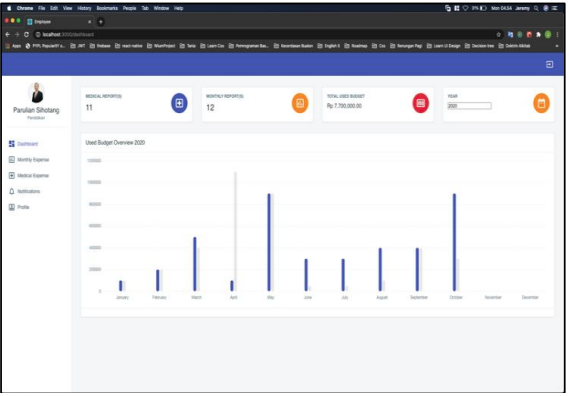
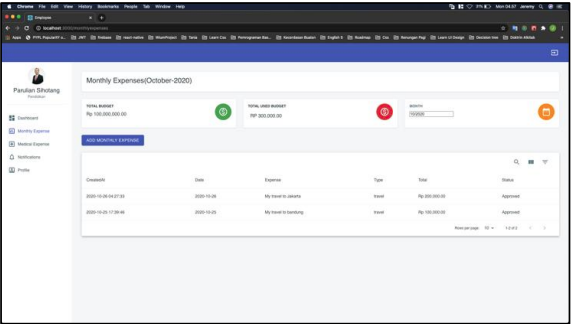
# BAB II TINJAUAN PUSTAKA

## Penelitian Terdahulu

**2.1.1 Perancangan Sistem E-reporting Menggunakan React JS dan Firebase**



# Gambar 2.1 Dashboard aplikasi



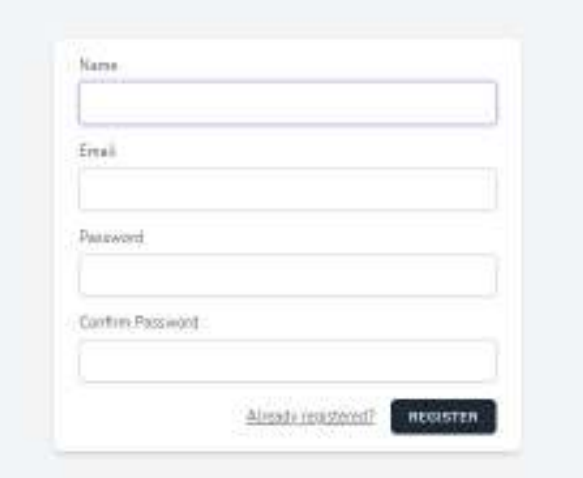
**Gambar 2.2** Dashboard aplikasi

Penelitian yang dilakukan oleh Jeremy Panjaitan dan Andrew Fernando Pakpahan (2021) dengan judul “Perancangan Sistem E-reporting Menggunakan React JS dan Firebase” meneliti tentang penyelenggaraan laporan transaksi dan berbagai kegiatan pada sebuah perusahaan. Peneliti menemukan berbagai macam masalah dalam penyelenggaraannya seperti proses laporan memakan waktu yang cukup lama dan bukti laporan yang kurang akurat. Masalah lainnya yaitu seringkali terdapat laporan yang statusnya tidak diketahui serta karyawan seringkali tidak mengetahui apakah laporannya disetujui atau tidak oleh departemen keuangan. Peneliti membuat sistem informasi ini dengan tujuan untuk mendesain sistem pelaporan elektronik yang mempermudah karyawan untuk melapor kepada departemen keuangan sehingga kegiatan laporan dapat dilakukan secara cepat serta akurat. Sistem informasi ini diimplementasikan menggunakan React JS sebagai teknologi web dan Firebase untuk menyimpan data. Peneliti juga menggunakan SDLC (Software Development Life Cycle) Agile untuk metode pembangunan perangkat lunak yang dinamis.

**2.1.2 Sistem Pendaftaran Mental Health Care Berbasis Website Di Masa Pandemi**



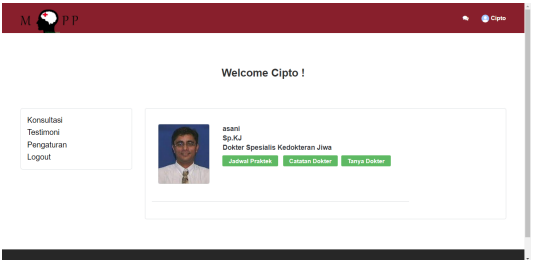
**Gambar 2.3** Landing page



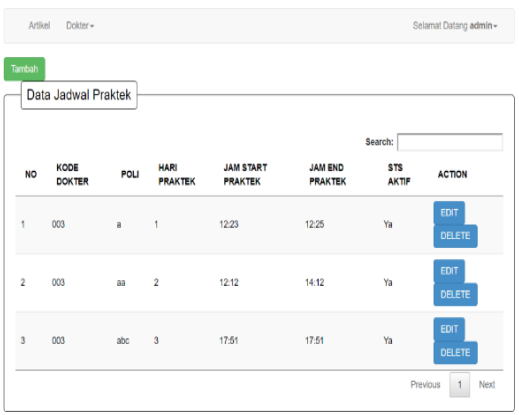
**Gambar 2.4** Form Registrasi

Penelitian yang dilakukan oleh I Gede Suputra Widharma, Putu Gde Sukarata, I Nengah Sunaya, I Gde Nyoman Sangka, Ni Ketut Sukasih (2021) dengan judul “SISTEM PENDAFTARAN MENTAL HEALTH CARE BERBASIS WEBSITE DI MASA PANDEMI” meneliti tentang kesehatan mental masyarakat dan sebuah ide untuk memfasilitasi masalah tersebut dengan cara merancang sistem pendaftaran mental health care yang memudahkan masyarakat bergabung dalam komunitas agar dapat menampung keresahan masyarakat pada umumnya dan berkonsultasi dengan ahlinya. Peneliti menemukan berbagai macam masalah di kehidupan pandemi seperti terlihatnya tanda awal dari gangguan kesehatan mental yaitu stres dan cemas berlebih. Banyaknya pekerja yang terkena PHK, turun nya tunjangan atau gaji pekerja, terkena atau kerabat ada yang terkena Virus COVID-19, kesulitan dalam kuliah maupun belajar melalui daring, dan kekhawatiran atau kecemasan yang dapat mengakibatkan stres lainnya dan juga sulitnya berinteraksi dengan para ahli kejiwaan karena diwajibkannya jaga jarak dan diterapkannya PPKM. Peneliti membuat sistem informasi ini dengan tujuan untuk agar terciptanya sistem pendaftaran yang memudahkan masyarakat untuk bergabung dalam komunitas mental health care sehingga bisa berkonsultasi dengan ahli kejiwaan. Sistem informasi ini diimplementasikan menggunakan HTML, CSS Bootstrap, framework Laravel dan database MySQL.

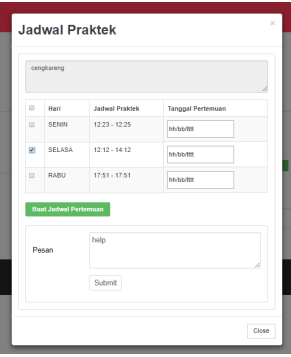
**2.1.3 Perancangan Aplikasi Konsultasi Online Untuk Gangguan Jiwa Berbasis Web**



**Gambar 2.5** Halaman Profil



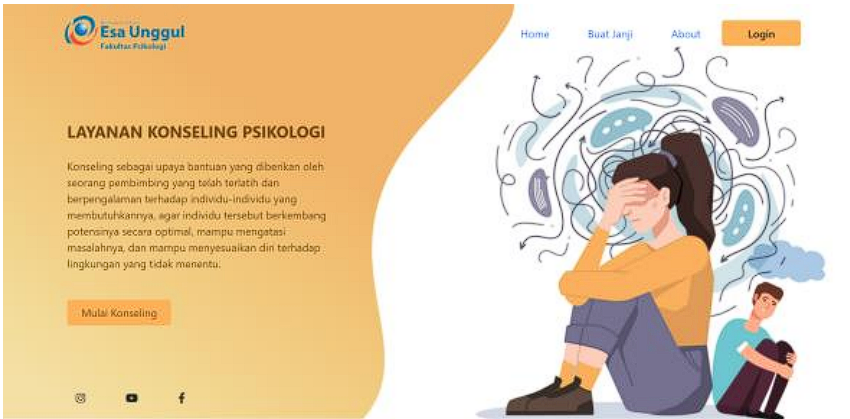
**Gambar 2.6** Dashboard aplikasi



**Gambar 2.7** Modal Jadwal Praktek

Penelitian yang dilakukan oleh Eka Sari Octaviani, Yunita Sari dan Erwin Suhandono (2019) dengan judul “PERANCANGAN APLIKASI KONSULTASI ONLINE UNTUK GANGGUAN JIWA BERBASIS WEB” meneliti tentang penyelenggaraan konsultasi online kesehatan mental. Peneliti menemukan berbagai macam masalah dalam penyelenggaraannya seperti sebagian besar orang yang mengunjungi psikiater merasa canggung bahkan malu karena kuatnya stigma negatif dari masyarakat terhadap orang-orang yang berobat ke dokter jiwa. Masyarakat cenderung masih mempercayai gangguan jiwa adalah penyebab kurangnya iman atau masuknya roh halus ke dalam tubuh manusia yang lemah. Tujuan yang ingin dicapai dari penelitian ini adalah merancang suatu sistem aplikasi berbasis web yang diharapkan dapat memudahkan pasien dalam melakukan konsultasi dengan dokter jiwa, serta, menjadi wadah untuk menyediakan informasi seputar kesehatan jiwa. Sistem informasi ini diimplementasikan menggunakan PHP, HTML, serta dengan framework Bootstrap agar web menjadi responsif.

**2.1.4 Sistem Informasi Layanan E-Konseling Psikologi Untuk Mahasiswa Berbasis Website Dengan Metode Prototype**

****

**Gambar 2.8** Landing page

Penelitian yang dilakukan oleh Nabilla, Arief Ichwani (2022) dengan judul “Sistem Informasi Layanan E-Konseling Psikologi Untuk Mahasiswa Berbasis Website Dengan Metode Prototype” meneliti tentang kesehatan psikologis, kesejahteraan sosial dan emosional mahasiswa mencapai kesuksesan akademis dalam kondisi pandemi COVID-19. selama ini pembuatan janji konseling masih menggunakan WhatsApp yang memakan waktu serta pengajuannya kurang efektif. maka dari itu peneliti membuat sebuah Sistem Informasi e-konseling psikologi berbasis web yang dibangun menggunakan bahasa pemrograman React JS untuk memfasilitasi mahasiswa dalam melakukan pembuatan janji konseling secara online dan terpusat dalam satu aplikasi.

**2.1.5 Research Gap**

Berdasarkan penelitian sebelumnya, penulis menemukan beberapa *research gap* dengan penelitian yang akan dilakukan. Beberapa penelitian sebelumnya berfokus pada pembangunan website menggunakan teknologi yang mulai ditinggalkan di dunia industri, sedangkan penelitian yang akan dilakukan berfokus pada pembangunan website menggunakan teknologi yang sedang banyak dibutuhkan industri saat ini untuk memberikan performa terbaik. Kemudian pada dua penelitian sebelumnya pembuatan sistem di bagian penyimpanan data dilakukan secara manual, sedangkan penelitian yang akan dilakukan menggunakan sebuah layanan cloud untuk menyimpan data untuk skalabilitas, kemudahan, meminimalkan *business cost* dan mempercepat pembangunan aplikasi secara keseluruhan. Lalu pada penelitian sebelumnya belum memiliki fungsi pencatatan jurnal pribadi untuk mencatat kondisi kesehatan mental dan fitur circle agar pengguna dapat saling dukung dan berbagi, sedangkan pada penelitian ini peneliti berencana untuk menambahkan fitur pencatatan jurnal dan fitur circle tersebut.

## Teori Terkait

**2.2.1 Sistem Informasi**

**2.2.1.1 Definisi**

Menurut Cegielski (2014:6), Sistem informasi merupakan proses pengolahan data (mengumpulkan, memproses, menganalisis, dan menyebarkan informasi) untuk tujuan tertentu.

Menurut O’brien (2011:62) Sistem informasi merupakan kombinasi teratur dan terstruktur dari people, hardware, software, serta database. Sistem informasi tersebut mengumpulkan, mengolah, dan menyebarkan informasi di dalam suatu bentuk organisasi.

**2.2.1.2 Tahapan Pembangunan Sistem Informasi**

Dikutip dari laman binaracademy.com yang diakses pada 13 oktober 2022 jam 11.25, pengembangan sistem informasi memiliki 6 tahapan yaitu :

1. **Analisis Kebutuhan**

Analisa Kebutuhan dipimpin oleh lead pada tim dengan masukan dari para stakeholder. kebutuhan user dikumpulkan dan dianalisis oleh tim developer untuk dibentuk menjadi kebutuhan dan standar yang diinginkan. Hal ini berguna untuk memberikan gambaran tentang masalah yang akan diselesaikan, ruang lingkup proyek secara *holistik*, resiko serta peluang yang mungkin terjadi di tahap developing nanti.‍‍

1. **Design**

Design proyek dilakukan oleh tim designer berdasarkan dokumen spesifikasi kebutuhan tahap sebelumnya. tim designer memproduksi sebuah design yang akan dibangun oleh tim developer nanti dan bisa menjadi gambaran awal terkait proyek yang akan dibangun.

1. **Development**

Di fase development, dilakukan proses pembangunan proyek aplikasi yang sudah direncanakan. Tugas pada fase ini dipecah menjadi bagian bagian kecil lalu dibagikan kepada tim setiap developer untuk dikerjakan. Hasil dari tahap development adalah sebuah software beserta source code nya.‍

1. **Testing**

Pada saat software selesai dibangun, software tersebut akan diuji untuk memastikan kualitas software apakah berjalan dengan baik dan sesuai dengan *requirement* user.‍

1. **Deployment**

Tahap ini adalah proses perilisan software agar bisa digunakan oleh user setelah kualitas software sudah diuji, bebas *bugs* dan sesuai dengan *requirement* user.‍

1. **Maintenance**

Proses yang dilakukan setelah proses deployment berhasil dan user sudah mulai menggunakan aplikasi. Ada 3 aktivitas yang dilakukan pada fase ini yaitu perbaikan bug yang ditemukan oleh user, proses peningkatan versi aplikasi dan penambahan beberapa fitur pada aplikasi. Fase ini memiliki fokus utama untuk memastikan sistem selalu bekerja sesuai kebutuhan awal *requirement* software.

**2.2.2 Javascript**

Dikutip dari laman developer.mozilla.org yang diakses pada 13 oktober 2022 jam 10.30, JavaScript adalah bahasa pemrograman yang memungkinkan *Developer* untuk mengimplementasikan fitur kompleks, dinamis dan interaktif pada halaman web yang akan dibangun seperti menampilkan pembaruan konten yang dinamis, peta interaktif, animasi 2D/ Grafik 3D, gambar atau video, dll.

**2.2.2.1 Typescript**

Dikutip dari laman [www.typescriptlang.org](http://www.typescriptlang.org) yang diakses pada 13 oktober 2022 jam 10.43, TypeScript adalah bahasa pemrograman *open source.* Bahasa ini dibangun di atas bahasa pemrograman JavaScript yang memiliki fitur tambahan seperti tipe data, error notification yang mumpuni serta skalabilitas yang bagus. Hal tersebut memungkinkan Typescript dapat menghemat waktu di masa depan dengan menampilkan kemungkinan error lebih awal saat masa *development* dan dapat digunakan di dalam proyek dengan skala kecil maupun besar.

**2.2.2.2 Next Js**

*Next.js* merupakan sebuah *framework React js* yang digunakan untuk mengembangkan aplikasi website berbasis *Single Page Application*. *Next.js* dapat digunakan untuk membuat situs website yang cepat berkat salah satu fitur yang disediakan Next js yaitu SSG ( static site generation ). Selain itu Next js juga memungkinkan pengembang dapat membangun sebuah aplikasi lebih cepat berkat beberapa fitur bawaan *Next js* yaitu *Routing* yang sudah dikonfigurasi, Built-in CSS Support dan Fast Refresh.

**2.2.3 Database**

Dikutip dari laman dicoding.com yang diakses pada 13 oktober 2022 jam 10.28, D*atabase* atau basis data adalah pusat kumpulan data ataupun informasi pada sebuah sistem. Data data tersebut akan diolah oleh sistem untuk memberikan output kepada pengguna.

**2.2.3.1 Firebase**

Merupakan sebuah *platform* Backend as a Service) yang dikembangkan oleh Google. Firebase memudahkan dan membantu mempercepat para software engineer untuk mengembangkan aplikasi yang akan dibangun. Layanan firebase yang digunakan pada sistem yang akan dirancang antara lain firebase authentication untuk melakukan autentikasi pada aplikasi dan firebase firestore yang memungkinkan peneliti untuk menyimpan database aplikasi.

**2.2.4 Software Development Life Cycle**

Dikutip dari website ids.ac.id yang diakses pada 12 oktober 2022 jam 9.30, pengertian SDLC adalah tahapan pembuatan sebuah *software* untuk meningkatkan efektifitas produksi dan menghasilkan output kualitas yang terbaik dimana *software* yang dihasilkan sesuai dengan kebutuhan dan sesuai *timeline* yang telah ditentukan.

**2.2.4.1 Prototyping**

Menurut Rizky Aditya, Viktor Handrianus Pranatawijaya dan Putu Bagus Adidyana Anugrah Putra (2021) metode prototyping adalah metode pembangunan perangkat lunak yang memproduksi sebuah *software* yang belum sempurna, kemudian aplikasi tersebut harus di evaluasi dan di modifikasi kembali sesuai masukan yang diterima. Metode prototyping memiliki lima tahapan yang harus dikerjakan. Langkah-langkah dalam metode prototype:

1. Pengumpulan data awal dan Komunikasi untuk menganalisis kebutuhan pengguna.
2. Planning, yaitu tahapan perencanaan kebutuhan.
3. Design, tahapan pembuatan desain.
4. Pembentukan prototype, yaitu tahap pembuatan produk prototype.
5. Deployment, Delivery & Feedback, tahap dilakukannya evaluasi prototype berdasarkan *feedback* pengguna pada aplikasi yang telah di *deploy*.
6. Perbaikan prototype, proses perbaikan prototype aplikasi berdasarkan hasil evaluasi prototype.
7. Produksi akhir, yaitu tahap memproduksi aplikasi dengan menyempurnakan hasil perbaikan prototype sehingga dapat digunakan oleh pengguna.

**2.2.5 UML (Unified Modelling Language)**

Menurut Rosa A.S dan M. Shalahudin (2015:133), UML adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.

**2.2.5.1 Use Case Diagram**

Use case diagram adalah teknik untuk merekam persyaratan fungsional sebuah sistem, menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Use case diagram menekankan kepada “apa” yang diperbuat oleh sistem, dan bukan “bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Use case merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-create sebuah daftar belanja, dan sebagainya. Seorang atau sebuah aktor adalah sebuah entitas dapat 72 berupa manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu (Nur Hasanah, Fitria dan Sri Untari, Rahmania. 2020: 71-72).

berikut dijelaskan beberapa simbol yang terdapat di use case diagram beserta deskripsinya (Rosa dan Shalahuddin, 2014:156):

|  |  |
| --- | --- |
| simbol | deskripsi |
|  | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling tertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama Use Case |
|  | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang: biasanya dinyatakan menggunakan kata benda di awal frase nama aktor |
|  | Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan aktor |
|  | Relasi use case tambahan ke sebuah use case dinamakan use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya use case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan. |
|  | Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya. |

**2.2.5.2 Activity Diagram**

Activity diagram adalah diagram yang menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana alir berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Activity diagram digunakan untuk menggambarkan langkah langkah atau aktivitas pada suatu sistem. Pada setiap use case yang ada, maka terdapat paling sedikit satu activity diagram. Diagram ini menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Activity diagram dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis (Nur Hasanah, Fitria dan Sri Untari, Rahmania. 2020: 79-80).

berikut dijelaskan beberapa simbol yang terdapat di activity diagram beserta deskripsinya (Rosa dan Shalahuddin, 2014:162):

|  |  |
| --- | --- |
| simbol | deskripsi |
|  | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal |
|  | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja |
|  | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu |
|  | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu |
|  | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir |
|  | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi |

**2.2.5 Konselor**

Konselor merupakan seorang ahli di bidang konseling psikologi yang bertugas untuk memberikan konseling dengan mendengarkan cerita klien dengan seksama dan empati, lalu menganalisis masalah yang dialami klien untuk mencari solusi terbaiknya. Konselor hanya memberikan layanan konseling dan tidak memberikan diagnosis. Dan jika kondisi klien cukup buruk, seorang konselor bisa memberi rujukan kepada seorang psikolog ataupun psikiater. Dalam jenjang pendidikan, dengan gelar sarjana S1 psikologi seseorang sudah bisa menjadi seorang konselor.

**2.2.7 Konseling**

Konseling adalah hubungan pribadi yang dilakukan secara tatap muka antara dua orang, dimana konselor melalui hubungan itu dengan kemampuan-kemampuan khusus yang dimilikinya, menyediakan situasi belajar. Dalam hal ini konseling dibantu untuk memahami diri sendiri, keadaannya sekarang, dan kemungkinan keadaannya masa depan yang dapat ia ciptakan dengan menggunakan potensi yang dimilikinya, demi kesejahteraan pribadi nya maupun masyarakat. Lebih lanjut konseling dapat diartikan bagaimana kita memecahkan masalah-masalah dan menemukan kebutuhan-kebutuhan yang akan datang. (Tolbert, dalam Prayitno 2004 : 101).

**2.2.8 Redux Toolkit**

Dikutip dari laman resmi redux-toolkit.js.org yang diakses pada 21 Juni 2023 jam 11.30, Redux Toolkit adalah sebuah *toolset state management* yang resmi, *powerful* dan efisien untuk pengembangan menggunakan redux. Toolset ini mencakup utilitas untuk menyederhanakan kasus penggunaan umum seperti penyiapan *store*, membuat *reducer*, Memberikan default yang baik untuk penyiapan *store*, dan menyertakan addon bawaan Redux yang paling umum digunakan.

**2.2.9 Tailwind**

Dikutip dari laman resmi tailwindcss.com yang diakses pada 21 Juni 2023 jam 11.30, Tailwind adalah sebuah kerangka kerja CSS yang mengutamakan utilitas yang dikemas dengan bentuk kelas pada html seperti flex, pt-4, text-center, dan rotate-90 yang dapat disusun untuk membuat desain apa pun, langsung di markup atau HTML Anda. Kelas utilitas membantu Anda bekerja langsung dalam sistem alih-alih mengotori stylesheet Anda dengan kode yang cukup banyak. Tailwind membuatnya mudah untuk tetap konsisten dengan pilihan warna, spasi, tipografi, bayangan, dan segala hal lain yang membentuk sistem desain yang dirancang dengan baik.

**2.2.10 Jest**

Dikutip dari laman resmi jestjs.io yang diakses pada 21 Juni 2023 jam 11.30, Jest adalah sebuah kerangka Pengujian *JavaScript* yang menyenangkan dengan fokus pada kesederhanaan. Ia bekerja dengan proyek menggunakan: Babel, TypeScript, Node, React, Angular, Vue dan banyak lagi. Jest bertujuan untuk bekerja secara out of the box, bebas konfigurasi, di sebagian besar proyek JavaScript. Serta Pengujian diparalelkan dengan menjalankannya dalam prosesnya sendiri untuk memaksimalkan kinerja pengujian.

**2.2.11 Node.Js**

Dikutip dari laman resmi nodejs.org/en/about yang diakses pada 21 Juni 2023 jam 11.30, Jest adalah sebuah runtime JavaScript berbasis event asinkron, Node.js dirancang untuk membangun aplikasi jaringan yang dapat diskalakan dengan mudah. Node.js memiliki desain yang mirip seperti bahasa pemrograman Ruby dan bahasa pemrograman Python. HTTP adalah first-class citizen di Node.js, dirancang dengan mempertimbangkan streaming dan latensi rendah. Ini membuat Node.js sangat cocok untuk pondasi perpustakaan web atau kerangka kerja.

**2.2.12 Clean Architecture**

Clean Architecture adalah sebuah pendekatan desain perangkat lunak yang bertujuan untuk memisahkan dan membangun sistem yang terstruktur dengan tingkat ketergantungan yang rendah antara komponen-komponennya. Tujuan utama dari Clean Architecture adalah menciptakan sistem yang dapat dipertahankan, mudah dimengerti, dan mudah diuji.

Clean Architecture mengedepankan prinsip pemisahan antara kode bisnis inti (core business logic) dengan detail teknis implementasinya. Arsitektur ini mengusulkan pembagian sistem menjadi beberapa lapisan (layers), dimana setiap lapisan memiliki tanggung jawab yang jelas dan terpisah satu sama lain. Lapisan-lapisan tersebut umumnya mencakup:

1. Domain Layer: Merupakan lapisan terdalam yang berisi logika bisnis inti (core business logic) dan aturan domain. Lapisan ini biasanya tidak bergantung pada lapisan lainnya dan berfungsi sebagai inti dari sistem.

2. Application Layer: Merupakan lapisan yang mengatur interaksi antara lapisan domain dengan lapisan-lapisan lainnya. Lapisan ini berfungsi sebagai mediator antara berbagai komponen dan mengatur alur eksekusi aplikasi.

3. Infrastructure Layer: Merupakan lapisan yang bertanggung jawab atas implementasi teknis seperti koneksi dengan database, komunikasi jaringan, penyimpanan data, dan sebagainya. Lapisan ini merupakan bagian yang paling bergantung pada detail teknis dan dapat berinteraksi dengan perangkat keras dan perangkat lunak eksternal. Pendekatan Clean Architecture memiliki beberapa manfaat, antara lain:

1. Memisahkan kode bisnis inti dari detail teknis sehingga memudahkan pengujian dan pemeliharaan.
2. Mengurangi ketergantungan antar komponen sehingga memungkinkan perubahan dan pengembangan lebih fleksibel.
3. Memungkinkan penggunaan pola-pola desain seperti Dependency Injection untuk meningkatkan skalabilitas dan modularitas.
4. Memperjelas pembagian tanggung jawab dalam pengembangan perangkat lunak.

Clean Architecture bukanlah suatu framework atau teknologi spesifik, melainkan sebuah konsep dan prinsip desain yang dapat diimplementasikan dalam berbagai bahasa pemrograman dan kerangka kerja. Tujuan utamanya adalah untuk menghasilkan sistem perangkat lunak yang lebih terstruktur, mudah dimengerti, dan dapat beradaptasi dengan perubahan kebutuhan bisnis.

**2.2.13 Preload**

Preload adalah sebuah teknik atau mekanisme yang digunakan untuk memuat (load) sumber daya seperti gambar, video, atau file lainnya sebelum dibutuhkan atau ditampilkan secara aktif oleh pengguna. Tujuan utama dari preloading adalah untuk meningkatkan performa dan pengalaman pengguna dengan meminimalkan waktu penundaan saat memuat konten. Ketika sebuah halaman web atau aplikasi dimuat, biasanya ada beberapa elemen (seperti gambar, audio, atau video) yang perlu diunduh dari server sebelum dapat ditampilkan sepenuhnya. Dalam beberapa kasus, elemen-elemen ini mungkin memiliki ukuran yang besar atau memerlukan waktu yang cukup lama untuk diunduh, sehingga dapat mengakibatkan penundaan dalam tampilan dan responsivitas halaman. Dengan menggunakan teknik preload, sumber daya yang diperlukan untuk halaman tersebut dapat dimuat di latar belakang atau sebelumnya, bahkan sebelum pengguna secara aktif berinteraksi dengan elemen tersebut. Dengan cara ini, ketika pengguna mencoba mengakses atau menavigasi ke elemen tersebut, sumber daya tersebut telah tersedia di cache atau telah diunduh, sehingga mempercepat tampilan dan responsivitas.

**2.2.14 SSG ( Static Site Generation )**

Static Site Generation (SSG) adalah suatu pendekatan dalam pengembangan situs web dimana halaman web yang ditampilkan kepada pengguna dibangun dan di-generate sebelumnya sebagai file statis pada tahap pembangunan, dan kemudian disajikan secara langsung kepada pengguna saat mengakses situs tersebut. Dalam SSG, proses pembangunan dan pengelolaan konten dilakukan di waktu kompilasi (build time) bukan saat permintaan (runtime). Kontras dengan pendekatan server-side rendering (SSR) atau client-side rendering (CSR), di mana halaman web dibangun secara dinamis saat permintaan pengguna, SSG menghasilkan halaman web yang telah siap secara statis. Ini berarti konten dan tampilan halaman telah ditentukan sebelumnya dan disimpan sebagai file HTML, CSS, dan JavaScript yang siap untuk diunggah ke server dan disajikan tanpa memerlukan komputasi server tambahan.

**2.2.15 Lighthouse**

Dikutip dari web resmi developers.google.com, Lighthouse adalah alat audit open source otomatis untuk meningkatkan kualitas halaman web. Alat ini memberikan cara yang jelas untuk meningkatkan kualitas situs dengan memungkinkan developer menjalankan audit untuk performa, aksesibilitas, kompatibilitas progressive web app, dan lainnya.