BAB II
KAJIAN PUSTAKA

2.1 Teori Dasar

Deskripsi teori berisi tentang penjelasan variabel-variabel yang diteliti melalui pendefinisian, dan uraian yang lengkap serta mendalam dari berbagai referensi, sehingga ruang lingkup dan prediksi terhadap hubungan antara variabel yang akan diteliti menjadi lebih jelas.

2.1.1 *Artificial Intelligence* (kecerdasan buatan)

Kecerdasan buatan atau *artificial intelligence* adalah merujuk pada mesin yang mampu berpikir, menimbang tindakan yang akan diambil, dan mampu mengambil keputusan seperti yang dilakukan oleh manusia (T.sutojo, Edy Mulyanto, 2011:1)

Menurut (T.sutojo, Edy Mulyanto, 2011:1-2) Alan Turing, ahli matematika berkebangsaan inggris yang dijuluki bapak komputer modern dan pembongkar sandi Nazi dalam era Perang Dunia II 1950, menetapkan defenisi *artificial intelligence*. "Jika komputer tidak bisa dibedakan dengan manusia saat berbincang melalui terminal komputer, maka bisa dikatakan komputer itu cerdas, mempunyai kecerdasan".

Kombinasi antara *AI* dengan bidang ilmu yang lainnya melahirkan subdisiplin ilmu dalam *AI*. Beberapa diantaranya adalah logika *fuzzy (fuzzy logic)*, jaringan syaraf tiruan *(artificial neural network)*, dan sistem pakar *(expert system)* (T.sutojo, Edy Mulyanto, 2011:12).

2.1.1.1 Logika Fuzzy *(fuzzy logic)*

Logika *fuzzy* adalah metedologi sistem kontrol pemecahan masalah, yang sesuai diimplementasikan pada sistem, mulai dari sitem yang sederhana, sistem kecil, *embedded system*, jaringan komputer, *multi-channel* atau *workstation* berbasis akuisisi data, dan sistem kontrol. Dalam logika *fuzzy* memungkinkan nilai keanggotaan berada di antara 0 dan 1, artinya suatu keadaan memungkinkan mempunyai dua nilai "Ya" dan "Tidak" secara bersamaan, namun besar nilainya tergantung pada bobot keanggotaan yang dimilikinya. Logika *fuzzy* dapat digunakan diberbagai bidang seperti pada sistem diagnosis penyakit (dalam bidang kedokteran), pemodelan sistem pemasaran, sistem operasi (dalam bidang ekonomi), kendali kualitas air, prediksi adanya gempa bumi, klasifikasi dan pencocokan pola (T.sutojo, Edy Mulyanto, 2011:211-212).

Kelebihan logika *fuzzy* adalah kemampuannya dalam proses penalaran secara bahasa sehingga dalam perancangannya tidak memerlukan persamaan matematik yang rumit.

Beberapa metode yang dapat digunakan dalam sistem inferensi *fuzzy* adalah (T.sutojo, Edy Mulyanto, 2011:233-237) :

1. Metode Tsukamoto

   Dalam inferensinya, metode Tsukamoto menggunakan tahapan berikut :

   a. *Fuzzy*fikasi

   b. Pembentukan basis pengetahuan *fuzzy* (*Rule* dalam bentuk IF…THEN).

   c. Mesin *inferensi*.

   d. Menggunakan fungsi implikasi Min.

2. Metode Mamdani

   Metode mamdani paling sering digunakan dalam aplikasi-aplikasi karena strukturnya yang sederhana, yaitu menggunakan operasi MIN-MAX atau MAX-PRODUCT. Untuk mendapatkan output, diperluhan 4 tahapan berikut:

   a. *Fuzyy*fikasi

   b. Pembentukan baris pengetahuan *fuzzy* (*rule* dalam bentuk IF…THEN

   c. Aplikasi fungsi implikasi menggunakan fungsi MIN dan komposisis antar *rule* menggunakan fungsi MAX (menghasilkan himpunan *fuzzy* baru).

   d. Deffuzifikasi menggunakan metode *centroid*.

3. Metode Sugeno

   Bila *output* dari penalaran dengan metode mamdani berupa himpunan *Fuzzy*, tidak demikian dengan metode Sugeno. Dalam metode Sugeno, output sistem berupa konstanta, atau persamaan linear. Dalam inferensinya, metode Sugeno menggunakan tahapan berikut :

a. *Fuzzy*fikasi

b. Pembentukan basis pengetahuan *fuzzy* (rule dalam bentuk IF…THEN)

c. Mesin inferensi, menggunakan fungsi implikasi MIN

d. *Defuzzyfikasi*, menggunakan metode rata-rata.

2.1.1.2 Jaringan Syaraf Tiruan *(Artificial Neural Network)*

Jaringan syaraf tiruan adalah paradigma pengolahan informasi yang terinspirasi oleh sitem saraf secara biologis, seperti proses informasi pada otak manusia. Cara kerja JST seperti cara kerja manusia, yaitu belajar melalui contoh. Sebuah JST dikonfigurasikan untuk aplikasi tertentu, seperti pengenalan pola atau klasifikasi data, melalui proses pembelajan. Belajar dalam sistem biologis melibatkan penyesuaian terhadap koneksi *synaptic* yang ada antara *neuron* (T.sutojo, Edy Mulyanto, 2011:283-285).

Kelebihan-kelebiha yang diberikan oleh JST antara lain :

1. Belajar *adaptive* : kemampuan untuk mempelajari bagaimana melakukan pekerjaan berdasarkan data yang diberikan untuk pelatihan atau pengalaman awal.
2. *Self-organisation* : sebuah JST dapat membuat organisasi sendiri atau representasi dari informasi yang diterimanya selama waktu belajar.
3. *Real time operation* : perhitungan JST dapat dilakukan secara paralel sehingga perangkat keras yang dirancang dan diproduksi secara khusus dapat mengambil keuntungan dari kemampuan ini.

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

1. Tidak efektif jika digunakan untuk melakukan operasi-operasi numerik dengan presisi yang tinggi.
2. Tidak efisien jika digunakan untuk melakukan operasi algoritma aritmatik, operasi logika, dan simbolis.
3. Untuk beroperasi JST butuh pelatihan sehingga bila jumlah datanya besar, waktu yang digunakan untuk proses pelatihan sangat lama.

Baik tidaknya suatu model JST ditentukan oleh hububngan antar neuron atau yang biasa disebut sebagai arsitektur jaringan. *Neuron-neuron* tersebut terkumpul dalam lapisan-lapisan yang disebut *neorun layer*. Lapisan-lapisan penyusun JST terbagi menjadi tiga, yaitu (T.sutojo, Edy Mulyanto, 2011:292) :

1. Lapisan Input *(Input Layer)*

   Unit-unit dalam lapisan *input* disebut unit-unit input yang bertugas menerima pola inputan dari luar yang menggambarkan suatu permasalahan.
2. Lapisan Tersembunyi *(Hidden Layer)*

   Unit-unit dalam lapisan tersembunyi disebut unit-unit tersembunyi yang mana nilai *output*nya tidak dapat diamati secara langsung.
3. Lapisan Output *(Output Layer)*

   Unit-unit dalam lapisan output disebut unit-unit *output* yang merupakan solusi JST terhadap suatu permasalahan.

Beberapa arsitektur jaringan yang sering digunakan dalam jaringan saraf tiruan antara lain (T.sutojo, Edy Mulyanto, 2011:292-295) :

1. Jaringan Lapisan Tunggal *(single layer net)*

   Jaringan dengan lapisan tunggal terdiri dari 1 lapisan *input* dan 1 lapisan *output*. Jaringan ini menerima input kemudian mengolahnya menjadi output tanpa melewati lapisan tersembunyi.

2. Jaringan Lapisan Banyak *(multilayer net)*

   Jaringan lapisan banyak mempunyai 3 jenis lapisan, yaitu lapisan *input*, lapisan tersembunyi, dan lapisan *output*. Jaringan ini dapat menyelesaikan permasalahan yang lebih kompleks dibandingkan dengan jaringan lapisan tunggal.

3. Jaringan dengan Lapisan Kompetitif *(competitive layer net)*

   Jaringan ini digunakan untuk mengetahui neuron pemenang dari sejumlah *neuron* yang ada. Akibatnya, pada jaringa ini sekumpulan *neuron* bersaing untuk mendapatkan hak menjadi aktif. Nilai bobot setiap *neuron* untuk dirinya sendiri adalah 1, sedangkan untuk *neuron* lainnya bernilai random negatif.

2.1.1.3 Sistem Pakar *(Expert System)*

Sistem pakar pertama kali dikembangkan pada pertengahan tahun 1960. Sistem pakar yang mucul pertama kali adalah *General-pupose 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, SHOPIE untuk analisis sirkuit elektronik, prospector digunakan dibidang geologi untuk membantu mencari dan menemukan deposit, FOLIO digunakan untuk membantu keputusan bagi seorang manager dalam stok dan investasi, DELTA dipakai untuk pemeliharaan lokomotif listrik diesel (T.sutojo, Edy Mulyanto, 2011:159-160).

Menurut (T.sutojo, Edy Mulyanto, 2011:13) sistem pakar adalah suatu sistem yang dirancang untuk dapat menirukan keahlian seorang pakar dalam menjawab pertanyaan dan memecahkan suatu masalah. Siste pakar akan memberikan pemecahan suatu masalah yang didapat dari dialog dengan pengguna. Dengan bantuan sistem pakar seorang yang bukan pakar/ahli dapat menjawab pertanyaan, menyelesaikan masalah serta mengambil keputusan yang bisanya dilakukan oleh seorang pakar.

Sistem pakar menjadi sangat popular karena sangat banyak kemampuan dan manfaat yang diberikan, diantaranya (T.sutojo, Edy Mulyanto, 2011:160-161) :

1. Meningkatkan produktivitas, karena sistem pakar dapat bekerja lebih cepat dari pada 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 seorang.
5. Dapat beroperasi di lingkungan yang bebahaya.
6. Memudahka akses pengetahuan seorang pakar.

7. Andal, 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.

Sistem pakar juga mempunyai beberapa kelemahan selain banyaknya kemampuan dan manfaat yang diberikan, antara lain (Andriani, 2016:12) :

1. Biaya yang diperlukan untuk membuat, memelihara, dan mengembangkan sistem pakar sangat mahal.
2. Sulit kembangkan karena ketersediaan pakar dibidangnya dan kepakaran sulit diekstrak dari manusia karena terkadang sulit bagi seorang pakar untuk menjelaskan langkah mereka dalam menangani masalah.
3. Sistem pakar tidak 100% benar karena seseorang yang terlibat dalam pembuatan sistem pakar tidak selalu benar. Oleh karena itu setelah pembuatan sistem pakar harus dilakukan pengujian terlebih dahulu secara teliti sebelum digunakan.
4. Pendekatan oleh setiap pakar untuk suatu situasi atau *problem* bisa berbeda-beda, meskipun sama-sama benar.
5. *Transfer* pengetahuan dan bersifat subjektif dan bias.
6. Kurangnya rasa percaya pengguna dapat menghalangi pemakaian sistem pakar.

Suatu sistem dikatakan sebagai sistem pakar jika memiliki ciri-ciri sebagai berikut (T.sutojo, Edy Mulyanto, 2011:162) :

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.

Representasi pengetahuan merupakan metode yang digunakan untuk mengodekan pengetahuan dalamsebuah sistem pakar yang berbasis pengetahuan. Hal ini dimaksudkan untuk menangkap sifat-sifat penting dari suatu masalah, sehingga suatu informasi itu dapat diakses oleh prosedur pemecahan masalah. Bahasa *representasi* harus dirancang agar fakta-fakta dan pengetahuan lain yang terkandung didalamnya dapat digunakan untuk penalaran.

Pada sistem pakar berbasis *rule*, domain pengetahuan direpresentasikan dalam sebuah kumpulan *rule* berbentuk IF-THEN, sedangkan 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) diletakkan dalam database sebagai fakta baru yang ditambahkan (T.sutojo, Edy Mulyanto, 2011:171-178).

Sistem pakar mempunyai komponen utama pada strukturnya, antara lain (Andriani, 2016:14-15) :

1. Basis Pengetahuan *(Knowledge Base)*

   Basis pengetahuan yang merupakan representasi pengetahuan yang dimiliki oleh seorang pakar yang tersusun atas fakta dan kaidah. Fakta merupakan informasi tentang objek, peristiwa, dan situasi. Sedangkan kaidah merupakan suatu cara untuk memunculkan fakta baru berdasarkan fakta yang sudah ada dan sudah diketahui.

2. Mesin Inferensi *(Inference Engine)*

   Otak dari sebuah sistem pakar adalah mesin inferensi yang berfungsi untuk memandu proses penalaran terhadap suatu kondisi berdasarkan pada basis pengetahuan yang tersedia. Di dalam mesin inferensi terjadi proses untuk memanipulasi dan mengarahkan kaidah, model, dan fakta yang disimpan dalam basis pengetahuan dalam rangka mencapai solusi atau kesimpulan.

Terdapat dua konsep penalaran yang dapat dilakukan dalam melakukan inferensi, yaitu :

1. Penalaran maju *(forward chaining)*

Penalaran maju merupakan cara penalaran dengan memulai dari fakta terlebih dahulu untuk menguji kebenaran hipotesis atau mencocokkan fakta atau pernyataan dimulai dari bagian sebelah kiri dulu (*IF*). *Forward chaining*

merupakan grup dari *multiple* inferensi yang melakukan pencarian dari suatu masalah kepada solusinya. Menurut (T.sutojo, Edy Mulyanto, 2011:171) bila ada fakta yang cocok dengan bagian *IF*, maka *rule* tersebut dieksekusi. Bila sebuah *rule* dieksekusi, maka sebuah fakta baru (bagian *THEN*) ditambahkan kedalam *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.

Pada metode *forward chaining*, pencarian dapat dilakukan dengan dua cara, yaitu (Andriani, 2016:15) :

a. Menginputkan semua data kedalam sistem pakar dalam sesi konsultasi. Cara seperti ini tepat dan berguna pada sistem pakar dimana proses di dalamnya terotomatisasi dan langsung menerima data dari *database* atau dari suatu set sensor.
b. Memberikan elemen spesifik dari data yang diperoleh selama sesi konsultasi dalam sistem pakar. Cara ini mengurangi jumlah data uang diminta, sehingga data yng diminta hanya data yang benar-benar dibutuhkan oleh sitem pakar tersebut yang nantinya akan digunakan untuk mengambil keputusan.

2. Penalaran mundur *(backward chaining)*

Penalaran mundur merupakan cara penalaran dengan memulai dari hipotesis (ekspektasi apa yang diinginkan terjadi) terlebih dahulu, dan untuk menguji kebenaran hipotesis tersebut harus dicari fakta-fakta yang ada dalam basis pengetahuan. *Backward chaining* adalah metode inferensi yang bekerja mundur

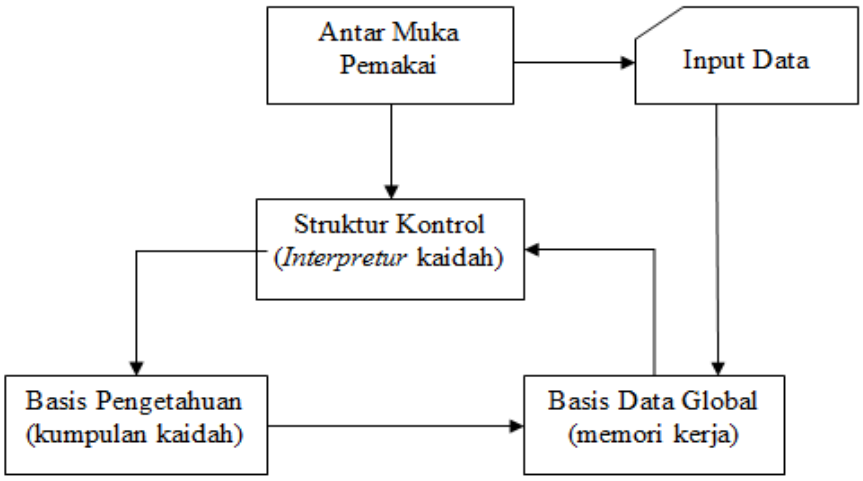
ke arah kondisi awal. Proses diawali dari *Goal* (yang berada dibagian *THEN* dari rule *IF-THEN*), kemudian pecarian mulai dijalankan untuk mencocokkan apakah fakta-fakta yang ada cocok dengan premis-premis dibagian *IF*. Jika cocok, *rule* dieksekusi, kemudian hipotesis di bagian *THEN* ditempatkan di basis data sebagai fakta baru. Jika tidak cocok, simpan premis di bagian IF ke dalam *stack* sebagai s*ubGoal*. *Backward chaining* cocok digunakan untuk suatu aplikasi yang menghasilkan *tree* yang sempit dan cukup dalam.

Ada dua bagian penting dari sistem pakar, yaitu lingkungan pengembangan *(development environment)* dan lingkungan konsultasi *(concultation environment)*. Lingkungan pengembangan digunakan oleh pembuat sistem pakar untuk membangun komponen-komponennya dan memperkenalkan pengetahuan ke dalam basis pengetahuan. Sedangkan lingkungan konsultasi digunakan oleh pengguna untuk berkonsultasi sehingga pengguna mendapatkan pengetahuan dan nasihat dari sistem pakar layaknya berkonsultasi dengan seorang pakar (T.sutojo, Edy Mulyanto, 2011:160).

Struktur sistem pakar berbasis kaidah produksi terdiri dari empat komponen, yaitu (Sri Hartati, 2008:10) :

1. Antar muka pemakai, antar muka penghubung antara pemakai dengan sistem pakar.
2. Basis pengetahuan, berisi sekumpulan kaidah yang berasal dari pengetahuan dalam doamain tertentu. Kaidah ini secara umum disajikan dalam bentuk kaidah produksi (IF…THEN…)

3. Struktur kontrol, merupakan interpreter kaidah atau mesin inferensi yang menggunakan pengetahuan-pengetahuan yang tersimpan dalam basis pengetahuan untuk memecahkan/menyelesaikan permsalahan yang ada.
4. *Working memory* atau basis data global, mencatat status problem saat ini dan histori solusi.



Gambar 2.1 struktur sistem pakar berbasis kaidah produksi
(Sumber : Firebaugh, 1998 daam Hartati. Sri, Iswanti 2008:10)

Menurut (Sri Hartati, 2008: 25) menjelaskan bahwa kaidah produksi menyediakan cara formal untuk merepresentasikan rekomendasi, arahan, atau strategi. Kaidah produksi dituliskan dalam bentuk jika-maka *(if-then)*. Kaidah *if-then* menghubungkan antesenden *(antecedent)* dengan konsekuensi yang diakibatkannya. Berikut ini adalah berbagai struktur kaidah *if-then* yang menghubungkan obyek atau atribut Adedeji,1992 dalam (Sri Hartati, 2008:25) :

IF premis THEN konklusi

IF masukan THEN keluaran

IF kondisi THEN tindakan

IF antesenden THEN konsekuen

IF data THEN hasil

IF tindakan THEN tujuan

IF aksi THEN reaksi

IF sebab THEN akibat

IF gejala THEN diagnosa

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. Sebab mengacu pada keadaan tertentu yang menimbulkan akibat teretntu. Gejala mengacu pada keadaan yang menyebabkan adanya kerusakan atau keadaan tertentu yang mendorong adanya pemeriksaan (Sri Hartati, 2008:25-26).

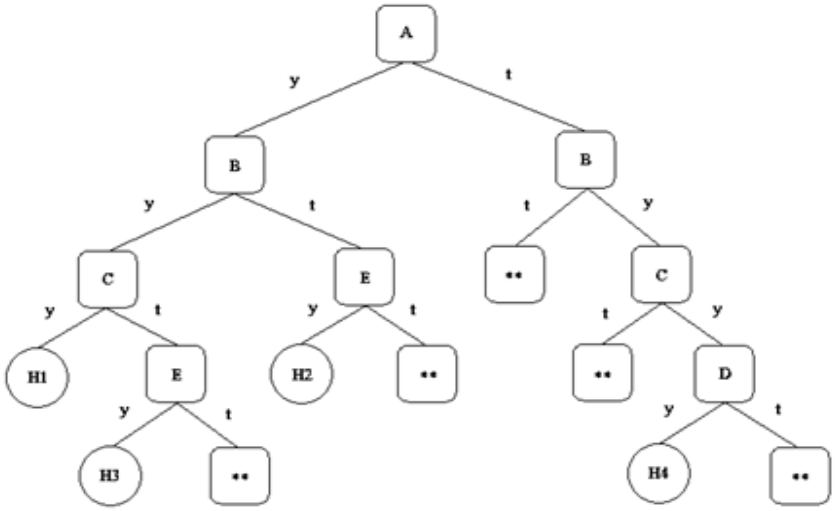
Langkah-langkah sebelum sampai pada bentuk kaidah produksi adalah menyajikan pengetahuan yang berhasil didapatkan dalam bentuk tabel keputusan kenudian dari tabel keputusan dibuat pohon keputusan. Tabel keputusan merupakan matrik kondisi yang dipertimbangkan dalam pendeskripsian kaidah (Sri Hartati, 2008:26).

Tabel 2.1 Tabel Keputusan

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

(**Sumber :** Sri Hartati 2008:32)

Mengacu tabel 2.1, maka dapat dibuat pohon keputusan sebagai berikut :



Gambar 2.2 Pohon keputusan
(**Sumber :** Hartati. Sri, Iswanti 2008:33)

Keterangan :

A = *evidence* A, H1 = hipotesa 1, y = ya

B = *evidence* B, H2 = hipotesa 2, t = tidak

C = *evidence* C, H3 = hipotesa 3, ** = tidak menghasilkan

D = *evidence* D, H4 = hipotesa 4, hipotesa tertentu

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,C dan E.

Demekian juga untuk 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.

Model representasi pengetahuan kaidah produksi banyak digunakan pada aplikasi sistem pakar karena representasi pengetahuan ini mudah dipahami dan bersifat deklaratif, sesuai dengan jalan pikiran manusia dalam menyelesaikan suatu masalah dan mudah diinterprestasikan (Sri Hartati, 2008:39).

Menurut (Andriani, 2016:13) pengembangan aplikasi sistem pakar yang dapat digunakan untuk menggantikan seorang pakar, dilandasi oleh beberapa alasan antara lain :

1. Ilmu pengetahuan dari sistem pakar dapat digunakan setiap waktu dan di berbagai lokasi.
2. Biaya untuk mendatangkan seorang pakar tergolong tinggi.
3. Seorang pakar suatu saat akan pergi atau pensiun.
4. Program dari sistem pakar dapat mengerjakan tugas-tugas rutin yang membutuhkan seorang pakar secara otomatis.
5. Terkadang kepaaran dibutuhkan pada lingkungan yang tidak bersahabat.

2.1.2 Web

Menurut (Murya, 2012:3) *web* merupakan salah satu layanan yang didapat oleh pengguna komputer yang terhubung dengan internet. *Web* adalah suatu layanan sajian informasi yang menggunakan konsep *hyperlink* (tautan), yang memudahkan *surfer* (sebutan para pemakai komputer yang melakukan *browsing*

atau penelusuran informasi melalui internet). Sekarang *web* menjadi standar *interface* pada layanan-layanan yang ada di internet seperti komunikasi melalui *e-mail*, *chatting*, transaksi bisnis, pencarian informasi, dan lain sebagainya. Keistimewaan inilah yang telah menjadikan *Web* sebagai *service* yang paling cepat pertumbuhannya.

2.1.3 *Database* (basis data)

Menurut (Rosa A.S, 2013:43-44) *database* adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. *Database* merupakan media untuk menyimpan data agar dapat diakes dengan mudah dan cepat. Kebutuhan *database* dalam sistem informasi yaitu meliputi memasukkan, menyimpan, dan mengambil data, serta membuat laporan berdasarkan data yang telah disimpan. Salah satu bentuk yang dibutuhkan dalam basis data dalam bentuk sistem yaitu *database management system* (DBMS) adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data.

Suatu sistem aplikasi disebut DBMS jika memenuhi syarat-syarat sebagai berikut (Rosa A.S, 2013:45) :

a. Menyediakan fasilitas untuk meneglola akses data
b. Mampu menangani integritas data
c. Mampu menangani akses data
d. Mampu menangani *backup* data.

2.2 Variabel Penelitian

Variabel penelitian merupakan segala sesuatu yang berbentuk apa saja yang ditetapkan oleh peneliti untuk dipelajari sehingga diperoleh informasi tentang hal tersebut, kemudian ditarik kesimpulannya (Sugiyono, 2014:38). Pada penelitian ini yang menjadi variabel adalah penyakit tanaman jagung, penyakit tanaman jagung ini juga memiliki beberapa jenis, yaitu penyakit busuk batang/busuk akar, penyakit layu fusarium, penyakit hawar daun, penyakit bulai.

Gambar 2.3 Tanaman jagung
(Sumber : Data Olahan, 2018)

2.2.1 Penyakit Busuk Batang/Busuk Akar

Gambar 2.4 Penyakit busuk batang dan busuk akar
(Sumber : Data Olahan, 2018)

Cendawan Fusarium sp merupakan salah satu patogen penyebab penyakit busuk batang/busuk akar pada tanaman jagung. Cendawan ini memproduksi konidia pada permukaan tanaman induknya. Konidia dapat disebarkan oleh angin, air hujan, ataupun serangga. Konidia yang terbawa angin dapat menginfeksi ke tongkol jagung, akibatnya jika biji yang terinfeksi ditanam maka dapat menyebabkan penyakit busuk batang.

Gejala yang terdapat pada penyakit busuk tongkol/busuk akar ini adalah sebagai berikut :

1. Daun layu pada siang hari.
2. Daun menggulung.
3. Biji jagung yang terinfeksi berwarna merah muda hingga coklat.

Solusi untuk penyakit busuk tongkol/busuk akar ini adalah sebagai berikut :

1. Membuat parit yang lancar disekitar tanaman jagung.

2. Membuat bedengan dengan ukuran minimal 20 cm dan tidak boleh terlalu rendah sehingga air tidak lansung mengenai akar dan agar lansung menyerap ke tanah.
3. Pastikan bulma (tanaman pengganggu) selalu bersih.

2.2.2 Penyakit Layu *Fusarium*

Gambar 2.5 Penyakit Layu *Fusarium*
(Sumber : Data Olahan, 2018)

Penyakit layu *fusarium* adalah salah satu patogen penyebab penyakit pada tanaman jagung. Patogen ini menyebabkab pembusukan pada batang dan tongkol jagung.

Gejala-gejala yang tedapat pada penyakit ini adalah :

1. Batang menguning.
2. Batang mengecil.
3. Batang patah.
4. Busuk pada bagian tongkol.
5. Adanya jamur pada tanaman jagung.
6. Jarak tanam terlalu dekat.

Adapun solusi untuk penyakit layu *fusarium* ini adalah :

1. Pastikan jarak tanamnya minimal 40 x 40 cm.
2. Menggunakan pestisida.

2.2.3 Penyakit Hawar Daun

Gambar 2.6 Peyakit Hawar Daun
(Sumber : Data Olahan, 2018)

Penyakit hawar daun yang disebabkan *Helminthosporium sp,* adalah salah satu penyebab penyakit yang terdapat pada tanaman jagung. Satu gejala bercak yang semakin melebar dapat bersatu dengan bercak yang lain sehingga menyebabkan jaringan daun mati.

Gejala-gejala yang muncul pada penyakit hawar daun ini adalah :

1. Daun layu tidak sehat.
2. Bercak kecil pada daun berwarna coklat kehijauan berbentuk bulat memanjang.
3. Bercak berkembang besar berbentuk oval dengan lebar 5-15 cm.

Solusi untuk penyakit hawar daun ini adalah dengan menyiram tanaman jagung pada pagi hari

2.2.4 Penyakit Bulai

Gambar 2.7 Penyakit Bulai
(Sumber : Data Olahan, 2018)

Penyakit bulai pada tanaman jagung disebabkan oleh *oomycete* patogen *Peronosclerospora* spp merupakan salah satu faktor penghambat produktivitas tanaman jagung. Penyakit bulai ini berasal dari genetika atau dari benih, namun jarang terjadi pada tanaman jagung, dalam 1 Ha tanaman jagung yang terkena penyakit bulai ini hanya sekitar 1%. Penyakit bulai juga dapat menyebabkan hasil buah tidak maksimal dan kehilangan hasil produksi tanaman jagung.

Gejala-gejala penyakit bulai, yaitu :

1. Daun berwarna kuning kehijauan.
2. Pada daun permukaan atas dan bawah terdapat warna putih seperti tepung.

3. Adanya warna khlorotik memanjang sejajar tulang daun dengan batas yang jelas antara daun sehat.

Solusi untuk penyakit bulai sebagai berikut :

1. Penanaman jagung secara serentak.
2. Pemusnahan seluruh bagian tanaman sampai ke akarnya pada tanaman yang terserang penyakit bulai.

2.3 Software Pendukung

Unified Modeling language (UML) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan pemodelan. UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek (Rosa A.S, 2013:137-138).

Terdapat 13 macam diagram dalam UML yang dibagi menjadi tiga kategori, yaitu (Rosa A.S, 2013:141-142).

1. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. Diagram *UML* yang termasuk dalam kategori ini adalah *class diagram, object diagram, component diagram, composite structure diagram, package diagram, deployment diagram*.
2. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi

pada sebuah sistem. *UML* yang termasuk dalam kategori ini adalah *use case diagram, activity diagram, state machine diagram.*

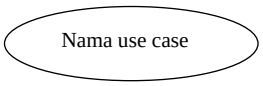
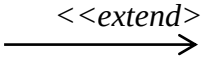
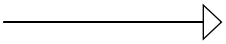
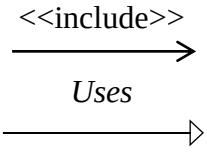
3. *Interaction diagrams* yaitu kumpulan digram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem. UML yang termasuk dalam kategori ini adalah *sequence diagram, communication diagram, timing diagram, interaction overview diagram*.

Menurut (Rosa A.S, 2013:155-156) *use case* dan *sequence* diagram merupakan bagian dari desain sistem. Dalam penelitian ini, diagram yang akan digunakan untuk desain sistem, yaitu :

1. *Use case diagram*

Use case diagram 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. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Berikut adalah simbol-simbol yang ada pada diagram *Use case* :

Tabel 2.2 Simbol *Use Case* Diagram

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. Biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.
Aktor / *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 orang, tapi aktor belum tentu merupakan orang.
Asosiasi / assosiaation 	Komunikasi antara actor dan *use case* yang berpartisipasi pada *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.
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* / *uses* 	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

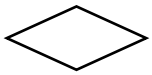
(**Sumber :** A.S dan Shalahuddin, 2013)

2. *Actifity diagram*

Diagram aktivitas menggambarkan aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis yang ada pada perangkat lunak. Pada diagram aktivitas

ini yang perlu diperhatiakan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Berikut adalah simbol-simbol yang ada pada diagram aktivitas (Rosa A.S, 2013:161-163) :

Tabel 2.3 Simbol *Actifity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki awal 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 digram aktivitas memiliki sebuah status akhir
Swimlame  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

(**Sumber :** A.S dan Shalahuddin, 2013)

3. *Sequence diagram*

Diagram sekuen menggambarkan kelkuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek (Rosa A.S, 2013:165-167).

Berikut adalah simbol-simbol yang ada pada diagram sekuen :

Tabel 2.4 Simbol *Sequence* Diagram

Simbol	Deskripsi
Aktor  atau Nama aktor Tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan 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 berada di awal *frase* nama aktor
Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
Objek nama objek : nama kelas	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.

(**Sumber :** A.S dan Shalahuddin, 2013)

Tabel 2.4 Lanjutan

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

(**Sumber :** A.S dan Shalahuddin, 2013)

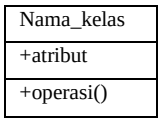
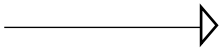
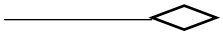
e

4. *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem. Diagram kelas dibuat agar *programmer* membuat membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron (Rosa A.S, 2013:141-146).

Berikut ini adalah simbol-simbol yang ada pada diagram kelas :

Tabel 2.5 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka / *interface*  Nama_interface	Sama dengan konsep *interface* dalam pemograman berorientasi objek
Asosiasi / *association* 	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah / *directed association* 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum-khusus)
Kebergantungan / *dependency* 	Kebergantungan antarkelas
Agregasi / *aggregation* 	Relasi antarkelas dengan makna semua-bagian *(whole-part)*

(**Sumber** : A.S dan Shalahuddin, 2013)

2.3.1 *Xampp*

Gambar 2.8 Logo *Xampp*
(**Sumber :** Reza Palevi, 2013)

Xampp merupakan sebuah perangkat lunak *web server apache* 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 sever basis data, *Filezila* sebagai *FTP server*, dan beberapa fitur tamahan seperti *Mercury* dan *Tomcat*. *Xampp* merupakan software yang mudah digunakan, gratis, dan mendukung instalasi di *Linux* dan *Windows* (Reza Palevi, 2013).

2.3.2 PHP *(Hypertext Processor)*

Gambar 2.9 Logo PHP
(**Sumber :** Amin Miftakul, 2010)

PHP (*Hypertext Processor*) merupakan sebuah bahasa pemrograman server side scripting yang lahir sejalan dengan perkembangan internet. PHP adalah sebuah script yang telah terintegrasi dengan HTML dan mampu menyajikan

informasi yang dinamis. PHP dapat bekerjasama dengan layanan-layanan yang ada di internet menggunakan protokol seperti IMAP, SMPT, HTTP, POP3 dan protokol lain. Selain itu PHP juga menyediakan beragam pilihan *database* yang dapat dipergunakan untuk membuat aplikasi yang berjalan di *web* yang sumber datanya berasal dari *database* (Amin, 2010:1-2).

2.3.3 HTML *(Hyper Text Markup Language)*

Gambar 2.10 Logo *HTML*
(**Sumber :** Agus Saputra, 2012)

Hyper Text Markup Language (HTML) adalah bahasa paling dasar dan penting yang digunakan untuk menampilkan dan mengelola tampilan pada halaman *website*, digunakan untuk menampilkan berbagai informasi didalam sebuah penjelajah *web* internet dan *formatting hypertext* sederhana yang ditulis ke dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegrasi. Dengan kata lain, berkas yang dibuat dalam format ASCII normal sehingga menjadi *homepage* dengan perintah-perintah HTML (Saputra, 2012:1).

Pada tahun 1986, ISO mengeluarkan standarisasi bahasa markup berdasarkan GML dengan nama *Standart Generalized Markup Language* (SGML). HTML dibuat oleh kalaborasi *Caillau Tim* dengan *Banners lee Robert* ketika mereka bekerja di CERN pada tahun 1989 yang kini digunakan untuk

membuat halaman *website*. CERN adalah sebuah lembaga penelitian fisika energi tinggi di jenewa, sejarah HTML (Saputra, 2012:4)

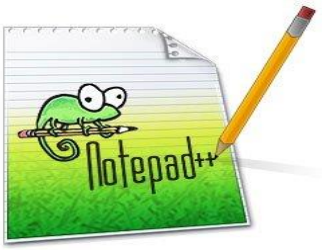
2.3.4 MySQL

Gambar 2.11 Logo *MySQL*
(**Sumber :** Agus Saputra, 2012)

Menurut (Saputra, 2012:77) *MySQL* merupakan salah satu *database* kelas dunia yang sangat cocok bila dipadukan dengan bahasa pemrograman PHP. *MySQL* bekerja menggunakan bahasa *SQL* (*Structure Query Language*) yang merupakan bahasa standar yang digunakan untuk manipulasi *database*. Perintah yang paling sering digunakan dalam *MySQL* adalah *SELECT* (mengambil), *INSERT* (menambah), *UPDATE* (mengubah), dan *DELETE* (menghapus). *SQL* juga menyediakan perintah untuk membuat *database*, *field*, ataupun i*ndex* untuk menambah atau menghapus data.

MySQL dilepaskan dengan suatu lisensi *open source*, dan tersedia secara cuma-cuma. *MySQL* bekerja pada berbagai sistem operasi, dan banyak bahasa. *MySQL* bekerja dengan cepat dan baik dengan data yang besar. *PHP* menyediakan banyak fungsi untuk mendukung *database MySQL*

2.3.5 Notepad ++



Gambar 2.12 Logo Notepad++
(**Sumber :** Reza Palevi, 2013**)**

Notepad++ merupakan sebuah aplikasi yang berguna untuk mengedit text dan skrip kode pemrograman. Notepad++ sebuah aplikasi yang bersifat gratis. Notepad++ menitikberatkan kegunaan aplikasi untuk editing text dalam waktu yang cepat dan praktis. Notepad++ mendukung banyak format bahasa pemrograman seperti PHP, HTML, JavaScript, dan CSS (Reza Palevi, 2013).

2.4 Penelitian Terdahulu

Untuk mendukung teori yang berkaitan dengan penelitian, peneliti mencantumkan beberapa peniliyian terdahulu di bidang sistem pakar.

1. Nama Penulis: (Syarifudin et al., 2018)

   Judul Jurnal : **Sistem Pakar Diagnosis Penyakit Pada Tanaman Jagung Menggunakan Meode Naïve Bayes Bebasis Web**

   ISSN : 2548-964X

Sistem pakar ini bertujuan untuk mendiagnosis penyakit yang terdapat pada tanaman jagung. Sistem pakar ini dirancang menggunakan metode naive bayes dengan menggunakan bahasa pemrograman berbasis android. Pada sistem pakar

ini dilakukan tiga pengujian yaitu *blackbox testing*, *usability testing* dan pengujian akurasi. Tingkat akurasi dengan menggunakan metode naive bayes pada penelitian ini menghasilkan tingkat akurasi sebesar 96%.

2. Nama penulis : (Armansyah, 2016)

Judul Jurnal : **Sistem Pakar Identifikasi Hama Dan Penyakit Tanaman Jagung Berbasis Web**

ISSN : 2302-8149

Sistem pakar ini menggunakan mesin inferensi dengan metode *forward chaining*. Penalaran dilakukan berdasarkan gejala-gejala yang tampak secara fisik terhadap tanaman jagung lalu kemudian disimpulkan nama penyakit yang menyerang tanaman jagung tersebut. Aplikasi sistem pakar ini dapat memudahkan pengguna dalam menentukan jenis hama dan penyakit yang ada pada tanaman jagung.

3. Nama Penulis: (Nusantara, Pamungkas, & Syaifudin, 2017)

Judul Jurnal : **Sistem Pakar Analisa Penyakit Tanaman Cabai merah Menggunakan Metode Backward Chaining**

ISSN :2302-3805

Sistem pakar ini menggunakan metode *backward chaining* yang telah disesuaikan dengan penyakit cabai merah dengan berbagai macam gejala, penyakit, beserta solusinya. Berdasarkan gejala-gejala tersebut maka dibuatlah *rule-rule* yang akan di terapkan dalam mesin inferensi untuk mengetahui penyakit apa yang dialami tenaman cabai merah tersebut. Pada penelitian ini menggunakan bahasa pemrograman *PHP* dengan berbasis *web*.

4. Nama Penulis: (Sihotang, 2018)

Judul Jurnal : **Sistem Pakar Untuk Mendiagnosa Penyakit Pada Tanaman Jagung Dengan Metode Bayes**

ISSN : 2541-3724

Proses pembuatan sistem pakar ini metode kepastiannya teorema bayes diamana metode ini didasarkan dari kondisi awal dimana kondisi awal merupakan kondisi gejala-gejala yang ada kemudian dikenakan aturan yang sudah ditentukan lalu diambil nilai kebenaran yang paling besar untuk menentukan kesimpulan dan solusi dari gejala yang ada. Hasil yang dihasilkan berupa pengurutan data penyakit pada tanaman jagung yang dijadikan sebagai alat bantu dalam pengambilan keputusan bagi petani.

5. Nama Penulis : (Cahyono, 2017)

Judul Jurnal : **Metode Penalaran Sistem Pakar Menggunakan Model Hibrid Fuzzy Dempster Shafer Untuk Identifikasi Hama Dan Penyakit Tanaman Jagung**

ISSN : 2442-7764

Pada sistem pakar ini penelitian menggunakan model *hybrid fuzzy dempster*. Penelitian ini akan menentukan nilai densitas (m) dengan menggunakan logika fuzzy. Sistem pakar yang dirancang ini dapat memberikan hasil identifikasi berdasarkan pada gejala yang diinputkan. Penentuan jenis hama dan penyakit yang menyerang didasarkan pada proses pencarian nilai kepercayaan terhadap suatu diagnosa.

6. Nama penulis : (Guoqi, Yuanxun, Sheng, & Bin, 2014)

   Judul Jurnal : ***An IPC-Based Prolog Design Pattern For Integrating Backward Chaining Inference Into Applications Or Eembedded System***

   ISSN : 1000-9361

   Prolog adalah salah satu yang paling penting untuk membangun sistem pakar dan AI terkait program dan memiliki aplikasi potensial dalam sistem penanaman. Pola desain prolog berasal dari metode mundur yang prinsip dan defenisinya disediakan secara rinci.

7. Nama penulis : (Abu-nasser et al., 2018)

   Judul Jurnal : **Rule Based System For Watermelon Diseases and Treatment**

   ISSN : 2000-002X

   Tujuan utama sistem pakar ini untuk membantu petani dalam mendeteksi penyakit dan solusi semangka. Metode dalam makalah ini di desain sistem pakar yang diusulkan yang diproduksi untuk membantu petani dalam mendiagnosis penyakit semangka. Sistem pakar yang diusulkan menyajikan gambaran tentang penyakit semangka yang diberikan, penyebabnya penyakit diuraikan dan pengobatan penyakit bila memungkinkan diberikan.

2.5 Kerangka Pemikiran

Gambar 2.13 Kerangka Pemikiran
(**Sumber :** Data Olahan, 2018)

Data-data yang dibutuhkan berkaitan dengan penyakit tanaman jagung dianalisis terlebih dahulu agar lebih sederhana dan mudah dilakukan proses pengolahan datanya. Data-datanya tersebut kemudian diolah menggunakan sistem pakar menggunakan metode *backward chaining*. Sistem pakar yang menggunakan metode *backward chaining* ini menggunakan *database* MySQL yang dapat digunakan untuk mendiagnosa penyakit tanaman jagung dan menghasilkan *output* (hasil diagnosa).