

Jurnal Skripsi

SISTEM MONITORING DAYA LISTRIK DENGAN NODEMCU BERBASIS PROGRESSIVE WEB APPLICATION

¹M. Syahrul Fadli, ²F. Yudi Limpraptono, ³Sotyohadi
Institut Teknologi Nasional, Malang, Indonesia

¹callsyahrul@gmail.com, ²fyudil@lecturer.itn.ac.id, ³sotyohadi@lecturer.itn.ac.id³

Abstrak—Sistem monitoring listrik sangat mempengaruhi sikap individu dalam melakukan penghematan listrik. Dimana informasi yang didapat dari kegiatan monitoring listrik dapat menjadi faktor eksternal yang mempengaruhi sikap individu dalam penghematan listrik. Sistem monitoring daya listrik dengan berbasis Progressive Web Application (PWA) akan memberikan fitur pengalaman akses monitoring website selayaknya aplikasi native pada perangkat sistem operasi. Diketahui dari sebuah survey pada 2015 bahwa pengguna mobile lebih populer dengan aplikasi native dibanding dengan mengakses website. Hal ini mendukung kebergunaan sistem monitoring berbasis PWA untuk lebih mudah digunakan oleh pengguna karena tingkah laku website berbasis PWA dapat menyerupai aplikasi native. Beberapa fitur PWA yang dapat menyerupai aplikasi native seperti, dapat diakses tanpa jaringan, installable, dapat berjalan pada background system, dan melakukan push notification pada perangkat. Dengan fitur cache sebuah website dapat meningkatkan performa kecepatan akses website dengan berbasis PWA serta website yang penerapan fitur installable dapat mengurangi penggunaan sumber daya RAM dan CPU ketika diakses langsung melalui shortcut dan fitur installable.

Kata Kunci: Sistem Monitoring, NodeMCU, Progressive Web Application (PWA), PZEM-004T.

I. PENDAHULUAN

Sistem monitoring daya listrik akan mempengaruhi sikap bijak individu dalam menggunakan atau memanfaatkan peralatan listrik. Sehingga perlu adanya cara untuk memberikan informasi mengenai penggunaan daya listrik sehingga dapat mempengaruhi sikap individu dalam penggunaan konsumsi listrik karena akses informasi merupakan faktor eksternal yang dapat mempengaruhi sikap penghematan listrik [1].

Pada penelitian terkait dengan sistem monitoring daya listrik dengan judul “Monitoring Biaya Dan Pengukuran Konsumsi Daya Listrik Berbasis Arduino Mega2560 Menggunakan WEB” peneliti melakukan uji coba membuat website (tanpa PWA) yang ditempatkan pada layanan hosting dengan metode menguji akurasi sensor-sensor [2]. Hasil penelitian dari pengujian website tersebut hanya ditampilkan pada versi desktop (PC) namun tidak membahas performa website baik pada PC ataupun mobile juga fungsionalitas website ketika diakses pada perangkat mobile. Sedangkan pada penelitian terkait dengan judul “Implementasi Sistem Monitoring Daya Listrik Berbasis Web dan Protokol

Komunikasi Websocket” penelitian membahas tentang penerapan protokol Websocket antara komunikasi NodeMCU dengan web server [3]. Pada penelitian ini website dilakukan uji coba fungsionalitasnya pada perangkat PC maupun mobile (perangkat android) namun tidak membahas performa akses website PC dan mobile. Penelitian ini juga dilakukan pada jaringan lokal sehingga performa page speed atau YSlow score website tidak dapat diuji. Kedua penelitian yang telah disebutkan di atas berbasis website native dan menggunakan library-library Javascript untuk menampilkan data monitoring dalam bentuk chart (Grafik). Penambahan library-library Javascript merupakan salah satu metrik yang mempengaruhi performa website pada waktu muat (loading time) [4].

Web native (website tanpa PWA) dalam sisi performa memiliki permasalahan klasik berupa waktu respon yang lambat seiring banyaknya aset yang akan dimuat hal ini akan berimbas pada kuantitas dan frekuensi penggunaan aplikasi tersebut [5]. Berdasarkan hasil survei yang dilakukan Gfk Indonesia pada 2015 [6] diketahui pengguna mobile lebih populer dengan aplikasi native yang terpasang pada perangkat dibanding dengan mengakses website melalui browser. Pada penelitian perbandingan fitur aplikasi native vs. PWA vs. web standar diketahui aplikasi berbasis web standar/native hanya dapat diakses saat berada di dalam jaringan sedangkan aplikasi native dapat diakses di luar dan di dalam jaringan [7].

Hasil penelitian terkait perbandingan *usability* performa antara web native dengan web PWA diketahui bahwa penerapan PWA pada website dapat meningkatkan performa page speed dan YSlow Score yang lebih baik dibanding website native [8]. Penelitian pengujian PWA dengan website native yang dilakukan oleh [5] juga menghasilkan waktu muat yang lebih baik ketika website diakses berulang kali. Penerapan PWA memungkinkan website untuk menghadirkan pengalaman seperti aplikasi *native* pada perangkat mobile [9]. Penelitian lainnya juga menyebutkan pemuatan aset statis stylesheet (CSS) lebih cepat jika dibanding halaman web tanpa PWA [7]. Dengan teknologi PWA website juga dapat diakses saat berada di luar jaringan karena saat berada di luar jaringan website akan menggunakan cache halaman yang pernah dimuat sebelumnya. Metode penyimpanan cache untuk ditampilkan

pada kondisi *offline* disebut dengan strategi *cache then network*. Website dengan teknologi PWA dapat diakses selayaknya aplikasi *native* pada perangkat mobile ataupun desktop. Hal ini salah satunya dimungkinkan karena dapat website dapat di-install seperti aplikasi *native*. Dengan demikian aplikasi sistem monitoring listrik berbasis website dapat bekerja dalam kondisi *offline* dan berperilaku selayaknya aplikasi *native*.

II TINJAUAN PUSTAKA

A. Progressive Web Application (PWA)

Progressive Web Application merupakan konsep teknologi pada website yang memungkinkan sebuah website memiliki fitur seperti aplikasi *mobile* [10]. Pada dasarnya website dengan teknologi PWA berjalan pada perangkat lunak browser dari klien dimana PWA memanfaatkan fitur teknologi dari browser tersebut untuk dapat berjalan dan mendukung PWA. Fitur-fitur dari PWA adalah sebagai berikut:

- Dapat bekerja walaupun *offline*.
- Dapat “ter-install” pada sistem operasi.
- Dapat melakukan *push notifications* pada perangkat.
- Dapat berjalan di *background* sistem.
- Dapat melakukan sinkronisasi data pada *background* sistem.
- Dan dapat mengakses fitur perangkat keras dari perangkat.

Perangkat lunak *browser* (peramban) populer, seperti Google Chrome, Firefox, Safari, dan berbagai peramban dengan basis Chromium sudah mendukung teknologi dari *Progressive Web Application*. Untuk menerapkan PWA pada website dibutuhkan dua komponen utama, yaitu Service Worker dan Web App Manifest.

B. Service Worker

Service Worker merupakan jenis dari Web Broker API yang berjalan pada *background* perangkat lunak peramban pada sisi klien. Service Worker merupakan file Javascript yang bekerja pada *thread* proses yang berbeda *main thread* dari perangkat lunak *browser*. Dengan Service Worker memungkinkan website untuk menjalankan *network request*, *caching* data, *background sync* dan *push notification* kepada sistem perangkat [11].

Service Worker memiliki tiga langkah dalam daur hidupnya (*lifecycle*), yang pertama adalah Registration (Pendaftaran), Installation (Instalasi), dan Activation (Pengaktifan). Sebelum menerapkan Service Worker, dilakukan tahap pendaftaran file dari Javascript Service Worker. Pendaftaran dengan melakukan memprogram berkas file Javascript untuk dapat mengenali file Service Worker (contoh program pada Gambar 1). Berkas dengan file Javascript yang berfungsi untuk meregistrasi file Service Worker harus terpisah dengan file Service worker itu sendiri.

Selanjutnya setelah berkas Service Worker diketahui maka akan dilanjutkan proses instalasi dari Service Worker pada browser. Pada proses Instalasi Service Worker akan memicu *event-event* yang sudah diprogram pada Service Worker seperti melalui *caching* pada website. Setelah proses Instalasi tahap terakhir adalah Pengaktifan, pada proses ini Service Worker sudah berjalan pada website.

```

if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('/service-worker.js')
  .then(function(registration) {
    console.log('Registration successful, scope is:', registration.scope);
  })
  .catch(function(error) {
    console.log('Service worker registration failed, error:', error);
  });
}

```

Gambar 1. Kode registrasi file Service Worker dengan nama file *service-worker.js*.

C. Web App Manifest

Web App Manifest merupakan file JSON yang mendeskripsikan sebuah *Progressive Web Application* untuk dikenali dan diperlakukan selayaknya Aplikasi yang terinstall. Dalam file Web App Manifest pengembang dapat memberikan informasi seperti nama aplikasi, *icons*, dan ragam konfigurasi tampilan. Berkas file dapat ditulis dengan ekstensi file *.json* ataupun *.webmanifest* namun tetap ditulis dalam struktur notasi JSON seperti Gambar 2.

```

() manifest.json > ...
1  {
2    "name": "Sistem Monitoring Listrik",
3    "short_name": "SisMo",
4    "description": "Sistem Monitoring Listrik",
5    "icons": [
6      {
7        "src": "/skripsi/favicon.webp",
8        "sizes": "256x256",
9        "type": "image/webp"
10     },
11     {
12       "src": "/skripsi/favicon.png",
13       "type": "image/png",
14       "sizes": "512x512",
15       "purpose": "any maskable"
16     }
17   ],
18   "start_url": "/skripsi/",
19   "display": "standalone",
20   "theme_color": "#e0f2fe",
21   "background_color": "#38bdf8"
22 }

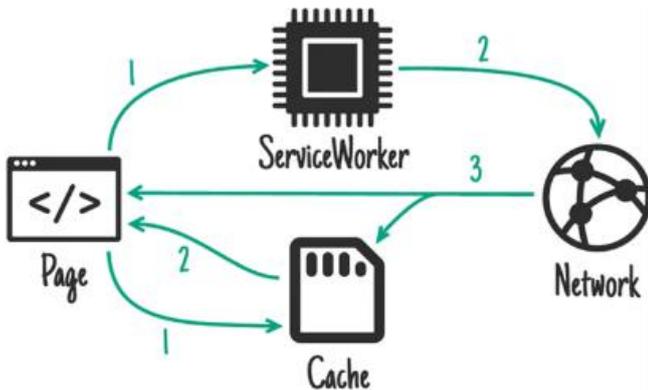
```

Gambar 2. Kode Web App Manifest dalam file *manifest.json*.

D. Cache then Network

Cache then network merupakan istilah dalam strategi menyimpan dan menyajikan data *cache* oleh Service Worker dengan cara melakukan dua *request* secara bersamaan [12]. Ide dari strategi ini adalah melakukan dua *request* secara bersamaan yang pertama kepada *cache* dan yang kedua

kepada jaringan melalui Service Worker. Ketika *request* terhadap jaringan berhasil dan didapatkan data baru maka data cache yang disajikan pada halaman web adalah yang terbaru, setelah data terbaru didapatkan selanjutnya Service Worker menyimpan data tersebut ke dalam *cache* browser. Ketika *request* menuju jaringan tidak berhasil maka data yang akan disajikan adalah data dari *request* sebelumnya yang sudah tersimpan dalam *cache*. Alur dari strategi ini memungkinkan halaman, dan data dinamis yang di-*request* setelah halaman dimuat dapat disimpan dalam *cache* dan diakses secara *offline*. Alur dari strategi *cache then network* dapat dilihat pada Gambar 3.



Gambar 3. Alur Cache then Network.

Dengan menerapkan strategi *caching then network* maka memungkinkan data aset statis ataupun dinamis dapat dimuat dalam keadaan *offline*. Hal ini dikarenakan ketika kondisi *offline* maka halaman akan mengambil *request* dari *cache* yang pernah dimuat sebelumnya, sebaliknya ketika kondisi *online* maka halaman akan mengambil *request* langsung ke *server* dan menyimpannya kembali ke dalam *cache* sehingga data dalam *cache* tetap mengalami pembaruan selama berada dalam jaringan.

E. Javascript

Javascript adalah bahasa pemrograman yang banyak diketahui digunakan pada website dengan berbasis OOP (*object-oriented programming*) pemrograman berbasis objek [13]. Javascript merupakan kunci untuk membangun HTML yang dinamis dimana dapat membuat website untuk mendukung pembuatan animasi pada website dan website yang interaktif. Javascript adalah jenis bahasa pemrograman dinamis yang dapat mengetahui bahwa variabel pada Javascript tidak perlu dikenalkan/dieksekusi sebelum ada pernyataan eksekusi variabel terkait pada program dan tipe data dari Javascript secara dinamis dapat dipecahkan sendiri selama eksekusi program berjalan.

F. Website

Website atau web site adalah kumpulan dari halaman web (*web page*) dan konten yang terkait dalam cakupan sebuah nama domain yang publikasi di sebuah web server ke dalam WWW (*World Wide Web*) di internet [14]. Web page sendiri adalah bagian dari website yang terbuat dari dokumen

hypertext yang dapat menampilkan konten-konten seperti tulisan, gambar, dan video. Dalam metaforanya web page adalah *book pages* (halaman buku) yang berisi konten dari buku itu sendiri. Website dapat dikases melalui protokol HTTP melalui *browser*.

G. NodeMCU

NodeMCU adalah modul *development board* (seperti Arduino) dengan *platform IoT (Internet of Things)* berbasis ESP8266 yang menggunakan *firmware* yang berbasis bahasa pemrograman Lua [15]. Firmware dan desain papan prototipe adalah *open source* (sumber terbuka). NodeMCU secara *built-in* terpasang modul Wi-Fi ESP8266 yang merupakan Wi-Fi SoC (*System on Chip*) yang secara luas digunakan pada aplikasi IoT [16]. ESP8266 adalah SoC dengan *built-in* perangkat lunak jaringan TCP/IP dan kemampuan mikrokontroler yang diproduksi Espressif Systems di Shanghai China.



Gambar 4. NodeMCU Devkit 1.0 oleh Vowstar.

H. Modul PZEM-004T

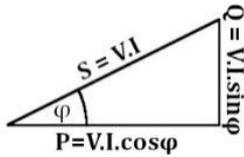
PZEM-004T merupakan modul sensor kWh *Power Meter* yang dibuat oleh perusahaan Peacefair yang mempunyai kemampuan pengukuran tegangan rms (V_{rms}), arus rms (I_{rms}), faktor daya, serta hasil kakulasi dari daya aktif dan penggunaan energy (*kWh meter*) [17]. PZEM-004T memiliki dimensi fisik 73.7mm x 30mm dengan berat 46g, bentuk dari sensor dapat dilihat pada Gambar 5. Varian modul PZEM-004T tersedia dalam dua jenis yaitu PZEM-004T-100A untuk jangkauan pengukuran arus sampai dengan 100 Ampere dan PZEM-004T-10A untuk jangkauan pengukuran arus sampai dengan 10 Ampere.



Gambar 5. Modul Sensor PZEM-004T.

I. Daya Listrik AC

Daya listrik merupakan kapasitas jumlah energi listrik yang digunakan dalam melakukan usaha pada sistem tenaga listrik yang dinyatakan dalam satuan Watt (P) [18]. Terdapat tiga macam daya listrik, yaitu daya aktif (P), daya reaktif (Q), dan daya semu (S). Ketiga hubungan daya tersebut digambarkan dalam istilah Segitiga Daya seperti yang ditunjukkan pada Gambar 6.



Gambar 6. Segitiga daya.

Daya semu (S) atau daya nyata adalah nilai daya dari hasil penjumlahan antara daya aktif (P) dengan daya reaktif (Q) atau perkalian antara tegangan rms (V_{rms}) dengan arus rms (I_{rms}) yang dinyatakan dalam satuan VA (Volt Ampere).

$$S = V \cdot I \tag{1}$$

Daya aktif (P) merupakan daya yang terpakai untuk melakukan usaha sebenarnya dalam tenaga listrik yang dinyatakan dalam satuan Watt. Rumus untuk daya aktif dituliskan sebagai berikut:

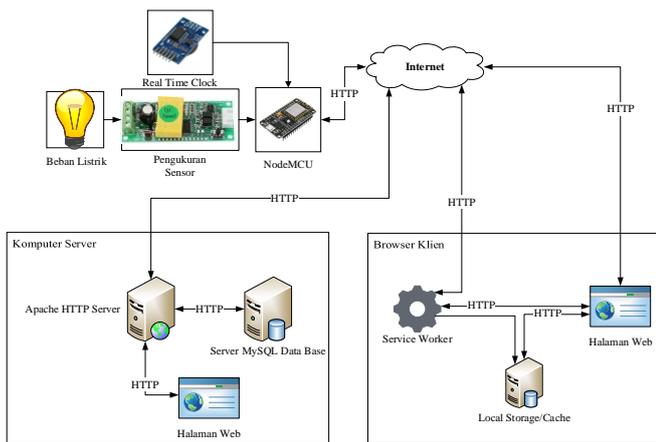
$$P = V \cdot I \cdot \cos \phi \tag{2}$$

Daya Reaktif (Q) merupakan jumlah daya yang diperlukan dalam usaha pembentukan medan magnet pada komponen reaktif [19]. Satuan untuk daya reaktif adalah VAR (Volt Ampere Reaktif)

$$Q = V \cdot I \cdot \sin \phi \tag{3}$$

III. METODE PENELITIAN

A. Diagram Rancangan Sistem



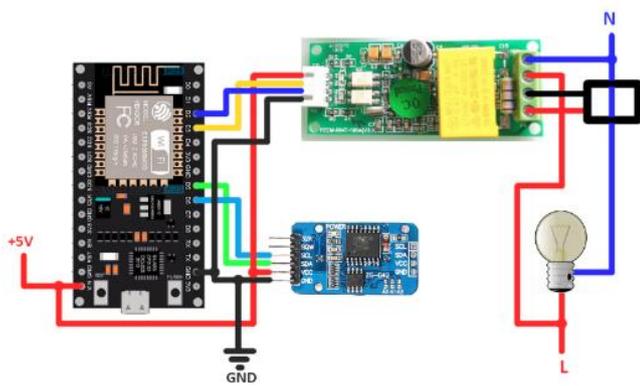
Gambar 7. Diagram rancangan sistem.

Penjelasan rancangan diagram sistem pada Gambar 7.

1. Pengukuran Sensor menggunakan modul PZEM-004T mendapatkan data arus, tegangan, daya, dan faktor daya pada beban listrik.
2. Modul RTC DS3231 mengirim data tanggal dan jam terbaru ke NodeMCU.
3. Protokol yang digunakan untuk berkomunikasi dengan web server menggunakan HTTP.
4. NodeMCU mengolah hasil pengukuran sensor arus dan tegangan lalu mengirim data arus, tegangan dan waktu data pengukuran ke server melalui internet dengan melakukan request HTTP pada halaman website yang diprogram menggunakan PHP.
5. Apache Web Server menerima request HTTP (lalu memberi kode response kembali ke klien) dan melanjutkan menuju URL direktori halaman website yang diminta oleh NodeMCU.
6. Halaman website yang diprogram menggunakan PHP yang di-request oleh NodeMCU akan mengirim perintah untuk menyimpan data yang dikirim dari request NodeMCU untuk disimpan pada basis data server MySQL.
7. Dalam MySQL terdapat tabel dan kolom-kolom yang telah diorganisir untuk menyimpan data-data yang dikirim oleh NodeMCU.
8. Pada sisi klien akan terjadi dua request secara bersamaan saat Service Worker ter-install dan halaman web di-request oleh klien inilah yang disebut dengan strategi *cache then network*.
 - a. Request pertama terjadi pada Service Worker, ketika dalam kondisi *online* maka Service Worker akan melakukan request ke server untuk mendapatkan data terbaru, setelah mendapat data terbaru maka selanjutnya menyimpan data terbaru tersebut pada *cache* peramban klien dan menampilkan data tersebut pada halaman web.
 - b. Request kedua terjadi ketika dalam kondisi *offline*, dimana halaman web mengambil data yang sudah tersimpan pada *cache* dan menyajikan pada halaman web.

B. Diagram Rangkaian Sistem

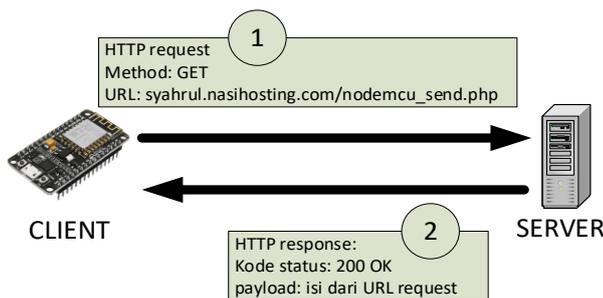
Rancangan diagram rangkaian antara NodeMCU dan PZEM-004T dirangkai dengan menghubungkan pin Tx NodeMCU dengan Rx PZEM-004T sedangkan Rx NodeMCU dihubungkan dengan pin Tx PZEM-004T seperti yang ditampilkan pada Gambar 8 dan RTC dihubungkan dengan NodeMCU pin SDA dan SCL masing-masing perangkat. Pada rangkaian ini NodeMCU berfungsi sebagai *client* yang akan mengambil data waktu dari RTC dan data tegangan AC, arus AC, faktor daya, daya aktif, dan energi (kWh) dari modul PZEM-004T lalu mengirimkan data tersebut kepada web server untuk disimpan pada database.



Gambar 8. Diagram Rangkaian Sistem.

C. Pengiriman Data NodeMCU menuju Web Server

Ketika NodeMCU sudah dihubungkan ke internet melalui WiFi maka tahap selanjutnya adalah pengiriman data menggunakan protokol HTTP. NodeMCU akan bertugas sebagai *client* yang akan melakukan *request* URL kepada web server. Dalam protokol HTTP ketika *client* melakukan *request* maka web server akan merespon dengan mengembalikan sebuah kode status dan sumber daya yang diminta sesuai URL yang di-*request* oleh *client* [20]. Pada penelitian ini URL yang akan di-*request* oleh NodeMCU adalah sebuah halaman web (syahrul.nasihosting.com/nodemcu_send.php) dengan tambahan parameter-parameter hasil pengukuran tegangan, arus, faktor daya, energi dan waktu dengan menggunakan metode HTTP/GET. Setelah *request* halaman web diterima oleh web server maka selanjutnya halaman web akan melakukan pengambilan data dari URL parameter yang selanjutnya melakukan *query database* SQL ke server untuk menyimpan data hasil pengukuran sensor menggunakan pemrograman PHP yang berada pada halaman web yang telah di-*request* tersebut. Setelah proses *query* SQL selesai, web server akan mengirim *response* berupa kode (kode = 200 jika berhasil) serta *payload* dari URL yang di-*request* seperti ilustrasi pada Gambar 9.



Gambar 9. HTTP Transaction.

D. Penerapan Progressive Web Application

Fitur *Progressive Web Application* (PWA) yang akan diterapkan pada *website* monitoring adalah *installable* dan *offline*. Kedua fitur PWA tersebut dapat membuat sebuah

website berperilaku layaknya aplikasi *native* pada perangkat lunak.

Untuk menerapkan fitur *installable* maka perlu dilakukan pemrograman javascript terhadap Service Worker untuk mengenali *website* sebagai *website* berbasis PWA dan melengkapi aset *website* untuk dapat dikenali sebagai *website* dengan PWA. Kelengkapan *website* PWA agar fitur *installable* dapat diterapkan adalah berkas *Web App Manifest* serta *icon*(ikon) untuk mewakili *website* saat terinstall pada perangkat.

Pada fitur *offline* penerapan dilakukan agar *website* tetap dapat diakses saat *offline* serta data-data pengukuran sebelumnya tetap dapat ditampilkan walaupun kondisi tanpa jaringan. Untuk menerapkan hal ini diperlukan pemrograman javascript pada *website* untuk dapat mengambil serta menyimpan data dalam *cache* dan memahami skenario penyajian data sesuai kebutuhan dan kondisi data saat dalam kondisi tanpa jaringan. Dalam segi skenario penyajian data hasil pengukuran menggunakan skenario yang disebut dengan *Cache then Network*. Skenario ini dipilih karena ideal untuk jenis penyajian data yang sering diperbarui [12].

Pada fitur *offline*, Service Worker digunakan untuk mengambil data *request* URL pengguna (peramban), data yang telah di-*request* selanjutnya disimpan oleh Service Worker pada *cache* peramban. Selanjutnya untuk menyajikan data dalam *cache* tersebut perlu dilakukan adalah ketika pengguna melakukan *request* kembali pada data yang sama maka dibuat sebuah pemrograman Javascript untuk mengambil data dalam *cache*.

IV. HASIL DAN PEMBAHASAN

A. Rangkaian Uji Coba Monitoring Daya Listrik

Rangkaian uji coba untuk monitoring daya (Gambar 10) listrik menggunakan empat beban lampu pijar sebagai media percobaan monitoring daya listrik dengan spesifikasi masing-masing nilai daya beban, yaitu satu lampu pijar 15 Watt, dan tiga lampu pijar 25 Watt. Seluruh beban dihubungkan dengan modul PZEM-004T seperti pada diagram rangkaian sistem (Gambar 8).



Gambar 10. Rangkaian uji coba monitoring daya listrik.

B. Pengambilan dan Penyajian Data menuju Website

Dalam hasil pengujian pengambilan data arus, tegangan, daya, energi, dan faktor daya didapat hasil pengambilan dan pengiriman data menuju web server diketahui membutuhkan waktu 3-6 detik untuk mengirim data hal ini dapat dilihat pada Gambar 11 dimana melihat selisih waktu data pada kolom "time".

	id	arus	volt	power	cosphi	frequency	energy	time	date	timer	STATE	
<input type="checkbox"/>	Edit Copy Delete	68293	0.054	223.7	6.1	0.5	50	0.002	19:50:45	2022-09-10	1731	ON
<input type="checkbox"/>	Edit Copy Delete	68308	0.054	224.5	6.1	0.5	50	0.002	19:51:39	2022-09-10	1786	ON
<input type="checkbox"/>	Edit Copy Delete	68258	0.055	222.7	6.1	0.5	50	0.002	19:48:37	2022-09-10	1604	ON
<input type="checkbox"/>	Edit Copy Delete	68311	0.054	225.4	6.1	0.5	49.9	0.002	19:51:50	2022-09-10	1797	ON
<input type="checkbox"/>	Edit Copy Delete	68305	0.054	226.3	6.1	0.5	50	0.002	19:51:29	2022-09-10	1775	ON
<input type="checkbox"/>	Edit Copy Delete	68296	0.054	224.5	6.2	0.51	50	0.002	19:50:55	2022-09-10	1741	ON
<input type="checkbox"/>	Edit Copy Delete	68265	0.055	223.3	6.2	0.5	50	0.002	19:49:02	2022-09-10	1628	ON
<input type="checkbox"/>	Edit Copy Delete	68301	0.055	227.2	6.2	0.5	50	0.002	19:51:15	2022-09-10	1761	ON
<input type="checkbox"/>	Edit Copy Delete	68267	0.054	222.6	6.1	0.51	50	0.002	19:49:09	2022-09-10	1635	ON
<input type="checkbox"/>	Edit Copy Delete	68264	0.053	221.3	6.1	0.52	50	0.002	19:48:58	2022-09-10	1625	ON
<input type="checkbox"/>	Edit Copy Delete	68274	0.054	222.3	6.2	0.52	50	0.002	19:49:34	2022-09-10	1660	ON
<input type="checkbox"/>	Edit Copy Delete	68248	0.054	219.6	6.2	0.52	50	0.002	19:47:59	2022-09-10	1566	ON
<input type="checkbox"/>	Edit Copy Delete	68319	0.054	224	6.2	0.51	50	0.002	19:52:19	2022-09-10	1825	ON
<input type="checkbox"/>	Edit Copy Delete	70161	0.033	213.4	0	0	50	0.01	21:50:50	2022-09-10	93	OFF
<input type="checkbox"/>	Edit Copy Delete	68281	0.055	222.4	6.2	0.51	49.9	0.002	19:50:01	2022-09-10	1687	ON
<input type="checkbox"/>	Edit Copy Delete	68256	0.055	220.7	6.1	0.5	49.9	0.002	19:48:30	2022-09-10	1596	ON
<input type="checkbox"/>	Edit Copy Delete	68326	0.054	223.1	6.1	0.51	50	0.002	19:52:45	2022-09-10	1852	ON
<input type="checkbox"/>	Edit Copy Delete	68273	0.054	222.7	6.2	0.52	50	0.002	19:49:31	2022-09-10	1657	ON
<input type="checkbox"/>	Edit Copy Delete	68289	0.055	223.3	6.2	0.5	50	0.002	19:50:31	2022-09-10	1718	ON
<input type="checkbox"/>	Edit Copy Delete	68378	0.054	224.7	6.1	0.51	50	0.002	19:49:40	2022-09-10	1676	ON

Gambar 11. Hasil pengiriman data.

Selanjutnya data yang sudah berhasil disimpan disajikan dalam bentuk Javascript Object Notation (JSON) (yang ditunjukkan pada Gambar 12) sehingga dapat disajikan dalam bentuk informasi.

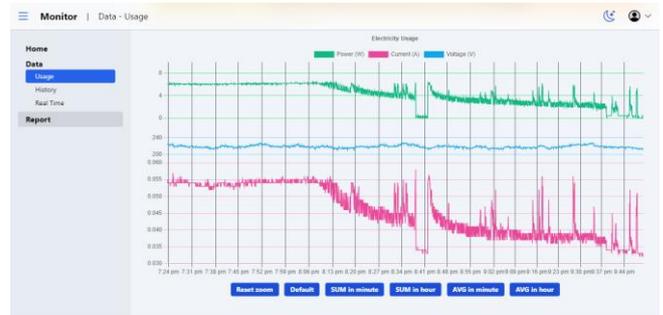
```

{
  "result": [
    {
      "id": "24072",
      "arus": "0.033",
      "volt": "216.8",
      "power": "0",
      "cosphi": "0",
      "frequency": "49.9",
      "energy": "0.152",
      "time": "19:09:16",
      "date": "2022-08-20",
      "timer": "442",
      "STATE": "ON"
    },
    {
      "id": "24073",
      "arus": "0.033",
      "volt": "216.4",
      "power": "0.4",
      "cosphi": "0.06",
      "frequency": "49.9",
      "energy": "0.152",
      "time": "19:09:18",
      "date": "2022-08-20",
      "timer": "444",
      "STATE": "ON"
    }
  ]
}

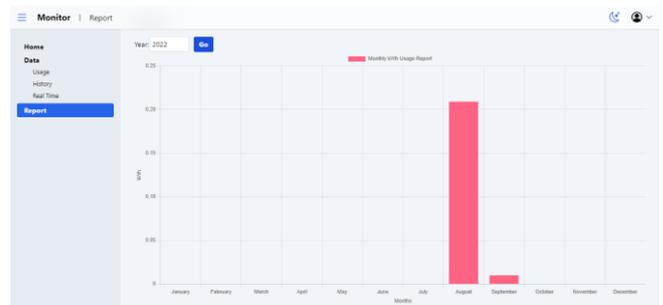
```

Gambar 12. Bentuk data yang berupa JSON.

Data dalam bentuk JSON tersebut diolah sehingga dapat menampilkan grafik pengukuran seperti pada Gambar 13 dan Gambar 14.



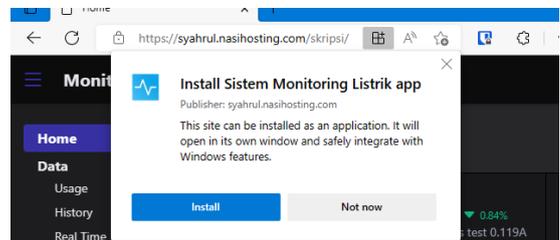
Gambar 13. Grafik hasil monitoring.



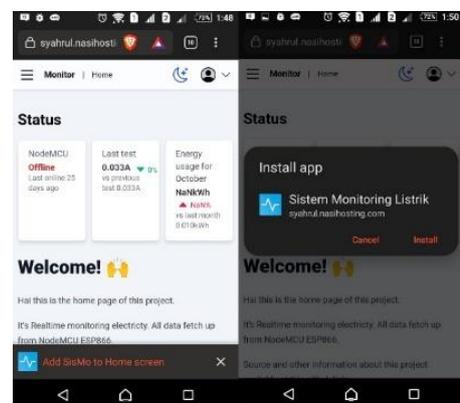
Gambar 14. Grafik kWh bulanan.

C. Pengujian Fitur Installable pada Website

Pengujian dilakukan untuk mengetahui apakah website berbasis PWA dapat "di-install" pada perangkat melalui perantara browser dengan basis Chromium (Google Chrome, Opera, Microsoft Edge, dll.). Website yang menggunakan fitur *installable* akan memunculkan *prompt* pemberitahuan pada layar untuk melakukan instalasi website PWA pada perangkat seperti Gambar 15 dan Gambar 16.

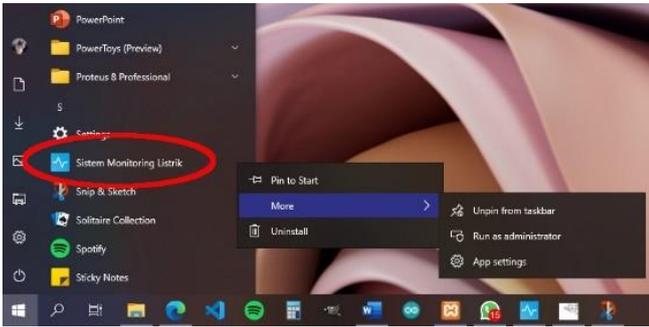


Gambar 15. Prompt instalasi website PWA pada PC.



Gambar 16. Prompt Instalasi pada perangkat android.

Pengguna yang sudah melakukan proses instalasi dari *prompt* yang muncul pada saat membuka website akan mendapatkan bahwa website sudah tersedia dalam perangkat selengkapnya aplikasi *native* perangkat yang sudah ter-*install* seperti yang ditunjukkan pada Gambar 17 dan 18.



Gambar 17. Website PWA yang sudah terinstall pada PC Windows.



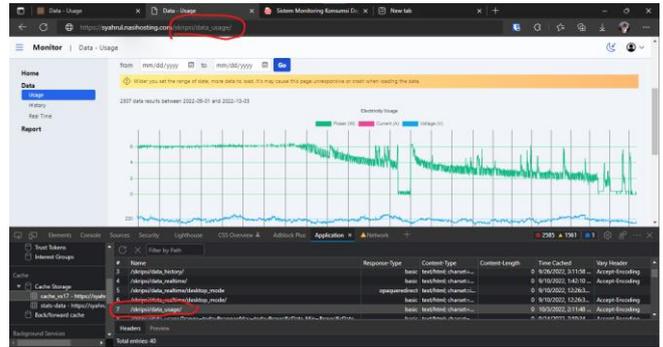
Gambar 18. Website PWA yang sudah terinstal pada *mobile* android.

D. Pengujian Fitur Offline

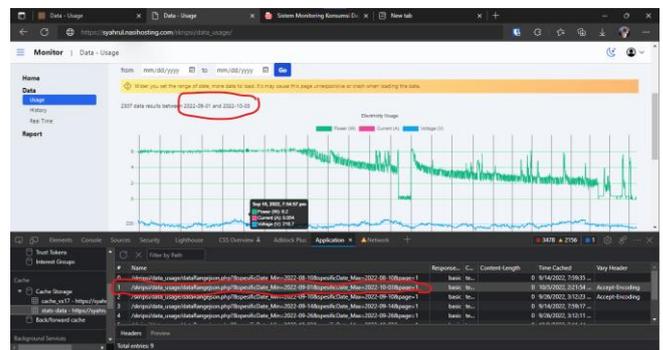
Pengujian fitur offline dilakukan untuk mengetahui apakah website yang sudah diterapkan fitur PWA dapat diakses walupun tanpa jaringan dan data monitoring yang sudah diakses sebelumnya (pernah dibuka saat *online*) tetap dapat diakses. Untuk melakukan pengujian ini halaman website berbasis PWA setidaknya pernah diakses satu kali sebelumnya (saat kondisi *online*) sebelum diuji coba untuk diakses secara *offline*. Hal ini bertujuan untuk memberi kesempatan Service Worker untuk melakukan *caching* pada data halaman yang akan diakses saat offline.

Adapun dua tahap pengujian website monitoring berbasis PWA sebagai berikut. Pertama halaman website dapat diakses secara *offline*. Kedua data grafik monitoring dapat diakses secara *offline*. Pada Gambar 19 ditunjukkan bahwa browser sudah melakukan *caching* pada halaman website dengan url "/skripsi/data_usage/" hal ini memungkinkan data yang telah di-*cache* untuk dipanggil saat kondisi *offline*. Sedangkan pada Gambar 20 menunjukkan bahwa *cache* data

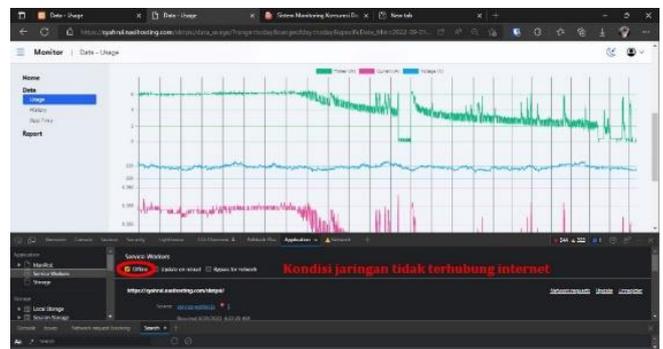
grafik monitoring untuk data monitoring pada tanggal 1 Agustus 2022 hingga tanggal 3 Oktober 2022 sudah disimpan pada *browser* sehingga memungkinkan grafik dapat dimunculkan pada saat *offline*. Hasil pengujian akses saat *offline* ditunjukkan pada Gambar 21.



Gambar 19. *Cache* url "/skripsi/data_usage/" berhasil disimpan pada browser.



Gambar 20. Data grafik monitoring dengan rentang 01-08-2022 hingga 01-10-2022 berhasil disimpan pada browser.

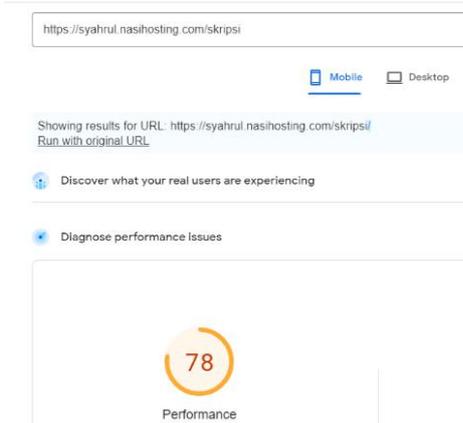


Gambar 21. Hasil uji coba akses halaman website dan grafik monitoring yang telah disimpan dalam *cache* saat kondisi *offline*.

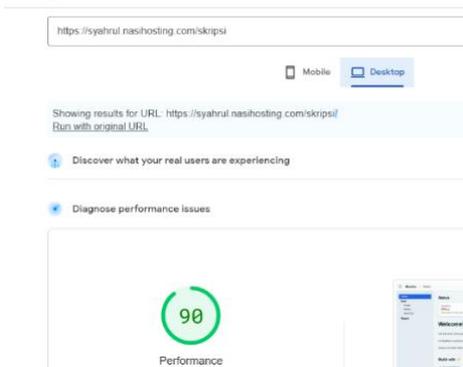
E. Perbandingan Performa Akses Website PWA dan non-PWA

Pengujian dilakukan menggunakan dua alat populer dalam uji coba performa website, yaitu PageSpeed Insights dan GTmetrix. Pada pengujian menggunakan PageSpeed Insights untuk website yang belum diaktifkan PWA didapatkan skor 78 pada perangkat *mobile* (yang ditunjukkan pada Gambar 22) dan 90 (yang ditunjukkan pada Gambar 23) pada perangkat PC. Sedangkan website monitoring yang telah diterapkan PWA maka hasil pengujian pada perangkat

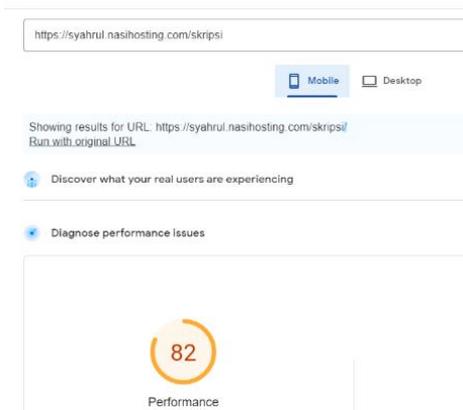
mobile mendapat skor 82 (yang ditunjukkan pada Gambar 24) dan 99 pada perangkat PC (yang ditunjukkan pada Gambar 25). Hasil pengujian menggunakan alat GTmetrix juga menampilkan peningkatan pada performa website ketika PWA diaktifkan (ditunjukkan pada Gambar 26).



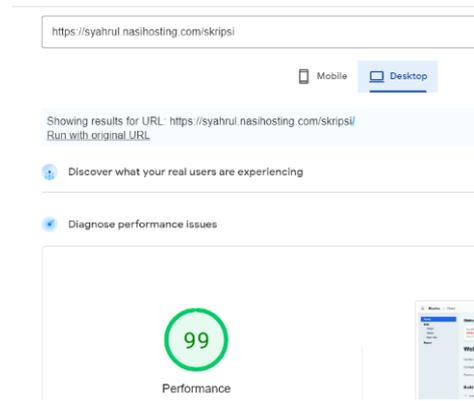
Gambar 22. Skor pengujian PageSpeed Insights tanpa PWA pada perangkat mobile.



Gambar 23. Skor pengujian PageSpeed Insights tanpa PWA pada perangkat PC.



Gambar 24. Skor pengujian PageSpeed Insights dengan PWA pada perangkat mobile.



Gambar 25. Skor pengujian PageSpeed Insights dengan PWA pada perangkat PC.



Gambar 26. Skor perbandingan performa website tanpa PWA (kiri) dan dengan PWA (kanan) menggunakan GTmetrix.

F. Perbandingan Penggunaan RAM dan CPU pada Website dengan Fitur Installable

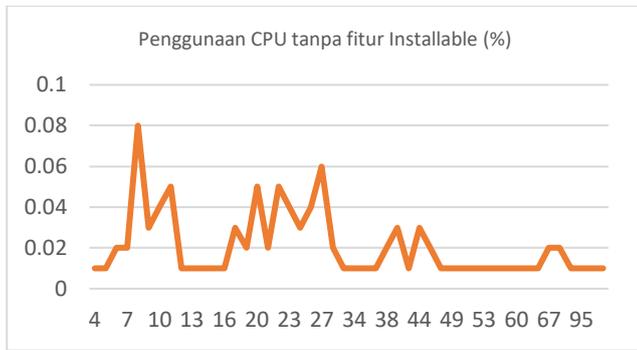
Pengujian konsumsi RAM dan CPU dilakukan dengan menggunakan perangkat lunak MSI Afterburner dengan jumlah sampel sebanyak 100 dan durasi pengambilan sampel 1 sampel/detik. Skenario pengujian untuk website berbasis PWA dengan fitur *installable* dilakukan dengan mengakses langsung melalui *shortcut* yang sudah tersedia pada perangkat setelah melakukan proses install pada saat mengakses website pertama kali. Sedangkan skenario kedua adalah dengan mengakses langsung dari web browser. Hasil pengujian diketahui bahwa penggunaan RAM dan CPU dengan mengakses via fitur *installable (shortcut)* menghasilkan penggunaan RAM dan CPU yang lebih rendah dibanding mengakses website langsung menggunakan web browser. Data disajikan dengan bentuk *Probability Density Function (PDF)*. Hal ini berguna untuk melihat distribusi nilai kemungkinan sehingga data yang disajikan dapat menunjukkan keadaan yang sesungguhnya [21]. Untuk mendapatkan nilai *mean/rata-rata* dari sebuah PDF digunakan rumus pada persamaan (4) [22].

$$\mu = \sum x \cdot P(x) \tag{4}$$

μ = mean

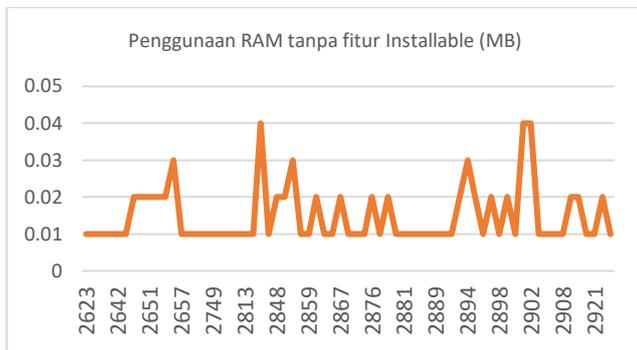
x = variabel yang muncul

$P(x)$ = Probabilitas dari variabel yang muncul



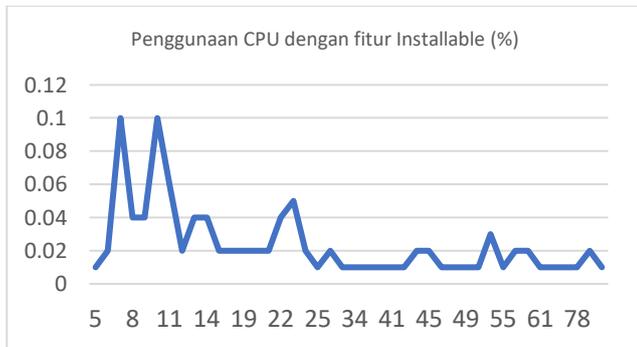
Gambar 27. Grafik PDF Penggunaan CPU tanpa fitur *installable*.

Gambar 27 menunjukkan bahwa distribusi jumlah penggunaan CPU paling tinggi terdapat pada 7% dan 10% serta pada 16% dan 27%. Rata-rata dari PDF grafik Gambar 26 adalah 29,75%.



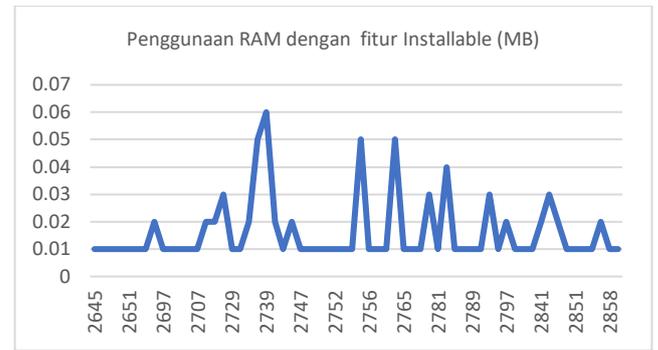
Gambar 28. Grafik PDF Penggunaan RAM tanpa fitur *installable*.

Dari grafik pada Gambar 28 ditunjukkan bahwa distribusi jumlah penggunaan RAM paling sering muncul pada jumlah 2813 MB dan 2908 MB. Rata-rata dari grafik Gambar 27 adalah 2824,8 MB.



Gambar 29. Grafik PDF Penggunaan CPU dengan fitur *installable*.

Grafik pada Gambar 29 menunjukkan bahwa distribusi jumlah penggunaan CPU paling tinggi terdapat pada 7% hingga 10% hal ini lebih rendah dibanding pada grafik Gambar 26 yang menunjukkan distribusi nilai penggunaan CPU dengan akses website tanpa *shortcut*. Dari hasil tersebut diketahui rata-rata PDF adalah 25,53%.



Gambar 30. Grafik PDF Penggunaan RAM dengan fitur *installable*.

Dari grafik pada Gambar 30 ditunjukkan bahwa distribusi jumlah penggunaan RAM paling tinggi pada nilai RAM 2739 MB lebih rendah dibanding grafik pada Gambar 27 yang menunjukkan distribusi jumlah penggunaan RAM ada pada 2813 MB dan 2908 MB. Pengujian menunjukkan rata-rata dari PDF pada data Gambar 29 adalah 2761,05 MB.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

1. Pada pengujian fitur *installable* pada website monitoring berbasis PWA menghasilkan sebuah *shortcut* aplikasi selayaknya aplikasi *native* pada perangkat *mobile* dan PC yang mengarah langsung ke website sistem monitoring daya listrik.
2. Dengan menerapkan PWA pada website sistem monitoring, website dapat diakses secara *offline* dengan memanfaatkan *cache* yang telah diakses sebelumnya saat *online*.
3. Nilai performa kecepatan akses website dengan PWA yang telah diterapkan skenario *Cache then Network* lebih baik dibanding website tanpa penerapan PWA. Skor lebih baik 5% pada sisi *mobile* menggunakan alat uji PageSpeed Insights dan pada sisi perangkat PC meningkat 9,5% pada alat uji PageSpeed Insight. Sedangkan pada alat uji GTmetrix skor dengan penerapan PWA lebih baik 4% pada penilaian *performance score* (ditunjukkan pada Gambar 23).
4. Dengan menerapkan fitur *installable* pada PWA diketahui dapat meringankan penggunaan RAM dan CPU ketika pengguna mengakses website. Penggunaan CPU ketika membuka website dengan fitur *installable* lebih rendah 4,22% dengan persentasi akses tanpa fitur *installable* sebesar 29,75% dan setelah akses menggunakan fitur *installable* sebesar 25,53%. Penggunaan RAM juga menurun 2,28% setelah penggunaan fitur *installable* dengan nilai awal sebelum penggunaan fitur sebesar 2824,8 MB menjadi 2761,05 MB.

B. Saran

Dalam pengerjaan penelitian ini tentu masih banyak kekurangan yang dapat menjadi bahan pengembangan yang berkaitan dengan topik *Progressive Web Application* (PWA) pada sebuah sistem monitoring. Adapun saran yang diberikan penulis antara lain adalah penerapan fitur-fitur lain dari website dengan teknologi PWA, seperti *Push Notification*, sistem yang dapat berjalan di-*background* sistem serta skenario-skenario *cache* sesuai dengan kondisi ideal penyajian data pada *website*.

VI. REFERENSI

- [1] L. N. Yuliatin, M. D. Djamaludin dan A. M. Sari, "Analisis Sikap dan Perilaku Penghematan Listrik Sektor Rumah Tangga," *Jurnal Ilmu Keluarga dan Konsumen*, 4(1), pp. 82-90, 2011.
- [2] H. Galla and B. Suprianto, "Monitoring Biaya Dan Pengukuran Konsumsi Daya Listrik Berbasis Arduino Mega2560 Menggunakan WEB," *Jurnal Teknik Elektro*, (8)3, pp. 695-702, 2019.
- [3] Z. Ramadhan, "Implementasi Sistem Monitoring Daya Listrik Berbasis Web dan Protokol Komunikasi Websocket," Universitas Brawijaya, Malang, 2018.
- [4] M. Butkiewicz, H. V. Madhyastha and V. Sekar, "Understanding Website Complexity: Measurements, Metrics, and Implications," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, 2011.
- [5] M. R. Ridho, "Perbandingan Performa Progressive Web Apps dan Mobile Web Terkait Waktu Respon, Penggunaan Memori dan Penggunaan Media Penyimpanan," Universitas Brawijaya, Malang, 2018.
- [6] Gfk Indonesia, "Mobile Apps Market Study," Gfk Crossmedia Link Indonesia, 2015.
- [7] S. S. Tandel and A. Jamadar, "Impact of Progressive Web Apps on Web App," *International Journal of Innovative Research in Science*, 7(9), pp. 9439-9444, 2018.
- [8] K. Syaifudin, E. Pranata, N. A. Wafa and A. R. Dian, "Analisis Usability pada Perbandingan Web-Native dengan," *Informatika*, 2019.
- [9] A. Laurensius, "Platform e-learning untuk pembelajaran pemrograman web menggunakan konsep," Institut Teknologi Sepuluh November, Semarang, 2017.
- [10] A. Faisol and S. Noertjahjono, "PERANCANGAN SISTEM PELAYANAN KELURAHAN DIGITAL TERPADU (SIPERDIT) BERBASIS PROGRESSIVE WEB APP (PWA)," *Jurnal MNEMONIC*, (2)2, pp. 1-4, 2019.
- [11] L. Adi, "Platform e-Learning untuk Pembelajaran Pemrograman Web Menggunakan Konsep Progressive Web Apps," Institut Teknologi Sepuluh November, Semarang, 2017.
- [12] J. Archibald, "The Offline Cookbook," 28 September 2020. [Online]. Available: <https://web.dev/offline-cookbook>.
- [13] M. Tommi and T. Antero, *Using JavaScript as a Real Programming Language*, Sun Microsystems Laboratories, 2007.
- [14] S. U. Tian, "PROTOTIPE SISTEM MONITORING PARAMETER PEMBANGKIT LISTRIK TENAGA SURYA BERBASIS INTERNET OF THINGS," Universitas Negeri Yogyakarta, 2017.
- [15] M. Yuan, "Set up and gets started programming for this open source IoT development environment, NodeMCU," 12 Juni 2017. [Online]. Available: <https://developer.ibm.com/tutorials/iot-nodemcu-open-why-use>.
- [16] circuito.io, "NODEMCU - A PERFECT BOARD FOR IOT," 21 November 2018. [Online]. Available: <https://www.circuito.io/blog/nodemcu-esp8266/>.
- [17] S. Anwar, T. Artono, Nasrul, Dasrul and A. Fadli, "Pengukuran Energi Listrik Berbasis PZEM-004T," *Proceeding Seminar Nasional Politeknik Negeri Lhokseumawe*, vol. III, no. 1, pp. 272-276, 2019.
- [18] N. Safitri, Suryati and Rachmawati, *ANALISA RANGKAIAN LISTRIK (Teori Dasar, Penyelesaian Soal dan Soal-Soal Latihan)*, Lhoksumawe: Politeknik Negeri Lhokseumawe, 2017.
- [19] I. Dinata and S. Wahri, "Implementasi Wireless Monitoring Energi Listrik Berbasis Web Database," *Jurnal Nasional Teknik Elektro*, 4(1), pp. 83-88, 2015.
- [20] C. Wong, *Http pocket reference: Hypertext transfer*, O'Reilly Media, Inc., 2000.
- [21] A. Basuki, *Fungsi Kepadatan Probabilitas*, Surabaya: Institut Teknologi Sepuluh November, 2004.
- [22] Texas Gateway, "4.2 Mean or Expected Value and Standard Deviation | Texas Gateway," Texas Education Agency, [Online]. Available: <https://www.texasgateway.org/resource/42-mean-or-expected-value-and-standard-deviation>. [Diakses 9 December 2022].

VII. BIODATA PENULIS



Nama penulis jurnal adalah M. Syahrul Fadli. Lahir pada tanggal 11 Januari 2000 di Desa Kelayu Utara, Kabupaten Lombok Timur, Provinsi Nusa Tenggara Barat. Penulis menempuh pendidikan SMK dengan jurusan Teknik Elektronika Industri di SMK Negeri 3 Mataram. Setelah lulus sekolah kejuruan, penulis melanjutkan pendidikan S1 di Institut Teknologi Nasional Malang jurusan Teknik Elektro dengan konsentrasi Teknik Komputer. Dengan riwayat pendidikan tersebut penulis memiliki keahlian pada bidang elektronika seperti pemrograman mikrokontroler dan robotika serta bidang komputer seperti pemrograman website.