BAB 2
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penelitian ini mengacu pada beberapa studi sebelumnya yang membahas pengembangan sistem informasi berbasis web, perancangan UI/UX, pengujian pengalaman pengguna, serta *usability testing*. Tinjauan pustaka ini digunakan untuk mengetahui sejauh mana penelitian sebelumnya membahas pengembangan sistem informasi, perancangan antarmuka, dan evaluasi pengalaman pengguna. Hasil dari tinjauan pustaka ini menjadi acuan untuk memperkuat fokus penelitian pada pengembangan *frontend* Sistem Informasi Profil SSB Rumbai Pratama berbasis web. Penelitian yang dilakukan oleh Imanudin (2022) membahas Sistem Informasi Manajemen Sekolah Sepak Bola Mandala Majalengka berbasis web. Sistem tersebut dikembangkan untuk membantu pengelolaan pendaftaran siswa, data nilai siswa, data kegiatan, dan jadwal latihan.

Namun, penelitian tersebut masih berfokus pada fungsi sistem secara umum dan belum menitikberatkan pada pengembangan *frontend* dengan penerapan prinsip UI/UX secara khusus. Penelitian oleh Ramadani dan Mahdiana (2024) membahas penerapan metode UI/UX dalam pengembangan sistem digital melalui systematic literature review. Penelitian tersebut menunjukkan bahwa UI/UX memiliki peran penting dalam meningkatkan kualitas interaksi, kenyamanan, dan kemudahan penggunaan sistem. Namun, penelitian tersebut masih bersifat umum dan tidak membahas penerapan UI/UX pada sistem informasi sekolah sepak bola. Penelitian oleh Andayani dan Frobenius (2026) membahas penggunaan UEQ dan Maze *usability testing* untuk mengevaluasi pengalaman pengguna aplikasi Sleman Mart. Penelitian ini menunjukkan bahwa UEQ dapat digunakan untuk menilai pengalaman pengguna, sedangkan Maze dapat digunakan untuk melihat hasil pengujian usability berdasarkan *task* yang dikerjakan pengguna.

Penelitian oleh Najaf, Permatasari, dan Sakti (2024) membahas evaluasi usability aplikasi menggunakan SUS, heuristic evaluation, dan Maze Tool. Penelitian tersebut menunjukkan bahwa Maze dapat digunakan untuk memperoleh data interaksi pengguna selama pengujian usability. Penelitian ini relevan sebagai

acuan dalam penggunaan Maze untuk mengevaluasi kemudahan pengguna dalam menyelesaikan *task*.

Berdasarkan penelitian-penelitian terdahulu tersebut, dapat disimpulkan bahwa penelitian sebelumnya telah membahas sistem informasi berbasis web, UI/UX, UEQ, dan *usability testing*. Namun, belum banyak penelitian yang secara khusus menitikberatkan pada pengembangan *frontend* sistem informasi sekolah sepak bola dengan penerapan UI/UX serta pengujian menggunakan UEQ-S dan Maze. Oleh karena itu, penelitian ini berfokus pada pengembangan *frontend* Sistem Informasi Profil SSB Rumbai Pratama agar tampilan sistem lebih informatif, mudah digunakan, dan sesuai dengan kebutuhan pengguna. Ringkasan penelitian terdahulu disajikan pada Tabel 2.1.

Tabel 2.1 Perbandingan variabel penelitian

No	Peneliti & Tahun	Judul Penelitian dan Teknologi	Hasil
1	Imanudin (2022)	Sistem Informasi Manajemen Sekolah Sepak Bola Mandala Majalengka Berbasis Web	Sistem membantu pengelolaan pendaftaran siswa, data nilai, kegiatan, dan jadwal latihan. Namun, fokus penelitian belum menitikberatkan pada pengembangan *frontend* dan UI/UX.
2	Ramadani & Mahdiana (2024)	*Systematic Literature Review on the Application of UI/UX Design Methods in System Development*	UI/UX berperan penting dalam meningkatkan kualitas interaksi dan kenyamanan pengguna pada sistem digital. Namun, penelitian masih bersifat umum.
3	Andayani & Frobenius (2026)	Implementasi Metode UEQ dan Maze *Usability testing* pada Pengalaman Pengguna Aplikasi Sleman Mart	UEQ digunakan untuk menilai pengalaman pengguna, sedangkan Maze digunakan untuk menguji usability berdasarkan *task* pengguna.
4	Najaf, Permatasari, & Sakti (2024)	*Evaluating the Usability of a Book Stock and Sales Recording Android App: SUS,*	Maze Tool digunakan untuk memperoleh data interaksi pengguna dalam pengujian usability.

No	Peneliti & Tahun	Judul Penelitian dan Teknologi	Hasil
		Heuristic Evaluation, and Maze Tool Insights	
5	Penelitian Saat Ini	Rancang Bangun *Frontend* Sistem Informasi Profil Sekolah Sepak Bola Berbasis Web sebagai Media Promosi dan Manajemen Data Siswa (Studi Kasus: Sekolah Sepak Bola Rumbai Pratama)	Penelitian ini berfokus pada perancangan *frontend* berbasis web dengan penerapan prinsip UI/UX untuk menghasilkan tampilan antarmuka yang modern, responsif, mudah dipahami, dan sesuai dengan kebutuhan pengguna.

2.2 Landasan Teori

2.2.1 Sistem Informasi Sekolah Sepak Bola

Sistem informasi sekolah sepak bola merupakan aplikasi berbasis digital yang dirancang untuk mempermudah pengelolaan kegiatan di lingkungan SSB, seperti *Admin*istrasi, komunikasi, evaluasi performa, dan dokumentasi. Sistem ini biasanya dapat diakses melalui perangkat *web* maupun *mobile,* sehingga memungkinkan berbagai pihak seperti pelatih, *Admin*, dan orang tua untuk berinteraksi dan memperoleh informasi dengan lebih cepat dan efisien. Menurut Wahyudi et al. (2022), "Dengan adanya sistem ini, diharapkan efisiensi operasional dapat meningkat, dan transparansi informasi kepada orang tua serta pelatih dapat terjaga dengan baik". Dengan adanya sistem yang terpusat, semua informasi penting seperti jadwal latihan, nilai evaluasi siswa, dan data pendaftaran dapat disimpan secara aman dan mudah diakses kapan saja. Oleh karena itu, penggunaan sistem informasi semacam ini akan sangat membantu dalam meningkatkan keteraturan dan transparansi dalam proses pembinaan, terutama dalam mendukung komunikasi yang lancar antara pihak sekolah sepak bola dengan orang tua siswa.

2.2.2 Software Development Life Cycle (SDLC)

Software Development Life Cycle atau SDLC adalah sebuah kerangka kerja sistematis yang digunakan untuk mengembangkan perangkat lunak secara bertahap, mulai dari tahap perencanaan hingga pemeliharaan. SDLC membantu pengembang untuk merancang, membangun, dan memelihara perangkat lunak dengan lebih

terstruktur dan efisien, serta meminimalkan risiko kesalahan selama proses pengembangan berlangsung. Menurut Rahmi, Yumami, dan Hidayasari (2023), pengembangan sistem informasi berbasis website membutuhkan metode pengembangan yang sesuai agar proses pembangunan sistem dapat berjalan lebih terarah dan sesuai kebutuhan pengguna. Tahap pertama dalam SDLC adalah perencanaan, di mana dilakukan identifikasi kebutuhan sistem dan kelayakan proyek baik dari sisi teknis, ekonomi, maupun operasional. Selanjutnya adalah tahap analisis, di mana pengembang mencoba memahami lebih dalam masalah yang dihadapi pengguna serta merumuskan solusi melalui pemodelan sistem seperti use case dan diagram alur data. Setelah itu, masuk ke tahap desain, yaitu proses membuat rancangan teknis dari sistem, termasuk desain *database*, struktur antarmuka pengguna, serta arsitektur sistem secara keseluruhan.

Tahap implementasi dilakukan setelah desain disetujui, di mana pengembang mulai menulis kode program sesuai dengan desain yang telah dibuat. Kemudian sistem akan masuk ke tahap pengujian (*testing*), untuk memastikan semua fitur berjalan dengan baik dan tidak ditemukan kesalahan (*bug*) sebelum diserahkan kepada pengguna. Tahap terakhir adalah pemeliharaan (*maintenance*), yang dilakukan setelah sistem digunakan. Tahap ini mencakup perbaikan *bug*, peningkatan sistem, dan pembaruan berdasarkan umpan balik dari pengguna.

Dengan menggunakan pendekatan SDLC, proses pengembangan sistem informasi seperti sistem informasi SSB menjadi lebih terkontrol dan sistematis, serta mampu beradaptasi dengan perubahan kebutuhan yang mungkin terjadi selama proyek berlangsung.

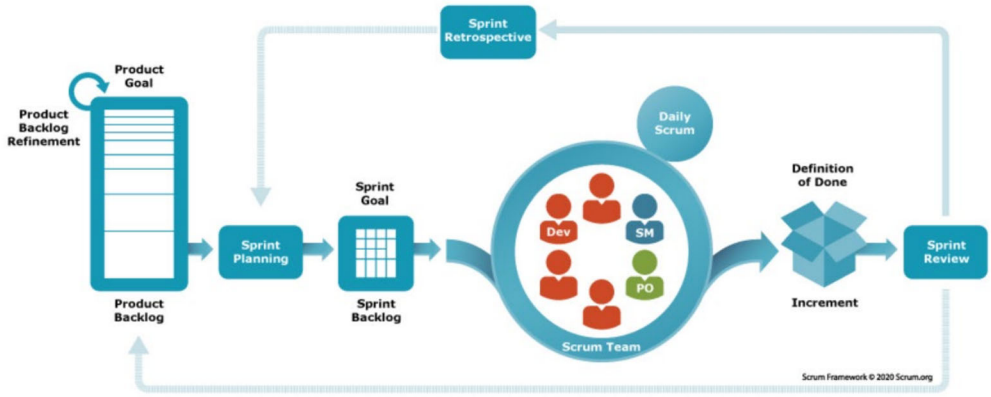
2.2.3 Metodologi Agile – Scrum

Metodologi *Agile* merupakan pendekatan dalam pengembangan perangkat lunak yang bersifat iteratif dan fleksibel terhadap perubahan kebutuhan selama proses pengembangan berlangsung. *Agile* membagi proses pengembangan menjadi siklus-siklus pendek atau disebut juga *iteration,* sehingga pengembang dapat memberikan hasil yang bisa langsung diuji dalam waktu singkat. Salah satu kerangka kerja yang paling umum digunakan dalam metodologi *Agile* adalah Scrum.

Scrum memungkinkan tim pengembang untuk bekerja secara kolaboratif dalam siklus kerja yang disebut *Sprint.* Setiap *Sprint* biasanya berlangsung antara dua hingga empat minggu, dengan tujuan menghasilkan increment atau bagian dari sistem yang siap digunakan. Menurut Schwaber dan Sutherland (2020), ”dalam pendekatan Scrum, proses pengembangan dibagi menjadi beberapa *Sprint,* yang biasanya berlangsung selama dua hingga empat minggu”. Di setiap *Sprint,* ada tahapan-tahapan penting seperti *Sprint planning* (perencanaan), *daily scrum* (rapat harian), *Sprint review* (peninjauan hasil), dan *Sprint retrospective* (evaluasi proses kerja), yang semuanya dilakukan untuk menjaga fokus tim, meningkatkan kualitas hasil, dan memperbaiki kinerja tim di *Sprint* berikutnya.

Struktur tim dalam Scrum terdiri dari tiga peran utama, yaitu *Product Owner* yang bertanggung jawab terhadap visi produk dan prioritas pengembangan, Scrum Master yang memfasilitasi proses kerja dan mengatasi hambatan, serta *Development Team* yang menjalankan proses pengembangan secara teknis. Selain itu, Scrum juga memiliki tiga artefak utama, yakni *Product Backlog* sebagai daftar kebutuhan yang harus dipenuhi, *Sprint Backlog* sebagai rincian tugas dalam satu *Sprint*, dan Increment sebagai hasil kerja yang telah selesai dan memenuhi kriteria penerimaan.

Berdasarkan kompleksitas kebutuhan yang ada di SSB Rumbai Pratama, seperti pengelolaan pendaftaran *online*, presensi siswa, evaluasi performa, hingga jadwal latihan, Scrum sangat cocok untuk diterapkan. Sistem yang dibangun dapat dikembangkan secara bertahap dan disesuaikan dengan masukan dari pengguna seperti pelatih, orang tua, dan *Admin*. Pendekatan ini memungkinkan sistem terus berkembang seiring dengan kebutuhan di lapangan tanpa harus menunggu proyek selesai secara keseluruhan. Dengan menggunakan Scrum, pengembangan sistem informasi untuk SSB ini dapat berjalan lebih adaptif dan responsif terhadap perubahan yang mungkin muncul di tengah proses.



Gambar 2.1 Metodologi *Agile* – Scrum

2.2.4 React JS

React JS adalah sebuah pustaka JavaScript yang digunakan untuk membangun antarmuka pengguna secara dinamis dan responsif melalui pendekatan pengelolaan tampilan yang efisien. React memanfaatkan konsep *virtual DOM* untuk mempercepat proses pembaruan elemen pada halaman *web,* sehingga perubahan dapat dilakukan tanpa harus memuat ulang keseluruhan halaman. Pendekatan ini membantu pengembang dalam menciptakan aplikasi *web* yang terstruktur, modern, serta memiliki kinerja yang lebih optimal. Menurut Meta (2023), React dirancang untuk memungkinkan pengembang membangun antarmuka pengguna yang dinamis dan responsif, sehingga sangat sesuai dengan kebutuhan pengembangan aplikasi masa kini yang menuntut kecepatan, fleksibilitas, dan efisiensi dalam pengolahan tampilan. Dengan kemampuannya tersebut, React menjadi pilihan yang tepat untuk membangun aplikasi yang memiliki banyak interaksi dan perlu menampilkan data secara *real time.* Dalam konteks pengembangan sistem informasi SSB, React sangat sesuai digunakan karena dapat memberikan pengalaman pengguna yang nyaman, baik bagi pelatih, orang tua, maupun *Admin* yang mengakses sistem melalui berbagai jenis perangkat.

2.2.5 Figma

Figma merupakan alat desain antarmuka (UI/UX) berbasis *cloud* yang digunakan oleh tim desain untuk membuat prototipe sistem secara interaktif dan kolaboratif. Desain yang dibuat di Figma dapat diuji dan ditinjau secara *real time* sehingga pengembang lebih mudah memahami apa yang perlu diimplementasikan. Menurut Figma (2024), *"Figma memungkinkan tim desain untuk membuat*

prototipe antarmuka sistem secara interaktif, sehingga pengembang dapat dengan mudah menerjemahkan desain tersebut ke dalam kode." Dengan fitur-fitur seperti desain berbasis vektor, prototyping, dan kolaborasi tim, Figma sangat mendukung pengembangan aplikasi dengan tampilan antarmuka kompleks. Dalam proyek sistem informasi SSB, penggunaan Figma sangat membantu karena mempermudah komunikasi antara tim desain dan pengembang serta mempercepat proses pembuatan antarmuka yang sesuai kebutuhan pengguna.

2.2.6 Visual Studio Code (VS Code)

Visual Studio Code (VS Code) adalah editor kode sumber ringan yang dikembangkan oleh Microsoft dan mendukung berbagai bahasa pemrograman serta dilengkapi fitur seperti integrasi Git, *debugging,* dan *live server.* Microsoft (2024) menyatakan bahwa *"VS Code menjadi pilihan utama bagi banyak pengembang dalam menulis dan mengelola kode mereka."* Dengan tampilan yang sederhana namun fungsional, VS Code memudahkan pengembang dalam menyusun dan mengelola struktur proyek, terutama pada proyek berbasis *web* seperti sistem informasi SSB. *Editor* ini juga memungkinkan penggunaan berbagai ekstensi yang membantu mempercepat proses pengembangan serta mempermudah kolaborasi tim.

2.2.7 XAMPP

XAMPP merupakan paket perangkat lunak yang menyediakan *server* lokal berbasis Apache, *database* MySQL, serta dukungan untuk bahasa pemrograman PHP dan Perl. Perangkat lunak ini banyak digunakan dalam pengembangan sistem karena memungkinkan pengujian aplikasi dilakukan secara lokal sebelum dipublikasikan ke *server* produksi. Apache Friends (2024) menjelaskan bahwa "XAMPP memungkinkan pengembang untuk melakukan pengujian dan pengembangan sistem dengan lebih mudah dan cepat." Dengan XAMPP, pengembang tidak perlu melakukan konfigurasi *server* secara manual karena seluruh komponen sudah tersedia dalam satu paket. Penggunaan XAMPP dalam pengembangan sistem informasi SSB sangat membantu karena mempercepat proses pengujian dan memastikan aplikasi dapat berjalan dengan baik sebelum digunakan oleh pengguna sebenarnya.

2.2.8 Inertia.Js

Inertia.js adalah teknologi yang digunakan untuk menghubungkan *frontend* dan *backend* tanpa perlu membuat API REST terpisah. Ini bukan *framework* mandiri; melainkan, ia bertindak sebagai jembatan antara *framework frontend* dan *backend* yang digunakan dalam pengembangan aplikasi web. Inertia.js mendukung teknologi *frontend* modern seperti React, Vue, dan Svelte, dan terintegrasi dengan *backend* seperti Laravel, Rails, Phoenix, dan Django (Inertia.js, 2026). Dalam pengembangan aplikasi menggunakan Laravel dan React, Inertia.js berfungsi sebagai jembatan antara *controller* Laravel dan komponen React. Laravel menangani *routing*, *controller*, *middleware*, validasi, otentikasi, dan manajemen data, sementara React digunakan untuk membangun antarmuka pengguna berbasis komponen (Laravel, 2026; React, 2026). Dengan pendekatan ini, data yang diproses di *backend* Laravel dapat dikirim langsung ke komponen React sebagai props melalui metode `Inertia::render()` (Inertia.js, 2026).

Inertia.js juga menyediakan pengalaman seperti aplikasi satu halaman (*single-page application*), karena transisi halaman tidak memerlukan pemuatan ulang seluruh dokumen HTML. Ketika pengguna menavigasi ke halaman baru, Inertia melakukan permintaan asinkron melalui XHR. Server kemudian mengembalikan respons JSON yang berisi nama komponen halaman, props, URL, dan versi aset yang dibutuhkan (Inertia.js, 2026). Proses ini membuat transisi halaman lebih responsif, karena halaman dapat diperbarui tanpa pemuatan ulang halaman penuh (Inertia.js, 2026).

Dalam studi ini, Inertia.js digunakan untuk menghubungkan *frontend* React JS dan *backend* Laravel untuk Sistem Informasi Sekolah Sepak Bola Rumbai Pratama. Penggunaan Inertia.js memungkinkan *frontend* untuk menerima data langsung dari *backend*, sehingga memungkinkan tampilan informasi yang dinamis seperti data siswa, jadwal pelatihan, kehadiran, berita, pembayaran, dan evaluasi pemain. Akibatnya, Inertia.js memfasilitasi integrasi sistem dengan menjaga *frontend* dan *backend* dalam alur kerja pengembangan yang terpadu dan saling terhubung (Laravel, 2026; Inertia.js, 2026). Selain memfasilitasi integrasi, penggunaan Inertia.js juga dapat meningkatkan kinerja aplikasi web. Penelitian oleh Tempariyawan, Akbar, Nofiyati, dan Waluyo (2025) menunjukkan bahwa

Laravel Inertia menawarkan waktu eksekusi yang lebih cepat daripada Laravel Blade untuk salah satu fitur yang diuji, khususnya 1,56 detik dibandingkan dengan 3,89 detik. Oleh karena itu, Inertia.js berfungsi sebagai teknologi yang cocok untuk mengembangkan sistem web yang membutuhkan antarmuka dinamis, integrasi *backend* yang efisien, dan pengalaman pengguna yang lebih responsif.

2.2.9 Javascript

JavaScript merupakan salah satu bahasa pemrograman utama dalam pengembangan aplikasi *web* yang bersifat interaktif. Bahasa ini memiliki kemampuan untuk mengatur elemen dinamis di halaman *web* secara langsung tanpa perlu memuat ulang seluruh laman. Seiring waktu, JavaScript terus mengalami pembaruan signifikan seperti penambahan fitur *private class fields, top-level await,* serta metode *array* baru yang membuat proses pengembangan semakin efisien dan rapi (Prepare *Frontend*, 2024). Dalam konteks pengembangan sistem informasi untuk SSB, JavaScript berperan penting dalam mengelola elemen-elemen antarmuka yang digunakan oleh pelatih, *Admin*, dan orang tua dalam melakukan interaksi langsung dengan sistem.

2.2.9 Tahapan User Interface (UI) dan User Experience (UX)

Tahapan *User Interface* (UI) dan *User Experience* (UX) merupakan proses perancangan antarmuka sistem yang bertujuan menghasilkan tampilan yang mudah digunakan, nyaman, dan sesuai dengan kebutuhan pengguna. UI berfokus pada aspek visual dan elemen antarmuka, sedangkan UX berfokus pada pengalaman pengguna saat berinteraksi dengan sistem. Menurut Ramadani dan Mahdiana (2024), UI/UX memiliki peran penting dalam pengembangan sistem digital karena berpengaruh langsung terhadap kemudahan penggunaan dan kepuasan pengguna.

Dalam pengembangan *frontend* sistem informasi berbasis web, tahapan UI/UX perlu dilakukan secara terstruktur agar tampilan antarmuka mampu mendukung efektivitas penggunaan sistem. Sari et al. (2022) menjelaskan bahwa perancangan UI/UX mencakup analisis kebutuhan pengguna, perancangan struktur dan navigasi antarmuka, perancangan tampilan visual, implementasi antarmuka, serta evaluasi pengalaman pengguna. Oleh karena itu, tahapan UI/UX dalam penelitian ini disusun berdasarkan alur pengembangan pada proposal penelitian dan

disesuaikan dengan pendekatan *Agile* Scrum, dengan fokus pada perancangan *frontend* tanpa membahas aspek teknis *backend*.

Tahap awal adalah analisis kebutuhan pengguna untuk memahami karakteristik serta informasi yang dibutuhkan oleh pengguna sistem. Putri dan Nugroho (2021) menyatakan bahwa pemahaman kebutuhan pengguna menjadi dasar dalam merancang antarmuka yang efektif dan sesuai ekspektasi. Selanjutnya dilakukan perancangan struktur dan navigasi antarmuka agar alur penggunaan sistem mudah dipahami dan tidak membingungkan pengguna. Menurut Hidayat et al. (2023), struktur navigasi yang baik dapat meningkatkan kenyamanan dan efisiensi dalam penggunaan sistem.

Tahap berikutnya adalah perancangan tampilan antarmuka dengan menyusun elemen visual seperti tata letak, menu, tombol, dan form secara konsisten. Pratama et al. (2022) menyebutkan bahwa desain antarmuka yang baik harus memperhatikan konsistensi, keterbacaan, dan kemudahan interaksi pengguna. Tampilan yang telah dirancang kemudian diimplementasikan ke dalam *frontend* berbasis React JS dengan pendekatan berbasis komponen. Menurut Ramadani dan Mahdiana (2024), kesesuaian antara rancangan UI/UX dan hasil implementasi sangat penting untuk menjaga konsistensi pengalaman pengguna. Tahap akhir adalah evaluasi tampilan dan pengalaman pengguna untuk mengidentifikasi kekurangan desain agar dapat dilakukan penyempurnaan guna meningkatkan kualitas UI/UX sistem (Sari et al., 2022).

2.2.10 UI Style Guide

UI *Style Guide* merupakan pedoman desain antarmuka yang digunakan untuk menjaga konsistensi tampilan visual dan interaksi pengguna dalam suatu sistem berbasis web. UI *Style Guide* berfungsi sebagai acuan dalam penggunaan elemen visual agar tampilan antarmuka memiliki keseragaman desain, mudah dipahami, dan mampu memberikan pengalaman pengguna yang baik. Menurut Pratama et al. (2022), penerapan UI *Style Guide* dapat meningkatkan konsistensi antarmuka serta membantu pengguna mengenali pola interaksi pada sistem informasi berbasis web. Dalam pengembangan *frontend* sistem informasi berbasis web, UI *Style Guide* berperan penting untuk memastikan bahwa elemen visual yang digunakan dapat mendukung kemudahan penggunaan dan kenyamanan pengguna.

Sari et al. (2022) menyatakan bahwa konsistensi elemen visual seperti warna, tipografi, dan komponen antarmuka berpengaruh terhadap peningkatan *usability* dan kualitas pengalaman pengguna. Secara umum, UI *Style Guide* mencakup pengaturan palet warna, tipografi, serta komponen antarmuka. Palet warna digunakan untuk membangun identitas visual sistem dan membantu pengguna dalam membedakan informasi penting. Tipografi digunakan untuk mengatur jenis huruf, ukuran, dan ketebalan teks agar informasi mudah dibaca dan dipahami. Selain itu, pengaturan komponen antarmuka seperti tombol, form, dan elemen navigasi dilakukan secara konsisten agar pengguna dapat berinteraksi dengan sistem secara lebih intuitif.

Dengan adanya UI *Style Guide*, proses perancangan *frontend* menjadi lebih terarah dan konsisten. UI *Style Guide* juga memudahkan proses pengembangan lanjutan karena setiap elemen antarmuka telah memiliki pedoman yang jelas, sehingga tampilan sistem yang dihasilkan dapat memberikan pengalaman pengguna yang lebih baik.

2.2.11 User Experience Questionnaire (UEQ)

User Experience Questionnaire versi singkat atau UEQ-S merupakan salah satu instrumen dalam UEQ Family yang digunakan untuk mengukur pengalaman pengguna terhadap suatu produk atau sistem secara ringkas. UEQ-S digunakan untuk mengetahui bagaimana pengguna menilai sistem setelah mencoba atau berinteraksi langsung dengan sistem tersebut. Menurut Kollmorgen, Hinderks, dan Thomaschewski (2024), UEQ *Family* terdiri dari beberapa instrumen yang dapat digunakan sesuai kebutuhan evaluasi pengalaman pengguna, salah satunya adalah UEQ-S. UEQ-S digunakan untuk menilai pengalaman pengguna dari dua aspek utama, yaitu *Pragmatic Quality* dan *Hedonic Quality*. *Pragmatic Quality* berkaitan dengan aspek kegunaan sistem, seperti kemudahan penggunaan, kejelasan informasi, efisiensi, dan kemampuan sistem dalam mendukung aktivitas pengguna. Sementara itu, *Hedonic Quality* berkaitan dengan kesan pengguna terhadap sistem, seperti daya tarik tampilan, kenyamanan, kebaruan, dan kesan modern saat sistem digunakan.

Dalam penelitian ini, UEQ-S dipilih karena fokus penelitian berada pada evaluasi pengalaman pengguna terhadap *frontend* Sistem Informasi Profil SSB

Rumbai Pratama. Instrumen ini lebih ringkas dibandingkan UEQ versi lengkap, sehingga lebih sesuai untuk kondisi pengujian lapangan dengan responden orang tua dan pelatih. Selain itu, Schaa, Kollmorgen, Schrepp, dan Thomaschewski (2024) menjelaskan bahwa bentuk UEQ-S yang ringkas dapat membantu proses evaluasi pengalaman pengguna karena waktu pengisian menjadi lebih singkat.

Pada UEQ-S, responden memberikan penilaian menggunakan pasangan kata berlawanan dengan skala 1 sampai 7. Hasil penilaian tersebut kemudian dikonversi ke rentang -3 sampai +3.

Nilai negatif menunjukkan pengalaman pengguna yang kurang baik, nilai 0 menunjukkan posisi netral, sedangkan nilai positif menunjukkan pengalaman pengguna yang baik. Hasil pengujian UEQ-S selanjutnya dapat digunakan untuk mengetahui apakah *frontend* sistem sudah memberikan pengalaman penggunaan yang positif bagi pengguna.

Tabel 2.2 Item Penilaian UEQ

Item	Pasangan Penilaian	Skala yang Diukur
1	Menghambat - Mendukung	*Pragmatic Quality*
2	Rumit - Mudah	*Pragmatic Quality*
3	Tidak efisien - Efisien	*Pragmatic Quality*
4	Membingungkan - Jelas	*Pragmatic Quality*
5	Membosankan - Menarik	*Hedonic Quality*
6	Tidak menarik - Menarik	*Hedonic Quality*
7	Konvensional - Inovatif	*Hedonic Quality*
8	Biasa - Terdepan	*Hedonic Quality*

Tabel 2.3 Rentang Nilai UEQ

Rentang Nilai	Interpretasi	Keterangan
> 0,8	Positif	Pengguna memberikan penilaian baik terhadap sistem.
-0,8 sampai 0,8	Netral	Penilaian pengguna berada pada tingkat sedang atau belum menunjukkan kecenderungan kuat.
< -0,8	Negatif	Pengguna memberikan penilaian kurang baik terhadap sistem.

Berdasarkan tabel diatas hasil UEQ-S dibaca berdasarkan nilai rata-rata yang diperoleh dari hasil pengolahan jawaban responden. Nilai UEQ-S berada pada rentang -3 sampai +3. Nilai yang semakin mendekati +3 menunjukkan bahwa pengalaman pengguna terhadap sistem semakin baik, sedangkan nilai yang semakin mendekati -3 menunjukkan bahwa pengalaman pengguna terhadap sistem kurang baik. Dalam pembacaan hasil UEQ, nilai lebih dari 0,8 termasuk kategori positif, nilai antara -0,8 sampai 0,8 termasuk kategori netral, dan nilai kurang dari -0,8 termasuk kategori negatif (Kollmorgen, Hinderks, & Thomaschewski, 2025). Pedoman resmi UEQ juga menjelaskan interpretasi yang sama, yaitu nilai di atas 0,8 menunjukkan evaluasi positif, nilai di antara -0,8 sampai 0,8 menunjukkan evaluasi netral, dan nilai di bawah -0,8 menunjukkan evaluasi negatif.

Pada penelitian ini, pembacaan nilai UEQ-S digunakan untuk melihat pengalaman pengguna terhadap *frontend* Sistem Informasi Profil SSB Rumbai Pratama. Jika nilai yang diperoleh berada pada kategori positif, maka sistem dapat dinilai sudah memberikan pengalaman penggunaan yang baik. Jika nilai berada pada kategori netral, maka sistem masih dapat digunakan, tetapi belum menunjukkan penilaian yang kuat dari pengguna. Sementara itu, jika nilai berada pada kategori negatif, maka sistem perlu mendapatkan perbaikan karena pengguna memberikan penilaian kurang baik terhadap pengalaman penggunaan sistem.

Nilai tersebut kemudian dibaca berdasarkan tiga hasil utama, yaitu *Pragmatic Quality*, *Hedonic Quality*, dan *Overall*. *Pragmatic Quality* digunakan untuk menilai pengalaman pengguna dari sisi kegunaan sistem, seperti kemudahan, kejelasan, efisiensi, dan kemampuan sistem dalam mendukung aktivitas pengguna. *Hedonic Quality* digunakan untuk menilai kesan pengguna terhadap tampilan dan pengalaman penggunaan, seperti daya tarik, kenyamanan, kebaruan, dan kesan modern. Sementara itu, nilai *Overall* digunakan untuk melihat gambaran umum pengalaman pengguna terhadap sistem secara keseluruhan.

Selain menggunakan pembacaan nilai positif, netral, dan negatif, hasil UEQ-S juga dapat dibandingkan dengan *benchmark*. *Benchmark* digunakan untuk mengetahui posisi hasil pengujian sistem dibandingkan dengan kumpulan hasil evaluasi produk lain. Dengan adanya *benchmark*, hasil UEQ-S tidak hanya dibaca dari tinggi atau rendahnya nilai rata-rata, tetapi juga dapat dilihat apakah hasil

tersebut termasuk kategori *Excellent*, *Good*, *Above Average*, *Below Average*, atau *Bad* (Kollmorgen, Hinderks, & Thomaschewski, 2025).

Tabel 2.4 *Benchmark* UEQ-S

Kategori Benchmark	Penjelasan
Excellent	Hasil evaluasi berada pada kelompok 10% hasil terbaik dibandingkan data pembanding.
Good	Hanya 10% hasil pada data pembanding yang lebih baik, sedangkan 75% hasil lainnya lebih rendah.
Above Average	Sebanyak 25% hasil data pembanding lebih baik, sedangkan 50% hasil lainnya lebih rendah.
Below Average	Sebanyak 50% hasil data pembanding lebih baik, sedangkan 25% hasil lainnya lebih rendah.
Bad	Hasil evaluasi berada pada kelompok 25% hasil terendah dibandingkan data pembanding.

Kategori *Excellent* menunjukkan bahwa hasil evaluasi sistem berada pada kelompok hasil terbaik dibandingkan data pembanding. Kategori *Good*, *Above Average*, dan *Below Average* menunjukkan posisi hasil evaluasi di antara kategori tertinggi dan terendah. Sementara itu, kategori *Bad* menunjukkan bahwa hasil evaluasi sistem berada pada kelompok hasil terendah dibandingkan data benchmark. Kategori *benchmark* tersebut digunakan untuk membantu pembacaan hasil evaluasi pengalaman pengguna agar hasil pengujian tidak hanya dilihat dari nilai rata-rata, tetapi juga dari posisi hasil terhadap data pembanding.

2.2.10 Testing Menggunakan Maze (Usability)

Usability testing merupakan salah satu metode pengujian yang bertujuan untuk mengevaluasi seberapa mudah dan efektif suatu sistem digunakan oleh pengguna. Dalam proyek ini, proses *usability testing* dilakukan dengan memanfaatkan Maze, sebuah *platform* berbasis *cloud* yang mendukung pengujian prototipe secara daring dan *real time*. Maze memungkinkan pengembang untuk mengunggah desain dari *tools* seperti Figma, kemudian menyusun skenario tugas *(Task-based test)* yang akan dijalankan langsung oleh responden secara *remote*. Seluruh interaksi pengguna, seperti klik, waktu penyelesaian, dan tingkat

keberhasilan, akan direkam dan disajikan dalam bentuk laporan analitik yang mudah dipahami.

Menurut Andayani dan Frobenius (2026) menggunakan UEQ dan Maze *usability testing* untuk mengevaluasi pengalaman pengguna berdasarkan skenario penggunaan aplikasi. Penelitian tersebut menunjukkan bahwa Maze dapat digunakan sebagai alat bantu pengujian usability untuk melihat hasil interaksi pengguna terhadap *task* yang diberikan. Kelebihan utama dari Maze adalah kemampuannya dalam menyajikan data kuantitatif maupun kualitatif dari pengguna akhir tanpa perlu proses pengujian secara tatap muka. Hal ini tentunya mempercepat proses evaluasi desain dan sangat relevan dalam pengembangan sistem berbasis *Agile*, di mana iterasi desain perlu dilakukan secara cepat dan responsif terhadap masukan pengguna. Dalam konteks pengembangan sistem informasi SSB Rumbai Pratama, platform ini digunakan untuk menguji kemudahan penggunaan antarmuka pada halaman *login*, *Dashboard*, pengisian presensi, hingga fitur notifikasi. Responden yang terlibat terdiri dari pelatih dan orang tua siswa, yang mewakili pengguna aktual sistem ini.

Menurut Castro, Garnica, & Rojas (2022), penggunaan *tools* otomatis dalam *usability testing*, seperti Maze, telah terbukti meningkatkan efektivitas evaluasi karena mampu mendeteksi permasalahan antarmuka secara langsung dari interaksi pengguna, tanpa memerlukan keterlibatan tim pengembang secara langsung. Studi mereka menunjukkan bahwa *tools* seperti Maze dapat mendukung proses perbaikan desain dengan lebih efisien dibandingkan pendekatan konvensional. Selain itu, hasil penelitian oleh Majrashi et al. (2024) juga memperkuat pentingnya pendekatan *usability* yang terstruktur dan berbasis metrik. Mereka menekankan bahwa pengukuran seperti efektivitas, efisiensi, dan kepuasan pengguna adalah indikator utama dalam menentukan keberhasilan suatu antarmuka.

Maze menyediakan seluruh metrik tersebut dalam laporan otomatis, sehingga memudahkan pengembang dalam mengambil keputusan desain secara cepat dan tepat. Dengan pendekatan ini, *usability testing* bukan hanya menjadi alat evaluasi akhir, tetapi menjadi bagian integral dari proses desain dan pengembangan sistem. Hal ini memastikan bahwa sistem yang dikembangkan benar-benar sesuai dengan kebutuhan dan kenyamanan pengguna, serta siap digunakan dalam praktik nyata.