BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Untuk mendukung keberhasilan penelitihan ini, penyusun melakukan pendekatan teoritis melalui beberapa literatur yang berhubungan dengan penelitihan yang dilakukan. Beberapa uraian penelitihan terdahulu yang menjadi acuan dalam penelitian ini yaitu:

Teguh Ansyor Lorosae, dkk 2018 dengan judul "Sistem Pakar Diagnosis Penyakit Gigi dan Mulut Menggunakan Metode *Dempster-Shafer dan Certainty Factor*" Penelitian ini menghasilkan dengan membandingkan akurasi hasil proses perhitungan dari kedua metode tersebut dengan penilaian keyakinan pakar secara kuantitatif, yaitu dengan menggunakan teori Confusion Matrix yang hasilnya adalah 98,03% pada akurasi metode Dempster-Shafer dan 98,41% pada akurasi metode Certainty Factor.

Muhammad Husni Rifq, dkk, 2019 dengan judul "Perbandingan Metode *Certainty Factor dan Dempster-Shafer* Pada Sistem Pakar *Diagnosa* Penyakit Gigi dan Mulut" penelitian ini menghasilkan memiliki perhitungan tidak pada nilai densitas, melainkan pada subset query dan hanya memiliki nilai kepercayaan dan tidak memiliki nilai ketidak pastian dan dipakai dalam sistem pakar yang mengandung ketidakpastian, dan dalam sekali proses perhitungan hanya dapat

mengolah 2 data saja sehingga kekurangan data dapat terjaga.

Efin Ahanudin dan Joko Sutopo, 2018 dengan judul "Sistem Pakar *Diagnosis* Penyakit Gigi Menggunakan Metode *Certainty Factor*" menghasilkan penelitian diharapkan mampu memberikan kemudahan bagi masyarakat umum khususnya penderita penyakit gigi, untuk lebih mudah mengetahui informasi mengenai tentang penyakit gigi

Nur Anjas Sari, 2019 dengan judul "Sistem Pakar Mendiagnosa Penyakit Demam Berdarah Menggunakan Metode *Certainty Factor*" Penelitian Ini menghasilkan Penerapan metode certainty factor dapat mempermudah dan memberikan perhitungan penyelesaian seberapa pasti para user atau pasien menderita penyakit demam berdarah.

Aryu Hanifah Aji dkk, 2018 dengan judul "Sistem Pakar *Diagnosa* Penyakit Ibu Hamil Menggunakan Metode *Certainty Factor* (CF)" Penelitian ini menghasilkan Sistem pakar ini dapat dijadikan alternatif bagi ibu hamil dalam mengenali tanda bahaya melalui gejala-gejala yang dirasakan, selain dapat memberikan informasi mengenai penyakit, sistem ini akan dapat membantu ibu hamil dalam menunjukkan tempat rujukan yang tepat sehingga dapat ditangani oleh paramedis dengan tepat pula. Setelah dilakukan pengujian fungsionalitas pada sistem pakar diagnosa penyakit ibu hamil ini memiliki tingkat validasi sebesar 100%. Sedangkan pengujian akurasi memiliki tingkat akurasi sebesar 100%.

II.2. Landasan Teoritis

II.2.1. Sistem Pakar

Istilah sistem pakar (*expert system*) berasal dari istilah sistem pakar berbasis pengetahuan. Sistem pakar adalah suatu sistem yang menggunakan pengetahuan manusia yang terekam dalam komputer umtuk memecahkan persoalan yang biasanya memerlukan keahlian manusia. Sistem pakar diterapkan untuk mendukung aktivitas pemecahan masalah.

Sistem pakar merupakan cabang dari kecerdasan buatan (*Artificial Intelligence*) yang cukup tua karena sistem ini mulai dikembangkan pada pertengahan 1960. Sistem ini bekerja untuk mengadopsi pengetahuan manusia ke komputer yang menggabungkan dasar pengetahuan untuk menggantikan seorang pakar dalam menyelasaikan suatu masalah. Sistem pakar berasal dari istilah knowledge base expert system. sistem pakar adalah suatu sistem yang dirancang agar dapat menyelesaikan suatu permasalahan tertentu dengan meniru kerja dari para ahli dalam menjawab pertanyaan dan memecahkan suatu masalah. Dengan sistem pakar ini orang awam pun dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli. Bagi para ahli sistem pakar ini juga membantu aktivitasnya sebagai asisten yang sangat berpengalaman (Neneng,2021)

II.2.2. Metode *Certainty Factor* (CF)

Certainty Factor (CF) merupakan sebuah metode yang diusulkan oleh

Shortliffe dan Buchanan pada 1975 untuk mengakomodasi ketidakpastian pemikiran (inexact reasoning) seorang pakar. Seorang pakar (contoh: dokter) sering menganalisi informasi dengan ungkapan “mungkin“, “kemungkinan besar“, “hampir pasti”. Sehingga dengan adanya metode Certainty Factor ini dapat mengambarkan tingkat keyakinan seorang pakar terhadap masalah yang sedang dihadapi. Saat ini ada dua model yang sering digunakan untuk mendapatkan tingkat keyakinan (CF), yaitu (Sutojo, 2017): Metode ‘Net Belief’ yang diusulkan oleh E.H. Shortliffe dan B. G. Buchanan. Seperti yang ditunjukkan pada persamaan

$$CF(Rule) = MB(H, E) – MD(H, E)$$

Di mana:

CF(Rule): Faktor kepastian

MB(H, E): *Measure of Belief* (ukuran kepercayaan) terhadap hipotesis H jika diberikan evidence E (antara 0 dan 1)

MD(H, E): *Measure of Disbelief* (ukuran ketidakpercayaan)terhadap evidence H, jika diberikan *evidence* E (antara 0 dan 1)

II.2.3. PHP (Hypertext Preprocessor)

Menurut Jubille Enterprise (2017:1) PHP adalah bahasa pemrograman yang digunakan untuk membuat sistem berbasis website. Sebagai sebuah sistem, website tersebut hendaknya memiliki sifat dinamis dan interaktif. Memiliki sifat dinamis, artinya website tersebut bisa berubah tampilan kontennya sesuai kondisi tertentu (misalnya, menampilkan produk yang berbeda-beda untuk setiap pengunjung).

Artinya interaktif website tersebut dapat memberi feedback bagi user (misalnya, menampilkan hasil pencarian produk). PHP merupakan bahasa pemrograman berjenis service-side. Dengan demikian, PHP akan diproses oleh server yang hasil olahannya akan dikirim kembali ke browser. Oleh karena itu, salah satu tool yang harus tersedia sebelum memulai pemrograman PHP adalah server.

II.2.4. Sublime Text

Sublime text merupakan perangkat lunak text editor yang digunakan untuk membuat atau meng-edit suatu aplikasi. Sublime text mempunyai fitur plugin tambahan yang memudahkan programmer (Supono dan Putratama, 2016:14).

II.2.5. Database

Database adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukkan dengan kunci dari tiap tabel yang ada. Satu database menunjukkan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi. Database mempunyai kegunaan dalam mengatasi penyusunan dan penyimpanan data, maka seringkali masalah yang dihadapi adalah:

1. Redudansi dan inkonsistensi data
2. Kesulitan dalam pengaksesan data
3. Isolasi data untuk standarisasi
4. Multiuser
5. Keamanan data

6. Integritas data
7. Kebebasan data (Urva dan Siregar, 2016:93)

II.2.6. MySQL

Menurut Solichin (2016:138), MySQL merupakan salah satu perangkat lunak basis data yang sangat populer. Saat ini tersedia versi MySQL yang berbayar (MySQL Enterprise Edition), namun tetap tersedia versi MySQL yang gratis (MySQL Community Edition). MySQL Community Edition dapat diunduh secara gratis dan bebas digunakan dalam berbagai keperluan. Walaupun demikian, secara fitur dan kemampuan, MySQL versi tidak terlalu jauh berbeda dengan versi berbayar. Kelebihan MySQL adalah sebagai berikut:

1. Bersifat open source, yang memiliki kemampuan untuk dapat dikembangkan lagi.
2. Menggunakan bahasa SQL (Structure Query Language), yang merupakan standar bahasa dunia dalam pengolahan data.
3. Super performance dan realible, tidak bisa diragukan, pemrosesan database-nya sangat cepat dan stabil.
4. Sangat mudah dipelajari (easy of use).
5. Memiliki dukungan support (group) penggunaan MySQL.
6. Multiuser, di mana MySQL dapat digunakan oleh beberapa user dalam waktu yang bersamaan tanpa mengalami konflik.

II.2.7. UML (*Unified Modeling Language*)

UML (*Unified Modeling Language*) digunakan sebagai suatu cara untuk mengkomunikasikan idenya kepada para pemrogram serta calon pengguna sistem/perangkat lunak. Dengan adanyan bahasa yang bersifat standar, komunikasi perancang dengan pemrogram (komunikasi antar anggota kelompok pengembang) serta calon pengguna diharapkan menjadi mulus. adapun pengertian UML menurut para ahli dapat dipaparkan sebagai berikut:

Menurut Shofwan Hanief & Dian Pramana (2018 : 166) menyatakan bahwa: "*Unified Modelling Language* (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisas, merancang dan mendokumentasikan sistem piranti lunak"

Menurut Eng RH. Sianipar (2017 :75) menyatakan bahwa: "UML merupakan singkatan *Unifed Modelling Language*, yang telah menjadi notasi popular untuk mempresentasikan perancangan atas sebuah program berorientasi objek".

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UMLadalah sebagai berikut:

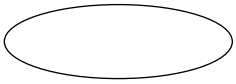
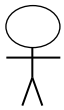
1. *Use case* Diagram

Menurut Sri Mulyani (2018 : 245) menyatakan bahwa *Use Case Diagram* yaitu *diagram* yang menggambarkan dan merepresentasikan aktor, *use cas*e dan *dependencies* suatu proyek dimana tujuan dari diagram ini adalah menjelaskan konsep hubungan antara sistem dengan dunia luar

Jadi, dapat disimpulkan *Use case* adalah langkah – langkah atau urutan

kegiatan yang dilakukan actor dan sistem informasi yang akan dibuat.Secara singkat, *Use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan.

Tabel II.1. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Sri Mulyani : 2018: 245)

2. Diagram Aktivitas (*Activity Diagram*)

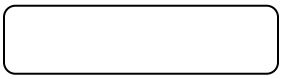
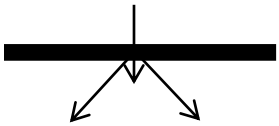
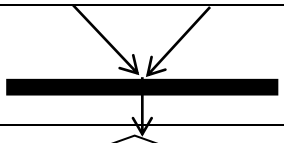
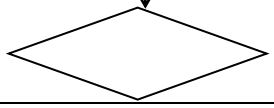
Activity diagram menggambarkan rangkaian aliran dari aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti use case atau interaksi.

Menurut Sri Mulyani (2018 : 249) : "*Activty diagram* adalah diagram UML yang digunakan untuk menggambarkan alur aktivitas dari suatu proses" Menurut

Muhammad Muslihudin dan Oktafianto (2018 :63) mengungkapkan: “Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem”.

Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2:

Tabel II.2. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Sri Mulyani : 2018: 249)

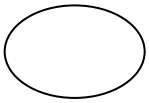
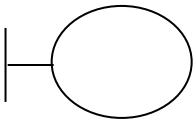
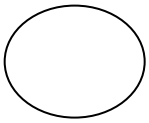
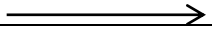
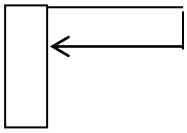
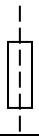
3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Menurut (Irmayani & Susyatih, 2017), *Sequence Diagram* menggambarkan

bagaimana sistem merespon kegiatan user. *Sequence Diagram* yang dibuat yaitu berhubungan langsung dengan kegiatan utama dari sistem anggaran pendapatan dan belanja desa berbasis objek.Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.3:

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class,* berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class,* suatu objek yang berisi logika sistem yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive,* menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Irmayani & Susyati, 2017)

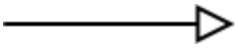
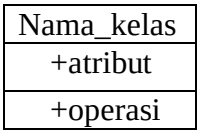
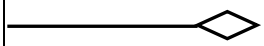
4. Diagram Kelas (*Class Diagram)*

Diagram kelas (*Class Diagram*) adalah diagram yang digunakan untuk menampilkan beberapa kelas serta paket-paket yang ada dalam sistem atau perangkat lunak yang sedang kita kembangkan. Dan berikut ini merupakan penjelasan mengenai

class diagram:

Menurut Sri Mulyani (2018 : 247) menyatakan bahwa : "Class Diagram adalah diagram yang digunakan untuk mempresentasikan kelas, komponen-komponen kelas dan hubungan antara masing-masing kelas" Simbol *class diagram* dan *multiplicity class diagram* dapat dilihat pada Tabel II.4 dan TabelII.5.:

Tabel II.4. Simbol *Class Diagram*

Gambar	Keterangan
	*Generalization*, untuk menghubungkan antar kelas dengan arti umum-khusus. Jadi jika ada kelas dengan arti umum-khusus. Jadi jika ada kelas bermakna umum dan kelas bermakna khusus dapat menggunakan simbol ini.
	*Class*, untuk sebuah kelas pada struktur sistem. Penulisan tidak boleh menggunakan spasi. Simbol ini memiliki 3 susunan, yaitu kotak pertama adalah nama kelas, kedua atribut dan ketiga operasi.
	*Interface*, untuk simbol *interface* atau dalam bahasa indonesianya antar muka. Konsep yang digunakan pun sama dengan pemrograman berorientasi object (OOP).
	Association,digunakan untuk menghubungkan atau merelasikan kelas satu dengan kelas yang lainnya dengan makna umum.
	Directed Association, adalah relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain.
	*Aggregation*, adalah relasi antar kelas dengan makna semua bagian.
	Dependency, adalah relasi antar kelas dengan makna kebergantungan antar kelas.

(Sumber : Sri Mulyani : 2018 : 247)

Tabel II.5. *Multiplicity Class Diagram*

Multiplicity	**Penjelasan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih

0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Sri Mulyani : 2018 : 247)