REPOSITOR, Vol. 4, No. 2, Mei 2022, Pp. 125-136
ISSN : 2714-7975
E-ISSN : 2716-1382

Penerapan *Progressive Web Application* Pada website *Online Public Access Catalog* (OPAC) UMM

Muchsin Bin Jafar Al Hamid*[1], Ilyas Nuryasin[2], Zamah Sari[3]
[1,2,3]Universitas Muhammadiyah Malang
Muchsinalhamid@webmail.umm.ac.id*

Abstrak

Online Public Access Catalog (OPAC) adalah sistem pencarian informasi yang berisikan catatan bibliografi pendek berupa Jurnal, Buku dan materi audio-visual yang tersedia pada perpustakaan. Kelemahan yang dimiliki OPAC | LASer UMM berbasis web saat ini adalah tidak mempunyai keunggulan dari web modern seperti penggunaan mode offline, dapat di install pada homescreen dan dapat memberikan User Experience (UX) seperti aplikasi native. Dengan Progressive Web Apps (PWA) untuk mengatasi keterbatasan tersebut dengan menggunakan teknologi baru dari web browser seperti Service Worker, Web Application Manifest dan Architecture Application Shell. Tujuan dari penelitian ini adalah membuat website OPAC | UMM Library Automation Services (LASer) dapat diakses secara offline dan berbasis mobile app menggunakan Progressive Web Apps (PWA). Hasil dari implementasi PWA pada sistem OPAC UMM menunjukan bahwa dengan service worker dapat membuat website tidak bergantung pada konektivitas jaringan internet sehingga dapat diakses dalam keadaan offline. Dengan menggunakan Manifest dan shell application, website dapat di tambahkan ke layar utama pada smartphone mobile dan desktop. Sedangkan fitur indexedDB dalam sistem website dapat menyimpan data ke browser walaupun tidak ada konektivitas dan akan mengirim setelah jaringan terhubung kembali. Hasil pengujian dengan metode baseline checklist telah terpenuhi serta mendapatkan skor diatas 90 dari 100 dan PWA sudah diimplementasikan dengan sempurna pada sistem online public access catalog.

Kata Kunci: *Progressive Web Application, OPAC, Service Worker, Library, Native Apps*

Abstract

Online Public Access Catalog (OPAC) is an information retrieval system that contains short bibliographic notes in the form of journals, books and audio-visual materials available in libraries. Weaknesses owned by OPAC | The current web-based UMM LASer does not have the advantages of the modern web such as the use of offline mode, can be installed on the homescreen and can provide User Experience (UX) like native applications. With Progressive Web Apps (PWA) to overcome these limitations by using new technologies from web browsers such as Service Workers, Web Application Manifest and Architecture Application Shell. The purpose of this research is to create a website OPAC | UMM Library Automation Services (LASer) can be accessed offline and based on mobile app using Progressive Web Apps (PWA). The results of the PWA implementation on the UMM OPAC system showed that with a service worker it was possible to make a website independent of internet network connectivity so that it could be accessed offline. By using the Manifest and shell application, websites can be added to the main screen on mobile and desktop smartphones. Meanwhile, the indexedDB feature in the website system can save data to the browser even though there is no connectivity and will send it after the network is reconnected. The test results with the baseline checklist method have been met and get a score above 90 out of 100 and the PWA has been implemented perfectly in the online public access catalog system.

Keywords: *Progressive Web Application, OPAC, Service Worker, Library, Native Apps*

1. Pendahuluan

Perpustakan pada perguruan tinggi dapat diibarakan sebagai ‘jantung’ dari perguruan tinggi tersebut. Hal itu karena semua penelitian dan pengetahuan berupa Jurnal, Buku dan Paper ada di dalam perpustakaan. Seiring dengan perkembangan zaman , Perpustakaan masa kini telah mengeimplementasikan perpustakaan berbasis teknologi informasi yang disebut sebagai

OPAC (*Online Public Access Catalog*)[1]. Pada akhir tahun 1990-an, *OPAC* berbasis web mulai diimplementasikan pada perpustakaan di perguruan tinggi. Kemajuan tersebut berfungsi sebagai akses jarak jauh oleh pengguna untuk mendapatkan sumber atau informasi yang ada pada perpustakaan di perguruan tinggi [2].

Online Public Access Catalog (OPAC) adalah sistem pencarian informasi yang berisikan catatan bibliografi pendek berupa Jurnal , Buku dan materi audio-visual yang tersedia pada perpustakaan [3]. Dengan adanya *OPAC* pengguna dapat dengan cepat mencari sumber daya perpustakaan dan mengambil cacatan bibliografi tanpa bantuan dari perantara manusia seperti anggota atau staf perpustakaan. Dalam hal ini , memungkinkan pengguna dapat mencari dokumen perpustakaan yang di akses dari terminal dan juga memungkinkan untuk melakukan pencetak , pengunduhan atau mengekspor catatan melalui *electronic mail* (*E-mail*) [4].

Perpustakaan di perguruan tinggi kampus Universitas Muhammadiyah Malang (UMM) juga menyediakan fasilitas seperti *Library Automatic Service* (LASer) untuk mempermudah mahasiswa mencari buku, jurnal, hasil penelitian dan bahan pustaka atau informasi yang di miliki perpustakaan dengan cepat menggunakan *Online Public Access Catalog* (*OPAC*). *OPAC* | *UMM Library Automation Services (LASer)* dirancang dan dibangun menggunakan aplikasi berbasis web dan dapat di akses oleh pengguna secara *online* baik di semua *platform*. *OPAC* UMM dibuat oleh tim MDRLG perpustakaan kampus Universitas Muhammadiyah Malang dengan menggunakan bahasa pemrograman *PHP* dan *database MYSQL* yang bersifat *open source* [5].

OPAC | *UMM Library Automation Services (LASer)* dapat diakses pada *smartphone* dan komputer tanpa terjadi *error* atau masalah pada website *http://laser.umm.ac.id*. Kelemahan yang dimiliki OPAC | LASer UMM berbasis web saat ini adalah tidak mempunyai keunggulan dari web *modern* seperti penggunaan *mode offline*, dapat di *install* pada *homescreen* dan dapat memberikan *User Experience (UX)* seperti aplikasi *native*. Oleh sebab itu, agar website dapat mendukung keunggulan dari web *modern* tersebut maka akan dibangun dengan *Progressive Web Apps* (PWA) untuk mengatasi keterbatasan tersebut dengan menggunakan teknologi baru dari *web browser* seperti *Service Worker*, *Web Application Manifest* dan *Architecture Application Shell*.

Progressive Web Application merupakan sebuah aplikasi web yang menggunakan teknologi paling terbaru dan mutakhir. *PWA* sebenarnya hanya aplikasi berbasis website biasa, namun memanfaatkan fitur yang modern agar tampilan mirip dengan aplikasi asli. *Progressive Web Application* dapat digambarkan sebagai kumpulan dari teknologi , konsep desain dan *WEB API* (*Application Programming Interface*) yang bekerja sama untuk memberikan sentuhan aplikasi pada sebuah *mobile web* [6].

Progressive Web App (PWA) adalah ide yang pertama kali yang di perkenalkan oleh google Alex Russell pada bulan Juni 2015 [7]. Konsep *PWA* mencakup penerapan pada teknologi baru dari *web browser* seperti *service worker* dan *application manifest*. *Progressive Web Application* mempunyai karakteristik utama yang dapat di andalkan (*reliable*), cepat (*fast*), dan menarik (*engaging*) yang dapat memastikan pengguna mendapatkan pegalaman terbaik dalam menggunakan suatu aplikasi web walaupun dalam jaringan yang tidak stabil ataupun *offline* sekalipun.

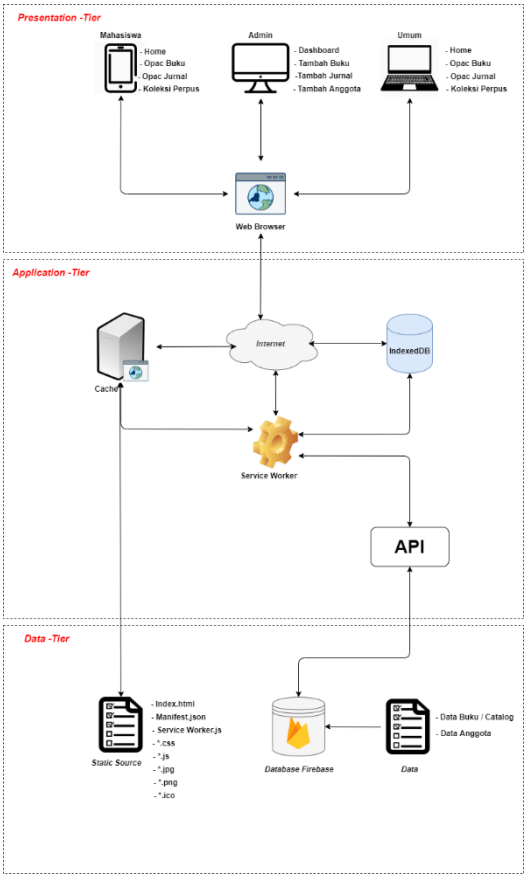
Berdasarkan permasalahan tersebut, maka dalam penelitian ini penulis akan melakukan tema tugas akhir dengan judul "Penerapan *Progressive Web Apps (PWA)* Pada Website *Online Public Access Catalog (OPAC)*" dengan studi kasus yang diambil pada website *Library Automation Services (LASer)* Universitas Muhammadiyah Malang. Penelitian ini diharapkan dapat menghasilkan sebuah sistem dari *OPAC* | *UMM Library Automatic Services (LASer)* berbasis web menggunakan *Progressive Web Application* (PWA) yang memiliki kemampuan web *modern* disaat ini yang dapat di akses secara *offline* atau dalam kondisi buruk sekalipun, terlihat seperti aplikasi native, dapat diinstall pada *homescreen* perangkat pengguna dan dapat menampilkan *splashscreen* ketika awal aplikasi dibuka.

2. Metode Penelitian

2.1 Analisis Sistem

Bagian dari analisis sistem ini menghasilkan sebuah arsitektur sistem yang akan dibangun pada website *Online Public Access Catalog (OPAC)* Universitas Muhammadiyah Malang dan akan menggambarkan kebutuhan fungsional pada sistem secara umum. Website *OPAC* | *UMM Library Automation Services (LASer)* akan dibangun menggunakan *framework Angular javascript* sebagai *server side* dan *client side* sebagai interaksi dari para pengguna.

Desain aplikasi perangkat lunak untuk *client* dan *server side* akan dibagi menggunakan model arsitektur *three tier*. Arsitektur *three tier* adalah evolusi dari model tradisional seperti *two tier* karena dapat meningkatkan peminat terutama untuk aplikasi dengan bisnis yang besar [8]. Arsitektur ini memiliki tiga bagian atau 3-*tier* yaitu *presentation*, *application* dan *data tier*. Ketiga arsitektur tersebut memiliki peran masing-masing dan dapat memudahkan interaksi antara *client* dan *server*.



Gambar 1. Arsitektur *Three Tier OPAC UMM*

Pada Gambar 1 *presentation-tier* menggambarkan *user interface* yang bisa diakses oleh beberapa pengguna seperti mahasiswa, admin dan umum. Ketiga pengguna tersebut digambarkan dalam *client side* dimana setiap pengguna akan mengakses situs atau website melalui *web browser*. Setiap pengguna akan dibedakan berdasarkan aturan yang telah dimiliki oleh sistem sebelum mengaksesnya. Pengguna seperti mahasiswa dan umum hanya dapat mengakses halaman luar saja seperti halaman *home*, pencarian *OPAC* buku / Jurnal dan melihat koleksi apa saja yang dimiliki oleh perpustakaan universitas muhammadiyah malang. Sedangkan admin mempunyai hak akses tersendiri yaitu dapat memasuki halaman *dashboard*, menambahkan buku atau jurnal, dan menambahkan anggota perpustakaan serta dapat mengakses seluruh halaman yang dapat diakses oleh mahasiswa dan pengguna umum.

Application-tier merupakan bagian *serverside* yang bertugas untuk mengontrol pertukaran data pada antara *clien* dan *server*. Seluruh alur dari pertukaran data antara *client* dan *server* akan melewati *service worker* yang dimiliki oleh *browser*. *Service worker* akan bertugas mengontrol file *static* yang nantinya akan disimpan kedalam *cache* serta dapat menyimpannya di *indexedDB* yang terdapat pada *browser client*.

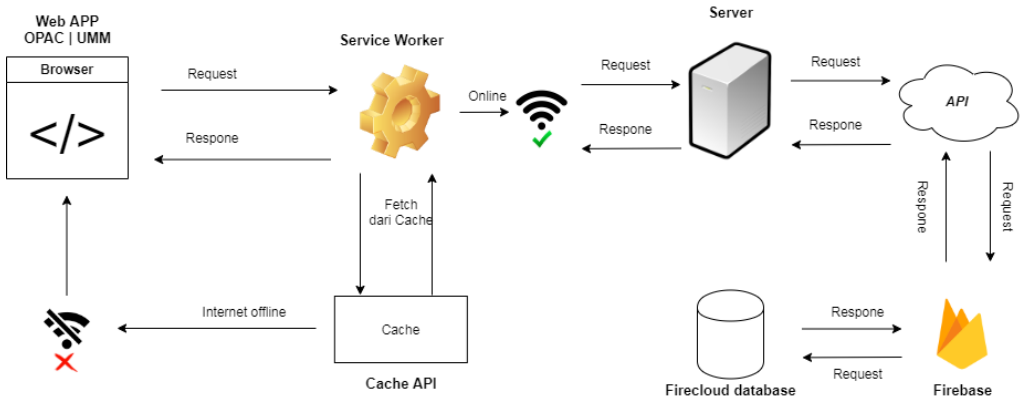
D*ata-tier* berisi tentang sekumpulan basis data yang tersimpan pada *database firebase* dan *static resource* yang tersimpan pada *web browser* pengguna. Dalam hal ini, basis data yang digunakan adalah *NoSQL* pada *database firebase* dan menghasilkan dokumen dengan format *JSON*. Sedangkan file yang tersimpan di *static resource* pada *cache browser* pengguna adalah file *index*, *webmanifest*, *css*, *js*, dan format pada file *image* seperti *PNG*, *JPG* dan sebagainya.

2.2 Perancangan Sistem

Tahap ini mempunyai beberapa perancangan yang akan diterapkan pada web *OPAC | UMM Library Automation Service (LASer)*. Perancangan yang dihasilakna antara lain adalah perancangan *progressive web application* pada OPAC UMM, perancangan kemampuan *offline* dan perancangan *database* pada sistem.

2.2.1 Perancangan *PWA OPAC UMM*

Arsitektur dari *progressive web application* mempunyai 3 bagian utama yaitu, web *application manifest*. *application shell* dan *service worker*. Bagian pertama yaitu *web application manifest* pada file manifest. json yang digunakan sebagai konfigurasi *progressive web application* yaitu nama aplikasi, deskripsi, *icon*, *theme color*, *background color* dan *orientation*. Selain itu, kemampuan dari file manifest. json juga dapat memunculkan sebuah *pop-up* "*add to homescreen*" pada browser pengguna untuk menambahkannya ke layar utama sebagai aplikasi. Bagian kedua adalah *application shell* atau kerangka aplikasi yang terdiri dari beberapa jenis file yaitu *HTML*, *CSS* dan *Javascript* serta sumber lainnya yang terdapat pada sistem website *online public access catalog UMM*. Bagian yang terakhir ialah *service worker* sebagai *application server* dan penghubung diantara *client side* dan *server side* serta dapat membuat website *Online Public Access Catalog UMM* dapat di akses secara *offline*.



Gambar 2. Arsitektur Progressive Web Application Pada *OPAC* UMM

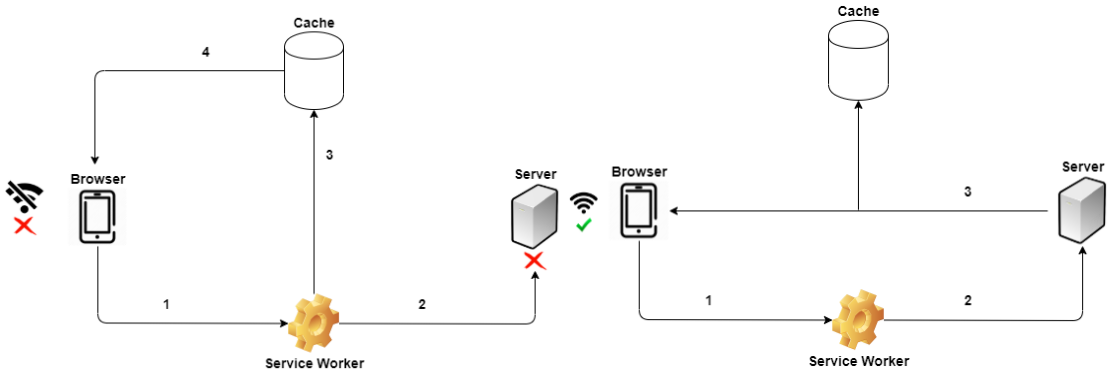
Pada strategi diatas, pertama-tama *service worker* akan mengecek apakah jaringan memberikan *respone*. Jika berhasil, data akan di kembalikan ke halaman web. Jika gagal maka *service worker* akan mengembalikan data dari *cache*. Strategi ini akan di pakai ketika membutuhkan data yang selalu *update* seperti *respone* dari *API*.

Service Worker akan meneruskan setiap permintaan atau *request* berjenis *GET* dari halaman web ke *server* dalam kondisi *online*, lalu akan menduplikasi *respone* dari *server* dan disimpan kedalam *cache* pada *browser*. *Respone* dari *server* akan diteruskan kembali ke halaman web seperti pada gambar 2. Setiap halaman yang telah di kunjungi oleh *user* atau pengguna secara otomatis akan tersimpan kedalam *cache browser*. Kondisi inilah yang nantinya akan dapat diakses oleh pengguna pada saat tidak ada koneksi internet atau secara *offline*. Namun pengguna tidak akan bisa mengirim data pada server karena tidak ada koneksi internet.

Untuk pengujian dari *progressive web application*, web haruslah memiliki *domain* dan juga layanan *hosting* agar dapat di akses. Peneliti akan menggunakan salah satu layanan dari *google* yaitu *firebase*. Dengan menggunakan *firebase* , peneliti akan lebih mudah untuk menerapkan atau mingimplementasikan *progressive web application* tanpa harus berurusan dengan masalah *backend* [9].

2.2.2 Perancangan Kemampuan *Offline*

Service worker bekerja seperti penghubung setiap *request* yang nantinya akan dikirim kepada *server*. *Service worker* selalu memeriksa *request* yang dikirimkan ke *cache*, jika ada pada *cache*, *request* yang dikirimkan akan di teruskan kepada *cache* tersebut. Jika tidak ada pada *cache* tersebut maka *request* yang dikirim tersebut akan di teruskan pada *server*. Gambar 3 menunjukan alur kerja *service worker* dalam kondisi jaringan *online*.



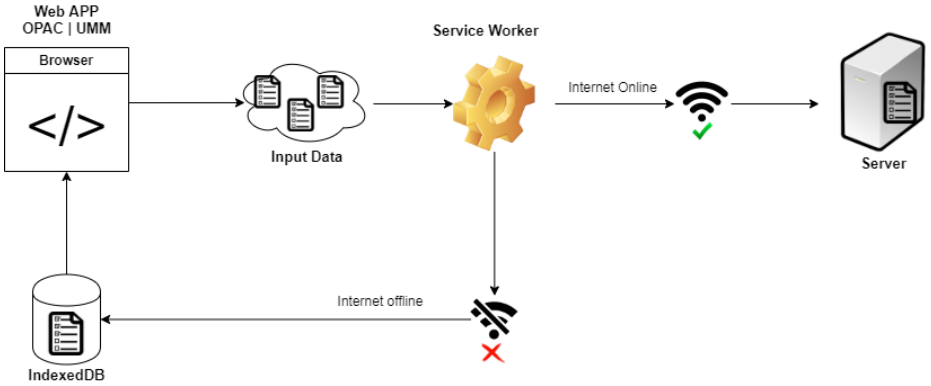
Gambar 3. Sercive Worker kondisi online dan offline

Di lain sisi, *service worker* akan menduplikasi setiap *request* yang baru dibuat dan menyimpan data tersebut ke dalam *cache browser* pengguna dan selanjutnya akan di teruskan ke halaman web. Hal ini bisa menjadikan setiap halaman yang telah dikunjungi dapat tersedia dalam jaringan *offline*. Gambar 3 menunjukan alur kerja *service worker* dalam kondisi jaringan *offline*.

Ketika kondisi sedang *offline*, pengguna tidak dapat mengakses semua halaman karena tidak ada *respone* dari *server* dan akan dikembalikan pada halaman yang telah tersimpan pada *cache*. Halaman bisa di tampilkan apabila halaman tersebut telah ada pada *cache* atau jejak terakhir dari pengguna sebelum terjadi kondisi *offline*.

2.3.3 Perancangan *IndexedDB*

IndexedDB merupakan media penyimpanan lokal *non-sql* yang dimiliki oleh berbagai *browser* [10]. Pada *indexedDB* kita dapat menyimpan seluruh komponen yang dikirim pada aplikasi atau situs walaupun jaringan internet dalam keadaan *offline*. Fitur seperti ini sangat bagus untuk memastikan setiap *request* dari *user* yang dikirim dan akan tersimpan di dalam *database browser*.

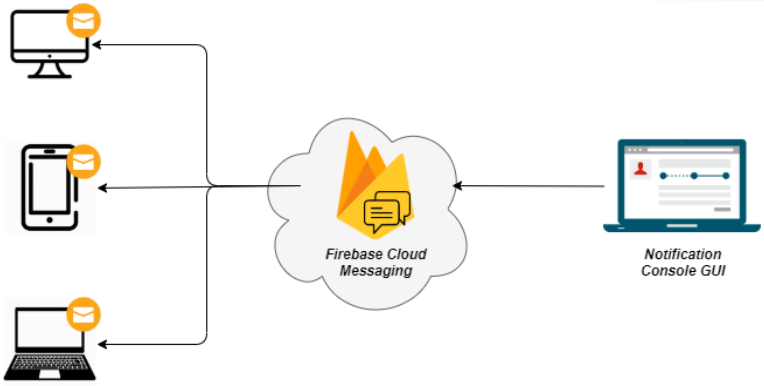


Gambar 4. Perancangan *IndexedDB* di *browser*

Pada saat pengguna menginput data pada website *OPAC*, data tersebut diproses melalui *service worker* yang akan melakukan pengecekan pada kondisi jaringan yang ditunjukan pada Gambar 4. Jika jaringan tidak terhubung atau *offline*, data yang telah dikirim akan dikembalikan dan tersimpan pada *indexedDB browser* pengguna. Penyimpanan yang terjadi di *indexedDB browser* akan tersimpan hingga jaringan terhubung kembali atau *online* dan data yang tersimpan akan secara otomatis dikirim ke *server*.

2.2.4 Perancangan *Push Notification*

Push notification adalah salah satu fitur yang diunggulkan pada aplikasi *firebase*. *Push notification* sendiri merupakan pemberitahuan pesan yang masuk secara instan ke *smartphone* ataupun *dekstop* pengguna dimana pemberitahuan tersebut akan dikirim langsung ke layar notifikasi. Arsitektur dari *push notification* dapat dilihat pada Gambar 5.



Gambar 5. Push Notification

Bentuk dari push notification mirip dengan pesan singkat seperti SMS, hanya saja push notification bisa muncul apabila pengguna telah memberi izin kepada browser untuk menampilkan notifikasi yang akan dikirimkan. Dengan menggunakan metode push, pesan atau notifikasi akan dikirimkan dari server admin (Notification Console GUI) menuju ke layar para pengguna (desktop, android/ios dan windows).

2.3 Perancangan Pengujian *PWA*

Untuk pengujian kualitas penerapan *Progressive Web Application (PWA)*, peneliti menggunakan *baseline PWA Checklist* yang akan dilakukan dengan menggunakan 2 cara yaitu menggunakan *tool* bernama *lighthouse* dan pengujian manual menggunakan *smartphone mobile*.

2.3.1 *Lighthouse*

Lighthouse merupakan aplikasi *open source* yang di bangun oleh *google* untuk menguji konsep *progressive web application (PWA)* dan kualitas aplikasi web yang dapat di jalankan sebagai ekstensi dari *google chrome* [11]. Aspek-aspek yang dapat dilakukan oleh *lighthouse* meliputi *Accesibility, performance, Best Practices, SEO* dan yang paling utama ialah pengujian terhadap *progressive web application (PWA).*

Tabel 1. Skor Dalam Penilaian Web.

Skor	Status	Warna Status
90-100	Sangat Baik	Hijau
50-89	Rata-rata	Jingga
0-49	Kurang Baik	Merah

Berdasarkan parameter penilaian suatu web dari google developer, *Lighthouse* menghasilkan skor antara 0 hingga 100 dari kelima aspek yang telah diuji. Warna dari status yang tampil pada pengujian lighthouse mempunyai 3 warna yaitu merah jingga dan hijau. Tabel 1 menunjukan bahwa jika web tersebut mendapatkan skor 90-100 maka status dari web tersebut sangat cepat atau sangat baik. Jika hasil uji memiliki nilai skor dari 50-89, maka status dari web tersebut adalah rata-rata atau standar seperti pada website umumnya. Dan jika hasil setelah diuji mendapatkan skor 0-49, maka status dari tersebut adalah lambat atau kurang baik.

Untuk *audit* dari *progressive web application* didasarkan kepada *baseline progressive web app checklist* yang memiliki 14 kriteria persyaratan agar web dapat dikatakan sebagai *progressive web application* yang dapat dilihat pada Tabel 1. Kriteria nilai dari *progressive web application* sendiri hanya 14 kriteria dan 3 kriteria lainnya akan diuji secara manual dengan menggunakan *Smartphone mobile*.

2.3.2 Pengujian Manual

Pengujian kedua adalah dengan menggunakan *smartphone mobile* untuk melihat kinerja dari *service worker* pada aplikasi website. Pengujian ini untuk menunjukan performa dari *service worker* pada website *OPAC | UMM* dengan melihat apakah website dapat diinstall pada browser manapun (*Firefox, Chrome* dan *Edge*), transisi halaman cepat tanpa memblokir jaringan dan apakah setiap halaman mempunyai *URL*.

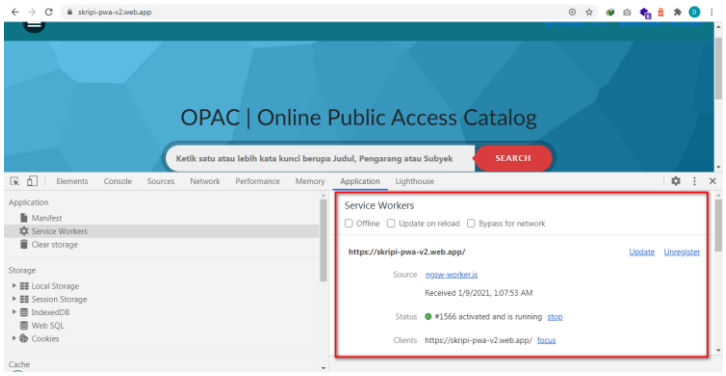
3. Hasil Penelitian dan Pembahasan

3.1 Implementasi Sistem

Dalam subbab ini akan dibahas tentang implementasi websie *online public access catalog (OPAC) UMM* yang meliputi: implementasi *service worker*, implementasi kemampuan *offline*, implementasi *push notification*, implementasi instalasi pada *smartphone* dan implementasi *indexedDB*.

3.1.1 Implementasi Service Worker

Dalam hal ini, *service worker* sebagai penghubung atau jembatan antara *client side* dan *server side*. Setiap *request* dan *respone* dari kedua sisi tersebut akan melewati *service worker* yang nantinya disimpan kedalam *cache* agar dapat berjalan pada kondisi *offline* sekalipun. Karena *service worker* hanya berjalan pada koneksi yang aman (*secure*) maka harus diimplemtasikan pada *domain* dengan protokol *HTTPS*. pembuatan *service worker* di *framework angular* berbeda dengan pembuatan file *javascript* untuk meregistrasi *service worker*. Pada *AngularJS*, *service worker* dapat dijalankan dengan cara *menginstall library* yang dimiiliki oleh *framework angular*. Dengan memanfaatkan *library progressive web application* dari *angular*, maka aplikasi website sudah memiliki *progressive web application* yang dapat dilihat pada Gambar 6.

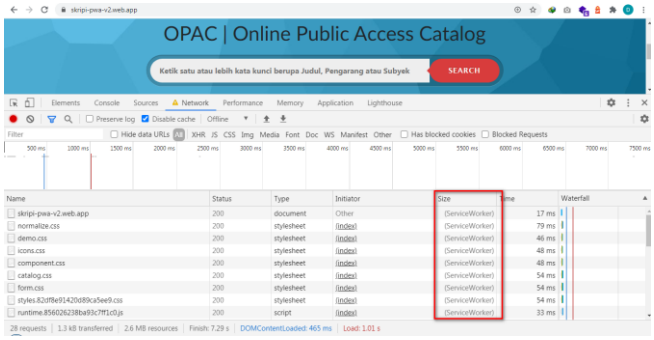


Gambar 6. Service Worker

Hasil di atas menunjukan bahwa *service worker* berhasil ditambahkan pada website *online public access catalog* dan sedang berjalan pada *browser*. Selain itu, *service worker* juga berhasil menyimpan file *static* dari sistem pada *cache storage* yang berada pada *browser* pengguna.

3.1.2 Implementasi Kemampuan Offline

Pada bagian ini menjelaskan tentang implementasi dari kemampuan *offline* yang dimiliki oleh *service worker* mengenai kemampuan aplikasi untuk berjalan walaupun tanpa koneksi internet sekalipun.



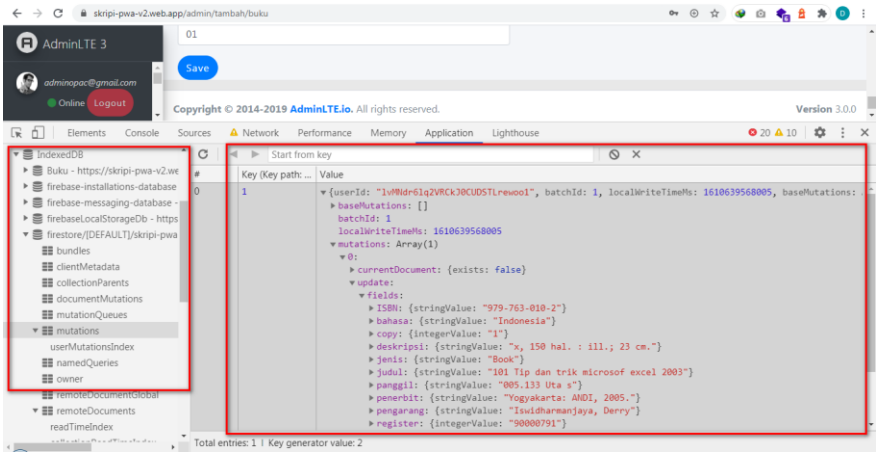
Gambar 7. Offline Cache Service Worker

Seluruh file yang ada pada aplikasi website *online public access catalog (OPAC)* telah di *cache* kedalam *service worker* yang ditunjukan pada Gambar 7. *Service worker* bertugas untuk memberikan *request* dan mengembalikan *respone* dari *client*. Jika ada *request* yang tidak

memiliki jaringan internet maka *service worker* akan mengambilnya dari *cache* yang tersimpan di *browser* pengguna. Sehingga pada keadaan jaringan yang buruk atau *offline* masih dapat mengakses website dengan sangat cepat.

3.1.3 Iplementasi *IndexedDB*

Bagian ini menjelaskan tentang implementasi dari *indexedDB* yang berada pada *browser google chrome* dan akan di implementasikan dengan *progressive web application (PWA)* sehingga dapat berjalan dalam kondisi jaringan *offline*. Dalam hal ini, uji coba akan dilakukan pada halaman admin yang akan memasukan data buku ke dalam penyimpanan *firestore*.

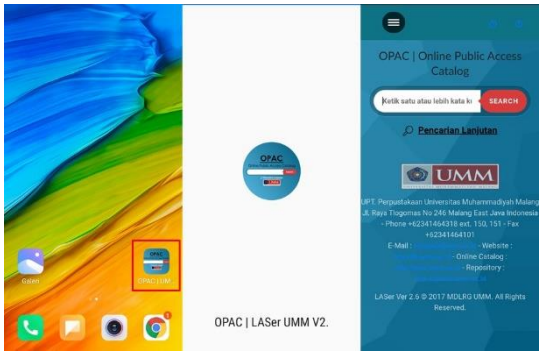


Gambar 8. Data tersimpan di IndexedDB Browser

Dari kesimpulan hasil Gambar 8 diatas, bisa dilihat bahwa data buku yang dimasukan pada halaman admin tersimpan dengan baik pada *indexedDB* walaupun tidak ada koneksi jaringan internet (*offline*). Saat jaringan internet terhubung kembali, data yang telah tersimpan pada *indexedDB* akan diteruskan pada penyimpanan basis data di *firestore* sehingga admin tidak perlu menunggu kondisi jaringan terhubung kembali (*online*) untuk menyimpan data tersebut. Begitu juga saat mengedit dan menghapus data tersebut akan disimpan kedalam *indexedDB* saat tidak ada koneksi dan akan di teruskan ke penyimpanan basis data pada *firestore* kembali saat terhubung ke jaringan internet.

3.1.4 Implementasi *Smarphone Mobile*

Setelah website mengimplementasikan teknik *progressive web application (PWA)*, maka saat pengguna mengakses website tersebut akan menampilkan sebuah *banner* untuk menambahkan aplikasi ke perangkat *mobile*. Hal itu menunjukan bahwa fitur *manifest* dari komponen *progressive web application* telah terbukti benar dan dapat berjalan dengan normal tanpa masalah sedikitpun.



Gambar 9. Aplikasi OPAC UMM versi Mobile

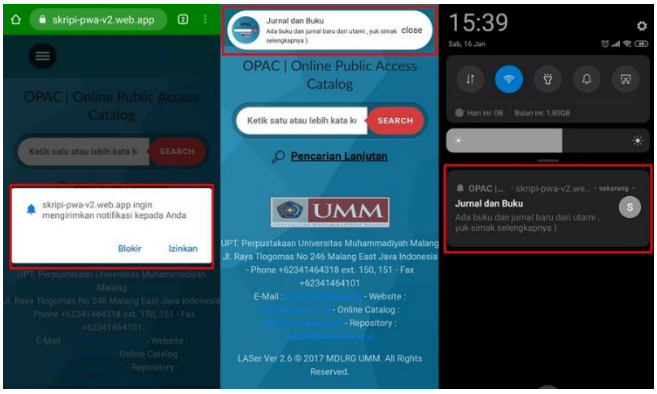
Dari hasil implementasi *web manifest* diatas menunjukan situs *online public access catalog UMM* yang telah menerapkan *progressive web application* dapat *diinstall* pada *smartphone*

mobile dan menampilkan sebuah *icon* pada layar beranda pemgguna. Sehingga para pengguna tidak perlu lagi untuk membuka browsernya dan mengetik ulang *URL* untuk menjalankan aplikasi *OPAC UMM* tetapi hanya dengan klik *icon* aplikasi yang telah terinstal di layar beranda tersebut.

Selain itu, dengan fitur *manifest* dari *PWA* juga memberikan kemampuan lainnya pada sistem seperti dapat memunculkan *splash screen* ketika diakses memalui *icon* pada layar beranda dan dapat membuat nama aplikasi sesuai dengan kebutuhan. Hasil akhirnya adalah aplikasi yang sudah terinstal pada *homescreen* tidak lagi menampilkan *address bar* seperti pada saat pengguna mengakses melalui *browser*.

3.1.5 Implementasi *Push Notification*

Push notification merupakan fitur mengirim pesan notifikasi ke perangkat pengguna. Dalam aplikasi *firebase*, memiiliki sebuah fitur yang bernama *cloud messaging* yang digunakan sebagai pengirim pesan lintas *platform*. Implementasi *push notification* dilakukan secara manual pada fitur *cloud messaging* dari aplikasi *firebase*.



Gambar 10. Push Notification Aplikasi OPAC UMM

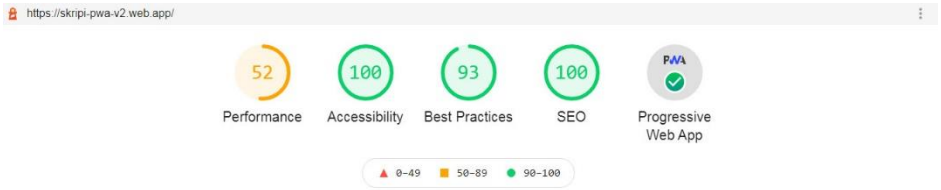
Gambar 10 menunjukan hasil dari *push notification* yang terjadi pada *smartphone mobile* di perangkat keras pengguna. Hasilnya saat pengguna sedang menggunakan aplikasi maka notifikasi yang telah di *push* dengan fitur *cloud messaging* akan muncul pada atas layar *smartphone*. Dan jika tidak sedang menggunakan aplikasi, hasilnya dapat dilihat pada layar pemberitahuan notifikasi di *smartphone mobile* pengguna.

3.2 Pengujian aplikasi

Untuk pengujian dari penerapan *progressive web application (PWA)* pada aplikasi website *online public access catalog* universitas muhammadiyah malang, akan dilakukan pengujian menggunakan dua cara yaitu dengan *tool* bernama *lighthouse* dan pengujian manual menggunakan *smartphone* berdasarkan *baseline checklist PWA*.

3.2.1 *Lighthouse*

Pada tahap ini, pengujian terkait seberapa berhasilnya sebuah web yang mengimplementasikan metode *progressive web application* di uji menggunakan *tool* bernama *lighthouse*. *Lighthouse* akan menjalankan serangkaian pengujian terhadap aplikasi web dan digunakan sebagai *benchmark* apakah aplikasi web sudah menggunakan aspek-aspek dari *performance, accessibility, best practices, SEO* dan yang terpenting adalah *progressive web application*.



Gambar 11. Benchmark skor lighthouse

Hasil dari pengujian menggunakan *lighthouse* pada Gambar 11 dapat dikatakan telah memenuhi semua kriteria dari *progressive web application* dengan nilai 100 dari 100. Aplikasi web juga mendapatkan skor yang sangat baik pada ketiga aspek seperti *Accessibility* dan *SEO* dengan skor 100 dari 100 serta *best practices* dengan skor 93 dari 100. Untuk aspek *performance* sendiri mendapatkan skor 52 dari 100 di karenakan banyak merender file *javascript* dari *framework AngularJS*.

Sementara untuk pengujian *progressive web application* memiliki 3 buah parameter utama yang dinilai oleh *tool lighthouse*. Tiga parameter tersebut yaitu *fast & reliable, installable* dan *PWA optimized* yang mempunyai 14 kriteria pengujian *progressive web application* pada aplikasi web yang disajikan pada Tabel 2.

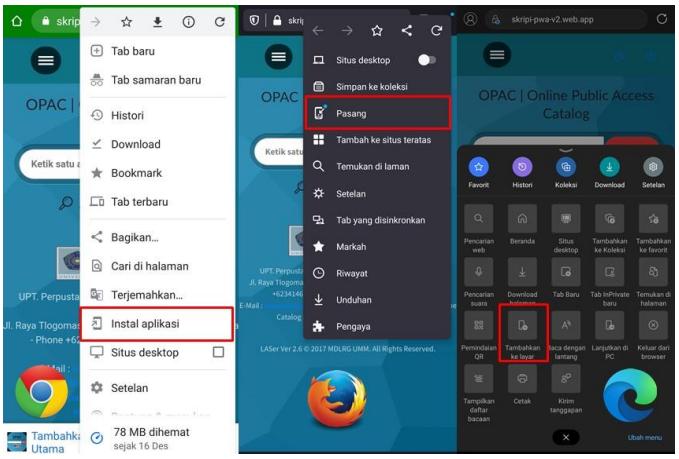
Tabel 2. Pengujian Baseline Checklist Progressive Web Application

No	Pengujian	Status
	Fast & Reliable	
1	Memuat halaman cukup cepat di jaringan seluler	*Passed*
2	Halaman saat ini merespons ketika *offline*	*Passed*
3	*start_url* merespons ketika *offline*	*Passed*
	Installable	
4	Menggunakan *HTTPS*	*Passed*
5	Mendaftarkan pekerja layanan yang mengontrol halaman dan *start_url*	*Passed*
6	*Manifest* aplikasi web memenuhi persyaratan pemasangan	*Passed*
	PWA Optimized	
7	Redirect lalu lintas *HTTP* ke *HTTPS*	*Passed*
8	Konfigurasikan untuk layar *splash* khusus	*Passed*
9	Menetapkan warna tema untuk bilah alamat.	*Passed*
10	Konten berukuran dengan benar untuk viewport	*Passed*
11	Memiliki *tag <meta name = "viewport">* dengan lebar atau skala awal	*Passed*
12	Berisi beberapa konten saat *JavaScript* tidak tersedia	*Passed*
13	Menyediakan ikon *apple-touch* yang valid	*Passed*
14	*Manifest* memiliki ikon *maskable*	*Passed*

Berdasarkan pengujian dari *progressive web application* dengan *tool lighthouse*, aplikasi web *online public access catalog* universitas muhammadiyah malang sudah memenuhi 14 kriteria pengujian. Seluruh kriteria dari *progressive web application* sudah terpenuhi sehingga dapat dipastikan nilai dari pengujian ini mendaatkan skor 100 dari 100.

3.2.2 Pengujian Manual

Pengujian manual pada *service worker* dengan menggunakan *smartphone mobile* berdasarkan dari *baseline checklist* untuk *progressive web application* mempunyai 3 kriteria. Ketiga kriteria tersebut disajikan pada Tabel 3 dan telah diuji dengan menggunakan perangat keras *smartphone mobile*.



Gambar 12. Pengujian service worker di browser Firefox, Chrome dan Microsoft edge

Dari hasil uji coba pertama dapat dilihat pada Gambar 12 bahwa *service worker* berjalan dengan baik dan dapat di*install* atau di tambahkan ke layar utama (*add to homescreen*) pada *browser Mozilla Firefox, Google Chrome* dan *Microsoft Edge* pada *smartphone mobile* tanpa menampilkan pesan *error*.

Hasil pengujian kedua adalah perpindahan halaman atau transisi sangat cepat tanpa memblokir jaringan. Hal itu dikarenakan pada *framework angular* memiliki teknologi *single page application (SPA)* dimana hanya satu halaman saja yang akan dimuat dan tidak me*reload* saat berpindah ke halaman berikutnya.

Pengujian terakhir ialah setiap perpindahan halaman tidak memiliki *URL* saat aplikasi web di*install* pada *smartphone mobile*. Hal itu karena salah satu syarat aplikasi web dikatakan *PWA* adalah seperti aplikasi *native* yang tidak memiliki *URL* pada *status bar smartphone mobile*.

Tabel 3. Pengujian manual service worker

No	Pengujian	Status
1	Situs berfungsi pada *browser* manapun (*Firefox*, *Chrome* dan *Edge*)	*Passed*
2	Transisi halaman cepat dan tidak memblokir jaringan	*Passed*
3	Setiap halaman memiliki *URL*	*Failed*

Hasil dari pengujian 3 kriteria tersebut adalah *service worker* dapat di*install* pada ketiga *browser* yaitu *mozilla firefox, google chrome* dan *microsoft edge* dengan status *passed*. Uji coba kedua adalah transisi atau perpindahan halaman sangat cepat dengan status *passed*. Namun hasil uji coba terakhir adalah setiap halaman memiliki alamat *URL* dengan *status failed* karena saat aplikasi web di*install* pada *smartphone mobile*, maka semua halaman tidak memiliki *URL* layaknya aplikasi *native*.

4. Kesimpulan

Berdasarkan hasil selama proses penerapan dan implementasi serta pengujian pada *progressive web application* dapat disimpulkan sebagai berikut:

a. Website *OPAC UMM* dengan penerapan teknologi *progressive web application* dapat menampilkan halaman secara *offline* tetapi saat menampilkan data dari *server* pengguna harus menjelajahi *URL* terlebih dahulu.
b. Application *Manifest* pada *PWA* berguna sebagai pembuatan aplikasi web yang dapat diinstall atau ditambahkan ke layar utama *smartphone mobile* layaknya aplikasi *native*.
c. Dapat diakses pada semua *browser* versi terbaru yang *support* dengan *service worker* seperti *Mozilla firefox, Google chrome* dan *Microsoft edge*.
d. Pembaruan dari sisi *server* akan secara otomatis tampil pada halaman pengguna menggunakan *realtime database* dari *firebase*.
e. Dengan fitur *indexedDB*, pengguna dapat menyimpan, mengedit dan menghapus data ke *server* secara *offline* dan akan dikirim saat kondisi jaringan *online* kembali.
f. Hasil pengujian menggunakan *tool lighthouse* memenuhi semua kriteria dari *progressive web application* dan mendapatkan skor diatas 90 hingga 100 namun dari asek *performance* masih dalam skor rata-rata karena banyak file *javascript* yang harus di *rendering*.

5. Saran

Saran untuk pengembangan dan perbaikan sistem dimasa mendatang diantaranya sebagai berikut.

a. Mendapatkan basis data dari *server* secara *offline* dengan mengimplementasikan *REST API* pada *firestore* di aplikasi *firebase*.
b. Dilakukannya pengamanan akses terhadapat data yang tersimpan secara *persisten* di *indexedDB* pada *browser* pengguna dengan menggunakan metode *enkripsi*.
c. Mengoptimisi skor kecepatan *performance* website dengan metode *critical rendering path* untuk mengatasi masalah *eliminate render blocking*.

Referensi

[1] T. T. Prabowo, “Perbandingan OPAC Perpustakaan Universitas Indonesia dan National University of Singapore Library,” no. October, 2017.
[2] B. R. Babu and A. O’Brien, “Web OPAC interfaces: An overview,” *Electron. Libr.*, vol. 18, no. 5, pp. 316–326, 2000, doi: 10.1108/02640470010354572.

[3] R. E. Eserada and S. E. Okolo, “Use of online public access catalogue (Opac) in selected university libraries in South- South Nigeria,” *Libr. Philos. Pract.*, vol. 2019, no. May, 2019.
[4] R. Kumar, “A Journey of Card Catalogue To Web-OPAC International Journal of Library and Information Studies,” vol. 4, no. 2, pp. 38–45, 2014.
[5] S. K. Gatot Subrata, “Penggunaan Otomasi Perpustakaan Laser Ver. 2.0,” *Society*.
[6] A. Kurniawan, I. S. Areni, and A. Achmad, “Implementasi Progressive Web Application pada Sistem Monitoring Keluhan Sampah Kota Makassar,” *J. Penelit. Enj.*, vol. 21, no. 2, pp. 34–38, 2018, doi: 10.25042/jpe.112017.05.
[7] V. Karpagam, “Performance Enhancement of Webpage Using Progressive Web App Features,” *Int. J. Innov. Res. Adv. Eng.*, vol. 03, no. 4, pp. 2349–2163, 2017.
[8] A. Aarsten, D. Brugali, and G. Menga, “Patterns for Three-Tier Client / Server Applications,” no. July, 2013.
[9] L. A. Sandy, R. J. Akbar, and R. R. Hariadi, “Rancang Bangun Aplikasi Chat pada Platform Android dengan Media Input Berupa Canvas dan Shareable Canvas untuk Bekerja dalam Satu Canvas Secara Online,” *J. Tek. ITS*, vol. 6, no. 2, 2017, doi: 10.12962/j23373539.v6i2.23782.
[10] N. Tutu, S. Lampah, and E. B. Setiawan, “Aplikasi Asesmen Anak Berkebutuhan Khusus di SLB Rafaha Arjasari Menggunakan Progressive Web App,” vol. X, no. 2, pp. 65–74, 2018.
[11] A. Aminudin, B. Basren, and I. Nuryasin, “Perancangan Sistem Repositori Tugas Akhir Menggunakan Progressive Web App (PWA),” *Techno.Com*, vol. 18, no. 2, pp. 154–165, 2019, doi: 10.33633/tc.v18i2.2309.