PERANCANGAN APLIKASI DIGITAL LIBRARY BERBASIS PROGRESSIVE WEB APPS

Disusun dan diajukan oleh

ROSIHAN ARDIANSYAH

D421 15 305

DEPARTEMEN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

MAKASSAR

2022

LEMBAR PENGESAHAN SKRIPSI

PERANCANGAN APLIKASI DIGITAL LIBRARY BERBASIS PROGRESSIVE WEB APPS

Disusun dan diajukan oleh

ROSIHAN ARDIANSYAH

D421 15 305

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Informatika

Fakultas Teknik Universitas Hasanuddin

Pada tanggal 30 Juni 2022

dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui

Pembimbing Utama,	Pembimbing Pendamping,
Dr. Ir. Ingrid Nurtanio, M.T	Anugrayani Bustamin,S.T.,M.T.
Nip. 19610813 198811 2 001	Nip. 19901201 201807 4 001

Plt. Ketua Departemen Teknik Informatika

Dr. Amil Ahmad Ilham, S.T., M.IT
Nip. 19731010 199802 1 001

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini :

Nama : ROSIHAN ARDIANSYAH

Nim : D421 15 305

Program Studi : S1 Teknik Informatika

Menyatakan dengan sebenar-benarnya bahwa skripsi yang berjudul :

PERANCANGAN APLIKASI DIGITAL LIBRARY

BERBASIS PROGGRESSIVE WEB APPS

Adalah karya ilmiah saya sendiri dan sepanjang pengetahuan saya di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan/ditulis/diterbitkan sebelumnya, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila dikemudian hari ternyata didalam naskah skripsi ini terdapat unsur-unsur djiplakan, saya bersedia menerima sanksi atas perbuatan tersebut dan diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2000, pasal 25 ayat 2 dan pasal 70).

Makassar, 30 Juni 2022

Yang membuat Pernyataan

ROSIHAN ARDIANSYAH

ABSTRAK

Perkembangan teknologi saat ini telah berkembang dengan pesat dan telah menyebar hampir di seluruh bidang dalam kehidupan, termasuk perpustakaan sebagai institusi pengelola informasi. Perkembangan dari penerapan teknologi informasi yang berkembang dengan sangat pesat dapat terlihat dari jenis perpustakaan yang berawal dari perpustakaan manual hingga *digital library*. Perkembangan *digital library* tidak lepas dari perkembangan teknologi informasi. *Digital library* dibuat dengan basis *web,* yang memungkinkan pengaksesan koleksi oleh anggota perpustakaan kapanpun dan di manapun. Saat ini, kebanyakan *digital library* hanya digunakan oleh pihak-pihak tertentu seperti anggota perpustakaan dari perpustakaan tertentu untuk menyimpan informasi mengenai *digital resource* seperti *softcopy* dari sebuah buku dan hanya dapat diakses oleh anggota perpustakaan tersebut, namun tidak menyimpan data-data mengenai koleksi-koleksi dalam bentuk *hardcopy* yang berada dalam gedung-gedung perpustakaan. Sistem informasi yang dimiliki oleh perpustakaan FTUH saat ini masih bersifat manual, sehingga diperlukan sistem informasi yang bersifat *digital* untuk mempermudah pelayanan perpustakaan FTUH. Penelitian ini menerapkan *Progressive Web App* yang memungkinkan sistem bekerja secara cepat dan handal dengan kemampuan mengelola *file-file* dan *data* secara *offline* dengan teknologi *service worker*. Teknologi ini diharapkan dapat menunjang proses sistem informasi pada perpustakaan FTUH baik untuk pengurus maupun anggota perpustakaan. Untuk menguji sistem informasi yang dibuat, digunakan metode *Black Box* untuk pengujian fungsional sistem, pengujian fitur *Progressive Web App* menggunakan aplikasi *Lighthouse* pada *Google Chrome,* dan pengujian kelayakan terhadap pengguna dengan menggunakan metode *User Acceptance Test*. Hasil penelitian menunjukkan bahwa bahwa peranan *service worker* dapat meningkatkan kinerja aplikasi dalam kategori *performance* melalui pengujian yang telah dilakukan. Pada pengujian *Black Box Testing*, tidak ditemukan adanya kesalahan fungsi pada aplikasi *Digital Library* dan berjalan sesuai dengan yang diharapkan berdasarkan masukan dan keluaran yang sesuai. Pada pengujian performa dengan menggunakan aplikasi *Google Lighthouse*, didapatkan nilai 100 pada seluruh kategori pengujian (*performance,accessibility, best practices, SEO)*. Pada pengujian kelayakan untuk pengguna, sebanyak 20 responden menjawab 7 buah pertanyaan dengan memberikan penilaian berskala 1 sampai 5, maka didapatkan nilai rata-rata 4,19 yang masuk ke dalam kategori sangat layak berdasarkan rumus pengkategorian dari *User Acceptance Test*. Dari hasil pengujian yang dilakukan, dapat disimpulkan bahwa aplikasi ini layak digunakan

Kata Kunci : Digital Library, Progressive Web App, service worker

ABSTRACT

Today's technological developments have grown rapidly and have spread in almost all areas of life, including libraries as information management institutions. The development of the application of information technology that is growing very rapidly can be seen from the types of libraries that start from manual libraries to digital libraries. The development of digital libraries cannot be separated from the development of information technology. The digital library is made on a web basis, which allows library members to access collections anytime and anywhere. Currently, most digital libraries are only used by certain parties such as library members from certain libraries to store information about digital resources such as softcopy of a book and can only be accessed by members of the library, but do not store data about the collections in the library. hardcopy form in library buildings. The information system owned by the FTUH library is currently still manual, so a digital information system is needed to facilitate FTUH library services. This research applies a Progressive Web App that allows the system to work quickly and reliably with the ability to manage files and data offline with service worker technology. This technology is expected to support the process of information systems in the FTUH library for both administrators and library members. To test the information system created, the Black Box method is used for system functional testing, testing the Progressive Web App features using the Lighthouse application on Google Chrome, and testing the feasibility of users using the User Acceptance Test method. The results of the study indicate that the role of service workers can improve application performance in the performance category through the tests that have been carried out. In the Black Box Testing test, no malfunction was found in the Digital Library application and it ran as expected based on the appropriate input and output. In performance testing using the Google Lighthouse application, we got a score of 100 in all test categories (performance, accessibility, best practices, SEO). In the feasibility test for users, as many as 20 respondents answered 7 questions by giving an assessment on a scale of 1 to 5, then an average value of 4.19 was obtained which was included in the very feasible category based on the categorization formula of the User Acceptance Test. From the results of the tests carried out, it can be concluded that this application is feasible to use

Keywords : Digital Library, Progressive Web App, service worker

KATA PENGANTAR

Assalamu Alaikum Wr. Wb.

Puji dan syukur penulis panjatkan atas kehadirat Allah SWT karena berkat Rahmat dan Karunia-Nya sehingga Tugas Akhir yang berjudul "**PERANCANGAN APLIKASI DIGITAL LIBRARY BERBASIS PROGRESSIVE WEB APPS**" ini dapat diselesaikan sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Dalam penyusunan penelitian ini disajikan hasil penelitian terkait judul yang telah diangkat dan telah melalui proses pencarian dari berbagai sumber baik jurnal penelitian, prosiding pada seminar-seminar nasional maupun internasional, buku, dan dari berbagai situs-situs terpercaya yang ada di internet.

Penulis menyadari bahwa dalam penyusunan dan penulisan skripsi ini tidak lepas dari bantuan, bimbingan serta dukungan dari berbagai pihak, dari masa perkuliahan sampai dengan masa penyusunan tugas akhir, sangatlah sulit untuk menyelesaikan tugas akhir ini. Oleh karena itu, penulis dengan senang hati menyampaikan terima kasih kepada:

1. Tuhan Yang Maha Esa atas semua berkat, karunia serta pertolongan-Nya yang telah diberikan kepada kami disetiap langkah dalam pembuatan program hingga penulisan laporan skripsi ini.
2. Orang tua penulis, Alm Bapak Ir. Rachmat Hidayat dan Ibu Harlina,S.E., yang selalu memberikan dukungan, doa, dan semangat serta selalu sabar dalam mendidik penulis sejak kecil.
3. Ibu Dr. Ir. Ingrid Nurtanio, M.T. selaku pembimbing 1 dan Ibu Anugrayani Bustamin, S.T.,M.T selaku pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan perhatian yang luar biasa untuk mengarahkan penulis dalam penyusunan tugas akhir;

4. Bapak Amil Ahmad Ilham, ST., M.IT., Ph.D selaku Ketua Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas bimbingannya selama masa perkuliahan penulis;
5. Para teman-teman dan kakak-kakak Laboratorium AIMP Unhas yang telah memberikan begitu banyak bantuan selama penelitian dan diskusi progress penyusunan Tugas Akhir;
6. Teman-teman Hypervisorr FT UH atas dukungan dan semangat yang diberikan selama ini;
7. Segenap Staf Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah membantu penulis;
8. Chae, Yuda, Ogi, Maul, Mino, Ami, yang telah memberikan dukungan dan bantuan kepada penulis untuk menyelesaikan penulisan tugas akhir;
9. Orang-orang berpengaruh lainnya yang tanpa sadar telah menjadi inspirasi penulis

Wassalam

Makassar, 9
September 2021

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi saat ini telah berkembang dengan pesat dan telah menyebar hampir di seluruh bidang dalam kehidupan, termasuk perpustakaan sebagai institusi pengelola informasi. Perkembangan dari penerapan teknologi informasi yang berkembang dengan sangat pesat dapat terlihat dari jenis perpustakaan yang berawal dari perpustakaan manual hingga *digital library* seperti Ipusnas, BSE Kemdikbud, ITB Library (Sismanto, 2008)

Perkembangan *digital library* tidak lepas dari perkembangan teknologi informasi. *Digital library* dibuat dengan basis *web,* yang memungkinkan pengaksesan koleksi oleh anggota perpustakaan kapanpun dan di manapun. Saat ini, kebanyakan *digital library* hanya digunakan oleh pihak-pihak tertentu seperti anggota perpustakaan dari perpustakaan tertentu untuk menyimpan informasi mengenai *digital resource* seperti *softcopy* dari sebuah buku dan hanya dapat diakses oleh anggota perpustakaan tersebut, namun tidak menyimpan data-data mengenai koleksi-koleksi dalam bentuk *hardcopy* yang berada dalam gedung-gedung perpustakaan (Ruddamayanti, 2019).

Progressive Web Apps (PWA) mulai diperkenalkan oleh Google pada tahun 2015. PWA merupakan sebuah *web* yang dilengkapi dengan teknologi yang membuatnya dapat menjadi seperti aplikasi pada perangkat *smartphone* dan fungsinya berjalan melalui *browser* yang terpasang pada perangkat. PWA dapat ditambahkan dan dibuka langsung melalui *home screen*. Pada saat dibuka PWA akan langsung memuat konten-konten yang ada di dalamnya tanpa memperhatikan koneksi jaringan yang dimiliki pada saat itu. Hal ini disebabkan karena adanya bantuan dari *Service Workers*, sebuah *Java Script Workers*, yang dapat melakukan *caching* pada konten untuk memberikan akses secara *offline.* Teknologi di balik PWA adalah *service worker*. Menurut Google Developer salah satu karakteristik dari PWA adalah *connectivity independent* di mana *service*

workers membantu web untuk dapat diakses secara *offline* ataupun pada kualitas koneksi jaringan yang rendah.

Pada tahun 2018, Ridho dkk. melakukan penelitian mengenai perbandingan performa *Progressive Web Apps* (PWA) dan *mobile web* terkait waktu respon, penggunaan memori dan penggunaan media penyimpanan pada suatu halaman *web*. Pada penelitian ini didapat beberapa kesimpulan yaitu pada ukuran berkas dan *cache* yang kecil, *mobile web* masih lebih unggul dibandingkan dengan PWA, sedangkan pada ukuran berkas dan *cache* yang cukup besar PWA mampu mengungguli *mobile web*. Untuk perfoma terkait penggunaan memori, *mobile web* menggunakan memori lebih sedikit dibandingkan dengan PWA dikarenakan adanya proses tambahan pada PWA yaitu *service worker*. Sedangkan untuk performa terkait media penyimpanan, pada *mobile web* tidak menggunakan media penyimpanan sama sekali, sedangkan PWA menggunakan media penyimpanan menyesuaikan dengan *cache* yang disimpan pada peramban.

Sistem informasi yang dimiliki oleh perpustakaan FTUH saat ini masih bersifat manual, sehingga diperlukan sistem informasi yang bersifat *digital* untuk mempermudah pelayanan perpustakaan FTUH. Penelitian ini menerapkan PWA yang memungkinkan sistem bekerja secara cepat dan handal dengan kemampuan mengelola *file-file* dan *data* secara *offline* dengan teknologi *service worker*. Teknologi ini diharapkan dapat menunjang proses sistem informasi pada perpustakaan FTUH baik untuk pengurus maupun anggota perpustakaan.

Berdasarkan latar belakang di atas, penulis kemudian mengangkat sebuah penelitian dengan judul "**PERANCANGAN APLIKASI DIGITAL LIBRARY BERBASIS PROGRESSIVE WEB APPS**".

1.2 Rumusan Masalah

Berdasarkan latar belakang, maka rumusan masalah pada tugas akhir ini adalah :

1. Bagaimana merancang aplikasi *Digital Library* yang dapat diakses secara *online* maupun *offline*?
2. Bagaimana cara mengukur kelayakan aplikasi *Digital Library* yang dikembangkan melalui pengujian dengan *Lighthouse*, *Black Box Testing*, dan *User Acceptance Test* ?

1.3 Tujuan Penelitian

1. Untuk merancang aplikasi *Digital Library* yang dapat diakses secara *online* maupun *offline*.
2. Untuk mengetahui kelayakan aplikasi *Digital Library* yang dikembangkan dengan pengukuran melalui pengujian dengan *Lighthouse*, *Black Box Testing*, dan *User Acceptance Test*

1.4 Manfaat Penelitian

Manfaat yang diharapkan dalam penelitian ini bagi peneliti adalah dapat menambah wawasan mengenai aplikasi berbasis Progressive Web Apps dan *Digital Library*. Bagi Pendidikan adalah dapat dijadikan sebagai bahan referensi mengenai aplikasi berbasis Progressive Web Apps yang dapat menunjang perkuliahan. Dan bagi mahasiswa adalah mampu meningkatkan minat baca terhadap literasi kampus

1.5 Batasan Masalah

1. *Digital library* berbasis web untuk lingkungan FTUH.

2. Penggunaan fitur hanya untuk mahasiswa, dosen aktif, dan pengurus perpustakaan.
3. *Website* yang menerapkan teknologi Progressive Web Apps dibuat menggunakan *framework* yaitu *Vue.js* dan Laravel.
4. Perangkat lunak yang digunakan untuk penyimpanan data adalah MySQL *database management system.*

1.6 Sistematika Penulisan

Untuk memberikan gambaran singkat mengenai isi tulisan secara keseluruhan, maka akan diuraikan beberapa tahapan dari penulisan secara sistematis, yaitu:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang diangkatnya judul penelitian Perancangan *Digital Library* Berbasis Progressive Web Apps disertai dengan rumusan masalah, tujuan dan manfaat penelitian, batasan masalah, metode penulisan, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini berisi teori-teori tentang hal-hal yang berhubungan dengan proses perancangan sistem informasi berbasis Progressive Web Apps.

BAB III METODOLOGI PENELITIAN

Bab ini berisi tentang gambaran umum sistem, dan skenario pengujian pada aplikasi Progressive Web Apps untuk membangun *Digital Library*.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil pengolahan data serta pembahasan yang disertai tabel hasil penelitian.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan yang didapatkan berdasarkan hasil penelitian yang telah dilakukan serta saran-saran untuk pengembangan lebih lanjut.

BAB II
TINJAUAN PUSTAKA

2.1 Perpustakaan Digital

Menurut Sismanto(2008), Perpustakaan *digital* adalah sebuah sistem yang memiliki berbagai layanan dan obyek informasi yang mendukung akses obyek informasi tersebut melalui perangkat *digital*. Layanan ini diharapkan dapat mempermudah pencarian informasi di dalam koleksi obyek informasi seperti dokumen, gambar dan database dalam format digital dengan cepat, tepat, dan akurat. Perpustakaan *digital* terkait dengan sumber-sumber lain dan pelayanan informasinya terbuka bagi pengguna di seluruh dunia. Koleksi perpustakaan *digital* tidaklah terbatas pada dokumen elektronik pengganti bentuk cetak saja, namun ruang lingkup koleksinya sampai pada tingkat di mana perangkat *digital* tidak bisa digantikan dalam bentuk tercetak. Koleksi menekankan pada isi informasi, jenisnya dari dokumen tradisional sampai hasil penelusuran. Perpustakaan ini melayani mesin, manajer informasi, dan pemakai informasi. Semuanya ini demi mendukung manajemen koleksi, menyimpan, pelayanan bantuan penelusuran informasi.

Perbedaan "perpustakaan biasa" dengan "perpustakaan *digital*" terlihat pada keberadaan koleksi. Koleksi *digital* tidak harus berada di sebuah tempat fisik, sedangkan koleksi biasa terletak pada sebuah tempat yang menetap, yaitu perpustakaan. Perbedaan kedua terlihat dari konsepnya. Konsep perpustakaan *digital* identik dengan internet atau komputer, sedangkan konsep perpustakaan biasa adalah buku-buku yang terletak pada suatu tempat. Perbedaan ketiga, perpustakaan *digital* bisa dinikmati pengguna di mana saja dan kapan saja, sedangkan pada perpustakaan biasa pengguna menikmati di perpustakaan dengan jam-jam yang telah diatur oleh kebijakan organisasi perpustakaan (Ruddamayanti, 2019).

2.2 Pemrograman Web

Berikut akan dipaparkan beberapa komponen-komponen bahasa pemrograman yang dipakai dalam penelitan ini.

2.2.1 HTML

HTML adalah singkatan dari *Hypertext Markup Language*, yaitu bahasa (aturan) standar yang digunakan untuk menampilkan teks, gambar, video dan audio ke dalam halaman *web*. HTML merupakan *file* text yang tersusun atas elemen-elemen yang disebut dengan tag. Dokumen atau *file* HTML yang dapat dibuat dengan menggunakan aplikasi text *editor* apa saja, dan disimpan dengan eksistensi .*html* atau .*htm* (Junaedi EP, 2005).

Generasi baru dari HTML adalah HTML*5* yang dirancang untuk memperbaiki teknologi HTML versi sebelumnya agar dapat mendukung teknologi multimedia terbaru dan tipe isi halaman *web (content)* lainnya. HTML *5* menyediakan elemen-elemen atau tag baru yang sebelumnya tidak tersedia dalam HTML versi sebelumnya. Untuk menyatakan (memberitahukan kepada *web* browser) bahwa dokumen HTML yang dibuat adalah dokumen HTML*5*, maka perlu dituliskan baris kode dibagian paling atas dokumen berupa <!DOCTYPEHTML> (Junaedi EP, 2005).

Adapun HTML merupakan suatu format data yang digunakan untuk membuat dokumen *hypertext* yang dapat dieksekusi dari satu platform komputer ke platform komputer lainya tanpa perlu melakukan suatu perubahan apapun dengan suatu alat tertentu (Junaedi EP, 2005).

2.2.2 CSS

CSS atau *Cascade Style Sheet* adalah suatu bahasa yang bekerja sama dengan dokumen HTML untuk mendefinisikan bagaimana suatu isi halaman

web ditampilkan atau dipresentasikan. Presentasi ini meliputi *style* atau gaya text, *link*, maupun tata letak (*layout*) halaman (Zaki, 2009).

Dengan adanya teknologi seperti ini, kita dapat memilah atau memisah antara kode untuk isi halaman *web* dan kode yang diperlukan khusus untuk menangani tampilan.

Pembuatan aplikasi *web* ataupun *web* responsif secara lebih mudah dapat dilakukan dengan menggunakan bantuan Bootstrap merupakan *framework* maupun *tools* yang terdiri dari CSS dan HTML untuk menghasilkan *grid*, *layout*, *typography*, *table*, *form*, *navigation*, dan lain-lain. Di dalam Bootstrap juga sudah terdapat jQuery *plugins* untuk menghasilkan komponen *user interface* yang cantik seperti *transitions*, modal, *dropdown*, *scrollspy*, *tooltip*, *tab*, *pop over*, *alert*, *button*, *carousel*, dan lain-lain (Juanda, 2008).

Dengan bantuan *Bootstrap*, *web* responsif dapat dibuat dengan cepat dan mudah dan dapat berjalan sempurna pada *browser-browser* populer seperti Chrome, Firefox dan Microsoft Edge (Juanda, 2008).

2.2.3 PHP

PHP adalah singkatan dari *Hypertext Preprocessor* yang digunakan sebagai bahasa *script server-side* dalam pengembangan *Web* yang disisipkan pada dokumen HTML (Peranginangin, 2006).

PHP ini script yang ditempatkan dalam *server* dan diproses di *server*. Hasilnya yang akan dikirim ke klien, tempat pemakai menggunakan *browser*. Salah satu kelebihan dari PHP adalah mampu untuk berkomunikasi dengan *database*. Dengan demikian, menampilkan data yang bersifat dinamis, yang ambil dari *database*, merupakan hal yang mudah untuk diimplementasikan. Itulah sebabnya sering dikatakan bahwa PHP sangat cocok untuk membangun halaman web dinamis (Peranginangin, 2006).

Beberapa kelebihan PHP dibandingkan bahasa pemrograman yang lain, yaitu:

1. Cara koneksi dan *query database* yang sederhana.
2. Dapat bekerja pada sistem operasi berbasis Windows, Linux, Mac OS, dan kebanyakan varian UNIX.
3. Biaya yang dibutuhkan untuk menggunakan PHP tidak mahal atau bahkan gratis.
4. Mudah dibangun karena memiliki fitur dan fungsi khusus untuk membuat *web* dinamis. Bahasa pemrograman PHP dirancang untuk dapat dimasukkan dalam HTML *(embedded script*).
5. *Security system* yang cukup tinggi.
6. Waktu eksekusi yang lebih cepat dibandingkan dengan bahasa pemrograman *web* lainnya yang berorientasi pada *serverside* scripting.
7. Akses ke sistem database yang lebih fleksibel dan mudah, seperti pada MySQL.

2.3 JavaScript

JavaScript adalah bahasa *scripting* kecil, ringan, berorientasi objek yang ditempelkan pada kode HTML dan di proses di sisi *client*. JavaScript digunakan dalam pembuatan *website* agar lebih interaktif dengan memberikan kemampuan tambahan terhadap HTML melalui eksekusi perintah di sisi browser. JavaScript dapat merespon perintah user dengan cepat dan menjadikan halaman web menjadi responsif. JavaScript memiliki struktur sederhana, kodenya dapat disisipkan pada dokumen HTML atau berdiri sebagai satu kesatuan aplikasi (Gok & Khanna, 2013).

2.3.1 Vue.js

Vue.js adalah JavaScript *framework* yang dikembangkan untuk membangun antarmuka suatu software. Vue.js telah menyediakan berbagai macam fungsi JavaScript yang telah dimodifikasi sehingga programmer dapat lebih mudah untuk membangun software, tentunya dengan aturan-aturan tertentu. Selain memudahkan dalam pengembangan, Vue.js juga memberi kemudahan kepada pengguna software dalam menggunakan software itu sendiri dengan *realtime response,* di mana Vue.js meminimalkan waktu antara aksi user dengan respons perangkat lunak. Vue.js pertama kali dirilis pada Februari 2014 oleh Evan You setelah bekerja di Google menggunakan Angular.js di beberapa proyek (Gok & Khanna, 2013).

Berbeda dengan *framework* lainnya yang menggunakan prinsip MVC (Model-View-Controller). Vue.js hanya difokuskan untuk membangun tampilan / hanya bekerja pada *view layer*. Dalam hal lainnya Vue.js juga dapat membantu programmer untuk membuat web dengan aplikasi single page yang canggih dan dapat dikombinasikan dengan *library browser* lainnya.Selain itu Vue.js menggunakan *template* sintaks berbasis HTML yang memungkinkan pengguna untuk mendeklarasikan *data* / *state* ke dalam DOM. Semua template Vue.js adalah HTML yang *valid* yang dapat diuraikan oleh *browser* sesuai spesifikasi dan *parser* HTML (Gok & Khanna, 2013).

2.4 Basis Data

Basis data adalah kumpulan data yang saling berelasi. Data sendiri merupakan fakta mengenai objek, orang dan lain-lain. Data dinyatakan dengan nilai baik itu angka, deretan karakter, atau simbol.

Basis data bertujuan untuk mengatur data sehingga diperoleh kemudahan, ketepatan dan kecepatan dalam pengembalian kembali. Untuk mencapai tujuannya,

syarat sebuah basis data yang baik adalah sebagai berikut: tidak adanya redudansi inkonsistensi data, dan kesulitan pengaksesan data (Peranginangin, 2006).

2.4.1 SQL

SQL (*Structured Query Language*) adalah sebuah bahasa yang dipergunakan untuk mengakses data dalam basis data relasional. Bahasa ini secara *de facto* merupakan bahasa standar yang digunakan dalam manajemen basis data relasional (Peranginangin, 2006).

SQL termasuk bahasa komputer yang mengikuti standar ISO dan ANSI (American National Standard Institute), dimana pertama kali ditetap dalam IBM pada tahun 1970, standar tersebut tidak bergantung pada mesin atau komputer yang digunakan. Pada umumnya semua perangkat lunak atau software database (DBMS) ini mengerti SQL atau mengenal SQL, sehingga perintah pada setiap perangkat lunak database hampir memiliki perintah yang sama (Peranginangin, 2006).

Pada umumnya terdapat 3 (tiga) jenis perintah SQL yang bisa digunakan oleh SQL, yaitu: DDL (*Data Definition Language*), DML (*Data Manipulation Language*), dan DCL (*Data Control Language*).

DDL merupakan perintah SQL yang berhubungan dengan pendefinisian suatu struktur database, dalam hal ini database dan *table*. Beberapa perintah dasar yang termasuk DDL ini antara lain :

- CREATE

  Perintah CREATE digunakan untuk membuat sesuatu, dalam hal ini adalah database dan *table*.
- ALTER

  Perintah ALTER digunakan untuk merubah struktur atau mengubah informasi. Perintah ALTER bisa digunakan untuk database ataupun *table*.

- RENAME

  Perintah RENAME biasanya digunakan untuk mengubah nama *table*, apabila sebuah tabel ingin diganti namanya.
- DROP

  Perintah DROP digunakan untuk menghapus, maka apabila menggunakan perintah ini harus berhati-hati karena drop dapat mengakses *database*, *table*, kolom, *index*, *procedure* dan yang lainnya.

DML merupakan perintah SQL yang berhubungan dengan manipulasi atau pengolahan data atau *record* dalam tabel. Perintah SQL yang termasuk dalam DML antara lain:

- SELECT

  Perintah SELECT digunakan untuk menampilkan data-data yang ada di dalam tabel pada suatu database.
- INSERT

  Perintah INSERT digunakan untuk menambahkan data pada tabel yang terdapat di dalam database.
- UPDATE

  UPDATE digunakan untuk mengubah data, atau memodifikasi data yang terdapat di dalam tabel.
- DELETE

  Perintah DELETE digunakan untuk menghapus data atau record di dalam tabel.

DCL merupakan perintah SQL yang berhubungan dengan manipulasi *user* dan hak akses (*privileges*). Perintah SQL yang termasuk dalam DCL antara lain:

- GRANT

Perintah GRANT digunakan untuk memberikan hak akses atau izin pada user di *database* untuk dapat mengakses *database* tersebut. Selain itu perintah *grant* juga dapat digunakan untuk menambahkan *user* atau pengguna baru di DBMS.

- REVOKE

  Perintah REVOKE adalah kebalikan dari perintah GRANT, perintah *revoke* digunakan untuk menghapus atau atau mencabut izin hak akses.

2.5 Progressive Web Apps (PWA)

Progressive Web Apps (PWAs) merupakan aplikasi berbasis web yang sama seperti halaman *web* ataupun situs *web* biasa hanya saja menawarkan fungsionalitas yang berbeda seperti dapat bekerja secara *offline*, dapat memberikan *push notification* dan akses *hardware* yang tersedia untuk aplikasi *native*. PWA ini sendiri dikembangkan oleh Google. Progressive Web Apps menggunakan beberapa teknologi diantaranya *Hypertext Transfer Protocol Secure* (HTTPS), *Manifest* dan *Service Workers*. (Juanda, 2008)

Pada Mei 2016 Google memperkenalkan Progressive Web Apps. Progressive Web Apps dirancang oleh Frances Berriman dan Google Chrome Engineer Alex Russel. Teknologi dibalik Progressive Web Apps adalah *service worker*. Menurut google developer karakteristik dari Progressive Web Apps adalah sebagai berikut :

- ***Progressive*** – dapat digunakan oleh semua pengguna, terlepas browser apa yang digunakan karena aplikasi dikembangkan secara *progressive*
- ***Responsive*** – dapat digunakan pada semua perangkat mulai dari desktop, tablet, smartphone dan lainnya.
- ***Connectivity independent*** – memiliki *service workers* untuk dapat diakses secara *offline* atau pada kualitas koneksi jaringan yang rendah.
- ***App-like*** – terasa seperti aplikasi karena model aplikasi shell memisahkan fungsi dari konten aplikasi.

- ***Fresh*** – selalu *up-to-date* dikarenakan update proses dari *service worker.*
- ***Safe*** – dilayani via HTTPS untuk mencegah pengintaian dan untuk memastikan konten tidak dirusak.
- ***Discoverable*** – yaitu diidentifikasi sebagai "aplikasi" berkat cakupan W3C dan ruang lingkup pendaftaran *service worker*, yang memungkinkan mesin pencari menemukannya.
- ***Re-engageable*** – yaitu membuat keterlibatan ulang menjadi mudah dengan adanya fitur seperti *push notification.*
- ***Installable*** – memungkinkan user untuk menambahkan aplikasi ke *home screen* tanpa melalui *appstore.*
- ***Linkable*** – dapat berbagi dengan mudah melalui tautan dan tidak memerlukan instalasi.

Progressive Web Apps memiliki tiga kebutuhan yang harus dipenuhi, yaitu :

- Harus dilayani melalui *Hypertext Transfer Protocol Secure* (HTTPS).
- Memiliki *web app manifest*
- Memiliki setidaknya satu *service worker.*

2.5.1 Single-page Application

Single page application adalah pendekatan pengembangan web di mana seluruh aplikasi bertempat di satu halaman. Dalam *single page application,* tidak terjadi halaman penuh yang refresh ketika aplikasi dijalankan. Sebaliknya, logika presentasi dimuat di depan dan disajikan dari segi tampilan yang berganti dalam wilayah konten. Seringkali format data yang digunakan dalam komunikasi *single page application* adalah teks berformat JSON (Scott, 2016).

Single page application adalah aplikasi web di mana seluruh pengalaman pengguna terkandung dalam satu halaman web. Arsitektur *single*

page application memberdayakan aplikasi web *mobile* menjadi lebih seragam. Aplikasi ini terus-menerus menjalankan halaman yang sama dengan pandangan yang berbeda atau konten, tapi tanpa reload atau menavigasi pergi ke sumber lain. Hal ini dicapai dengan memisahkan data dari lapisan presentasi dan sangat bergantung pada JavaScript (Gok & Khanna, 2013).

2.5.2 Web App Manifest

Web app manifest merupakan *file JSON* yang memberitahukan *browser* tentang aplikasi *web* yang dijalankan seperti bagaimana harusnya berperilaku ketika dilakukan instalasi pada perangkat selular atau *desktop*, ini diperlukan untuk memunculkan permintaan "Tambahkan ke Layar Beranda" (Scott, 2016).

2.5.3 Lighthouse

Lighthouse adalah aplikasi *open source* yang dibangun oleh Google untuk menguji konsep PWA dengan aspek-aspeknya, *performance*, *accessibility* dan *best practices*. *Lighthouse* dapat dijalankan dari *browser* pengguna berupa ekstensi pada Google Chrome. Pengujian menggunakan *Lighthouse* hanya bisa dijalankan untuk satu pengujian pada satu *browser client*. Untuk melakukan pengujian menggunakan *Lighthouse* pada banyak *client* digunakan layanan *third-party* Calibre (Juanda, 2008).

Pengguna dapat menjalankan *Lighthouse* di Chrome DevTools, dari baris perintah, atau sebagai modul node. Dengan memberikan *url* untuk diaudit, *Lighthouse* akan menjalankan serangkaian audit terhadap halaman, dan menghasilkan serangkaian laporan tentang seberapa baik halaman itu. Dari sana, gunakan audit yang gagal sebagai indikator tentang cara meningkatkan halaman. Setiap audit memiliki dokumen rujukan yang menjelaskan mengapa audit itu penting, serta cara memperbaikinya.

Kategori penilaian pada *Lighthouse* terdiri dari *performance*, *accessibility*, *best practices*, *SEO*, dan *Progressive Web App*. Pada kategori *performance*, aspek yang dinilai adalah waktu untuk berinteraksi, *latency*, indeks kecepatan, optimisasi *resource*, TTFB, penerimaan aset, waktu untuk mengeksekusi *script*, dan ukuran DOM. Pada kategori *accessibility*, aspek yang dinilai adalah elemen pada halaman, bahasa yang tersedia,dan ARIA *attributes.* Pada kategori *best practice*, aspek yang dinilai adalah optimasi gambar, *JS libraries*, *browser error logging*, akses pada HTTPS, dan JS *vulnerabilities* yang diketahui. Pada kategori SEO, aspek yang dinilai adalah tampilan yang responsif, *meta*, *crawling*, *canonical*,dan struktur aplikasi. Pada kategori *Progressive Web App*, aspek yang dinilai adalah *redirect* HTTP ke HTTPS, *response code ok*, kecepatan *loading* pada jaringan 3G, *splash screen*, *viewport.*

Indeks penilaian pada *Lighthouse* terbagi menjadi tiga, yaitu buruk(0-49), sedang(50-89), dan bagus(90-100)

2.5.4 Service Worker

Service worker adalah salah satu jenis dari *web worker*, yaitu *script* yang berjalan di belakang *browser* pengguna. *Service worker* pada dasarnya adalah berkas JavaScript yang berjalan pada *thread* yang berbeda dengan *main thread browser*, menangani *network request*, *caching*, mengembali-kan *resource* dari *cache*, dan bisa mengirimkan *push message*. Aset web dapat disimpan sebagai *cache* lokal, sehingga dengan jaringan internet yang kurang memadai pun, pengguna tetap mendapat pengalaman yang baik. Aplikasi dapat tetap menjalankan halaman web yang sudah di-*cache* atau memberikan status koneksi tanpa *browser* menampilkan tulisan *error* karena ketiadaan koneksi internet (Ridho, Pinandhito, & Dewi, 2018).

Langkah-langkah dalam melakukan konfigurasi dasar *service worker* sebagai berikut:

- Daftarkan *service worker* melalui URL (*Uniform Resource Locator*) fungsi *serviceWorkerContainer*.register().
- Jika berhasil, *service worker* dijalankan di *ServiceWorkerGlobalScope*.
- *Service worker* telah siap untuk memproses *event*.
- Instalasi *service worker* dicoba ketika *service worker* mengontrol halaman yang diakses setelah dan sebelumnya. *Event install* akan selalu dikirim pertama kali ke *service worker*.
- Ketika handler on install selesai, *service worker* dipasang.
- Proses aktivasi. Ketika *service worker* terpasang, selanjutnya akan menerima *event activate*. Penggunaan utama dari *onactivate* ini adalah untuk membersihkan sumber daya yang digunakan sebelumnya.
- *Service control* sekarang dapat mengontrol halaman, tapi hanya dibuka setelah *register* telah sukses seperti dokumen mulai aktif dengan atau tanpa *service worker* dan menjaganya selama masih digunakan. Jadi dokumen harus dimuat ulang agar benar-benar terkontrol.

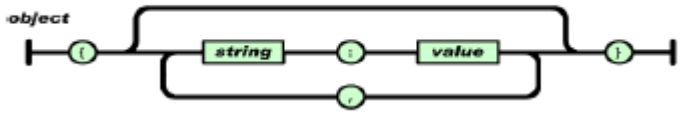
2.5.5 JSON

JSON (JavaScript Object Notation) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan 12 dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll (Scott, 2016). Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), Tabel hash (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

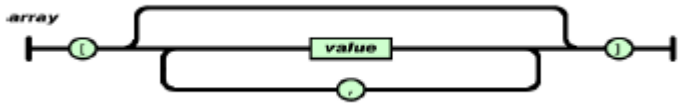
Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut:

1. Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



Gambar 2.1 **JSON Object**

2. Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).



Gambar 2.2 **JSON Array**

2.5.6 Cache

Cache interface menyediakan mekanisme penyimpanan untuk pasangan objek *request* dan *response* mau disimpan ke dalam *cache*, contohnya sebagai bagian dari daur hidup *service worker*. Perlu diketahui bahwa *cache interface* terbuka terhadap halaman web dan juga *workers*. *Cache* tidak harus selalu digunakan bersamaan dengan *service worker* walaupun *cache* tercantum di dalam spesifikasinya. *Cache* digambarkan sebagai sebuah *array* berisi objek *request* yang bertindak sebagai pasangan untuk responsnya yang disimpan di dalam *browser* (Juanda, 2008).

2.6 *User Acceptance Testing(*UAT*)*

Menurut Perry (2006), *User Acceptance Testing* atau disingkat UAT merupakan pengujian yang dilakukan oleh end-user yaitu pengguna yang langsung berinteraksi dengan sistem dan dilakukan verifikasi apakah fungsi yang ada telah berjalan sesuai dengan kebutuhan/fungsinya. Setelah dilakukan system testing, acceptance testing menyatakan bahwa sistem software memenuhi persyaratan. Acceptance testing merupakan pengujian yang dilakukan oleh pengguna yang menggunakan teknik pengujian black box untuk menguji sistem terhadap spesifikasinya. Pengguna akhir bertanggung jawab untuk memastikan semua fungsionalitas yang relevan telah diuji.

Data yang diperoleh dari hasil pengujian kemudian dinilai dengan melihat ketegorisasi tingkat kecenderungan pada masing-masing aspek yang dilakukan

dengan mengkategorikan tingkat kecenderungan. Oleh karena itu, kriteria diperlukan yaitu: rata-rata ideal dan simpangan baku ideal, skor tertinggi ideal, serta skor terendah ideal yang dapat dicapai pada instrumen. Menurut Arikunto (2002), pengelompokan tersebut menggunakan rumus yang dapat dilihat pada Tabel 2.1.

Tabel 2.1 **Rumus Pengkategorian**

Interval rata-rata skor	**Kategori**
$x > Mi - 1{,}5\ SDi$	Sangat Layak
$Mi + 0{,}5\ SDi < x \leq Mi + 1{,}5\ SDi$	Layak
$Mi - 0{,}5\ SDi < x \leq Mi + 0{,}5\ SDi$	Cukup
$Mi - 1{,}5\ SDi < x \leq Mi - 0{,}5\ SDi$	Tidak Layak
$x \leq Mi - 1{,}5\ SDi$	Sangat Tidak Layak

Keterangan:

Mi = 1/2 x (skor tertinggi + skor terendah)
SDi = 1/6 x (skor tertinggi - skor terendah)
Mi : Mean ideal (Rata-rata ideal)
SDi : Standard Deviant ideal (Simpangan baku ideal)
x : Skor aktual