Laporan Praktikum

Mata Kuliah Pemrograman Web & PBO

Pertemuan 15

"Web Socket"

Dosen Pengampu:
Wildan Aprizal Arifin, S.Pd., M.Kom.

Disusun Oleh:

Angga Fadzar	:	2306201
Annisa Nur Fadillah	:	2300349
Nur Annisa Vian Husaine	:	2307061

PROGRAM STUDI SISTEM INFORMASI KELAUTAN

UNIVERSITAS PENDIDIKAN INDONESIA

2024

I. PENDAHULUAN

Dalam era teknologi saat ini, komunikasi data real-time menjadi salah satu kebutuhan penting dalam pengembangan aplikasi modern, seperti sistem notifikasi, aplikasi chat, dan layanan streaming. WebSocket adalah salah satu protokol komunikasi yang memungkinkan koneksi dua arah antara klien dan server secara penuh, sehingga data dapat dikirimkan secara efisien tanpa perlu membuka koneksi baru untuk setiap permintaan.

Pada praktikum ini, WebSocket digunakan untuk membangun sistem komunikasi antara klien dan server, dengan fitur tambahan berupa pengolahan data dari database MySQL. Sistem ini mencakup pengiriman data real-time, notifikasi berdasarkan keyword tertentu dalam pesan, serta penyimpanan data ke dalam database. Selain itu, frontend sederhana dibuat dengan HTML dan JavaScript untuk menampilkan pesan yang diterima dari server WebSocket.

Tujuan dari praktikum ini adalah untuk mengintegrasikan protokol WebSocket dengan database MySQL dan mengembangkan sistem notifikasi real-time, yang dapat digunakan dalam berbagai aplikasi modern berbasis web. Dengan pendekatan ini, diharapkan peserta praktikum dapat memahami bagaimana protokol WebSocket bekerja, serta bagaimana mengintegrasikannya dengan sistem database.

II. ALAT DAN BAHAN

1. **Perangkat Lunak:**

- Node.js
- MySQL
- Editor Kode (Visual Studio Code)
- Browser (Chrome, Firefox, dll.)

2. **File Kode:**

- server.js
- app.js
- index.html
- db.js (untuk koneksi database)

III. LANGKAH KERJA

FILE KODE

1. server.js
2. app.js
3. index.html
4. db.js

INISIALISASI PROJEK

npm init -y

npm install mysql ws

KONFIGURASI DATABASE

```
CREATE DATABASE websocket_db;
USE websocket_db;

CREATE TABLE logs (
    id INT AUTO_INCREMENT PRIMARY KEY,
    message VARCHAR(255),
    timestamp DATETIME
);
```

Kode untuk Server WebSocket (server.js) Server membaca data dari MySQL dan mengirimkannya ke klien melalui WebSocket, serta mengirimkan notifikasi jika terdapat keyword tertentu.

```
const WebSocket = require('ws');
const db = require('./db'); // Import koneksi MySQL
const wss = new WebSocket.Server({ port: 3000 });

wss.on('connection', (ws) => {
    console.log('Client connected');

    // Kirim data dari MySQL ke klien (batasi 10 entri)
    db.query('SELECT * FROM logs LIMIT 10', (err, results) => {
        if (err) {
            console.error('Error membaca data dari database:', err);
            return;
        }
        ws.send(JSON.stringify({ type: 'database', data: results }));
    });

    // Interval untuk kirim pesan real-time ke klien
    const interval = setInterval(() => {
        if (ws.readyState === WebSocket.OPEN) {
            ws.send(JSON.stringify({
                type: 'realtime',
                message: 'Data dari server',
                timestamp: new Date()
            }));
        }
    }, 3000);

    // Terima pesan dari klien
    ws.on('message', (message) => {
        try {
```

```
            const msg = JSON.parse(message);

            // Notifikasi jika terdapat keyword "urgent"
            if (msg.text && msg.text.includes('urgent')) {
                ws.send(JSON.stringify({ type: 'notification', notification:
'Keyword detected: urgent' }));
            }

            console.log(`Pesan dari client (JSON):`, msg);
        } catch (err) {
            console.error('Pesan dari client (non-JSON):', message);
        }
    });

    ws.on('close', () => {
        console.log('Client disconnected');
        clearInterval(interval); // Hentikan interval saat klien terputus
    });
});

console.log('WebSocket server berjalan di ws://localhost:3000');
```

Kode untuk Klien Node.js (app.js) Klien menerima data dari server WebSocket dan menyimpannya ke MySQL.

```
const WebSocket = require('ws');
const db = require('./db'); // Import koneksi MySQL

// Koneksi ke WebSocket Server
const ws = new WebSocket('ws://localhost:3000');

ws.on('open', () => {
    console.log('Terhubung ke WebSocket server');

    // Kirim pesan dalam format JSON
    ws.send(JSON.stringify({ text: 'Halo Server! Ini dari Client' }));
});

ws.on('message', (data) => {
    try {
        const jsonData = JSON.parse(data);
        console.log('Data diterima dari server:', jsonData); // Log data

        if (jsonData.type === 'database') {
            jsonData.data.forEach(item => {
                console.log('Data untuk disimpan ke database:', item); //
Debug data per item
```

```
                const query = 'INSERT INTO logs (message, timestamp) VALUES
(?, ?)';
                const values = [item.message, item.timestamp];

                db.query(query, values, (err, result) => {
                    if (err) {
                        console.error('Gagal menyimpan data ke MySQL:', err);
                    } else {
                        console.log('Data berhasil disimpan ke database, ID:',
result.insertId);
                    }
                });
            });
        } else if (jsonData.type === 'realtime') {
            console.log('Data real-time diterima dari server:', jsonData);
        } else if (jsonData.type === 'notification') {
            console.log('Notifikasi diterima:', jsonData.notification);
        }
    } catch (err) {
        console.error('Error parsing message:', err);
    }
});


// Handle error
ws.on('error', (err) => {
    console.error('Error WebSocket:', err);
});

ws.on('close', () => {
    console.log('WebSocket connection closed');
});
```

Kode untuk Antarmuka HTML (index.html) Klien browser menerima data dari server WebSocket dan menampilkannya.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>WebSocket Client</title>
  </head>
  <body>
    <h1>WebSocket Client</h1>
    <div id="messages"></div>
    <script>
      const ws = new WebSocket("ws://localhost:3000");
```

```
      ws.onopen = () => {
        console.log("Connected to server");
        ws.send(JSON.stringify({ text: "Hello from client!" }));
      };

      ws.onmessage = (event) => {
        const messages = document.getElementById("messages");
        try {
          const data = JSON.parse(event.data);

          if (data.type === 'database') {
            // Tampilkan data dari database
            data.data.forEach(item => {
              const messageElement = document.createElement("p");
              messageElement.textContent = `Database: ${item.message},
Timestamp: ${item.timestamp}`;
              messages.appendChild(messageElement);
            });
          } else if (data.type === 'realtime') {
            // Tampilkan data real-time
            const messageElement = document.createElement("p");
            messageElement.textContent = `Realtime: ${data.message},
Timestamp: ${data.timestamp}`;
            messages.appendChild(messageElement);
          } else if (data.type === 'notification') {
            // Tampilkan notifikasi
            const messageElement = document.createElement("p");
            messageElement.style.color = "red";
            messageElement.textContent = `Notification: ${data.notification}`;
            messages.appendChild(messageElement);
          }
        } catch (err) {
          console.error("Error parsing message from server:", err);
        }
      };

      ws.onerror = (error) => {
        console.error("WebSocket error:", error);
      };

      ws.onclose = () => {
        console.log("Disconnected from server");
      };
    </script>
  </body>
</html>
```

Koneksi ke Database (db.js) Koneksi ke MySQL:

```
const mysql = require('mysql');
// konfigurasi koneksi MySQL
const db = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: '',
    database: 'websocket_db'
});

// koneksi ke database
db.connect((err)=>{
    if (err){
        console.error('Error koneksi MySQL:', err);
    } else {
        console.log('Terhubung ke MySQL');
    }
});

module.exports = db;
```

Hasil Pengujian

1. **Server WebSocket**:
    - Server berhasil mengirim data dari database dan pesan real-time.
    - Notifikasi dikirim jika keyword urgent terdeteksi.
2. **Klien Node.js**:
    - Klien berhasil menerima data dan menyimpannya ke database logs.
3. **Klien Browser**:
    - Data ditampilkan di browser, baik data dari database maupun pesan real-time.

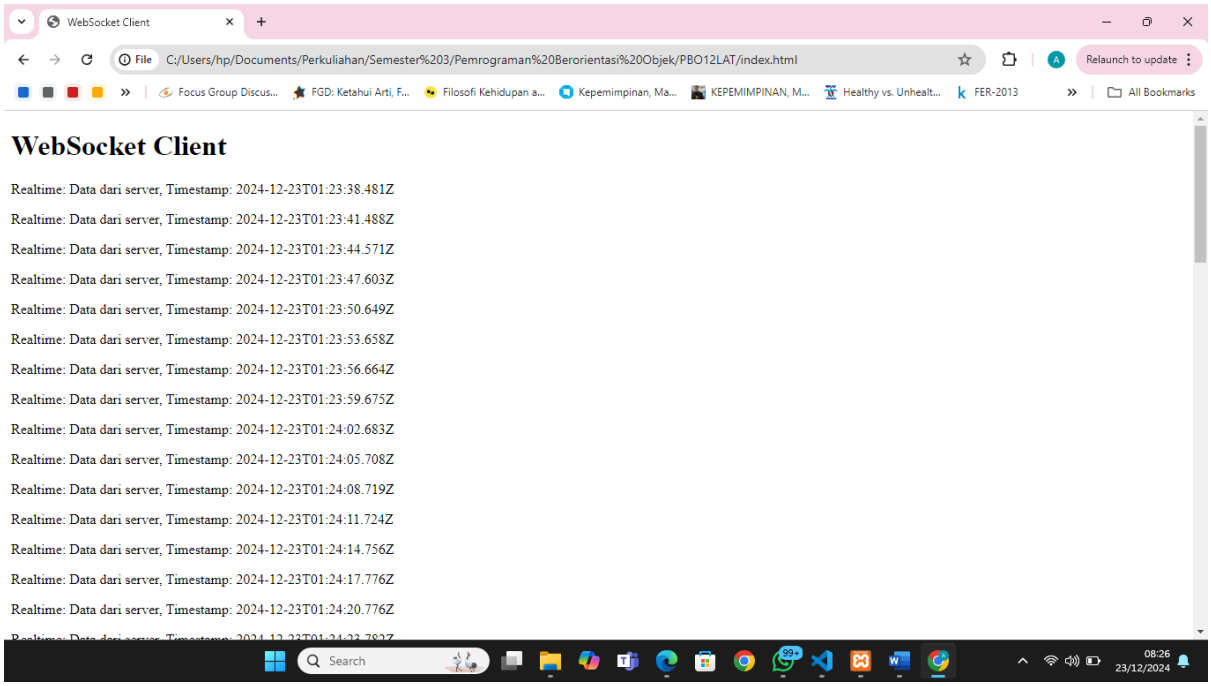
Untuk Menjalankan Server

```
PS C:\Users\hp\Documents\Perkuliahan\Semester 3\Pemrograman Berorientasi Objek\PBO12LAT> node server
WebSocket server berjalan di ws://localhost:3000
Terhubung ke MySQL
Client connected
Pesan dari client (JSON): { text: 'Halo Server! Ini dari Client' }
```

Untuk Menjalankan Client

```
PS C:\Users\hp\Documents\Perkuliahan\Semester 3\Pemrograman Berorientasi Objek\PBO12LAT> node app
Terhubung ke MySQL
Terhubung ke WebSocket server
Data diterima dari server: { type: 'database', data: [] }
Data diterima dari server: {
  type: 'realtime',
  message: 'Data dari server',
  timestamp: '2024-12-23T01:21:53.774Z'
```

Tampilan di index.html



IV. KESIMPULAN

Dari hasil praktikum, dapat disimpulkan bahwa:

1. Implementasi WebSocket berhasil dilakukan untuk komunikasi dua arah antara klien dan server. WebSocket memungkinkan pengiriman data secara real-time tanpa latensi yang signifikan.
2. Integrasi dengan MySQL berhasil dilakukan, di mana server dapat membaca data dari tabel logs dan mengirimkannya ke klien melalui WebSocket, serta menyimpan data yang diterima klien ke dalam database.
3. Sistem notifikasi real-time bekerja dengan baik, di mana keyword tertentu dalam pesan yang diterima dapat memicu pengiriman notifikasi khusus ke klien.
4. Frontend sederhana berbasis HTML dan JavaScript berhasil digunakan untuk menampilkan data dari server WebSocket, baik data dari database, pesan real-time, maupun notifikasi.

Melalui praktikum ini, peserta memperoleh pemahaman tentang bagaimana membangun sistem komunikasi real-time berbasis WebSocket yang terintegrasi dengan MySQL. Implementasi ini dapat diperluas untuk berbagai aplikasi modern seperti sistem monitoring, layanan notifikasi, dan aplikasi chat.

Link GitHub file: https://github.com/angfdzr/latihanwebsocket