|  |  |  |
| --- | --- | --- |
|  | **KONGU ENGINEERING COLLEGE**  (Autonomous)  Perundurai, Erode – 638 060  **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING** | KEC | Kongu Engineering College |

**22CSL52 - COMPUTER NETWORKS LABORATORY**

**MINI PROJECT**

**COLLABRATIVE TEXT EDITOR**

**Submitted by**

ARVIND M 22CSR023

ASWANTH K N 22CSR027

ASHWANTH S 22CSR025

|  |  |
| --- | --- |
| PROGRAM (10) |  |
| UI DESIGN (10) |  |
| DEMO (10) |  |
| DOCUMENTATION (10) |  |

**BLOCK DIAGRAM**

**UDP:**

**TCP:**

**Code in TCP:**

Client1.html  
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Collaborative Code Editor (TCP)</title>

    <style>

        #editor {

            width: 100%;

            height: 300px;

            border: 1px solid #ccc;

            font-family: monospace;

            padding: 10px;

        }

    </style>

</head>

<body>

    <h2>Collaborative Code Editor (TCP)</h2>

    <textarea id="editor" placeholder="Start typing..."></textarea>

    <script>

        const ws = new WebSocket('ws://localhost:8080');

        const editor = document.getElementById('editor');

        let isUpdating = false;

        ws.onmessage = (event) => {

            const message = JSON.parse(event.data);

            if (message.type === 'update' && !isUpdating) {

                isUpdating = true;

                editor.value = message.data;

                isUpdating = false;

            } else if (message.type === 'history') {

                editor.value = message.data;

            }

        };

        editor.addEventListener('input', () => {

            if (!isUpdating) {

                const data = { type: 'update', data: editor.value };

                ws.send(JSON.stringify(data));

            }

        });

    </script>

</body>

</html>

Client2.html  
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Collaborative Code Editor (TCP)</title>

    <style>

        #editor {

            width: 100%;

            height: 300px;

            border: 1px solid #ccc;

            font-family: monospace;

            padding: 10px;

        }

    </style>

</head>

<body>

    <h2>Collaborative Code Editor (TCP)</h2>

    <textarea id="editor" placeholder="Start typing..."></textarea>

    <script>

        const ws = new WebSocket('ws://localhost:8080');

        const editor = document.getElementById('editor');

        let isUpdating = false;

        ws.onmessage = (event) => {

            const message = JSON.parse(event.data);

            if (message.type === 'update' && !isUpdating) {

                isUpdating = true;

                editor.value = message.data;

                isUpdating = false;

            } else if (message.type === 'history') {

                editor.value = message.data;

            }

        };

        editor.addEventListener('input', () => {

            if (!isUpdating) {

                const data = { type: 'update', data: editor.value };

                ws.send(JSON.stringify(data));

            }

        });

    </script>

</body>

</html>

Tcp.js

const WebSocket = require('ws');

const net = require('net');

const PORT = 8080;

const wss = new WebSocket.Server({ port: PORT });

let clients = [];

let editorContent = ''; // Store the current state of the editor content

wss.on('connection', (ws) => {

    clients.push(ws);

    console.log('New client connected via TCP');

    // Send the editor content history to the new client

    ws.send(JSON.stringify({

        type: 'history',

        data: editorContent

    }));

    // Handle messages sent by clients

    ws.on('message', (message) => {

        try {

            const parsedMessage = JSON.parse(message);  // Parse incoming message as JSON

            if (parsedMessage.type === 'update') {

                editorContent = parsedMessage.data;

                // Broadcast the updated editor content to all clients

                clients.forEach(client => {

                    if (client.readyState === WebSocket.OPEN) {

                        client.send(JSON.stringify({

                            type: 'update',

                            data: editorContent

                        }));

                    }

                });

            }

        } catch (err) {

            console.log('Error parsing message:', err);

        }

    });

    // Handle client disconnection

    ws.on('close', () => {

        clients = clients.filter(client => client !== ws);

        console.log('Client disconnected');

    });

});

console.log(`WebSocket TCP server is running on ws://localhost:${PORT}`);

**Code in UDP:**

**Client1.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Collaborative Text Editor</title>

    <style>

        #editor {

            width: 100%;

            height: 300px;

            border: 1px solid #ccc;

            font-family: monospace;

            padding: 10px;

        }

    </style>

</head>

<body>

    <h2>Collaborative Text Editor</h2>

    <textarea id="editor" placeholder="Start typing..."></textarea>

    <script>

        const ws = new WebSocket('ws://localhost:8080');

        const editor = document.getElementById('editor');

        let isUpdating = false;

        ws.onopen = () => {

            console.log('Connected to WebSocket server');

        };

        ws.onmessage = (event) => {

            const message = JSON.parse(event.data);

            if (message.type === 'update' && !isUpdating) {

                isUpdating = true;

                editor.value = message.data;

                isUpdating = false;

            } else if (message.type === 'history') {

                editor.value = message.data;

            }

        };

        editor.addEventListener('input', () => {

            if (!isUpdating) {

                const data = { type: 'update', data: editor.value };

                ws.send(JSON.stringify(data));

            }

        });

        ws.onclose = () => {

            console.log('Disconnected from WebSocket server');

        };

    </script>

</body>

</html>

**Client2.html**  
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Collaborative Text Editor</title>

    <style>

        #editor {

            width: 100%;

            height: 300px;

            border: 1px solid #ccc;

            font-family: monospace;

            padding: 10px;

        }

    </style>

</head>

<body>

    <h2>Collaborative Text Editor</h2>

    <textarea id="editor" placeholder="Start typing..."></textarea>

    <script>

        const ws = new WebSocket('ws://localhost:8080');

        const editor = document.getElementById('editor');

        let isUpdating = false;

        ws.onopen = () => {

            console.log('Connected to WebSocket server');

        };

        ws.onmessage = (event) => {

            const message = JSON.parse(event.data);

            if (message.type === 'update' && !isUpdating) {

                isUpdating = true;

                editor.value = message.data;

                isUpdating = false;

            } else if (message.type === 'history') {

                editor.value = message.data;

            }

        };

        editor.addEventListener('input', () => {

            if (!isUpdating) {

                const data = { type: 'update', data: editor.value };

                ws.send(JSON.stringify(data));

            }

        });

        ws.onclose = () => {

            console.log('Disconnected from WebSocket server');

        };

    </script>

</body>

</html>

**udp.js**

const WebSocket = require('ws');

const PORT = 8080;

const wss = new WebSocket.Server({ port: PORT });

let clients = [];

let editorContent = ''; // Store the current state of the editor content

wss.on('connection', (ws) => {

clients.push(ws);

console.log('New client connected');

// Send the editor content history to the new client

ws.send(JSON.stringify({

type: 'history',

data: editorContent

}));

// Handle messages sent by clients

ws.on('message', (message) => {

try {

const parsedMessage = JSON.parse(message);  // Parse incoming message as JSON

if (parsedMessage.type === 'update') {

editorContent = parsedMessage.data;

// Broadcast the updated editor content to all clients

clients.forEach(client => {

if (client.readyState === WebSocket.OPEN) {

client.send(JSON.stringify({

type: 'update',

data: editorContent

}));

}

});

}

} catch (err) {

console.log('Error parsing message:', err);

}

});

// Handle client disconnection

ws.on('close', () => {

clients = clients.filter(client => client !== ws);

console.log('Client disconnected');

});

});

console.log(`WebSocket server is running on ws://localhost:${PORT}`);