BAB II
KAJIAN LITERATUR

2.1 Sistem Informasi

Sistem informasi merupakan suatu kesatuan komponen yang saling terintegrasi untuk mengumpulkan, mengelola, menyimpan, dan menyajikan informasi guna mendukung pemrosesan operasional serta pengambilan keputusan dalam suatu organisasi. Keberadaan sistem informasi menjadi penting ketika suatu instansi harus mengelola data dalam jumlah besar secara efisien dan akurat. Dalam dunia pendidikan, misalnya, sistem informasi akademik digunakan secara digital agar lebih terkordinasi dan mudah untuk diakses oleh pengguna [8].

Penerapan sistem informasi dirancang tidak hanya untuk mempermudah pekerjaan pengguna internal, tetapi juga untuk meningkatkan kualitas pelayanan kepada pihak eksternal seperti pelanggan atau masyarakat. Hal ini sejalan dengan hasil penelitian yang menunjukan bahwa sistem informasi manajemen mampu memberikan dampak signifikan terhadap peningkatan efisiensi proses bisnis dengan cara mengotomatisasi alur kerja, mempercepat arus informasi, serta meminimalkan terjadinya kesalahan pencatatan manual [9]. Dengan demikian sistem informasi berperan penting dalam menciptakan proses kerja yang lebih efektif, cepat, dan terukur.

Secara umum, implementasi sistem informasi dilakukan melalui platform berbasis *web* atau aplikasi *mobile*, tergantung pada kebutuhan organisasi dan pengguna akhir. Dalam era digital saat ini, sistem informasi semakin banyak diterapkan pada berbagai sektor seperti Pendidikan, bisnis, kesehatan, dan layanan publik guna memberikan kemudahan akses *real-time* dari berbagai lokasi. Oleh karena itu, sistem informasi menjadi landasan utama dalam pengembangan aplikasi pemesanan lapangan secara digital yang bertujuan untuk meningkatkan efisiensi, kecepatan, serta akurasi dalam pengelolaan data penyewaan dan penjadwalan.

2.2 Aplikasi Mobile

Aplikasi mobile merupakan perangkat lunak yang dirancang untuk berjalan pada perangkat bergerak seperti *smartphone* atau *tablet*, dengan tujuan memberikan akses layanan secara cepat, praktis, dan responsif sesuai kebutuhan pengguna. Kehadiran aplikasi *mobile* memungkinkan aktivitas digital dilakukan kapan saja dan dimana saja, sehingga memberikan fleksibilitas bagi pengguna dalam mengakses informasi maupun melakukan transaksi secara *real-time*. Pada penelitian sebelumnya, aplikasi *mobile* digunakan sebagai

sarana layanan pemesanan lapangan olahraga untuk meningkatkan kenyamanan pengguna dalam melakukan pemesanan lapangan tanpa harus datang langsung ke lokasi lapangan. Hal ini menunjukan bahwa aplikasi *mobile* memiliki peran penting dalam mendukung transformasi digital layanan berbasis kebutuhan masyarakat modern [10].

Penggunaan aplikasi *mobile* dalam berbagai bidang layanan publik maupun komersial terbukti mampu meningkatkan efektivitas dan efisiensi sistem yang sebelumnya dilakukan secara manual. Dalam konteks pengelolaan informasi akademik, aplikasi *mobile* dikembangkan untuk memberikan kemudahan akses data serta meningkatkan pengalaman pengguna melalui antarmuka yang interaktif. Selain itu, penerapan aplikasi *mobile* dalam sistem layanan memungkinkan pengguna untuk melakukan aktivitas seperti pemesanan, pelacakan informasi, dan validasi data secara digital dengan lebih cepat dan terstruktur. Kemampuan aplikasi *mobile* dalam mendukung mobilitas dan konektivitas menjadikannya solusi ideal dalam mendukung proses digitalisasi layanan [11].

Proses pengembangan aplikasi *mobile* umumnya mengikuti tahapan yang terstruktur agar aplikasi yang dihasilkan sesuai dengan kebutuhan pengguna dan berjalan secara optimal. Tahapan pengembangan tersebut antara lain meliputi:

1. Analisis Kebutuhan (*Requirement Analysis*): Pada tahap ini dilakukan identifikasi kebutuhan pengguna serta tujuan aplikasi. Kebutuhan yang diperoleh digunakan sebagai dasar dalam penentuan fitur utama dan fungsionalitas aplikasi agar dapat menyelesaikan permasalahan pengguna secara efektif.
2. Perancangan Antarmuka (*User Interface Design*): Tahap ini merupakan perancangan struktur visual aplikasi seperti wireframe atau prototipe awal untuk menentukan tata letak elemen, ikon, tombol, serta alur navigasi aplikasi. Tujuannya adalah menciptakan antarmuka yang mudah dipahami dan nyaman digunakan.
3. Pengembangan Fitur (*Coding/Implementation*): Pada tahap ini pengembang mulai menuliskan kode program sesuai desain yang telah dibuat. Seluruh fungsi utama dan pendukung diimplementasikan menggunakan *framework* pengembangan aplikasi *mobile*.
4. Pengujian Sistem (*Testing*): Tahap ini bertujuan untuk memastikan seluruh fitur berjalan dengan baik sesuai kebutuhan pengguna. Pengujian dilakukan untuk menemukan potensi kesalahan (*bug*) serta mengevaluasi kestabilan dan responsivitas aplikasi.

Dengan mengikuti tahapan tersebut, aplikasi *mobile* dapat dikembangkan secara sistematis sehingga mampu memberikan pengalaman penggunaan (*user experience*) yang baik serta meningkatkan efektivitas layanan digital secara keseluruhan [10][11].

2.3 Database

Database merupakan komponen vital dalam sistem informasi karena befungsi sebagai tempat penyimpanan, pengelolaan, serta pengambilan data yang dibutuhkan oleh sistem secara efisien dan terstruktur, dalam konteks pengembangan aplikasi *mobile*, khusunya layanan pemesanan lapangan secara daring, peran *database* menjadi krusial untuk mencatat informasi pengguna, jadwal pemesanan, notifikasi, hingga riwayat transaksi. Dengan menggunakan *database*, informasi dapat diakses secara *real-time* dan akurat, sehingga mendukung pengambilan keputusan yang cepat dan tepat [12].

Secara umum, database terbagi menjadi dua kategori. Yaitu database relasional (SQL) dan non-relasional (NoSQL). SQL seperti MySQL atau PostgreSQL menggunakan model tabel dan memiliki struktur skema yang baku, sehingga sangat cocok untuk sistem yang memerlukan integritas data tinggi dan hubungan kompleks antar entitas. Sebaliknya, NoSQL seperti *Firebase Realtime Database* atau MongoDB menawarkan fleksibilitas lebih tinggi dalam menyimpan data yang tidak memiliki struktur tetap. NoSQL biasanya digunakan pada aplikasi yang membutuhkan performa tinggi, skalabilitas, dan akses data secara *real-time* [13].

Firebase Realtime Database, sebagai salah satu platform NoSQL, telah banyak digunakan dalam pengembangan aplikasi *mobile* karena kemampuanya dalam melakukan sinkronisasi data secara langsung antar pengguna. Selain itu, *Firebase* mendukung fitur autentikasi pengguna, pengelolaan akses data berbasis aturan, dan mudah diintegrasikan dengan *framework* seperti flutter. Platform ini sangat sesuai untuk sistem pemesanan yang membutuhkan ketersediaan data secara instan, seperti pengecekan jadwal atau verifikasi booking secara langsung[14].

Berikut merupakan kelebihan dari masing-masing jenis database:

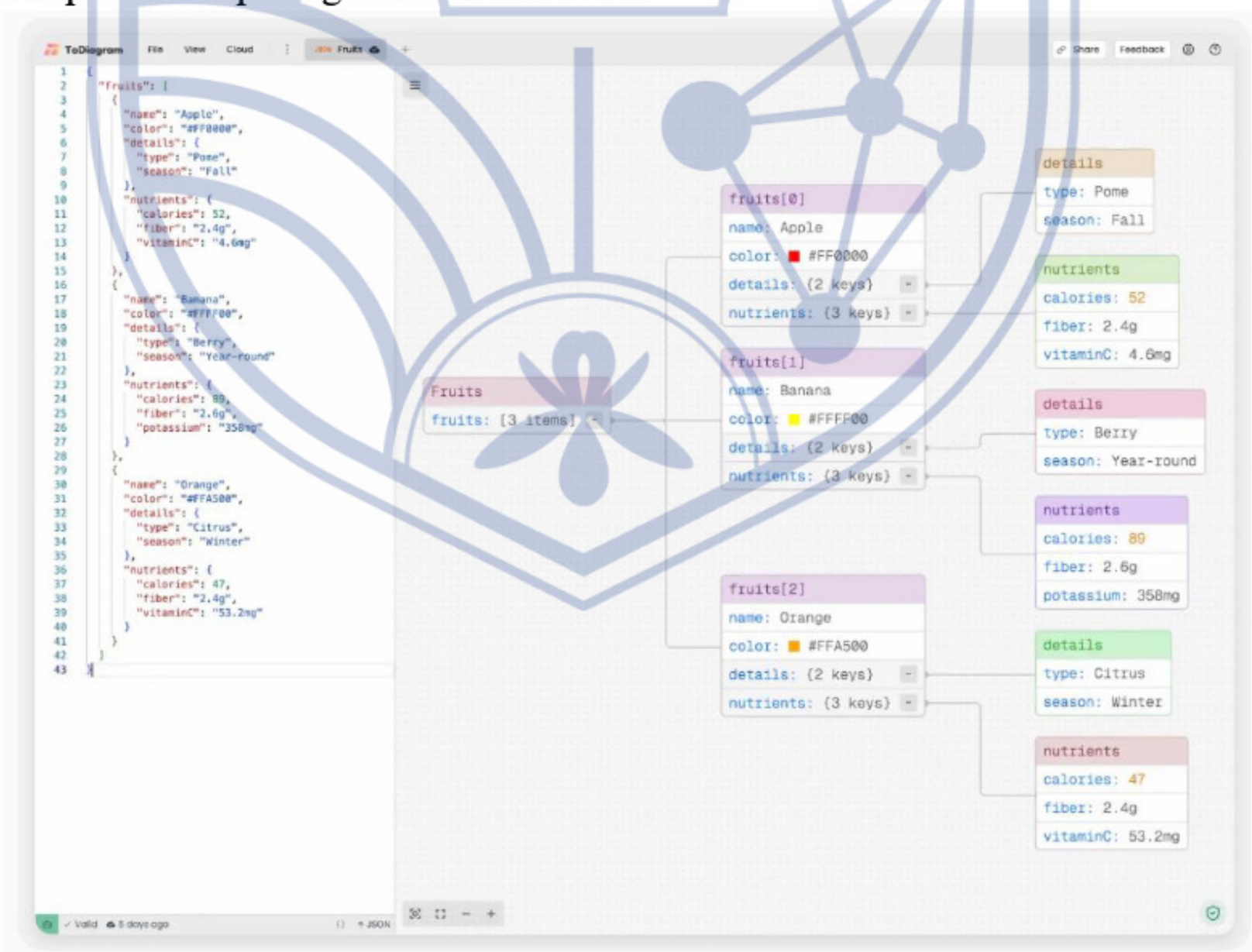
1. Kelebihan SQL:
    a. Mendukung transaksi yang konsisten dan terjamin dengan prinsip ACID
    b. Cocok untuk data yang memiliki hubungan antar table secara kompleks.
    c. Menyediakan query language standar (SQL) yang kuat dan terstruktur.

2. Kelebihan NoSQL (*Firebase*):
    a. Mendukung sinkronisasi data secara *real-time* antar perangkat pengguna
    b. Fleksibel dalam penyimpanan data semi-terstruktur maupun tidak terstruktur.
    c. Skalabilitas tinggi, cocok untuk aplikasi *mobile* dengan jumlah pengguna besar.

Dengan mempertimbangkan kebutuhan aplikasi pemesanan lapangan yang bersifat dinamis, fleksibel, dan membutuhkan akses data real-time, maka database jenis NoSQL seperti *Firebase* menjadi Solusi yang lebih tepat dibandingkan SQL dalam konteks pengembangan aplikasi ini.

2.3.1 Firebase Realtime Database

Firebase Realtime Database merupakan layanan basis data berbasis cloud yang dikembangkan oleh Google dan termasuk dalam kategori database NoSQL. Teknologi ini memungkinkan aplikasi untuk menyimpan, mengelola, serta melakukan sinkronisasi data secara langsung antara server dan perangkat pengguna secara real-time. Dengan kemampuan tersebut, Firebase Realtime Database banyak digunakan dalam pengembangan aplikasi mobile karena dapat menampilkan data terbaru secara otomatis tanpa perlu melakukan pemanggilan data secara berulang. Contoh struktur dasar JSON tree pada Firebase Realtime Database dapat dilihat pada gambar di bawah ini.



Gambar 2.1 JSON tree

Firebase Realtime Database menyimpan data dalam bentuk struktur JSON tree, sehingga data tidak disimpan dalam bentuk tabel seperti pada database relasional. Struktur

ini memberikan fleksibilitas dalam pengelolaan data dan memudahkan pengembang dalam menyimpan data yang bersifat dinamis. Selain itu, Firebase menyediakan berbagai library yang mendukung berbagai platform seperti Android, iOS, dan web sehingga memudahkan integrasi sistem pada aplikasi mobile [15].

Selain Firebase Realtime Database, Google juga menyediakan layanan database lain yaitu Firebase Firestore, yang merupakan database NoSQL berbasis cloud dengan kemampuan penyimpanan dan pengelolaan data secara real-time. Firestore menawarkan struktur data yang fleksibel, kemampuan query yang efisien, serta kemudahan integrasi dengan framework pengembangan aplikasi mobile seperti Flutter. Dengan fitur tersebut, Firestore sangat cocok digunakan pada aplikasi yang membutuhkan pembaruan data secara cepat dan akurat.

Beberapa penelitian menunjukkan bahwa penggunaan Firebase sebagai database pada aplikasi mobile dapat meningkatkan efisiensi pengelolaan data karena memiliki kemampuan sinkronisasi otomatis antar perangkat pengguna. Selain itu, penggunaan database berbasis cloud juga memungkinkan aplikasi untuk diakses secara online kapan saja dan di mana saja, sehingga memudahkan pengguna dalam memperoleh informasi secara cepat dan real-time [16].

Berdasarkan keunggulan tersebut, Firebase Realtime Database menjadi salah satu solusi yang efektif dalam pengembangan aplikasi mobile yang membutuhkan pengelolaan data secara dinamis, akses data yang cepat, serta kemampuan sinkronisasi data secara langsung antar pengguna.

2.3.2 Struktur Firebase Realtime Database

Firebase Realtime Database merupakan salah satu jenis basis data NoSQL yang digunakan untuk menyimpan dan mengelola data pada aplikasi secara online. Berbeda dengan database relasional yang menggunakan tabel dan relasi antar tabel, database NoSQL tidak memerlukan definisi struktur tabel terlebih dahulu sehingga lebih fleksibel dalam penyimpanan data. Database NoSQL memungkinkan penyimpanan berbagai model data seperti dokumen, graph, maupun key-value sehingga dapat menyesuaikan kebutuhan aplikasi yang memiliki struktur data dinamis [17].

Firebase Realtime Database menyimpan data dalam bentuk struktur JSON (JavaScript Object Notation) yang tersusun secara hierarki. Struktur ini terdiri dari beberapa bagian yang saling terhubung dalam bentuk child dan property yang menyimpan nilai data

tertentu. Dengan struktur tersebut, data dapat disimpan dan diakses secara terorganisir sesuai kebutuhan aplikasi yang sedang dikembangkan.

Setiap child pada struktur JSON dapat memiliki property yang berisi berbagai informasi yang diperlukan oleh sistem, seperti data pengguna, data pemesanan, maupun informasi transaksi. Hubungan antar child dalam struktur database ini memungkinkan aplikasi untuk mengakses serta mengelola data secara efisien sesuai dengan proses yang berjalan dalam sistem. Dengan menggunakan pendekatan struktur data berbasis hierarki tersebut, Firebase Realtime Database dapat mempermudah proses pengelolaan dan pengambilan data pada aplikasi. Struktur database yang fleksibel ini juga memungkinkan pengembang untuk menyesuaikan model data dengan kebutuhan sistem sehingga mendukung pengembangan aplikasi yang dinamis dan scalable [18].

2.4 User Interface

User interface (UI) merupakan bagian penting dari sistem aplikasi yang berfungsi sebagai jembatan interaksi antara pengguna dan sistem. UI dirancang untuk menyampaikan informasi dan memfasilitasi Tindakan pengguna melalui tampilan visul yang intuitif. Tujuan utama dari UI adalah menciptakan pengalaman penggunaan yang efisien dan menyenangkan. Studi oleh Ahmad et al. Menunjukan bahwa desain UI harus berorientasi pada kebutuhan pengguna agar interaksi menjadi efektif [19].

Dalam proses perancangan, penting untuk memperhatikan siapa pengguna yang akan menggunakan aplikasi, apa fitur yang paling sering digunakan, dan mengapa tata letak serta navigasi perlu dibuat sederhana mungkin. Berdasarkan analisis terhadap aplikasi Ipusnasm pradnyaswari. Menekankan bahwa antarmuka pengguna yang dirancang secara konsisten dapat meningkatkan pemahaman dan mengurangi kesalahan input [20]. Ciri-ciri UI yang baik meliputi:

1. Konsistensi tampilan dan navigasi.
2. Keterbacaan teks dan ikon yang jelas.
3. Navigasi yang sederhana dan mudah dipahami.

Desain UI juga sangat dipengaruhi oleh bagaimana pengguna berinteraksi dengan aplikasi. Pendekatan User Centered Design (UCD) digunakan untuk merancang antarmuka berbasis perilaku pengguna dengan mempertimbangkan kenyamanan dan efisiensi [21]. Tahapan UCD mencakup observasi pengguna, pengembangan protitpe, dan pengujian berulang untuk memastikan bahwa UI benar-benar sesuai dengan kebutuhan pengguna.

Kapan dan dimana UI menjadi elemen penting adalah ketika aplikasi mulai digunakan dalam skenario nyata oleh pelanggan dan pengelola. UI yang buruk dapat menurunkan kualitas layanan dan menimbulkan kebingungan, sedangkan UI yang dirancang dengan baik dapat meningkatkan kepuasan pengguna dan mempercepat proses interaksi dengan sistem. Oleh karena itu, dalam aplikasi pemesanan lapangan ini, UI berperan besar dalam memastikan bahwa pengguna dapat melakukan pemesanan secara mudah, cepat, dan tanpa hambatan.

2.5 Booking/Pemesanan

Sistem *booking* atau pemesanan merupakan proses awal yang penting dalam layanan penyewaan, termasuk pada layanan pemesanan lapangan olahraga. *Booking* berasal dari bahasa Inggris yang berarti memesan atau melakukan reservasi terlebih dahulu sebelum menggunakan layanan tertentu. Dalam konteks digital, *booking* dilakukan secara daring (*online*), yang memberikan kemudahan akses kapan saja dan di mana saja kepada pelanggan [22].

Tujuan dari implementasi sistem pemesanan adalah untuk memberikan jaminan waktu kepada pengguna dalam menggunakan suatu layanan atau fasilitas. Sistem ini menjawab kebutuhan pengguna yang ingin memastikan ketersediaan layanan pada waktu tertentu, serta membantu pengelola dalam menyusun jadwal agar tidak terjadi tumpang tindih pemakaian [23]. Sistem pemesanan juga berfungsi sebagai kontrol administratif bagi pihak pengelola dalam mengelola sumber daya secara efisien.

Sistem *booking* biasanya dijalankan dengan aplikasi digital yang memiliki fitur seperti penjadwalan otomatis, pemberitahuan (*notification*), dan integrasi dengan sistem pembayaran. Penerapan sistem ini terbukti meningkatkan efektivitas bisnis dan kepuasan pelanggan karena waktu pelayanan menjadi lebih terstruktur dan dapat diprediksi dengan baik [24]. Hal ini juga mendukung fleksibilitas pengelolaan, terutama dalam bisinis yang memiliki frekuensi pemesanan tinggi.

Dari segi teknis penggunaan, *booking* atau reservasi memiliki perbedaan dibandingkan dengan pemesanan biasa. Berikut ini adalah perbandinganya.

1. Waktu Penetapan:
    a. Booking: Dilakukan sebelum waktu penggunaan dengan jadwal tertentu yang disepakati.
    b. Pemesanan Biasa: Dapat dilakukan secara langsung dan spontan tanpa waktu tertentu.

2. Ketersediaan Sistem Digital:
    a. Booking: Umumnya tersedia dalam aplikasi sistem daring berbasis data.
    b. Pemesanan Biasa: Umumnya melalui komunikasi langsung, seperti lisan atau via telepon.
3. Kepastian Layanan:
    a. Booking: Memberikan jaminan kepastian waktu dan layanan bagi pelanggan.
    b. Pemesanan Biasa: Bergantung pada ketersediaan saat itu juga.

Dengan semakin berkembangnya teknologi dan kebutuhan pengguna akan efisiensi layanan, sistem *booking* berbasis aplikasi digital menjadi solusi ideal. Penerapan sistem ini pada layanan pemesanan lapangan olahraga, seperti Rahayu *Minisoccer*, memungkinkan pelanggan untuk memperoleh informasi ketersediaan lapangan secara *real-time* dan memesan tanpa harus datang langsung. Sementara bagi pengelola, sistem ini memberikan kontrol terpusat terhadap aktivitas sewa dan riwayat transaksi yang terdokumentasi dengan baik [25].

2.6 Penjadwalan Otomatis

Penjadwalan otomatis adalah proses pengaturan waktu pelaksanaan aktivitas secara sistematis dengan bantuan sistem sehingga jadwal dapat terbentuk tanpa perlu intervensi manual. Tujuan utama dari penjadwalan otomatis adalah memastikan kegiatan berjalan tepat waktu, menghindari benturan jadwal, dan meningkatkan efisiensi operasional. Sistem penjadwalan otomatis bekerja dengan menentukan waktu eksekusi berdasarkan aturan atau alur kerja tertentu yang telah diatur sebelumnya oleh pengguna atau administrator sistem. Pendekatan ini telah digunakan dalam berbagai aplikasi digital untuk membantu pengguna mengelola aktivitas tepat waktu secara konsisten [26].

Dalam pengembangan sistem informasi, penjadwalan otomatis sering diterapkan pada layanan digital yang membutuhkan eksekusi terjadwal dan berulang. Penelitian mengenai penjadwalan unggahan konten media sosial menunjukkan bahwa sistem dapat menentukan dan melaksanakan jadwal posting secara otomatis berdasarkan pengaturan pengguna, sehingga proses distribusi konten dapat berjalan lebih efektif dan terjadwal tanpa campur tangan manual [26]. Begitu juga pada aplikasi penjadwalan promosi perguruan tinggi, sistem mampu mengeksekusi jadwal publikasi informasi tanpa keterlibatan operator secara langsung, sehingga mempercepat penyebaran informasi sekaligus meminimalkan kesalahan dan keterlambatan input data [27].

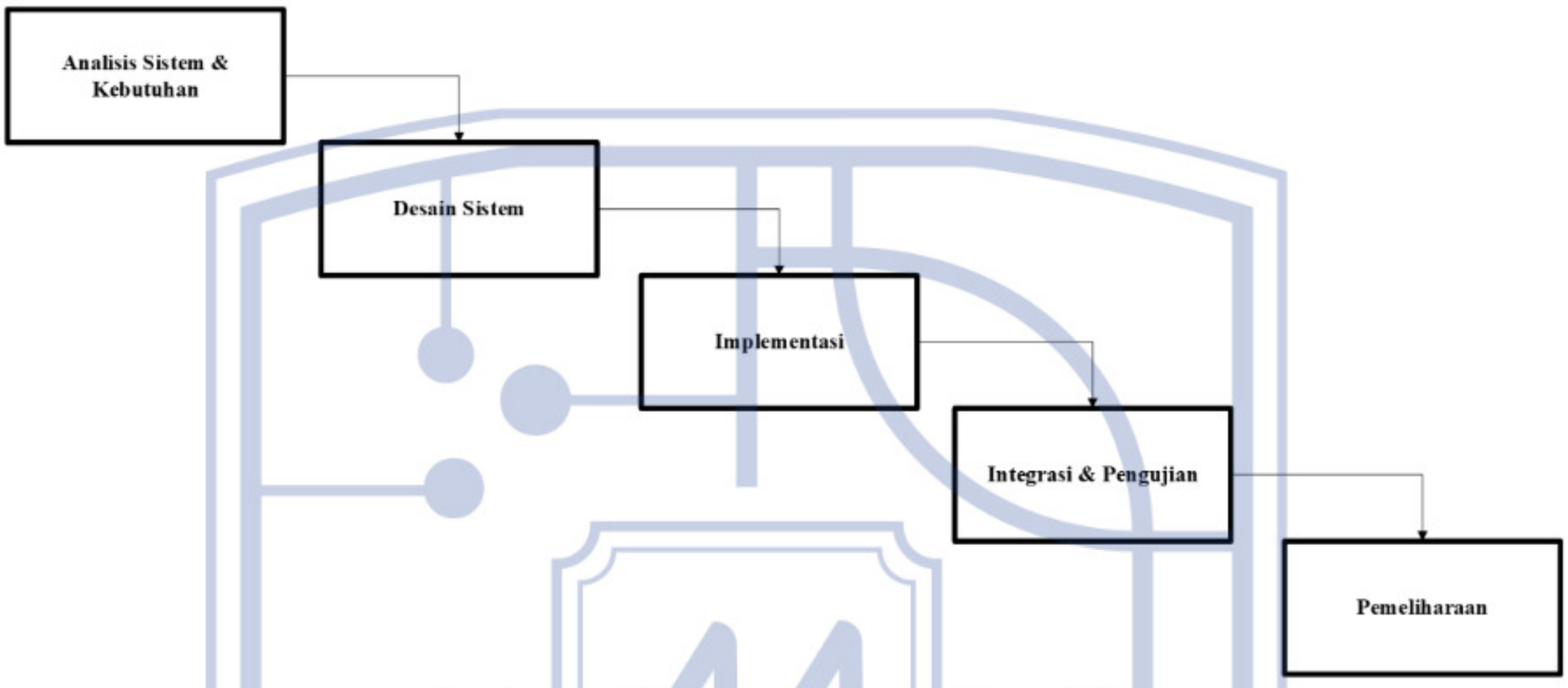
Dalam konteks aplikasi pemesanan lapangan *minisoccer*, penjadwalan otomatis berperan penting dalam memastikan setiap pemesanan diproses sesuai urutan waktu permintaan serta menghindari terjadinya pemesanan ganda (*double booking*). Sistem akan melakukan pengecekan ketersediaan waktu secara otomatis, menyimpan jadwal yang telah dipesan, dan memblokir waktu tersebut agar tidak dapat dipesan ulang oleh pengguna lain. Dengan mengadaptasi konsep penjadwalan otomatis pada penelitian sebelumnya, pengembangan aplikasi ini diharapkan mampu meningkatkan ketepatan pengelolaan jadwal penyewaan, mempercepat proses pemesanan, dan memberikan pengalaman layanan yang lebih efisien bagi pengguna. Pada akhirnya, penerapan penjadwalan otomatis pada sistem pemesanan lapangan sangat mendukung kebutuhan proses digitalisasi dalam layanan olahraga.

2.7 Metode Waterfall

Metode *Waterfall* merupakan salah satu model pengembangan perangkat lunak yang menggunakan pendekatan berurutan atau linier, di mana setiap tahapan harus diselesaikan sepenuhnya sebelum melanjutkan ke tahap berikutnya [28]. Model ini sering digunakan pada proyek perangkat lunak yang kebutuhan sistemnya sudah jelas sejak awal pengembangan, sehingga proses dapat berjalan terstruktur dan terencana. Waterfall menjadi pilihan dalam pengembangan aplikasi operasional karena memberikan gambaran yang jelas mengenai alur sistem dari tahap awal hingga implementasi.

Dalam praktiknya, model *waterfall* berfungsi untuk memastikan proses pengembangan berjalan secara sistematis sehingga pengembangan dapat mengidentifikasi kebutuhan, merancang solusi, mengimplementasikannya, lalu melakukan pengujian sebelum sistem di terapkan ke pengguna [29]. Pendekatan ini membantu mengurangi kesalahan karena setiap tahap memiliki dokumentasi dan hasil evaluasi sebelum melanjutkan ke tahap selanjutnya. *Waterfall* cocok diterapkan yang memerlukan dokumentasi lengkap dan kestabilan sistem.

Model *Waterfall* diterapkan karena kelebihanya dalam memberikan struktur yang jelas, sehingga pengembang dapat mengontrol setiap tahapan selama proses pengembangan aplikasi [30]. Dalam konteks Pembangunan aplikasi pemesanan lapangan *minisoccer*, metode ini memastikan sistem dapat dikembangkan secara bertahap dengan validasi fungsi pada setiap fase untuk memenuhi kebutuhan pengguna dan pengelola lapangan secara optimal [31]. Berikut ini adalah tahapan metode *waterfall* dapat dilihat pada Gambar 2.2.



Gambar 2.2 Tahapan Metode *Waterfall*

1. Analisis Sistem & Kebutuhan (*Requirement Analysis*):

   Pada tahap ini dilakukan proses pengumpulan informasi dari pengguna untuk memahami kebutuhan sisem yang akan dibangun. Semua kebutuhan dicatat secara rinci agar pengembangan mengetahui fungsi yang harus tersedia dalam aplikasi.

2. Desain sistem (*System Design*):

   Tahap ini berfokus pada penyusunan rancangan sistem berdasarkan kebutuhan yang telah di identifikasi. Desain mencakup struktur data, arsitektur perangkat lunak, alur proses, dan antarmuka pengguna untuk memastikan sistem dapat memenuhi kebutuhan pengguna secara fungsional

3. Implementasi (*coding)*:

   Setelah tahap desain selesai, proses dilanjutkan dengan menerjemahkan rancangan menjadi kode program menggunakan bahasa dan framework yang sesuai. Dalam penelitian ini, pengembangan aplikasi dilakukan menggunakan *Flutter* sebagai framework utama dan *Firebase* sebagai layanan basis data.

4. Integrasi & Pengujian (*Integration & Testing*):

   Tahap pengujian (*testing)* bertujuan untuk memastikan bahwa aplikasi telah berjalan sesuai kebutuhan dan bebas dari kesalahan. Pengujian dilakukan untuk mengecek

setiap fitur yang diterapkan, memastikan tidak ada bug, serta memvalidasi hasil kerja sistem sebelum diterapkan secara penuh kepada pengguna.

5. Pemeliharaan (*Maintenance*):

   Tahap terakhir dari metode *waterfall* adalah pemeliharaan, yaitu proses memperbaiki kesalahan yang ditemukan setelah aplikasi digunakan dan melakukan peningkatan fitur agar sistem tetap relevan dan sesuai kebutuhan pengguna di masa mendatang.

Dengan adanya metode *Waterfall,* proses pengembangan aplikasi menjadi lebih terstruktur dan mudah dikendalikan, karena setiap tahap menghasilkan dokumentasi dan keluaran yang jelas sebelum melanjutkan ke tahap selanjutnya. Metode ini tepat digunakan dalam pengembangan aplikasi pemesanan lapangan *Minisoccer* karena kebutuhan sistem telah ditentukan sejak awal dan memerlukan alur kerja yang sistematis untuk menjamin kualitas hasil akhir.

2.8 Blackbox Testing

Black Box Testing merupakan metode pengujian perangkat lunak yang dilakukan untuk mengevaluasi fungsionalitas sistem tanpa mengetahui struktur internal atau kode program dari aplikasi yang di uji. Teknik ini difokuskan pada pengujian *input* dan *output*, serta bagaimana sistem merespon terhadapa berbagai scenario penggunaan yang telah dirancang berdasarkan spesifikasi kebutuhan [32]. Dalam pendekatan ini, penguji hanya perlu mengetahui apa yang seharusnya dilakukan oleh sistem, bukan bagaiman sistem tersebut bekerja secara teknis di balik layer.

Metode *Black box testing* sangat sesuai untuk pengujian aplikasi yang telah dikembangkan secara lengkap dan siap di uji oleh pengguna atau tim pengujian. Teknik ini efektif untuk mengidentifikasi kesalahan fungsi, kegagalan integrasi antar komponen, serta ketidaksesuaian antara antarmuka pengguna dan kebutuhan pengguna. Selain itu, metode ini tidak memerlukan pemahaman teknis struktur kode, sehingga dapat dilakukan oleh tester atau penguji awal [33].

Perbedaan antar *Black box* dan *White box testing* terletak pada fokus pengujian dan siapa yang melakukanya. Untuk memperjelas perbedaan antara kedua metode tersebut. disajikan tabel perbandingan 2.1.

Tabel 2.1 Perbedaan White Box Testing dan Black Box Testing

Aspek	*Whitebox Testing*	*Blackbox Testing*
Fokus Pengujian	Struktur internal dan logika kode program	Fungsionalitas dan keluaran sistem
Pengetahuan Penguji	Harus mengetahui kode sumber	Tidak perlu mengetahui kode program
Dilakukan Oleh	Developer atau programmer	tester atau pengguna akhir
Teknik yang Digunakan	*Path testing*, *loop testing*, *condition testing*	*Equivalence partitioning, boundary value analysis*
Tujuan	Memastikan logika kode bekerja dengan benar	Memastikan sistem berfungsi sesuai spesifikasi
Waktu Pelaksanaan	Dilakukan selama pengembangan	Setelah sistem selesai dikembangkan

Dalam penelitian ini, *Blackbox testing* digunakan untuk menguji seluruh fitur pada aplikasi *mobile* pemesanan lapangan Rahayu *Minisoccer*. Pengujian dilakukan berdasarkan scenario penggunaan nyata oleh pengguna, misalnya melakukan *login*, memesan lapangan, melihat jadwal, menerima notifikasi, dan lainnya. Dengan pendekatan ini, seluruh alur interaksi pengguna dapat dievaluasi secara menyeluruh tanpa harus memeriksa bagian dalam dari sistem atau kode program [34].

Kelebihan metode ini terletak pada kemampuanya mendeteksi kesalahan dari sudut pandang pengguna dan memastikan bahwa aplikasi benar-benar memenuhi kebutuhan bisnis serta kenyamanan pengguna. Black Box Testing ini juga dianggap lebih realistis karena mencerminkan bagaimana aplikasi akan digunakan dalam situasi nyata. Oleh karena itu, metode ini sangat cocok diterapkan dalam pengujian akhir (*Final Testing*) sebelum aplikasi dirilis ke publik[35].

2.9 Rahayu Minisoccer

Rahayu *MiniSoccer* merupakan salah satu fasilitas olahraga yang menyediakan sarana permainan sepak bola mini (*Minisoccer*) di wilayah Sumatera Utara. Fasilitas ini berlokasi di Jl. Rahayu, Pasar 6 Tembung, Kecamatan Percut Sei Tuan, Kabupaten Deli Serdang. Letaknya yang strategis di kawasan padat penduduk menjadikan Rahayu Mini Soccer sebagai pusat aktivitas olahraga bagi masyarakat sekitar, komunitas lokal, serta tim sepak bola amatir dari wilayah Medan dan sekitarnya. Keberadaan fasilitas ini turut mendukung kebutuhan masyarakat terhadap sarana olahraga yang mudah diakses dan representatif.

Gambar 2.3 Rahayu Minisoccer

Sebagai unit usaha penyewaan lapangan, Rahayu *MiniSoccer* menyediakan dua lapangan dengan spesifikasi yang mendukung aktivitas permainan secara optimal. Lapangan pertama memiliki dimensi 30 × 50 meter, sedangkan lapangan kedua berukuran 33 × 52 meter. Kedua lapangan menggunakan rumput sintetis yang dirancang untuk mendukung format permainan 7 lawan 7 (7 vs 7), sehingga sesuai dengan karakteristik mini soccer sebagai bentuk modifikasi dari sepak bola standar yang menekankan efisiensi ruang dan intensitas permainan.

Untuk menunjang operasional dan kenyamanan pengguna, Rahayu Mini Soccer dilengkapi dengan berbagai fasilitas pendukung seperti sistem pencahayaan stadion yang memungkinkan penggunaan lapangan pada malam hari, area tunggu bagi penonton, serta area parkir yang memadai. Selain itu, pengelolaan reservasi telah memanfaatkan platform

digital seperti AYO Indonesia guna mempermudah proses pemesanan dan meningkatkan transparansi jadwal. Secara fungsional, fasilitas ini tidak hanya berperan sebagai sarana olahraga, tetapi juga sebagai ruang interaksi sosial masyarakat melalui kegiatan rekreasi, pertandingan persahabatan, maupun turnamen lokal.

2.10 User Acceptance Testing (UAT)

User Acceptance Testing (UAT) merupakan metode pengujian yang dilakukan oleh pengguna akhir (end-user) untuk memastikan bahwa sistem yang dikembangkan telah memenuhi kebutuhan pengguna dan dapat diterima untuk digunakan. Pengujian ini bertujuan untuk mengetahui tingkat penerimaan pengguna terhadap sistem berdasarkan pengalaman pengguna saat menggunakan fitur-fitur yang tersedia pada aplikasi[36].

Menurut Aliyah, Hartono, dan Muin, UAT dilakukan dengan melibatkan pengguna secara langsung dalam proses pengujian sistem melalui penyusunan pertanyaan kuesioner, pelaksanaan pengujian oleh responden, serta analisis hasil pengujian untuk mengetahui tingkat penerimaan pengguna terhadap sistem yang dibangun [36]. Hasil pengujian tersebut dapat digunakan sebagai dasar untuk menentukan apakah sistem telah layak digunakan atau masih memerlukan perbaikan. Selain itu, Yakub et al. menyatakan bahwa tujuan utama UAT bukan hanya untuk memeriksa kesesuaian fungsi sistem, tetapi juga untuk memastikan bahwa perangkat lunak telah memenuhi kebutuhan pengguna akhir. UAT dapat digunakan sebagai evaluasi terhadap tingkat penerimaan pengguna setelah sistem selesai dikembangkan dan sebelum diimplementasikan secara penuh [37].

Pada penelitian ini, UAT digunakan sebagai pengujian tambahan untuk mengetahui tingkat penerimaan pengguna terhadap aplikasi mobile pemesanan lapangan Rahayu Minisoccer. Pengujian dilakukan dengan melibatkan pelanggan dan pengelola sebagai responden. Responden diminta untuk mencoba fitur-fitur aplikasi, kemudian memberikan penilaian melalui kuesioner menggunakan skala Likert lima tingkat.

Tabel 2.2 Skala Likert UAT

Bobot	Keterangan
1	Sangat Tidak Setuju (STS)
2	Tidak Setuju (TS)
3	Cukup Setuju (CS)
4	Setuju (S)
5	Sangat Setuju (SS)