BAB II
LANDASAN TEORI

A. Website

Website adalah sebuah kumpulan halaman yang berisi informasi tertentu dan dapat diakses oleh banyak orang melalui internet. *Website* dapat dibuka dengan menuliskan URL atau alamat *website* di *browser*.

Website pertama kali dibuat oleh Tim Berners-Lee pada akhir 1980-an dan baru resmi *online* pada tahun 1991. Tujuan awal Tim Berners-Lee membuat sebuah *website* adalah supaya lebih memudahkan para peneliti di tempatnya bekerja ketika akan bertukar atau melakukan perubahan informasi. Pada saat itu, *website* mulai dapat digunakan secara gratis oleh publik baru diumumkan oleh CERN tepatnya tanggal 30 April 1993.[13]

B. Frontend

Frontend adalah bagian yang langsung dilihat oleh user. User dapat langsung berinteraksi pada bagian yang dibangun menggunakan *HTML*, *CSS*, dan *JavaScript*. Meskipun ada banyak jenis teknologi dan tumpukan yang berbeda, sebagian besar *developer web front-end* menggunakan *HTML*, *CSS* dan *JavaScript*, *the de facto building blocks of the web*, dan *frameworks* dari sisi klien seperti *Angular*, *React*, *Stencil* dan *Vue*[4].

C. Prototype

Prototipe adalah perwujudan fisik atau digital dari elemen penting dari desain yang dimaksudkan, dan *tool iterative* untuk meningkatkan komunikasi, memungkinkan pembelajaran, dan menginformasikan pengambilan keputusan pada setiap titik dalam proses desain. Prototyping adalah proses menciptakan perwujudan fisik atau digital dari elemen-elemen penting dari desain yang dimaksud[6].

D. PWA (*Progressive Web App*)

PWA adalah situs web yang terlihat dan terasa seperti *mobile app* asli. PWA berjalan di *browser*, jadi tidak perlu mengunduh aplikasi dari *Google Play Store* atau *iOS App Store*. PWA dimaksudkan untuk menghilangkan berbagai masalah mulai dari jaringan yang lambat hingga keterbatasan data

atau kurangnya konektivitas. Situs web yang memenuhi semua *requirement* PWA memanfaatkan teknologi web terbaru untuk memberikan pengalaman pengguna yang andal, cepat, dan menarik[7].

E. Sheet.js

Sheet.js adalah parser dan penulis untuk berbagai format spreadsheet. Sheet.js juga merupakan implementasi Cleanroom Pure-JS dari spesifikasi resmi, dokumen terkait, dan file pengujian. Sheet.js menekankan pada ketahanan parsing dan penulisan, kompatibilitas fitur lintas format dengan representasi JS terpadu, dan kompatibilitas browser ES3/ES5 kembali ke IE6.

F. *Polyfill*

Polyfill adalah potongan kode (biasanya *JavaScript* di *Web*) yang digunakan untuk menyediakan fungsionalitas modern pada *browser* lama yang tidak mendukungnya secara *native*. *Polyfill* digunakan untuk mengatasi masalah di mana *browser* mengimplementasikan fitur yang sama dengan cara yang berbeda. *Polyfill* menggunakan fitur *non-standar* di *browser* tertentu untuk memberikan *JavaScript* cara yang sesuai standar untuk mengakses fitur tersebut[8].

G. SASS

Situs web pada awalnya didasarkan pada bahasa markup *HTML*. Jika ingin mendesain dalam *HTML*, maka harus menautkan konten langsung ke desain – di setiap halaman *HTML*. Sederhananya, *HTML* tidak dimaksudkan untuk desain, dan sangat terbatas di area ini.

Cascading Style Sheets (*CSS*) mengatur presentasi halaman *HTML*. *CSS* berada di atas kode *HTML*, seperti template, dan mendefinisikan desain untuk setiap elemen pada halaman: *Font*, warna *font*, latar belakang – desainer web dapat mengatur semua elemen desain ini dengan *CSS*. Tetapi *CSS* memiliki batasannya, yang sangat jelas jika telah bekerja dengan bahasa tersebut selama bertahun-tahun.

SASS membuat semuanya sedikit lebih canggih, dan sangat menyederhanakan pekerjaan membuat desain. Sehingga *SASS* dianggap sebagai praprosesor. Kode dalam dokumen *SASS* terlebih dahulu dikonversi ke *CSS* sebelum kode sumber, yaitu situs web, dikirimkan ke sistem[9].

H. *CSS Normalization*

CSS Normalization mencoba meratakan perbedaan antara *browser* saat merender elemen *HTML*. Banyak browser memiliki "*pre-settings*":

1. elemen `p` dan `h` memiliki *margin vertikal*
2. `*lists*` memiliki beberapa *margin* dan *padding* juga
3. *tag* `em` dan `*strong*` memiliki `*font-weight*` yang dicetak tebal

Semua pra-pengaturan ini diatur ulang, sehingga pengembang memiliki basis kerja yang konsisten di semua *browser*.

I. *Cross Browser Testing*

Cross Browser Testing adalah jenis pengujian non-fungsional yang memungkinkan Anda memeriksa apakah situs web Anda berfungsi sebagaimana dimaksud saat diakses melalui:

1. Kombinasi *Browser-OS* yang berbeda yaitu, pada browser populer seperti *Firefox, Chrome, Edge, Safari* pada salah satu sistem operasi populer seperti *Windows, macOS, iOS*, dan *Android*.
2. Perangkat yang berbeda yaitu, pengguna dapat melihat dan berinteraksi dengan situs web Anda di perangkat populer—*smartphone*, *tablet*, *desktop*, dan laptop, dll.
3. *Assistive Tools* yaitu, situs web kompatibel dengan teknologi bantu seperti pembaca layar untuk individu yang memiliki kemampuan berbeda[10].

J. *Responsive Design*

Responsive Design adalah pendekatan desain *Graphic User Interface* (GUI) yang digunakan untuk membuat konten yang menyesuaikan dengan lancar ke berbagai ukuran layar. Desainer mengukur elemen dalam unit relatif (%) dan menerapkan kueri media, sehingga desain desainer dapat secara otomatis beradaptasi dengan ruang browser untuk memastikan konsistensi konten di seluruh perangkat[11].