BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Gita Perdani (2019) membangun sebuah sistem informasi desa berbasis *website* yang menampilkan informasi berita dan agenda desa. Pembuatan sistem ini menerapkan teknologi *Progressive Web Apps* (PWA) dan *service worker* yang membuat sistem dapat menyimpan data masukan admin saat *offline* sehingga saat sistem sudah terhubung dengan jaringan internet atau *online* data secara otomatis terkirim menuju *database MySQL*.

Deviana Wulandari (2019) membuat aplikasi pelacakan pengiriman barang pada PT. Oriflame berbasis *website* yang dapat menampilkan informasi pelacakan berupa teks dan grafis kepada konsultan dan mengirim laporan kepada pihak Oriflame dengan menerapkan *Progressive Web Apps* (PWA) dan teknologi *service worker* agar aplikasi dapat diakses secara *offline*.

Nur Najmi Wicaksana (2019) membangun sebuah *website* aplikasi *monitoring service* laptop Bengkel OS dengan penerapan *Progressive Web Apps* dan ditambah bantuan teknologi *service worker* yang membuat pelanggan dapat melihat informasi pengerjaan laptop secara *offline* dan mendapatkan *push notification* ketika terdapat informasi baru.

Riyanto Iqbal Firdaus (2016) merancang sistem media informasi berbasis *website* yang akan dikembangkan sehingga masyarakat dapat mengakses informasi pemerintah desa secara *online*. Informasi yang ditampilkan meliputi berita-berita,

agenda kegiatan pemerintah desa, potensi desa, file-file yang dapat didownload, galeri foto, dan kontak dari pemerintahan desa yang dapat dihubungi. Bagi pemerintah desa, sistem ini akan meningkatkan pelayanan dari perangkat desa kepada penduduk desa terkait.

Hery Pujiwiyanto (2019) membuat aplikasi sistem informasi Desa Kerjo Kidul berbasis *website* dengan menggunakan *framework Codeigniter* dan Bootstrap yang menampilkan informasi profil desa, data penduduk, agenda kegiatan, dokumentasi dan kontak perangkat desa.

Tabel 2.1 Perbandingan Tinjauan Pustaka

Penulis	Objek	Teknologi	Hasil
Gita Perdani (2019)	Desa Banguntapan	PWA	Website informasi tentang kegiatan pada kelurahan yang bisa diakses pada saat ofline atau dalam kondisi jaringan tidak begitu bagus
Deviana Wulandari (2019)	PT. Oriflame	PWA	Aplikasi yang dapat melakukan pelacakan pengiriman barang dan laporan pengiriman sudah melalui web
Nur Najmi Wicaksana (2019)	Bengkel OS	PWA	Sistem informasi monitoring service laptop secara user friendly dan informasi pengerjaan laptop secara offline
Hery Pujiwiyanto (2019)	Desa Kerjo Kidul	Framework Codeigniter dan Bootstrap	Informasi Kependudukan dan agenda kegiatan desa

Riyanto Iqbal Firdaus (2016)	Desa di Kecamatan Kedawung Sragen	Website	Menampilkan informasi-informasi dari pemerintah desa yang dapat dihubungkan dengan beberapa fitur tambahan
Usulan	Desa Wates	PWA	Informasi mengenai kegiatan dan kepengurusan kependudukan berupa website yang dapat diakses secara offline atau dalam koneksi yang lemah dan dapat diinstall

2.2 Dasar Teori

2.2.1 Progressive Web Apps

Teknologi aplikasi web saat ini sudah banyak mengalami perubahan fungsi. Awal mula perkembangan teknologi web dimulai dari web 1.0 yang diperkenalkan tahun 1990 dimana masih bersifat statis hingga menjadi aplikasi web yang dapat menangani masalah pengecekan status baterai, penggunaan mode *offline*, hingga *speech recognition*. Salah satu teknologi yang tengah banyak diperhatikan saat ini adalah teknologi *Progressive Web Apps* (PWA).

Progressive Web Apps (PWA) adalah isitilah aplikasi berbasis web yang memanfaatkan fitur perambanan modern agar tampil seperti aplikasi *native*. PWA digambarkan sebagai penengah antara aplikasi *native* dan *hybrid* dengan berbagai rekomendasi yang tidak spesifik pada desain aplikasi web untuk perangkat mobile, seperti preferensi *HTTPS* melalui *HTTP* dan desain yang *responsive*.

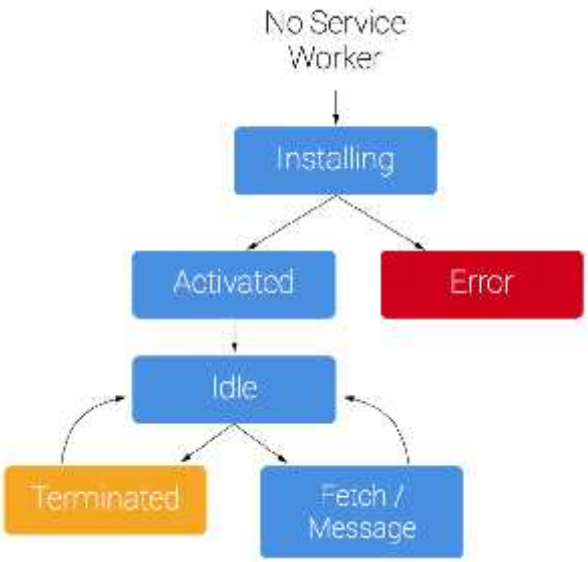
Pada dasarnya teknologi *Progressive Web Apps* (PWA) bekerja layaknya aplikasi berbasis web pada biasanya, yang membedakan PWA dengan aplikasi *website* lainnya adalah PWA yang bekerja dengan konektivitas yang independen. Artinya, aplikasi *Progressive Web Apps* dapat bekerja pada jaringan yang lemah atau dalam keadaan *offline* dengan adanya *service worker*. Dan selain mampu berjalan dalam keadaan *offline Progressive Web Apps* juga menggunakan teknologi *instant loading* yang membuat aplikasi website tersebut berjalan dengan cepat, *screenhome* dimana aplikasi *website* tersebut dapat dijadikan *icon* pada *desktop* atau *homescreen*.

2.2.2 Service Worker

Salah satu Konsep yang dibangun oleh PWA adalah *Service Worker*. *Service worker* adalah *script* yang berjalan di belakang *browser* pengguna. *Service worker* tidak membutuhkan sebuah halaman ataupun interaksi dari pengguna untuk menjalankan tugasnya, dengan begitu *service worker* akan terus berjalan walaupun halaman web tidak terbuka.

Service Worker adalah *script* 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 ditersukan ke *server* atau ke *cache* berdasarkan kondisi jaringan *online* atau *offline* (Gaunt, 2019).



Gambar 2.1 Siklus Service Worker

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

1. Daftarkan *service worker* melalui URL (*Uniform Resource Locator*) fungsi *serviceWorkerContainer.register*().
2. Jika berhasil, *service worker* dijalankan di *ServiceWorkerGlobalScope*.
3. *Service worker* telah siap untuk memproses *event*.
4. Instalasi *service worker* dicoba ketika *service worker* mengontrol halaman yang diakses setelah dan sebelumnya. *Event install* akan selalu dikirim pertama kali ke *service worker*.
5. Ketika *handler oninstall* selesai, *service worker* akan dipasang.

6. Proses aktivasi. Ketika *service worker* terpasang, selanjutnya akan menerima *event activate*. Pengguna utama dari *onactivate* ini adalah untuk membersihkan sumber daya yang digunakan sebelumnya.
7. *Service control* sekarang dapat mengatur 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.

2.2.3 IndexedDB

IndexedDB adalah sistem basis data transaksional, seperti RDBMS berbasis SQL. Namun, tidak seperti RDBMS berbasis SQL yang menggunakan tabel kolom tetap, *IndexedDB* adalah *database* berorientasi objek berbasis *JavaScript*. IndexedDB memungkinkan penyimpanan dan pengambilan objek yang diindeks dengan *key*, objek apa pun yang didukung oleh algoritma klon terstruktur dapat disimpan. Yang diperlukan ialah menentukan skema *database*, membuka koneksi ke *database*, dan kemudian mengambil dan memperbarui data dalam serangkaian transaksi. *IndexedDB* adalah cara untuk terus menyimpan data di dalam *browser* pengguna. Karena memungkinkan untuk membuat aplikasi web dengan kemampuan *query* yang kaya terlepas dari ketersediaan jaringan, aplikasi ini dapat berfungsi baik *online* maupun *offline*. *IndexedDB* sendiri memiliki kapasitas yang berbeda untuk setiap *browser*nya, diantaranya yaitu pada *Chrome* menyediakan sebanyak 6% dari *free space memory*, *Mozilla Firefox* menyediakan sebanyak 10% dari *free space memory* dan pada *Internet Explorer* sebanyak 250 *megabyte*.

IndexedDB berguna untuk aplikasi yang menyimpan sejumlah besar data (misalnya, katalog DVD di perpustakaan peminjaman) dan aplikasi yang tidak memerlukan konektivitas internet terus-menerus untuk bekerja (misalnya, *client email*, daftar tugas, dan buku catatan) (Walton, 2019).

2.2.4 Web App Manifest

Web Apps Manifest adalah file *JSON* sederhana yang memberitahu browser tentang aplikasi web dan bagaimana aplikasi itu dipasang pada perangkat *mobile* atau *desktop*. Diperlukan manifes oleh *Chrome* untuk menampilkan permintaan Add To Home Screen (A2HS). File manifes mencakup informasi tentang nama aplikasi, ikon yang harus digunakan, *start url* dimana manifes harus dimulai saat diluncurkan, dan banyak lagi (Gaunt & Kinlan 2019).

Saat pengguna menambahkan *website* ke layar beranda, sistem dapat menentukan serangkaian ikon untuk digunakan *browser*. Ikon-ikon ini digunakan di tempat-tempat seperti layar beranda, layar pembuka, dll. Ikon adalah *array* dari objek gambar. Setiap objek harus menyertakan *src*, *sizes property*, dan tipe gambar.