SERVER-SIDE PROCESSING TECHNIQUES FOR OPTIMIZING THE SPEED OF PRESENTING BIG DATA

Heni Sulastri[1]; Alam Rahmatulloh[2]; Deri Kurnia Hidayat[3]

[1,2,3]Program Studi Informatika, Fakultas Teknik, Universitas Siliwangi
http://if.unsil.ac.id
[1]henisulastri@unsil.ac.id, [2]alam@unsil.ac.id, [3]deri.kurniahidayat@gmail.com

Abstract—*Big data is the latest industry keyword to describe large volumes of structured and unstructured data that are difficult to process and analyze. Most organizations are looking for the best approach to managing and analyzing large volumes of data, especially in decision making. Large data causes the process of presenting information to be slow because all the large amounts of data must be displayed so that specific techniques are needed so that the presentation of information remains fast even though the data is already large. The website generally processes requests to the server, and then if the required data is available, the server will send all the data. This causes all processes to be based on the client-side. So that the client load becomes heavy in displaying all the data. In this study, server-side processing techniques will be applied so that all processes will be handled by the server and the data sent is not all direct but based on periodic requests from the client. The results of this study indicate the use of server-side processing techniques is more optimal. Based on the results of testing the data presentation speed comparison with server-side processing techniques 98.6% is better than client-side processing.*

Keywords*: Big Data, Client-side Processing, Data Tables, Server-side Processing*

Intisari—Data besar merupakan kata kunci industri terbaru untuk menggambarkan volume besar data terstruktur dan tidak terstruktur yang sulit untuk diproses dan dianalisis. Sebagian besar organisasi mencari pendekatan terbaik untuk mengelola dan menganalisa volume data yang besar terutama dalam pengambilan keputusan. Data besar menyebabkan proses penyajian informasi menjadi lambat dikarenakan semua data yang banyak harus ditampilkan, sehingga diperlukan teknik tertentu agar penyajian informasi tetap cepat walaupun data sudah banyak. Website pada umumnya melakukan proses request kepada server, kemudian jika data yang diperlukan tersedia maka server akan mengirimkan semua datanya. Hal tersebut menyebabkan semua proses bertumpu pada sisi client. Sehingga beban client menjadi berat dalam menampilkan semua data tersebut. Pada penelitian ini akan diterapkan teknik server-side processing sehingga semua proses akan diatasi oleh server dan data yang dikirim tidak langsung semuanya, melainkan berdasarkan request secara berkala dari client. Hasil penelitian ini menunjukkan penggunaan teknik server-side processing lebih optimal. Berdasarkan hasil pengujian perbandingan kecepatan penyajian data dengan teknik server-side processing 98,6% lebih unggul daripada client-side processing.

Kata Kunci: **Big Data, Client-side Processing, Data Tables, Server-side Processing**

PENDAHULUAN

Saat ini semenjak maraknya penggunaan slogan "paperless" menjadikan permasalah baru bagi dunia digital. Data digital semakin bertumpuk, hari ke hari database berjumlah ratusan ribuan bahkan jutaan. Para ahli, pakar maupun organisasi mencari alternatif dan pendekatan terbaik untuk mengelola data besar (big data)(Gartner, n.d.). Tren penyajian data yang paling banyak digunakan para developer adalah library datatable. Datatable menggunakan tipe koneksi *half-duplex* yaitu dua arah namum tidak bisa bersamaan (Husen, Rahmatulloh, & Sulastri, 2019). Datatable merupakan library jquery javascript yang bertujuan untuk manipulasi data yang panjang menjadi data pendek, selain itu mempunyai fitur seperti pencarian, halaman, sorting dan tampilan data yang rapi, interaktif dan responsive (SpyMedia, 2019). Namun seiring dengan waktu, datatable menuai kontroversi bagi web developer. Penggunaan datatable menyebabkan kinerja suatu web menjadi lambat, hal ini dikarenakan datatable meload semua record yang ada pada suatu table dalam database. Ketika data masih sedikit tidak ada permasalahan, kinerja penyajian data berjalan normal. Namun ketika data besar hal ini menjadi kendala dan mengurangi kinerja sehingga performa web menjadi (Gope, Schlais, & Lipasti, 2017). Untuk mengatasi permasalahan tersebut, akan dicoba penerapan teknik server-side processing sehingga dengan menitik beratkan semua proses pada sisi server akan membuat

proses load data pada sisi client menjadi ringan dan cepat. Karena dengan teknik ini data besar tidak akan di load semua secara langsung, melainkan dibatasi oleh server berdasarkan dengan request yang dilakukan oleh client. Dengan demikian berapapun jumlah record yang akan disajikan tidak akan menjadi kendala lagi. Maka pada penelitian ini akan dicoba menguji kinerja penggunaan teknik server-side processing pada penyajian big data sehingga diharapkan kinerja web lebih optimal.

BAHAN DAN METODE

A. *Literatur Review*

Penelitian (Burd et al., 2006) membahas metode dan peralatan untuk membuat bahasa perantara atau *file* kode sumber dari sumber daya dari sisi *server* atau halaman web yang dinamis. Kode sumber kemudian dapat dikompilasi menjadi kelas yang dapat dieksekusi yang memungkinkan untuk pembuatan halaman web yang cepat untuk mengontrol objek yang menjalankan fungsi dari sisi *server*, termasuk render dari *client*.

Penelitian (Yannick Saillet, 2013) membahas sebuah metode tentang masukan ke komputer. Masukannya adalah terkait dengan *frame* aplikasi web dari sisi *client browser*. Metode ini mencakup pengkodean kontrol karakteristik dari input atau setidaknya sebagian dari permintaan ke aplikasi dari sisi *server web*.

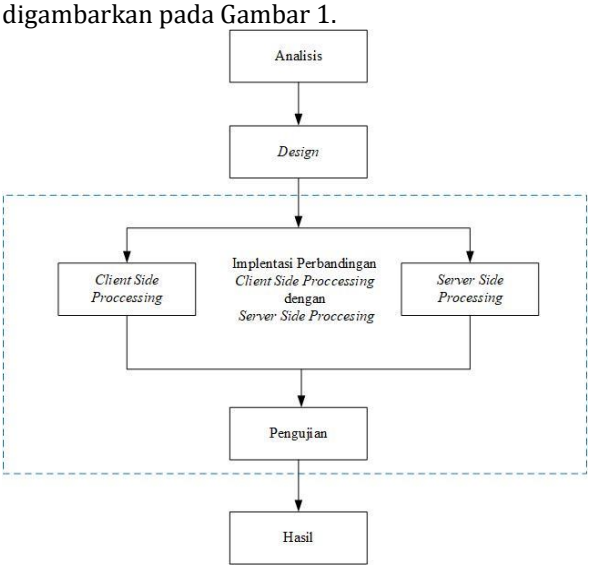
Penelitian (Gope et al., 2017) membahas tentang PHP yang merupakan bahasa skrip dari sisi *server* yang dominan yang digunakan untuk menerapkan konten *web* dinamis. Kompilasi tepat waktu, seperti yang diterapkan di *HipHopVM* mutakhir *Facebook*, untuk membantu mengurangi kinerja PHP yang buruk, tetapi *overhead* yang besar tetap ada, terutama untuk aplikasi PHP berskala besar yang realistis.

Penelitian (Wang, Zender, & Jenks, 2009) membahas tentang teknologi yang terus memungkinkan para ilmuwan untuk mengatur arsip baru dalam pengumpulan dan produksi data, mengintensifkan kebutuhan alat skala besar untuk secara efisien memproses dan menganalisa data besar yang terus meningkat.

Penelitian (Kersten & Goedicke, 2010) membahas Pengujian berbasis browser adalah instrumen yang memadai untuk mendeteksi kesalahan dalam halaman web yang dihasilkan dengan mempertimbangkan proses sisi server dan kompleksitas jalur *black-box.*

B. *Kerangka Peneltian*

Dalam penelitian ini dibuat sebuah kerangka penelitian yang dijadikan pedoman untuk dapat menyelesaikan masalah terkait yang akan diteliti, digambarkan pada Gambar 1.



Sumber: (Sulastri, Rahmatulloh, & Hidayat, 2019)

Gambar 1. Kerangka Penelitian

Gambar 1 merupakan kerangka metode penelitian, tahap pertama yaitu tahap analisis, tahap kedua adalah tahap design, tahap ketiga merupakan tahap implementasi perbandingan *server-side processing* dengan *client-side processing,* tahap selanjutnya adalah tahap pengujian dan tahap terakhir adalah hasil dan simpulan.

HASIL DAN PEMBAHASAN

A. *Analisis*

Dalam penelitian ini data yang diperlukan untuk pengujian data *server-side processing* dan *client-side processing* maka dibutuhkan data besar sebagai pengujian kedua proses tersebut. Big datasets didapatkan dari situs dataset *archive* yaitu https://archive.ics.uci.edu berupa *file* data bank yang berjumlah 45.211 *row* data.

1. Analisis Kebutuhan Fungsional
   Analisis kebutuhan fungsional ini meliputi:
   a) Penyajian data dari *database.*
   b) Menampilkan informasi mengenai ukuran dan kecepatan pemanggilan data

2. Analisis Kebutuhan Non Fungsional

a. Analisis Perangkat Lunak / *Software*

Perangkat lunak yang digunakan untuk penyajian teknik *server-side processing* dan *client-side processing* saat ini adalah sebagai berikut :

1) Sistem Operasi : Microsoft Windows 10 Pro
2) Web Server : Apache 2.4.23
3) *Database* : Mysql 5.0.11
4) *Web browser* : Google Chrome 69.0.349
5) Text Editor : Sublime Text 3.1.1

b. Analisis Perangkat Keras / *Hardware*

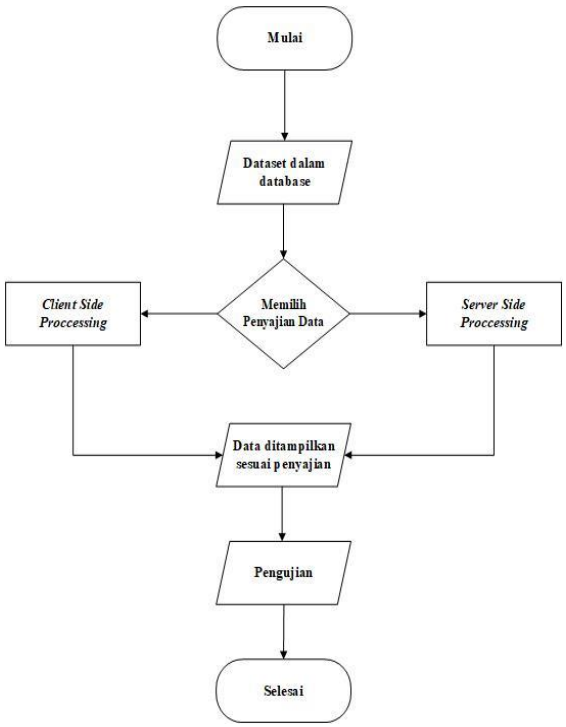
Perangkat keras yang digunakan untuk menerapkan teknik *server-side processing* saat ini spesifikasinya adalah sebagai berikut :

1) *Processor* dengan kecepatan minimal 1.5 GHz.
2) *Memory* RAM minimal 2GB
3) VGA minimal 512 MB
4) *Harddisk* dengan kapasitas penyimpanan minimal 500 GB
5) Monitor dengan resolusi 1366 x 768

B. *Design*

Setelah melakukan analisis terhadap sistem, sesuai dengan kerangka penelitian maka tahap selanjutnya yang dilakukan adalah tahap desain. Dalam implentasi teknik *server-side processing* beberapa rancangan yang dibuat adalah *flowchart* alur sistem, struktur tabel *database* dan rancangan antar muka sistem.

Gambar 2 merupakan *flowchart* alur sistem untuk penerapan teknik *server-side processing* dimana dalam proses menampilkan data yang dipilih sesuai penyajian data tersebut dimulai dari memilih penyajian data kemudian data akan ditampilkan menurut penyajian data yang dipilih kemudian data akan ditampilkan dan dilakukan pengujian berdasarkan hasil tersebut.

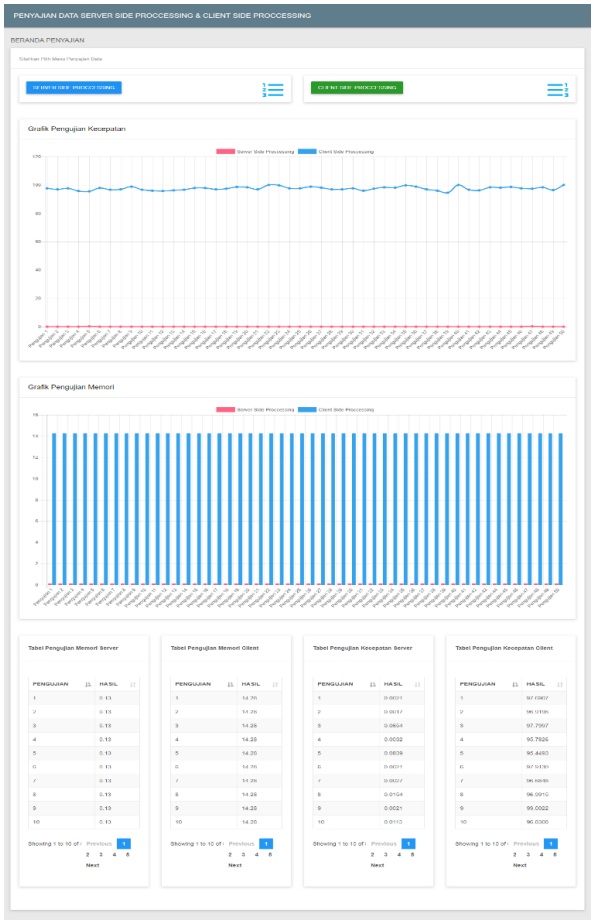


Sumber: (Sulastri et al., 2019)

Gambar 2. Flowchart Alur Sistem

C. *Implementasi Perbandingan Cliet Side Processing dengan Server-side Processing*

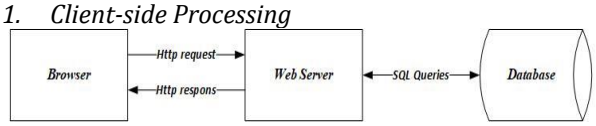
Gambar 3 merupakan tampilan antarmuka *dashboard* yang digunakan untuk memilih penyajian data, grafik pengujian dan tabel pengujian.



Sumber: (Sulastri et al., 2019)

Gambar 3. Tampilan Antar Muka Dashboard

1. *Client-side Processing*



Sumber: (Sulastri et al., 2019)

Gambar 4. Client-side Processing

Gambar 4 merupakan alur *client-side processing, browser* melakukan *request* semua data ke *web server* yang terhubung ke *database*. Sebelum ditampilkan semua data akan diload terlebih dahulu oleh *web browser,* setelah data permintaan di dapatkan barulah *web server* akan mengirimkan data untuk ditampilkan di *web browser*.

Dalam mode pemrosesan sisi *client,* pengurutan data dalam tabel, penelusuran, *paging,* dan semua operasi pemrosesan data lainnya dilakukan di *web browser*. Ketika data yang diload oleh *browser* berjumlah sedikit, waktu pemrosesan tidak akan memerlukan waktu yang lama tetapi ketika data berjumlah sangat besar, proses load data akan

memerlukan waktu yang sangat lama dan tidak efisien.

```
<div class=”table-responsive”>
<table id=”client” class=”table table-
bordered table-striped table-hover”>

<thead>
        <tr>
                <th>ID</th>
                <th>JOB</th>
                <th>MARTIAL</th>
                <th>EDUCATION</th>
        </tr>
</thead>
<tbody>
<?php
     include “db/koneksi.php”;
     $query = mysql_query(“SELECT id, job,
marital, education FROM bank”);
     while ($row =
mysql_fetch_array($query)) {
?>
    <tr>
             <td><?=$row[‘id’];?></td>
             <td><?=$row[‘job’];?></td>
             <td><?=$row[‘marital’];?></td>
             <td><?=$row[‘education’];?></td
>
    </tr>
<?php } ?>
</tbody>
</table>
</div>
```

Sumber: (Sulastri et al., 2019)

Gambar 5. Contoh Sintaks Proses Penyajian Data Client-side Processing

Penjelasan gambar 5 yakni, perintah digunakan agar tabel yang dibuat menjadi *responsive* artinya tabel tidak akan berantakan walaupun dibuka melalui *device* lain seperti *smartphone* yang memiliki resolusi layar yang lebih kecil. Perintah id=”client” adalah id tabel yang nantinya akan digunakan untuk pemanggilan *Jquery DataTables* dengan pemanggilan *javascript,* sementara sintak class=”table table-bordered table-striped table-hover” merupakan class tambahan dengan menggunakan *bootstrap,* class ini juga berfungsi untuk membuat baris tabel yang bergaya strip, efek *hover* pada tabel pada row tabel pada saat cursor *mouse* diletakan diatas *row* tabel. Perintah include “db/koneksi.php”; digunakan untuk mengkoneksikan ke database dan perintah $query = mysql_query(“SELECT id, job, marital, education FROM bank”); pemanggilan data yang terdapat di *database* untuk ditampilkan di web *browser.*

```
<script>
  $(function () {
    $(document).ready( function () {
    $(‘#client’).DataTable();
} );
</script>
```

Sumber: (Sulastri et al., 2019)

Gambar 6. Contoh Sintaks Proses Menampilkan Jquery DataTables Client-side Processing

Penjelasan gambar 6 yakni, $(‘#client’).DataTable(); adalah sintak untuk mengkonversi tabel *HTML* ke dalam format *DataTables*.

```
<script
src=”js/jquery.dataTables.min.js”></script>
<script
src=”js/dataTables.bootstrap.min.js”></script
>
<link href=”css/dataTables.bootstrap.min.css”
rel=”stylesheet”>
```

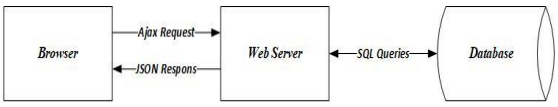
Sumber: (Sulastri et al., 2019)

Gambar 7. Contoh Sintaks Proses Pemanggilan Jquery DataTables

Gambar 7 merupakan sintak untuk bisa menggunakan *DataTables* dengan cara memanggil *Plugin Jquery DataTables* tersebut di direktori penyimpanan, ataupun bisa memanggilnya dengan cara *online* menggunakan *server* publik.

2. Server-side Proceessing

Pada tahap ini dijelaskan tahap pemrosesan *DataTables* menggunakan *server-side processing.*



Sumber: (Sulastri et al., 2019)

Gambar 8. Server-side Processing

Gambar 8 merupakan alur *server-side processing*, ketika menggunakan pemrosesan *sisi server*, *browser* akan membuat permintaan *Ajax* baru ke *server* untuk setiap permintaan data yang diminta. Permintaan ini kemudian dikodekan ke format *JSON* yang akan mengirimkan respon dari *script DataTables AJAX.* Sehingga permintaan *load* data akan terasa ringan dikarenakan data akan melakukan *request* perhalaman ketika akan di tampilkan di *web browser*.

```
<div class="row">
<div class="col-lg-12 col-md-12 col-sm-12
col-xs-12">
<div class="table-responsive">
<table id="tes" class="table table-striped
table-bordered">
<thead>
        <tr>
                <th>ID</th>
                 <th>JOB</th>
                 <th>MARTIAL</th>
                 <th>EDUCATION</th>

        </tr>
</thead>
</table>
</div></div></div>
```

Sumber: (Sulastri et al., 2019)

Gambar 9. Contoh Sintaks Proses Penyajian Data Server-side Processing.

Gambar 9 merupakan contoh sintak untuk penyajian data *server-side processing*, sama seperti penyajian data *client-side processing* tabel ini mempunyai id tabel yaitu id="tes" untuk pemanggilan *DataTables.* Seperti pada penjelasan alur proses server-side processing pada gambar 11, *DataTables* akan melakukan pemintaan *AJAX* baru kepada *server* untuk pemanggilan data.

```
<script>
        $(document).ready(function() {
                $('#tes').dataTable( {
                        "bProcessing": true,
                        "bServerSide": true,
                        "sAjaxSource":
"data.php",
                        "aoColumns": [
                          null,null,null,
                    null,
                          ]
                } );
        } );
</script>
```

Sumber: (Sulastri et al., 2019)

Gambar 10. Contoh Sintaks Proses Menampilkan Jquery DataTables Server-side Processing

Gambar 10 merupakan contoh sintak untuk mengkonversi tabel kedalam bentuk *DataTables* dengan permintaan *AJAX* baru untuk pemrosesan sisi *server*. Pada sintak *javascript* diatas, *request AJAX* akan diarahkan ke file data.php yang bertugas menghandle *server-side procesing* dan untuk perintah null digunakan untuk menampilkan kolom yang diisikan data dari *database.*

```
<?php
//data kolom yang akan ditampilkan
$aColums =
array('id','job','material','education','hou
sing','loan','contact','day','month');

//primary key
$sIndexColumn = "id";

//nama table database
$sTable = "bank";

//informasi koneksi ke database
$gaSql['user'] = "root";
$gaSql['password'] = "";
$gaSql['db'] = "bigdataset";
$gaSql['server'] = "localhost";

$gaSql['link'] =
mysql_pconnect($gaSql['server'],
$gaSql['user'], $gaSql['password']) or die
('Could not open connection to server'_;

Mysql_select_db($gaSql['db'],
$gaSql['link']) or die ('Could not select
database '.$gaSql['db']);

$sLimit = "";
If (isset($_GET['iDisplayStart']) &&
$_GET['iDisplayLength'] != '-1')
{
$sLimit = "LIMIT
".mysql_real_escape_string($_GET['iDisplaySt
art']).",".mysql_real_escape_string($_GET['i
DisplayLength']);
}
```

Sumber: (Sulastri et al., 2019)

Gambar 11. Sintak Pemrosesan Server-side Processing

Gambar 11 merupakan *script* yang bertugas untuk menghandle *server-side processing* dimana terdapat perintah $aColumns = array('id', 'job', 'marital', 'balance', 'housing', 'loan','contact', 'day', 'month'); yang berfungsi untuk menampilkan kolom pada tabel. Perintah $sIndexColumn = "id"; digunakan sebagai pemanggilan *primary key,* dalam database ini yang digunakan sebagai *primary key* adalah id. Perintah $sTable ="bank"; digunakan sebagai pemanggilan tabel dari *database*.Perintah $gaSql['user']="root";,$gaSql['password']="";,$gaSql['db']="bigdataset";,$gaSql['server']= "localhost"; digunakan sebagai koneksi *database,* dalam data ini database yang digunakan adalah bigdataset sementara perintah echo json_encode($output); digunakan respon permintaan data untuk ditampilkan dan perintah lainnya adalah perintah bawaan yang disediakan *DataTables* untuk pemrosesan dari sisi *server.*

D. *Pengujian*

Pengujian dilakukan menggunakan tools JMeter dengan cara membandingkan penyajian data menggunakan *client-side processing* dengan *server-side processing* berdasarkan *size*, kecepatan load data dan *Performance test*. Percobaan dilakukan secara *fair*, yaitu dengan menggunakan *database* yang sama sebanyak 50 kali.

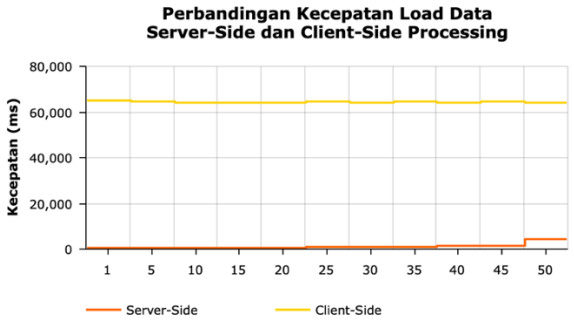
Tabel 1. Pengujian Server dan Client-Side Processing Menggunakan Tools JMeter

Pengujian	*Kecepatan (ms)*		*Memory (bytes)*	
	Server	*Client*	*Server*	*Client*
1	679	65013	8610	2208377
2	699	65062	8610	2207446
3	667	64286	8610	2208488
4	608	64311	8610	2209227
5	588	64760	8610	2207872
6	570	64967	8610	2207745
7	632	64637	8610	2207479
8	533	64592	8610	2207155
9	554	64590	8610	2207752
10	5151	64050	8610	2209042
11	476	64700	8610	2207816
12	497	64155	8610	2208626
13	437	64559	8610	2207830
14	458	64240	8610	2207727
15	399	64138	8610	2207740
16	359	64523	8610	2208026
17	341	64222	8610	2208121
18	303	64182	8610	2208025
19	179	64021	8610	2207855
20	324	64122	8610	2206916
21	223	64266	8610	2208669

22	204	64631	8610	2207156
23	268	64430	8610	2207684
24	960	64288	8610	2207309
25	868	64371	8610	2208591
26	1172	64736	8610	2207855
27	814	64798	8610	2207824
28	1199	64010	8610	2208133
29	837	64616	8610	2208126
30	937	64052	8610	2208412
31	1118	64598	8610	2207765
32	1183	64402	8610	2207885
33	880	64484	8610	2208267
34	1041	64828	8610	2207210
35	1101	64507	8610	2208189
36	1084	64899	8610	2207874
37	1123	64364	8610	2207298
38	1166	64695	8610	2207262
39	342	64239	8610	2207482
40	1274	63957	8610	2207651
41	850	63965	8610	2207725
42	953	64350	8610	2207518
43	1019	64545	8610	2208014
44	693	64924	8604	2207068
45	1355	64687	8610	2208092
46	1648	64014	8610	2208476
47	2184	64782	8610	2207516
48	2043	63930	8610	2207727
49	3510	64719	8610	2207082
50	4147	64334	8616	2207899
Rata Rata	960,28	62651,02	8610	2207860

Sumber: (Sulastri et al., 2019)

Tabel 1 menjelaskan riwayat pengujian *performance test* menggunakan tools JMeter, pengujian ini dilakukan sebanyak 50 kali percobaan. Dari hasil pengujian *performance test* didapatkan nilai rata-rata 62651,02 ms kecepatan load data dan 2207860 *bytes* penggunaan *memory* untuk *client-side processing* sedangkan untuk *server-side processing* didapatkan hasil dengan rata-rata 960,28 ms kecepatan load data dan 8610 bytes untuk penggunaan *memory*. Adapun hasil perbandingan kecepatan load data disajikan dalam bentuk grafik yang dapat dilihat pada Gambar 12.



Sumber: (Sulastri et al., 2019)

Gambar 12. Grafik Perbandingan Server dan Client-side Processing

KESIMPULAN

Penyajian data *server-side processing* telah berhasil di implementasikan pada aplikasi sederhana dan dilakukan proses perbandingan ukuran dan kecepatan load data dengan *client-side processing*. Dari hasil pengujian tools menggunakan JMeter terhadap *client-side processing* dan *server-side processing* didapat presentase 98,6% lebih cepat *server-side processing* dibandingkan *client-side processing.* Sementara pada penggunaan *memory server-side processing* lebih kecil 99,6% dibandingkan dengan *client-side processing.* Hal tersebut menunjukkan penggunaan teknik *server-side processing* lebih optimal. Untuk pengembangan teknik *server-side processing* ini, selanjutnya dapat dicoba menerapkan teknik *server-side processing* ke dalam proses lain selain *DataTables* dan proses pengujian menggunakan beberapa server terdistribusi *(distributed server)*.

REFERENSI

Burd, I. G. S., Us, W. A., Cooper, K. B., Us, W. A., Guthrie, S. D., Us, W. A., … Us, W. A. (2006). (12) United States Patent T A 1910, *1*(12).

Gartner. (n.d.). What Is Big Data ? Gartner IT Glossary - Big Data.

Gope, D., Schlais, D. J., & Lipasti, M. H. (2017). Architectural Support for Server-Side PHP Processing. *ACM SIGARCH Computer Architecture News*, *45*(2), 507–520. https://doi.org/10.1145/3140659.3080234

Husen, H., Rahmatulloh, A., & Sulastri, H. (2019). Implementasi Komunikasi Full Duplex Menggunakan Web Socket Pada Sistem Informasi Pengelolaan Anggaran Universitas Abc. *Simetris: Jurnal Teknik Mesin, Elektro Dan Ilmu Komputer*, *9*(1), 603–612.

Kersten, B., & Goedicke, M. (2010). Browser-based Analysis of Web Framework Applications. *Electronic Proceedings in Theoretical Computer Science*, *35*, 51–62. https://doi.org/10.4204/eptcs.35.5

SpyMedia. (2019). DataTables Examples - Server Side Proccessing. Retrieved from https://datatables.net/examples/server_side

Sulastri, H., Rahmatulloh, A., & Hidayat, D. K. (2019). *Laporan Akhir Penelitian "Server-Side Processing Techniques For Optimizing The Speed Of Presenting Big Data."* Jawa Barat.

Wang, D. L., Zender, C. S., & Jenks, S. F. (2009). Efficient clustered server-side data analysis workflows using SWAMP. *Earth Science Informatics*, *2*(3), 141–155. https://doi.org/10.1007/s12145-009-0021-z

Yannick Saillet, S. (DE). (2013). Server Side Processing Of User Interactions With A Web Browser, *2*(12).