**Word Document: Real-Time Chat App Using Socket.io**

### 1. package.json Dependencies (Backend)

{  
 "name": "chat-server",  
 "version": "1.0.0",  
 "main": "server.js",  
 "scripts": {  
 "start": "node server.js"  
 },  
 "dependencies": {  
 "express": "^4.18.2",  
 "socket.io": "^4.7.1",  
 "cors": "^2.8.5"  
 }  
}

### 2. server.js (Node.js + Socket.io Backend)

const express = require('express');  
const http = require('http');  
const { Server } = require('socket.io');  
const cors = require('cors');  
  
const app = express();  
app.use(cors());  
const server = http.createServer(app);  
  
const io = new Server(server, {  
 cors: {  
 origin: '\*',  
 methods: ['GET', 'POST']  
 }  
});  
  
io.on('connection', (socket) => {  
 console.log('New client connected:', socket.id);  
  
 socket.on('send\_message', (data) => {  
 io.emit('receive\_message', data);  
 });  
  
 socket.on('disconnect', () => {  
 console.log('Client disconnected:', socket.id);  
 });  
});  
  
const PORT = 5000;  
server.listen(PORT, () => console.log(`Server running on port ${PORT}`));

### 3. package.json Dependencies (Frontend - React)

{  
 "name": "chat-client",  
 "version": "1.0.0",  
 "private": true,  
 "dependencies": {  
 "react": "^18.2.0",  
 "react-dom": "^18.2.0",  
 "react-scripts": "5.0.1",  
 "socket.io-client": "^4.7.1"  
 },  
 "scripts": {  
 "start": "react-scripts start",  
 "build": "react-scripts build"  
 }  
}

### 4. src/App.js (React Frontend)

import React, { useState, useEffect } from 'react';  
import io from 'socket.io-client';  
  
const socket = io('http://localhost:5000');  
  
function App() {  
 const [message, setMessage] = useState('');  
 const [name, setName] = useState('');  
 const [messages, setMessages] = useState([]);  
  
 useEffect(() => {  
 socket.on('receive\_message', (data) => {  
 setMessages((prev) => [...prev, data]);  
 });  
  
 return () => {  
 socket.off('receive\_message');  
 };  
 }, []);  
  
 const sendMessage = () => {  
 if (message && name) {  
 const data = { name, message };  
 socket.emit('send\_message', data);  
 setMessage('');  
 }  
 };  
  
 return (  
 <div style={{ padding: '20px', maxWidth: '500px', margin: 'auto' }}>  
 <h2>Real-Time Chat</h2>  
 <input  
 type="text"  
 placeholder="Your Name"  
 value={name}  
 onChange={(e) => setName(e.target.value)}  
 style={{ width: '100%', marginBottom: '10px', padding: '5px' }}  
 />  
 <div style={{ border: '1px solid #ccc', padding: '10px', height: '300px', overflowY: 'scroll', marginBottom: '10px' }}>  
 {messages.map((msg, index) => (  
 <div key={index} style={{ marginBottom: '5px' }}>  
 <b>{msg.name}: </b>{msg.message}  
 </div>  
 ))}  
 </div>  
 <input  
 type="text"  
 placeholder="Type a message"  
 value={message}  
 onChange={(e) => setMessage(e.target.value)}  
 style={{ width: '100%', marginBottom: '10px', padding: '5px' }}  
 onKeyPress={(e) => e.key === 'Enter' && sendMessage()}  
 />  
 <button onClick={sendMessage} style={{ width: '100%', padding: '10px' }}>Send</button>  
 </div>  
 );  
}  
  
export default App;

### 5. Basic Styling (Optional)

body { font-family: Arial, sans-serif; }  
input { box-sizing: border-box; }  
button { cursor: pointer; }

This Word document contains all code required to build a **real-time chat application** using Node.js, Express, Socket.io, and React, following the expected UI from the provided image.