Terakreditasi SINTA Peringkat 4
Surat Keputusan Direktur Jenderal Pendidikan Tinggi, Riset, dan Teknologi, Nomor: 158/E/KPT/2021
masa berlaku mulai Volume 5 Nomor 2 Tahun 2018 sampai Volume 10 Nomor 1 Tahun 2023

Terbit online pada laman web jurnal: **https://jurnal.plb.ac.id/index.php/tematik/index**

T E M A T I K

Jurnal **Teknologi Informasi Komunikasi (e-Journal)**

Vol. 9 No. 2 (2022) 191 - 201 ISSN Media Elektronik: 2443-3640

Analisis Dan Peningkatan Performa Aplikasi Berbasis Website Menggunakan *Stress Tools* Gtmetrix

Analysis and Performance Improvement of Website-Based Applications Using Stress Tools Gtmetrix

Asri Putri Dwi Gita Andini[1], Dian Wahyuningsih[2], Mahmud Yunus[3]
[1,2,3]Teknologi Informasi, STMIK PPKIA Pradnya Paramita Malang
asri_21520012@stimata.ac.id[1], dianwahyu@stimata.ac.id[2], myoenoes@stimata.ac.id[3]

Abstract

In the process of making a WebApp, there are several stages, one of which is implementation. At this stage the application will start to used and if there are obstacles that will be seen at this stage, such as the application performance decreases when accessed by a number of users, the waiting time for loading the page will be longer. This usually happens because the WebApp developer hasn't analyzed the application's performance using various stresses tools. Users or developers can test WebApps through various stress tools, like Gtmetrix. In the alpha beta test with GTmetrix, SIMLSP WebApp received grade D with a 68% performance and 72% structure on the login page, grade F with a 24% performance and 72% structure on the assessor account, an error analysis on the exam scheduling page. The optimization is the removal of unnecessary application code, adding the disabled attribute to the tag stylesheets, local access transfers in CSS/JavaScript, parameter use in JavaScript functions, and pagination. After that, alpha beta testing was carried out by retesting, it was found that grade B increased with a 84% and structure 87% on the login page, grade B with a 91% and structure 83% on the assessor account, and grade A with a 94% performance and 81% structure on the session scheduling page. With these results, the optimization has been successful.

Keywords: website application (WebApp), SIMLSP, alpha beta testing, cyclomatic complexity, gtmetrix

Abstrak

Pada proses pembuatan WebApp terdapat beberapa tahapan salah satunya adalah implementasi, yaitu ketika WebApp telah siap digunakan oleh pengguna. Di tahap ini aplikasi akan mulai digunakan dan jika terdapat kendala akan terlihat pada tahap ini, seperti performa aplikasi menurun ketika diakses oleh sejumlah pengguna, waktu tunggu memuat halaman (loading page) akan semakin lama. Hal ini biasa terjadi karena pengembang WebApp tersebut belum melakukan analisis terhadap performa aplikasi menggunakan berbagai macam stress tools. Pengguna ataupun pengembang dapat melakukan tes terhadap WebApp melalui berbagai stress tools, salah satunya adalah Gtmetrix. Pada pengujian alpha beta dengan pengetesan GTmetrix, WebApp SIMLSP mendapatkan grade D dengan persentase 68% performance dan 72% structure pada halaman login, grade F dengan persentase 24% performance dan 72% structure pada halaman daftar akun asesor, serta analisis eror pada halaman penjadwalan ujian asesi. Optimasi yang dilakukan adalah penghapusan application code yang tidak diperlukan, penambahan atribut disabled pada tag link styleesheet, pemindahan local access pada CSS/JavaScript, pemanfaatan parameter pada fungsi JavaScript, dan paginasi. Setelah melakukan optimasi tersebut dilakukan pengujian alpha beta dengan pengetesan kembali didapatkan peningkatan grade B dengan persentase performance 84% dan structure 87% pada halaman login, grade B dengan persentase performance 91% dan structure 83% pada halaman daftar akun asesor, dan grade A dengan persentase performance 94% dan structure 81% pada halaman penjadwalan asesi. Dengan hasil tersebut maka optimasi yang dilakukan telah berhasil.

Kata kunci: website application (WebApp), SIMLSP, pengujian alpha beta, cyclomatic complexity, gtmetrix

1. Pendahuluan

Website Application atau biasa disebut WebApp adalah suatu aplikasi yang diakses menggunakan penelusuran web (*browser*) melalui suatu jaringan internet. Pada proses pembuatan WebApp terdapat beberapa tahapan salah satunya adalah implementasi, yaitu ketika WebApp telah siap digunakan oleh

Diterima Redaksi: 25-10-2022 | Selesai Revisi: 17-11-2022 | Diterbitkan Online: 01-12-2022

pengguna. Di tahap ini aplikasi akan mulai digunakan. Jika terdapat kendala akan terlihat pada bagian ini, seperti performa aplikasi menurun ketika diakses oleh sejumlah pengguna, waktu tunggu memuat halaman (*loading page*) akan semakin lama. Hal ini biasa terjadi karena pengembang WebApp tersebut belum melakukan analisis terhadap performa aplikasi menggunakan berbagai macam *stress tools*. Menurut [1], kualitas *web server* dikatakan baik, apabila mampu melayani tiap permintaan (*request*) dari pengguna ke Uniform Resource Locator (URL) secara cepat dan kemampuan untuk mengurangi terjadinya kesalahan (*error*). Faktor kestabilan *web server* sangat penting untuk mempercepat proses *loading time* web. Dapat dikatakan bahwa *website* harus memberikan suatu pengalaman pengguna dalam berinteraksi dengan aplikasi atau situs *website* sampai pengguna dapat mengoperasikannya dengan mudah dan cepat. Pengoperasian yang cepat dan mudah memberikan pengalaman pengguna pada aplikasi menjadi puas.

Pengguna ataupun pengembang dapat melakukan tes terhadap WebApp melalui berbagai *stress tools*, salah satu contohnya adalah Gtmetrix dengan kelebihan tanpa perlu instalasi aplikasi karena berbasis *website* dan gratis tanpa maksimum jumlah *website* yang dites. Pada *tools* tersebut dapat diketahui nilai performa dan struktur yang merupakan bagian dari *usability* WebApp. Contohnya WebApp SIMLSP yang merupakan aplikasi Lembaga Sertifikasi Profesi di Politeknik Negeri Malang untuk proses asesmen Badan Nasional Sertifikasi Profesi (BPNSP) meliputi ujian asesmen, penyusunan dokumen sertifikasi, penjadwalan asesmen, dan laporan asesmen. Pada saat ujian asesmen berlangsung jumlah rata-rata pengguna yang mengakses WebApp tersebut pada sekali asesmennya berkisar 300 pengguna. Dengan jumlah tersebut WebApp mengalami penurunan performa kecepatan memuat halaman (*loading page*). Pada WebApp SIMLSP terlihat permasalahan ada pada halaman login, penjadwalan ujian dan daftar akun asesor. Hal ini dapat disebabkan belum sempurnanya optimasi performa WebApp yang ada. SIMLSP tersebut dapat diakases di alamat http://program.dutatechnology.com/simlsp.

SIMLSP mendapatkan *grade* D dengan rincian 68% *performance* dan 72% *structure* pada saat pengecekan *stress tools* halaman utama (login). Sedangkan pada halaman penjadwalan ujian tidak keluar *grade* karena terlalu lamanya waktu *loading page*. Kemudian pada pengecekan halaman daftar akun asesor mendapatkan *grade* F dengan rincian 24% *performance* dan 72% *structure*. Hal ini berarti *usability* dari WebApp tersebut masih rendah, sehingga memungkinkan lambatnya pengaksesan WebApp tersebut pada saat WebApp diakses oleh banyak orang. Masalah tersebut dapat diatasi dengan teknik optimasi baik dari sisi *server* maupun WebApp. Untuk melakukan optimasi WebApp agar lebih cepat dalam mengakses dapat dilakukan dengan cara konfigurasi *server* tersebut meliputi pembaruan *library* yang ada pada *server* dan aplikasi meliputi penggunaan Image Compression, Browser Cache, Minify CSS, HTML dan Javascript *file*, Anti Render-Blocking Javascript, serta penempatan stylesheet CSS dan Javascript, sehingga mampu meningkatkan kecepatan dan kestabilan akses *website* [1]. Optimasi dilakukan pada sisi *front end* relatif lebih murah dan mudah jika dibanding dengan optimasi pada sisi *server* ataupun jaringan internet. Metodenya adalah dengan meminimalkan jumlah HTTP *request*, meminimalkan ukuran *file* dan membuat halaman *cacheable* [2].

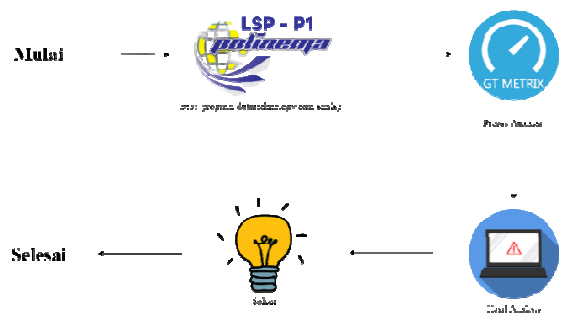
Selain itu pengaruh ukuran gambar juga memberikan dampak pada kecepatan aplikasi. Dikutip dari halaman GTmetrix, gambar yang besar dan tidak fleksibel menyebabkan penurunan pada performa aplikasi. Dikutip dari dokumentasi [3], GTMetrix adalah sebuah *tools* gratis yang menilai kecepatan situs *website*. *Tools* GTMetrix dapat menganalisis *grade* kecepatan halaman situs *website* dan memberi saran bagaimana agar situs *website* menjadi lebih cepat. Untuk melakukan pengetesan dapat dilakukan dengan cara memasukkan *link* halaman yang akan dilakukan pengetesan pada kolom analisis. Kemudian GTMetrix memeriksa total *loading time of page* dan total *page size*. Hasil analisis tersebut berupa *grade* angka dan diikuti persentase dari *performance* dan *structure*. Selain itu GTmetrix juga memberikan permasalahan utama yang menyebabkan aplikasi yang dites mendapatkan hasil *grade* dan persentase. Permasalahan tersebut dipetakan dari dampak paling tinggi sampai tidak berdampak sama sekali, permasalahan tersebut yang dilakukan optimasi agar aplikasi menjadi optimal secara kecepatan. GTMetrix menyediakan struktur secara rinci dari komponen-komponen yang berkontribusi pada ukuran total situs.

2. Metode Penelitian

Optimasi aplikasi menggunakan parameter pengujian *stress tools* pada *website* GTmetrix dengan metode eksperimen. Metode eksperimen merupakan suatu penelitian yang dilakukan dengan manipulasi terhadap beberapa variabel dengan cara tertentu sehingga berpengaruh pada variabel lain yang di ukur [3]. Kompleksitas *query* dapat mempengaruhi pada penambahan beban kinerja CPU, *bandwidth* dan memori, untuk meminimalisir beban tersebut dapat dilakukan perbaikan pada *pagination* dan *query* [4]. Dari kutipan tersebut dapat disimpulkan optimasi *website* dapat dilakukan dengan penambahan *pagination* pada data dengan jumlah baris yang banyak serta merampingkan *query*. Selain itu pengaruh ukuran gambar juga memberikan dampak pada kecepatan aplikasi. Dikutip dari halaman GTmetrix, gambar yang besar dan tidak fleksibel menyebabkan penurunan

DOI: 10.38204/tematik.v9i2.1071

pada performa aplikasi [4]. Langkah-langkah penyelesaian dengan metode tersebut dapat dilihat pada gambar1 .



Gambar 1. Skema Penyelesaian

Sebelum melakukan pengecekan performa di GTmetrix tentukan *website* yang akan dites terlebih dahulu. Lalu masukkan *link* halaman *website* pada kolom analisis. Tunggu beberapa saat maka akan muncul hasil dari pengecekan. Hasil pengecekan berupa persentase dari *performance* dan *structure website* serta analisis permasalahan utama yang mengakibatkan performa menurun. Dengan permasalahan tersebut GTmetrix juga memberikan saran dan solusi perbaikan dari permasalahan utama tersebut untuk menghilangkan/menurunkan tingkat permasalahan utama yang ada. Permasalahan utama tersebut berpengaruh pada hasil pengecekan berupa *grade* dan persentase *performance* dan *structure*. Apabila *grade* yang didapat adalah A atau B maka *website* sudah baik, tidak perlu optimasi lebih lanjut. Apabila mendapat C-F maka *website* harus segera dioptimasi agar pengguna nyaman dalam menggunakan aplikasi. Setelah mendapatkan *grade* dari GTmetrix akan terlihat permasalahan utama dari *website* yang diuji beserta saran perbaikan oleh GTmetrix. Salah satu permasalahan utama yang selalu ada ketika mendapat grade B-F adalah masalah *loading time*.

Berdasarkan pengetesan WebApp SIMLSP yang dilakukan pada *stress tools* GTmetrix didapatkan *grade* D yang berarti membutuhkan perbaikan performa agar nyaman digunakan oleh pengguna. Permasalahan tersebut yaitu waktu respons *server* awal melebihi 600ms; ukuran gambar terlalu besar dan blokir *assets* dari *website*; pemanfaatan *cache* pada *website* belum maksimal; banyak pengalihan halaman.

Dari permasalahan utama yang didapatkan dari pengetesan SIMLSP pada GTmetrix tersebut maka dapat dilakukan optimasi *website* dengan cara sebagai berikut:

Minify HTML, CSS, JS dilakukan untuk penghapusan karakter yang tidak perlu dalam *code* untuk dieksekusi. Minify dilakukan sebagai solusi pengurangan beban memori aplikasi ketika berlangsung. sehingga aplikasi akan lebih cepat diakses.

Image Compression dilakukan untuk menurunkan resolusi dan ukuran gambar pada aplikasi. Dibutuhkan karena *load* gambar memerlukan *source* yang cukup besar, sehingga memengaruhi kecepatan *website* pula.

Paginasi dilakukan untuk membagi data sesuai dengan kebutuhan dan kapasitas pengguna. Paginasi sangat dibutuhkan jika data yang diterima oleh pengguna terlampau banyak. hal ini akan mengakibatkan penurunan performa aplikasi (*loading time*).

Selain poin di atas perlu juga untuk melakukan perbaikan pada masalah teratas dari hasil *testing* GTmetrix minimal dari dampak paling tinggi lalu rendah. Pada dokumentasi [3] diberikan pula beberapa saran yang sesuai dengan hasil analisis performa aplikasi. Seperti contohnya jika waktu respons *server* terlalu lama maka diberikan saran seperti mengoptimalkan *application code* (termasuk *query database*), meningkatkan perangkat keras *server* untuk sumber daya CPU atau memori yang lebih banyak, dan menerapkan *caching* pada sisi *server*.

2.1. Alat Penelitian

Alat yang digunakan dalam penelitian ini terdiri dari GTmetrix dan alat penunjang lainnya seperti laptop dan server. Dalam pengujian alpha spesifikasi *hardware* yang digunakan berspesifikasi sebagai berikut:

Tabel 1. Tabel *Hardware* Pendukung

Alat	Spesifikasi
Laptop	– CPU (AMD 8)
	– Memori (500 GB)
	– RAM (6 GB)
	– OS (Linux Ubuntu 20.04)
Server	– CPU Core 8
	– Memori 16 GB
	– Internet 40 Mbps
	– Linux 20.04

Pada tabel selain digunakan untuk melakukan optimasi beserta *hosting* WebApp SIMLSP, juga untuk melakukan pengujian alpha (hanya laptop).

Spesifikasi *Software*

Tabel 2. Tabel *Software* Pendukung

Alat	Spesifikasi
Visual Studio Code	versi 1.63.2
WebApp GTmetrix	gtmetrix.com
WebApp SIMLSP Politeknik Negeri Malang	http://program.dutatechnology.com:5580/skripsi_asri/simlsp_asri2 (sebagai studi kasus)

Pada tabel 2 ditampilkan *software* pendukung dalam melakukan optimasi aplikasi pada penelitian ini.

DOI: 10.38204/tematik.v9i2.1071

GTmetrix sebagai pengukuran performa aplikasi, SIMLSP sebagai studi kasus yang akan dilakukan penelitian dan VS Code sebagai *text editor*.

Tabel 1. Spesifikasi Internet

Alat	Spesifikasi
Provider	My Republic
Kecepatan	40 Mbps

Koneksi internet yang digunakan dalam melakukan optimasi menggunakan *provider* MyRepublic dengan kecepatan seperti pada tabel 3.

Untuk pengujian beta spesifikasi yang digunakan tergantung dari *hardware* dan jaringan internet pengguna. Oleh karena itu diperlukan pendataan spesifikasi pengguna meliput: *provider* internet yang digunakan; kecepatan internet, menggunakan https://fast.com/id/ untuk mengetahui kecepatan internet; komputer meliputi CPU, memori, RAM, dan OS yang digunakan.

2.2. Bahan Penelitian

Bahan yang digunakan dalam penelitian ini yaitu halaman aplikasi yang diujikan. Berikut merupakan daftar halaman aplikasi yang digunakan sebagai objek dalam penelitian.

Tabel 2. Spesifikasi *Software* Uji

Halaman yang akan dioptimasi	Grade yang didapat dari GTMetrix	Persentase
Halaman login	D	68% performance dan 72% structure
Penjadwalan ujian asesi	Tidak bisa dilakukan pengecekan karena loading time terlalu lama.	
Daftar akun asesor	F	24% performance dan 72% structure

Pada tabel 4 diberikan daftar bahan penelitian untuk optimasi yang telah dilakukan pengetesan pada GTmetriks. Bahan yang akan dilakukan optimasi yaitu halaman utama SIMLSP (login), halaman jadwal ujian dan halaman akun asesor.

2.3. Parameter Pengujian

Parameter pengujian dalam penelitian ini adalah *grade* yang didapatkan dari pengujian *website* pada GTmetrix dan pengujian alpha beta dari pengguna WebApp SIMLSP. Setelah melakukan optimasi berupa perbaikan yang dianjurkan oleh GTmetrix website akan dites kembali. Serta mengambil data dari hasil pengujian alpha beta sebelum dan sesudah optimasai dilakukan. Pengujian alpha dilakukan untuk melihat apakah semua sistem dapat berjalan dengan baik dan dilakukan oleh pembuat sistem atau yang terlibat dalam pembuatan sistem sedangkan pengujian beta digunakan untuk melakukan evaluasi terhadap sistem yang telah dibuat, pihak yang akan melakukan penilaian sistem adalah pengguna atau orang-orang yang tidak terlibat dalam pembuatan sistem [5]. Tujuan dari beta *testing* dapat beraneka ragam, seperti memberikan kesempatan kepada media pers untuk menuliskan tinjauan awal dari software untuk mendapatkan masukan mengenai user interface sebagai usaha terakhir untuk menghilangkan bugs [6]. Hasil tersebut berupa perbandingan kenaikan/penurunan yang didapatkan pada pengetesan GTmetrix.

2.4. Indikator Keberhasilan

Indikator keberhasilan dari penelitian ini adalah terdapat peningkatan pada *grade* yang didapatkan oleh website ketika dilakukan pengetesan performa di GTmetrix pada pengujian alpha beta. Perubahan persentase *performance* dan *structure* akan dibandingkan dari hasil persentase sebelum melakukan optimasi dan setelah dilakukannya optimasi. Selain perbandingan tersebut, perubahan permasalahan utama dari pengetesan yang dilakukan akan dijadikan sebagai indikator keberhasilan dari penelitian ini. Ketika permasalahan utama dengan dampak paling tinggi menghilang/menurun maka optimasi yang dilakukan tepat sasaran dan berhasil.

3. Hasil dan Pembahasan

Optimasi WebApp SIMLSP dilakukan dengan perangkat yang spesifikasinya telah disebutkan pada poin sebelumnya dibagi berdasarkan halaman yang dioptimasi yaitu halaman login, daftar asesor, dan penjadwalan ujian asesi. Hasil optimasi ditampilan dalam bentuk tabel, gambar, dan *application code* untuk mengetahui hasil optimasi yang telah dilakukan.

3.1. Halaman Login

Optimasi yang dilakukan pada halaman login adalah sebagai berikut:

Waktu respons awal aplikasi : optimasi yang dilakukan adalah menghapus *application code* yang tidak digunakan, penerapan *caching* agar tidak selalu mengunduh jika pernah dimuat sebelumnya, dan peningkatan memori *server* yang digunakan (bila perlu).

Tabel 3. Perbedaan *Application Code* Sesudah dan Sebelum Optimasi Halaman Login

Sebelum Optimasi

```
$data['features'] = $this->config->item
('features');

if (!$data['features']
['disable_all_social_logins']) {
    $this->social_login_init();

    $sociallogin = $this->social_login(); //
    Return facebook, twitter and google login
    urls array from main controller

    if ($data['features']
    ['login_via_facebook']) {
        $data['facebook'] = $sociallogin
        ['facebook']; // facebook login is
```

Nilai *cyclomatic complexcity* = 5

Setelah Optimasi

DOI: 10.38204/tematik.v9i2.1071

```
{
    if ($this->ion_auth->logged_in()) {
        redirect(last_url());
    }

    $this->load->model('logs');

    if ($this->input->post('identity') ||
    $identity != null) {
        if ($identity == null) {
            $identity = $this->input->post
            ('identity');
            $password = $this->input->post
            ('password');
            $remember = (bool) $this->input->post
            ('remember');
```

Nilai *cyclomatic complexcity* = 4

Pada halaman login sebelum optimasi terdapat *application code* yang tidak diperlukan yaitu $data['features'] = $this->config->item('features'); yang dicetak tebal dinyatakan tidak diperlukan karena pada WebApp SIMLSP tidak menggunakan akun sosial media untuk melakukan login. Namun yang dihapus hanya pada *frontend*-nya saja, di bagian *backend* masih terdapat *application code* tersebut. Permasalahan ini pada pengetesan GTmetrix memberikan dampak yang tinggi pada saat aplikasi pertama kali dimuat. Setelah dilakukan optimasi, penghapusan *application code* yang tidak perlu seperti pada *application code* diatas, permasalahan utama waktu respons awal *server* menghilang. Serta nilai *cyclomatic complexcity* mendapatkan penurunan yaitu dari 5 menjadi 4.

Kesimpulan: permasalahan waktu respons awal *server* pada WebApp SIMLSP dipicu oleh adanya *application code* yang tidak digunakan namun tidak dihapus.

Eliminate rander-blocking Resources

Optimasi yang dilakukan pada permasalahan ini adalah dengan memberikan atribut atribut defer atau async pada tag <script> yang ada pada <head> dokumen *apllication code*, memberikan atribut disabled atau media pada tag <link rel=”styleesheet”>, dan memberikan atribut async pada tag <link rel=”import”>. Hal ini bertujuan agar *file* yang dihubungkan tidak selalu terpancar/terpancar hanya jika aplikasi sedang berjalan. Tabel 6 menyajikan perbedaan *application code* sebelum dan sesudah optimasi.

Tabel 4. Perbedaan Tampilan Sesudah dan Sebelum Optimasi Halaman Login

Sebelum Optimasi

```
<link                href="<?php echo base_url
("assets/admin/");?>css/style.default.css"
rel="stylesheet">
```

Nilai *cyclomatic complexcity* = 5

Setelah Optimasi

```
<link id="first_style" href="<?php echo base_url
("assets/admin/");?>css/style.default.css"
rel="stylesheet" disabled>
<script type="text/javascript">
        window.onload = function()
        {
            document.getElementById('first_style')
            removeAttribute('disabled');
        }
    </script>
```

Nilai *cyclomatic complexcity* = 4

Pada WebApp SIMLSP menggunakan <link rel=”styleesheet”>, sehingga pada *application code* dilakukan optimasi dengan pemberian atribut disabled. Ketika optimasi dilakukan dengan pemberian atribut disabled, tampilan *frontend* menjadi hancur karena disabled digunakan untuk mematikan CSS pada tag <link rel=”styleesheet”>. Kemudian diberikan id dan tambahan *script* untuk mengatur CSS tersebut menggunakan perintah onLoad JavaScript. Sehingga CSS tersebut hanya akan berjalan ketika *object* tersebut dipanggil.

Kesimpulan: permasalahan *render-blocking* dapat diselesaikan dengan menambahkan atribut yang disarankan oleh GTmetrix.

Ukuran gambar yang sesuai : Optimasi yang dilakukan pada permasalahan ini adalah dengan melakukan *compress* gambar. Tabel 7 menyajikan informasi sebelum dan sesudah optimasi gambar.

Tabel 5. Perbandingan Application Code Sebelum dan Sesudah Optimasi Ukuran Gambar

Sebelum Optimasi		
Nama Gambar	Ekstensi	Ukuran
MyIgniter	PNG	390 KB
id-card	SVG	70 KB
Sesudah Optimasi		
Nama Gambar	Ekstensi	Ukuran
MyIgniter	PNG	78 KB
id-card	SVG	3KB

Pada pelaksanaan optimasi bagian ini dilakukan dengan *image compress*, sebelum dilakukan optimasi, ukuran gambar ada 390 KB dengan ekstensi PNG. *Image compress* yang dilakukan hanya dilakukan sampai 78 KB saja karena dibawah ukuran tersebut kualitas gambar menjadi sangat buruk terutama warna. Dengan ukuran 78 KB pengetesan yang dilakukan pada GTmeetrix tidak menunjukkan keberhasilan. Kemudian dilakukan perubahan gambar dengan ekstensi SVG ukuran 70 KB dan dilakukan proses *image compress* hingga pada ukuran 3 KB tidak didapati perubahan kualitas gambar. Sehingga penggunaan logo diganti menjadi SVG dan permasalahan utama ukuran gambar menjadi hilang.

Kesimpulan: pemakaian logo berupa gambar hendaknya memakai gambar berekstensi SVG karena ditulis dalam markup berbasis XML, mirip seperti HTML. Serta SVG tidak bergantung pada pixel karena

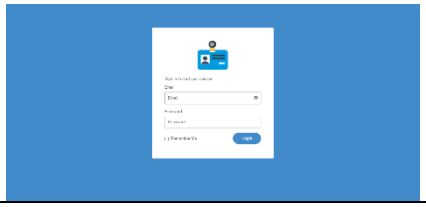
DOI: 10.38204/tematik.v9i2.1071

SVG diambil dari bentuk dan kurva matematis yang dideklarasikan atau semacamnya dan terlihat sama, tidak peduli ukuran layar dan tingkat perbesaran layar dan membuatnya bagus untuk desain responsif, dan SVG dapat dimodifikasi menggunakan CSS dan JavaScript [7]. Ketika melakukan compress ukuran gambar, SVG tidak menunjukkan hilangnya kualitas gambar, sedangkan pada PNG semakin dilakukan pengurangan ukuran gambar maka akan semakin berkurang pula kualitasnya. Sehingga dianjurkan untuk memakai gambar dengan ekstensi SVG.

Tabel 8 menyajikan perbedaan tampilan halaman login sebelum dan sesudah dilakukan optimasi.

Tabel 6. Perbedaan Tampilan Sesudah dan Sebelum Optimasi Halaman Login

Sebelum Optimasi

Setelah Optimasi


Logo yang digunakan berganti menjadi SVG dan informasi nomor admin dipindah, tidak lagi pada halaman login. Dengan optimasi yang dilakukan, *grade* yang didapatkan menjadi B dengan persentase 84% *performance* dan 87% *structure*. Hasil tersebut disajikan dalam gambar 2.

Gambar 1. Hasil Optimasi Halaman Login WebApp SIMLSP

Dengan demikian, optimasi yang dilakukan dan permasalahan utama yang ada pada WebApp SIMLSP telah berhasil dilakukan dengan teknik yang disarankan oleh GTmetrix. Sedangkan permasalahan utama disajikan pada gambar 3.

Gambar 2. Hasil Permasalahan Utama Halaman Login

Permasalahan utama dengan dampak tertinggi telah diatasi dan hilang, sehingga aplikasi telah berhasil dioptimasi.

3.2. Halaman Daftar Akun Asesor

Optimasi yang dilakukan pada halaman daftar akun asesor adalah melakukan pengecekan *render-blocking* seperti pada halaman login, *main-thread* yang terlalu panjang pada *assets* CSS dan JS, dan *loading time* JavaScript. Berikut perbedaan sebelum optimasi dan sesudah optimasi dilakukan:

Eliminate rander-blocking resources : optimasi yang dilakukan pada permasalahan ini adalah dengan memberikan atribut atribut defer atau async pada tag <script> yang ada pada <head> dokumen apllication code, memberikan atribut disabled atau media pada tag <link rel=”styleesheet”>, dan memberikan atribut async pada tag <link rel=”import”>. Hal ini bertujuan agar *file* yang dihubungkan tidak selalu terpancar/terpancar hanya jika aplikasi sedang berjalan. Tabel 9 perbedaan application code sebelum dan sesudah optimasi:

Tabel 7. Perbedaan *Application Code* Sesudah dan Sebelum Optimasi Halaman Login

Sebelum Optimasi

```
$this->layout->set_wrapper('grocery', $data,
'page', false);
// $this->layout->auth();

$template_data['grocery_css'] = $data
['css_files'];
$template_data['grocery_js'] = $data
['js_files'];

$template_data['title'] = 'Asesor';
$template_data['crumb'] = [
    'Users' => '',
];
```

Nilai *cyclomatic complexcity* = 5

Setelah Optimasi

```
$this->layout->set_wrapper('grocery', $data,
'page', false);
// $this->layout->auth();

$this->layout->setCacheAssets();

$this->layout->render('admin', $template_data);
```

Nilai *cyclomatic complexcity* = 3

Pada WebApp SIMLSP halaman daftar akun asesor menggunakan <link rel=”styleesheet”>, sehingga pada *application code* dilakukan optimasi dengan

DOI: 10.38204/tematik.v9i2.1071

pemberian atribut disabled. Ketika optimasi dilakukan dengan pemberian atribut disabled, tampilan frontend menjadi hancur karena disabled digunakan untuk mematikan CSS pada tag <link rel=”styleesheet”>. Kemudian diberikan id dan tambahan script untuk mengatur CSS tersebut menggunakan onLoad JavaScript. Sehingga CSS tersebut hanya akan berjalan ketika object tersebut dipanggil.

Kesimpulan: permasalahan render-blocking dapat diselesaikan dengan menambahkan atribut yang disarankan oleh GTmetrix.

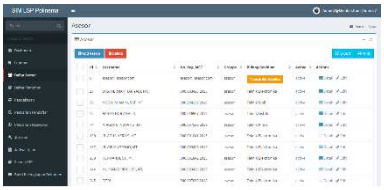
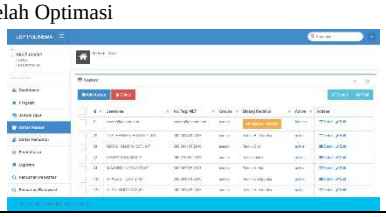
Main-thread yang panjang pada assets CSS dan JS dan loading time JavaScript : Optimasi yang dilakukan pada permasalahan ini adalah penghapusan script CSS dan JavaScript yang tidak digunakan namun tertulis pada *application code*. Tabel 10 menyajikan perbedaan sebelum dan sesudah melakukan optimasi.

Tabel 8. Perbedaan *Application Code* Sesudah dan Sebelum Optimasi Halaman Daftar Akun Asesor

Sebelum Optimasi `<link href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,600,700" rel="stylesheet">` Nilai *cyclomatic complexcity* = 5
Setelah Optimasi `<link href="<?php echo base_url("assets/css/font.css");?>" rel="stylesheet">` Nilai *cyclomatic complexcity* = 3

Kesimpulan: ketika melakukan pembuatan website hendaknya *application code* yang tidak di-*render* dihapus agar tidak memberatkan aplikasi ketika diakses. Serta memberikan atribut tambahan pada tag <link rel=”styleesheet”> yang diperlukan WebApp agar tidak mendapatkan *render-blocking*.
Berikut disajikan tabel perbedaan tampilan halaman daftar akun asesor sebelum dan sesudah dilakukan optimasi.

Tabel 9. Perbedaan Tampilan Sesudah dan Sebelum Optimasi Halaman Daftar Akun Asesor

Sebelum Optimasi 
Setelah Optimasi 

Dengan tampilan dan optimasi yang dilakukan, *grade* yang didapatkan menjadi B dengan persentase 91% *performance* dan 83% *structure*. Hasil tersebut disajikan dalam gambar 4..

Gambar 3. Hasil Optimasi Halaman Daftar Akun Asesor WebApp SIMLSP

Dengan demikian, optimasi yang dilakukan dan permasalahan utama yang ada pada WebApp SIMLSP telah berhasil dilakukan dengan teknik yang disarankan oleh GTmetrix. Sedangkan pada permasalahan utama disajikan pada gambar 5.

Gambar 4. Hasil Permasalahan Utama Halaman Daftar Akun Asesor

Permasalahan utama dengan dampak tertinggi telah diatasi dan mengalami penurunan tingkat dampak dari high menjadi *medium-high*. Sebelum dilakukan optimasi potential saving adalah 1.1s setelah dilakukan optimasi menjadi 0.5s, sehingga aplikasi telah berhasil dioptimasi.

3.3. Halaman Penjadwalan Ujian Asesi

Pengetesan yang dilakukan menggunakan *stress tools* GTmetrix menunjukkan analisis *ereor* dan tidak mengeluarkan hasil apapun untuk melakukan optimasi. Sehingga optimasi yang dilakukan pada halaman penjadwalan ujian asesi, dilakukan seperti pada halaman login dan daftar akun asesor, yaitu sebagai berikut:

Waktu respons awal aplikasi : Optimasi yang dilakukan adalah menghapus *application code* yang tidak digunakan serta memodifikasi pada *application code* apabila berpicu pada kelambatan akses aplikasi, penerapan *caching* agar tidak selalu mengunduh jika pernah dimuat sebelumnya, dan peningkatan memori server yang digunakan. Tabel 12 menyajikan perbedaan *application code* sebelum dan sesudah optimasi.

DOI: 10.38204/tematik.v9i2.1071

Tabel 10. Perbedaan Tampilan Sesudah dan Sebelum Optimasi Halaman Penjadwalan Ujian Asesi

Sebelum Optimasi

```
function ijadwal()
{
    $t = $this->db->query("SELECT*FROM jadwal_ujian
    if ($t->num_rows() == 0) {...
    } else {
        foreach ($t->result() as $tmjd) {...
        }
        $this->load->view('jadwal', $tampil);
    }
}

public function jadwal()
{
    $template = "admin";
    $template_data["title"] = "Jadwal Ujian
    <div style='height:10px;'></div>
<iframe src='" . site_url("jadwal_ujian/ijadwal/0")
    $template_data["crumb"] = ["page" => ""];

    // $this->load->view("smp_cover");

    $this->layout->render($template, $template_data)
}
```

Nilai *cyclomatic complexcity* = 7

Setelah Optimasi

```
public function jadwal_v2()
{

    // $this->layout->auth(); // Login required

    $data['helloWorld'] = 'Hello World'; // Parsing

    $template_data['title'] = 'Hellow World'; // Dat
    $template_data['crumb'] = [
        'Hellow World' => '', // Breadcrumb menu
    ];

    $this->layout->set_title($this->title); // Set t
    $this->layout->set_wrapper('jadwal_v2', $data);

    $this->layout->setCacheAssets(); // Cache assets

    $this->layout->render('admin', $template_data);
}
```

Nilai *cyclomatic complexcity* = 1

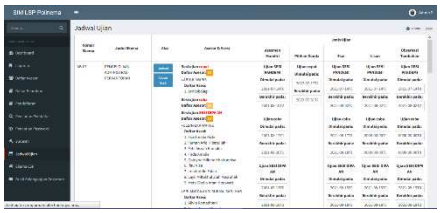
Pada *application code* sebelum optimasi terlihat menggunakan iframe dalam menampilkan halaman jadwal. Pada penggunaan iframe ini kurang tepat karena iframe menampilkan banyak data dari *database*. Sebelum optimasi semua data nomor skema beserta jadwal asesi dan asesor ditampilkan dalam 1 halaman iframe. Sedangkan pada penjadwalan masing-masing nomor skema setidaknya memiliki 3 sesi ujian yang berbeda, kemudian pada masing-masing sesi setidaknya terdapat 4 jadwal asesor dan 6-15 asesi. Dengan data yang cukup banyak tersebut harus di-*load* menggunakan iframe. Hal ini membebani *server* karena aplikasi melakukan dua kali loading time, untuk *load* pertama kali dan *load file* yang berisi data sebenarnya. Kemudian setelah melakukan optimasi dilakukan perubahan dengan *load view* yang dicetak tebal pada Tabel 10 diatas. Dengan begitu *loading time* pada halaman penjadwalan sedikit lebih ringan dibanding menggunakan iframe. Untuk melanjutkan optimasi ini diperlukan optimasi pada paginasi untuk meringankan *loading time*.

Kesimpulan: penggunaan iframe lebih baik untuk menampilkan dokumen kecil agar tidak membebani server.

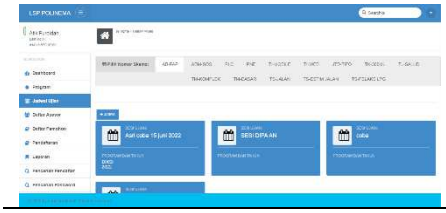
Membuat paginasi : Teknik optimasi paginasi menggunakan *limit query* agar data yang diambil dari *database* tidak brutal, diambil sesuai kebutuhan yang akan ditampilkan. Berikut hasil optimasi yang dilakukan:

Tabel 11. Perbedaan Tampilan Sebelum dan Sesudah Optimasi Pada Halaman Penjadwalan Ujian Asesi

Sebelum Optimasi



Setelah Optimasi



Pada Tabel 11 menunjukkan adanya perbedaan tampilan sebelum optimasi semua data nomor skema beserta jadwal asesi dan asesor. Dengan jumlah data yang telah diuraikan pada optimasi *loading time*, diperlukan optimasi lanjutan yaitu paginasi. Optimasi yang dilakukan adalah memberikan paginasi dengan bantuan nomor skema. Kemudian dilakukan paginasi lagi pada sesi ujian pada masing-masing nomor skema. Hal ini sangat berpengaruh pada optimasi karena data yang diambil tidak langsung semua, mengambil dan menampilkan yang diperlukan.
Kesimpulan: apabila pada WebApp memungkinkan untuk menampilkan data yang banyak, paginasi adalah cara yang tepat untuk meringankan dan mempercepat akses WebApp.

Mengurangi perulangan JavaScript : Pembuatan halaman penjadwalan memanfaatkan perulangan JavaScript dengan jumlah data yang tidak sedikit. Hal ini dapat juga memicu lambatnya aplikasi ketika diakses. Berikut hasil optimasi yang dilakukan:

Tabel 12. Perbedaan *Application Code* Sebelum dan Sesudah Optimasi Halaman Penjadwalan Ujian Asesi

Sebelum Optimasi

```
<?php
for(x=1; x < count(); x++){
?>
  <input onchange=lalal<?php
echo x;?>() ...>
...
  function lalal<?php echo
x;?>(){
...
}
<?php
}
?>
```

Nilai *cyclomatic complexcity* = 5

DOI: 10.38204/tematik.v9i2.1071

Setelah Optimasi

```
<?php
for(x=1; x < count(); x++){
?>
  <input onchange=lalal(<?php
echo x;?>) ...>
...
<?php
}
?>
function lalal(<?php echo
x;?>){
...
}
```

Nilai *cyclomatic complexcity* = 2

Pada halaman Simlp edit jadwal ujian dibangun menggunakan onChange. Sebelum dilakukan optimasi perintah onChange akan dilakukan perulangan sejumlah sesi yang telah dibuat. Perulangan ini dilakukan dengan perbedaan nama id dan nama fuction. Sehingga ketika halaman ini dipanggil akan mencetak fungsi JavaScript sejumlah sesi yang dibuat. Sedangkan pada 1 sesi ujian setidaknya terdapat 3 jenis ujian dan edit dilakukan 2 kali yaitu tanggal mulai dan tanggal selesai. Dengan kondisi tersebut JavaScript berpengaruh dengan lambatnya akses halaman penjadwalan ujian. Sehingga dilakukan optimasi JavaScript dengan memanfaatkan parameter fungsi seperti pada Tabel 14 yang dicetak tebal. Pemanfaatan parameter ini terpengaruh besar karena ketika halaman dipanggil fungsi javascript hanya satu kali dicetak.
Kesimpulan: pemanfaatan parameter dalam penggunaan fungsi JavaScript berguna untuk mempercepat halaman ketika diakses.

Dengan optimasi yang dilakukan, grade yang didapatkan menjadi A dengan persentase 94% performance dan 81% structure. Hasil tersebut disajikan pada Gambar 6.

Gambar 5. Hasil Optimasi Halaman Penjadwalan Ujian Asesi WebApp SIMLSP

Dengan demikian, optimasi yang dilakukan dan permasalahan utama yang ada pada WebApp SIMLSP telah berhasil dilakukan dengan teknik yang disarankan oleh GTmetrix. Sedangkan pada permasalahan utama disajikan pada Gambar 7.

Gambar 6. Hasil Permasalahan Utama Halman Penjadwalan Ujian Asesi

Permasalahan utama dengan dampak tertinggi tidak ada dan website dapat dites tanpa hasil analisis eror, sehingga aplikasi telah berhasil dioptimasi.

3.4 Analisis Pengujian

Dengan hasil pengujian alpha dan beta yang telah dilakukan pada poin diatas, maka dibandingkan pula hasil pengujian alpha dengan hasil rata-rata dari pengujian beta. Berikut Tabel 15 menunjukkan hasil perbedaan dari pengujian yang telah dilakukan:

Tabel 13. Hasil Perbandingan Pengujian Alpha dan Beta Pada Masing-masing Halaman

Skenario	Pengujian Alpha		Pengujian Beta		Hasil
	Sebelum	Sesudah	Sebelum	Sesudah	
Halaman Login					
Grade	D	B	C	B	√
%	68%	84%	72%	81%	√
Perform					
%	72%	87%	79%	86%	√
Structure					
Halaman Daftar Akun Asesor					
Grade	F	B	C	B	√
%	24%	91%	71%	82%	√
Perform					
%	72%	83%	77%	84%	√
Structure					
Halaman Penjadwalan Ujian Asesi					
Grade	err	A	err	B	√
%	err	94%	err	85%	√
Perform					
%	err	81%	err	86%	√
Structure					

Dari hasil perbandingan hasil uji alpha dan beta tersebut dapat disimpulkan bahwa pada pengujian alpha maupun beta telah mengalami peningkatan performa yang berpengaruh terhadap kecepatan aplikasi ketika diakses oleh pengguna. Dengan begitu optimasi yang dilakukan telah berhasil memenuhi tujuan dari penelitian ini.

Peningkatan tersebut terlihat pada kondisi sebelum optimasi pada pengujian alpha halaman login mendapat grade D, setelah melakukan optimasi grade meningat menjadi B dengan prosentase kenaikan sebesar 16% untuk performance dan 15% untuk structure. Pada hasil rata-rata pengujian beta halaman login mendapatkan grade C sebelum melakukan optimasi, setelah dilakukan optimasi mendapatkan

DOI: 10.38204/tematik.v9i2.1071

grade B dengan prosentase kenaikan sebesar 9% untuk performance dan 7% untuk structure.

Begitu pula dengan halaman daftar akun asesor, peningkatan terlihat pada kondisi sebelum optimasi pada pengujian alpha halaman login mendapat grade F, setelah melakukan optimasi grade meningat menjadi B dengan prosentase kenaikan sebesar 67% untuk performance dan 11% untuk structure. Pada hasil rata-rata pengujian beta halaman daftar akun asesor mendapatkan grade C sebelum melakukan optimasi, setelah dilakukan optimasi mendapatkan grade B dengan prosentase kenaikan sebesar 11% untuk performance dan 7% untuk structure.

Sedangkan pada halaman penjadwalan peningkatan terlihat pada kondisi sebelum optimasi pada pengujian alpha halaman penjadwalan tidak dapat grade karena analisis eror, setelah melakukan oprimasi halaman penjadwalan ujian asesi mendapatkan grade A dengan persentase 94% untuk performance dan 81% untuk structure. Pada hasil rata-rata pengujian beta halaman penjadwalan mendapatkan grade B dengan persentase 85% untuk performance dan 86% untuk structure setelah melakukan optimasi.

4. Kesimpulan

Berdasarkan pengujian yang dilakukan optimasi telah memenuhi tujuan awal penelitian yaitu meningkatkan performa WebApp SIMLSP menggunakan stress tools GTmetrix adalah pada halaman login menunjukkan peningkatan grade 2 tingkat dari D menjadi B, persentase performance mengalami peningkatan sebanyak 16% dari 68% menjadi 84%, serta persentase structure mengalami peningkatan sebanyak 15% dari 72% menjadi 87%. Peningkatan tersebut diperoleh setelah melakukan optimasi berupa penghapusan application code yang tidak ditampilkan, penambahan atribut disabled pada load css, serta penggantian ekstensi gambar dari PNG menjadi SVG.

Halaman daftar akun asesor menunjukkan peningkatan grade 4 tingkat dari F menjadi B, persentase performance mengalami peningkatan sebanyak 67% dari 24% menjadi 91%, serta persentase structure mengalami peningkatan sebanyak 11% dari 72% menjadi 83%. Peningkatan tersebut diperoleh setelah melakukan optimasi berupa penghapusan script CSS dan JS yang tidak digunakan/dipanggil, penambahan atribut disabled pada load CSS, serta mengganti CSS/JavaScript menjadi local access.

Halaman penjadwalan ujian asesi menunjukkan peningkatan, yaitu semula tidak dapat dilakukan pengetesan performa dengan hasil analisis eror menjadi dapat dilakukan pengetesan dengan hasil grade A, persentase performance mengalami peningkatan dari analisis eror menjadi 94%, serta persentase structure mengalami peningkatan dari analisis eror menjadi 81%. Peningkatan tersebut diperoleh setelah melakukan optimasi berupa penambahan paginasi, mengurangi perulangan fungsi javascript dengan memanfaatkan parameter.

Berdasarkan penelitian yang telah dilakukan, stress tools GTmetrix memiliki kekurangan yaitu tidak dapat memberikan dokumentasi optimasi dengan rinci terkait permasalahan waktu respons awal server beserta solusinya. Sehingga pada penelitian lanjutan dapat menggunakan stress tools yang lain.

Ucapan Terimakasih

Terima kasih saya ucapkan kepada STMIK PPKIA Pradnya Paramita yang telah memberikan kesempatan dalam prosiding ini. Serta kepada Bu Dian dan Pak Yunus yang telah membimbing dalam penyusunan penelitian ini.

Daftar Rujukan

[1] I. A. A. Amrullah, "Review Teknik Optimasi Performa pada Front End Website," Universitas Muhammadiyah Malang, Malang, 2021.

[2] Y. . T. Arumoadi, W. Laksito Y.S. and T. Susyanto, "OPTIMASI KINERJA MOBILE WEBSITE DENGAN TEKNIK FRONT- END OPTIMIZATION PADA TOKO ONLINE IMPERIAL PARFUM," *TIKomSiN*, p. 53, 2019.

[3] Badan Pengembangan dan Pembinaan Bahasa (Pusat Bah, "Kamus Besar Bahasa Indonesia (KBBI)," Pusat Bahasa, 11 November 2021. [Online]. Available: https://kbbi.web.id/. [Accessed 11 November 2021].

[4] E. N. Delta and A. , "Performance Test Dan Stress Website Menggunakan Open Source Tools," *Jurnal Manajemen Informatika*, pp. 208-2015, 2017.

[5] A. Y. Fadli, "Optimasi Kecepatan Loading Time Web Template Dengan Implementasi Teknik Front-End," University of Muhammadiyah, Malang, 2019.

[6] GTmetrix, "GTmetrix," Google, 18 Desember 2021. [Online]. Available: https://gtmetrix.com/. [Accessed 18 Desember 2021].

[7] L. G. S. Kartika and K. Rinartha, "Pengaruh Pagination dan Kompleksitas Query Data Terhadap Aspek Kehijauan dari Sistem Informasi Manajemen," *Jurnal Sistem dan Informatika (JSI)*, vol. Vol 13 No 2 , p. 155, 2019.

[8] H. Kurniawan and E. P. Widiyanto, "Analisis Peningkatan Performa Akses Website dengan Web Server Stress Tool," *JATISI*, vol. Vol 2 No 2 , p. 108, 2016.

[9] T. Limbong and J. Simarmata, Media dan Multimedia Pembelajaran: Teori & Praktik, Medan: Yayasan Kita Menulis, 2020.

[10] S. Masripah and L. Ramayanti, "Penerapan Pengujian Alpha Dan Beta," *JURNAL SWABUMI*, p. 100, 2020.

[11] E. Nugroho, Prinsip-prinsip menyusun kuesioner, Malang: UB Press, 2018.

[12] S. Tjandra and . C. P. , "APLIKASI METODE-METODE SOFTWARE TESTING PADA CONFIGURATION, COMPATIBILITY DAN USABILITY PERANGKAT LUNAK," *IDeaTech*, p. 371, 2015.

[13] F. Yudha and A. M. Panji, "PERANCANGAN APLIKASI

DOI: 10.38204/tematik.v9i2.1071

PENGUJIAN CELAH KEAMANAN PADA APLIKASI BERBASIS WEB," *CyberSecurity dan Forensik*, pp. 1-6, 2018.

[14] Suliman, "Analisis Performa Website Universitas Teuku Umar Dan Universitas Samudera Menggunakan Pingdom Tools Dan Gtmetrix," *Jurnal Sistem Informasi dan Sistem Komputer*, vol. 5, pp. 24-32, Januari 2020.

[15] H. Mokhtari, M. K. Saberi, M. R. Amiri, H. Vakilimofrad and Z. Moradi, "Evaluating the Speed and Performance of the Websites of Hospitals and Specialty and Super-specialty Clinics of Hamadan University of Medical Sciences by GTmetrix," *Informology*, vol. 1, pp. 57-66, 2022.

[16] K. Anam, F. A. Haq, S. Fajrin and F. R. Rifai, "Performance Testing and Stress Levels on the Heymale.Id . Website Using Autosaved Software Testing GTmetrix and K6," *Journal of information system and information technology*, vol. 1, pp. 1-12, 2022.

DOI: 10.38204/tematik.v9i2.1071