BAB II
LANDASAN TEORI

2.1. Kecerdasan Buatan

Menurut [1] menguraikan bahwa kecerdasan buatan berasal dari bahasa inggris “Artifiacial intellegence” atau di singkat AI ,yaitu intellegence adalah kata sifat yang berarti cerdas, sedangkan artificial artinya buatan. Kecerdasan yang dimaksud di sini merujuk pada mesin yang mampu berpikir, menimbang tindakan yang akan diambil, dan mampu mengambil keputusan seperti yang di lakukan manusia. Kecerdasan diciptakan dan dimasukan ke dalam suatu mesin (komputer) agar dapat dilakukan manusia, beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, game playing, logika fuzzy, jaringan saraf tiruan (JST) dan robotika.

2.2. Sistem Pakar

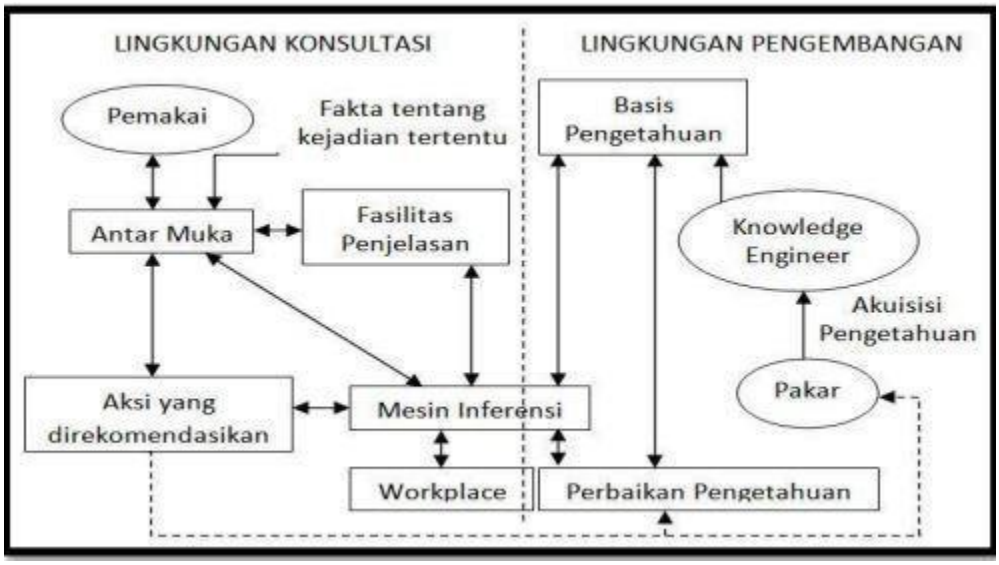
Sistem pakar (*expert system*) secara umum adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli. Atau dengan kata lain sistem pakar adalah sistem yang didesain dan diimplementasikan dengan bantuan bahasa pemrograman tertentu untuk dapat menyelesaikan masalah seperti yang dilakukan oleh para ahli. Diharapkan dengan sistem ini, orang awam dapat menyelesaikan masalah tertentu baik sedikit‟ rumit ataupun rumit sekalipun „tanpa‟ bantuan para ahli dalam bidang tersebut. Sedangkan bagi para ahli, sistem ini dapat digunakan sebagai asisten yang berpengalaman. Sistem pakar merupakan cabang dari Artificial Intelligence (AI) yang cukup tua karena sistem ini telah mulai dikembangkan pada pertengahan tahun 1960. Sistem pakar yang muncul pertama kali adalah General-purpose problem solver (GPS) yang dikembangkan oleh Newl dan Simon. Sampai saat ini sudah banyak sistem pakar yang dibuat, seperti MYCIN, DENDRAL, XCON & XSEL, SOPHIE, Prospector, FOLIO, DELTA, dan sebagainya[2].

2.2.1. Konsep Dasar Sistem Pakar

Secara abstrak sistem pakar (*expert system*) menggambarkan sistem yang memakai pengetahuan manusia yang direpresentasikan dalam komputer dan akhirnya dipergunakan untuk memecahkan problem yang kebanyakan memakai kepakaran seseorang. Lebih lanjut dinyatakan, bahwa konsep dasar sistem pakar mengandung keahlian, ahli, pengalihan keahlian, inferensi, aturan dan kemampuan menjelaskan

2.2.2. Struktur Sistem Pakar

Sistem pakar terdiri dari dua bagian pokok, yaitu : Lingkungan pengembangan (*development environment*) digunakan sebagai pembangun sistem pakar, baik dari aspek pembangun elemen maupun basis pengetahuan. Lingkungan konsultasi (*consultation environment*) digunakan oleh pengguna untuk berkonsultasi dan mendapatkan kesimpulan dan solusi akhir.



Gambar 2. 1 Struktur dalam Sistem Pakar

Struktur yang terdapat dalam Sistem Pakar adalah sebagai berikut:

1) Antarmuka Pengguna

   Media komunikasi antara pengguna/ pemakai/ user dengan sistem. Pada bagian antar muka ini akan terjadi dialog antara pengguna dengan sistem pakar dengantampilan yang dapat dimengerti oleh pengguna.

2) Basis pengetahuan (*knowledge base*)

   Dalam basis pengetahuan terdapat pengetahuan yang diperlukan dalam system.Ada 2 basis pengetahuan yaitu:

   a. Fakta merupakan kondisi, situasi, atau permasalahan yang ada.
   b. Aturan (Rule) merupakan cara menggunakan pengetahuan untuk memecahkanmasalah.

3) Akuisisi pengetahuan

   Memasukkan pengetahuan ke dalam sistem. Pengetahuan yang di transfer ke dalam sistem bisa berasal dari seseorang yang ahli dalam bidang tertentu ataupun pengetahuan tersebut bisa berasal dari laporan riset, buku, atau informasi yang valid / bisa dipertanggung jawabkan.

4) Mesin inferensi (*Interference Engine*)

   Berfungsi untuk memperoleh solusi atau menarik kesimpulan berdasarkan basispengetahuan.

5) Workplace / Blackboard

   Area Untuk merekam hasil sementara yang akan dijadikan sebagai keputusan sementara.

6) Fasilitas Penjelasan

   Berfungsi untuk memberikan penjelasan kepada pengguna tentang cara kerja sistem dalam mengambil suatu kesimpulan. Penjelasan kepada pengguna dapatberbentuk keterangan setelah pertanyaan diajukan.\

7) Perbaikan pengetahuan

   Seorang pakar memiliki kemampuan untuk memperbaiki pengetahuan dengan cara balajar dan menganalisis. Perbaikan pengetahuan juga diperlukan dalam sistem pakar agar sistem dapat memberikan alasan sukses dan gagalnya dalam mengambil keputusan dan dapat mengevaluasi pengetahuan yang ada dalam sistem dapat digunakan di masa yang akan datang atau ada perubahan pengetahuan.

8) Pengguna/ Pemakai

   Pengguna adalah seseorang yang menggunakan sistem pakar tersebut untuk mendapatkan pengetahuan, solusi, atau saran dari suatu permasalahan yang berhubungan dengan sistem pakar tersebut.

2.3. Metode *Case Based Reasoning*

Case based reasoning (CBR) merupakan salah satu metode yang menggunakan pendekatan kecerdasan buatan (artificial intelegent) yang menitik beratkan pemecahan masalah pada pengetahuan dari kasus-kasus sebelumnya. *Case Based Reasoning* (CBR) adalah suatu pendekatan untuk menyelesaikan suatu permasalahan (*problem solving*) berdasarkan solusi dari permasalahan sebelumnya [3]. Secara umum, metode ini memiliki 4 langkah, yaitu :

1. *Retrieve*

   Mencari kasus-kasus sebelumnya pada case memory yang paling mirip dengan permasalahan kasus baru.
2. *Reuse*

   Menggunakan kembali kasus terdahulu sebagai acuan diagnose yang ada pada case memory.
3. *Revise*

   Pada proses ini informasi mengenai solusi yang diberikan akan dikalkulasi, dievaluasi, dan diperbaiki kembali untuk meminimalisir kesalahan-kesalahan yang terjadi pada permasalahan baru.
4. *Retain*

   Proses ini solusi akan diindeksikan, diintegrasikan, dan mengkstrak solusi yang baru dan selanjutnya disimpan dalam Knowledge base untuk menyelesaikan permasalahan selanjutnya.

2.4. Algoritma K-Nearest Neighbor

K-Nearest neighbor adalah metode yang menghasilkan kesimpulan dan penyeslesaian masalah dengan pendekatan terhadap kasus sebelumnya berdasar tingkat kemiripan gejala yang ada pada kasus baru.

Algoritma *K-Nearest Neighbor* (K-NN) adalah suatu metode yang menggunakan algoritma supervised. K-NN termasuk kelompok *instance- based learning*.Algoritma ini juga merupakan salah satu teknik lazy learning.K-NN dilakukan dengan mencari kelompok k objek dalam data training yang paling dekat (mirip) dengan objek pada data baru atau data testing [4] . Berikut persamaan fungsi dari K-Nearest Neighbor :

Similarity(T,S) $= \frac{\sum_{i=1}^{n} f(Ti,Si) * Wi}{Wi}$

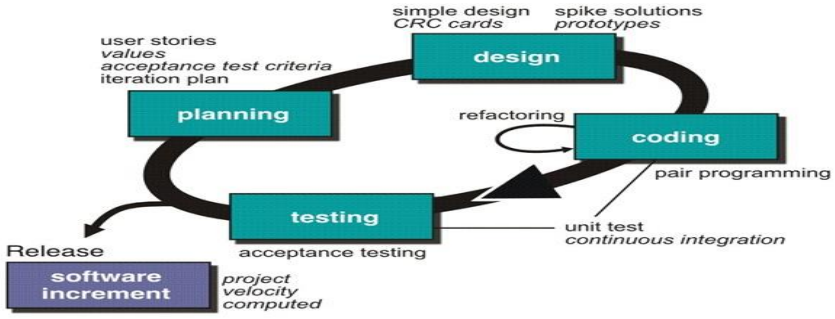
Keterangan : S = similarity (nilai kemiripan)
W = weight (bobot yang diberikan)

2.5. Pengembangan Sistem *Extreme Programming*

XP adalah metodologi pengembangan perangkat lunak yang ditujukan untuk meningkatkan kualitas perangkat lunak dan tanggap terhadap perubahan kebutuhan pelanggan. Jenis pengembangan perangkat lunak semacam ini dimaksudkan untuk meningkatkan produktivitas dan memperkenalkan pos pemeriksaan dimana persyaratan pelanggan baru dapat diadopsi [5]

Tahapan-tahapan dari *Extreme Programming* terdiri dari *planning* seperti memahami kriteria pengguna dan perencanaan pengembangan, *designing* seperti perancangan *prototype* dan tampilan, *coding* termasuk pengintegrasian dan yang terakhir adalah testing. Unsur-unsur lain dari *Extreme Programming* meliputi *paired programming* pada tahapan *coding*, unit testing pada semua kode, penghindaran pemrograman fitur kecuali benar-benar diperlukan, struktur manajemen yang datar, kode yang sederhana dan jelas, dan seringnya terjadi komunikasi antara programmer dan pelanggan ketika terjadi perubahan kebutuhan pelanggan seiring berlalunya waktu berlalu.

Metode ini membawa unsur-unsur yang menguntungkan dari praktek rekayasa perangkat lunak tradisional ke tingkat "ekstrem", sehingga metode ini dinamai *Extreme Programming*. Unsur-unsur yang menjadi karakteristik metodologi adalah kesederhanaan, komunikasi, umpan balik, dan keberanian. Gambar tahapan XP dapat dilihat pada gambar 2.2:



Gambar 2. 2 Tahapan *Extreme Programming*

Dibawah ini adalah penjelasan tahapan *Extreme Programming* yaitu :

1. *Planning* (Perencanaan)

Kegiatan Perencanaan disebut juga *planning game* biasanya dimulai dengan mendengarkan suatu kegiatan yang bertujuan mengumpulkan kebutuhan-kebutuhan untuk memahami konteks bisnis dan perlunya keluaran-keluaran (*output*), fungsi utama dan fungsionalitas.

Pada perencanaan terdapat *user stories values* yaitu story dengan value tertinggi akan dipindahkan dari jadwal dan diimplementasikan pertama, *acceptance test criteria interation plan* melakukan perhitungan kecepatan *project* selama *development*, kustomer dapat menambah story, merubah value, membagi story atau menghapusnya.

2. *Design* (Perancangan)

Perancangan yang menarik, dan sederhana selalu memberikan hasil yang lebih disukai daripada gambaran-gambaran yang lebih kompleks. Perancangan XP memberikan panduan implementasi untuk suatu cerita ketika ditulis, tidak kurang, tidak lebih.

Terdapat *simple design CRC Casrds* untuk mengenali dan mengatur *object oriented class* sesuai dengan *software increment* dan *spike solutions prototypes* melakukan spesifikasi solusi dari *object oriented class.*

3. *Coding* (Pengkodean)

Pengkodean ini dilanjutkan setelah cerita yang telah dikembangkan dan rancangan yang telah dilakukan oleh tim perangkat lunak. Pengkodean ini tidak langsung mengarah ke kode-kode program. Tim akan mengembangkan serangkaian unit pengujian lalu beralih ke pengkodean.

Pada tahapan *pair programming* melakukan kerja sama untuk membuat code dari satu story. Dan *refactoring* adalah proses restrukturisasi kode program computer yang ada tanpa mengubah perilaku eksternalnya.

4. *Testing* (Pengujian)

Unit pengujian yang harus dibuat dan kemudian dijalankan menggunakan kerangka kerja yang memungkinkan mereka untuk diotomatisasi sehingga dapat dijalankan dengan mudah dan berulang kali.

Pada tahapan pengujian yaitu *unit test continous integration* yaitu tahapan

pengujian code yang diintegrasikan dengan kerja lainnya dengan pengujian yang dilakukan oleh customer dan fokus pada keseluruhan dan fungsional sistem dan *acceptance testing* yaitu pengujian yang dilakukan *customer stories* yang akan diimplementasikan sebagai bagian dari *software realease.*

Selanjutnya terdapat tahapan *software increment project velocity computed* yaitu tahapan yang telah diimplementasikan dari *software realease* yang nantinya akan diterapkan dalam suatu sistem.

2.6. *Unified Modeling Language* (UML)

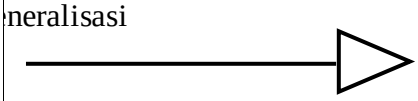
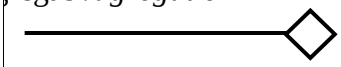
UML (*Unified Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement,* membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek [6]

2.6.1. *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Simbol-simbol yang ada pada diagram kelas pada tabel class diagram 2.1:

Tabel 2. 1 Simbol *Class Diagram*

Simbol	Deskripsi
las **ma_kelas** tribut perasi()	Kelas pada struktur sistem
tarmuka/*Interface* **ma_interface**	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
osiasi/*association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*

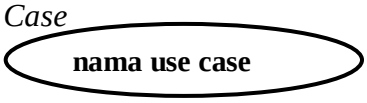
osiasi berarah/*directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya disertai dengan *multiplicity*
neralisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi(umum khusus)
bergantungan/*dependecy* 	Relasi antar kelas dengan makna kebergantungan antar kelas
regasi/*agregation* 	Relasi antar kelas dengan makna semua bagian (*whole-part*)

Sumber: [6]

2.6.2. *Use Case Diagram*

Use case diagram atau diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat [6]. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. simbol-simbol yang ada pada diagram *use case* dapat dilihat pada tabel 2.2:

Tabel 2. 2 Simbol diagram *use case*

Simbol	Deskripsi
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor/*actor* 	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 benda di awal frase nama actor
Asosiasi/*association*	Komunikasi antara aktor dan *use case*

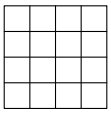
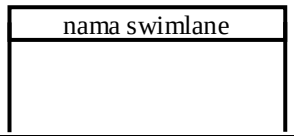
	yang berpatisipasi pada *use case* atau *use case* memiliki interaksi dengan actor
Ekstensi/*extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan
<<extend>> 	dapat berdiri sendiri walau tanpa *use case* tambahan itu mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek biasanya *use case* tambahan memiliki nama depan
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* *<<include>>* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini

2.6.3. *Activity Diagram*

Activity diagram atau Diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. simbol-simbol yang ada pada *activity diagram* dapat dilihat pada tabel 2.3:

Tabel 2. 3 Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu

Penggabungan/*join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Tabel 	Suatu file komputer dari mana data bisa dibaca atau direkam selama kejadian bisnis
Dokumen 	Menunjukkan dokumen sumber atau laporan
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

2.7. Pengertian MySQL

MySQL merupakan *Database* yang menghubungkan *script php* menggunakan perintah *query* dan *escaps character* yang sama dengan *php*. MySQL mempunyai tampilan *client* yang mempermudah anda dalam mengakses *database* dengan kata sandi untuk mengizinkan proses yang bisa anda lakukan [7]

2.8. Web

Website atau situs dapat diartikan sebagai kumpulan halaman yang menampilkan informasi data teks, data gambar diam atau bergerak, data animasi, suara, video atau gabungan dari semuanya, baik yang bersifat statis ataupun dinamis yang membentuk suatu rangkaian bangunan yang saling terkait dimana masing-masing dihubungkan dengan jaringan-jaringan halaman (*hyperlink*). *Website* sendiri merupakan sebuah kumpulan halaman-halaman situs yang tersimpan dalam sebuah *server*/*hosting*, dan teridentifikasi melalui sebuah nama yang disebut juga sebagai domain atau sub domain [7]

2.9. PHP (*Personal Home Page*)

PHP (*Personal Home Page*) adalah bahasa pemrograman yang digunakan secara luas untuk penanganan pembuatan dan pengembangan sebuah web dan bisa digunakan pada HTML. PHP merupakan singkatan dari "PHP: *Hypertext Prepocessor*" dan merupakan bahasa yang disertakan dalam dokumen HTML sekaligus bekerja di sisi *server* (*server-side* HTML-*embedded scripting*). Artinya sintaks dan perintah yang diberikan akan sepenuhnya dijalankan di *server* tetapi disertakan pada halaman HTML biasa, sehingga *script*-nya tak tampak di sisi *client* [7].

2.10. XAMPP

Menurut [8] XAMPP adalah paket program web lengkap yang dapat Anda pakai untuk belajar pemrograman web, khususnya PHP dan MySQL.

Dalam XAMPP banyak diaplikasikan dan digunakan oleh kalangan pengguna komputer di bidang pemrograman web, XAMPP terdiri dari beberapa program yaitu Apache HTTP Server, MYSQL Database, PHP, dan Pearl XAMPP juga dilengkapi fitur manajemen database PHP My Admin seperti pada server hosting sungguhan, sehingga pengembang web dapat mengembangkan aplikasi web berbasis database secara mudah.

Berikut ini Penjelasan dari bagian-bagian XAMPP:

1. X , Kenapa disebut dengan system operasi? Karena XAMPP bisa dijalankan di 4 OS besar yang sering digunakan oleh pengguna komputer saat ini. Dan 4 OS tersebut tidak lain dan tidak bukan adalah Windows, Linux, Mac OS dan Solaris.
2. A (*Apacahe*) merupakan aplikasi web server. Apache ini bersifat *open source* yang berarti gratis dan bisa diedit oleh penggunanya. Tugas utama Apache adalah menghasilkan halaman web yang benar kepada user berdasarkan kode PHP yang dituliskan oleh pembuat halaman web. Jika diperlukan juga berdasarkan kode PHP yang dituliskan, maka dapat saja suatu database diakses terlebih dahulu (misalnya dalam MySQL) untuk mendukung halaman web yang dihasilkan.

3. M (MySQL), merupakan aplikasi database server. Perkembangannya disebut SQL yang merupakan kepanjangan dari *Structured Query Language*. SQL merupakan bahasa terstruktur yang digunakan untuk mengolah database. MySQL dapat digunakan untuk membuat dan mengelola database beserta isinya. Kita dapat memanfaatkan MySQL untuk menambahkan, mengubah, dan menghapus data yang berada dalam database.
4. P (PHP), bahasa pemrograman web. Bahasa pemrograman PHP merupakan bahasa pemrograman untuk membuat web yang bersifat server-side scripting. PHP memungkinkan kita untuk membuat halaman web yang bersifat dinamis. Sistem manajemen basis data yang sering digunakan bersama PHP adalah MySQL. Namun PHP juga mendukung sistem manajement *database Oracle, Microsoft Access, Interbase, d-base, PostgreSQL,* dan lain sebagainya.
5. P (Perl), bahasa pemrograman, pertama kali dikembangkan oleh Larry Wall di mesin Unix. Perl pertama kali dirilis pada tanggal 18 Desember 1987 ditandai dengan keluarnya Perl 1. Dua diantara karakteristik utama perl adalah penanganan teks dan berbagai jalan pintas untuk meyelesaikan persoalan-persoalan umum. Perl sangat populer digunakan dalam program-program CGI *(Common Gateway Interface)* dan protokol internet lainnya.

2.11. *Sublime Text*

Sublime Text adalah aplikasi editor untuk kode dan teks yang dapat berjalan diberbagai platform *operating system* dengan menggunakan teknologi *Phyton API.* Terciptanya aplikasi ini terinspirasi dari aplikasi Vim, Aplikasi ini sangatlah fleksibel dan powerfull. Fungsionalitas dari aplikasi ini dapat dikembangkan dengan menggunakan *sublime-packages*. *Sublime Text* bukanlah aplikasi *open source* dan juga aplikasi yang dapat digunakan dan didapatkan secara gratis, akan tetapi beberapa fitur pengembangan fungsionalitas (*packages*) dari aplikasi ini merupakan hasil dari temuan dan mendapat dukungan penuh dari komunitas serta memiliki lisensi aplikasi gratis.

Sublime Text mendukung berbagai bahasa pemrograman dan mampu menyajikan fitur *syntax highlight* hampir disemua bahasa pemrogramman yang didukung ataupun dikembangkan oleh komunitas seperti; C, C++, C#, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua,

Markdown, MATLAB, OCaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textile and XML. Biasanya bagi bahasa pemrograman yang didukung ataupun belum terdukung secara default dapat lebih dimaksimalkan atau didukung dengan menggunakan add-ons yang bisa didownload sesuai kebutuhan user [7]

2.12. Pengujian *Black Box*

Menurut [9]"Pengujian *black-box* berfokus pada persyaratan fungsional perangkat lunak. Dengan demikian, pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program". Pengujian *black-box* berusaha menemukan kesalahan dalam kategori sebagai berikut :

a. Fungsi – fungsi yang tidak benar atau hilang,
b. Kesalahan interface
c. Kesalahan dalam struktur data atau akses eksternal
d. Kesalahan kinerja
e. Inisialisasi dan kesalahan terminasi

Mengemukakan ciri-ciri *black boxtesting*, diantaranya sebagai berikut [10]:

1. *Black box* testing berfokus pada kebutuhan fungsional pada*software*, berdasarkan pada spesifikasi kebutuhan dari *software*.
2. *Black box* testing bukan teknik alternatif daripada *white boxtesting*. Lebih dari pada itu, ia merupakan pendekatan pelengkap dalam mencakup *error* dengan kelas yang berbeda dari metode*white box testing*.
3. *Black boxtesting* melakukan pengujian tanpa pengetahuan detail struktur *internal* dari *sistem* atau komponen yang dites. Juga disebut sebagai *behavioraltesting*, *specification-based testing*, *input/output testing* atau *functional testing*.

Pada *black boxtesting* terdapat jenis teknik *design* tes yang dapat dipilihberdasarkan pada tipe *testing*, diantaranya sebagai berikut:

1. *Equivalence Class Partitioning*
2. *Boundary Value Analysis*
3. *State Transitions Testing*
4. *Cause-Effect Graphing*

Kategori kesalahan/*error* yang akan diketahui melalui *blackboxtesting*:

1. Fungsi yang hilang atau tak benar/salah
2. *Error* dari antar-muka/*interface*
3. *Error* dari struktur data atau akses *eksternaldatabase*
4. *Error* dari kinerja atau tingkah laku/*perform*
5. *Error* dari inisialisasi dan terminasi

2.13. Tinjauan Pustaka

Beberapa penelitian yang berkaitan dengan penerapan sistem pendukung bantuan oprasional pendidikan dalam menentukan kelayakan berdasarkan jurnal penelitian terlihat pada Tabel 2.4:

Tabel 2. 4 Tinjauan Pustaka

No Literartur	Penulis	Judul	Metode	Hasil
Literatur 01	[2]	Rancang Bangun Aplikasi Sistem Pakar Untuk Menentukan Jenis Gangguan Perkembangan Pada Anak (Study Kasus Taman Kanak-Kanak Islamiyah Sukoharjo)	*Certanty Factor*	Aplikasi yang dikembangkan ini bertujuan untuk menentukan jenis gangguan perkembangan pada anak di bawah umur 10 tahun dengan hanya memperhatikan gejalagejala yang dialami. Dengan menggunakan metode *Certanty Factor* (CF), didapatkan nilai Kemungkinan gangguan yang dialami pasien
Literatur 02	[3]	Penerapan Metode Case Based Reasoning (CBR) Dalam Sistem Pakar Untuk	Metode *Case Based Reasoning*	aplikasi sistem pakar ini, pemilik tanaman hidroponik dapat memperoleh hasil diagnosa dengan cepat dan akurat

No Literartur	Penulis	Judul	Metode	Hasil
		Menentukan Diagnosa Penyakit Pada Tanaman Hidroponik		tanpa harus berkonsultasi dengan pegawai dapur hidup hidroponik dan Dapat membantu keterbatasan pegawai dalam mengetahui penyakit tanaman hidroponik, gejala serta solusi secara pasti tanpa perlu adanya seorang pakar berada di toko
Literatur 03	[1]	Aplikasi *Case Based Reasoning* Untuk Identifikasi Serangan Hama Pada Tanaman Jeruk	*Case Based Reasoning*	Hasil penelitian ini menunjukkan Sistem Case Based Reasoning ini dapat digunakan untutk membantu user mengidentifikasi hama yang menyerang tanaman jeruk.