekurangan tenaga pakar dapat terselesaikan, dengan sistem pakar ini user dapat berinteraksi dengan sistem seperti halnya berinteraksi dengan pakar.

4. (Andreyana, Piarsa, & Buana, 2015), **SISTEM PAKAR ANALISIS KEPRIBADIAN DIRI DENGAN METODE CERTAINTY FACTOR**, kesimpulan dari penelitian ini adalah Sistem Pakar Analisis Kepribadian Diri berbasis *web* menghasilkan jenis-jenis kepribadian menggunakan teori Big Five Personality. *Metode Certainty Factor* digunakan dalam Sistem Pakar Analisis Kepribadian Diri untuk mendapat nilai persentase kepercayaan pakar pada *output* sistem dengan tingkat akurasi mencapai 74% berdasarkan hasil *survey* yang dilakukan kepada 25 orang koresponden untuk mencoba dan memberi penilaian mengenai ketepatan pertanyaan-pertanyaan dan hasil dari sistem. Sistem Pakar Analisis Kepribadian Diri menghasilkan output jenis-jenis kepribadian yang didapat dari jawaban pengguna dengan memberi pertanyaan-pertanyaan mengenai sifat kepribadian. *Output* tes kepribadian yang dihasilkan sistem sesuai dengan pengetahuan yang dimiliki oleh pakar psikolog.
5. (Abu Haris & Rahman, 2017) **A STUDY ON APPLICATION OF FUZZY METHODS IN ENTREPRENEURSHIP DOMAIN** kesimpulan pada penelitian ini adalah Fuzzy methods can be versatile and flexible tool for data that are complex, vague and imprecise. Fuzzy addresses application

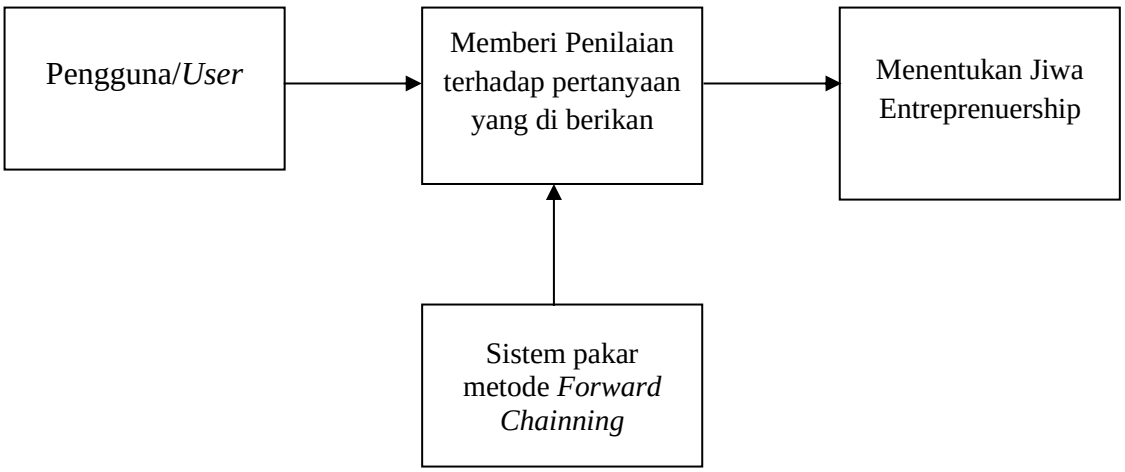
that resembles human in decision making. Fuzzy methods have the ability to generate precise solutions for certain or approximate information and the data generates through fuzzy methods has the advantage of reducing uncertainty.

6. (Al-lamki, Al-sumri, Al-ismaili, & Al-busaidi, 2016) **AN EXPERT SYSTEM TO DETERMINE THE PROBABILITY OF SUCCESS AND READINESS IN ENTREPRENEURSHIP** study presented the development of an expert system to assess the probability of success and readiness in entrepreneurship, based on personal traits. The study also indicated, based on the evaluation of 15 users, that the system adds value and provides good quality in terms of inputs, outputs, and overall system quality.

2.5. Kerangka Pemikiran

Kerangka pemikiran adalah merupakan konseptual mengenai bagaiaman satu teori berhubungan diantara berbagai faktor yang telah di identifikasikan penting terhadap masalah penelitian. Dalam kerangka pemikiran, peneliti harus menguraikan konsep atau variabel penelitiannya secara lebih terperinci. Tidak hanya mendefinisikan variabel tadi, tetapi juga menjelaskan keterkaitan diantara varibel tadi (Noor, 2011:76).

Penjelasan sementara terhadap suatu gejala yang menjadi objek permasalahan peneliti. Kerangka berpikir ini disusun dengan berdasarkan pada tinjauan pustaka dan hasil penelitian yang relevan atau terkait.



Gambar 2.3 Kerangka pemikiran
Sumber: Data Penelitian (2018)