BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Tinjauan pustaka merupakan landasan teori terkait penelitian terdahulu untuk mendukung penelitian yang sedang dilakukan penulis. Tinjauan pustaka ini dapat dilihat pada tabel 2.1

Tabel 2.1 Tinjauan Pustaka

No.	Peneliti	Tahun	Judul
1.	Rohmat Tulloh, Dadan Nur Ramadan, Dendi Gusnadi	2020	Aplikasi E-KMS untuk Pendataan dan Rekapitulasi Tumbuh Kembang Balita di Posyandu Mekar Arum 18
2.	Faizatul Ummah, Ari Kusdiana, Muhammad Ganda Saputra	2021	Pemberdayaan Kader Posyandu dalam Pencatatan dan Pelaporan Berbasis *Website*
3.	Donaya Pasha, Ajeng Savitri Puspaningrum, Delicia Izazi Eka Eritiana	2023	Permodelan E-Posyandu untuk Perkembangan Balita Menggunakan *Extreme Programming*
4.	Duwi Kartini Aprilia, Rodianto	2023	Perancangan Sistem Informasi Manajemen Posyandu (SIMPADU) untuk Meningkatkan Kualitas Layanan Posyandu
5.	Muhammad Saefudin, Dyah Ayu Megawaty, Debby Alita, Rillya Arunda, Edwin Tenda	2023	Penerapan *Framework* Laravel pada Sistem Informasi Posyandu Berbasis *Website*

Berdasarkan tabel 2.1, penjelasan dari tinjauan pustaka sebagai berikut:

1. Penelitian yang dilakukan oleh Tulloh et al., (2020) meneliti tentang Aplikasi E-KMS untuk Pendataan dan Rekapitulasi Tumbuh Kembang Balita di Posyandu Mekar Arum 18. Penelitian bertujuan membuat aplikasi E-KMS (Elektronik Kartu Menuju Sehat) untuk mengatasi permasalahan pendataan balita yang sebelumnya masih manual dan meningkatkan angka partisipasi ibu dan balita untuk datang ke posyandu.

2. Penelitian yang dilakukan oleh Ummah et al., (2021) meneliti tentang Pemberdayaan Kader Posyandu dalam Pencatatan dan Pelaporan Berbasis *Website*. Penelitian bertujuan membuat aplikasi SIPANDU (Sistem Informasi Posyandu) untuk melakukan pendataan secara *online* terhadap pasien khususnya bayi dan balita, menyimpan data hasil pemeriksaan, serta menyajikan laporan hasil pemeriksaan pasien kepada ibu balita dan pihak puskesmas agar dapat menyesuaikan dengan tingkat kebutuhan masing-masing pengguna.
3. Penelitian yang dilakukan oleh Pasha et al., (2023) meneliti tentang Permodelan E-Posyandu untuk Perkembangan Balita Menggunakan *Extreme Programming.* Penelitian bertujuan membuat sistem elektronik posyandu secara *online* untuk pengelolaan data dan informasi. Bagian admin dapat melakukan pengolahan data anggota, manajemen data balita, dan melihat status tumbuh kembang balita, sedangkan bagi anggota sistem memberikan akses melalui perangkat *mobile* untuk informasi gizi, dokter, dan status tumbuh kembang balita dengan mudah menggunakan nomor NIK.
4. Penelitian yang dilakukan oleh Aprilia & Rodianto, (2023) meneliti tentang Perancangan Sistem Informasi Manajemen Posyandu (SIMPADU) Untuk Meningkatkan Kualitas Layanan Posyandu. Penelitian bertujuan membuat Sistem Informasi Manajemen Posyandu (SIMPADU) berbasis web yang memberikan kemudahan bagi masyarakat dalam pendaftaran dan mengakses jadwal posyandu. Petugas kesehatan dan kader posyandu dapat menginput data sasaran dan kunjungan guna pengambilan data, sedangkan kepala puskesmas dapat memantau hasil sasaran kunjungan posyandu.

5. Penelitian yang dilakukan oleh Saefudin et al., (2023) meneliti tentang Penerapan *Framework* Laravel pada Sistem Informasi Posyandu Berbasis *Website*. Penelitian bertujuan membuat sistem dengan menerapkan *framework* laravel untuk memudahkan admin atau kader posyandu yang bertugas dalam mengelola data, sehingga dalam hal pengelompokan ataupun pencarian data menjadi lebih efisien.

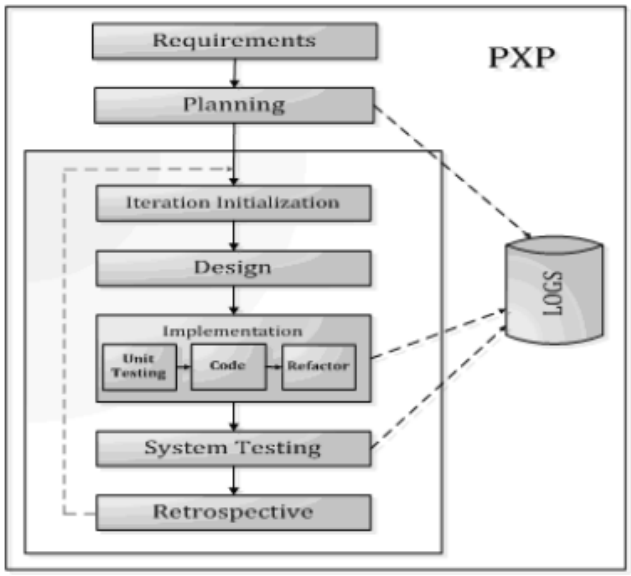
2.2 Posyandu

Menurut Aprilia & Rodianto, (2023) posyandu adalah perpanjangan dari puskesmas guna memberikan pelayanan kesehatan berkala karena banyak masyarakat yang enggan berobat ke puskesmas. Posyandu termasuk kegiatan kesehatan yang bersifat UKBM, yaitu Upaya Kesehatan Bersumberdaya Masyarakat. Oleh karena itu, prinsip pelaksanaan posyandu adalah dari, oleh, dan untuk masyarakat, sedangkan lokasinya berada di wilayah kerja puskesmas. Meskipun digerakkan oleh masyarakat dengan tujuan untuk meningkatkan masyarakat sendiri, tetapi penyelenggaraan posyandu harus diawasi oleh petugas kesehatan pada puskesmas setempat.

Menurut Tulloh et al., (2020) posyandu merupakan salah satu upaya keterlibatan masyarakat dalam memberikan pelayanan dan pemantauan kesehatan terpadu serta menjadi wadah peran masyarakat dalam menyelenggarakan kegiatan, seperti pelayanan keluarga berencana, program kesehatan ibu dan anak, imunisasi, pencegahan diare, juga pemantauan gizi masyarakat.

2.3 Metode *Personal Extreme Programming* (PXP)

Menurut Dzhurov et al., (2009) metode PXP adalah metode yang digunakan untuk mengembangkan perangkat lunak oleh programmer secara perseorangan. Tahapan metode *personal extreme programming* dapat dilihat pada gambar 2.1



Gambar 2.1 Tahapan *Personal Extreme Programming*
Sumber : Dzhurov et al., (2009)

Berdasarkan gambar 2.1, terdapat 7 tahapan dalam metode PXP, sebagai berikut:

1. *Requirement*, terdiri dari persyaratan fungsional dan non-fungsional sistem. Persyaratan fungsional merupakan fitur-fitur yang diperlukan sistem, sedangkan persyaratan non-fungsional membatasi bagaimana sistem bekerja.
2. *Planning,* tahap pengembang menyusun serangkaian tugas berdasarkan daftar persyaratan. Setiap tugas dikategorikan dan diperkirakan menggunakan data perencanaan dari proyek sebelumnya. Jika data sebelumnya tidak tersedia maka pengembang harus memperkirakan waktu yang diperlukan untuk menyelesaikan tugas.

3. *Iteration initialization,* setiap iterasi dimulai dengan pemilihan tugas yang akan difokuskan. Panjang iterasi bervariasi antara 1-3 minggu tergantung pada lingkup proyek.
4. *Design,* pengembang membuat sistem yang akan diimplementasikan dalam iterasi yang sedang berjalan. Pengembang merancang sistem untuk memenuhi persyaratan klien saat ini dengan memperkirakan apa yang dibutuhkan.
5. *Implementation*, pada fase ini tahap pembuatan kode dilakukan. Pengembang mengimplementasikan semua objek yang telah didefinisikan dalam fase desain sebelumnya dan mengujinya. Fase ini terdiri dari 3 fase, yaitu pembuatan kode, refaktorisasi kode, dan pengujian.
6. *System testing*, pada fase ini kode digabungkan tanpa kesalahan dan semua pengujian unit harus berhasil lolos.
7. *Retrospective,* tahap yang menandai akhir dari proses keseluruhan. Pengembang harus memverifikasi waktu yang diperkirakan untuk tugas sesuai dengan waktu aktual. Fase retrospektif dapat memulai iterasi baru dengan berpindah ke fase inisialisasi iterasi atau menandai akhir pengembangan proyek ketika semua persyaratan klien telah terpenuhi dan tidak ada kesalahan yang tersisa.

2.4 *Flowchart*

Menurut Pahlevi & Ismawati, (2019) *flowchart* merupakan sebuah bagan alir yang menjelaskan alur jalannya program dari algoritma yang ada. Suatu proses di *flowchart* dinyatakan dengan simbol-simbol yang sesuai dan dihubungkan oleh garis penghubung. *Flowchart* dalam tahap analisis masalah berperan memeriksa

kembali bagian-bagian yang mungkin terlewatkan. Simbol *flowchart* dapat dilihat pada tabel 2.2.

Tabel 2.2 Simbol *Flowchart*

No.	Simbol	Keterangan
1.	*Connector off page* 	Simbol untuk menghubungkan proses pada halaman berbeda.
2.	*Connector on page* 	Simbol untuk menghubungkan proses pada halaman yang sama.
3.	*Terminal* point 	Simbol untuk memulai atau mengakhiri suatu proses.
4.	*Processing* 	Simbol untuk menunjukkan proses yang ada pada program.
5.	*Decision* 	Simbol untuk menentukan keputusan berdasarkan kondisi yang ada di program.
6.	*Magnetic-tape unit* 	Simbol untuk menunjukkan *input* atau *output* pada pita *magnetic*.
7.	*Input-ouput* 	Simbol untuk menampilkan proses *input-output* dalam program.
8.	*Predefined* 	Simbol untuk melaksanakan suatu bagian dari proses.
9.	*Preparation* 	Simbol untuk persiapan penyimpanan yang terdapat pada *storage.*
10.	Manual *input* 	Simbol untuk melakukan *input* (masukkan) secara manual.
11.	Manual *operation* 	Simbol untuk melakukan pemrosesan secara manual.
12.	*Punch card* 	Simbol untuk menunjukkan *input* berasal dari kartu atau *output* yang ditulis di kartu.
13.	*Disk online & storage* 	Simbol untuk menunjukkan bahwa *input* telah disimpan ke *disk*.

Sumber : Pahlevi & Ismawati, (2019)

No.	Simbol	Keterangan
14.	*Document* 	Simbol untuk menunjukkan bahwa *input* dokumen dalam bentuk kertas atau *output* yang dicetak ke kertas.
15.	*Flow direction*	Simbol untuk menghubungkan satu simbol dengan simbol lainnya.
16.	*Display* 	Simbol untuk menunjukkan perangkat *output.* Misalnya printer, layar, dan lain-lain.

Sumber : Pahlevi & Ismawati, (2019)

2.5 *Unified Modelling Language* (UML)

Menurut Fitriani et al., (2022) UML adalah bahasa yang dipakai untuk menggambarkan *requirement*, melakukan analisis dan desain, serta mendeskripsikan arsitektur dalam pemrograman berorientasi objek. Terdapat 3 diagram pada UML yang memiliki fungsi masing-masing diantaranya, yaitu:

2.5.1 *Use Case Diagram*

Menurut Fitriani et al., (2022) *use case* adalah interaksi antara satu atau lebih aktor pada sistem yang dibangun. *Use case* digunakan untuk menentukan fitur-fitur yang tersedia dalam sistem dan siapa saja yang berwenang untuk menggunakannya. Simbol *use case diagram* dapat dilihat pada tabel 2.3

Tabel 2.3 Simbol *Use Case Diagram*

No.	Simbol	Keterangan
1.	*Use case* 	Unit atau aktor biasanya dinyatakan dengan kata kerja diawal frase.
2.	*Actor* 	Orang atau proses yang akan berinteraksi dengan sistem yang dibuat.
3.	Asosiasi	Komunikasi antara aktor dengan *use case* yang berpartisipasi.

Sumber : Fitriani et al., (2022)

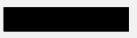
No.	Simbol	Keterangan
4.	Generalisasi	Hubungan umum-khusus antara dua *use case* di mana suatu fungsi adalah fungsi yang lebih umum.
5.	*<<Extend>>*	*Use case* dapat berdiri sendiri walaupun tanpa *use case* tambahan.
6.	*<<Include>>*	*Use case* yang ditambahkan akan dipanggil ketika *use case* tambahan dijalankan.

Sumber : Fitriani et al., (2022)

2.5.2 *Activity Diagram*

Menurut Fitriani et al., (2022) *activity diagram* adalah gambaran aktivitas yang menunjukkan aliran pengoperasian sistem. Simbol *activity diagram* dapat dilihat pada tabel 2.4.

Tabel 2.4 Simbol *Activity Diagram*

No.	Simbol	Keterangan
1.	Status awal 	Diagram aktivitas yang mempunyai sebuah status awal.
2.	Aktivitas 	Kegiatan yang dilaksanakan sistem diawali dengan kata kerja.
3.	*Decision* 	Asosiasi percabangan dilakukan apabila pilihan aktivitas lebih dari satu.
4.	*Join* 	Asosiasi penggabungan dilakukan ketika lebih dari satu aktivitas dikumpulkan menjadi satu.
5.	*Swimlane* 	Memisahkan organisasi bisnis dengan aktivitas yang berlangsung.
6.	Status akhir 	Diagram aktivitas mempunyai sebuah status akhir.

Sumber : Fitriani et al., (2022)

2.5.3 *Class Diagram*

Menurut Fitriani et al., (2022) *class diagram* adalah bentuk penggambaran struktur sistem yang mencakup uraian kelas-kelas guna membentuk sistem. Simbol *class diagram* dapat dilihat pada tabel 2.5.

Tabel 2.5 Simbol *Class Diagram*

No	Simbol	Keterangan
1.	*Class* (kelas) Nama_Kelas +Atribut - Operasi	Stuktur yang berisi nama kelas, atribut dan operasi
2.	*Interface* (antarmuka)	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3.	Asosiasi berarah	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lain.
4.	Generalisasi	Relasi antar kelas dengan makna umum khusus.
5.	*Dependency*	Relasi antar kelas dengan makna ketergantungan antar kelas.
6.	Agregasi	Relasi antar kelas dengan makna semua bagian.
7.	Asosiasi	Relasi antar kelas dengan makna umum

Sumber : Fitriani et al., (2022)

2.6 *Website*

Menurut Nursyanti et al., (2019) *website* adalah halaman yang menampilkan berbagai teks, data, gambar, video, atau gabungan dari semuanya bersifat tetap dan dapat berubah.

Menurut Abdulloh, (2020) *website* adalah data digital yang menampilkan informasi berupa teks, gambar, atau animasi melalui koneksi internet sehingga semua orang di dunia dapat mengakses dan melihatnya.

2.7 Laravel

Menurut Zanin et al., (2019) dalam artikelnya yang berjudul ”*A Comparative Study of PHP Framework Performance*” bahwa laravel mengungguli Symfony dan CodeIgniter dalam hal permintaan per detik, penggunaan memori, serta merespon dengan cepat. Namun, dibandingkan kedua *framework* tersebut. Laravel memiliki satu kelemahan, yaitu pada jumlah file.

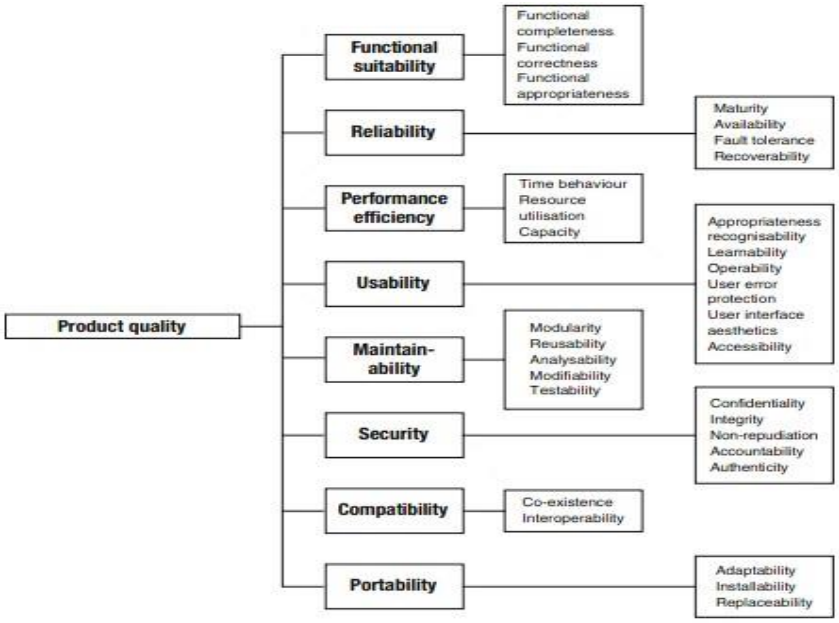
Menurut Sunardi & Suharjito, (2019) terdapat perbandingan antara *framework* laravel dan slim. Laravel mampu untuk mengatur berbagai *function* dan *library* membuatnya ideal untuk proyek berskala besar. Namun, ada satu kelemahan laravel, yaitu tingkat kesulitan dalam penggunaannya lebih tinggi daripada slim karena lebih rumit.

2.8 MySQL

Menurut Jantce TJ Sitinjak et al., (2020) MySQL merupakan server aplikasi *database*. SQL singkatan dari *Structured Query Language.* Bahasa yang digunakan untuk mengelola *database.* MySQL juga memiliki kemampuan menambah, mengubah, dan menghapus data di *database.*

2.9 Pengujian ISO 25010

Menurut Harun, (2020) Pengujian ISO 25010 terdiri dari 8 karakteristik yang berfokus pada sifat statis perangkat lunak dan sifat dinamis sistem komputer. Delapan karakteristik ISO 25010 dapat dilihat pada gambar 2.2



Gambar 2.2 ISO 25010
Sumber : Harun, (2020)

Berdasarkan gambar 2.2, terdapat 8 karakteristik pada ISO 25010, sebagai berikut:

1. *Functional suitability* merupakan kemampuan sistem memiliki fitur-fitur yang hanya tersedia untuk situasi tertentu.
    a. *Functional completeness,* fungsi yang ditetapkan mencakup semua tugas yang ditentukan dan tujuan pengguna.
    b. *Functional correctness,* sistem mendapatkan hasil tepat dengan keakuratan yang diperlukan.
    c. *Functional appropriateness,* sistem membantu pencapaian suatu tugas dan tujuan.
2. *Reliability* merupakan fungsi yang dijalankan oleh sistem hanya setelah jangka waktu dan dalam kondisi tertentu.
    a. *Maturity*, sistem memenuhi persyaratan keandalan selama pengoperasian normal.

b. *Availability*, sistem operasional dapat diakses bila diperlukan untuk digunakan.

c. *Fault tolerance,* sistem berfungsi sebagaimana mestinya meskipun terjadi kegagalan di perangkat keras dan perangkat lunak.

d. *Recoverability,* sistem memulihkan data yang terkena dampak langsung dengan memulihkan sistem ke kondisi yang diinginkan.

3. *Performance efficiency* merupakan kemampuan sistem memberikan kinerja lebih baik daripada jumlah sumber daya yang digunakan.

   a. *Time behavior*, respon waktu pemrosesan dan *output* memenuhi persyaratan untuk menjalankan fungsinya.

   b. *Resource utilization,* jumlah dan jenis sumber daya yang digunakan sistem untuk menjalankan fungsinya memenuhi persyaratan.

   c. *Capacity,* batas maksimum atau parameter sistem memenuhi persyaratan.

4. *Usability* merupakan efisiensi, efektivitas, dan kepuasan pengguna dalam memanfaatkan sistem konteks penggunaan.

   a. *Appropriateness recognizability*, sistem yang membantu pengguna memutuskan fitur apabila cocok untuk tugas mereka.

   b. *Learnability*, sistem untuk memungkinkan pengguna mencapai tingkat kemahiran yang diperlukan dalam waktu yang diperlukan.

   c. *Operability,* sistem untuk memungkinkan pengguna mengoperasikan dan mengendalikan sistem dengan mudah dan efektif.

d. *User error protection*, sistem melindungi pengguna dari kesalahan.

e. *User interface aesthetics*, antarmuka pengguna untuk menarik dan memuaskan pengguna.

f. *Accessibility*, sistem untuk digunakan oleh penyandang disabilitas.

5. *Maintainability* merupakan kemampuan sistem dapat dimodifikasi meliputi perbaikan dan pengembangan untuk menyesuaikan dengan lingkungan.

   a. *Modularity*, sistem terdiri sedemikian rupa sehingga perubahan pada satu komponen memiliki dampak minimal pada komponen lainnya.

   b. *Reusability*, suatu aset dapat digunakan dalam beberapa sistem atau dibangun diatas aset lain.

   c. *Analysability*, menilai dampak terhadap sistem dari perubahan yang direncanakan pada satu atau lebih bagian untuk mendiagnosis penyebab kegagalan produk dan mengidentifikasi bagian yang akan diubah.

   d. *Modifiability*, sistem dapat diubah secara efektif dan efisien tanpa menimbulkan cacat atau mempengaruhi kualitas produk yang sudah ada.

   e. *Testability,* pengujian dapat dilakukan untuk menetapkan kriteria pengujian sistem dan menentukan apabila kriteria tersebut terpenuhi.

6. *Security* merupakan kemampuan sistem menyediakan layanan untuk melindungi terhadap akses, modifikasi, dan perusakan yang berbahaya.

   a. *Confidentiality*, sistem menjamin kerahasiaan sehingga hanya pengguna berwenang yang dapat mengakses data.

b. *Integrity*, sistem melindungi program komputer dan data dari akses atau modifikasi yang tidak sah.

c. *Non-repudiation*, bukti bahwa suatu tindakan terjadi dan tidak dapat disangkal di kemudian hari.

d. *Accountability*, tindakan suatu entitas dapat ditelusuri dengan jelas ke entitas tersebut.

e. *Authenticity*, identitas subjek atau sumber daya dapat dibuktikan sesuai dengan klaimnya.

7. *Compatibility* merupakan kemampuan sistem untuk bertukar informasi dan menjalankan fungsi yang diperlukan saat menggunakan perangkat keras atau perangkat lunak yang sama.

   a. *Co-existence*, suatu produk secara efisien menjalankan fungsi yang diperlukan sambil berbagi lingkungan dan sumber dayanya dengan produk lain tanpa memberikan dampak negatif pada produk lain.

   b. *Interoperability*, dua atau lebih komponen dapat bertukar informasi dan menggunakan informasi yang dipertukarkan.

8. *Portability* merupakan kemampuan sistem dapat dipindahkan dari satu ruang ke ruang lain.

   a. *Adaptability*, sistem dapat beradaptasi terhadap perangkat keras dan perangkat lunak yang berbeda atau terus berkembang.

   b. *Installability*, sistem dapat berhasil dipasang dan dihapus dalam lingkungan tertentu.

c. *Replaceability*, suatu produk dapat menggantikan produk perangkat lunak lain yang ditujukan untuk tujuan yang sama di lingkungan yang sama.

2.10 Skala Likert

Menurut Sugiyono, (2020) skala likert dipakai untuk mengukur sikap dan persepsi orang atau kelompok tentang situasi sosial. Variabel yang akan diukur dijabarkan dengan indikator-indikator kemudian dijadikan titik tolak pengembangan instrumen berupa pernyataan atau pertanyaan. Skala respon setiap item dalam instrumen berkisar dari sangat positif hingga sangat negatif. Penilaian skala likert dapat dilihat pada tabel 2.6

Tabel 2.6 Penilaian Skala Likert

Pernyataan	Kode	Bobot Nilai
Sangat Setuju	SS	5
Setuju	S	4
Netral	N	3
Tidak Setuju	TS	2
Sangat Tidak Setuju	STS	1

Sumber : Sugiyono, (2020)

Hasil penilaian responden dihitung dengan presentase kelayakan menggunakan perhitungan dibawah ini:

$$\text{Presentase} = \frac{\text{skor aktual}}{\text{skor ideal}} \times 100\%$$

Kemudian presentase kelayakan yang didapat dibandingkan dengan tabel konversi nilai. Skala konversi nilai dapat dilihat pada tabel 2.7

Tabel 2.7 Skala Konversi Nilai

Presentase (%)	Interpretasi
$90 \leq x$	Sangat Baik
$80 \leq x < 90$	Baik
$70 \leq x < 80$	Cukup
$60 \leq x < 70$	Kurang
$x < 60$	Sangat Kurang

Sumber : Jogiyanto, (2008)