

Prosiding
ANNUAL RESEARCH SEMINAR 2016

6 Desember 2016, Vol 2 No. 1

ISBN : 979-587-626-0 | UNSRI

<http://ars.ilkom.unsri.ac.id>

Sistem Pemanggilan Antrian Menggunakan WebSocket

Nur Rachmat

STMIK Global Informatika MDP
Magister Teknik Informatika,
Universitas Sriwijaya
Palembang, Indonesia
rachmat.nur91@mdp.ac.id

Orissa Octaria

STMIK Global Informatika MDP
Magister Teknik Informatika,
Universitas Sriwijaya
Palembang, Indonesia
orissa.octaria@mdp.ac.id

Dwi Meilitasari Tarigan

Magister Teknik Informatika
Universitas Sriwijaya
Palembang, Indonesia
dwimeylitasaritargan@gmail.com

Samsuryadi

Fakultas Ilmu Komputer,
Universitas Sriwijaya
Palembang, Indonesia
samsuryadi@unsri.ac.id

Abstrak— WebSocket memungkinkan komunikasi *full-duplex* secara *asynchronous* antara aplikasi berbasis web dan web-server. WebSocket adalah bagian dari HTML5 yang telah distandarisasi, tetapi saat ini telah dikembangkan secara terpisah dengan HTML5. Dengan WebSocket, sangat mudah sekali mengembangkan aplikasi berbasis *Server - Client* dengan komunikasi yang efisien dan *real-time*. Aplikasi berbasis WebSocket dapat berkomunikasi dengan *server* WebSocket menggunakan pesan berbasis teks, sehingga WebSocket dapat dengan mudah diaplikasikan ke dalam sistem pemanggilan antrian. Sistem pemanggilan antrian ini dibuat berbasis web sehingga penerapannya sangat mudah, hanya memerlukan Web Browser yang ada pada komputer *client*. Sementara untuk WebSocket *server* banyak pilihan *library* yang dikembangkan secara *open source* salah satunya adalah Ratchet yang dibuat dengan bahasa pemrograman PHP.

Kata Kunci— Antrian; WebSocket;

I. PENDAHULUAN

Sistem pemanggilan antrian sering dijumpai pada kantor pelayanan publik seperti Bank, Kantor Pemerintahan, Rumah Sakit, Kepolisian serta kantor pelayanan publik lainnya. Sistem tersebut dioperasikan oleh seorang atau lebih petugas loket dimana informasi panggilan berupa nomor antrian yang akan dilayani satu per satu oleh petugas loket. Informasi panggilan nomor antrian biasanya ditampilkan pada layar monitor yang dapat dilihat oleh *customer*.

Sebagai usaha untuk memberikan pilihan platform, pada penelitian ini dibuat sistem pemanggilan antrian berbasis web dengan memanfaatkan WebSocket untuk meningkatkan fleksibilitas, keandalan serta kemudahan dalam penerapannya. Sebagai server yang melayani proses pemanggilan menggunakan WebSocket *library* Ratchet berbasis bahasa pemrograman PHP.

Pada beberapa penelitian sebelumnya, aplikasi antrian dibangun pada *platform* berbasis *desktop* seperti Visual Basic .NET dan Delphi 7[1][2]. Pada penelitian [1] penulis membuat

aplikasi antrian pada pelayanan Bank menggunakan Visual Basic .NET. Sedangkan pada penelitian [2] penulis membuat aplikasi pelayanan loket dengan bahasa pemrograman Delphi7. Belum pernah dijumpai sebuah sistem maupun aplikasi antrian dengan memanfaatkan *protocol* WebSocket, padahal penerapannya jauh lebih mudah, karena hanya dengan memanfaatkan Web Browser pada perangkat baik berbasis *desktop* ataupun *mobile*.

Penelitian yang memanfaatkan *protocol* WebSocket sudah pernah dilakukan sebelumnya. Seperti penelitian [3] yang menggunakan WebSocket untuk aplikasi pengontrol suatu sistem. Penelitian [4] memvisualisasikan data dari jauh sehingga pengguna tidak perlu mengunduhnya terlebih dahulu. Begitupun pada penelitian [5] menggunakan WebSocket untuk aplikasi *chatting* yang terenkripsi dengan algoritma Rijndael 128.

Pada penelitian ini dikembangkan sistem pemanggilan antrian berbasis web memanfaatkan WebSocket sebagai *protocol* komunikasi untuk mengirim dan menerima pesan berbasis teks antara petugas loket dan layar informasi antrian.

II. LANDASAN TEORI

2.1. WebSocket

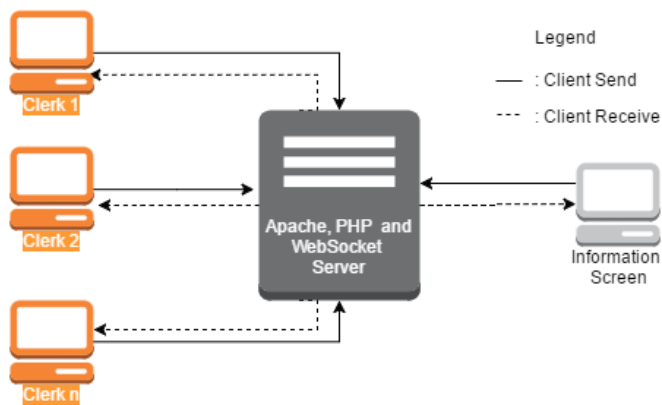
WebSocket pada dasarnya menggunakan koneksi dua arah dalam satu *TCP Connection*. [3][6][7] Protocol WebSocket mengurangi *overhead* pada komunikasi internet serta meningkatkan efisiensi, komunikasi yang pasti antara *Web server* dan *client*. [8]

2.2. Ratchet

Ratchet adalah *library* WebSocket berbasis bahasa pemrograman PHP. Ratchet dapat diunduh melalui socketo.me dan dikembangkan secara terbuka melalui github.

III. HASIL DAN PEMBAHASAN

Pada penelitian ini, alur proses *transport* pada WebSocket dalam aplikasi pemanggilan antrian dapat dilihat pada gambar 1. Pertama-tama, petugas loket akan mengirimkan pesan perintah berupa teks kepada WebSocket *server*. Pada pesan tersebut terdapat perintah yang akan diterjemahkan oleh WebSocket *server* dengan menggunakan teknik *parsing* dan *matching* pada *string* dimana pola pesan yang dikirim dan pesan yang diterima dapat dilihat pada tabel 1. Setelah pesan berhasil di terjemahkan oleh WebSocket *server* maka, WebSocket *server* akan mengirimkan pesan tersebut ke layar informasi untuk ditampilkan dan dilakukan pemanggilan suara yang dikontrol dengan Javascript Audio API[9]. Pesan yang dikirim dan diterima melalui *protocol* WebSocket dirangkai dalam satu pesan dengan tanda ":" sebagai pemisah antar *string*, sehingga akan terbentuk sebuah pesan seperti tabel 2.



Gambar 1. Alur *transport* WebSocket pada sistem pemanggilan antrian

TABLE III. TABEL POLA PESAN

Proses	Pola Pesan			Keterangan / Alur Proses
	Perintah	Pengirim	Pesan	
Kirim	calling	Nomor Loket	-	Loket - Server
Kirim dan Terima	call	Nomor Loket	Nomor Antrian	Server - Layar Informasi
Kirim dan Terima	recall	Nomor Loket	Nomor Antrian	Loket - Server - Layar Informasi
Terima	keep	Nomor Loket	Nomor Antrian	Server - Loket
Kirim dan Terima	next	-	-	Layar Informasi - Server - Loket

TABLE IV. TABEL PESAN

Proses	Isi Pesan	Keterangan
Kirim	calling:1 calling:n	Loket memulai pemanggilan
Kirim dan Terima	call:1:1 call:n:n	Server memanggil nomor antrian <i>n</i>
Kirim dan Terima	recall:1:1 recall:n:n	Loket <i>n</i> memanggil ulang nomor antrian <i>n</i>

Proses	Isi Pesan	Keterangan
Terima	keep:1:1 keep:n:n	Server mengirim informasi nomor yang sedang dipanggil kepada loket <i>n</i>
Kirim dan Terima	next	Layar informasi memberikan tanda jika pemanggilan telah selesai dilakukan

2.3. Implementasi WebSocket Server Application

Untuk WebSocket *server*, pada penelitian ini menggunakan WebSocket yang disediakan oleh socketo.me berupa *library* Ratchet yang ditulis dalam bahasa pemrograman PHP. Implementasi penggunaan WebSocket tersebut dapat dilihat pada listing program pada Gambar 2. Fungsi *onMessage* yang ada pada *class* Antrian digunakan untuk menangani setiap pesan yang diterima *server* dan dikirimkan ke *client*. Pada fungsi ini juga proses *parsing* dan *matching string* dari pesan yang diterima *server* dilakukan. Setiap *client* yang tersambung ke *server* memiliki *resource id* yang berbeda-beda. Fungsi ini memastikan pesan yang dikirimkan *client* akan dikirim ke *client* lain yang telah terkoneksi ke *server* atau membalas pesan ke *client* pengirim berdasarkan *resource id*.

```

namespace MyApp;
use Ratchet\MessageComponentInterface;
use Ratchet\ConnectionInterface;

class Antrian implements MessageComponentInterface {
    protected $clients;
    public function __construct() {
        $this->clients = new \SplObjectStorage;
    }
    public function onOpen(ConnectionInterface $conn) {
        // Store the new connection to send messages to later
        $this->clients->attach($conn);
    }
    public function onMessage(ConnectionInterface $from, $msg) {
        foreach ($this->clients as $client) {
            //the main program with explode string parsing
            //and string matching logic

            // The sender is the receiver, send to each client connected
            if($from == $client){
                //compose message to send with ":" delimiter
                $client->send("<messages>");
            }
            // The sender is not the receiver, send to each client connected
            if ($from != $client) {
                //compose message to send with ":" delimiter
                $client->send("<messages>");
            }
        }
    }
    public function onClose(ConnectionInterface $conn) {
        // The connection is closed, remove it, as we can no longer send it messages
        $this->clients->detach($conn);
    }
    public function onError(ConnectionInterface $conn, \Exception $e) {
        $conn->close();
    }
}
    
```

Gambar 2. Implementasi *library* WebSocket Ratchet pada Aplikasi *server*

Aplikasi WebSocket *server* ini harus dijalankan terlebih dahulu melalui PHP CLI (*Command Line Interface*) agar dapat menerima dan mengirim pesan. Pada penelitian ini WebSocket *server* dijalankan pada *port* 8080 seperti yang terlihat pada listing program pada Gambar 3. Perintah *run()* pada program berfungsi untuk menjalankan *library* Ratchet

dan *class* *Antrian* yang merupakan program utama dari aplikasi WebSocket *server*[10].

```
use Ratchet\Server\IoServer;
use Ratchet\Http\HttpServer;
use Ratchet\WebSocket\WsServer;
use MyApp\Antrian;
require dirname(__DIR__) . '/vendor/autoload.php';

$server = IoServer::factory(
    new HttpServer(
        new WsServer(
            new Antrian()
        )
    ),
    8080
);

$server->run();
```

Gambar 3. WebSocket *server* yang akan dijalankan dari PHP CLI

2.4. Implementasi WebSocket Client

Aplikasi *client* ditulis dalam bahasa pemrograman Javascript. Untuk dapat terkoneksi dengan WebSocket *server* harus dibuat sebuah *object* WebSocket baru. Pada penelitian ini dibuat sebuah *object* "ws" untuk melakukan koneksi ke WebSocket *server* seperti pada Gambar 4. Perlu diingat bahwa komunikasi pada WebSocket dilakukan secara *asynchronous*[3][6], jadi dibutuhkan *event-driven handler* untuk menanganinya[3].

Pesan yang diterima dari WebSocket *server* akan diterima oleh *event onMessage()*. Pada bagian inilah program utama WebSocket *client* dijalankan. Pesan yang diterima dari *server* kembali diterjemahkan dengan teknik *string parsing* dan *string matching*. Untuk mengirim pesan ke *server*, aplikasi *client* menggunakan fungsi *send()* dari WebSocket API. Pada *event* ini juga dibuat sebuah fungsi untuk memutar suara dengan Javascript Audio API.

```
ws = WebSocket("ws://<hostname>[:port]");

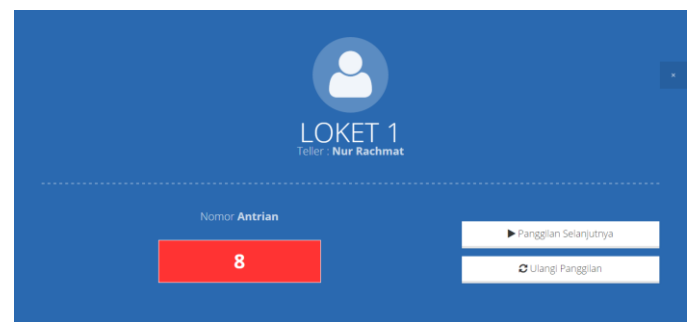
//send a message with
ws.send(<message>);
//receive a message with
ws.onmessage = function(event) {
    var message = event.data;
    //the main program with split() string parsing
    //and string matching logic
}
//Event handler on Connection Opened
ws.onopen = function(event) {
    console.log("Connection established!");
};
//Event handler on Connection Closed
ws.onclose = function() {
    console.log("Connection closed!");
};
```

Gambar 4. Implementasi WebSocket API pada *client*

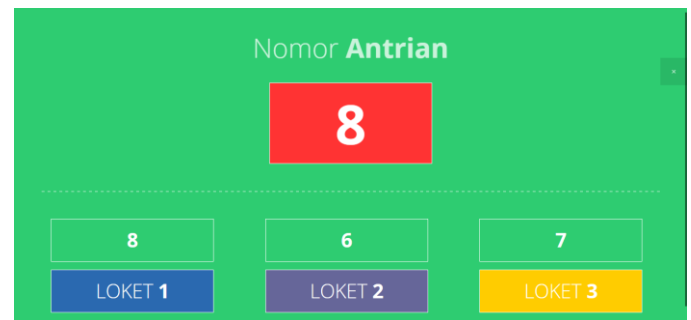
2.5. Implementasi Antarmuka

Gambar 4 dan 5 menunjukkan *capture* dari sistem pemanggilan antrian dengan WebSocket yang telah dibuat.

Gambar 4 merupakan tampilan dari halaman kontrol petugas loket. Pada halaman ini petugas loket dapat melakukan pemanggilan nomor antrian atau mengulang proses pemanggilan nomor antrian. Gambar 5 merupakan tampilan dari halaman informasi antrian. Halaman ini akan ditampilkan pada layar monitor yang dapat dilihat dengan jelas oleh *customer*. Pada halaman ini juga proses pemanggilan suara dilakukan dan dilakukan secara bergantian, artinya jika proses pemanggilan dari petugas loket 1 sedang berlangsung maka petugas loket lain harus menunggu hingga pemanggilan dari loket 1 selesai dilakukan.



Gambar 4. Implementasi Antarmuka Petugas Loket



Gambar 5. Implementasi Antarmuka Layar Informasi Antrian

IV. KESIMPULAN

Sistem pemanggilan antrian dapat dibangun dengan memanfaatkan *protocol* WebSocket. Penggunaan dan pemasangannya pun cukup mudah, tidak membutuhkan *plugin* tambahan untuk menjalankan sistem ini, hanya sebuah Web Browser yang mendukung HTML5. Ketersediaan *library* WebSocket *server* sangat beragam sehingga memberikan kemudahan bagi pengembang, salah satunya adalah Ratchet berbasis PHP ini. Dari contoh sistem ini dapat dikembangkan sistem serupa lainnya yang mengharuskan terjadinya koneksi antara *Client* dan *Server* serta ketersediaan data yang lebih *real-time*.

Prosiding
ANNUAL RESEARCH SEMINAR 2016
6 Desember 2016, Vol 2 No. 1

ISBN : 979-587-626-0 | UNSRI

<http://ars.ilkom.unsri.ac.id>

DAFTAR PUSTAKA

- [1] A. A. Nugroho and H. Agustian, "PENGEMBANGAN APLIKASI DAN ANALISIS SISTEM ANTRIAN NASABAH PADA PELAYANAN BANK BERBASIS CLIENT-SERVER Studi Kasus Pada PT BRI Unit Sako Kenten Palembang," STMIK GI MDP, 2011.
- [2] P. HARI WIBOWO, "PEMBUATAN APLIKASI SISTEM ANTRIAN PADA LOKET BAGIAN KEUANGAN UNIVERSITAS STIKUBANK SEMARANG," Universitas STIKUBANK Semarang, 2013.
- [3] Y. Furukawa, "Web-Based Control Application Using WebSocket," *Proc. ICALEPCS2011*, pp. 673–675, 2011.
- [4] A. Wessels, M. Purvis, J. Jackson, and S. Rahman, "Remote Data Visualization through WebSockets," in *2011 Eighth International Conference on Information Technology: New Generations*, 2011, pp. 1050–1051.
- [5] E. Sularsono, W. S. Raharjo, and Y. Lukito, "IMPLEMENTASI ALGORITMA RIJNDAEL 128 PADA APLIKASI CHATTING BERBASIS HTML5 WEBSOCKET," *J. Teknol. Komput. dan Inform.*, vol. 10, no. 2, pp. 66–79, 2014.
- [6] D. G. Puranik, D. C. Feiock, and J. H. Hill, "Real-time monitoring using AJAX and WebSockets," in *Proceedings of the International Symposium and Workshop on Engineering of Computer Based Systems*, 2013, pp. 110–118.
- [7] V. Wang, F. Salim, and P. Moskovits, *The Definitive Guide to HTML5 WebSocket*, 1st ed. Berkely, CA, USA: Apress, 2013.
- [8] V. Pimentel and B. G. Nickerson, "Communicating and displaying real-time data with WebSocket," *IEEE Internet Comput.*, vol. 16, no. 4, pp. 45–53, 2012.
- [9] W3Schools, "HTML Audio and Video DOM Reference." [Online]. Available: http://www.w3schools.com/tags/ref_av_dom.asp. [Accessed: 20-Nov-2016].
- [10] Ratchet, "Creating Your First Application." [Online]. Available: <http://socketo.me/docs/hello-world>. [Accessed: 23-Nov-2016].