BAB II
KAJIAN PUSTAKA

2.1. Teori Dasar

2.1.1. Kecerdasan Buatan

2.1.1.1. Pengertian Kecerdasan Buatan

Menurut (Budiharto & Suhartono, 2014, p. 2), kecerdasan buatan merupakan bidang ilmu komputer yang mempunyai peran penting di era kini dan masa akan datang. Bidang kecerdasan buatan telah berkembang sangat pesat di 20 tahun terakhir seiring dengan pertumbuhan kebutuhan akan perangkat cerdas pada industri dan rumah tangga.

Selama lebih dari ribuan tahun, cara manusia berpikir terus diteliti. Proses tersebut mencangkup cara manusia mengetahui, memahami, memprediksi dan melakukan manipulasi terhadap hal-hal yang lebih besar dan rumit dari yang pernah ada. Bidang keilmuan kecerdasan buatan sampai saat ini terus mencoba untuk melakukan hal tersebut. Tidak hanya untuk memecahkan berbagai masalah, tetapi juga untuk membangun sebuah sistem atau alat yang memiliki kecerdasan. (Budiharto & Suhartono, 2014, pp. 2–3)

AI mencangkup bidang yang cukup besar. Mulai dari yang paling umum hingga yang khusus. Dari *learning* atau *perception* hingga pada permainan catur,

pembuktian teori matematika, menulis puisi, mengemudikan mobil, dan melakukan diagnosis penyakit. Sehingga AI merupakan sebuah ilmu yang *universal*. (Budiharto & Suhartono, 2014, p. 3)

2.1.1.2. Sejarah Kecerdasan Buatan

Kata *intelligence* berasal dari bahasa latin *intellingo* yang berarti "saya paham". Jadi, dasar dari *intelligence* adalah kemampuan memahami dan melakukan aksi. Sebenarnya, area kecerdasan buatan (*Artificial Intelligence*) atau disingkat AI, bermula dari kemunculan komputer sekitar tahun 1940-an, meskipun sejarah perkembangannya dapat dilacak hingga zaman mesir kuno. (Budiharto & Suhartono, 2014, p. 3)

McMulloh dan Pitts pada tahun 1943 mengusulkan model matematis bernama *perceptron* dari neuron di dalam otak. Mereka juga menunjukkan bagaimana neuron menjadi aktif seperti sakelar *on-off*. Neuron tersebut mampu belajar dan memberikan aksi berbeda terhadap waktu dan *input* yang diberikan. (Budiharto & Suhartono, 2014, p. 3)

Paper Alan Turing pada tahun 1950 yang berjudul *Computing Machinery and Intelligence* mendiskusikan syarat sebuah mesin dianggap cerdas. Turing beranggapan bahwa jika mesin dapat dengan sukses berperilaku layaknya manusia, maka mesin itu dapat dianggap cerdas. Kemudian pada akhir tahun 1995, Newel dan Simon mengembangkan *The Logic Theorist*. Program *The Logic Theorist* merupakan program AI pertama dan berdampak besar dalam perkembangan di bidang AI. Program ini merepresentasikan masalah sebagai pohon model, lalu

penyelasaiannya dengan memliki cabang yang akan menghasilkan kesimpulan terbenar. (Budiharto & Suhartono, 2014, p. 4)

Pada tahun 1956, John McCarthy dari Massacuhetts Institute of Technology yang dianggap sebagai bapak AI, menyelenggarakan konferensi *The Dartmouth Summer Research Project on Artificial Intelligence*. Konferensi tersebut bertujuan untuk menarik bakat dan keahlian orang banyak untuk masuk dalam dunia kecerdasan buatan. (Budiharto & Suhartono, 2014, p. 4)

Pada tahun 1960 hingga 1970, muncul berbagai diskusi bagaimana komputer dapat meniru sedetail mungkin kemampuan otak manusia. Masa-masa tersebut dikategorikan sebagai *Classical AI*. Di tahun 1980, komputer semakin mudah diperoleh dengan harga yang lebih murah. Sehingga, berbagai riset di bidang kecerdasan buatan berkembang sangat pesat. (Budiharto & Suhartono, 2014, p. 5)

2.1.1.3. Bidang Ilmu Kecerdasan Buatan

Kecerdasan buatan memiliki tiga buah bidang ilmu yang memiliki pemecahan masalah yang berbeda-beda, berikut ini adalah beberapa bidang ilmu yang termasuk ke dalam kecerdasan buatan:

1. *Fuzzy logic* atau logika fuzzy. Menurut (Budiharto & Suhartono, 2014, p. 151), *fuzzy logic* memiliki derajat keanggotaan dalam rentang 0 (nol) hingga 1 (satu), berbeda dengan logika digital atau diskrit yang hanya memiliki dua nilai yaitu 1 (satu) dan 0 (nol). Logika fuzzy digunakan untuk menerjemahkan suatu besaran yang diekspresikan menggunakan bahasa (*linguistic*). Contohnya,

besaran kecepatan laju kendaraan yang diekspresikan dengan pelan, agak cepat, cepat, dan sangat cepat.

2. *Expert system* atau sistem pakar. Menurut (Kursini, 2008, p. 3), sistem pakar adalah aplikasi berbasis komputer yang digunakan untuk menyelesaikan masalah sebagaimana yang dipikirkan oleh pakar. Pakar yang dimaksud adalah orang yang mempunyai keahlian khusus yang dapat menyelesaikan masalah yang tidak dapat diselesaikan oleh orang awam.
3. Jaringan saraf tiruan. Menurut (Suyanto, 2014, pp. 169–170), jaringan saraf tiruan merupakan salah satu upaya manusia untuk memodelkan cara kerja atau fungsi sistem syaraf manusia dalam melaksanakan tugas tertentu. Pemodelan ini didasari oleh kemampuan otak manusia dalam mengorganisasikan sel-sel penyusunnya yang disebut *neuron*, sehingga mampu melakukan tugas tertentu, khususnya pengenalan pola dengan efektifitas yang sangat tinggi.

2.1.2. Sistem Pakar

2.1.2.1. Pengertian Sistem Pakar

Menurut (Budiharto & Suhartono, 2014, p. 132), sistem pakar adalah program komputer yang menyimulasi penilaian dan perilaku manusia atau organisasi yang memiliki pengetahuan dan pengalaman ahli 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.

Menurut (Hartati & Iswanti, 2008, p. 22), sistem pakar merupakan sistem yang berbasis pengetahuan, mengerjakan tugas yang biasanya dilakukan oleh seorang pakar. Menurut (Merlina & Hidayat, 2012, p. 1), pakar adalah seseorang yang memiliki kemampuan khusus terhadap suatu permasalahan, misalnya: dokter, petani, ahli permesinan dan lain-lainnya.

Berdasarkan penelitian (Resmiati & Supriatna, 2016, p. 192), sistem pakar (*Expert System*) adalah program yang menggabungkan basis pengetahuan (*Knowledge Base*) yang berisi *knowledge* dengan sistem inferensi dan merupakan subset dari kecerdasan buatan (*Artificial Intelligence*). Sistem pakar ditujukan sebagai penyedia nasihat dan sarana bantu dalam memecahkan masalah di bidang spesialisasi tertentu. Program ini akan bertindak sebagai seorang konsultan yang cerdas atau penasehat dalam suatu lingkungan keahlian tertentu.

Dengan sistem pakar, permasalahan yang seharusnya hanya dapat diselesaikan oleh para pakar atau ahli, dapat diselesaikan oleh orang biasa atau awam. Sedangkan, untuk para ahli, sistem pakar membantu aktivitas mereka sebagai asisten yang seolah-olah sudah mempunyai banyak pengalaman. (Budiharto & Suhartono, 2014, pp. 133–134)

2.1.2.2. Manfaat Dan Kemampuan Sistem Pakar

Dengan keahlian sistem pakar yang diperoleh dari pengetahuan manusia, sistem pakar dapat diterapkan pada berbagai bidang untuk memecahkan masalah

tertentu. Berikut ini adalah beberapa manfaat dan kemampuan yang dimiliki oleh sistem pakar, yaitu (Merlina & Hidayat, 2012, pp. 4–5):

1. Meningkatkan *output* dan produktivitas.
2. Menurunkan waktu pengendalian keputusan.
3. Meningkatkan kualitas proses dari produk.
4. Mengurangi *downtime*.
5. Menyerap keahlian pakar.
6. Fleksibilitas.
7. Operasi peralatan yang lebih mudah.
8. Eliminasi kebutuhan peralatan yang mahal.
9. Operasi di lingkungan yang berbahaya.
10. Aksebilitas ke pengetahuan dan *help desk*.
11. Kemampuan untuk bekerja dengan informasi yang tidak akan pernah lengkap atau tidak pasti.
12. Kelengkapan pelatihan.
13. Peningkaan pemecahan masalah dan pengambilan keputusan.
14. Meningkatkan proses pengambilan keputusan.
15. Meningkatkan kualitas keputusan.
16. Kemampuan untuk memecahkan persoalan kompleks.
17. Transfer pengetahuan ke lokasi terpencil.

2.1.2.3. Kelemahan Sistem Pakar

Sistem pakar tidak hanya memiliki kelebihan saja, tetapi ada beberapa kekurangan atau kelemahan yang dimilikinya. Berikut ini adalah beberapa kelemahan yang dimiliki oleh sistem pakar, yaitu (Merlina & Hidayat, 2012, p. 4):

1. Pengetahuan tidak selalu siap tersedia.
2. Akan sulit mengekstrak keahlian dari manusia.
3. Pendekatan tiap pakar pada suatu penilaian situasi mungkin berbeda, tetapi benar.
4. Sulit, bahkan bagi pakar berkemampuan tinggi untuk mengintisarkan penilaian situasi yang baik pada saat berada di dalam tekanan waktu.
5. Penggunaan sistem pakar memiliki batasan kognitif alami.
6. Sistem pakar bekerja dengan baik hanya dalam *domain* pengetahuan sempit.
7. Kebanyakan pakar tidak memiliki sarana mandiri untuk memeriksa apakah kesimpulannya masuk akal.
8. Kosa kata yang digunakan oleh pakar untuk menyatakan fakta dan hubungan.

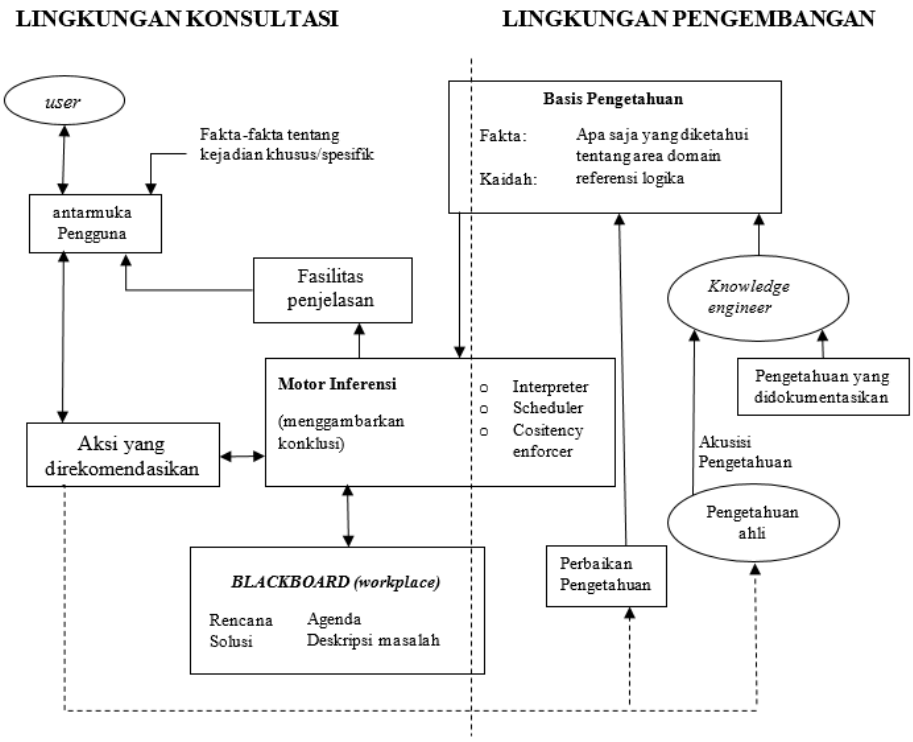
2.1.2.4. Bentuk Sistem Pakar

Setiap sistem memiliki bentuk yang berbeda-beda baik dari segi tampilan hingga beberapa komponen yang terdapat di dalamnya. Hal ini bertujuan untuk menyesuaikan dengan kebutuhan penggunanya. Menurut (Merlina & Hidayat, 2012, p. 3), terdapat empat bentuk sistem pakar, yaitu:

1. Berdiri sendiri. Sistem pakar jenis ini merupakan *software* yang berdiri sendiri, tidak tergantung dengan *software* yang lainnya.

2. Tergabung. Sistem pakar jenis ini merupakan bagian program yang terkandung di dalam suatu algoritma (konvensional) atau merupakan program dimana di dalamnya memanggil algoritma subrutin lain (konvensional).
3. Menggabungkan ke *software* lain. Bentuk ini biasanya merupakan sistem pakar yang menghubungkan ke suatu paket program tertentu, misalnya DBMS.
4. Sistem mengabdi. Sistem pakar merupakan bagian dari komputer khusus yang dihubungkan dengan suatu fungsi tertentu, misalnya sistem pakar yang digunakan untuk membantu menganalisis data radar.

2.1.2.5. Struktur Sistem Pakar



Gambar 2.1 Struktur sistem pakar
Sumber: (Hartati & Iswanti, 2008, p. 9)

Menurut (Hartati & Iswanti, 2008, p. 8), sistem pakar dapat dilihat dari sudut pandang lingkungan (*environment*) dalam sistem. Terdapat dua lingkungan yaitu lingkungan konsultasi dan lingkungan pengembangan. Pada lingkungan konsultasi diperuntukan bagi pengguna non pakar untuk melakukan konsultasi dengan sistem yang tujuannya adalah untuk mendapatkan nasehat pakar. Sedangkan, pada lingkungan pengembangan ditujukan bagi pembangun sistem pakar untuk membangun komponen dan memasukkan pengetahuan hasil akusisi pengetahuan ke dalam basis pengetahuan.

Menurut (Hartati & Iswanti, 2008, pp. 4–7), berikut ini adalah struktur yang terdapat pada sistem pakar:

1. Antarmuka pengguna

Sistem pakar harus menyediakan pendukung yang diperlukan oleh pemakai yang tidak memahami masalah teknis. Sistem pakar juga menyediakan komunikasi antara sistem dan pemakainya, yang disebut antarmuka. Antarmuka yang efektif dan ramah pengguna penting sekali terutama bagi pemakai yang tidak ahli dalam bidang yang diterapkan pada sistem pakar. (Hartati & Iswanti, 2008, pp. 4–5)

2. Basis pengetahuan

Basis pengetahuan merupakan kumpulan pengetahuan bidang tertentu pada tingkatan pakar dalam format tertentu. Pengetahuan ini diperoleh dari akumulasi pengetahuan pakar dan sumber-sumber pengetahuan lainnya. Basis pengetahuan bisa berkembang dari waktu ke waktu, karena pengetahuan selalu bertambah. (Hartati & Iswanti, 2008, p. 5)

3. Mesin inferensi

Mesin inferensi merupakan otak dari sistem pakar, berupa perangkat lunak yang melakukan tugas inferensi penalaran sistem pakar, biasa dikatakan sebagai mesin pemikir (*Thinking Machine*). Pada prinsipnya mesin inferensi inilah yang akan mencari solusi dari suatu permasalahan. (Hartati & Iswanti, 2008, p. 5)

4. Memori kerja

Memori kerja merupakan bagian dari sistem pakar yang menyimpan fakta-fakta yang diperoleh saat dilakukan proses konsultasi. Fakta-fakta inilah yang nantinya akan diolah oleh mesin inferensi berdasarkan pengetahuan yang disimpan dalam basis pengetahuan untuk menentukan suatu keputusan pemecahan masalah. (Hartati & Iswanti, 2008, p. 6)

5. Fasilitas penjelasan

Fasilitas penjelasan dapat memberikan informasi kepada pemakai mengenai jalannya penalaran sehingga dihasilkan suatu keputusan. Bentuk penjelasannya dapat berupa keterangan yang diberikan setelah suatu pertanyaan diajukan. Tujuan adanya fasilitas penjelasan dalam sistem pakar antara lain membuat sistem menjadi lebih cerdas, menunjukkan adanya proses analisa dan memuaskan psikologis pemakai. (Hartati & Iswanti, 2008, pp. 6–7)

6. Fasilitas akusisi pengetahuan

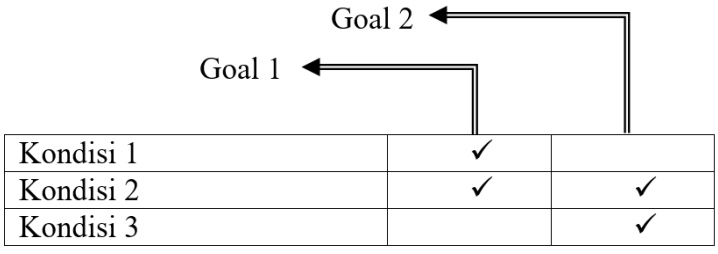
Pengetahuan sistem pakar dapat ditambahkan kapan saja pengetahuan baru diperoleh atau saat pengetahuan yang sudah ada tidak berlaku lagi. Untuk melakukan proses penambahan ini sistem pakar dilengkapi dengan fasilitas akusisi pengetahuan. Akusisi pengetahuan adalah proses pengumpulan, pemindahan dan

transformasi dari keahlian pemecahan masalah yang berasal dari berbagai sumber ke dalam bentuk yang dimengerti oleh komputer. Dengan adanya fasilitas ini, maka seorang pakar akan mudah menambahkan pengetahuan atau kaidah baru pada sistem pakar. (Hartati & Iswanti, 2008, p. 7)

2.1.2.6. Tabel Keputusan dan Pohon Keputusan

Tabel keputusan merupakan suatu cara untuk mendokumentasikan pengetahuan. Tabel keputusan merupakan matrik kondisi yang dipertimbangkan dalam pendeskripsian kaidah. Kaidah yang disajikan dalam bentuk kaidah produksi disusun dari tabel keputusan. (Hartati & Iswanti, 2008, p. 26)

Pengetahuan relasi dapat direpresentasikan dalam tabel keputusan. Dalam tabel keputusan, pengetahuan disusun dalam proses *spreadsheet* menggunakan kolom dan baris. Tabel dibagi menjadi dua bagian. Pertama dikembangkan suatu daftar atribut, dan untuk tiap atribut dirinci semua kemungkinan nilai. Kemudian daftar kesimpulan dikembangkan. Akhirnya, kombinasi atribut yang berbeda disesuaikan terhadap kesimpulan. Pengetahuan untuk tabel dikumpulkan dalam sesi akusisi pengetahuan. Setelah terbentuk, pengetahuan dalam tabel dapat digunakan sebagai *input* untuk metode representasi pengetahuan yang lain. (Merlina & Hidayat, 2012, p. 13)

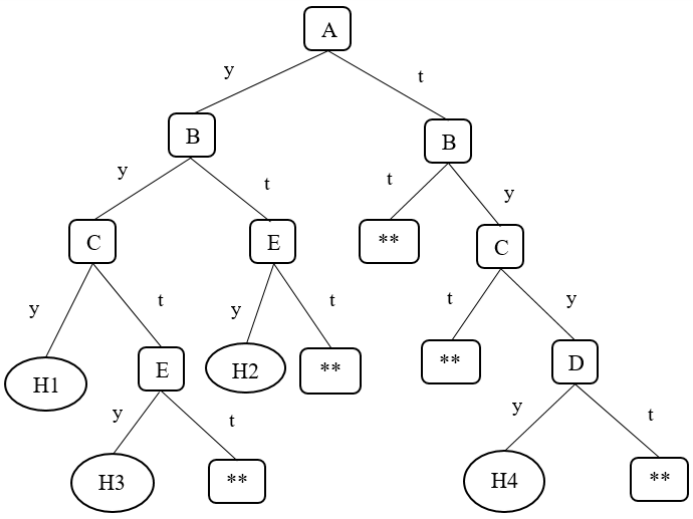


Gambar 2.2 Tabel Keputusan
Sumber: (Hartati & Iswanti, 2008, p. 26)

Menurut (Merlina & Hidayat, 2012, p. 14) pohon keputusan adalah sebuah jawaban akan sebuah sistem atau cara yang kita kembangkan untuk membantu mencari dan membuat keputusan untuk masalah-masalah tersebut. Dengan pohon keputusan, manusia dapat dengan mudah mengidentifikasi dan melihat hubungan antara faktor-faktor yang mempengaruhi suatu masalah dan dapat mencari penyelesaian terbaik dengan memperhitungkan faktor-faktor tersebut.

Pohon keputusan juga dapat menganalisis nilai resiko dan nilai suatu informasi yang terdapat dalam suatu alternatif pemecahan masalah. Pohon keputusan memiliki peranan sebagai alat bantu dalam mengambil keputusan (*decision support tool*). (Merlina & Hidayat, 2012, p. 14)

Pohon terbentuk dari suatu *graph*. Pohon adalah sebuah *graph*, tak berarah, terhubung yang tidak mengandung sirkuit. *Graph* adalah suatu representasi visual dari objek-objek diskrit yang dinyatakan dengan noktah, bulatan, atau titik, serta hubungan yang ada antara objek-objek tersebut. (Merlina & Hidayat, 2012, p. 14)



Gambar 2.3 Pohon keputusan
Sumber: (Hartati & Iswanti, 2008, p. 33)

2.1.2.7. Basis Pengetahuan Sistem Pakar

Menurut (Merlina & Hidayat, 2012, p. 3), basis pengetahuan berisi pengetahuan-pengetahuan untuk menyelesaikan masalah yang masih dalam *domain* tertentu. Basis pengetahuan memiliki dua bentuk pendekatan yang sangat umum digunakan, yaitu (Merlina & Hidayat, 2012, pp. 3–4):

1. Penalaran berbasis aturan (*Rule-Base Reasoning*)

Pada penalaran berbasis aturan, pengetahuan direpresentasikan dengan menggunakan aturan berbentuk: IF-THEN. Bentuk ini digunakan apabila kita memiliki sejumlah pengetahuan pakar pada suatu permasalahan tertentu dan pakar dapat menyelesaikan masalah tersebut secara berurutan. Disamping itu, bentuk ini juga digunakan apabila dibutuhkan penjelasan tentang jejak (langkah-langkah) pencapaian solusi. (Merlina & Hidayat, 2012, pp. 3–4)

2. Penalaran berbasis kasus (*Case-Base Reasoning*)

Pada penalaran berbasis kasus, basis pengetahuan akan berisi solusi-solusi yang telah dicapai sebelumnya, kemudian akan diturunkan suatu solusi untuk keadaan yang terjadi sekarang (fakta yang ada). Bentuk ini digunakan apabila *user* menginginkan untuk tahu lebih banyak lagi pada kasus-kasus yang hampir sama (mirip). Selain itu, bentuk ini digunakan apabila kita telah memiliki sejumlah situasi atau kasus tertentu dalam basis pengetahuan. (Merlina & Hidayat, 2012, p. 4)

2.1.2.8. Metode Inferensi

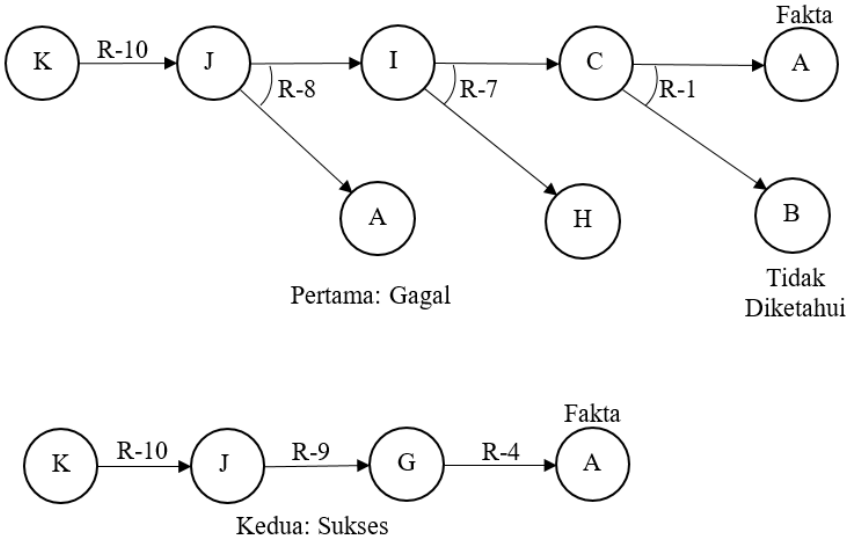
Menurut (Merlina & Hidayat, 2012, p. 21), Metode inferensi pada sistem pakar adalah bagian yang menyediakan mekanisme fungsi berfikir dan pola-pola

penalaran sistem yang digunakan oleh seorang pakar. Terdapat dua pendekatan dalam menentukan metode inferensi, yaitu (Merlina & Hidayat, 2012:21-22):

1. *Backward Chaining*

Menurut (Merlina & Hidayat, 2012, p. 21), *backward chaining* adalah pendekatan *goal-driven* yang dimulai dari harapan apa yang akan terjadi (hipotesis) dan kemudian mencari bukti yang mendukung (atau berlawanan) dengan harapan. Sedangkan menurut (Hartati & Iswanti, 2008, p. 46), runut balik atau *backward chaining* merupakan proses perunutan yang arahnya kebalikan dari runut maju.

Proses penalaran runut balik dimulai dengan tujuan atau goal kemudian merunut balik ke jalur yang akan mengarahkan ke goal tersebut, mencari bukti-bukti bahwa bagian kondisi terpenuhi. Jadi secara umum runut balik itu diaplikasikan ketika tujuan atau hipotesis yang dipilih itu sebagai titik awal penyelesaian masalah. (Hartati & Iswanti, 2008, p. 46)



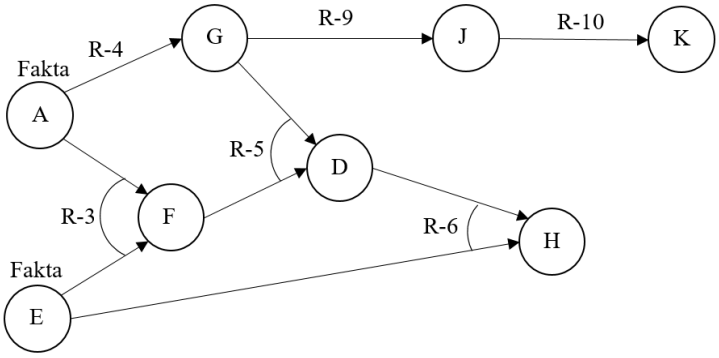
Gambar 2.4 Cara kerja mesin inferensi *backward chaining*
Sumber: (Merlina & Hidayat, 2012, p. 22)

Pemodelan runut balik atau *backward chaining* dapat dimodelkan sebagai berikut (Hartati & Iswanti, 2008, p. 46):

Tujuan,
IF (kondisi).

2. *Forward Chaining*

Menurut (Merlina & Hidayat, 2012, p. 22), *forward chaining* adalah pendekatan *data-driven* yang dimulai dari informasi yang tersedia atau dari ide dasar, kemudian mencoba menarik kesimpulan. Sedangkan menurut (Hartati & Iswanti, 2008, p. 45), runut maju atau *forward chaining* merupakan proses perunutan yang dimulai dengan menampilkan data atau fakta yang meyakinkan menuju konklusi akhir.



Gambar 2.5 Cara kerja mesin inferensi *forward chaining*
Sumber: (Merlina & Hidayat, 2012, p. 22)

Forward chaining dimulai dari premis-premis atau informasi masukan (if) dahulu kemudian menuju konklusi atau *derived information* (then). Informasi masukan dapat berupa data, bukti, temuan, atau pengamatan. Sedangkan konklusi dapat berupa tujuan, hipotesa, penjelasan, atau diagnosis. Sehingga jalannya penalaran runut maju dapat dimulai dari data menuju tujuan, dari bukti menuju

hipotesa, dari temuan menuju penjelasan, atau dari pengamatan menuju diagnosa. (Hartati & Iswanti, 2008, p. 45)

Pemodelan runut maju atau *forward chaining* dapat dimodelkan sebagai berikut (Hartati & Iswanti, 2008, p. 45):

IF (informasi masukan)
THEN (konklusi)

2.1.2.9. Ciri dan Karakteristik Sistem Pakar

Beberapa s*oftware* atau sistem selalu memiliki ciri dan karakteristik yang berbeda. Berikut ini adalah ciri dan karakteristik yang menunjukkan sistem pakar, yaitu (Andi, 2009, pp. 6–7):

1. Pengetahuan sistem pakar merupakan suatu konsep, bukan berbentuk numeris. Hal ini dikarenakan komputer melakukan proses pengolahan data secara numerik sedangkan keahlian seorang pakar adalah fakta dan aturan-aturan, bukan numerik.
2. Informasi dalam sistem pakar tidak selalu lengkap, subyektif, tidak konsisten, subyek terus berubah dan tergantung pada kondisi lingkungan sehingga keputusan yang diambil bersifat tidak pasti dan tidak mutlak “ya” atau “tidak” akan tetapi menurut ukuran kebenaran tertentu. Oleh karena itu dibutuhkan kemampuan sistem untuk belajar secara mandiri dalam menyelesaikan masalah-masalah dengan pertimbangan-pertimbangan khusus.
3. Kemungkinan solusi sistem pakar terhadap suatu permasalahan adalah bervariasi dan mempunyai banyak pilihan jawaban yang dapat diterima, semua faktor yang ditelusuri memiliki ruang masalah yang luas dan tidak pasti. Oleh

karena itu diperlukan fleksibilitas sistem dalam menangani kemungkinan solusi dari berbagai permasalahan.

4. Perubahan atau pengembangan pengetahuan dalam sistem pakar dapat terjadi setiap saat bahkan sepanjang waktu sehingga diperlukan kemudahan dalam modifikasi sistem untuk menampung jumlah pengetahuan yang semakin besar dan semakin bervariasi.
5. Pandangan dan pendapat setiap pakar tidak selalu sama, yang oleh karena itu tidak ada jaminan bahwa solusi sistem pakar merupakan jawaban yang pasti benar. Setiap pakar akan memberikan pertimbangan-pertimbangan berdasarkan faktor subjektif.
6. Keputusan merupakan bagian terpenting dari sistem pakar. Sistem pakar harus memberikan solusi yang akuran berdasarkan masukan pengetahuan meskipun solusinya sulit sehingga fasilitas informasi sistem harus selalu diperlukan.

2.1.2.10. Bidang-Bidang Pengembangan Sistem Pakar

Terdapat berbagai kategori pengembangan sistem pakar, antara lain (Andi, 2009, pp. 7–8):

1. Kontrol. Contoh pengembangan ini banyak ditemukan dalam kasus pasien di rumah sakit, dimana dengan kemampuan sistem pakar dapat dilakukan control terhadap cara pengobatan dan perawatan melalui sensor data atau kode alarm dan memberikan solusi terapi pengobatan yang tepat bagi pasien yang sakit.
2. Desain. Contoh sistem pakar di bidang ini adalah PEACE yang dibuat oleh Dincbas pada tahun 1980 untuk membantu pengembangan sirkuit elektronik.

Selain itu, sistem pakar ini juga untuk membantu desain komputer dengan komponen-komponennya.

3. Diagnosis. Pengembangan sistem pakar terbesar adalah di bidang diagnosis, seperti diagnosis penyakit, diagnosis kerusakan mesin kendaraan bermotor, diagnosis kerusakan komponen komputer dan lain-lain.
4. Instruksi. Instruksi merupakan pengembangan sistem pakar yang sangat berguna dalam bidang ilmu pengetahuan dan pendidikan, dimana sistem pakar dapat memberikan instruksi dan pengajaran tertentu terhadap suatu topik masalah.
5. Intepretasi. Sistem pakar yang dikembangkan dalam bidang interpretasi melakukan proses pemahaman akan suatu situasi dari beberapa informasi yang direkam.
6. Monitor. Sistem pakar di bidang ini banyak digunakan militer, yaitu menggunakan sensor radar kemudian menganalisisnya dan menentukan posisi obyek berdasarkan posisi radar tersebut.
7. Perencanaan. Perencanaan banyak digunakan dalam bidang bisnis dan keuangan suatu proyek, di mana sistem pakar dalam membuat perencanaan suatu pekerjaan berdasarkan jumlah tenaga kerja, biaya dan waktu sehingga pekerjaan menjadi lebih efisien dan lebih optimal.
8. Prediksi. Sistem pakar ini mampu memprediksi kejadian masa mendatang berdasarkan informasi dan model permasalahan yang dihadapi. Biasanya sistem memberikan simulasi kejadian masa mendatang tersebut, misalnya

memprediksi tingkat kerusakan tanaman apalagi terserang hama dalam jangka waktu tertentu.

9. Seleksi. Sistem pakar dengan seleksi mengidentifikasi pilihan baik dari beberapa daftar pilihan kemungkinan solusi. Biasanya sistem mengidentifikasi permasalahan secara spesifik kemudian memcoba untuk menemukan solusi yang paling mendekati kebenaran.
10. Simulasi. Sistem ini memproses operasi dari beberapa variasi kondisi yang ada dan menampilkannya dalam bentuk simulasi. Salah satu contohnya adalah program PLANT yang sudah menggabungkan antara prediksi dan simulasi, di mana program tersebut mampu menganalisis hama dengan berbagai kondisi suhu dan cuaca.

2.1.2.11. Mengembangkan Sistem Pakar

Pengembangan sistem pakar sama seperti pengembangan perangkat lunak pada umumnya. Berikut ini adalah beberapa tahap-tahap atau fase-fase pengembangan sistem pakar (Andi, 2009, pp. 19–21):

1. Identifikasi

Tahap ini merupakan tahap penentuan hal-hal penting sebagai dasar dari permasalahan yang akan dianalisis. Tahap ini merupakan tahap untuk mengkaji dan membatasi masalah yang akan diimplementasikan dalam sistem. Setiap masalah yang diidentifikasi harus dicari solusi, fasilitas yang akan dikembangkan, penentuan jenis bahasa pemprogram dan tujuan yang ingin dicapai dari proses pengembangan tersebut. (Andi, 2009, p. 19)

2. Konseptualisasi

Hasil identifikasi masalah dikonseptualisasikan dalam bentuk relasi antar data, hubungan antar pengetahuan dan konsep-konsep penting dan ideal yang akan diterapkan dalam sistem. Konseptualisasi juga menganalisis data-data penting yang harus didalami bersama dengan pakar dibidang permasalahan tersebut. Tujuan dilakukan ini adalah untuk memperoleh konfirmasi hasil wawancara dan observasi sehingga hasilnya dapat memberikan jawaban pasti bahwa sasaran permasalahan tepat, benar dan sudah sesuai. (Andi, 2009, p. 20)

3. Formalisasi

Di tahap formalisasi konsep-konsep tersebut diimplementasikan secara formal, misalnya memberikan kategori sistem yang akan dibangun, mempertimbangkan beberapa faktor pengambilan keputusan seperti keahlian manusia, kesulitan dan tingkat kesulitan yang mungkin terjadi, dokumentasi kerja dan sebagainya. (Andi, 2009, p. 20)

4. Implementasi

Di tahap implementasi dapat dimulai dengan membuat garis besar masalah kemudian memecahkan masalah ke dalam modul-modul. Untuk memudahkan maka harus diidentifikasi inputan, proses, basis aturannya, hasil dan kesimpulannya. Sesudah itu semuanya diubah dalam bahasa yang mudah dimengerti oleh komputer. (Andi, 2009, p. 20)

5. Evaluasi

Sistem pakar yang selesai dibangun, perlu untuk dievaluasi untuk menguji dan menemukan kesalahan. Hal ini merupakan hal yang umum dilakukan karena

suatu sistem belum tentu sempurna setelah selesai pembuatannya sehingga proses evaluasi diperlukan untuk penyempurnaannya. Dalam evaluasi akan ditemukan bagian-bagian yang harus dikoreksi untuk menyamakan permasalahan dan tujuan akhir pembuatan sistem. (Andi, 2009, pp. 20–21)

6. Pengembangan Sistem

Pengembangan sistem diperlukan sehingga sistem yang dibangun tidak menjadi usang dan investasi sistem tidak sia-sia. Hal pengembangan sistem yang paling berguna adalah proses dokumentasi sistem di mana di dalamnya tersimpan semua hal penting yang dapat menjadi tolok ukur pengembangan sistem di masa mendatang termasuk di dalamnya adalah kamus pengetahuan masalah yang diselesaikan. (Andi, 2009, p. 21)

2.2. Variabel

Dalam penelitian ini, objek yang digunakan dalam penelitian ini adalah tanaman salak. Terdapat dua variabel yang ditetapkan dalam penelitian ini. Variabel pertama adalah penyakit tanaman salak dan variabel dua adalah hama tanaman salak. Dibawah ini adalah beberapa penjelasan mengenai objek yang diteliti dan kedua variabel yang peneliti pilih untuk memenuhi kebutuhan sistem pakar ini.

2.2.1. Tanaman Salak

Tanaman salak dikenal sebagai tanaman yang tumbuh merumpun. Batangnya sangat pendek tertutup pelepah daun. Seluruh permukaan tanaman tertutup duri tajam. Tanaman salak tergolong tanaman buah tahunan, yaitu hidup menahun

(*perenial*) dengan umur dapat mencapai ratusan tahun. Tanaman salak yang telah berumur 200 tahun, produksinya masih baik, rata-rata 4 kg/pohon/tahun. Namun di perkebunan budidaya umumnya umur produksi dibatasi sampai 30-50 tahun. Pohon salak tidak bercabang dan berhabitus perdu, dengan tinggi tanaman salak dapat mencapai 7 m atau lebih dengan lingkaran berkisar 29-41 cm. Tanaman salak berbuah sepanjang tahun. Pada umumnya tanaman salak berbuah dua kali setahun. Panen raya umumnya terjadi pada bulan Desember sampai Februari, sedangkan panen gadu terjadi pada bulan Juni sampai Agustus. (Cahyono, 2016, p. 15)

Secara morfologis, organ-organ penting pada tanaman salak adalah sebagai berikut (Cahyono, 2016, pp. 16–21):

1. Akar

Tanaman salak berakar serabut (tidak memiliki akar tunggal). Penyebaran akar serabut tidak begitu dalam dan tidak begitu luas (dangkal). Perakaran tanaman salak mudah rusak jika kekurangan air. Di tanah yang gembur dan subur, perakaran akan tumbuh dan berkembang baik. Akar tanaman berfungsi sebagai penopang berdirinya tanaman dan penyerapan zat-zat makanan (hara) serta air dari tanah. Kondisi fisik tanah yang gembur dan subur sangat baik untuk pertumbuhan akar dan pertumbuhan tanaman karena penyerapan air dan zat-zat hara dapat berjalan sempurna. (Cahyono, 2016, p. 16)

2. Batang

Batang tanaman salak sangat pendek, berkayu dan keras. Bentuk batang mirip batang tanaman kurma atau kelapa. Batang tertutup oleh pelepah-pelepah daun. Batang tanaman dapat mencapai tinggi 7 m atau lebih, namun rata-rata tingginya

kurang dari 4,5 m. Batang tanaman berfungsi sebagai jalan pengangkutan air dan zat-zat hara ke daun serta sebagai jalan pengangkutan zat-zat hasil asimilasi ke seluruh bagian tubuh tanaman. (Cahyono, 2016, pp. 16–17)

3. Daun

Daun tanaman salak umumnya pecah-pecah, tumbuh pada pelepah daun. Anak-anak daun berbentuk menyirip. Permukaan daun bagian atas berwarna hijau tua dan bagian bawah hijau keabu-abuan atau putih seperti berlapis lilin. Panjang pelepah daun berkisar antara 3,5-6 m. Daun tanaman merupakan bagian tumbuhan yang berfungsi sebagai tempat berlangsungnya proses asimilasi yang menghasilkan zat-zat yang diperlukan tanaman untuk pertumbuhan vegetatif (batang, akar, daun) dan pertumbuhan generatif (bunga, buah, dan biji). (Cahyono, 2016, p. 17)

4. Bunga

Bunga tumbuh bergerombol, tersusun seperti genteng dalam tandan atau tongkol yang dilindungi oleh seludung bunga. Bunga berwarna merah jambu dengan aroma seperti aroma bunga pisang atau jambe. Bunga tanaman salak tergolong bunga tidak sempurna (berumah dua), yaitu pada satu pohon hanya terdapat bunga jantan atau bunga betina saja sehingga tanaman jantan hanya mempunyai bunga jantan dan tanaman betina hanya mempunyai bunga betina. (Cahyono, 2016, pp. 17–19)

Sistem penyerbukan tanaman salak adalah penyerbukan silang. Secara alami, penyerbukan berlangsung dengan bantuan angin dan serangga. Namun, penyerbukan juga dapat terjadi dengan bantuan manusia. Penyerbukan bunga yang berlangsung dengan bantuan angin memberikan hasil rendah, sedangkan

penyerbukan yang berlangsung dengan bantuan serangga dan manusia dapat memberikan hasil yang lebih tinggi. (Cahyono, 2016, pp. 19–20)

5. Buah

Pada umumnya buah salak berbentuk bulat atau bulat telur terbalik dengan salah satu ujung meruncing. Buah terangkai rapat dalam tandan buah. Kulit buah tersusun dari sisik-sisik dengan susunan seperti genteng atap rumah. Kulit berwarna coklat kekuningan sampai coklat kehitaman dengan bagian ujung buah lebih mengilap. Daging buah tidak berserat, ada yang masir, ada yang berwarna putih kekuningan, putih atau kekuningan kecoklatan. Tekstur daging buah bersifat agak keras dan renyah. (Cahyono, 2016, p. 20)

6. Biji

Biji buah salak berbentuk persegi sampai bulat agak gepeng, berwarna coklat muda hingga berwarna coklat kehitaman atau cokelat tua. Biji sangat keras dan berkeping satu. Biji salak digunakan untuk perbanyakan tanaman (pembiakan). Namun sejauh ini biji salak tidak banyak digunakan untuk perbanyakan tanaman. Perbanyakan tanaman salak umumnya dilakukan dengan cangkok. (Cahyono, 2016, p. 21)

2.2.2. Penyakit Tanaman Salak

Menurut (Cahyono, 2016, p. 84), penyakit yang menyerang tanaman bukanlah disebabkan binatang, melainkan disebabkan oleh makhluk mikroskopis, misalnya bakteri, virus, cendawan (jamur) dan lain-lainnya. Dibawah ini adalah

beberapa kerugian yang ditimbulkan akibat serangan penyakit (Cahyono, 2016, pp. 84–85):

1. Tanaman mengalami gangguan fisiologi sehingga pertumbuhannya terlambat.
2. Menurunkan hasil panen, baik dalam hal kuantitas maupun kualitas.
3. Dapat menimbulkan infeksi sekunder sehingga menimbulkan kerusakan yang lebih parah.
4. Biaya produksi menjadi lebih besar karena harus mengeluarkan biaya untuk obat-obatan dan tenaga kerja untuk penanganannya.

Penyakit yang sering dijumpai menyerang tanaman salak adalah busuk bunga dan busuk buah. Pada musim penghujan atau pada saat kelembapan di sekitar tanaman cukup tinggi, seringkali dijumpai bunga yang tidak dapat berkembang menjadi bakal buah karena busuk dan juga buah yang busuk dan gugur sebelum dipanen. (Cahyono, 2016, pp. 91–92)

2.2.3. Hama Tanaman Salak

Tanaman salak pasti tidak hanya terserang penyakit saja, tetapi ada faktor lain seperti hama yang menyebabkan kerusakan pada tanaman salak. Menurut (Cahyono, 2016, p. 84), hama adalah binatang yang dianggap dapat mengganggu atau merusak tanaman dengan memakan bagian tanaman yang disukainya.

Keberadaan hama dan penyakit dapat menimbulkan kerusakan tanaman yang pada akhirnya akan menurunkan produksi dan menimbulkan kerugian ekonomi.

Serangan pada tanaman salak dapat datang mendadak dan dapat bersifat meluas (eksplosif) sehingga dalam waktu yang relatif singkat dapat mematikan tanaman dalam jumlah banyak dan menggagalkan panen. (Cahyono, 2016, p. 84)

2.3. Software Pendukung

2.3.1. UML

2.3.1.1. Tentang UML

UML merupakan salah satu pemodelan yang saat ini paling banyak digunakan. UML (*Unified Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemprograman berorientasi objek. (A.S & Shalahuddin, 2014, p. 133)

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang di berbagai negara dapat mengerti pemodelan perangkat lunak. Pada perkembangan teknik pemprograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemprograman berorientasi objek, yaitu UML atau *Unified Modeling Language*. (A.S & Shalahuddin, 2014, p. 137)

UML muncul karena adanya kebutuhan pemodelan 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. (A.S & Shalahuddin, 2014, p. 137)

2.3.1.2. Pemodelan

Menurut (A.S & Shalahuddin, 2014, p. 135), pemodelan adalah gambaran dari realita yang simpel dan dituangkan dalam bentuk pemetaan dengan aturan tertentu. Salah satu perangkat pemodelan adalah *Unified Modeling Language* (UML) (A.S & Shalahuddin, 2014, p. 137).

Pada dunia pembangunan perangkat lunak sistem informasi juga diperlukan pemodelan. Pemodelan perangkat lunak digunakan untuk mempermudah langkah berikutnya dari pengembangan sebuah sistem informasi sehingga lebih terencana. Seperti halnya maket, pemodelan pada pembangunan perangkat lunak digunakan untuk memvisualkan perangkat lunak yang akan dibuat. (A.S & Shalahuddin, 2014, p. 136)

Pemodelan perangkat lunak memiliki beberapa abstraksi sebagai berikut (A.S & Shalahuddin, 2014, p. 136):

- Petunjuk yang terfokus pada proses yang dimiliki oleh sistem.
- Spesifikasi struktur secara abstrak dari suatu sistem (belum detail).
- Spesifikasi lengkap dari sebuah sistem yang sudah *final*.
- Spesifikasi umum atau khusus sistem.

- Bagian penuh atau parsial dari sebuah sistem.

Perangkat pemodelan adalah suatu model yang digunakan untuk menguraikan sistem menjadi bagian-bagian yang dapat diatur dan mengomunikasi ciri konseptual dan fungsional kepada pengamat (A.S & Shalahuddin, 2014:136). Berikut ini adalah beberapa peran dari perangkat pemodelan (A.S & Shalahuddin, 2014, pp. 136–137):

- Komunikasi

  Perangkat Pemodelan dapat digunakan sebagai alat komunikasi antara pemakai dengan analis sistem maupun *developer* dalam pengembangan sistem.

- Eksperimentasi

  Pengembangan sistem yang bersifat "*trial and error".*

- Prediksi

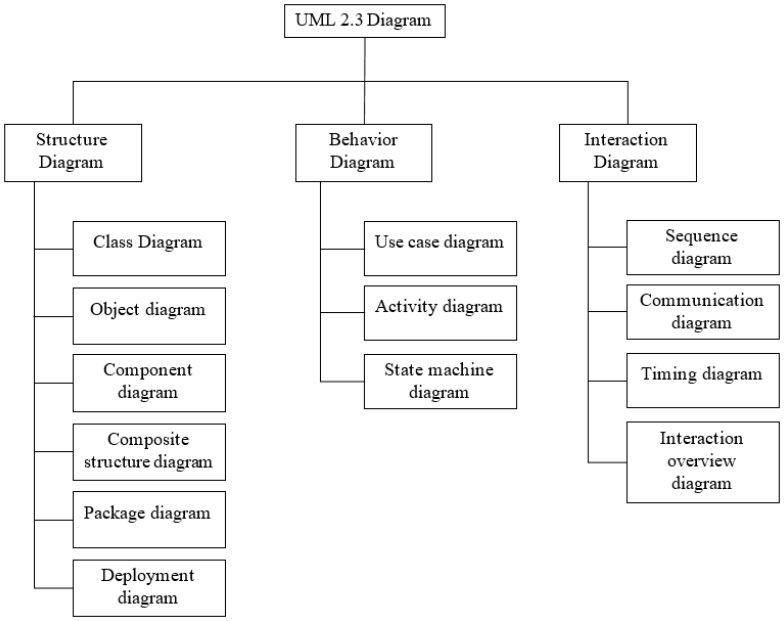
  Model meramalkan bagaimana suatu sistem akan bekerja.

2.3.1.3. Diagram UML

Menurut (A.S & Shalahuddin, 2014, p. 140), UML 2.3 memiliki 13 macam diagram yang dikelompokkan dalam 3 kategori. Berikut ini adalah pembagian kategori UML 2.3 (A.S & Shalahuddin, 2014, p. 141):

- *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

- *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
- *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.



Gambar 2.6 Diagram UML
Sumber: (A.S & Shalahuddin, 2014, p. 140)

2.3.1.4. *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. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. (A.S & Shalahuddin, 2014, p. 155)

Menurut (A.S & Shalahuddin, 2014, p. 155), syarat penamaan *use case* adalah nama yang didefinisikan harus sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* dalam pendefinisian apa yang disebut aktor dan *use case* (A.S & Shalahuddin, 2014, p. 155):

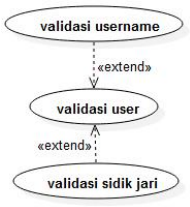
- Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
- *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antara unit atau aktor.

Berikut ini adalah simbol-simbol yang ada pada diagram *use case* (A.S & Shalahuddin, 2014, pp. 156–158):

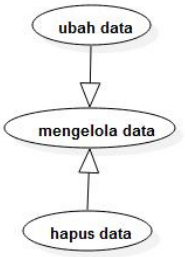
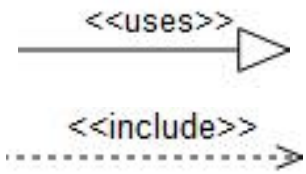
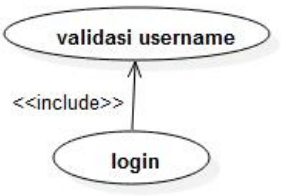
Tabel 2.1 Simbol-simbol diagram *use case*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antara satu 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 gambar orang, tapi aktor belum tentu

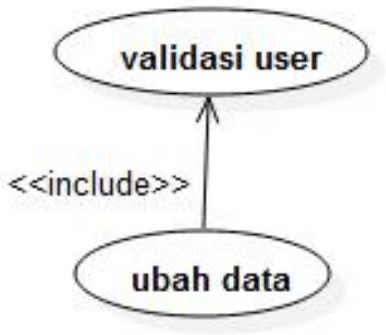
Tabel 2.1 Lanjutan

Simbol	Deskripsi
	merupakan orang; bisanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi / *association*	Komunikasi antara aktor dengan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi / *extend* «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 pemprogram berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misal  arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
Generalisasi / *generalization*	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah

Tabel 2.1 Lanjutan

Simbol	Deskripsi
	fungsi yang lebih umum dari lainnya, misalnya:  arah panah mengarah pada *use case* yang menjadi generalisasinya. (umum)
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. Ada dua sudut pandang yang cukup besar mengenai include di *use case*: • Include berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, misal pada kasus berikut:  • Include berarti *use case* yang ditambahkan akan selalu melakukan pengecekan apakah

Tabel 2.1 Lanjutan

Simbol	Deskripsi
	use case yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut:  Kedua interpretasi di atas dapat dianut saat satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

Sumber: (A.S & Shalahuddin, 2014, pp. 156–158)

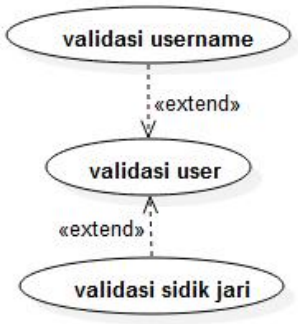
Use case nantinya akan menjadi kelas proses pada diagram kelas sehingga perlu dipertimbangkan penamaan yang dilakukan apakah sudah layak menjadi kelas atau belum sesuai aturan pendefinisian kelas yang baik (A.S & Shalahuddin, 2014, p. 159). Berikut ini adalah aturan perubahan *use case* yang layak menjadi kelas proses (A.S & Shalahuddin, 2014, pp. 159–160):

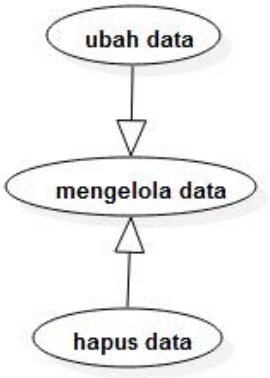
Tabel 2.2 Perubahan *use case* yang layak menjadi kelas proses

Hubungan	Keterangan
Ekstensi / *extend*	Pada hubungan ekstensi maka dapat hanya diambil *use case* induknya yang dijadikan kelas dengan metode berupa

Tabel 2.2 Lanjutan

Hubungan	Keterangan

	*use case* ekstensinya.
	Class ValidasiUser{
	//atribut
	…….
	prosedure
	validasiUsername(){
	//proses
	…….
	}
	procedure
	validasiSidikJari (){
	//proses
	…….
	}
	}

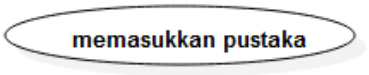
Generalisasi / *generalization*
 
	Pada hubungan generalisasi maka dapat hanya diambil *use case* umumnya dijadikan kelas dengan metode berupa *use case* khususnya.
	Class MengelolaData{
	//atribut
	…….
	prosedure ubahData(){
	//proses
	…….
	}
	procedure hapusData(){
	//proses
	…….
	}
	}

Use case yang berdiri sendiri 	Metode yang mungkin bisa ada di dalam kelas proses *login* adalah sebagai berikut:

Tabel 2.2 Lanjutan

Hubungan	Keterangan

	Class Login{
	//atribut
	……..
	prosedure login(){
	//proses
	…….
	}
	procedure logout(){
	//proses
	…….
	}
	}

Use case yang kurang tepat sebagai sebuah *use case* yang berdiri sendiri
 
 	
	Kurang tepat karena kelasnya akan menjadi:
	Class MemasukkanPustaka{
	//atribut
	……..
	prosedure memasukkanPustaka(){
	//proses
	…….
	}
	}
	Kelas yang hanya terdiri dari satu metode sebenarnya kurang efisien.

Sumber: (A.S & Shalahuddin, 2014, pp. 159–160)

Menurut (A.S & Shalahuddin, 2014, p. 161), skenario *use case* adalah alur jalannya proses *use case* dari sisi aktor dan sistem. Berikut adalah format tabel skenario *use case* (A.S & Shalahuddin, 2014, p. 161):

Tabel 2.3 Tabel skenario *use case*

Aksi Aktor	Reaksi Sistem
Skenario Normal	
Skenario Alternatif	

Sumber: (A.S & Shalahuddin, 2014, p. 161)

Skenario *use case* dibuat per *use case* terkecil, misalkan untuk generalisasi maka skenario yang dibuat adalah *use case* yang lebih khusus. Skenario normal adalah skenario bila sistem berjalan normal tanpa terjadi kesalahan atau *error*. Sedangkan skenario alternatif adalah skenario bila sistem tidak berjalan normal atau mengalami *error*. Skenario normal dan skenario alternatif dapat lebih dari satu. Alur dari skenario inilah yang nantinya menjadi dasar pembuatan diagram sekuen. (A.S & Shalahuddin, 2014, p. 161)

2.3.1.5. *Activity Diagram*

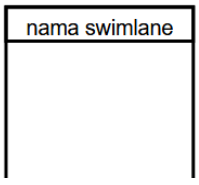
Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. (A.S & Shalahuddin, 2014, p. 161)

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal sebagai berikut (A.S & Shalahuddin, 2014, p. 161):

- Racangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
- Urutan atau pengelompokan tampilan dari sistem atau *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
- Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
- Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut ini adalah simbol-simbol yang terdapat pada diagram aktivitas (A.S & Shalahuddin, 2014, pp. 162–163):

Tabel 2.4 Simbol diagram aktivitas

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* 	Asosisasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan / *join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane  Atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Sumber: (A.S & Shalahuddin, 2014, pp. 162–163)

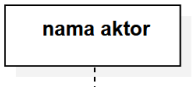
2.3.1.6. *Sequence Diagram*

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh sebab itu, untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*. (A.S & Shalahuddin, 2014, p. 165)

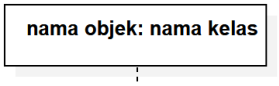
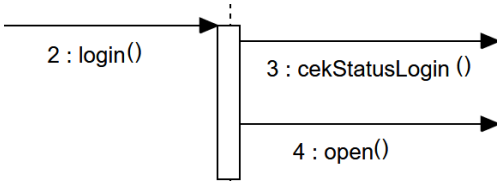
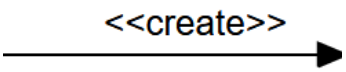
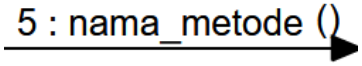
Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak. (A.S & Shalahuddin, 2014, p. 165)

Berikut ini adalah beberapa simbol yang ada pada diagram sekuen (A.S & Shalahuddin, 2014, pp. 165–167):

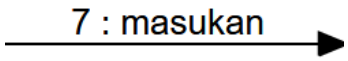
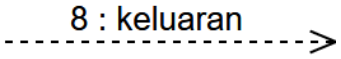
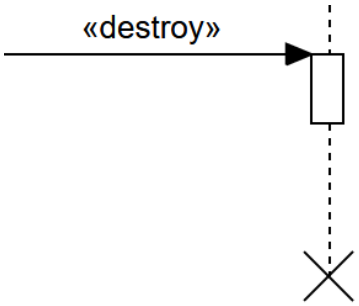
Tabel 2.5 Simbol-simbol diagram sekuen

Simbol	Deskripsi
Aktor  atau  tanpa waktu aktif	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan

Tabel 2.5 Lanjutan

Simbol	Deskripsi
	menggunakan kata benda di awal frase nama aktor.
Garis hidup / *lifeline*	Menyatakan kehidupan suatu objek.
Objek 	Menyatakan objek yang berinteraksi pesan.
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya,  maka cekStatusLogin() dan open() dilakukan di dalam metode login(). Aktor tidak memiliki waktu aktif.
Pesan tipe create 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
Pesan tipe call 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri, arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode

Tabel 2.5 Lanjutan

Simbol	Deskripsi
	yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
Pesan tipe send 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
Pesan tipe return 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy.

Sumber: (A.S & Shalahuddin, 2014, pp. 165–167)

Penomoran pesan berdasarkan urutan interaksi pesan. Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu. Semua metode di dalam kelas harus ada di dalam kolaborasi atau sekuen, jika tidak ada berarti perancangan metode di dalam kelas itu kurang baik. Hal ini dikarenakan ada metode yang tidak dapat dipertanggungjawabkan kegunaannya. (A.S & Shalahuddin, 2014, p. 167)

2.3.2. Bahasa Pemprograman WEB

2.3.2.1. Tentang *World Wide Web*

Situs web (*web site*) awalnya merupakan suatu layanan sajian informasi yang menggunakan konsep hyperlink, yang memudahkan pemakai komputer melakukan penelusuran informasi di Internet, dengan cukup mengklik satu link berupa teks atau gambar, maka informasi dari teks atau gambar akan ditampilkan secara lebih rinci. Web cepat sekali popular di lingkungan pengguna Internet, karena kemudahan yang diberikan kepada pengguna Internet untuk melakukan penelusuran, penjelajahan dan pencarian informasi. (Sidik, 2014, p. 1)

Informasi yang disajikan dalam halaman web menggunakan konsep multimedia, informasi dapat disajikan dengan menggunakan banyak media (teks, gambar, animasi, suara dan atau film). Dalam suatu halaman web, informasi akan dapat disajikan dalam kombinasi media teks, gambar, animasi, suara atau film yang semuanya dapat disajikan dalam satu halaman. (Sidik, 2014, p. 1)

2.3.2.2. HTML

HTML memiliki kependekan dari *Hyper Text Markup Language*. Dokumen HTML adalah *file* teks murni yang dapat dibuat dengan editor teks sembarang. Dokumen ini dikenal sebagai *web page*. (Sidik & Pohan, 2012, p. 9)

Berkas html dibuat dengan perangkat lunak pengolah kata yang disimpan ke dalam format ASCII normal sehingga menjadi sebuah *homepage* dengan perintah-perintah HTML. HTML menggunakan dua macam eksteksi *file* yaitu .htm dan

.html. Format ekstensi berformat .htm awalnya untuk mengakomodasi penggunaan html dalam sistem operasi DOS. (Saputra, 2012, p. 1)

2.3.2.3. CSS

Menurut (Madcoms, 2011, p. 116), CSS atau *Cascading Style Sheets* adalah suatu kumpulan kode-kode memformat, yang mengendalikan tampilan isi dalam suatu halaman web. Penggunaan *style* CSS pada format suatu halaman diletakkan terpisah dari tampilan halaman. Isi dari halaman kode HTML terletak di dalam file HTML, sedangkan kode CSS dapat berupa kode yang berada di dalam *file* lain atau dalam salah satu bagian dari dokumen HTML dan biasanya diletakkan dibagian kepala atau tag <*head*>.

Menurut (Saputra, 2012, p. 27), CSS merupakan bahasa pemprograman web yang didesain khusus untuk mengendalikan dan membangun berbagai komponen dalam web sehingga tampilan web lebih rapih, terstruktur dan seragam. CSS merupakan salah satu pemprograman yang wajib disamping html yang harus dikuasai oleh para setiap pemprogram web, terlebih lagi itu adalah *Web Designer*.

CSS saat ini dikembangkan oleh *World Wide Web Consortium* atau yang biasa lebih dikenal dengan istilah W3C. Sehingga CSS menjadi bahasa standar dalam pembuatan web. CSS bukan menggantikan kode html, tetapi hanya difungsikan sebagai penopang atau pendukung (pelengkap) dari *file* html yang berperan dalam penataan kerangka dan layout. (Saputra, 2012, p. 28)

Kelebihan lain dari penggunaan CSS dibandingkan dengan menggunakan kode HTML saja yaitu lebih hemat waktu dan lebih mudah dalam mengedit

tampilan web. Hal ini disebabkan CSS fungsinya seperti *master* halaman. Jadi ketika ingin mengubah seluruh web hanya cukup mengubah dari *file* CSS-nya tanpa perlu satu per satu dari tiap halaman. Ini juga berefek pada *loading* halaman yang menjadi cepat, karena ukuran *file* tiap web jadi lebih kecil. Sedangkan kekuranganya adalah, beberapa kode CSS yang berjalan baik dari satu web browser, belum tentu bisa aktif pada web browser yang lainnya. (Madcoms, 2011, p. 116)

2.3.2.4. PHP

Gambar 2.7 Logo PHP

PHP merupakan singkatan dari "*Hypertext Preprocessor"*. Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (situs personal) dan pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995 dan pada saat itu PHP masih bernama FI (*Form Interpreter*), yang wujudnya berupa sekumpulan *script* yang digunakan untuk mengelola data *form* dari web. Selanjutnya Rasmus merilis kode sumber tersebut untuk umum. PHP adalah sebuah bahasa *scripting* yang terpasang pada HTML. (Madcoms, 2011, p. 228)

PHP digunakan untuk membuat tampilan web menjadi lebih dinamis. PHP dapat menampilkan atau menjalankan beberapa *file* dalam satu *file* dengan cara di-*include* atau *require*. PHP sudah dapat berinteraksi dengan *database* walaupun dengan kelengkapan yang berbeda. (Madcoms, 2011, p. 228)

Sama seperti bahasa pemprograman lainnya, PHP juga memiliki variabel. Menurut (Saputra, 2012:92), variabel merupakan tempat penyimpanan sementara didalam *memory* komputer. Penulisan variabel didalam pemprograman PHP ada aturannya, yaitu (Saputra, 2012, p. 92):

1. Penulisan variabel harus diawali dengan simbol dolar ($).
2. Karakter pertama setelah simbol dolar tidak boleh menggunakan angka (harus huruf).

   Contoh penggunaan yang salah: $123

   Contoh penggunaan yang benar: $hore
3. Setelah simbol dolar ($) dan huruf, maka karakter selanjutnya boleh menggunakan angka.

   Contoh: $hore123

2.3.3. Atom

Atom digunakan sebagai *text editor* yang memberikan kenyamanan dalam menulis kode program web dan bersifat *open source* atau gratis. Meskipun bersifat gratis, Atom memiliki tampilan yang menarik dibandingkan dengan *software text editor* gratis lainnya. Atom juga memiliki kemampuan yang ada dalam *software text editor* berbayar pada umumnya.

Gambar 2.8 Logo Atom

2.3.4. XAMPP

Menurut (Komputer, 2009, p. 30), XAMPP adalah salah satu paket instalasi Apache, PHP, dan MySQL secara instan yang dapat digunakan untuk membantu proses instalasi ketiga produk tersebut sama seperti PHPTriad. Selain paket instalasi instan, XAMPP juga memberikan fasilitas pilihan pengguna PHP 4 atau PHP 5. Untuk melakukan migrasi ke versi lebih tinggi juga sangat mudah dilakukan dengan bantuan PHP-Switch yang telah disertakan oleh XAMPP. Sama halnya dengan PHP, XAMPP bersifat *free* atau gratis untuk digunakan.

Gambar 2.9 Logo Xampp

Bagi sebagian programmer berpengalaman, menginstall web server Apache tidak mudah dan menyulitkan saat user ingin menambahkan MySQL, PHP dan Perl. Namun demikian, XAMPP dapat menjawab dan mengatasi semua permasalahan tersebut. Dilengkapi dengan Control Panel berbasis GUI, PHPMyAdmin, dan *add-ons* yang mendukung, XAMPP bisa dijadikan sebagai web server dan database server serta pendukung PHP. (Komputer, 2009, p. 30)

Menurut (Madcoms, 2011, p. 229), web server merupakan suatu program komputer yang mempunyai tanggung jawab atau tugas menerima permintaan HTTP dari komputer klien, yang dikenal dengan nama web browser dan melayani mereka dengan merespon HTTP berupa konten data, biasanya berupa halaman web yang terdiri dari dokumen HTML dan objek yang terkait seperti gambar dan lain-lain.

2.3.5. MySQL

Menurut (Saputra, 2012, p. 77), MySQL merupakan salah satu *database* kelas dunia yang sangat cocok bila dipadukan dengan bahasa pemprograman PHP. MySQL bekerja menggunakan bahasa SQL (*Structure Query Language*) yang merupakan bahasa standar yang digunakan untuk manipulasi *database*.

Gambar 2.10 Logo MySQL

Penyimpanan data yang fleksibel dan cepat aksesnya sangat dibutuhkan dalam sebuah *website* yang interaktif dan dinamis. Database sendiri berfungsi untuk menampung data yang diinputkan melalui *form website*. Selain itu dapat juga dibalik dengan menampilkan data yang tersimpan dalam *database* ke dalam halaman website. MySQL bersifat gratis, selain itu MySQL dapat berjalan diberbagai *platform* antara lain Linux, Windows dan sebagainya. (Madcoms, 2011, p. 260)

Perintah yang 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. (Saputra, 2012, p. 77)

2.4. Penelitian Terdahulu

Supaya penelitian ini dapat berjalan dengan baik, maka peneliti mencoba menggali informasi di beberapa penelitian yang sudah ada. Dibawah ini adalah beberapa penelitian terdahulu yang memiliki kaitan dengan penelitian ini, yaitu:

1. **Teri Mangkarisnal dan Muhammad Zaki Rusti** (2016) dengan ISSN 2502-8758, dalam penelitian yang berjudul **Sistem Pakar Diagnosa Penyakit Tanaman Hias Anthurium Menggunakan Metode *Foward Chaining***, peneliti mengatakan bahwa permintaan tanaman hias semakin meningkat. Sehingga para petani berusaha untuk menghasilkan tanaman hias yang berkualitas dan dalam jumlah yang banyak. Tantangan yang dihadapi oleh para petani tanaman hias adalah penyakit yang menyerang pada ditanaman hias. Di antara beberapa tanaman hias yang ada, tanaman hias *anthurium* dijadikan sebagai objek penelitiannya. Dalam penelitian tersebut, peneliti berusaha membuat sistem pakar yang dapat mendiagnosis penyakit yang terdapat pada tanaman hias *anthurium*. Dengan adanya sistem pakar tanaman hias *anthurium*, para petani yang kurang pengetahuannya dengan tanaman hias tersebut bisa terbantu dengan adanya sistem pakar ini.
2. **Dadi Rosadi dan Asril Hamid** (2014) dengan ISSN 2442-4943, dalam penelitian yang berjudul **Sistem Pakar Diagnosa Penyakit Tanaman Padi Menggunakan Metode Forward Chaining**, peneliti menggungkapkan bahwa tanaman padi sering terjadi gagal panen yang disebabkan oleh terserangnya berbagai macam penyakit. Tidak hanya itu saja, peneliti menyebutkan bahwa

terbatasnya pengetahuan para petani tentang penyakit padi dan kurangnya seorang ahli dibidang tersebut yang dapat terjun langsung ke para petani. Dalam sistem pakar tersebut, metode inferensi yang digunakan adalah *forward chaining.* Dengan menggunakan sistem pakar diagnosis penyakit tanaman padi dengan menggunakan metode *forward chaining* dapat membantu para petani dalam menemukan penyakit yang terdapat pada tanaman padinya berdasarkan gejala-gejala yang dipilih. Sehingga para petani tidak perlu lagi menunggu seorang ahli tanaman padi untuk memberikan solusi terhadap penyakit yang ada pada tanamannya.

3. **Yusuf Hidayat dan Dini Destiani** (2015) dengan ISSN 2302-7339, dalam penelitian **Pengembangan Sistem Pakar Diagnosis Penyakit Jeruk Keprok Garut**, peneliti menyebutkan bahwa para petani jeruk sering kali mengalami kesulitan akibat serangan penyakit *Citrus Phloem Vein Degeneration* (CPVD) yang dapat menghancurkan tanaman jeruk keprok Garut. Para petani jeruk keprok Garut juga mengalami kesulitan untuk berkonsultasi dengan seorang ahli, karena rata-rata seorang ahli memiliki jam kerja yang terbatas dan kunjungan kerja yang banyak. Dengan keterbatasan tersebut, peneliti berusaha membuat sebuah sistem pakar diagnosis penyakit tanaman jeruk keprok Garut berbasis web. Sehingga para petani jeruk keprok Garut dapat melakukan konsultasi dengan sistem pakar seperti bertemu langsung dengan pakarnya.
4. **Resi Resmiati dan Asep Deddy Supriatna** (2016) dengan ISSN 2302-7339, dalam penelitian yang berjudul **Pengembangan Sistem Pakar Diagnosis Penyakit Cabai Paprika Berbasis Android**, peneliti menyampaikan bahwa

tanaman cabai paprika salah satu tanaman holtikultural yang rentan terserang berbagai penyakit. Kurangnya pemahaman petani dalam melakukan pemeliharaan tanaman cabai paprika menyebabkan tanaman mudah terserang penyakit. Sehingga perlu adanya penyuluhan kepada para petani dengan memberikan informasi secara teori mengenai cara berbudidaya tanaman cabai paprika yang baik. Peneliti dalam penelitian tersebut berpendapat, sistem pakar dapat membantu penyampaian informasi dalam proses penyuluhan kepada petani mengenai pencegahan dan penangulangan penyakit cabai paprika.

5. **Fernandya Riski Hartantri dan Ardi Pujiyanta** (2014) dengan ISSN 2338-5197, dalam penelitian yang berjudul **Deteksi Penyakit Dan Serangan Hama Tanaman Buah Salak Menggunakan Jaringan Syaraf Tiruan (JST) dengan Metode Perceptron**, peneliti mengatakan bahwa buah salak merupakan buah yang dihasilkan oleh tanaman yang hanya terdapat di Indonesia. Peneliti dalam penelitian tersebut melakukan pengumpulan dengan menggunakan metode observasi, studi kasus dan wawancara, sehingga ditemukan 30 gejala dan 12 macam jenis penyakit tanaman salak. Penelitian tersebut dilakukan untuk membuat aplikasi jaringan syaraf tiruan yang dapat mendeteksi penyakit dan serangan hama tanaman salak serta memberikan pengobatan atau pencengahannya.
6. **Joko Triono dan Tomi Tristono** (2016) dengan ISSN 0976-5697, dalam penelitian yang berjudul ***Expert System Identification of Pest and Diseases of Rice using Html5***. Penelitian tersebut terjadi karena tanaman padi yang rentan terhadap penyakit dan hama. Peneliti melakukan pembuatan sistem pakar yang

digunakan untuk mengidentifikasi hama dan penyakit pada tanaman padi. Sistem pakar tersebut menggunakan metode penalaran berbasis aturan dan mesin inferensi *forward chaining*. Sistem pakar untuk mengindetifikasi penyakit dan hama tanaman padi tersebut dibangun menggunakan bahasa pemprograman PHP, basis data MySQL dan HTML5. Sehingga sistem pakar tersebut dapat diakses pada berbagai perangkat komputer maupun perangkat *mobile*.

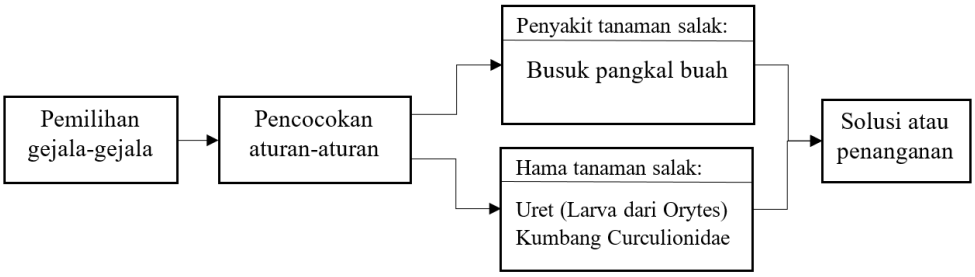
7. **Ch.Viswanadha Sarma** (2012) dengan ISSN 2278-0181, dalam penelitian yang berjudul ***Rule Based Expert System for Rose Plant***. Peneliti membuat aplikasi sistem pakar untuk menemukan penyakit tanaman mawar dengan menggunakan fakta dan aturan yang sesuai dengan basis pengetahuan. Aplikasi sistem pakar tersebut menggunakan metode *forward chaining* atau runut maju. Dengan menggunakan aplikasi sistem pakar tersebut, pengguna dapat menemukan penyakit tanaman mawar dan tidak perlu lagi bertemu dengan ahli penyakit tanaman mawar.

2.5. Kerangka Pemikiran

Kerangka berfikir merupakan model konseptual tentang bagaimana teori berhubungan dengan berbagai faktor yang telah diidentifikasi sebagai masalah yang penting (Sugiyono, 2014, p. 60). Kerangka pemikiran dibutuhkan supaya penelitian ini menjadi lebih terarah dalam menyelesaikan masalah yang ada.

Dari masalah yang sudah disampaikan pada penjelasan sebelumnya, peneliti membuat kerangka pemikiran yang bertujuan untuk memberikan gambaran seperti

apa proses kerja dari sistem pakar yang akan dibuat oleh peneliti. Hal ini sangatlah penting, karena nantinya sistem pakar tidak hanya mendiagnosis penyakit dan hama tanaman salak saja, tetapi juga akan memberikan solusi yang terbaik. Berikut ini adalah rancangan dari kerangka pemikiran dalam penelitian ini.



Gambar 2.11 Kerangka Pemikiran
Sumber: Data Penelitian (2017)

Dari gambar kerangka pemikiran tersebut, peneliti akan menjelaskan secara singkat tahapan apa saja yang dibutuhkan sistem untuk menemukan solusi:

1. Pada tahap pemilihan gejala-gejala, pengguna akan dihadapkan pada gejala-gejala apa saja yang terdapat pada tanaman salak. Kemudian, pengguna akan memilih gejala-gejala yang sesuai dengan apa yang ia amati pada tanaman salaknya.
2. Pada proses pencocokan aturan-aturan, sistem akan menerima gejala-gejala yang dipilih oleh pengguna. Selanjutnya gejala-gejala yang dipilih oleh pengguna akan disamakan dengan aturan-aturan yang ada.
3. Pengguna akan mendapat hasil diagnosis berupa penyakit, jika gejala yang dipilih sesuai dengan aturan-aturan yang berkaitan penyakit. Sedangkan pengguna akan mendapatkan hasil diagnosis berupa hama, jika gejala yang

dipilih sesuai dengan aturan-aturan yang berkaitan dengan hama tanaman salak.

4. Setelah penyakit dan hama ditemukan, sistem akan memberikan saran bagaimana pengendalian atau penanganan yang tepat.