



BAB II

TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pelayanan

2.1.1.1 Definisi

Levey dan Loomba (1973) dalam (Saudi, 2010:117) menyebutkan bahwa pelayanan kesehatan merupakan upaya yang diselenggarakan secara sendiri atau bersama-sama dalam suatu organisasi, untuk meningkatkan dan memelihara kesehatan, mencegah dan mengobati penyakit serta memulihkan kesehatan perorangan, kelompok, dan ataupun masyarakat.

Sutadi (2005) dalam (Zahwa, 2017:24) pelayanan kesehatan masyarakat merupakan komoditi yang unik dan khusus, tidak dapat disamakan dengan komoditi lain karena pelayanan yang diberikan berupa jasa, sehingga sulit mencapai kepuasan pelanggan.

2.1.1.2 Bentuk dan Jenis

Hodgetts dan Casio (1983) dalam (Saudi, 2010:117) mengemukakan bahwa pelayanan kesehatan secara umum dibagi menjadi dua macam, yaitu:

a. Pelayanan Kesehatan Kedokteran (*Medical Service*)

Pelayanan kedokteran ialah bagian dari pelayanan kesehatan yang penyelenggaraannya dapat dilakukan secara sendiri, bertujuan untuk mengobati penyakit dan memulihkan kesehatan serta sasarannya terutama terhadap perseorangan.

b. Pelayanan Kesehatan Masyarakat (*Public Health Service*)

Pelayanan kesehatan masyarakat ialah bagian dari pelayanan kesehatan yang penyelenggaraannya pada umumnya dilakukan secara bersama dalam suatu organisasi, bertujuan untuk meningkatkan kesehatan dan mencegah penyakit serta sasarannya terutama kepada masyarakat.



Sedangkan Muninjaya (2004) dalam Husainah (2017:544), berpendapat jika bentuk-bentuk pelayanan kesehatan yang harus diterapkan pada masyarakat, yaitu:

- 1) Pelayanannya dilaksanakan secara wajar, tidak melebihi kebutuhan dan daya jangkau masyarakat (*appropriateness*).
- 2) Dari segi biaya, pelayanan kesehatan harus dijangkau oleh masyarakat pada umumnya (*affordable*).
- 3) Manajemennya harus efisien (*efficient*).

2.1.1.3 Tujuan

A.A.Maulana (2013) dalam Husainah (2017:544), pelayanan kesehatan bertujuan untuk:

- 1) Promotif, berarti memelihara dan meningkatkan kesehatan.
- 2) Preventif, berarti pencegahan terhadap orang yang beresiko penyakit.
- 3) Kuratif, berarti penyembuhan penyakit.

2.1.1.4 Kualitas

Lima dimensi yang digunakan konsumen dalam menilai suatu kualitas pelayanan menurut Zeithaml, dkk (1990) dalam Wowor, dkk (2016:114), yaitu:

- 1) Nyata (*Tangibles*)
- 2) Keandalan (*Reabilitas*)
- 3) Kesigapan (*Responsive*)
- 4) Jaminan untuk kepastian (*Assurance*)
- 5) Empati (*Empathy*)

2.1.2 Konsultasi Gizi

2.1.2.1 Definisi

Cornelia (2010) dalam Irawan dan Saurina (2017:1), Konsultasi Gizi merupakan suatu proses yang didalamnya terdapat kegiatan pengumpulan, verifikasi, dan interpretasi data yang sistematis dalam upaya mengidentifikasi masalah gizi serta penyebabnya.

Depkes (1999), konsultasi gizi adalah kegiatan penyampaian pesan-pesan gizi yang dilaksanakan untuk menambah dan meningkatkan sikap, pengertian



serta perilaku positif penderita dan lingkungannya terhadap upaya peningkatan gizi dan kesehatan. Dikutip oleh Agustina (2006:3).

2.1.2.2 Komponen

- 1) Klien
- 2) Konsultan
- 3) Komunikasi
- 4) Waktu
- 5) Pelayanan
- 6) Sarana

2.1.2.3 Tujuan

Suariasa (2012) dalam Sukraniti, dkk (2018:22), tujuan konseling gizi adalah sebagai berikut:

- 1) Membantu klien dalam mengidentifikasi dan menganalisis masalah klien serta memberi alternatif pemecahan masalah. Melalui konseling klien dapat berbagi masalah, penyebab masalah dan memperoleh informasi tentang cara mengatasi masalah.
- 2) Menjadikan cara-cara hidup sehat di bidang gizi sebagai kebiasaan hidup klien. Melalui konseling klien dapat belajar merubah pola hidup, pola aktivitas, pola makan.
- 3) Meningkatkan pengetahuan dan kemampuan individu atau keluarga klien tentang gizi. Melalui konseling klien mendapatkan informasi pengetahuan tentang gizi, diet dan kesehatan.

2.2 Teori Khusus

2.2.1 UML (*Unified Modeling Language*)

Braun, dkk. (Haviluddin, 2011:1) “*Unified Modelling Language (UML)* adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual”.



Gambar 2.1 Logo UML (*Unified Modelling Language*)

Sumber: mahesa.id

2.2.2 Kategori dan Jenis UML (*Unified Modeling Language*)

Haviluddin dalam jurnal penelitiannya yang berjudul Memahami Penggunaan UML (*Unified Modelling Language*), halaman 3-5, menyebutkan bahwa pada UML versi 2 terdiri atas tiga kategori dan memiliki 13 jenis diagram, beberapa di antaranya yaitu:

1) *Class Diagram*

Class diagram menggambarkan struktur statis dari kelas dalam sistem anda dan menggambarkan atribut, operasi dan hubungan antara kelas.

2) *Use Case Diagram*

Diagram yang menggambarkan *actor*, *use case* dan relasinya sebagai suatu urutan tindakan yang memberikan nilai terukur untuk aktor.

3) *Activity Diagram*

Menggambarkan aktifitas-aktifitas, objek, state, transisi state dan event. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas.

4) *Sequence Diagram*

Diagram yang menjelaskan interaksi objek yang disusun berdasarkan urutan waktu.

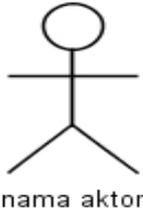
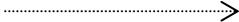
2.2.3 Simbol-Simbol UML (*Unified Modeling Language*)

UML memiliki komponen yang akan digunakan untuk menggambarkan analisa dan desain diagram.



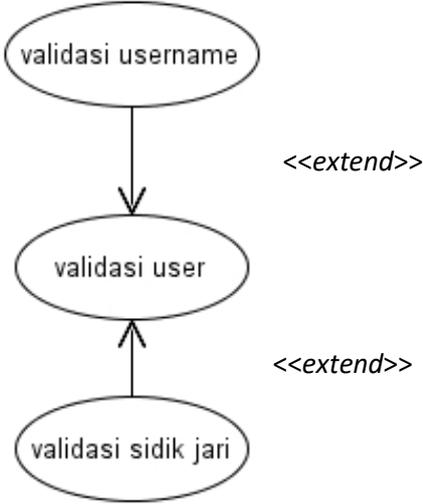
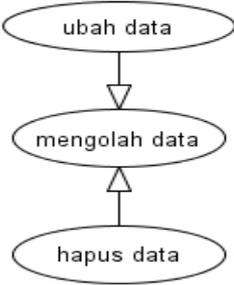
2.2.3.1 Use Case Diagram

Tabel 2.1 Simbol-Simbol pada Use Case Diagram

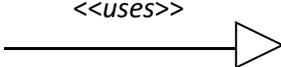
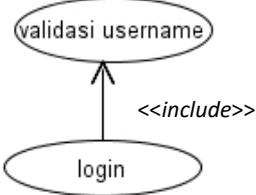
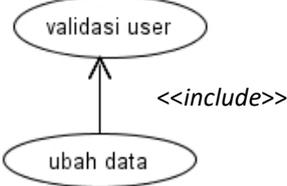
No	Simbol	Deskripsi
1.	<p><i>Use case</i></p> 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal-awal frase nama <i>use case</i> .
2.	<p>Aktor / <i>Actor</i></p> 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
3.	<p>Asosiasi / <i>Association</i></p> 	Komunikasi antar aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor.
4.	<p>Ekstensi / <i>Extend</i></p> <p><code><<extend>></code></p> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang di tambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu; mirip dengan prinsip <i>inheritance</i> pada pemrograman berorientasi objek; biasanya <i>use case</i> tambahan memiliki nama depan yang sama dengan <i>use case</i> yang ditambahkan, misalnya



Lanjutan Tabel 2.1 Simbol-Symbol pada *Use Case Diagram*

		 <p>Biasanya, <i>use case</i> yang menjadi <i>extend</i>-nya merupakan jenis yang sama dengan <i>use case</i> yang menjadi induknya.</p>
5.	<p>Generalisasi/ <i>Generalization</i></p> 	<p>Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya,</p>  <p><u>misalnya:</u> arah panah mengarah pada <i>use case</i> yang menjadi generalisasinya (umum).</p>

Lanjutan Tabel 2.1 Simbol-Simbol pada *Use Case Diagram*

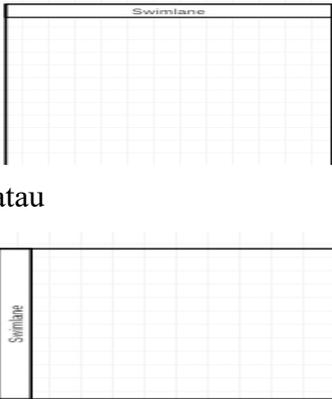
No	Simbol	Deskripsi
6.	Menggunakan/ <i>Include / Uses</i>  	<p>relasi tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankan <i>use case</i> ini</p> <p>ada dua sudut pandang yang cukup besar mengenai include di <i>use case</i>:</p> <ul style="list-style-type: none"> ● <i>Include</i> berarti <i>use case</i> yang ditambahkan akan selalu di panggil saat <i>use case</i> tambahan dijalankan, misalnya pada kasus berikut:  <ul style="list-style-type: none"> ● <i>Include</i> berarti <i>use case</i> yang tambahan akan selalu melakukan pengecekan apakah <i>use case</i> yang di tambahkan telah dijalankan sebelum <i>use case</i> tambahan dijalankan, misal pada kasus berikut:  <p>kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.</p>

Sumber: Rosa dan Shalahuddin (2013:156-158)



2.2.3.2 Activity Diagram

Tabel 2.2 Simbol-Simbol pada Activity Diagram

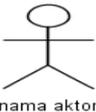
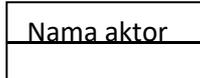
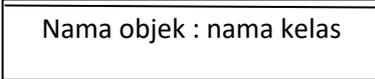
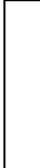
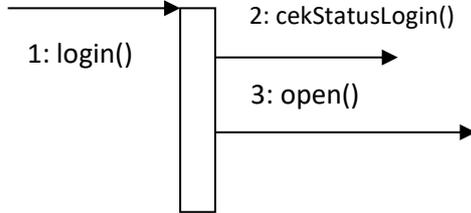
No	Simbol	Deskripsi
1.	Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
2.	Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
3.	Percabangan / <i>decision</i>	Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu.
4.	Penggabungan / <i>join</i> 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
5.	Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
6.	Swimlane  atau	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber: Rosa dan Shalahuddin (2013:162-163)



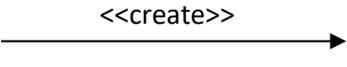
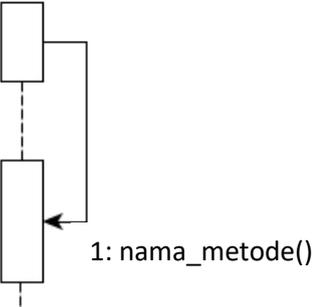
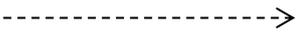
2.2.3.3 Sequence Diagram

Tabel 2.3 Simbol-Simbol pada *Sequence Diagram*

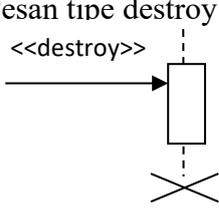
No	Simbol	Deskripsi
1.	<p>Actor</p>  <p>atau</p>  <p>tanpa waktu aktif</p>	<p>Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor</p>
2.	<p>Garis hidup / <i>lifeline</i></p> 	<p>Menyatakan kehidupan suatu objek.</p>
3.	<p>Objek</p> 	<p>Menyatakan objek yang berinteraksi pesan.</p>
4.	<p>Waktu Aktif</p> 	<p>Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya</p>  <p>maka <code>cekStatusLogin ()</code> dan <code>open()</code> dilakukan di dalam metode <code>login()</code> aktor tidak memiliki waktu aktif</p>



Lanjutan Tabel 2.3 Simbol-simbol pada *Sequence* Diagram

5.	Pesan tipe create 	menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
6.	Pesan tipe call 1: nama metode() 	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 yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
7.	Pesan tipe send 1: masukan 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
8.	Pesan tipe return 1: keluaran 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian

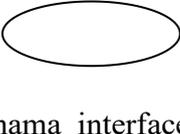
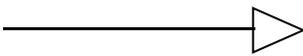
Lanjutan Tabel 2.3 Simbol-simbol pada *Sequence Diagram*

9.	Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada <i>create</i> maka ada <i>destroy</i>
----	---------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------

Sumber: Rosa dan Shalahuddin (2013:165-167)

2.2.3.4 Class Diagram

Tabel 2.4 Simbol-Simbol *Class Diagram*

No	Simbol	Deskripsi
1.	kelas 	Kelas pada struktur sistem
2	antarmuka / interface 	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek
3.	asosiasi / association 	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai <i>multiplicity</i>
4.	asosiasi berarah / <i>directed association</i> 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
5.	generalisasi 	Relasi antarkelas dengan makna generalisasi – spesialisasi (umum khusus)
6.	kebergantungan / <i>dependency</i> 	Relasi antarkelas dengan makna kebergantungan antar kelas



Lanjutan Tabel 2.4 Simbol-Simbol *Class Diagram*

7.	agregasi / <i>aggregation</i> 	Relasi antarkelas dengan makna semua-bagian (<i>whole-part</i>)
----	--------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------

Sumber: Rosa dan Shalahuddin (2013:146-147)

2.3 Teori Program

2.3.1 PHP

PHP adalah singkatan dari PHP *Hypertext Preprocessor*. Berdasarkan jenis bahasanya, PHP merupakan bahasan *scripting* seperti HTML dimana *source code* dari program PHP tetap dalam bentuk file teks murni dan langsung bisa diparser oleh software PHP di server tanpa harus diubah ke bentuk lainnya terlebih dahulu (Tim EMS, 2014:59).



Gambar 2.2 Logo PHP

Sumber: <https://www.php.net/>

2.3.2 XAMPP

Sebelum melakukan pengujian aplikasi, harus ada web server yang sudah terinstall. Dalam Laporan Akhir ini penulis menggunakan XAMPP sebagai *web server*. Menurut Ramadhan dan Rusmawan (2018:3) XAMPP server adalah server yang paling populer di kalangan programmer, selain fiturnya yang lengkap, XAMPP sangat mudah digunakan.



Gambar 2.3 Logo XAMPP

Sumber: <https://www.apachefriends.org/index.html>



2.3.3 MySQL

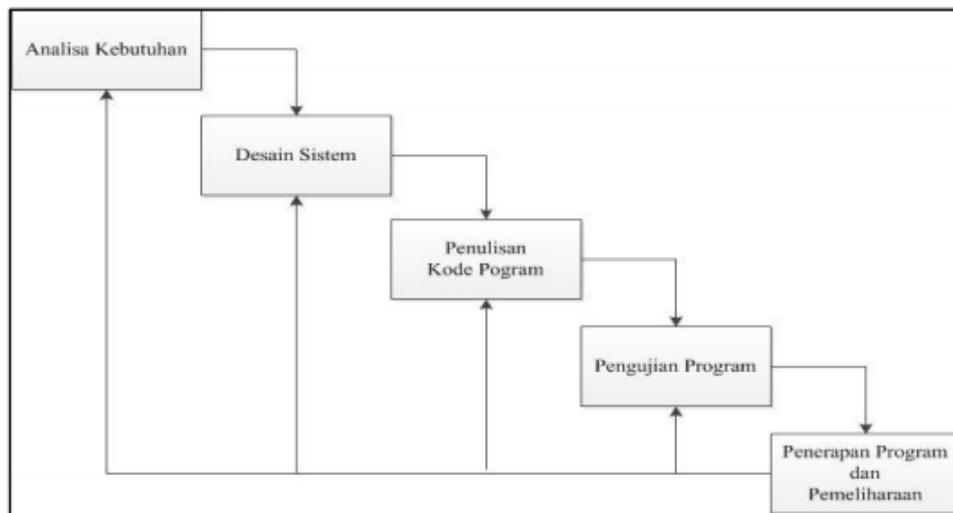
Feri Hari Utami dan Asnawati (2015:89) mengemukakan bahwa MySQL adalah sebuah perangkat lunak system manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread* dan *multi user*.



Gambar 2.4 Logo *MySQL*

Sumber: <https://www.mysql.com/>

2.4 Metode Pengembangan Sistem



Gambar 2.5 Ilustrasi Model *Waterfall*

Dalle, dkk (2020:398) menjelaskan tentang metode *waterfall* menyediakan pendekatan alur hidup software secara *sequential* atau terurut dimulai dari analisa, desain, pengcodingan, uji coba, dan tahap pendukung. Jika langkah satu belum dikerjakan maka tidak akan bisa melakukan pengerjaan langkah 2, 3, dan seterusnya.

1. Analisa

Sebuah analisis persyaratan sistem yang dilakukan dengan cara wawancara atau literatur belajar. Menghasilkan data yang berkaitan dengan kebutuhan klien



dalam proses pembuatan sistem atau dapat dikatakan dokumen persyaratan pengguna, yang merupakan tujuan dari sistem analisis untuk menerjemahkannya ke dalam bahasa pemrograman.

2. Desain

Proses ini menerjemahkan persyaratan ke dalam desain perangkat lunak yang dapat diperkirakan sebelum pengkodean dibuat. Berfokus pada: struktur data, arsitektur perangkat lunak, representasi antarmuka, dan detail prosedural. Menghasilkan sebuah dokumen yang disebut persyaratan perangkat lunak, yang merupakan apa yang programmer gunakan dalam kegiatan manufaktur sistemnya.

3. Coding dan Testing

Terjemahkan desain ke dalam bahasa pemrograman sehingga dapat dibaca oleh komputer. Setelah coding tahap berikutnya, pengujian. Pengujian dalam hal ini memiliki manfaat menemukan kesalahan dengan sistem dan kemudian memperbaikinya.

4. Penerapan

Step ini bisa dikatakan akhir dalam pembuatan sebuah sistem. Setelah melakukan analisa, design dan pengkodean maka sistem yang sudah jadi akan dipakai oleh user.

5. Pemeliharaan

Software yang sudah disampaikan kepada client pasti mengalami revisi. Revisi tersebut bisa karena mengalami kesalahan karena software harus menyesuaikan dengan daerah nya (*peripheral* atau sistem operasi baru) baru, atau karena client membutuhkan perkembangan fungsional.