BAB II
TINJAUAN PUSTAKA

2.1 Rancang Bangun

Menurut (Surahmat, 2023) rancang bangun merupakan proses mengembangkan dan memperbaiki sistem atau aplikasi yang sudah ada ataupun belum ada dengan beberapa komponen yang digunakan dari hasil proses analisa sistem dimana mengharapkan dapat menghasilkan aplikasi yang berkualitas sesuai dengan kebutuhan yang diinginkan.

Menurut (Hasmia, Nirsal, 2022) Rancang bangun adalah suatu kegiatan yang bertujuan untuk merancang suatu sistem baru yang dapat memecahkan masalah yang dihadapi perusahaan yang diperoleh dari pemilihan alternatif sistem yang terbaik. Rancang bangun adalah program yang menentukan aktivitas pemrosesan informasi yang diperlukan untuk menyelesaikan tugas tertentu dari pengguna komputer.

Berdasarkan pernyataan di atas dapat disimpulkan bahwa rancang bangun merupakan sebuah proses dan kegiatan sistematis dalam merancang, mengembangkan, atau memperbaiki suatu sistem maupun aplikasi, baik yang benar-benar baru maupun modifikasi dari sistem yang sudah berjalan. Proses ini dilakukan berdasarkan hasil analisis mendalam serta pemilihan alternatif solusi terbaik guna memecahkan permasalahan yang dihadapi oleh pengguna atau institusi. Melalui tahapan tersebut, rancang bangun berfungsi untuk menentukan aktivitas pemrosesan informasi yang tepat, sehingga mampu menghasilkan sebuah aplikasi yang fungsional, berkualitas tinggi, dan relevan dengan kebutuhan yang diinginkan.

2.2 Sistem Informasi

Sistem informasi dapat didefinisikan sebagai suatu kesatuan yang terdiri atas berbagai elemen yang saling terintegrasi, meliputi sumber daya manusia, perangkat keras (*hardware*), perangkat lunak (*software*), infrastruktur jaringan telekomunikasi, hingga basis data. Seluruh komponen

tersebut berkolaborasi dalam memproses dan mendistribusikan data menjadi informasi yang bermakna. Lebih lanjut, sistem ini juga memfasilitasi adanya mekanisme umpan balik (*feedback*) sebagai alat kontrol evaluasi guna memastikan bahwa seluruh aktivitas operasional berjalan sejalan dengan tujuan spesifik yang ingin dicapai. (Kamal et al., 2023).

Sementara itu, Dalam perspektif para ahli, sistem informasi dipandang sebagai elemen fundamental yang menyusun arsitektur organisasi modern. Keberadaan sistem ini di dalam sebuah institusi memiliki peran krusial untuk mengakomodasi rutinitas pengolahan data transaksi harian serta mengoptimalkan kelancaran operasional. Lebih dari itu, sistem informasi juga berfungsi sebagai instrumen pendukung yang krusial dalam memfasilitasi aktivitas manajerial dan perumusan kebijakan strategis. (Purnama, 2024).

Dapat disimpulkan bahwa sistem informasi adalah gabungan dari berbagai komponen seperti manusia, perangkat lunak, dan perangkat keras yang bekerja sama untuk mengumpulkan, memproses, dan menyebarkan informasi. Tujuan utamanya adalah untuk mendukung berbagai aktivitas penting dalam sebuah organisasi, mulai dari operasi harian hingga pengambilan keputusan manajerial dan strategis, guna mencapai sasaran yang telah ditetapkan.

2.3 Aplikasi Mobile

Aplikasi mobile merupakan produk dari sistem komputasi bergerak, yang dirancang untuk memberikan kemampuan komputasi pada perangkat yang mudah dipindahkan secara fisik. Contoh umum dari perangkat ini meliputi *Personal Digital Asistant* (PDA), *smartphone*, dan ponsel. Sistem komputasi bergerak memudahkan pengguna untuk tetap terhubung dan menjalankan berbagai fungsi komputasi dimanapun mereka berada.

Dalam konteks ini, terminologi *mobile* merujuk pada perangkat komputasi berukuran ringkas yang dirancang agar mudah digenggam dan

dioperasikan. Karakteristik utama dari teknologi ini adalah tingkat portabilitasnya yang tinggi, sehingga memfasilitasi pengguna untuk membawanya bermobilitas dengan sangat praktis. Salah satu bentuk implementasi paling umum dari kategori perangkat ini adalah telepon pintar (*smartphone*), yang pada dasarnya merupakan sebuah perangkat multifungsi penunjang aktivitas harian. Perangkat *mobile* yang paling popular adalah perangkat yang menggunakan sistem operasi Android serta iOS yang mana merupakan sistem operasi terbanyak di dunia. Kedua adalah *computer* tablet. *Computer* tablet merupakan perangkat *mobile* yang tidak sepopuler *smartphone* walaupun pada perangkat tablet terdapat juga sistem operasi *Android* dan *IOS*.

Aplikasi mobile juga memiliki beberapa kelebihan dalam beberapa hal, seperti memberikan kemudahan penggunaanya misal dapat diakses oleh siapa saja, kapan saja dan dimana saja serta kemampuan dari aplikasi mobile ini yang mudah terhubung dengan internet (Setiawan et al., 2025).

2.4 Posyandu

Pos Pelayanan Terpadu, atau yang lebih dikenal dengan istilah Posyandu, merupakan sebuah fasilitas layanan yang dirancang untuk mempermudah masyarakat dalam menjangkau fasilitas kesehatan tingkat dasar. Kehadiran program ini memiliki target krusial, yakni menekan laju angka kematian pada balita secara efektif. Sebagai salah satu wujud nyata dari Upaya Kesehatan Bersumber Daya Masyarakat (UKBM), operasional Posyandu dikelola dengan prinsip partisipatif yakni dari, oleh, untuk, serta bersama masyarakat. Pendekatan ini menjadi langkah strategis untuk mengoptimalkan pemberdayaan warga sekaligus mendukung percepatan pembangunan kesehatan di lingkungan setempat. (Tarigan et al., 2021).

2.5 Posyandu Lengkeng

Posyandu Lengkeng merupakan pos pelayanan terpadu yang berada di RW 014 Kelurahan Harapan Baru Kecamatan Bekasi Utara, Kota Bekasi

yang menjadi objek pada penelitian ini. Posyandu Lengkeng memiliki visi dan misi nya sendiri yaitu :

1. Visi :

   Menjadikan masyarakat sejahtera dan mandiri.

2. Misi :

   1. Lebih mendapatkan pelayanan kesehatan masyarakat bagi warga sekitar.

   2. Meningkatkan kehadiran balita ke posyandu untuk tumbuh kembang.

   3. Menggalakan pemberiam ASI ekslusif.

   4. Meningkatkan kesadaran ibu untuk memeriksa kehamilan.

   5. Meningkatkan kesadaran masyarakat untuk memeriksa kesehatan secara berkala dan penyuluhan.

   6. Meningkatkan kesadaran masyarakat agar hidup sehat dan bersih.

Posyandu Lengkeng yang berlokasi di RW 014 Kelurahan Harapan Baru, Kecamatan Bekasi Utara memiliki komitmen kuat dalam mewujudkan masyarakat yang sejahtera dan mandiri. Visi tersebut diimplementasikan secara nyata melalui berbagai misi yang berfokus pada perluasan akses pelayanan kesehatan, pemantauan tumbuh kembang balita, pemenuhan gizi ibu dan anak, serta peningkatan kesadaran masyarakat luas akan pentingnya pemeriksaan kesehatan berkala dan penerapan pola hidup bersih serta sehat.

2.6 React Native

React Native adalah *framework cross-platform* yang memungkinkan pengembang membuat aplikasi *mobile* menggunakan bahasa pemrograman *JavaScript* tanpa mengurangi pengalaman penggunaannya. Dengan satu pengkodean, pengembang dapat membuat aplikasi yang berbasis *Android* dan *iOS* sekaligus. Perusahaan besar di seluruh dunia telah menggunakan framework ini (Rohman & Henny Dwi Bhakti, 2024).

React Native adalah framework pengembangan *hybrid mobile* yang dibuat menggunakan *library React* dan bahasa pemrograman *JavaScript*. Penggunaan *React Native* pada proses pengembangan aplikasi dimulai dengan versi 0.71.14, sedangkan *React* yang digunakan adalah versi 18.2.0. Penggunaan *React Native* disebabkan oleh alasan yang subjektif. Ini menawarkan pengalaman pengembangan yang lebih cepat dan lebih mudah dibandingkan dengan platform lain (Muhammad Abel Jollando et al., 2024).

Dapat disimpulkan *React Native* adalah framework pengembangan aplikasi *mobile hybrid* berbasis JavaScript yang sangat efisien karena memiliki kemampuan cross-platform (satu basis kode untuk Android dan iOS sekaligus). Framework ini menjadi pilihan utama dibanding bahasa *native* (seperti Kotlin atau Java) karena menawarkan pengalaman pengembangan yang lebih mudah dan ringan, namun tetap menjaga kualitas pengalaman pengguna, serta telah teruji keandalannya karena banyak digunakan oleh perusahaan besar.

2.7 JavaScript

JavaScript merupakan salah satu bahasa pemrograman yang paling dominan diimplementasikan dalam ekosistem pengembangan aplikasi, baik pada platform berbasis web maupun seluler (*mobile*). Bahasa ini memiliki fleksibilitas tinggi karena dapat diaplikasikan secara menyeluruh, baik pada sisi antarmuka pengguna (*frontend*) maupun sisi server (*backend*), serta didukung oleh komunitas pengembang yang sangat masif. Di samping itu, *JavaScript* dinilai mampu memberikan efisiensi yang lebih baik karena memiliki arsitektur yang lebih ringan jika dikomparasikan dengan beberapa bahasa pemrograman lainnya. Pada akhirnya, adopsi teknologi ini ditujukan untuk menyederhanakan dan mengakselerasi siklus pembangunan sebuah sistem (Muhammad Abel Jollando et al., 2024).

JavaScript merupakan sebuah bahasa pemrograman yang mengadopsi paradigma pemrograman berbasis prototipe (*prototype-based*

programming). Secara historis, teknologi ini sangat populer diimplementasikan pada lingkungan situs web untuk menangani berbagai fungsionalitas dan interaksi di sisi pengguna (*client-side*). Meski demikian, kapabilitas skrip ini kini telah berkembang pesat dan tidak hanya terbatas pada peramban web. *JavaScript* juga dapat dimanfaatkan untuk memberikan instruksi kontrol maupun mengeksekusi objek-objek yang tersemat (*embedded objects*) di dalam berbagai platform atau aplikasi eksternal lainnya (Wibawa & Naufal, 2023).

Berdasarkan uraian tersebut, dapat ditarik kesimpulan bahwa *JavaScript* adalah sebuah bahasa pemrograman berparadigma prototipe yang sangat mendominasi ekosistem pengembangan aplikasi, baik pada platform web maupun seluler (*mobile*). Tingginya tingkat adopsi teknologi ini didorong oleh fleksibilitasnya yang mampu mengakomodasi kebutuhan komputasi pada sisi antarmuka pengguna (*frontend*) maupun sisi server (*backend*). Selain ditopang oleh skala komunitas pengembang yang sangat masif, keunggulan utama *JavaScript* juga terletak pada kemampuannya dalam memfasilitasi proses pembangunan sistem yang jauh lebih ringan dan efisien apabila dikomparasikan dengan alternatif bahasa pemrograman lainnya. Keunggulan karakteristik *JavaScript* ini menjadi landasan utama bagi *React Native*, sebuah *framework* pengembangan aplikasi *mobile hybrid* berbasis pustaka React yang memungkinkan pembuatan aplikasi lintas *platform* (Android dan iOS) hanya dengan satu basis kode. Penggunaan *React Native* dinilai lebih menguntungkan dibandingkan pendekatan *native* menggunakan Java atau Kotlin karena proses pengembangannya yang jauh lebih mudah, cepat, dan ringan, namun tetap mampu menghasilkan aplikasi dengan pengalaman pengguna yang berkualitas.

2.8 Firebase

Firebase Firestore adalah salah satu layanan cloud *database NoSQL* dari Google yang memungkinkan penyimpanan dan sinkronisasi data secara *real-*

time untuk aplikasi *mobile* dan web. *Firestore* menawarkan fleksibilitas dalam struktur data dan kecepatan dalam proses baca-tulis data. *Firebase* mempermudah pengembangan aplikasi tanpa harus membangun infrastruktur *backend* kompleks (Ganesha, 2025).

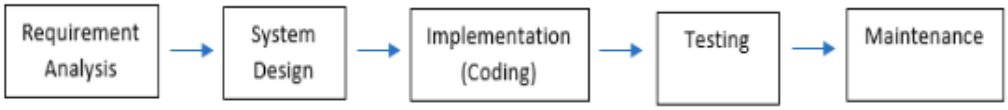
Firebase merupakan layanan API dari Google yang dirancang untuk sinkronisasi dan penyimpanan data lintas platform, termasuk Android, iOS, serta aplikasi berbasis web. Salah satu fitur unggulannya, yaitu *Firebase Realtime Database*, menawarkan mekanisme penyimpanan dan pengambilan data dengan tingkat keamanan serta kecepatan yang optimal. Keunggulan utamanya terletak pada kemampuan sinkronisasi otomatis, di mana setiap perubahan data pada basis data akan langsung diperbarui di sisi pengguna tanpa memerlukan pemanggilan ulang (*query*) secara manual. Hal ini dimungkinkan berkat dukungan pustaka (*library*) yang luas bagi platform *mobile* dan web. (Imbalo et al., 2022).

Berdasarkan uraian di atas dapat disimpulkan bahwa *firebase* merupakan layanan cloud *database NoSQL* dari Google yang memfasilitasi penyimpanan dan sinkronisasi data secara otomatis dan real-time untuk aplikasi berbasis *mobile* maupun web. Melalui fitur unggulannya seperti *Firestore dan Realtime Database*, platform ini menawarkan kecepatan, keamanan, serta fleksibilitas struktur data yang tinggi. Sistem ini mampu memperbarui informasi secara langsung ketika terjadi perubahan pada *database* tanpa memerlukan pemanggilan data secara berulang. Oleh karena itu, penggunaan *Firebase* dinilai sangat efisien karena mempermudah proses pengembangan aplikasi tanpa mengharuskan pengembang untuk membangun infrastruktur backend yang kompleks dari awal.

2.9 Waterfall

Model *Waterfall* dikenal sebagai pendekatan pengembangan perangkat lunak yang bersifat sistematis dan berurutan. Dalam kerangka SDLC, metode ini menginstruksikan alur pengerjaan yang dimulai dari identifikasi

kebutuhan pengguna, desain arsitektur, hingga proses *maintenance*. Karena sifatnya yang sederhana namun disiplin, setiap tahapan dalam siklus ini bertindak sebagai prasyarat bagi langkah selanjutnya, sehingga tidak memungkinkan adanya pengerjaan yang tumpang tindih sebelum tahap terdahulu diselesaikan (Haniva et al., 2023).



Gambar 2.1 Tahapan-tahapan metodologi Waterfall

Sumber : (Haniva et al., 2023)

Berikut ini merupakan tahapan dari metode *Waterfall*, yakni:

a) Perencanaan Konsep (*Requirement Analysis*)

Tahapan ini bertujuan untuk mengidentifikasi kebutuhan serta ekspektasi pengguna secara akurat. Proses pengumpulan data dilakukan melalui wawancara langsung dengan para pemangku kepentingan guna mendapatkan informasi yang mendalam. Hasil dari proses ini adalah dokumen analisis dan spesifikasi kebutuhan sistem, yang nantinya akan berfungsi sebagai panduan utama dalam seluruh rangkaian pengembangan perangkat lunak.

b) Pemodelan sistem (*System Design*)

Pada tahap ini, seluruh hasil analisis kebutuhan yang telah disusun sebelumnya ditransformasikan ke dalam bentuk rancangan atau desain sistem yang komprehensif. Perancangan ini berfungsi sebagai cetak biru (blueprint) teknis yang mendetail, sehingga memberikan instruksi yang jelas bagi pengembang saat memasuki proses pengodean nantinya.

c) Implementasi

Dalam fase ini, rancangan sistem yang telah disiapkan mulai direalisasikan melalui proses pengodean atau coding. Tujuannya

adalah untuk menerjemahkan seluruh desain teknis ke dalam bahasa pemrograman, sehingga arsitektur tersebut dapat diwujudkan menjadi sebuah aplikasi yang fungsional.

d) Pengujian

Setelah proses pengembangan selesai, sistem memasuki tahap evaluasi guna menguji kinerja dan tingkat optimalitasnya secara menyeluruh. Pengujian ini dilakukan untuk memvalidasi apakah seluruh fitur yang telah dibangun sudah berfungsi secara tepat dan sesuai dengan standar kebutuhan yang ditetapkan sebelumnya.

e) Pemeliharaan

Fase pemeliharaan ini dilaksanakan sebagai langkah antisipasi dan perbaikan jika ditemukan kendala teknis atau kerusakan pada sistem di kemudian hari. Selain untuk menangani gangguan, tahapan ini juga berfungsi untuk menjaga stabilitas aplikasi agar tetap beroperasi secara optimal dalam jangka panjang.

2.9.1 Kelebihan *Waterfall*

Model *Waterfall* menawarkan sejumlah keunggulan, di antaranya adalah alur kerja yang sederhana dan mudah dipahami karena setiap fasenya memiliki batasan yang jelas serta terstruktur. Selain itu, metodologi ini menghasilkan dokumentasi yang sangat lengkap di setiap tahapannya, sehingga memudahkan tim dalam melakukan proses pemeliharaan sistem. Karakteristik tersebut membuat metode ini sangat ideal untuk diimplementasikan pada proyek dengan skala kecil dan kebutuhan yang stabil.

2.9.2 Kekurangan *Waterfall*

Model *Waterfall* ini memiliki keterbatasan pada sifatnya yang cenderung kaku, di mana kekeliruan pada fase terdahulu mengharuskan pengembang untuk mengulang proses dari tahap paling awal. Selain itu, keberhasilan metode ini sangat bergantung

pada ketepatan spesifikasi kebutuhan yang disusun di awal penelitian. Hal tersebut menyebabkan metode *Waterfall* kurang fleksibel jika terdapat perubahan mendadak dan kurang efektif untuk kebutuhan pengembangan sistem yang memerlukan durasi waktu yang singkat.

2.10 *Unified Modeling Language (UML)*

Unified Modeling Language atau *UML* merupakan standar pemodelan visual yang umum digunakan dalam merancang dan membangun perangkat lunak berbasis objek. Sebagai bahasa pemodelan yang telah terstandarisasi, *UML* berfungsi sebagai media untuk menyusun cetak biru (*blueprint*) sebuah sistem secara terperinci. Di dalam UML, pengembang dapat mendefinisikan berbagai elemen penting, mulai dari alur proses bisnis hingga struktur kelas dalam bahasa pemrograman tertentu. Untuk menggambarkan arsitektur sistem tersebut, terdapat beberapa diagram UML yang sering diaplikasikan, antara lain: (Ramdany, 2024).

2.10.1 *Use Case*

Use Case diagram memberikan gambaran mengenai fungsionalitas yang diharapkan dari sebuah sistem melalui representasi interaksi antara aktor dan sistem tersebut. Diagram ini berfungsi untuk memetakan berbagai jenis interaksi yang dilakukan oleh pengguna, sehingga sangat membantu dalam merancang alur sistem secara lebih terstruktur. Di dalam pemodelan ini, aktor didefinisikan sebagai entitas, baik berupa manusia maupun sistem lain, yang memiliki peran atau melakukan aktivitas tertentu di dalam sistem yang dikembangkan.

2.10.4 *Class Diagram*

Class Diagram merupakan representasi visual yang menggambarkan struktur sistem melalui kumpulan kelas, paket, dan objek. Diagram ini merinci setiap elemen kelas beserta atributnya, serta menjelaskan hubungan keterkaitan antar kelas seperti pewarisan (*inheritance*), asosiasi, dan *relasi* lainnya. Dalam pengembangan sistem berorientasi objek, diagram ini berfungsi sebagai model data yang mendefinisikan bagaimana informasi dan struktur internal sistem saling terhubung.

2.10.2 *Activity Diagram*

Activity diagram merupakan diagram yang digunakan untuk menggambarkan alur kerja atau perilaku sistem. Diagram ini bersifat intuitif dan fleksibel, memudahkan perancangan logika internal dari operasi yang kompleks. Oleh karena itu, activity diagram sering digunakan untuk merancang sistem perangkat lunak dan perangkat keras pada tahap awal desain.

2.10.3 *Sequence Diagram*

Sequence Diagram berfungsi untuk menggambarkan interaksi antar objek, baik di dalam maupun di sekitar sistem, melalui pertukaran pesan (*message*) yang disusun berdasarkan urutan waktu. Sebagai salah satu diagram UML, model ini memperlihatkan secara detail bagaimana setiap komponen saling berkomunikasi secara kronologis. Pada umumnya, diagram ini diterapkan pada fase perancangan teknis untuk menyusun alur komunikasi antar proses yang lebih spesifik dan formal.

2.11 Analisis PIECES

Analisis PIECES merupakan sebuah teknik evaluasi yang digunakan untuk membedah kinerja sistem yang sedang berjalan sebagai dasar dalam merancang sistem baru yang lebih baik. Melalui metode ini, setiap permasalahan yang ditemukan akan dipetakan ke dalam enam kategori

utama, yaitu *performance* (kinerja), *information* (informasi), *economics* (ekonomi), *control* (pengendalian), *efficiency* (efisiensi), dan *service* (layanan). Berikut adalah penjelasan mendalam mengenai keenam kategori dalam analisis PIECES tersebut:

1. *Performance*

   Secara umum, kategori ini menyoroti perbandingan antara volume pekerjaan yang harus diselesaikan dengan durasi waktu yang dihabiskan untuk melakukan aktivitas tersebut. Fokus utamanya adalah mengevaluasi sejauh mana sistem mampu mengelola beban kerja secara produktif tanpa memerlukan waktu yang berlebihan.

2. *Information*

   Kategori informasi berfokus pada mekanisme aliran data, baik yang masuk maupun keluar dalam sebuah proses, serta mencakup metode penyimpanan data tersebut. Evaluasi pada poin ini bertujuan untuk memastikan bahwa data dikelola dengan baik sehingga informasi yang dihasilkan tetap akurat dan mudah diakses saat dibutuhkan.

3. *Economics*

   Kategori ini mengevaluasi sejauh mana sebuah sistem mampu mengefisienkan biaya operasional sekaligus mendorong peningkatan nilai tambah serta keuntungan finansial yang maksimal bagi sebuah organisasi.

4. *Control*

   Kategori ini menyoroti penanganan berbagai kendala yang berkaitan dengan manajemen hak akses dan perlindungan keamanan data. Selain itu, evaluasi pada kategori ini juga bertujuan untuk menyederhanakan alur prosedur operasional agar sistem terhindar dari rantai birokrasi yang berbelit-belit dan tidak efisien.

5. *Efficiency*

Kategori ini berfokus pada evaluasi operasional sistem, baik tugas tersebut dieksekusi oleh sumber daya manusia maupun mesin. Penilaian ini bertujuan untuk mengidentifikasi apakah suatu alur kerja mengonsumsi sumber daya secara berlebihan, guna mencegah terjadinya pemborosan pada alokasi waktu, tenaga, hingga material.

6. *Service*

Kategori ini mengevaluasi terhadap tingkat kemudahan operasional dari suatu proses kerja. Selain itu, aspek ini juga ditujukan untuk memastikan bahwa alur tersebut benar-benar mampu memberikan hasil akhir (*output*) yang relevan dan sesuai dengan target yang telah ditetapkan.

Pengkategorian masalah yang dilakukan bertujuan untuk mendapatkan wawasan atau pandangan dari suatu sistem atau proses bisnis yang lebih baik.

2.12 *Black Box Testing*

Pendekatan *Black Box Testing* diaplikasikan untuk mengevaluasi fungsionalitas perangkat lunak tanpa melibatkan analisis terhadap struktur internal atau kode programnya. Fokus utama metode ini terletak pada verifikasi hasil luaran (*output*) yang dihasilkan dari sekumpulan data masukan (*input*) tertentu guna memastikan sistem beroperasi sesuai spesifikasi kebutuhan. Melalui pengujian ini, setiap formulir dan proses diuji secara langsung untuk mendeteksi potensi kesalahan atau *bug*. Langkah perbaikan yang dilakukan pasca-pengujian bertujuan untuk menjamin bahwa sistem berada dalam kondisi stabil dan layak untuk diimplementasikan (Ginting & Lubis, 2024).

Senada dengan hal tersebut, evaluasi fungsionalitas perangkat lunak melalui metode *Black Box testing* berfokus pada analisis ko*relasi* antara data masukan dan hasil luaran tanpa melibatkan pemeriksaan terhadap arsitektur kode internal. Teknik ini umumnya diterapkan pada fase final siklus

pengembangan guna memvalidasi efektivitas serta kinerja sistem secara menyeluruh sebelum diimplementasikan. (Nurfathullah, 2024).

Dapat disimpulkan, *Black box testing* merupakan sebuah pendekatan pengujian perangkat lunak yang menitikberatkan pada evaluasi fungsionalitas eksternal. Dalam pelaksanaannya, proses evaluasi ini sama sekali tidak melibatkan pemeriksaan terhadap arsitektur internal maupun susunan kode sumber (*source code*) dari suatu program. Pengujian murni dilakukan dengan cara memasukkan berbagai skenario data uji (*input*) untuk memvalidasi apakah respons yang dimunculkan (*output*) sudah selaras dengan spesifikasi kebutuhan yang ditetapkan di awal. Pada umumnya, metode ini diimplementasikan pada fase akhir pengembangan guna mengidentifikasi potensi kecacatan sistem (*bug*), menjamin kualitas performa secara menyeluruh, serta memastikan bahwa aplikasi tersebut benar-benar layak untuk dioperasikan oleh pengguna.

2.13 *Penelitian Terdahulu*

Terdapat beberapa penelitian yang sudah pernah dilakukan berhubungan dengan penelitian ini adalah:

Tabel 2.1 Penelitian Terdahulu

No.	:	1
Penulis	:	(Usin et al., 2025)
Judul	:	Rancang Bangun Sistem Manajemen Posyandu Seroja di Desa Saka Mangkahai Berbasis Android (SINTA 5)
Metode	:	*Waterfall*
Hasil	:	Hasil penelitian tersebut adalah aplikasi Sistem Manajemen Posyandu Seroja berbasis Android berhasil dikembangkan dan dinilai dapat memberikan akses yang lebih mudah dan cepat terhadap informasi kesehatan serta layanan Posyandu Seroja, sekaligus meningkatkan efisiensi pengelolaan kegiatan posyandu di Desa Saka Mangkahai. Aplikasi ini juga mencakup fitur seperti jadwal posyandu, edukasi,

		pesan/konsultasi, riwayat kunjungan, riwayat imunisasi, laporan kegiatan, dan notifikasi.
Link	:	https://doi.org/10.47111/jointecoms.v5i4.25346
No.	:	2
Penulis	:	(Rahmi et al., 2025)
Judul	:	RANCANG BANGUN WEBSITE SISTEM INFROMASI POSYANDU WIJAYA KUSUMA PADA DESA RAWANG SARI (SINTA 4)
Metode	:	*Waterfall*
Hasil	:	Hasil penelitian tersebut adalah pengembangan website Sistem Informasi Posyandu Wijaya Kusuma yang sesuai dengan proses bisnis kegiatan posyandu dan memungkinkan pencatatan kegiatan dilakukan secara digital oleh kader dan bidan menggunakan komputer. Selain itu, sistem ini juga membantu meminimalkan kesalahan pencatatan, serta mempermudah rekapitulasi dan pencarian data, sehingga waktu yang dibutuhkan menjadi lebih singkat
Link	:	https://doi.org/10.51876/simtek.v10i1.1527
No.	:	3
Penulis		(Maulana et al., 2025)
Judul		Perancangan Aplikasi Mobile Posyandu Sehat Untuk Pelayanan Kunjungan dan Imunisasi Anak (SINTA 4)
Metode		*Waterfall*
Hasil		Penelitian ini berhasil dirancang sebagai pengganti metode manual, sehingga mempermudah pengelolaan data dan akses informasi di posyandu. Sistem ini memungkinkan kader mengelola data secara digital dan memudahkan akses informasi terbaru bagi kader dan

		masyarakat. Hasil pengujian blackbox menunjukkan seluruh fitur berjalan dengan baik, dan pengujian usability testing dengan 6 responden menghasilkan rata-rata 95%, yang menandakan sistem sangat layak digunakan.
Link		https://doi.org/10.36040/jati.v9i1.12500
No.		4
Penulis		(Nikmah et al., 2026)
Judul		RANCANG BANGUN SISTEM INFORMASI POSYANDU PENCATATAN DATA BALITA DAN EDUKASI KESEHATAN BERBASIS WEBSITE (STUDI KASUS : POSYANDU MARDIRAHAYU 2 DESA CANGKRING REMBANG) (SINTA 4)
Metode		*Waterfall*
Hasil		Hasil penelitian tersebut menunjukkan bahwa sistem informasi Posyandu berbasis website berhasil dibangun dan berjalan dengan baik sesuai rencana. Semua fitur utama telah diuji dan sistem mampu merespons perintah dengan tepat tanpa kendala teknis, sehingga membantu kader mencatat data dengan lebih akurat dan mempercepat penyampaian informasi jadwal serta artikel kesehatan kepada orang tua melalui WhatsApp.
Link		https://doi.org/10.36040/jati.v10i2.17850
No.		5
Penulis		(Pratrian et al., 2026)
Judul		Development of Web and Mobile Health Services Information System Using Waterfall Method (SINTA 3)
Metode		*Waterfall*
Hasil		Aplikasi dan *website* yang bernama SILAKES berhasil diimplementasikan sebagai sistem layanan kesehatan web dan mobile

		dengan metode Waterfall, dan sistem ini meningkatkan efisiensi rumah sakit serta kualitas pelayanan. Pasien dapat mengakses jadwal dokter, antrean online, konsultasi, ketersediaan stok darah, dan survei kepuasan kapan saja, sementara staf terbantu dengan pengelolaan data yang lebih cepat dan alur kerja yang lebih efisien.
Link		https://doi.org/10.31294/co-science.v6i1.10072