BAB 2
LANDASAN TEORI

2.1 Tinjauan Tempat Penelitian

Teamstar adalah aplikasi gaya hidup multi-profil dan multi-platform yang menyasar komunitas STEM. Tujuannya adalah memberikan tempat tunggal bagi jaringan, partisipasi, dan pertumbuhan profesional bagi komunitas STEM. Dengan fitur seperti home feed yang familier, showcase untuk konten promosi, channel berita, forum, dan opsi pencarian yang kuat, Teamstar dirancang untuk berguna, adaptif, dan menyenangkan. Selain itu, aplikasi ini juga mencakup elemen seperti repository white paper, forum peer review, dan job board untuk membantu menghubungkan calon dengan posisi pekerjaan, membuatnya menjadi alat yang sempurna bagi profesional dalam bidang STEM. Dengan fokus pada komunitas dan kerjasama, Teamstar adalah wajib bagi siapa pun yang ingin memajukan kariernya dalam bidang STEM [1].

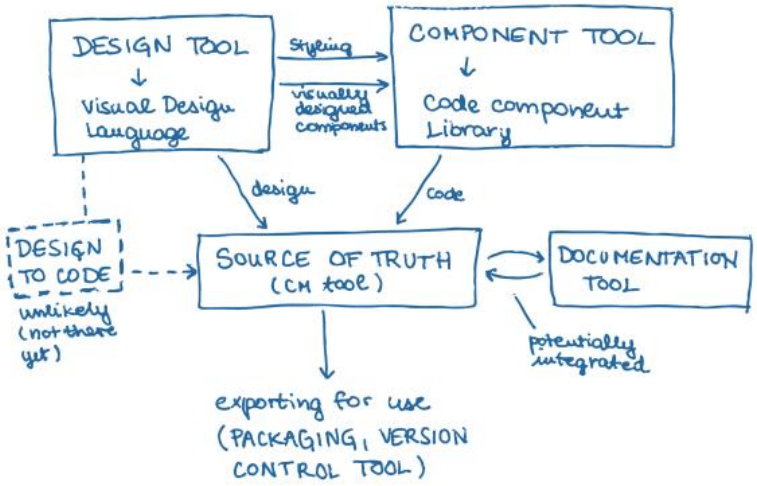
Teamstar menawarkan solusi yang inovatif dan membantu profesional STEM untuk membangun jaringan, menemukan peluang, dan memperluas wawasan mereka. Aplikasi ini memiliki fitur yang user-friendly dan mempermudah interaksi antar pengguna, sehingga memungkinkan mereka untuk berkolaborasi dan berbagi informasi. Fitur seperti showcase, channel berita, dan job board membantu pengguna untuk mempromosikan konten mereka, menemukan informasi terbaru, dan mencari pekerjaan baru. Selain itu, aplikasi ini juga mencakup elemen seperti forum dan repository white paper, yang membantu profesional untuk berdiskusi dan berbagi pengetahuan. Dengan semua fitur ini, Teamstar menjadi alat yang sempurna bagi profesional STEM untuk memajukan karier mereka dan terus belajar.

2.2 Definisi *Design System*

Design System adalah sekumpulan pola yang saling terkait dan melibatkan tahapan-tahapan sistematis dan kerja sama untuk mempercepat dan mempermudah proses desain dan pengembangan dalam mencapai tujuan produk digital. Tujuannya adalah untuk membuat standar desain dan memberikan dokumentasi, termasuk

perpustakaan kode, perpustakaan pola, dan panduan gaya, sehingga bisa digunakan oleh semua tim dalam mengembangkan solusi yang dapat digunakan kembali. Komponen-komponen *UI* dalam *Design System* bisa digunakan ulang untuk membantu tim pengembang membangun tampilan yang kompleks, kuat, dan bisa diakses di seluruh projek, karena developer dan desainer bekerja sama dalam menentukan komponen-komponen UI [3]. *Design System* membawa tata tertib dan konsistensi ke produk digital. Mereka membantu melindungi merek, meningkatkan pengalaman pengguna, dan meningkatkan kecepatan dan efisiensi dalam membuat dan membangun produk. menjadi sumber kebenaran dan sistem catatan untuk keputusan desain kita. memegang kita pada standar tinggi, memastikan tim selalu berada pada halaman yang sama, dan membantu dalam proses onboarding anggota tim baru. Mereka mencatat mengapa, kapan, di mana, dan bagaimana. [5]

Struktur *Design System* perlu memiliki organisasi yang sangat jelas, baik dalam bentuk terpusat atau tersebar. Memilih bentuk terpusat berarti bahwa pengelolaan *Design System* dilakukan oleh satu kelompok orang yang menentukan aturan dan pola dan memiliki wewenang atas semua pembaruan Sistem. Kepemilikan ini memastikan bahwa sistem akan dipelihara dan berkembang sementara tim ini juga memfasilitasi pekerjaan untuk tim lain. Dalam sistem tersebar, orang yang menggunakannya juga berkontribusi pada *Design System* dan membantu memelihara dan berkembang. Ini membuat adopsi lebih mudah dan memberikan lebih banyak otonomi dan daya tahan [9].



Gambar 2.1 Struktur Desain sistem

2.3 In-Depth Interview

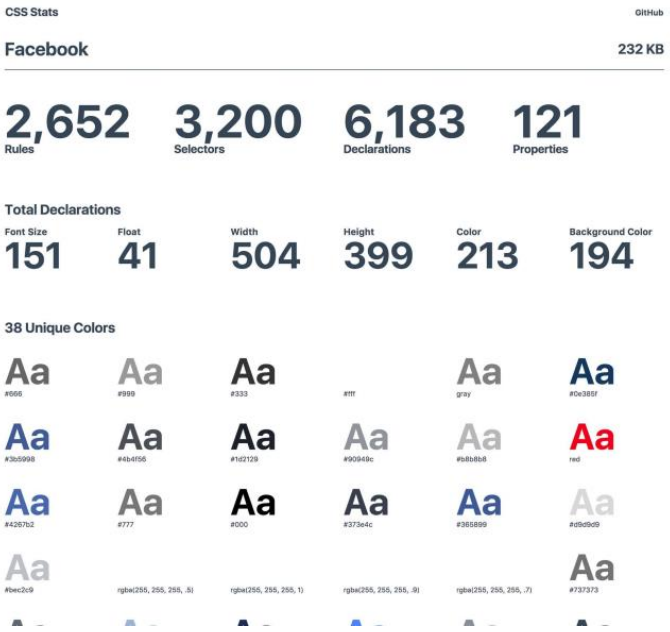
Indepth interview adalah teknik riset yang digunakan dalam UX untuk memahami perilaku dan harapan pengguna dengan lebih dalam. Ini dilakukan melalui wawancara tatap muka atau telepon dengan responden terpilih, dimana mereka dapat berbicara secara bebas dan mendalam tentang masalah atau topik tertentu. Tujuannya adalah untuk memperoleh informasi yang lebih dalam dan pemahaman yang lebih baik tentang apa yang memotivasi, membatasi, atau mempengaruhi perilaku pengguna. (Steve Krug, "Don't Make Me Think").

"Indepth interview memberikan pemahaman mendalam tentang bagaimana orang benar-benar berpikir dan bertindak dalam situasi tertentu. Ini membantu untuk menemukan solusi yang lebih baik untuk masalah yang mereka hadapi, dan membuat desain produk yang lebih sesuai dengan kebutuhan dan harapan pengguna"

2.4 Design Audit

Audit visual (atau yang juga dikenal dengan "inventori tampilan") adalah cara untuk menemukan ketidakkonsistenan pada produk digital. Anda melakukan pemindaian pada produk dan membuat inventaris dari seluruh aspek dari tampilan

visual, mengidentifikasi di mana masalah terjadi, dan memetakan skala masalah. Dengan informasi ini, Anda dapat merencanakan untuk memperbaiki masalah-masalah kecil dalam jangka pendek dan memformulasikan rencana untuk memperbaiki masalah besar dalam jangka panjang. Ini juga memberikan bahan yang sangat baik untuk memasarkan pekerjaan *design system* Anda [10].



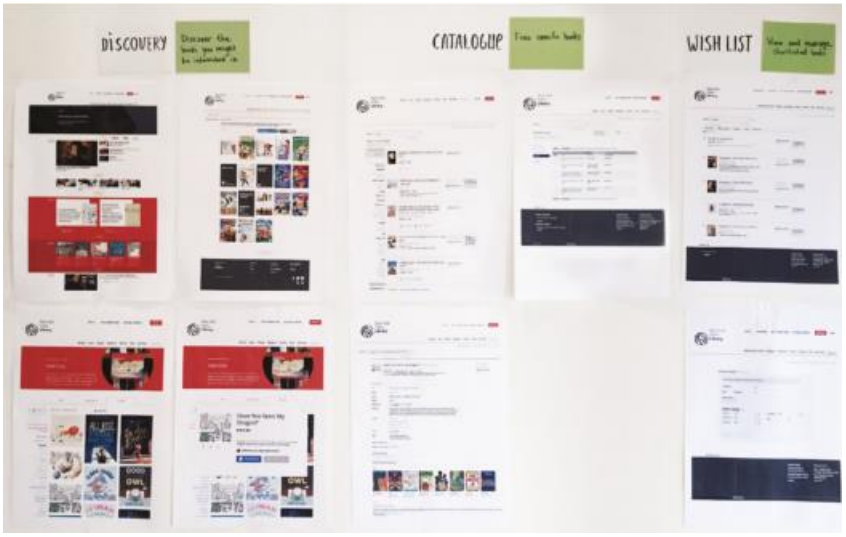
Gambar 2.2 Desain Audit Facebook

"Desain Audit pada *UI* adalah langkah penting dalam memastikan bahwa tampilan pengguna produk Anda memenuhi standar dan memuaskan kebutuhan pengguna. Ini membantu untuk mengidentifikasi masalah sebelum mereka menjadi masalah besar dan memastikan bahwa produk Anda terus memenuhi kebutuhan pengguna yang berubah[11].

Dengan demikian, desain audit pada *UI* adalah alat yang sangat berguna bagi desainer dan tim produk untuk memastikan bahwa tampilan mereka memuaskan dan memenuhi standar. Ini membantu mereka untuk memperbaiki masalah dan memastikan bahwa produk mereka dapat digunakan dengan efisien dan efektif.

2.4.1 Identifikasi Key Behavior

Identifikasi key behaviour adalah upaya untuk mengetahui kebutuhan dan perilaku pengguna saat menggunakan produk. Ini dilakukan dengan melakukan serangkaian aktivitas yang melacak setiap segmen dalam journey pengguna. Dalam proses identifikasi key behaviour, aktivitas yang dilakukan meliputi identifikasi perilaku pengguna pada halaman informasi dalam aplikasi. Kemudian, informasi tersebut dikategorikan dan diberi label sesuai dengan fungsi dari halaman tersebut untuk mempermudah pemahaman akan perilaku pengguna.



Gambar 2.3 Key Behavior

2.4.2 Break Behavior into Action

Tahap ini berkaitan dengan membagi aktivitas tindakan menjadi lebih detail dan spesifik. Ini dilakukan dengan memperhatikan setiap bagian pada suatu halaman. Tujuan dari tahap ini adalah untuk mengidentifikasi aktivitas tindakan yang sering terulang dan memiliki pola yang sama. Hal ini bertujuan untuk mengetahui elemen *user interface* yang bisa digunakan kembali (*reuse*), sehingga dapat mempermudah pengembangan produk dan meminimalisir duplikasi kode[3].

Gambar 2.4 Behaviour Into Action

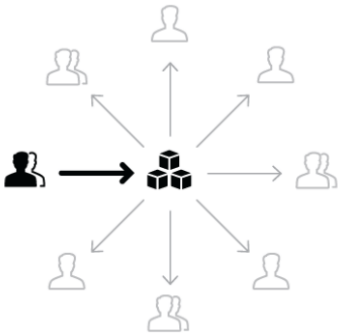
2.5 Governance Model

Dalam membangun *design system*, diperlukan tim yang terdiri dari berbagai role yang memiliki spesialisasi masing-masing, seperti desainer, *frontend engineer*, *researcher*, *information architect*, dan lain-lain. Setiap role memiliki perspektif dan pandangan yang berbeda dan dapat memberikan sumbangsih bagi *design system* [11].

Nathan Curtis menyatakan bahwa ada beberapa model tim dalam membangun *design system* yang memperhitungkan peran penting setiap anggota tim. Model tim tersebut membantu mengarahkan setiap anggota agar bisa memberikan sumbangsih terbaik dalam pembangunan *design system*, seperti desainer yang bertugas membuat definisi elemen visual, dan frontend engineer yang membangun kode modular yang efisien.

2.5.1 The Solitary Team Model

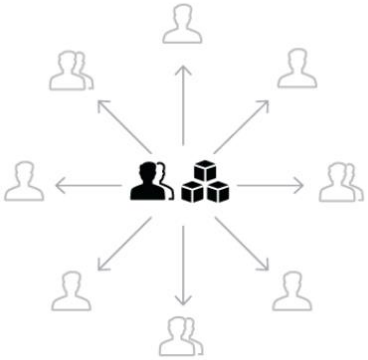
Model ini hanya ada seorang individu yang bertindak sebagai pemimpin atau "ujung tombak" dalam pengelolaan *design system*. Ia berperan sebagai penghubung antara setiap role yang ada dalam tim.



Gambar 2.5 The Solitary Model

2.5.2 The Centralized Team Model

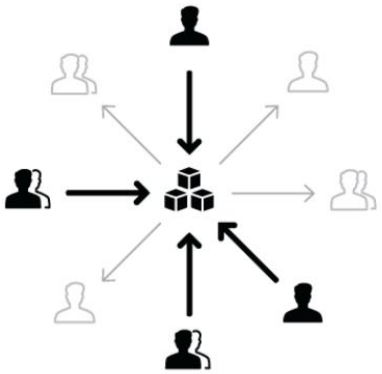
Suatu *Design System* dengan model terpusat adalah dimana suatu kelompok orang yang dedikasi bertanggung jawab untuk mengurus dan memelihara sistem tersebut. Model ini seringkali digunakan oleh perusahaan yang berorientasi pada desain seperti Airbnb dan Apple, misalnya (Anne, 2015). Tim ini membuat sebuah perpustakaan pola dan memelihara *Design System* dengan baik, tetapi mereka tidak selalu terlibat dalam desain produk itu sendiri. Ini melindungi tim dari pandangan yang terpengaruh oleh permintaan yang berlebihan dan arah kreatif yang tidak konsisten. Namun, suatu *Design System* dengan model terpusat dapat mengakibatkan kurangnya kesadaran tentang kebutuhan pengguna akhir dan melupakan tantangan harian tim desain.



Gambar 2.6 The Centralized Model

2.5.3 The Federated Team Model

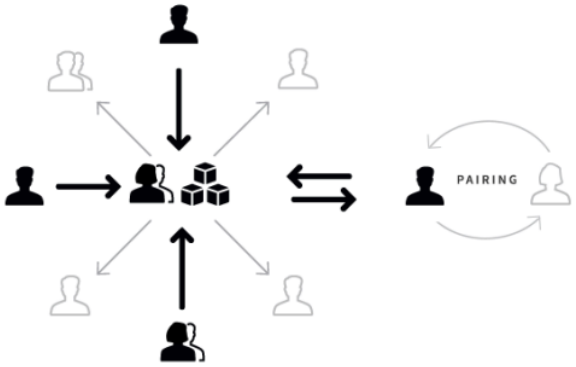
Design System yang beroperasi dengan *model federasi* adalah sistem di mana setiap anggotanya berkontribusi pada sistem. Setiap orang dalam tim memiliki hak dan tanggung jawab untuk bekerja pada sistem, yang memungkinkan sistem untuk dibangun berdasarkan kebutuhan aktual yang diterjemahkan dari pengguna akhir dan tim produksi. Ini berarti bahwa sistem mampu melayani tujuan yang tepat. Namun, kelemahan potensial adalah bahwa tanpa adanya anggota tim tertentu yang bertanggung jawab untuk memelihara dan meningkatkan sistem, perhatian dapat berpindah ke tugas lain, yang dapat menghentikan evolusi sistem dari waktu ke waktu.



Gambar 2.7 The Federated Team Model

2.5.4 The Cyclical Team Model

*The cyclical team mode*l adalah gabungan antara model *Centralized* dan *Federated*. Dalam model ini, setiap anggota tim memiliki hak dan tanggung jawab untuk berkontribusi dalam membangun *design system*, seperti menambahkan komponen baru dan lainnya. Namun, terdapat juga tim khusus yang bertugas mengelola seluruh *design system*. Mereka memiliki peran penting dalam memastikan bahwa *Design System* berjalan dengan baik dan sesuai dengan tujuan yang ditetapkan. Tim ini bertanggung jawab untuk memelihara dan meningkatkan sistem, sehingga dapat terus berkembang dan memberikan hasil terbaik bagi penggunanya.



Gambar 2.8 Cyclical Team Model

2.6 Brand

Brand adalah identitas visual dan persepsi yang melekat pada sebuah produk, jasa, atau perusahaan. Dalam bidang desain, brand sangat penting karena memainkan peran besar dalam membangun citra dan reputasi sebuah perusahaan. Desainer bertanggung jawab untuk memastikan bahwa semua produk dan komunikasi visual perusahaan terlihat konsisten dan mewakili merek dengan baik. Neumeier menjelaskan bahwa merek tidak hanya terdiri dari elemen visual seperti logo dan tipografi, tetapi juga termasuk persepsi dan harapan konsumen terhadap produk atau jasa yang ditawarkan. Oleh karena itu, strategi pemasaran harus didesain dengan baik dan konsisten untuk membantu membangun persepsi positif dari merek. Brand mencangkup aspek pemasaran dan desain, termasuk penentuan tujuan, pemahaman target pasar, dan penerapan prinsip-prinsip desain yang efektif, untuk membantu perusahaan membangun merek yang kuat dan berkesan. [13]

2.6.1 Brand Identity

Brand identity adalah bagian dari brand yang berkaitan dengan bagaimana mereka terlihat dan terasa. Ini meliputi semua elemen visual dan kesan yang ditinggalkan dengan pengguna, termasuk logo, typografi, warna, bahan visual, dan interaksi digital. Brand identity adalah cara bagaimana perusahaan

mempresentasikan diri mereka kepada audiens mereka dan membedakan diri dari pesaing. [3]

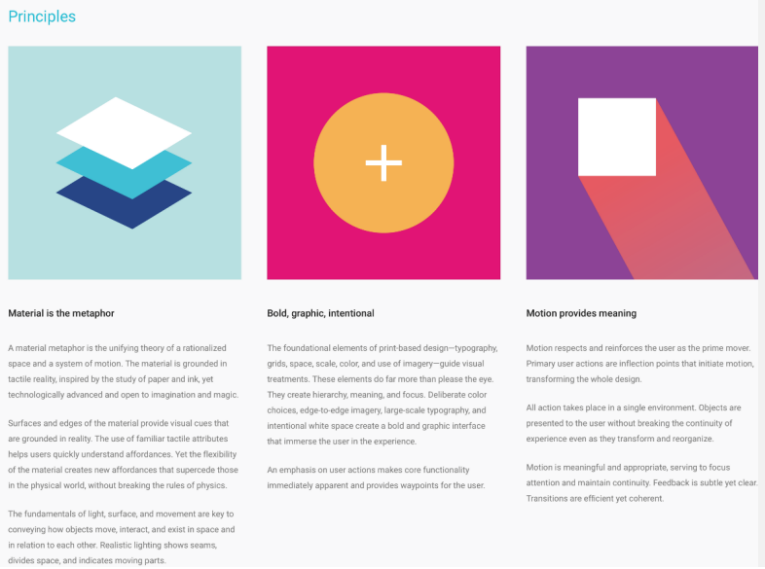
"Brand identity adalah jantung dari brand. Ini adalah bagaimana mereka terlihat dan terasa. Ini adalah bagaimana mereka membedakan diri dari orang lain dan membangun koneksi emosional dengan audiens mereka. Brand identity harus terintegrasi dengan strategi dan filosofi brand yang lebih luas untuk memastikan konsistensi dan efektivitas yang maksimal" (Brad Frost).

Dengan demikian, brand identity memainkan peran penting dalam membangun merek yang kuat dan konsisten. Ini membantu perusahaan untuk membangun koneksi yang kuat dengan audiens mereka dan memastikan bahwa mereka dikenali dan diingat dengan baik.

2.6.2 Prinsip desain

Sebelumnya telah dijelaskan bahwa brand identity memberikan identitas suatu produk dengan mudah. Namun, brand identity saja tidak cukup sehingga dibutuhkan suatu prinsip desain dalam merancang suatu produk yaitu prinsip desain. Prinsip desain bertujuan untuk mengartikulasikan arah, filosofi, dan pendekatan desain umum untuk suatu projek atau produk digital [12]

Prinsip desain adalah pengembangan dari identitas merek, fokus pada menerjemahkan tujuan produk ke dalam desain dan pengembangan. Sebagai contoh, jika tujuannya adalah memberikan layanan yang lebih efisien waktu, interaksi yang diperlukan harus dibatasi dan sangat sederhana. Prinsip-prinsip tersebut juga bisa lebih spesifik untuk produk daripada identitas merek secara umum, tetapi mereka harus menjadi komitmen tim untuk mencapai hasil yang kohesif [9].



Gambar 2.9 Material Prinsip desain

Prinsip-prinsip ini tidak bisa dibandingkan dan diukur. Mereka bisa digunakan dalam proyek untuk mengungkapkan arah tertentu untuk jangka waktu tertentu untuk mencapai tujuan tertentu. Di sisi lain, prinsip desain juga bisa digunakan untuk tujuan jangka panjang yang merefleksikan nilai-nilai bersama perusahaan yang berlangsung lama. Tergantung pada ukuran organisasi, setiap tim mungkin memiliki prinsip atau panduan spesifik mereka sendiri [13]. Jika Anda memiliki prinsip desain seperti "*Insightful* lebih penting dari efisien", Anda mungkin akan memilih visualisasi data yang menampilkan informasi baru, bahkan lebih penting dari visualisasi standar dalam produk Anda yang lebih cepat untuk diakses. [14]

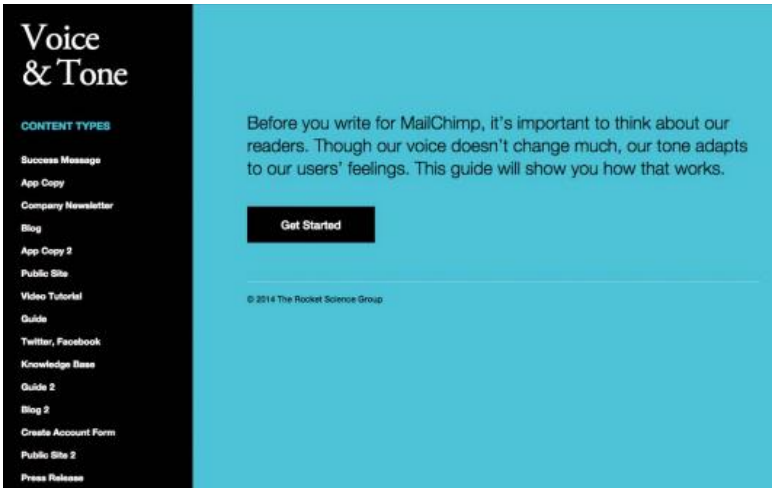
Gambar 2.10 Prinsip desain

2.6.3 Tone and voice

Voice atau suara adalah aspek utama identitas produk, sehingga disebut dengan *brand voice*, sehingga panduan *brand identity* menyertakan beberapa

referensi *brand voice.* Namun, *voice* dalam identitas produk saja tidak cukup, sehingga *voice* harus selalu terikat dengan *tone* [3]

Panduan suara dan tone memasuki hal-hal teknis dengan menguraikan bagaimana suara dan tone perusahaan harus berubah di berbagai skenario. Panduan suara dan tone yang brilian dari MailChimp menjelaskan bagaimana tone merek berubah di berbagai jenis konten, sehingga ketika kartu kredit pengguna ditolak, penulis tahu untuk beralih dari tone suara yang umumnya lucu dan bermain-main dan mengadopsi tone yang lebih serius [3].



Gambar 2.11Tone & Voice

2.6.4 Shared Language

Bahasa sangat penting saat bekerja sama dalam sebuah tim. Saat tim bekerja membuat suatu produk, bahasa yang sama harus diterapkan oleh setiap anggota tim agar dapat memahami satu sama lain. Tanpa bahasa yang sama, masing-masing anggota tim akan memiliki pandangan yang berbeda dan memiliki model pemikiran yang berbeda tentang apa yang ingin mereka capai [9].

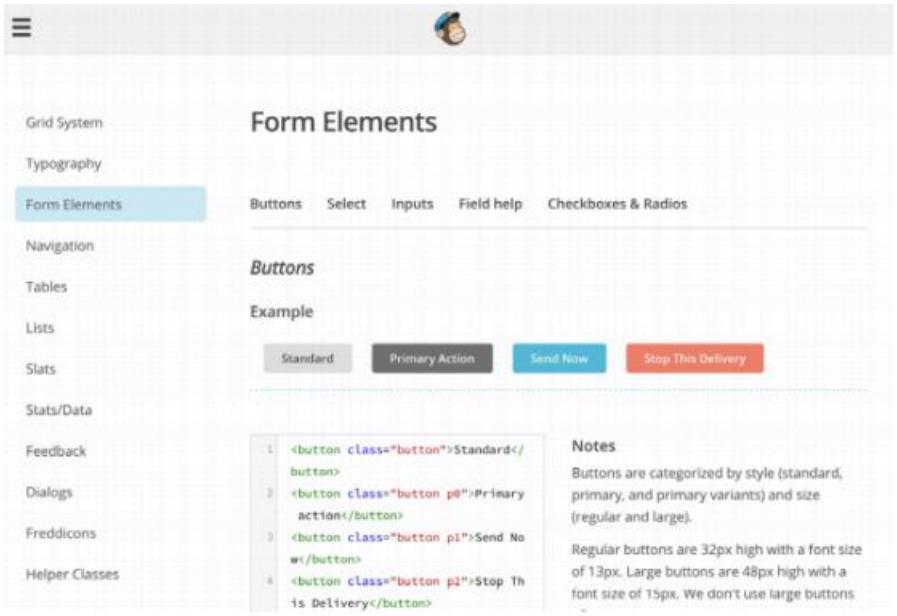
Menurut Abby Covert, sebelum desainer mulai bekerja pada tampilan, bahasa yang sama harus diterapkan oleh setiap anggota tim melalui diskusi, pengecekan, dan dokumentasi. Hal ini dilakukan untuk menentukan konsep dan istilah-istilah bahasa yang akan digunakan oleh tim dalam membahas keputusan desain. Ini dilakukan untuk memastikan bahwa setiap anggota tim memiliki pemahaman yang sama tentang hal-hal tersebut. Abby

Bahasa yang sama diciptakan untuk menghindari kesalahpahaman antara anggota tim. Contohnya adalah penggunaan komponen button, setiap orang mungkin memiliki pemahaman yang berbeda tentang hal tersebut. Tim harus tahu mengapa dan bagaimana menggunakan button, dalam situasi apa button tersebut digunakan, dan tujuan dari penggunaan button itu sendiri. Seperti halnya elemen yang disebut "sequence". Dengan menggunakan istilah ini, tim bertujuan untuk berkomunikasi dengan pengguna bahwa tindakan yang diambil oleh pengguna harus dilakukan dalam urutan yang spesifik.

2.6.5 Pattern Library

Pattern library adalah kumpulan pola yang digunakan untuk berkomunikasi dan meningkatkan keputusan desain. Ini termasuk solusi yang dapat digunakan kembali untuk masalah yang berfokus pada interaksi dan komponen UX. Secara umum, pattern library adalah kumpulan *UI* Patterns abstrak, termasuk UI-patterns.com dan UIPatterns.io. Di sini, Anda akan menemukan bahwa satu pola mungkin diterangkan dengan puluhan contoh beragam penggunaannya dalam dunia nyata.

Meskipun demikian, Anda akan menemukan istilah "pattern library" digunakan untuk menggambarkan pustaka internal dalam satu organisasi, yang seringkali lebih spesifik dan disesuaikan dengan kebutuhan satu entitas. Di sini, Anda akan menemukan bahwa setiap pola hanya memiliki satu representasi visual utama saat diterapkan pada organisasi (jika memiliki representasi visual sama sekali).



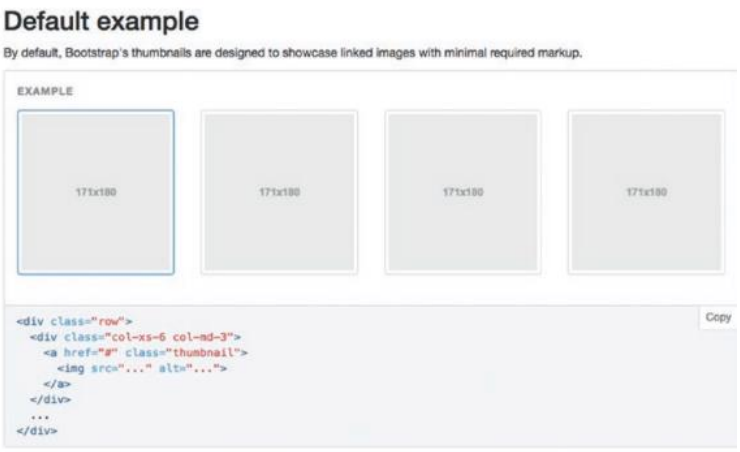
Gambar 2.12 Mailchimp Pattren Library

2.7 Design Guidelines dan Visual Language

Panduan desain yang menjelaskan bahasa visual sebagai hal yang berbeda dari brand atau panduan gaya visual biasanya membahas topik konseptual. Material Design, misalnya, menentukan metafora yang konsisten untuk digunakan dalam desain dalam gaya Material Design.

2.8 UI Patterns

Pola tampilan pengguna (*UI Patterns*) dapat ditemukan di situs web ranah digital, aplikasi, aplikasi seluler asli, dan perangkat lunak atau perangkat lainnya. Pola-pola ini menyediakan bahasa untuk mendiskusikan desain interaktif. Mereka mengusulkan fungsi, interaksi dan tujuan. *UI Patterns* mendokumentasikan bagian yang dapat digunakan kembali dari tampilan satu tujuan. Untuk memahami *UI* Patterns (dan perbedaannya dengan komponen), mari kita jelajahi beberapa ide dari kerangka kerja *UI*, Bootstrap. Mari kita lihat terlebih dahulu pada komponen thumbnails (https://getbootstrap.com/docs/3.4/components/#thumbnails) [14]



Gambar 2.13 UI Patterns

UI Patterns membandingkan pendekatan, pertimbangan penyulingan, dan pencapaian desainer untuk Anda. Pengetahuan tentang formula a Anda dapat memperoleh manfaat dengan memahami keputusan yang dibuatnya tentang kebijaksanaan yang muncul dari seluruh generasi dan industri yang dibawa oleh standar-standar ini tanpa menciptakan kembali roda. Kecil dan dapat digunakan kembali Solusi *UI* yang ditemukan dalam pola-pola ini dapat disusun secara bersamaan dengan menciptakan pengalaman yang kohesif dan intuitif yang beresonansi dengan orang-orang. Mari kita lihat beberapa manfaat lain dari mempelajari pola *UI*

2.8.1 Design Efficiency

Mengetahui pola membantu Anda mendesain secara efektif dengan pengenalan cepat alat terbaik untuk pekerjaan tersebut, memahami nilai dari solusi yang berbeda, dan menyelesaikan sejumlah besar masalah sekaligus. Sebagai contoh, kotak pencarian pelengkapan otomatis dapat membantu pengunjung situs Anda menavigasi konten situs Anda, mengidentifikasi istilah yang mereka cari tanpa mengetahui nama atau ejaannya secara pasti, dan memilih hasil setelah memasukkan beberapa karakter tanpa membuang energi untuk mengetik seluruh istilah pencarian. Dengan mempelajari bagian dari pola *UI* pelengkapan otomatis, Anda dapat lebih mudah mengenali kapan Anda harus menggunakannya, seperti semua standar tampilan pengguna.

2.8.2 Consistency and familiarity

Consistency adalah unsur utama dalam desain, desain yang konsisten adalah desain yang intuitif. Hal ini sangat berguna dalam mengembangkan produk. Consistency bertujuan untuk meningkatkan kemampuan belajar ketika pengguna menemukan elemen desain yang serupa dan memiliki fungsi yang sama. Consistency pada desain dapat mentransfer pengetahuan pengguna kepada konteks baru dan belajar hal-hal baru tanpa mengurangi kualitas experience pengguna. Consistency sangat penting untuk mengurangi load coginitive pada tampilan

Dengan pola yang sudah dikenal, Anda bisa meningkatkan prediktabilitas. Seret dan lepas Anda dapat langsung memanipulasi objek dengan menyeret dan melepas Penggunaan seret dan lepas yang umum adalah mengunggah gambar melalui seret gambar dari sistem file lokal komputer Anda ke tampilan area tujuan. Sebagian besar aplikasi yang harus dilakukan memungkinkan Anda untuk menyeret dan melepaskan item yang harus dilakukan dan mengubah urutannya. atau pindah ke daftar lain. Tampilan seret dan lepas yang lebih canggih dapat melalui web, lebih mungkin dipahami oleh manusia bagaimana cara menanganinya. Sebagai contoh, GitHub menjelaskan bahwa Anda dapat melampirkan file ke komentar dan Anda dapat melakukannya dengan beberapa cara (seret, pilih, tempel).

Metode	Keterangan
Centralized Consistency	Sebuah metode consistency dengan membuat grup pusat dalam menetapkan standar perusahaan dan meminta developer untuk mematuhinya.
User-defined Consistency	Metode ini berfokus pada pendefinisian awal dari pengguna, misalnya suatu perusahaan membuat sebuat standar yang disesuaikan dengan perangkat lunak dari vendor dan menyesuaikannya agar konsisten.
Exemplary Applications	Metode untuk membuat standar desain yang mengacu kepada desain tampilan yang sudah familiar dengan pengguna, sehingga ketika mengembangkan suatu aplikasi, maka dilakukan perancangan dan meniru aplikasi yang sudah umum.
Promotion of Consistencty	*Developers* disadarkan akan perlunya konsistensi melalui *"Art Exibits"* dari tampilan yang konsisten, yang artinya dilakukan edukasi dan pemaparan tentang pentingnya konsistensi dalam membangun aplikasi.
Shared Code	Program dibuat konsisten karena *developer* menggunakan kode yang sama untuk mengimplementasikan desain tampilan. Hal inidapat dilakukan dengan menggunakan *tools* seperti *code library*, UIMS atau pustaka kode yang tersentralisasi.
Hardware support for consistency	Memastikan bahwa semua pengguna memiliki perangkat yang sama sehingga semua aplikasi akan menggunakan perangkat tersebut.

Metode	Keterangan
Style-based consistency	*Style* tampilan ditentukan dan diterapkan secara terpisah dengan konten tampilan. Dengan cara yang sama seperti ketika menggunakan *style sheet* dalam pengolahan kata.
Model analysis	Sebuah metode berupa model perkiraan yang dibuat untuk interaksi pengguna terhadap pengguna. Misalnya menggunakan sistem produksi atau

2.8.3 Consistency and reuse

Karena sifatnya yang berulang, pola memungkinkan Anda untuk menggunakan kembali solusi desain Anda baik secara visual maupun di dalam kode. Pengulangan visual memungkinkan Anda untuk menciptakan konsistensi dan memprediksi tampilan Anda, menciptakan pengalaman belajar bagi pengguna Anda Penggunaan ulang kode juga menghemat waktu, memungkinkan Anda untuk memperbaiki dan meningkatkan fitur yang ada alih-alih membuat ulang fitur baru setiap saat (namun jika ada yang serupa). Pola ini berisi banyak solusi desain untuk memecahkan masalah saat pemrograman kode itu keputusan;

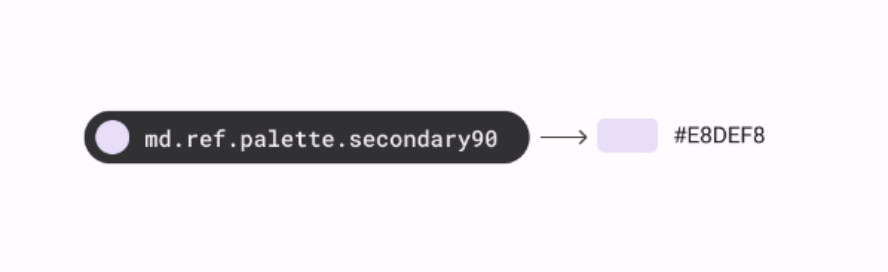
2.8.4 Communicating decisions

Sebagai alat komunikasi, Patterns dapat melibatkan para pemangku kepentingan dan mengapresiasi para mitra dalam solusi. Dapatkah Anda menjelaskan alasannya? Solusi tersebut adalah yang terbaik untuk konteks tertentu. Anda mengetahui tujuan dari pola tersebut, bagaimana pola tersebut memenuhi kebutuhan pengguna, alternatif serupa apa yang tersedia, dan bagaimana menerapkannya untuk brand Anda.

Design Patterns memungkinkan Anda untuk merujuk pada solusi yang telah terbukti yang berarti Anda memiliki dukungan dalam keputusan Anda. Desainer agensi bekerja dengan klien yang skeptis dan desainer internal yang menghadapi kebuntuan internal terkadang perlu untuk melindungi metode tertentu atau menyelesaikan hambatan. Dalam situasi seperti ini, mungkin akan berguna untuk melihat bagaimana Apple menggunakan standar ini untuk membantu pelanggan terhubung dengan dukungan pelanggan atau Amazon menggunakan standar pengembangan konversi. Hal ini menunjukkan contoh konkret dan nyata dari sebuah standar yang diimplementasikan di dunia nyata dan membantu para pemangku kepentingan dan mitra untuk melihat hasil yang diinginkan.

2.9 Design Tokens

Token desain mewakili keputusan desain kecil dan berulang yang membentuk gaya visual *Design System*. Token menggantikan nilai statis, seperti kode heksa untuk warna, dengan nama yang cukup jelas.[15] Token *Design System* adalah nilai gaya elemen *UI* seperti warna, tipografi, spasi, bayangan, dll., yang digunakan di seluruh produk dan dapat dikonversi ke format untuk platform apa pun (web, seluler, desktop). Token adalah blok bangunan dari *Design System* - anggap saja sebagai sub atom, bagian terkecil dari nilai gaya yang memungkinkan desainer membuat gaya untuk suatu produk. [16]

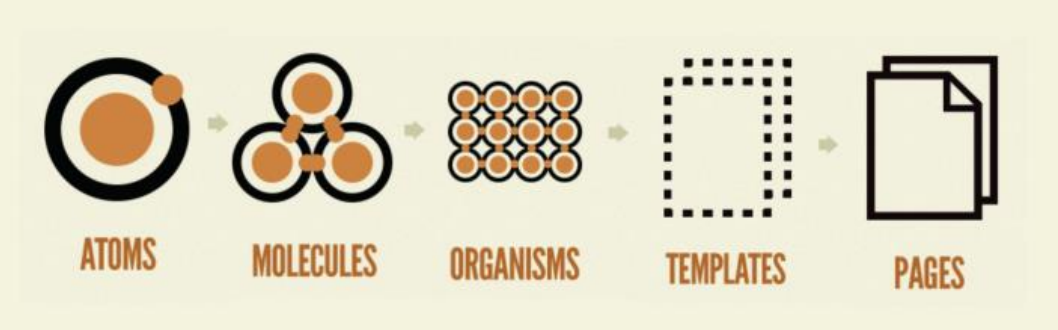


Gambar 2.14 Design Tokens

Memperbarui nilai gaya tunggal dalam desain tidak memakan waktu. Namun, untuk proyek nyata, atribut seperti warna merek atau tipografi digunakan di lusinan tempat yang berbeda, dan butuh banyak waktu untuk menyesuaikan nilai secara manual. Ditambah lagi, relatif mudah untuk lupa mengubah nilai tertentu untuk beberapa elemen (atau beberapa elemen) dan menimbulkan ketidakkonsistenan dalam desain. Berbeda dengan kelemahan nilai yang dikodekan dengan keras ini, token menawarkan berbagai manfaat untuk proses desain. token desain menyederhanakan proses pengembangan. Pengembang memiliki akses ke atribut desain terbaru melalui perangkat lunak *Design System*. Mereka dapat menggunakan token desain dalam kode seperti paket npm dan menerima pembaruan desain tanpa mengubah kode.

2.10 Atomic Design Methodology

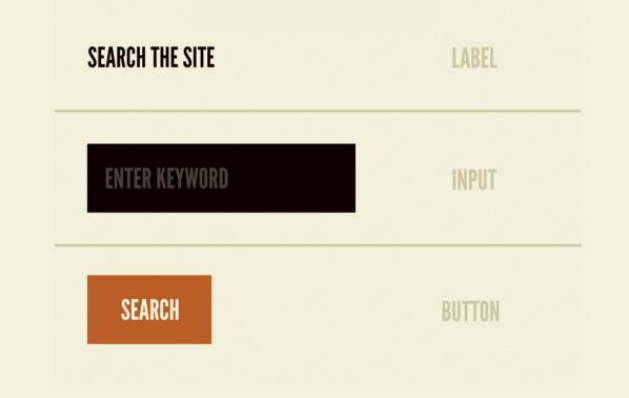
Metodologi Atomic Design adalah suatu model yang mempresentasikan user interface sebagai kumpulan komponen yang saling terhubung dan terorganisasi secara hirarkis baik dari segi informasi maupun fungsionalitas, sehingga dapat membangun produk digital secara terstruktur. Terdiri dari lima tahap yang bekerja bersama untuk membuat sistem design secara sistematis dan hirarkis [3].



Gambar 2.15 Atomic Design

1. Atoms

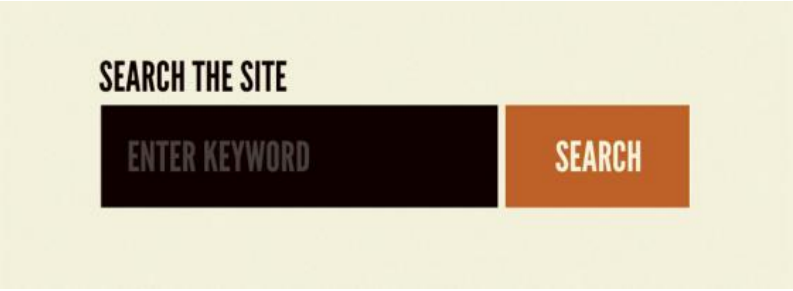
Seperti dalam dunia kimia, bagian terkecil dari rancangan atom adalah atom itu sendiri. Bentuknya berupa tombol, input, label, dll. Ciri khas dari komponen ini adalah tidak dapat dibongkar pasang. Saat diurai, fungsi dan maknanya pecah dan berbeda. Contohnya adalah tombol pencarian. Jika Anda memecahnya menjadi tombol-tombol dan kata-kata "search", arti keduanya tentu saja akan berbeda.



Gambar 2.16 Atoms

2. Molecules

Molekul adalah grup dari atom yang dikelompokkan sebagai elemen dari user interface dan berfungsi sebagai satu unit. Molekul mengambil beberapa atom dan menggabungkan nya bersama untuk memiliki fungsi tertentu. Dengan menggabungkan atom label, input, dan button, maka atom-atom tersebut memiliki fungsi tertentu. Seperti label yang menjelaskan input dan button yang berfungsi untuk mengirimkan isian form dalam input. Hasil dari gabungan ini adalah molekul form pencarian, yang merupakan komponen yang dapat digunakan kembali dan dapat digunakan kapan saja diperlukan.



Gambar 2.17 Moleculs

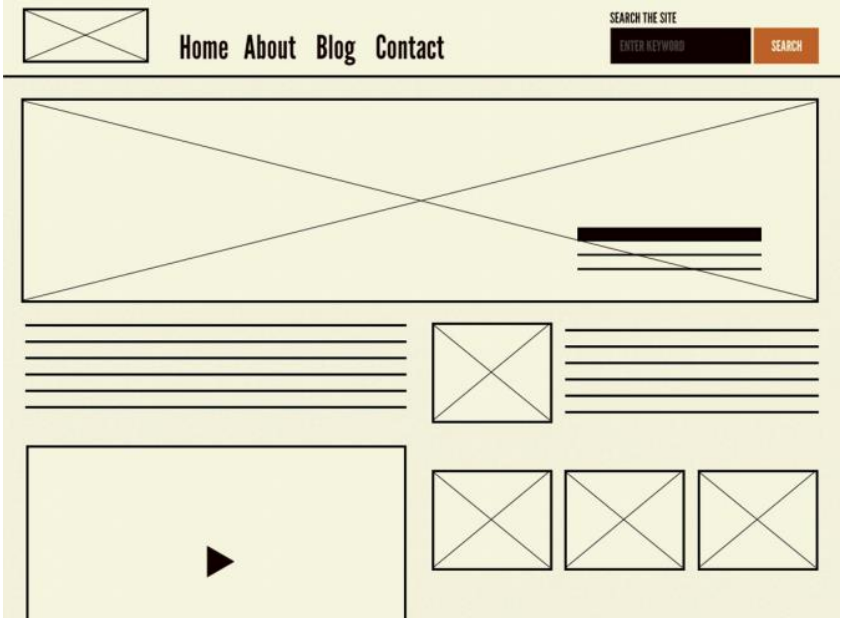
3. Organisms

Organism adalah komponen dari user interface yang lebih kompleks dan terdiri dari grup molekul, atom, atau organism lain. Organism membentuk bagian-bagian yang berbeda dari suatu tampilan pengguna.

Gambar 2.18 Organisms

4. Templates

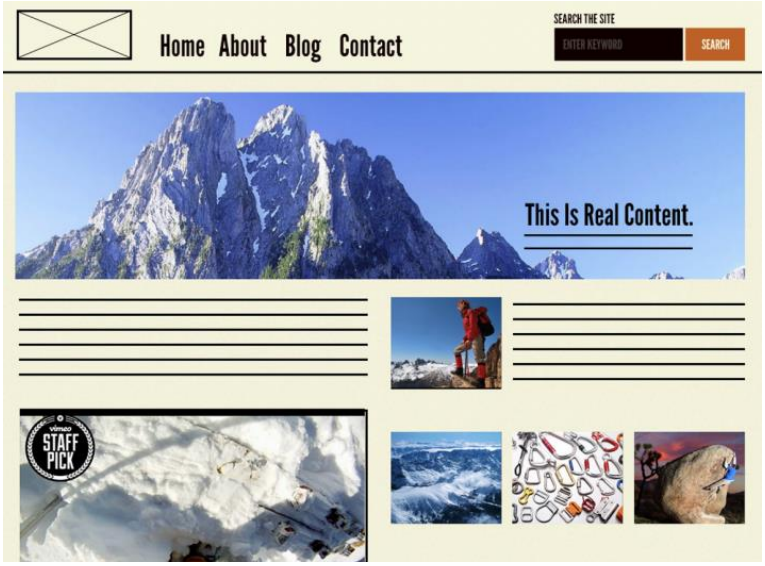
Templates adalah objek pada tingkat halaman yang menempatkan komponen dalam tata letak halaman dan menjelaskan struktur konten dasar dari desain tampilan. Dalam hal lain, templates adalah struktur dasar dari grup organism yang akan menjadi tampilan utama dari suatu tampilan.



Gambar 2.19 Templates

5. Pages

Pages adalah contoh spesifik dari suatu template yang menampilkan bagaimana tampilan pengguna akan terlihat dengan konten yang representatif. Pages sangat penting karena mereka yang akan dilihat oleh pengguna akhir dan bagaimana mereka akan berinteraksi dengan tampilan.



Gambar 2.20 Pages

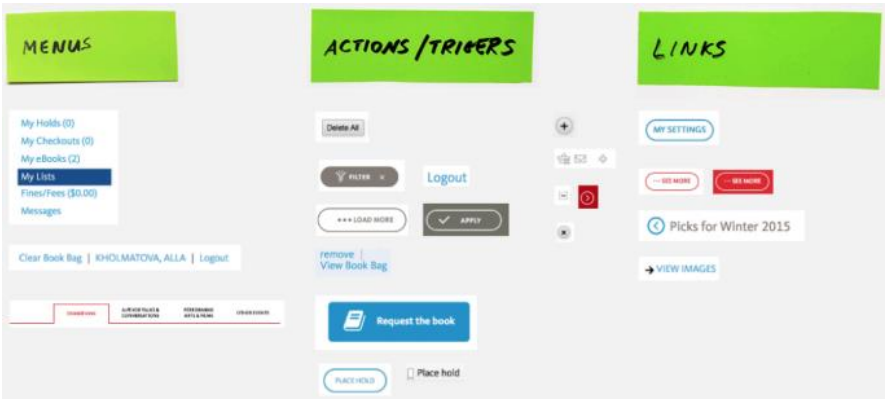
2.11 Functional Pattern

Design system terdiri dari elemen-elemen yang independen namun saling terkait dan mempengaruhi satu sama lain. *Design system* terbagi menjadi dua bagian, pertama adalah functional pattern yang mencakup komponen UI, panduan, model interaksi, dan elemen independen lainnya yang saling terikat. Kedua adalah perceptual pattern yang mencakup dasar aturan visual dari elemen dasar seperti palet warna, tipografi, jarak & layout, ikonografi dan ilustrasi.

Functional Pattern adalah kumpulan elemen komponen tampilan yang memiliki pola tampilan. Tujuannya adalah untuk mendorong perilaku dan kebiasaan pengguna dalam situasi tertentu [9]. Functional pattern harus didefinisikan untuk menemukan perilaku dan mental model pengguna terhadap tampilan. Dalam mendefinisikan functional pattern, beberapa tahap dilakukan seperti yang disebutkan selanjutnya.

1. *Conduct interface Inventory*

Koleksi komprehensif dari komponen interface disebut *interface inventory*. Ini adalah tahap untuk mengelompokkan konten, komponen, atau elemen visual lainnya untuk mengidentifikasi ketidakkonsistenan dalam desain sistem saat ini.



Gambar 2.21Interface Inventory

2. *Define Pattern*

Pada tahap ini, komponen akan dikelompokkan dan ditentukan apakah setiap elemen akan digabungkan menjadi satu pola yang sama atau dipisahkan. Ada dua langkah yang diambil untuk menentukan pola komponen, yaitu skala spesifitas dan memetakan struktur konten.

2.12 Perceptual Pattern

Perceptual pattern terdiri dari pola-pola yang membentuk dasar atribut style dari suatu produk atau sistem. Ini mencakup lapisan dasar dengan membuat aturan visual dari elemen seperti *color palette, tipografi, jarak & layout, ikon,* dan ilustrasi. *Perceptual pattern* juga meliputi *style guidelines*, yaitu panduan untuk desainer dalam merancang tampilan user interface, yang tersedia dalam bentuk dokumentasi seperti buku atau laporan. Tujuan utama dari *style guidelines* adalah membantu desainer mematuhi *"look and feel"* tertentu dan meningkatkan konsistensi di berbagai aplikasi, yang meliputi *tipografi, colors*, dan *ikonography*.

2.13 Usability Metrics

Usability metrics adalah aktivitas pengujian yang dirancang untuk mengukur kinerja pengguna dalam menyelesaikan serangkaian tugas tertentu. Metrik utama untuk metrik kegunaan adalah tingkat keberhasilan, waktu yang diperlukan untuk pengoperasian, dan tingkat kesalahan [2]. Kegiatan yang dilakukan adalah peneliti memberikan serangkaian tugas kepada partisipan untuk dilakukan dengan menggunakan desain yang diujikan. Selama pengujian, peneliti mengamati perilaku setiap peserta selama perekaman video. Setelah menganalisis hasil tes, para peneliti mencatat beberapa poin

ektivitas desain, efisiensi desain, kepuasan pengguna, serta isu-isu yang muncul seperti aspek desain yang menyebabkan masalah. Berikut *jenis-jenis usability* testing berdasarkan tujuannya:

1. *Quantitative Usability Metric*

   Pengukuran kegunaan kuantitatif telah menunjukkan bahwa desain dapat digunakan secara efektif. Evaluasi ini dilakukan dengan mengukur dan menggunakan hasil numerik. Tugas Selesai menghitung jumlah orang yang telah menyelesaikan tugas dan waktu yang dihabiskan untuk menyelesaikan tugas tersebut. Beberapa hal yang dapat Anda ukur dan hitung adalah:

a. Efektivitas

   Efektivitas yang dimaksud merupakan nilai yang didapat berdasarkan tingkat keberhasilan desainer dalam menyelesaikan tugas tertentu. Efektivitas diukur dengan cara membandingkan jumlah partisipan yang berhasil menyelesaikan tugasdengan jumlah keseluruhan partisipan.

$$Effectiveness = \frac{\text{Number of tasks completed successfully}}{} \times 100\%$$

Gambar 2.22 Efektivitas

b. Efisiensi Relatif Secara Keseluruhan

Setiap tugas dapat dinilai dalam nilai relatifnya secara keseluruhan. Nilai ini menunjukkan seberapa cepat perancang dapat menyelesaikan tugasnya. Ukuran kinerja relatif keseluruhan dapat dibuat dengan membandingkan waktu peserta menyelesaikan tugas dengan total waktu yang dihabiskan oleh masing-masing peserta

$$Overall\ Relative\ Efficiency = \frac{\sum_{j=1}^{R}\sum_{i=1}^{N} n_{ij}t_{ij}}{\sum_{j=1}^{R}\sum_{i=1}^{N} t_{ij}} \times 100\%$$

Gambar 2.23 Efisiensi Relatif

c. Kepuasan Pengguna SUS

System Usability Scale (SUS) adalah pendekatan yang digunakan untuk mengevaluasi kemanfaatan suatu situs web. Dalam metode ini, pengguna mengisi kuesioner dengan 10 pertanyaan yang dirancang untuk mengukur sejauh mana pengguna setuju atau tidak setuju terhadap pernyataan mengenai kemudahan penggunaan situs [17]. Skor akhir SUS dihitung dengan mengurangkan atau mengubah beberapa skor pertanyaan tertentu, dan hasilnya dijumlahkan serta dikalikan untuk menghasilkan skor dalam rentang 0 hingga 100. Skor di atas 68 mengindikasikan kemanfaatan yang layak, sementara skor di bawah 68 menunjukkan adanya potensi masalah yang perlu diperhatikan. Metode ini telah terbukti efektif dalam mengevaluasi situs web selama bertahun-tahun, dengan hubungan antara skor dan nilai yang membantu interpretasi hasil evaluasi.

Proses penggunaan SUS dalam menilai kemanfaatan situs web melibatkan beberapa langkah yang terdiri dari pengenalan terhadap metode, pemberian kuesioner SUS, perhitungan skor, interpretasi skor, dan penggunaan SUS untuk evaluasi. Pengenalan terhadap SUS memberikan gambaran tentang alat evaluasi ini dan mengapa ia penting dalam

mengevaluasi kemanfaatan suatu situs web. Selanjutnya, pengisian kuesioner oleh pengguna mengumpulkan data penting untuk mengukur persepsi pengguna terhadap kemudahan penggunaan situs. Dari hasil tanggapan pengguna, skor akhir SUS dihitung dan diinterpretasikan untuk memberikan wawasan tentang sejauh mana situs web tersebut bermanfaat. Rentang skor ini memiliki hubungan dengan tingkat nilai yang dapat digunakan untuk mengklasifikasikan hasil penilaian, seperti A (80,3 atau lebih tinggi), C (sekitar 68), dan F (51 atau kurang).

2. *Qualitative Usability Metric*

Qualitative usability metric Berguna bagi para desainer untuk mendapatkan informasi yang dapat membantu mereka meningkatkan desain mereka. Tidak seperti banyak jenis, dalam jenis interaksi peneliti-peserta inilah yang perlu dilakukan untuk mendapatkan wawasan. Berikut adalah langkah-langkah yang harus diambil selama tes validasi Anda:

a. Menentukan Tujuan Pengujian

Tujuan pengujian ini adalah untuk mengetahui keefektifan program yang diuji. Sasaran ini mencakup hasil kinerja seperti efisiensi, efektivitas, kepuasan pengguna, dll. Suatu penelitian dikatakan berhasil jika tujuannya tercapai. Artinya hasil yang diperoleh dalam tes kompetitif yang telah diputuskan.

b. Membuat Daftar Tugas Partisipan dan Skenario

daftar tugas untuk menentukan apa yang perlu dilakukan peserta untuk menyelesaikan tugas. Setiap pekerjaan yang disimpan memiliki kondisi. Situasi kreatif mencakup banyak elemen, seperti peran peserta dan kegiatan yang harus dilakukan tanpa bimbingan.

c. Membuat Naskah Pengujian

Banyak poin yang harus dievaluasi selama tes. Detail ini mungkin diingat dengan baik oleh penguji, tetapi mungkin dilupakan. Inilah

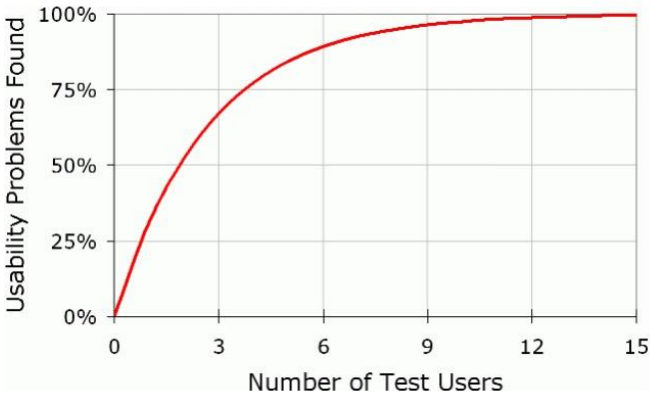
mengapa skrip memainkan peran penting dalam implementasi tes validasi. Esai merekam semua yang perlu dikatakan dan semua yang perlu dilakukan peneliti. Skrip juga berisi daftar tugas dan kasus uji.

d. Melakukan Pengujian dan Mencatat Hasil Pengujian

Pada tahap ini, dilakukan pengujian dengan mengikuti alur yang telah dituliskan dalam naskah. Peneliti juga harus mencatat setiap hal yang terjadi ketikapengujian berlangsung, seperti bagaimana cara pengguna berinteraksi dengan produk dan apa yang mereka katakan.

e. Melakukan evaluasi hasil pengujian

Penilaian dilakukan untuk memperoleh pengetahuan dari tes yang diperoleh pada fase ini. Menurut penelitian Jakob Nielsen dan Tom Landauer, percobaan dengan lima peserta menunjukkan hasil terbaik. Setelah berdiskusi dengan peserta kelima, mereka tidak menemukan perbedaan hasil yang terlihat pada sesi sebelumnya. Di bawah ini adalah ikhtisar dari hasil

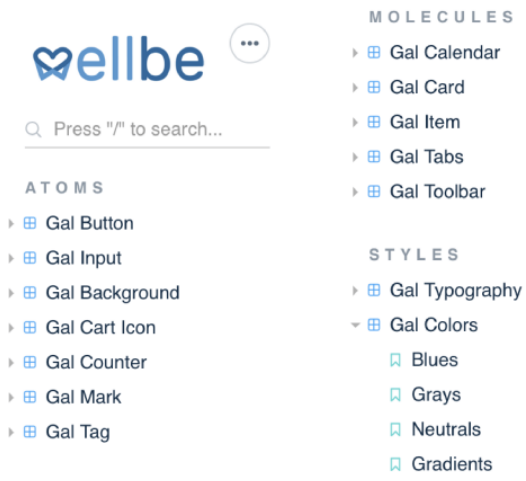


Gambar 2.24Usability Metrics

2.14 Design Documentation

Untuk mendokumentasikan *Design System* dan juga memastikan agar tetap sinkron, saya membuat sebuah Storybook sebagai sumber tunggal kebenarannya. Saya menggunakan versi Storybook untuk html karena pada saat penulisan, versi untuk web components belum memiliki dukungan resmi atau dokumentasi dan

hanya sedang dites.[10] Hal ini membatasi implementasinya, tetapi tidak ada masalah besar. Untuk struktur komponen dalam *Storybook*, saya menggunakan struktur Atoms dan Molecules yang sama seperti dalam bahasa desain visual dan perpustakaan komponen, dan menambahkan bagian gaya untuk menunjukkan tipografi dan warna. Struktur pohon *Storybook* ditunjukkan pada Gambar.



Gambar 2.25 Design Documentation

2.14.1 Code Style Guides

Panduan gaya kode atau standar kode sering fokus pada format dan konvensi penamaan kode dari tim rekayasa perangkat lunak, seperti apakah mereka menggunakan tab atau spasi untuk mengidentifikasi kode dan bagaimana mereka memberi nama metode. Contoh formal dari standar ini adalah "PSR-2" untuk PHP. Panduan gaya kode sering terpisah dari masalah desain. Oleh karena itu, mereka sering disimpan terpisah dari panduan gaya dan sumber daya *Design System* yang berorientasi desain, ditempatkan pada halaman README basis kode atau wiki repository kode. Secara umum, hanya pengembang yang memiliki akses ke ini.

Mungkin ada beberapa kemiripan. Misalnya, bayangkan jika panduan merek mengusulkan warna #fe6481 yang disebut sebagai "warna utama merek", dan panduan gaya kode menentukan variabel Sass bernama $brink-pink: #fe6481;

sesuai dengan Name that Color. Ini mungkin menyebabkan kesalahpahaman saat desainer dan pengembang berbicara satu sama lain tentang warna.

2.14.2 Component library

Component libraries, *UI libraries*, atau code libraries memberikan kode depan untuk komponen UI (widget, modul, potongan, atau blok). Secara internal, Anda bisa menggunakan pustaka komponen sebagai koleksi bersama dari potongan *UI* yang menerapkan pola yang dapat dikontribusikan oleh siapa saja dalam organisasi. Lihat pustaka komponen UI open-source U.S. Web *Design System* (https://designsystem.digital.gov/components/). Berbeda dengan *framework UI* seperti Bootstrap, pustaka komponen dibuat sesuai dengan tujuan tertentu, seperti merek internal.

Perpustakaan komponen internal menghadapi banyak tantangan yang sama seperti proyek sumber terbuka, termasuk masalah pemberian dan peluncuran. Mereka juga menghadapi tantangan pemasaran dan manajemen produk, seperti perencanaan jalan cerita, perencanaan rilis, tantangan adopsi, dan tantangan komunikasi pengumuman produk.

2.15 Unit Testing

Unit testing adalah proses memecah aplikasi menjadi fungsi-fungsi terkecil, dan membuat tes yang dapat diulang dan otomatis yang harus terus-menerus menghasilkan hasil yang sama. Tes ini adalah jantung dan jiwa dari aplikasi Anda. Mereka memberikan fondasi bagi semua kode masa depan dibangun. Tanpa unit testing, mungkin fungsi jarang digunakan bisa tetap rusak selama beberapa bulan tanpa ada yang melaporkan. Dengan unit testing, setiap fungsi sistem dapat diverifikasi sebelum satu baris kode bahkan digabungkan ke cabang master, apalagi diteruskan ke produksi.[8]

Unit testing pada design system adalah proses memecah aplikasi menjadi fungsi-fungsi terkecil dan membuat tes ulang-alik, otomatis yang harus terus menghasilkan hasil yang sama. Tes ini merupakan tulang punggung dari aplikasi

Anda. Mereka menyediakan fondasi tempat semua kode masa depan dibangun. Tanpa tes unit, mungkin fungsi jarang digunakan bisa tetap rusak selama beberapa bulan tanpa ada yang menyadari. Dengan tes unit, setiap fungsi sistem dapat diverifikasi sebelum satu baris kode bahkan digabungkan ke cabang utama, apalagi dikirimkan ke produksi.