BAB II

LANDASAN TEORI

2.1 Sistem Pakar

Sistem pakar (*expert system*) adalah salah satu bagian dari kecerdasan buatan yang menggabungkan dasar-dasar pengetahuan dan mesin inferensial agar dapat mengadopsi kemampuan ahli menjadi alat untuk memecahkan masalah seperti apa yang dilakukan ahli. Sistem pakar merupakan cabang dari *Artificial Intelligence* yang secara khusus menggunakan pengetahuan untuk memecahkan masalah di tingkat ahli. Sistem pakar adalah salah satu teknik penalaran terkenal yang digunakan dalam domain aplikasi diagnosis. Pada sistem pakar, pengetahuan manusia tentang keahlian tertentu untuk menyelesaikan tugas tertentu direpresentasikan sebagai fakta dan aturan dalam basis pengetahuan. Sistem pakar adalah perangkat lunak yang mensimulasikan kinerja ahli dalam bidang tertentu. Sistem pakar saat ini telah digunakan di banyak daerah di mana memerlukan pengambilan keputusan atau memprediksi dengan keahlian. Sistem pakar adalah sistem kecerdasan buatan yang menggabungkan basis pengetahuan dengan mesin inferensi sehingga dapat mengadopsi kemampuan ahli ke dalam komputer, sehingga komputer dapat memecahkan masalah seperti yang dilakukan oleh ahli-ahli. Dari beberapa definisi tersebut, maka dapat diambil kesimpulan umum yaitu sistem pakar merupakan sistem yang mengadopsi kemampuan para pakar, sehingga sistem dapat memecahkan permasalahan pada domain tertentu seperti yang biasanya dilakukan oleh para pakar.

2.1.1 Konsep Dasar Sistem Pakar

Menurut Efrain Turban dalam (Suyoto, 2004), konsep dasar sistem pakar mengandung keahlian, ahli, pengalihan keahlian, inferensi, aturan dan kemampuan untuk menjelaskan. Keahlian adalah suatu kelebihan penguasaan pengetahuan di bidang tertentu yang diperoleh dari pelatihan, membaca dan pengalaman. Seorang ahli atau pakar adalah seseorang yang mampu menjelaskan suatu tanggapan.

Mempelajari hal-hal yang baru seputar topik permasalahan (domain), menyusun kembali pengetahuan jika dipandang perlu, memecah aturan-aturan jika dibutuhkan, dan menentukan relevan tidaknya keahlian mereka. Pengalihan keahlian yang dimaksud adalah pengalihan keahlian dari para ahli ke komputer untuk kemudian dialihkan lagi ke orang lain yang bukan ahli, merupakan tujuan utama dari sistem pakar. Proses ini membutuhkan empat aktivitas yaitu : tambahan pengetahuan (dari ahli atau sumber-sumber lainnya) → representasi pengetahuan ke komputer → inferensi pengetahuan → pengalihan pengetahuan ke user. Pengetahuan yang disimpan di komputer disebut dengan nama basis pengetahuan. Ada dua tipe pengetahuan, yaitu: fakta dan prosedur (biasanya berupa aturan). Salah satu fitur yang harus dimiliki oleh sistem pakar adalah kemampuan untuk menalar. Jika keahlian- keahlian sudah tersimpan sebagai basis pengetahuan dan sudah tersedia program yang mampu mengakses basis data, maka komputer harus dapat diprogram untuk membuat inferensi. Proses inferensi ini dikemas dalam bentuk motor inferensi (*inference engine*). Jadi secara umum sistem pakar terdiri atas tiga komponen utama yaitu : *Knowledge base* (Basis Pegetahuan), *Motor Inferensi, User Interface.*

2.1.2 Bentuk Sistem Pakar

Ada empat bentuk sistem pakar, yaitu :

a. Berdiri sendiri.

Sistem pakar jenis ini merupakan *software* yang berdiri sendiri tidak tergabung dengan *software* yang lainnya.

b. Tergabung.

Sistem pakar jenis ini merupakan bagian program yang terkandung di dalam suatu *algoritma* atau merupakan program dimana di dalamnya memanggil *algoritma sub* rutin lain.

c. Menghubungkan ke software lain.

Bentuk ini biasanya merupakan sistem pakar yang menghubungkan ke suatu paket program tertentu, misalnya dengan DBMS.

d. Sistem mengabdi.

Sistem pakar ini merupakan bagian dari komputer khusus yang dihubungkan dengan suatu fungsi tertentu. Misalnya sistem pakar yang digunakan untuk membantu menganalisis data radar.

2.1.3 Ciri-ciri Sistem Pakar

Sistem pakar yang baik harus memenuhi ciri-ciri sebagai berikut :

a. Memiliki fasilitas informasi yang handal.

b. Mudah dimodifikasi.

c. Dapat digunakan dalam berbagai jenis komputer.

2.1.4 Arsitektur Sistem Pakar

Dalam Ramadhan, Fatimah & Pane, (2018) Sistem pakar disusun oleh 6 bagian utama, yaitu : Basis Pengetahuan *(Knowledge Base)*, Basis Data *(Database Spreadsheet)*, Antarmuka Pengguna *(User Interface)*, Fasilitas Penjelasan *(Explanation Subsystem)* dan Pengguna *(User)*. Berikut ini penjelasan tentang komponen-komponen arsitektur Sistem Pakar :

2.1.4.1.Basis Pengetahuan *(Knowledge Base)*

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

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

2.1.4.2. Basis Data *(Database Spreadsheet)*

Digunakan sebagai media yang berfungsi untuk menampung fakta-fakta, kondisi yang diperoleh dari basis pengetahuan untuk disimpan dan diproses oleh komputer.

2.1.4.3. Mesin Inferensi *(Inference Engine)*

Mesin Inferenesi adalah sebuah program yang berfungsi untuk memandu proses penalaran terhadap suatu kondisi berdasarkan pada basis pengetahuan yang ada, memanipulasi dan mengarahkan kaidah, model dan fakta yang disimpan dalam basis pengetahuan untuk mencapai solusi atau kesimpulan. Dalam prosesnya, mesin

inferensi menggunakan strategi pengendalian yang berfungsi sebagai panduan arah dalam melakukan proses penalaran.

2.1.4.4. Antarmuka Pengguna *(User Interface)*

Digunakan sebagai media komunikasi antara pengguna dan sistem pakar. Komunikasi ini paling bagus jika dalam bahasa alami dan dilengkapi dengan *graphic, menu*, dan formulir elektronik. Pada bagian ini akan terjadi dialog antara sistem pakar dan pengguna.

2.1.4.5. Fasilitas Penjelasan *(Explanation Susbsystem)*

Berfungsi memberi penjelasan kepada pengguna, bagaiamana suatu kesimpulan dapat diambil. Kemampuan seperti ini sangat penting bagi pengguna untuk mengetahui proses pemindahan keahlian pakar dan pemechan masalah.

2.1.4.6. Pengguna *(User)*

Pada umumnya pengguna sistem pakar bukanlah seorang pakar *(Non-expert)* yang membutuhkan solusi atau saran dari berbagi permaslahan yang ada.

2.2 Metode Inferensi : *Forward Chaining*

Nurajizah & Saputra (2018) menyatakan inferensi adalah suatu prosedur (program) yang mempunyai kemampuan dalam melakukan penalaran. Inferensi ditampilkan pada suatu komponen yang disebut mesin inferensi yang mencakup prosedur-prosedur mengenai pemecahan masalah. Semua pengetahuan yang dimiliki oleh seorang pakar disimpan pada basis pengetahuan oleh sistem pakar. Tugas mesin inferensi adalah mengambil kesimpulan berdasarkan basis pengetahuan yang dimilikinya.

Terdapat berbagai cara pemecahan masalah didalam sistem pakar. Beberapa hal yang perlu diperhatikan adalah arah penelusuran dan topologi penelusuran. Dalam hal ini, pemecahan masalah yang ada pada sistem menggunakan *forward chaining*. *Forward chaining* adalah teknik pencarian yang dimulai dari inputan beberapa fakta, kemudian menurunkan beberapa fakta dari aturan-aturan yang cocok pada *knowledge base* dan melanjutkan prosesnya sampai jawaban sesuai.

Forward Chaining adalah teknik pencarian yang dimulai dengan fakta yang diketahui, kemudian mencocokkan fakta-fakta tersebut dengan bagian IF dari rule IF-THEN. Bila ada fakta yang cocok dengan bagian IF, maka rule tersebut dieksekusi. Bila sebuah rule dieksekusi, maka sebuah fakta baru (bagian THEN) ditambahkan ke dalam database. Setiap kali pencocokan, dimulai dari rule teratas. Setiap rule hanya boleh dieksekusi sekali saja. Proses pencocokan berhenti bila tidak ada lagi rule yang bias dieksekusi (Puji & Usti 2018).

Untuk mempermudah pemahaman mengenai metode ini, akan diberikan ilustrasi kasus pembuatan sistem pakar dengan daftar aturannya sebagai berikut:

R1: Jika Premis 1 Dan Premis 2 Dan Premis 3 Maka Konklusi 1

R2: Jika Premis 1 Dan Premis 3 Dan Premis 4 Maka Konklusi 2

R3: Jika Premis 2 Dan Premis 3 Dan Premis 5 Maka Konklusi 3

R4: Jika Premis 1 Dan Premis 4 Dan Premis 5 Dan Premis 6 Maka Konklusi 4

Penelusuran maju pada kasus ini adalah untuk mengetahui apakah suatu fakta yang dialami oleh pengguna itu termasuk konklusi 1, konklusi 2, konklusi 3, atau konklusi 4 atau bahkan bukan salah satu dari konklusi tersebut, yang artinya sistem belum mampu mengambil kesimpulan karena terbatas aturan. Seandainya user memilih premis 1, premis 2, dan premis 3, maka aturan yang terpilih adalah aturan R1 dengan konklusinya adalah konklusi 1. Seandainya user memilih premis 1 dan premis 6, maka sistem akan mengarah pada aturan R4 dengan konklusinya adalah konklusi 4, tetapi karena aturan tersebut premisnya adalah premis 1, premis 4, premis 5, dan premis 6, maka premis-premis yang dipilih oleh user tidak cukup untuk mengambil kesimpulan kzonklusi 4 sebagai konklusi terpilih.

2.3 Televisi LED

Yasin et al., (2016) Televisi LCD *(Liquid Crystal Display)* merupakan sebuah teknologi layar digital yang menghasilkan citra pada sebuah permukaan yang rata *(flat)* dengan sistem pencahayaan menggunakan lampu CCFL *(Cold Cathode Fluorescent Lamps).* Sedangkan televisi LED *(Light Emitting Dioda)* sistem pencahayaannya yang menggunakan teknologi LED *backlight.* Secara kualitas dan teknologi, televisi LED juga sudah jauh lebih berkembang. Belakangan ini banyak masyarakat indonesia yang lebih memilih televisi LED dari pada televisi CRT, hal ini dikarenakan televisi banyak memiliki kelebihan dari pada televisi CRT selah satunya adalah tidak membutuhkan daya listrik yang terlalu besar.

2.4 Android

Arfida, Amnah & Wibowo (2018) menjelaskan bahwa Android merupakan sebuah sistem operasi yang berbasis linux untuk perangkat portable seperti *semartphone* dan komputer tablet. Sistem operasi dapat diilustrasikan sebagai jembatan antara piranti *(device)* dan penggunanya, sehingga pengguna bisa berinteraksi dengan device-nya dan menjalankan aplikasi – aplikasi yang tersedia pada *device*. Android juga menyediakan *platform* terbuka *(open source)* bagi para pengembang untuk mengembangkan aplikasi pada berbagai perangkat dengan sistem operasi android.

Firly (2018) menjelaskan bahwa android pertama kali dirilis pada bulan oktober 2003 oleh Andy Rubin, Rich Miner, Nich Sears dan Chris White di bawah sebuah perusahaan bernama android Inc di Palo Antom, California. Dan akhirnya android diakuisisi oleh google pada tahun 2005. 5 november 2007 adalah kali pertama android meluncurkan versi beta yang bersamaan dengan berdirinya open handset alliance atau OHA. Android sendiri memiliki beberapa kelebihan yaitu adalah sebagai berikut :

a. *Open Source* alias Gratis
b. Cepat dan Responsive
c. *User Friendly*
d. Variasi harga produk yang beragam

e. Google sebagai pengembang

f. *Hardware* pendukung yang beragam

2.5 Perangkat Lunak Pengembangan Sistem

Pengembangan sistem untuk membangun aplikasi sistem pakar deteksi kerusakan televisi led berbasis android diperlukan beberapa perangkat lunak yang digunakan dalam membangun aplikasi tersebut. Beberapa perangkat lunak yang digunakan adalah sebagai berikut:

2.5.1. Android studio

Firly (2018) menjelaskan bahwa Android studio merupakan *integrated development environment* (IDE) atau dalam artian lain adalah sebuah lingkungan pengembangan terintegrasi resmi yang memang merancang khusus untuk pengembangan sistem operasi Google Android. Aplikasi ini dibangun di atas sebuah perangkat lunak yang dinamakan Intellij IDEA milik JetBrains. Bisa juga dibilang bahwa android studio merupakan pengganti dari *Eclipse android development tool* atau ADT sebagai IDE utama dalam pengembangan aplikasi android yang asli.

Android studio diluncurkan pada tanggal 16 mei 2013 dalam konferensi google I/O yang pada saat itu masih dalam tahap pratinjau akses versi 0.1 sebagai perintis. Hingga pada akhirnya versi stabil 3.0 yang liris pada pertengahan bulan oktober 2017 dan menjadi software terlaris dikalangan *developer* muda. Aplikasi ini dapat digunakan diberbagai sistem operasi yaitu *windows, linux dan macOS.*

Aplikasi ini menawarkan berbagai fitur canggih yang akan meningkatkan kemampuan produktivitas dalam proses pengembangan aplikasi. Berikut ini adalah beberapa hal yang akhirnya banyak mengundang *developer* untuk melirik android studio sebagai software pengembang :

a. Dukungan dari C++, NDK dan sekarang kotlin

b. Perkembangan yang *up to date*

c. Sistem berbasis *Gradle* yang dinilai *fleksibel*

d. Lingkungan yang mencakup seluruh perangkat android
e. Emulator yang cepat dan kaya akan fitur
f. Alat pengujian dan kerangka yang juga ekstensif
g. *Instant Run*
h. Dukungan *google cloud platform*

2.5.2. Java

Firly (2018) menjelaskan bahwa *Java* adalah bahasa pemrograman *multi platform. Java* tidak menyediakan IDE khusus seperti halnya bahasa pemrograman yang lain. Pemrogram bisa menggunakan IDE yang *support* ke *java*, misalnya *Netbeans*, *Eclips*, *TexPad*, dan lain-lain. Elemen-elemen dasar pemrograman *Java* terdiri dari Himpunan karakter, Pengenal (identifier), Kata Kunci, Tipe Data Primitif. Tipe data primitif yang didukung oleh bahasa pemrograman *Java* adalah *byte, short, int, long, float, double, Boolean, char.*

2.5.3. Adobe XD

Adobe XD adalah perangkat lunak yang bisa digunakan oleh para desainer aplikasi mobile. Adobe xd bisa memudahkan desainer aplikasi mobile dalam pengembangan UX/UI, adobe xd ini sudah menyediakan fitur UI Desain dan juga UX Desain sebagai prototype tanpa membutuhkan third-party atau aplikasi lain untuk membantu membuat sebuah *prototype*. (garudapixel.com, dewaweb.com).

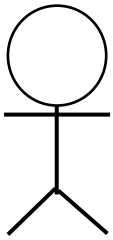
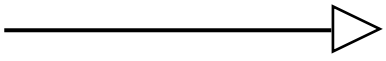
2.6 *Unifed Modelling language* (UML)

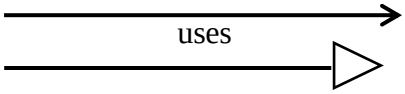
Rosa (2018) menjelaskan bahwa *Unifed Modeling Language* (UML) adalah metode perancangan berorientasi objek yang memeriksa syarat – syarat dari sudut pandang kelas-kelas dan objek yang ditemui pada ruang lingkup permasalahan dengan tujuan untuk memahami domain masalah dan meningkatkan ketelitian, konsistensi, kelengkapan analisis. *Unified Modeling Language* (UML) adalah satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek.

2.6.1 *Use Case Diagram*

Rosa & Shalahuddin (2018) menguraikan bahwa *Use Case Diagram* merupakan deskripsi peringkat tinggi bagaimana perangkat lunak (aplikasi) akan digunakan oleh penggunanya. Selanjutnya *use case* tidak hanya penting pada tahap analisis, tetapi juga sangat penting untuk perancangan, untuk mencari kelas-kelas yang terlibat dalam aplikasi, serta untuk melakukan pengujian. Diagram *use case* bersifat statis, diagram ini memperlihatkan himpunan *use-case* dan aktor-aktor (suatu jenis khusus dari kelas). Simbol – simbol *Use Case* dapat dilihat pada tabel 2.1

Tabel 2.1 Simbol *Use Case Diagram*

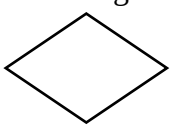
Simbol	Keterangan
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit dan aktor.
Aktor / *Actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi.
Asosiasi / *Association* 	Komunikasi antar aktor dan *Use Case* yang berpartisipasi.
Ekstensi / *extend* <<extend>> 	Relasi Use Case tambahan ke sebuah Use Case dimana Use Case yang ditambah dapat berdiri sendiri walau tanpa *Use Case* tambahan.
Generalisasi / generalization 	Hubungan generalisasi dan spesialisasi antara dua buah *Use Case* yang mana fungsi yang satu lebih umum dari yang lainnya.

Include / Use Case <<include>>  uses	Relasi *Use Case* tambahan ke sebuah *Use Case* dimana *Use Case* yang ditambahkan memerlukan *Use Case* ini untuk menjalankan fungsinya.

2.6.2 Activity Diagram

Rosa & Shalahuddin (2018) menguraikan bahwa *Activity diagram* menggambarkan *workflow* (alir kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Tahap perancangan *activity diagram* menjabarkan masing – masing *activity* pada perancangan *use case.* Simbol – simbol *Activity Diagram* dapat dilihat pada tabel 2.2.

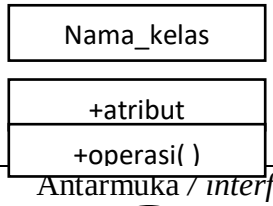
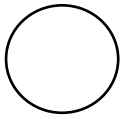
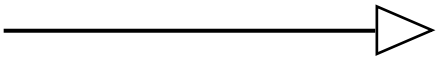
Tabel 2.2. Simbol *Activity Diagram*

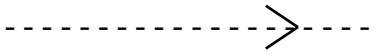
Simbol	Keterangan
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Status awal 	Bagaimana objek dibentuk atau diawali.
Status akhir 	Bagaimana objek dibentuk dan diakhiri.
Percabangan / join 	Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu.
Penggabungan / join 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu

2.6.3 *Class Diagram*

Gunawan & Sari (2017) menguraikan bahwa *Class diagram* menggambarkan atribut / poperti suatu system, sekaligus menawarkan layanan untuk memanipulasi keadaan metode atau fungsi. *Class diagram* menggambarkan struktur dan deskripsi *class, package,* dan *objek* beserta hubungan satu sama lain. Simbol – simbol yang ada pada *Class Diagram* ditunjukan oleh Tabel 2.3.

Tabel 2.3. Simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur system.
Antarmuka / *interface*  **Nama_*interface***	Sama dengan konsep *interface* dalam pemograman berorientasi objek.
Asosisasi / *association*	Relasi antarkelas dengan makna umum, asosia biasanya disertai dengan *multiplicity*.
Asosisasi 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* 	Relasi antarkelas dengan makna kebergantungan antarkelas.
Agregasi / *aggregation* 	Relasi antar kelas dengan makna semua –bagian

2.7 Penelitian Terdahulu

Berikut merupakan jurnal terkait dengan penelitian terdahulu :

No	Nama	Judul	Terbit / Tahun	Keterangan
1	Bagus Fery Yanto, Indah Werdiningsih, Endah Purwanti	Aplikasi Sistem Pakar Deteksi Penyakit Pada anak Bawah lima Tahun Menggunkan Metode Forward Chaining	Jurnal of Information Sistem engineering and Bussines Intelligence Vol.3, No.1 / 2017	Penelitian ini bertujuan untuk membuat aplikasi sistem pakar diagnose penyakit pada balita berbasis mobile.
2	Nency Extise Putri	Sistem pakar Kerusakan hardware komputer dengan Metode Forward Chaining	Jurnal Momentum Vol.18 No.2/ 2016	Penelitian ini bertujuan untuk emmbantu para pengguna komputer dalam mengidentifikasi kerusakan hardware pada komputer
3	Putra Adi Bima, Syamsul Bakhri	Sistem Pakar Diagnosis Kerusakan Mesin Sepeda Motor Non Injeksi Yamaha Pada	Paradigma Vol.20 No.1 / 2018	Penelitian ini bermaksud merancang aplikasi sistem pakar dapat mendiagnosis kerusakan mesin sepeda motor non injeksi untuk membantu, khususnya untuk pemilik kendaraan yang masih

		Bengkel Dirgantara Motor		awam tentang jenis kerusakan mesin sepeda motor serta waktu yang padat dan keberadaan bengkel resmi Yamaha yang masih jarang untuk di daerah- daerah terpencil.