

BAB II LANDASAN TEORI

Menyajikan berbagai konsep teoritis yang menjadi landasan penelitian dan acuan dalam pemecahan masalah melalui studi pustaka.

BAB III METODELOGI PENELITIAN

Menjelaskan metode, prosedur, serta perancangan sistem yang digunakan dalam penelitian.

BAB IV HASIL DAN PEMBAHASAN

Memaparkan temuan penelitian dalam pengembangan sistem Informasi pemantauan perkembangan calistung anak di Rumah Belajar Fatimah Az-zahra beserta hasil uji coba menggunakan *blackbox testing*.

BAB V KESIMPULAN DAN SARAN

Berisi rangkuman seluruh hasil penelitian serta rekomendasi untuk pengembangan sistem ke depan agar lebih optimal.

BAB II

LANDASAN TEORI

2.1 Kajian Teori

Kajian teori yang digunakan sebagai dasar dalam penelitian ini meliputi beberapa konsep berikut:

2.1.1 Perancangan

Menurut (Siahaan et al., 2022), Mendefinisikan perancangan sebagai tahapan pembuatan desain atau skema sistem yang berfungsi sebagai representasi awal dalam pengembangan sistem Informasi berbasis web. Sementara itu, (Fitri et al., 2024) mengemukakan bahwa perancangan merupakan serangkaian kegiatan untuk menyusun dan mengembangkan sistem Informasi dengan mengutamakan kesesuaian terhadap keperluan pemakai, sehingga sistem dapat dioperasikan secara efektif dan efisien. Dari dengan mengacu pada kedua sudut pandang tersebut, perancangan dapat dipahami sebagai proses penyusunan dan pembuatan arsitektur sistem secara terstruktur guna menghasilkan produk yang selaras dengan kebutuhan pengguna.

2.1.2 Sistem

Menurut (Gunawan et al., 2022), sistem adalah kumpulan komponen yang saling terhubung dan bekerja sama dalam mengolah data menjadi Informasi yang berguna. Sementara itu, (Brata, 2021), mendefinisikan sistem sebagai sekelompok elemen atau prosedur yang berkaitan satu sama lain untuk mencapai sasaran secara teratur. Berdasarkan kedua pandangan tersebut, dapat disimpulkan bahwa sistem tersusun atas komponen-komponen yang saling terintegrasi dan berfungsi secara kolektif untuk mencapai tujuan bersama.

2.1.3 Informasi

Menurut (Syifa Fauziyah 2022), Informasi adalah hasil olahan data yang bermanfaat untuk meningkatkan efektivitas dan efisiensi kerja. Adapun (Hasan et al., 2023), berpendapat bahwa Informasi merupakan data yang telah diproses sehingga bernilai dan dapat menjadi dasar pengambilan keputusan.

Dengan demikian, Informasi dapat diartikan sebagai data yang telah diolah hingga memiliki makna dan kegunaan bagi penggunanya.

2.1.4 Monitoring

Monitoring merupakan suatu mekanisme pengawasan yang dirancang untuk memantau kondisi suatu entitas atau proses secara berkala dan *real-time*, serta memberi Informasi atau notifikasi ketika terjadi perubahan penting atau anomali pada sistem yang diawasi. Dalam konteks teknologi Informasi, sistem ini dapat diterapkan untuk memantau jaringan, *server*, perangkat *IoT*, maupun parameter lingkungan, sehingga dapat meningkatkan efisiensi, respons cepat terhadap gangguan, serta mendukung pengambilan keputusan yang berbasis data (Ishaq, 2023).

2.1.5 perkembangan Anak Usia Dini

Usia dini merupakan masa di mana anak mengalami transformasi perkembangan dalam beragam aspek, seperti kemampuan berpikir, keterampilan gerak, interaksi sosial, serta kestabilan emosi. Tahap ini dianggap sangat fundamental karena berperan sebagai pijakan awal bagi proses tumbuh kembang anak di kemudian hari (Hasanah et al., 2023), menyatakan bahwa sistem pendataan perkembangan anak yang masih bersifat manual rentan memicu ketidakakuratan *input* data dan memperlambat proses distribusi Informasi kepada orang tua siswa.

2.1.6 Calistung

Calistung merupakan kompetensi fundamental yang wajib dikuasai oleh anak pada masa prasekolah, yaitu keterampilan membaca, menulis, dan berhitung. Kemampuan ini menjadi pondasi utama dalam proses pembelajaran selanjutnya serta berperan penting dalam perkembangan kognitif anak. Menurut (Hasanah et al., 2023) kemampuan dasar anak seperti membaca, menulis, dan berhitung perlu dimonitor secara terstruktur agar perkembangan anak dapat diketahui secara optimal dan dapat diberikan stimulasi yang sesuai.

2.1.7 Website

Menurut (Hidayah & Rofiqoh, 2024) dalam e-jurnal perancangan *website* berbasis sistem Informasi, *website* yang dirancang dengan struktur dan

navigasi yang baik mampu meningkatkan efektivitas penggunaan sistem karena memudahkan pengguna dalam mengakses dan memahami Informasi yang disediakan. *Website* merupakan kumpulan halaman *web* yang saling terhubung dan digunakan sebagai media penyampaian Informasi serta sarana interaksi antara pengguna dan sistem. Dalam konteks sistem Informasi berbasis *web*, *website* berperan penting sebagai antarmuka utama yang mendukung proses pengelolaan data, penyajian Informasi, serta interaksi pengguna secara efektif dan efisien.

2.1.8 Unified Modeling Language (UML)

Unified Modeling Language (UML) adalah suatu Bahasa standar yang digunakan untuk membuat diagram visual yang menggambarkan sistem perangkat lunak. Sederhananya, UML adalah seperti “peta” atau “*blueprint*” yang membantu kita memahami dan merancang sebuah sistem, baik itu sistem Informasi, aplikasi *mobile*, atau perangkat lunak lainnya. UML adalah alat yang sangat berharga dari para pengembang perangkat lunak dan analisis sistem. Dengan memahami konsep dasar UML dan berbagai jenis diagramnya, kita dapat merancang sistem yang lebih baik, lebih mudah dipahami, dan lebih mudah dipelihara (Putri et al., 2025).



Gambar2. 1 Logo UML

2.1.9 Use Case Diagram

Dalam konteks *Unified Modeling Language* (UML), *Use Case Diagram* dimanfaatkan untuk memodelkan interaksi yang terjadi antara aktor dan sistem, sekaligus menggambarkan fungsi-fungsi operasional yang disediakan oleh sistem. Diagram ini menjadi landasan awal dalam tahap pengembangan untuk mengidentifikasi kebutuhan fungsional serta relasi antara pemakai dan sistem yang dibangun. Dengan memanfaatkan *Use Case Diagram*, pengembang dapat menganalisis peran masing-masing aktor serta pola respons sistem terhadap setiap aksi yang dilakukan (Kurniawan, 2018)

Fungsi-fungsi strategis *Use Case Diagram* dalam pengembangan sistem Informasi mencakup :

1. Pemetaan kebutuhan sistem

Diagram ini memudahkan proses identifikasi dan pemetaan kebutuhan fungsional yang diinginkan dari sistem secara teratur dan sistematis.

2. Representasi Interaksi Pengguna

Use Case Diagram menggambarkan pihak-pihak (*aktor*) yang terlibat dalam interaksi dengan sistem serta fungsi-fungsi apa saja yang dapat mereka akses.

3. Visualisasi Proses Bisnis

Diagram ini memperlihatkan alur proses bisnis secara runtut, sehingga mempermudah pemahaman terhadap rangkaian aktivitas yang terjadi di dalam sistem.

4. Sarana Komunikasi

Use Case Diagram berfungsi sebagai alat komunikasi antara pengembang dan para pemangku kepentingan untuk menjelaskan gambaran umum sistem serta memastikan bahwa kebutuhan pengguna telah terpenuhi.

5. Validasi Kebutuhan Sistem

Melalui diagram ini, pengembang dapat memeriksa apakah kebutuhan sistem telah lengkap dan selaras dengan ekspektasi pengguna.

6. Gambaran Interaksi Sistem

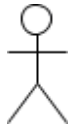
Diagram ini memberikan ilustrasi mengenai bagaimana sistem berinteraksi dengan pengguna maupun sistem lain, sehingga membantu dalam perancangan antarmuka dan pengembangan lebih lanjut.

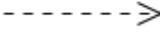
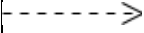
7. Identifikasi Peran Pengguna

Use Case Diagram membantu mengidentifikasi aktor yang terlibat beserta hak akses atau fungsi yang dapat mereka lakukan di dalam sistem.

Berikut adalah simbol-simbol yang terdapat pada diagram *Use case*:

Tabel 2. 1 Simbol *Use Case Diagram*

No	Gambar	Nama	Keterangan
1		<i>Actor</i>	Orang, proses atau sistem lain yang berinteraksi dengan sistem Informasi yang akan dibuat di luar sistem Informasi itu sendiri.
2		<i>Dependency</i>	Relasi ini terjadi ketika suatu komponen yang tidak terikat dengan komponen lain mengalami perubahan, sehingga komponen lain yang menggantungkan diri padanya turut terpengaruh
3		<i>Generalization</i>	Relasi ini menunjukkan hubungan antara dua <i>use case</i> yang bersifat umum dan khusus, di mana salah satu <i>use case</i> berperan sebagai fungsi yang lebih luas dibandingkan dengan fungsi lainnya

No	Gambar	Nama	Keterangan
4		<i>Include</i>	Hubungan ini menghubungkan suatu <i>use case</i> dengan <i>use case</i> lain yang menjadi syarat agar <i>use case</i> tersebut dapat beroperasi
5		<i>Extend</i>	Hubungan ini bersifat opsional, di mana suatu <i>use case</i> dapat berfungsi secara mandiri tanpa terikat dengan <i>use case</i> lainnya
6		<i>Association</i>	Hubungan ini menggambarkan adanya interaksi timbal balik antara aktor dengan <i>use case</i> yang terlibat di dalam sistem
7		<i>System</i>	Simbol ini berfungsi untuk mendefinisikan ruang lingkup sistem yang ditampilkan dalam bentuk paket
8		<i>Use case</i>	Simbol ini menggambarkan serangkaian langkah operasional yang dijalankan oleh sistem untuk memberikan keluaran yang dapat diukur kepada pengguna
9		<i>Collaboration</i>	Interaksi antara berbagai aturan dan komponen yang saling bersinergi menghasilkan dampak yang lebih besar dibandingkan jika masing-masing elemen bekerja secara terpisah

No	Gambar	Nama	Keterangan
10		<i>Note</i>	Komponen fisik yang beroperasi ketika aplikasi sedang aktif dan merepresentasikan suatu kapasitas pemrosesan

2.1.10 Activity Diagram

Sebagai salah satu diagram dalam UML, *Activity Diagram* berfungsi untuk memodelkan alur proses sistem secara sistematis dan bertahap, yang mencakup keseluruhan jalur dari permulaan hingga penyelesaian, termasuk di dalamnya titik-titik keputusan yang dapat memicu alur alternatif. Diagram ini membantu menggambarkan proses bisnis sehingga memudahkan pengembang dalam memahami jalannya sistem sebelum tahap implementasi dilakukan (Putra, et al.,2022).

Tabel 2. 2 Simbol *Activity Diagram*

No	Nama	Simbol	Keterangan
1	<i>Activity</i>		<i>Activity</i> menggambarkan sebuah pekerjaan atau tugas dalam alur kerja (<i>workflow</i>)
2	<i>Start state</i>		Simbol ini menandakan titik awal dari sebuah alur kerja dalam <i>Activity Diagram</i> , yang keberadaannya hanya satu dalam setiap rangkaian proses
3	<i>End state</i>		Simbol ini digunakan untuk menandai titik akhir atau penyelesaian dari suatu alur kerja dalam <i>Activity Diagram</i> , yang dapat hadir lebih dari satu dalam satu diagram
4	<i>State transition</i>		Simbol ini berfungsi untuk mengarahkan alur menuju langkah berikutnya setelah suatu aktivitas selesai dilaksanakan

No	Nama	Simbol	Keterangan
5	<i>Decision</i>		Decision merupakan sebuah titik percabangan dalam <i>Activity Diagram</i> yang menunjukkan adanya kemungkinan lebih dari satu pilihan alur berdasarkan kondisi tertentu

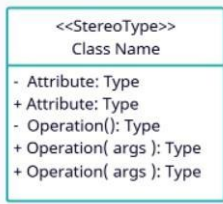
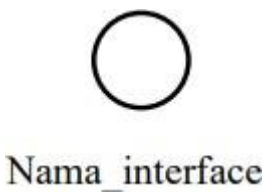
2.1.11 Sequence Diagram

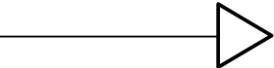
Sequence Diagram termasuk salah satu jenis diagram UML yang berperan dalam memvisualisasikan alur komunikasi antar objek melalui pesan-pesan yang terurut sesuai waktu, dengan representasi objek secara horizontal dan rentang waktu secara vertikal (Firanda et al., 2021).

2.1.12 Class Diagram

Class Diagram berfungsi untuk menggambarkan tipe-tipe objek yang ada dalam sistem serta relasi statik di antara objek-objek tersebut, sekaligus menampilkan atribut, metode, dan batasan dari setiap *Class*, di mana istilah fitur dalam UML digunakan untuk merujuk secara kolektif pada atribut dan metode suatu *Class* (Putra et al., 2022).

Tabel 2. 3 Simbol *Class Diagram*

No	Nama	Simbol	Keterangan
1	<i>Class</i>		Simbol kelas pada struktur sistem terdiri dari tiga bagian berurutan, yaitu nama kelas, atribut, dan operasi, dengan ketentuan bahwa penulisannya tidak boleh mengandung spasi
2	<i>Antarmuka / interface</i>		Konsep yang digunakan sama dengan pemrograman berorientasi objek (<i>OOP</i>)
3	<i>Asosiasi / association</i>		Simbol ini menunjukkan hubungan antar kelas, yang digunakan ketika suatu kelas memanfaatkan kelas lain dalam implementasinya

No	Nama	Simbol	Keterangan
4	Asosiasi berarah / <i>directed association</i>		Relasi ini menghubungkan dua kelas di mana salah satu kelas memanfaatkan kelas lainnya, dan umumnya dilengkapi dengan <i>multiplicity</i> untuk menunjukkan jumlah keterlibatan
5	<i>Generalisasi</i>		Relasi ini menggambarkan hubungan hierarkis antara kelas yang bersifat umum dengan kelas yang lebih spesifik
6	Kebergantungan / <i>dependency</i>		Relasi ini menunjukkan adanya ketergantungan antara satu kelas dengan kelas lainnya
7	Agregasi / <i>aggregation</i>		Relasi ini menggambarkan hubungan keseluruhan-bagian antara suatu kelas dengan kelas-kelas penyusunnya

2.1.13 Framework

Framework dalam pengembangan *web* modern menyediakan fondasi kode yang terstruktur serta modul-modul standar yang membantu *developer* meningkatkan efisiensi, konsistensi, dan kolaborasi dalam pembuatan aplikasi *web* yang responsif, aman, dan mudah diakses. *Framework* Bootstrap merupakan kumpulan komponen yang memudahkan pengembangan tampilan *website* agar responsif dan modern. Penggunaan Bootstrap mempercepat proses pembuatan antarmuka dan meningkatkan pengalaman pengguna pada berbagai perangkat, khususnya *website* (Fadilah & Firdaus, 2024).

Dengan demikian, *framework* dapat diartikan sebagai suatu landasan yang dilengkapi dengan perangkat pendukung standar untuk mempercepat dan menyederhanakan proses pembuatan aplikasi, sehingga menghasilkan karya yang lebih sistematis dan mudah dikelola.

2.1.14 Laravel

Laravel merupakan *Framework PHP* yang banyak digunakan dalam pengembangan aplikasi *website* karena mampu menyediakan struktur pengembangan yang terorganisir dan efisien. Penggunaan Laravel dinilai

dapat mempercepat proses pengembangan aplikasi *web*, khususnya pada aplikasi yang memiliki fitur kompleks seperti *autentikasi* pengguna, manajemen basis data, dan pengaturan *routing*. Laravel membantu pengembang mengurangi jumlah kode yang harus ditulis melalui penyediaan berbagai fitur bawaan serta kemudahan dalam menghubungkan dengan layanan eksternal. Di samping itu, Laravel juga dilengkapi dengan panduan yang komprehensif serta komunitas pengembang yang dinamis, yang pada akhirnya memperlancar jalannya pengembangan serta pemecahan permasalahan yang muncul selama pembuatan aplikasi. Dari sisi kinerja, aplikasi *web* yang dikembangkan menggunakan Laravel juga menunjukkan tingkat responsivitas dan skalabilitas yang baik (Restaldo et al., 2022).

2.1.15 Sistem Database

Berdasarkan penelitian (Putri et al., 2025) *database* merupakan sekumpulan data yang terorganisir dan dikelola secara sistematis untuk mendukung proses penyimpanan, pengolahan, dan pengambilan data secara efisien dalam sebuah sistem Informasi. Keberadaan *database* sangat krusial dalam menunjang berbagai aplikasi, seperti sistem akademik, manajemen sumber daya, hingga sistem Informasi berskala besar. Adapun proses perancangan basis data mencakup tahap konseptual, logis, hingga implementasi, yang bertujuan untuk menghasilkan struktur data yang sesuai dengan kebutuhan pengguna serta menjaga integritas dan kinerja sistem Informasi secara optimal.

2.1.16 Entity Relationship Diagram (ERD)

Menurut (Utami, 2022), Entity Relationship Diagram (ERD) merupakan representasi visual berupa notasi grafis yang dimanfaatkan dalam proses pembuatan *database* untuk memperlihatkan keterkaitan antar data. Diagram ini berperan sebagai sarana pendukung dalam perancangan *database* sekaligus memberikan ilustrasi mengenai relasi antar data pada sistem yang akan dikembangkan.

2.1.17 Laragon

Laragon merupakan salah satu jenis perangkat lunak *development environment* (lingkungan pengembangan) lokal yang banyak digunakan dalam pengembangan sistem Informasi berbasis *web*. Laragon menyediakan ekosistem yang terintegrasi untuk mengelola *server web*, *database server* (seperti *MySQL* atau *PostgreSQL*), serta bahasa pemrograman (seperti *PHP*, *Python*, atau *Node.js*) sebagai bahan dasar dalam pengolahan, pengaksesan, dan pengujian data yang disimpan di dalam *database*. Selain itu, Laragon bersifat *open source* dan ringan, sehingga dapat digunakan dan dikembangkan secara bebas serta efisien oleh pengguna (Jajang Winanjar, 2021).

2.1.18 PHP

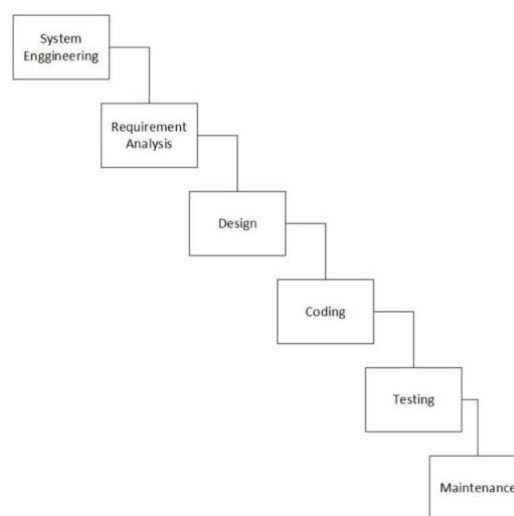
PHP (Hypertext Preprocessor) merupakan bahasa pemrograman yang digunakan untuk mengolah dan menerjemahkan kode program menjadi instruksi yang dapat dipahami oleh komputer. *PHP* bersifat *server-side*, yaitu proses eksekusinya dilakukan di sisi *server* dan hasilnya akan di tampilkan dalam bentuk halaman *web* kepada pengguna (Utami, 2022).

2.1.19 Visual Studio

Menurut Hidayah (Hidayah & Rofiqoh, 2024) dalam e-jurnal Evaluasi *Software* Visual Studio Code Menggunakan Metode *Usability*, Visual Studio Code merupakan *code editor* modern yang dirancang untuk mendukung produktivitas pengembang perangkat lunak. Visual Studio Code menyediakan berbagai fitur pendukung seperti *syntax highlighting*, *auto-completion*, *integrated terminal*, serta *debugging* yang terintegrasi sehingga memudahkan pengembang dalam menulis, mengelola, dan memperbaiki kode program. Penggunaan editor yang tepat seperti *Visual Studio Code* terbukti dapat meningkatkan efisiensi kerja serta meminimalkan kesalahan dalam proses pengembangan perangkat lunak, khususnya pada pengembangan aplikasi berbasis *web*.

2.1.20 Metode *Waterfall*

Metode *Waterfall* merupakan salah satu model pengembangan perangkat lunak dalam *System Development Life Cycle (SDLC)* yang menggunakan pendekatan sistematis dan berurutan, dimana setiap tahapan harus diselesaikan terlebih dahulu sebelum melanjutkan ke tahap berikutnya sehingga menghasilkan proses pengembangan yang terstruktur dan terdokumentasi dengan baik (Zen et al., 2022).



Gambar2. 2 Diagram Metode Waterfall

(Sumber : [Mochammad Haldi Widiyanto](#))

2.1.21 Tahapan Metode *Waterfall*

Menurut (Zen et al., 2022), tahapan dalam Metode *Waterfall* terdiri dari:

1. Analisis Kebutuhan (*Requirements Analysis*)

Tahap identifikasi kebutuhan pengguna dan kebutuhan sistem yang akan dikembangkan.

2. Perancangan Sistem (*System Design*)

Tahap perancangan arsitektur sistem, basis data, antarmuka pengguna, serta kebutuhan perangkat lunak.

3. Implementasi (*Implementation*)

Tahap penerjemahan rancangan sistem ke dalam bentuk kode program.

4. Pengujian (*Testing*)

Tahap pengujian sistem untuk memastikan fungsi yang dibangun berjalan sesuai kebutuhan.

5. Pemeliharaan (*Maintenance*)

Tahap penerapan sistem kepada pengguna serta perbaikan apabila ditemukan kesalahan atau kebutuhan pengembangan lebih lanjut.

2.1.22 Kelebihan Metode *Waterfall*

Metode *Waterfall* memiliki beberapa kelebihan yang menjadikannya masih banyak digunakan dalam pengembangan perangkat lunak, terutama pada proyek yang memiliki kebutuhan yang jelas dan terdefinisi dengan baik. Adapun kelebihan Metode *Waterfall* adalah sebagai berikut:

1. Alur pengembangan yang terstruktur dan mudah dipahami
Setiap tahapan dilakukan secara berurutan mulai dari analisis kebutuhan hingga pemeliharaan sistem, sehingga memudahkan pengembang dalam memahami proses pengembangan yang sedang berlangsung.
2. Dokumentasi lebih lengkap dan sistematis
Setiap tahapan menghasilkan dokumen yang dapat digunakan sebagai acuan pada tahap berikutnya, sehingga mempermudah proses pengelolaan dan pemeliharaan sistem.
3. Mudah dalam pengelolaan proyek
Pembagian tahapan yang jelas memudahkan pengembang maupun manajer proyek dalam melakukan perencanaan, pengawasan, dan evaluasi terhadap proses pengembangan sistem.
4. Cocok untuk kebutuhan sistem yang stabil
Model *Waterfall* dinilai paling tepat untuk proyek yang persyaratan sistemnya telah ditetapkan secara jelas dari awal dan tidak banyak berubah seiring berjalannya proses pengembangan.

2.1.23 Kekurangan Metode *Waterfall*

Selain memiliki kelebihan, Metode *Waterfall* juga memiliki beberapa kekurangan yang perlu diperhatikan sebelum digunakan dalam pengembangan perangkat lunak. Adapun kekurangan Metode *Waterfall*

adalah sebagai berikut:

1. Metode ini kurang fleksibel dalam merespons perubahan kebutuhan yang muncul pasca tahap analisis, mengingat setiap fase harus dituntaskan secara berurutan sebelum melanjutkan ke fase berikutnya
2. Pengguna tidak dapat melihat sistem secara langsung pada tahap awal. Pengguna baru dapat mencoba sistem setelah proses implementasi dan pengujian selesai, sehingga risiko ketidaksesuaian kebutuhan pengguna menjadi lebih besar.
3. Kesalahan pada tahap awal dapat berdampak pada tahap berikutnya. Jika terjadi kesalahan dalam analisis kebutuhan atau perancangan sistem, maka kesalahan tersebut dapat mempengaruhi proses implementasi dan menyebabkan biaya perbaikan menjadi lebih besar.
4. Kurang fleksibel terhadap perubahan lingkungan bisnis. Metode *Waterfall* kurang sesuai untuk proyek yang memiliki kebutuhan dinamis karena proses pengembangannya tidak dirancang untuk menerima perubahan secara berulang selama proyek berlangsung.

2.1.24 Perbedaan Metode *Waterfall*, *Agile*, *Extreme Programming*

Tabel 2. 4 Perbandingan Metode

Aspek	<i>Waterfall</i>	<i>Agile</i>	<i>Extreme Programming (XP)</i>
Pendekatan	Berurutan (<i>Sequential</i>)	Iteratif dan <i>Incremental</i>	Iteratif dengan fokus kualitas kode
Perubahan Kebutuhan	Sulit dilakukan	Mudah dilakukan	Sangat mudah dilakukan
Keterlibatan Pengguna	Rendah	Tinggi	Sangat tinggi
Dokumentasi	Lengkap	Secukupnya	Secukupnya
Kecepatan	Relatif lambat	Cepat	Cepat

Aspek	<i>Waterfall</i>	<i>Agile</i>	<i>Extreme Programming (XP)</i>
Pengembangan			
Cocok Untuk	Kebutuhan stabil	Kebutuhan dinamis	Proyek yang membutuhkan perubahan cepat

Berdasarkan perbandingan Tabel 2.4, Metode *Waterfall* lebih sesuai digunakan pada proyek yang memiliki kebutuhan sistem yang jelas dan tidak banyak mengalami perubahan. Sementara itu, *Agile* dan *Extreme Programming* lebih cocok digunakan pada proyek yang membutuhkan fleksibilitas tinggi dan keterlibatan pengguna secara berkelanjutan.

2.1.25 Blackbox Testing

Blackbox testing merupakan metode pengujian perangkat lunak yang berfokus pada pengujian fungsi sistem tanpa melihat atau mengetahui struktur kode program di dalamnya. Pengujian ini dilakukan dengan cara memberikan berbagai *input* pada sistem yang kemudian mengamati apakah *output* yang di hasilkan sudah sesuai dengan yang di harapkan. Ichsanudin (2022).

2.1.26 User Acceptance Testing (UAT)

Salah satu metode pengujian perangkat lunak yang berfokus pada evaluasi sistem dari sudut pandang pengguna akhir. UAT bertujuan untuk memastikan bahwa sistem yang dikembangkan telah memenuhi kebutuhan dan harapan pengguna, serta dapat diintegrasikan secara mulus ke dalam alur kerja nyata pengguna. Menurut Zulkarnaini et al.,(2023), UAT yang dikombinasikan dengan pendekatan *Black Box Testing* terbukti efektif dalam menjembatani keterbatasan pengujian tradisional. Pendekatan ini memberikan penilaian fungsional yang lebih menyeluruh dan mampu mengidentifikasi area yang sebelumnya tidak terdeteksi. Melalui interaksi langsung antara pengguna akhir dan sistem, UAT dapat mengekstraksi

wawasan kritis terkait fitur tambahan atau perbaikan sistem, sehingga memperkuat keselarasan antara sistem dengan kebutuhan pengguna.

2.2 Penelitian Terdahulu

Penelitian yang dilakukan oleh Anwar Fu'adi (2022) menunjukkan bahwa penerapan Metode *Waterfall* pada sistem *Monitoring* PAUD berbasis web mampu membantu guru dalam menginput data *Monitoring* dan nilai siswa, namun sistem tersebut masih terfokus pada pengolahan data internal dan belum optimal dalam mengintegrasikan Informasi kepada orang tua secara *real-time*. Sementara itu, penelitian oleh Wandu Aprianto (2026) menggunakan pendekatan SDLC *Waterfall* untuk merancang sistem rekam medis dan *Monitoring* perkembangan anak pada layanan terapi wicara, dengan hasil yang mampu mengelola data anak, jadwal terapi, dan catatan terapi secara terstruktur. Penelitian oleh Narti Eka Putria, Arif Rahman Hakim, dan Yulia (2024) juga mengembangkan sistem *Monitoring* PAUD berbasis web dengan Metode *Waterfall*, namun perbedaannya terletak pada objek penelitian dan pengembangan fitur yang disesuaikan dengan kebutuhan pengguna pada studi kasus yang berbeda.

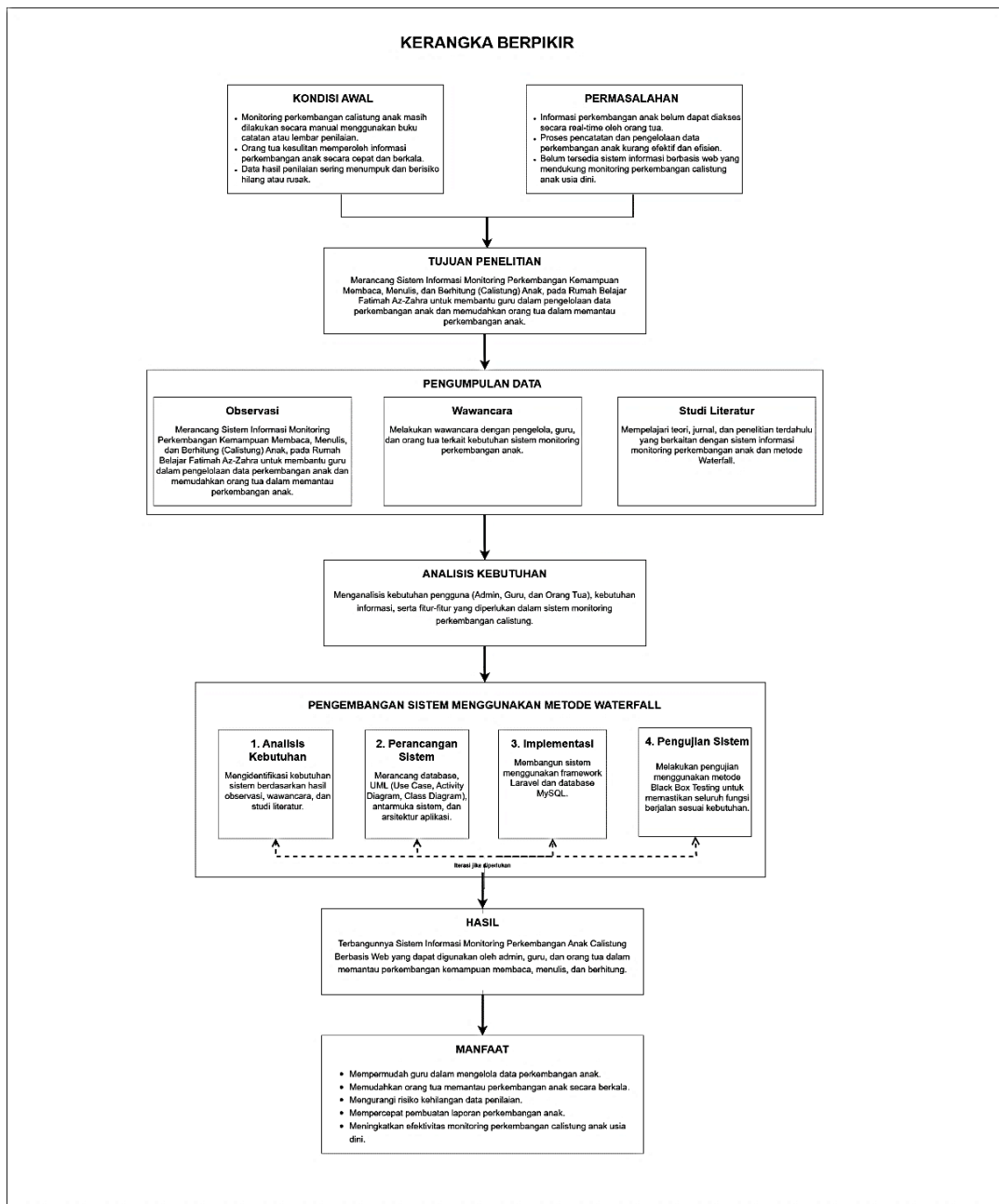
Perbedaan penelitian ini dengan penelitian-penelitian sebelumnya terletak pada objek dan konteks implementasi, yaitu merancang sistem Informasi *Monitoring* perkembangan anak calistung yang secara khusus disesuaikan dengan kebutuhan operasional Rumah Belajar Fatimah Az-zahra, di mana sistem berbasis digital semacam ini belum pernah diimplementasikan sebelumnya di lembaga tersebut, serta menekankan pada integrasi data yang terstruktur dan akses Informasi yang mudah bagi pengajar dan orang tua dalam satu *platform* berbasis web.

2.3 Kerangka Pemikiran

Penyusunan kerangka berpikir dalam penelitian ini didasarkan pada situasi nyata yang ada di Rumah Belajar Fatimah Az-zahra, di mana proses *Monitoring* perkembangan kemampuan membaca, menulis, dan berhitung (*calistung*) hingga kini masih dikerjakan dengan cara konvensional, yaitu memakai buku penghubung. Proses tersebut mengakibatkan pengelolaan

data belum berjalan secara optimal karena Informasi sulit diakses, pencatatan kurang efisien, serta terdapat risiko kehilangan maupun kerusakan data. Permasalahan tersebut menjadi dasar dalam perancangan sebuah sistem yang mampu mendukung proses *Monitoring* perkembangan anak secara lebih efektif.

Untuk mewujudkan tujuan tersebut, kebutuhan sistem diidentifikasi melalui kegiatan observasi, wawancara, dan studi literatur. Seluruh data yang diperoleh selanjutnya dianalisis sebagai dasar dalam menentukan kebutuhan pengguna dan merancang fitur-fitur yang diperlukan. Tahap pengembangan sistem menerapkan Metode *Waterfall* yang diterapkan melalui fase analisis kebutuhan, perancangan sistem, hingga tahap implementasi dengan memanfaatkan *framework Laravel* dan *MySQL* sebagai sistem manajemen basis data, serta pengujian melalui metode *Black Box Testing*. Hasil akhir penelitian ini berupa Sistem Informasi *Monitoring* perkembangan Anak Calistung berbasis web yang mampu membantu guru dalam mengelola data perkembangan anak secara lebih terstruktur sekaligus memudahkan orang tua memperoleh Informasi perkembangan anak secara berkala.



Gambar2. 3 Kerangka Berpikir