BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Tinjauan Pustaka

Dari tinjauan pustaka yang telah dilakukan penulis maka didapat beberapa penelitian sejenis yang pernah dilakukan oleh peneliti lainnya diantaranya adalah penelitian yang dilakukan oleh Kurniawan dkk (2017) dalam jurnalnya yang membahas tentang pengembangan aplikasi web menggunakan *progressive web apps* pada sistem *monitoring* keluhan sampah di kota Makassar. Penelitian ini menghasilkan data keluhan sampah yang di inputkan oleh *user* berupa foto keluhan, nama pelapor, tanggal laporan serta deskripsi keluhan dengan menggunakan sebuah API yang telah tersedia di *server*, kemudian data keluhan di tampilkan secara *offline* dengan menggunakan fitur *service worker* yang ada di *Progressive Web Apps.*

Penelitian yang dilakukan oleh Adi dkk (2017) dalam jurnalnya yang meneliti mengenai *Platform E-Learning* untuk pembelajaran pemrograman web menggunakan konsep *progressive web apps* yang dapat membantu proses pembelajaran di Teknik Informatika Institut Teknologi Sepuluh Nopember, Surabaya. Pada penelitian ini *platform E-Learning* mampu menampilkan halaman *E-Learning* yang dapat diakses oleh mahasiswa untuk melihat kelas yang diikuti, melihat daftar kuis, melihat daftar soal dari kuis, dan menjawab soal.

Penelitian yang dilakukan oleh Ridho dkk (2017) dalam jurnalnya mengenai perbandingan performa *progressive web apps* dan m*obile* web terkait waktu respon, penggunaan memori dan penggunaan media penyimpanan pada suatu halaman web, Universitas Brawijaya. Pada penelitian ini didapat beberapa

kesimpulan yaitu pada ukuran berkas dan *cache* yang kecil, *mobile* web masih lebih unggul dibandingkan dengan *Progressive Web Apps*, sedangkan pada ukuran berkas dan *cache* yang cukup besar *Progressive Web Apps* mampu mengungguli *mobile* web. Untuk performa terkait penggunaan memori, *mobile* web menggunakan memori lebih sedikit dibandingkan dengan *Progressive Web Apps* dikarenakan adanya proses tambahan pada *Progressive Web Apps* yaitu *service worker*. Sedangkan untuk performa terkait media penyimpanan, pada *mobile* web tidak menggunakan media penyimpanan sama sekali, sedangkan pada *progressive web apps* penggunaan media penyimpanan menyesuaikan dengan *cache* yang disimpan pada peramban.

Penelitian yang dilakukan oleh Pratama (2017) tentang pengembangan sistem informasi pemantauan lahan kelapa sawit dengan pendekatan progressive web apps. Pada penelitian ini peneliti mengintegrasikan sebuah API (Application Programming Interface) kedalam User Interface (UI), API tersebut belum dapat dimanfaatkan sebagai sistem informasi pemantauan lahan kelapa sawit yang dapat dipahami oleh para pegawai di perkebunan.

Ada juga penelitian yang dilakukan oleh Rizki (2018) tentang pengembangan aplikasi web menggunakan *progressive web apps* untuk menampilkan informasi lowongan pekerjaan pada studi kasus Akakom *Career Center.* Pada penelitian ini aplikasi web yang di bangun diperuntukkan bagi mahasiswa STMIK AKAKOM Yogyakarta yaitu prodi Teknik Informatika, Sistem Informasi, Manajemen Informatika, Komputerisasi Akuntansi, Teknik Komputer yang telah lulus, dengan memanfaatkan *service worker* untuk menyimpan *cache* kedalam website untuk dapat diakses ketika koneksi tidak stabil bahkan offline.

Perbedaan yang terdapat pada pustaka di atas dengan penelitian yang akan dibuat ini adalah penelitian ini akan membahas tentang pengembangan aplikasi web IMTA JOGJA menggunakan pendekatan *Progressive Web Apps* serta memanfaatkan *database* indexeddb pada database lokal sebagai basis data yang dapat melakukan pengolahan data pada aplikasi IMTA JOGJA. menampilkan informasi pengumuman terbaru dan berita terbaru mengenai organisasi tersebut.

Tabel 2.1 Penelitian yang Berhubungan dengan *Progressive Web Apps*

Penulis	Lokasi	Objek	Progressive Web Apps	Bahasa Pemrograman	Kriteria Lain
Kurniawan dkk (2017)	Universitas Islam Negeri Alaudin	Data keluhan masayarakat	Ya	*TypeScript*	-
Adi dkk (2017)	Institut Teknologi Sepuluh Nopember	Mahasiswa ITS jurusan Teknik Informatika	Ya	PHP *Framework* (Laravel)	-
Ridho dkk (2017)	-	Antarmuka, struktur berkas, alur pertukaran data dan manajemen *cache* pada aplikasi	Ya	-	Membandingkan performa *progressive web apps* dengan *mobile* web
Rizki (2018)	STMIK Akakom Yogyakarta	ACC (Akakom Career Center)	Ya	PHP	-
Pratama (2017)	Universitas Gadjah Mada	Lahan Kelapa Sawit	Ya	Javascript	
Hakim		IMTA JOGJA	Ya	JavaScript	Menggunakan database indexeddb sebagai pengolahan data

2.2. Dasar Teori

2.2.1. Progressive Web Apps

Progressive Web Apps adalah pengalaman yang menggabungkan pengalaman yang terbaik dari web dan pengalaman yang terbaik dari aplikasi. Pengalaman ini bermanfaat untuk pengguna dari kunjungan pertamanya di tab

browser, tanpa harus melakukan pemasangan seperti di *playstore* maupun di *app store*. Aplikasi dimuat dengan cepat, bahkan pada jaringan yang tidak stabil, mengirimkan pemberitahuan *push* yang relevan, memiliki ikon pada layar beranda, dan dimuat dengan pengalaman tingkat atas selayar penuh.(Lepage,2017)

- ***Progresif*** - Bekerja untuk setiap pengguna, apa pun pilihan *browser* mereka karena dibangun dengan peningkatan progresif sebagai konsep intinya.

- **Responsif** - Cocok dengan setiap faktor bentuk: perangkat *desktop*, seluler, tablet, atau apa saja yang muncul berikutnya.

- **Konektivitas *independen*** - Disempurnakan dengan *service worker* agar bisa bekerja *offline* atau pada jaringan berkualitas-rendah.

- **Seperti-Aplikasi** - Terasa seperti sebuah aplikasi untuk pengguna dengan interaksi dan navigasi bergaya-aplikasi karena dibangun di atas model *shell* aplikasi.

- **Segar** - Selalu terkini berkat proses pembaruan *service worker*.

- **Aman** - Disediakan melalui *HTTPS* untuk mencegah *snooping* dan memastikan materi belum dirusak.

- **Dapat ditemukan** - Dapat diidentifikasi sebagai "aplikasi" berkat manifes W3C dan cakupan registrasi *service worker*, yang memungkinkan mesin telusur untuk menemukannya.

- **Bisa dilibatkan-kembali** - Kemudahan untuk dilibatkan-kembali dengan fitur seperti pemberitahuan *push*.

- **Dapat dipasang** - Memungkinkan pengguna untuk "menyimpan" aplikasi yang mereka anggap paling berguna di layar beranda tanpa kerumitan seperti pada toko aplikasi.

- **Bisa ditautkan** - Dapat dengan mudah dibagikan melalui *URL*, tidak memerlukan pemasangan yang rumit.

2.2.2. Service Worker

Service Worker adalah skrip yang berjalan di latar belakang browser pengguna, yang tidak memerlukan halaman web atau interaksi dari pengguna. Service worker pada dasarnya adalah berkas JavaScript yang berjalan pada thread yang berbeda dengan main thread browser, menangani *network request*, *caching*, dan mengembalikan *resource* dari *cache*, dan bisa mengirim *push message*. Service worker bekerja sebagai pengatur *event fetch* dari browser, lalu *service worker* memutuskan apakah *request* akan diteruskan ke *server* atau ke *cache* berdasarkan kondisi jaringan *online* atau *offline*.(Gaunt,2017)

2.2.3. JavaScript

JavaScript adalah bahasa paling populer di web dan ekosistemnya merupakan *open source*. http://githut.info/ memetakan jumlah repositori aktif dan secara keseluruhan popularitas bahasa program di GitHub selama beberapa tahun terakhir adalah bahasa pemrograman JavaScript. Meskipun web browser adalah *platform* yang paling banyak digunakan untuk JavaScript, *database modern*

seperti MongoDB dan CouchDB menggunakan JavaScript sebagai *scripting* dan bahasa kueri. JavaScript telah menjadi *platform* yang penting di luar browser. (Antani. 2016)

JavaScript adalah bahasa *script* yang ringan dan mudah digunakan. JavaScript dapat membuat halaman web tidak sekedar menjadi halaman data dan informasi saja, tetapi juga dapat menjadi suatu program aplikasi dengan antarmuka web.

2.2.4. HTTPS

Hypertext Transfer Protocol Secure memiliki pengertian yang sama dengan http hanya saja https memiliki kelebihan fungsi di bidang keamanan (*secure*). Dengan menggunakan *Secure Socket Layer* (SSL) atau *Transport Layer Security* (TLS) sebagai sublayer di bawah http aplikasi *layer* yang biasa. Teknologi https protokol mencegah kemungkinan “dicurinya” informasi penting yang dikirimkan selama proses komunikasi berlangsung antara *user* dengan web *server* atau sebaliknya.

Secara teknis, website yang menggunakan https akan melakukan enkripsi terhadap informasi (data) menggunakan teknik enkripsi SSL. Dengan cara ini meskipun seseorang berhasil “mencuri” data tersebut selama dalam perjalanan *user* web *server*, orang tersebut tidak akan bisa membacanya karena sudah diubah oleh teknik enkripsi SSL. Umumnya website yang menggunakan https ini adalah website yang memiliki tingkat kerawanan tinggi yang berhubungan dengan masalah keuangan dan privasi dari pelanggannya seperti website perbankan dan investasi.

HTTPS dienkripsi dan deskripsi dari halaman yang di minta oleh pengguna dan halaman yang di kembalikan oleh web *server*. Kedua protokol tersebut memberikan perlindungan yang memadai dari serangan *eavesdroppers*, dan *man in the middle attacks*.

HTTPS menyediakan tiga jaminan keamanan:

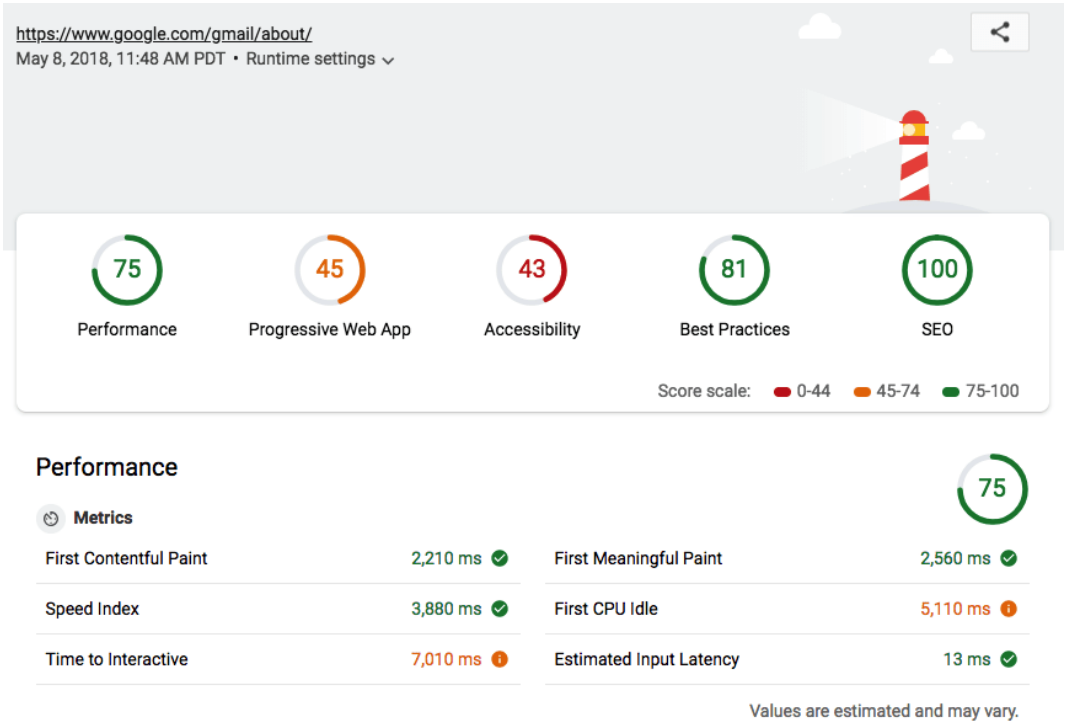
1. Otentikasi *server* memungkinkan pengguna untuk memiliki keyakinan bahwa mereka berbicara dengan *server* aplikasi yang benar. Tanpa jaminan ini, tidak ada jaminan kerahasiaan atau integritas.
2. Kerahasiaan data berarti penyadap tidak dapat memahami isi komunikasi antara browser pengguna dan *server* web, karena data dienkripsi.
3. Integritas data berarti penyerang jaringan tidak dapat merusak atau mengubah konten komunikasi antara browser pengguna dan *server* web, karena mereka divalidasi dengan kode otentikasi pesan kriptografi.

HTTP tidak memberikan jaminan keamanan, dan aplikasi yang menggunakannya tidak mungkin memberikan keamanan kepada pengguna. Saat menggunakan aplikasi web yang dihosting melalui HTTP, pengguna tidak memiliki cara untuk mengetahui apakah mereka berbicara dengan server aplikasi yang sebenarnya, atau mereka tidak dapat memastikan bahwa penyerang belum membaca atau memodifikasi komunikasi antara komputer pengguna dan server. (Palmer, 2017)

2.2.5. Lighthouse

Menurut Google, dilansir dari halaman https://developers.google.com/web/tools/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*.



Gambar 2.1 Contoh Penilaian pada *tools* lighthouse

Gambar 2.1 merupakan contoh penilaian pada tools lighthouse, ada beberapa kriteria diantaranya *performance*, progressive web apps, *accessibility*, *best practices* dan SEO.