Modul 4: Pengembangan Web Client & Offline Capability (PWA)

Mata Kuliah: Rekayasa Perangkat Lunak Lintas Platform / Integrasi Sistem

Tujuan:

1. Mahasiswa mampu membangun aplikasi web yang responsif menggunakan pola *Progressive Web Application* (PWA).
2. Mahasiswa mampu mengimplementasikan fitur PWA untuk mendukung akses *offline* melalui *Service Workers*.
3. Mahasiswa mampu menerapkan strategi *caching* untuk meningkatkan performa aplikasi di jaringan yang tidak stabil.

1. Pendahuluan

Pada era lintas platform, aplikasi web harus berperilaku seperti aplikasi *native* untuk menangani variasi kualitas jaringan. **Progressive Web App (PWA)** hadir sebagai solusi agar aplikasi tetap responsif meskipun koneksi internet terputus. Komponen kunci dari teknologi ini adalah **Service Worker**, sebuah skrip latar belakang yang bertindak sebagai proksi untuk mengelola aset dalam memori lokal (*cache*).

2. Persiapan Lingkungan Kerja

- **Runtime & IDE:** Node.js dan VS Code.
- **Framework:** Vite + React.js (melanjutkan proyek dari Modul 3).
- **Library Utama:** vite-plugin-pwa untuk otomatisasi konfigurasi.
- **Browser:** Google Chrome (memanfaatkan fitur **DevTools Application**).

3. Implementasi Kode Secara Rinci

A. Instalasi & Konfigurasi PWA Plugin

Jalankan perintah berikut di terminal proyek Anda (folder proyek RLP):

Bash

```
npm install vite-plugin-pwa -D
```

Konfigurasikan vite.config.js untuk mengaktifkan fitur manifest dan strategi pembaruan otomatis:

JavaScript

```
// vite.config.js
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'
import { VitePWA } from 'vite-plugin-pwa'

export default defineConfig({
  plugins: [
    react(),
    VitePWA({
          registerType: 'autoUpdate', // Memperbarui aplikasi secara otomatis di latar belakang
    // Mengatur agar semua aset (html, js, css) masuk ke cache
      workbox: {
        globPatterns: ['**/*.{js,css,html,ico,png,svg}']
      },
      manifest: {
        name: 'Sistem Terintegrasi Lintas Platform',
        short_name: 'SysIntegrate',
        theme_color: '#1A73E8', // Sesuai Design System Modul 3
        icons: [
          { src: 'pwa-192x192.png', sizes: '192x192', type: 'image/png' },
          { src: 'pwa-512x512.png', sizes: '512x512', type: 'image/png' }
        ]
      }
    })
  ]
})
```

Tambahkan Meta Tag di **index.html**

- Pastikan di dalam tag `<head>` pada file `index.html` terdapat script untuk mendaftarkan Service Worker:

HTML

```
<head>
```

```
  <!-- ... tag lainnya ... -->
  <script>
    if ('serviceWorker' in navigator) {
      window.addEventListener('load', () => {
        navigator.serviceWorker.register('/sw.js');
      });
    }
  </script>
</head>
```

B. Implementasi Pendeteksi Status Jaringan (UX)

Gunakan *hook* useEffect untuk mendeteksi status koneksi pengguna secara *real-time*:

src/App.jsx

JavaScript

```
// src/App.jsx
// import './App.css'
import './index.css'

import StatusCard from './components/StatusCard';
import { useState, useEffect } from 'react'; // Pendeteksi Status Jaringan

function App() {
  // Pendeteksi Status Jaringan
  const [isOnline, setIsOnline] = useState(navigator.onLine);
  // -------------------------

  // Data simulasi dari BFF NestJS Modul 2
  const sensorList = [
    { id: 'S1', name: 'Ambient Temp', val: 24.5, unit: '°C', state: 'OK' },
    { id: 'S2', name: 'Pressure', val: 1012, unit: 'hPa', state: 'DANGER' },
    { id: 'S3', name: 'Core Heat', val: 88.2, unit: '°C', state: 'WARNING' },
    { id: 'S4', name: 'Battery Level', val: 12, unit: '%', state: 'WARNING' },
  ];

  // Pendeteksi Status Jaringan
  useEffect(() => {
    window.addEventListener('online', () => setIsOnline(true));
    window.addEventListener('offline', () => setIsOnline(false));
```

```
  }, []);
  // ------------------------

  return (
    // Menggunakan bg-tech-bg untuk konsistensi warna latar dari config
    <div className="min-h-screen bg-tech-bg p-safe-area font-sans text-slate-900">
      {/* Kode Pendeteksi Status Jaringan */}
      {/* Notifikasi Offline */}
      {!isOnline && (
          <div className="fixed top-0 left-0 right-0 bg-red-600 text-white p-2 text-center
z-50 animate-bounce">
          Anda sedang offline. Menampilkan data dari cache lokal.
        </div>
      )}
      {/* ------------------------------ */}

      {/* Header Section */}
      <header className="max-w-5xl mx-auto mb-8">
        <div className="flex items-center gap-3">
          {/* Indikator Live Status */}
          <div className="relative flex h-3 w-3">
              <span className="animate-ping absolute inline-flex h-full w-full rounded-full
bg-green-400 opacity-75"></span>
                          <span  className="relative  inline-flex  rounded-full  h-3  w-3
bg-green-500"></span>
          </div>

          <div>
            <h1 className="text-xl font-black uppercase tracking-tighter">
              SensorNodes <span className="text-tech-primary">Controller</span>
            </h1>
                    <p  className="text-slate-500  text-[10px]  font-bold  tracking-widest
uppercase">
              Real-time Monitoring Active
            </p>
          </div>
        </div>
      </header>

      {/* Grid Dashboard */}
```

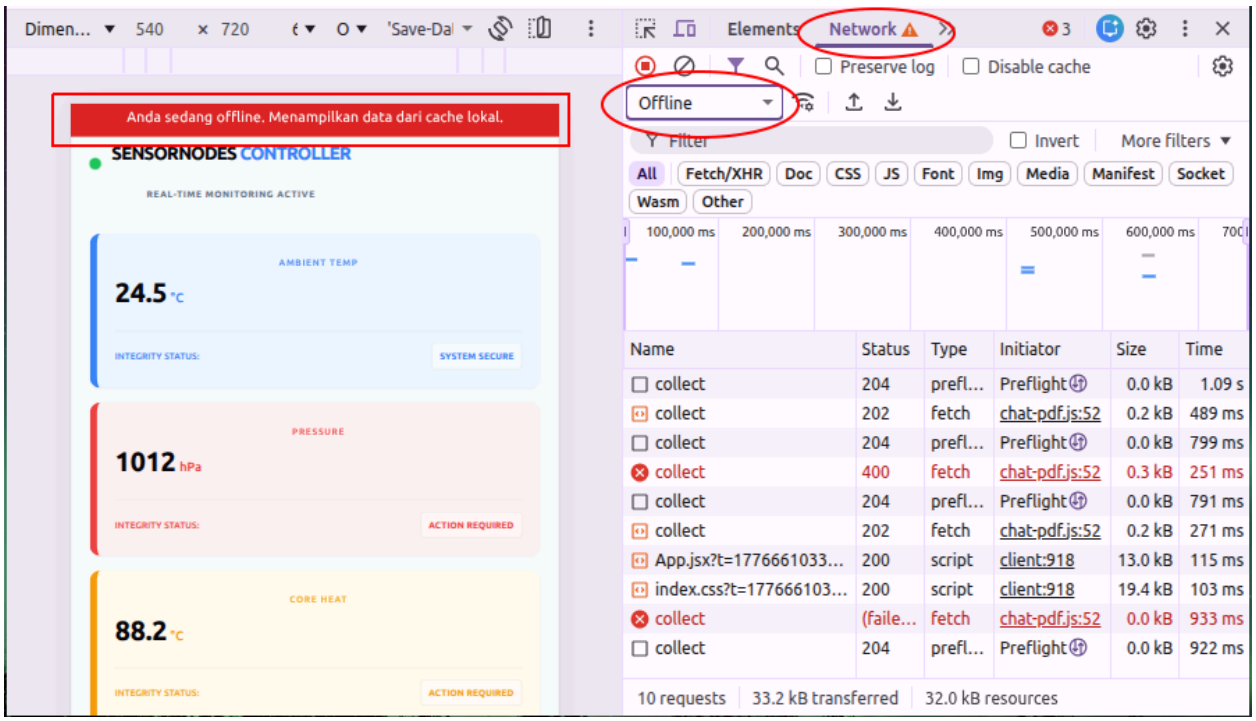
```
            <main className="max-w-5xl mx-auto grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4">
        {sensorList.map(sensor => (
          <StatusCard
            key={sensor.id}
            label={sensor.name}
            value={sensor.val}
            unit={sensor.unit}
            status={sensor.state}
            // isCompact diatur false agar detail "Integrity Status" terlihat di layar besar
            isCompact={false}
          />
        ))}
      </main>

      {/* Footer / Status Bar untuk Mobile */}
      <footer className="max-w-5xl mx-auto mt-12 opacity-50 text-center"><br/>
        <p className="text-[10px] font-medium italic">
          BFF Node.js Connected • System Version 2.0.4
        </p>
      </footer>
    </div>
  );
}

export default App;
```

C. Pengujian status koneksi OFFLINE

Untuk menguji fitur **offline detector** pada aplikasi React Anda tanpa harus benar-benar mematikan koneksi internet komputer, Anda bisa menggunakan dua cara berikut:

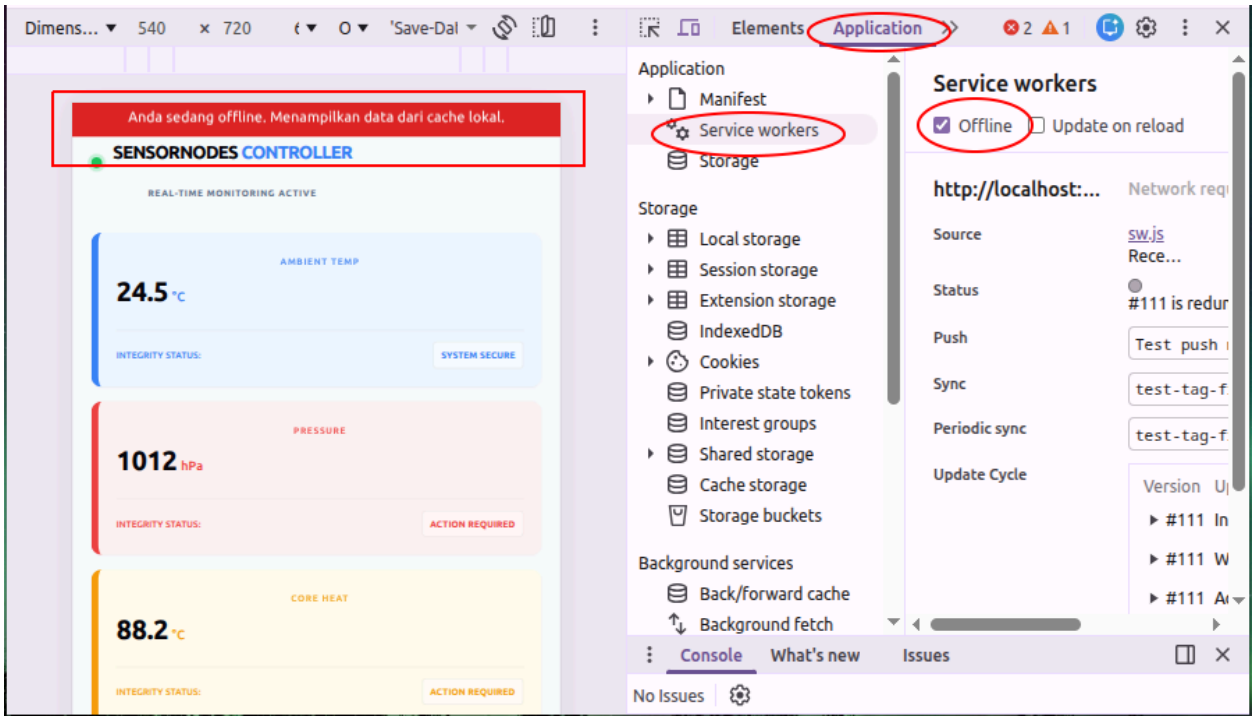
1. Jalankan `npm run build`.
2. Jalankan `npm run preview`.
3. Buka aplikasi di browser (biasanya `localhost:4173`)

- Melalui fitur **Network Throttling** di Browser **Chrome/Edge DevTools.**
    a. **Buka Inspeksi Elemen:** Klik kanan di mana saja pada aplikasi Anda dan pilih Inspect (atau tekan `F12` / `Cmd+Option+I`).
    b. **Pilih Tab Network:** Cari tab bernama Network di bagian atas panel DevTools.
    c. Ubah Status **Throttling**:
        i. Cari drop-down yang secara default bertuliskan **No Throttling** (biasanya di barisan icon filter).
        ii. Klik drop-down tersebut dan pilih **Offline**.
    d. **Lihat Hasilnya:** Aplikasi akan mendeteksi perubahan status secara instan dan menampilkan pesan: *"Anda sedang offline. Menampilkan data dari cache lokal."*
    e. **Refresh:** Setelah halaman di refresh, aplikasi harusnya tetap muncul.



- Melalui fitur **Service Worker** di Browser **Chrome/Edge DevTools**.

a. **Buka Inspeksi Elemen:** Klik kanan di mana saja pada aplikasi Anda dan pilih Inspect (atau tekan `F12` / `Cmd+Option+I`).
b. **Pilih Tab Application:** Cari tab bernama Application di bagian atas panel DevTools.
c. Ubah Status **Service workers**: beri centang pada opsi **Offline**
d. **Lihat Hasilnya:** Aplikasi akan mendeteksi perubahan status secara instan dan menampilkan pesan: *"Anda sedang offline. Menampilkan data dari cache lokal."*
e. **Refresh:** Setelah halaman di refresh, aplikasi harusnya tetap muncul.



D. Menjalankan Audit Lighthouse

Audit Lighthouse adalah alat otomatis yang dibuat oleh Google untuk membantu pengembang meningkatkan kualitas halaman web. Menjalankan audit untuk kategori **PWA (Progressive Web App)** bertujuan untuk memastikan aplikasi Anda memenuhi standar minimum aplikasi web yang bisa diinstal dan bekerja secara offline.

Berikut adalah langkah-langkah detail untuk menjalankannya di browser Google Chrome:

1. Buka Aplikasi dalam Mode Produksi
    a. Lighthouse memberikan hasil terbaik pada kode yang sudah dioptimasi.
    b. Jalankan perintah: `npm run build` diikuti `npm run preview`.
    c. Buka alamat URL yang diberikan (biasanya `http://localhost:4173`).
2. Gunakan Mode Incognito
    a. Buka jendela Incognito baru.
    b. Copy-paste URL aplikasi Anda ke sana, agar ekstensi browser lain tidak mengganggu hasil penilaian performa.
3. Buka Chrome DevTools
    a. Klik kanan di halaman aplikasi dan pilih Inspect, atau tekan `F12`.
4. Pilih Tab "Lighthouse"
    a. Di barisan tab atas (Elements, Console, Network), cari tab Lighthouse. Jika tidak terlihat, klik ikon panah ganda `>>`.
5. Konfigurasi Audit PWA
    a. Mode: Pilih ***Navigation*** *(Default)*.
    b. Device: Pilih ***Mobile*** (standar umum PWA) atau *Desktop*.
    c. Categories: Centang hanya pada bagian **Progressive Web App**. (Anda bisa mencentang yang lain jika ingin melihat skor performa secara umum).
6. Klik "Analyze page load"
    a. Tunggu beberapa saat. Lighthouse akan me-refresh halaman Anda secara otomatis dan melakukan serangkaian tes (seperti mematikan koneksi secara virtual).

4. Penjelasan Logika & Strategi Caching

- **Web App Manifest:** Memberikan instruksi agar web dapat diinstal di *homescreen* tanpa *address bar*, memberikan pengalaman seperti aplikasi *native*.
- **Network Detection:** Menggunakan navigator.onLine untuk menyesuaikan perilaku aplikasi, seperti menonaktifkan fitur sinkronisasi saat internet mati.

- **Offline Cache:** Aset utama (HTML, CSS, JS) disimpan di **Cache Storage** pada kunjungan pertama, sehingga pada akses berikutnya browser memprioritaskan data lokal untuk kecepatan maksimal.

5. Tugas Praktikum: "Optimasi Dashboard Area Minim Sinyal"

Skenario: Anda diminta mengoptimalkan dasbor agar tetap berfungsi bagi teknisi di area dengan sinyal tidak stabil.

Instruksi:

1. **Konfigurasi PWA:** Terapkan vite-plugin-pwa dan sesuaikan warna tema sesuai *Design System* Modul 3.
2. **Uji Coba Offline:**
    - Jalankan npm run build diikuti npm run preview.
    - Di Chrome DevTools, buka tab **Application** > **Service Workers** dan centang **Offline**.
    - *Refresh* halaman untuk memastikan aplikasi tetap tampil.
3. **Audit Performa:** Jalankan audit **Lighthouse** untuk kategori "Progressive Web App".

Hasil yang Harus Dikumpulkan:

- Tangkapan layar aplikasi yang berjalan sukses dalam mode *offline*.
- Laporan skor PWA dari Lighthouse (skor minimal harus berwarna hijau).