TASK - 2

Real - Time Chat Application

A real - time Chat application allow users to exchange messages instantly over the internet. it use Web Sockets to create a persistent connection b/w the server and client so messages appear immediately without refreshing the page.

Technologies used

- Node.js : javascript runtime for backend
- Express.js : Framework to handle server routes
- Socket io : Library for real-time event - based
- React js : Front-end Library to create responsive Uk.
- css/Tailwint : For responsive design and styling.

Working Principle :-

1. User connects to the server through Web Socket.
2. Server establishes a two - way persistent connection
3. When a user sends a message, it is immediately broadcasted to all connected clients.
4. The chat interface updates dynamically without Page reload.

Backend Code (Node.js with Express + Socket io)

Javascript :-

```
Const express = require ("express");
Const http  =  require ("http");
const { Server} = require ("socket-io");
Const app = express ();
Const Server = http . create Server (app):
 Const io= new Server (Server . {
      Cors: { origin : "*" }
   });
 Let Messages = [];
io, on ("Connection", (socket) => {
  Console, log ("user connected:", Socket, id);
 Socket, emit (" chat History", Messages);
Socket, on ("send Message", (data) => {
 Message . push (data);
  io, emit (" reciere Message", data);
 });
 Socket, on ("disconnect", () => {

   Console, log ("user disconnected:", Socket, id);
 });
 });
Server, Listen (4000, () => console . log ("✓ Server running
on port 4000 "));
```

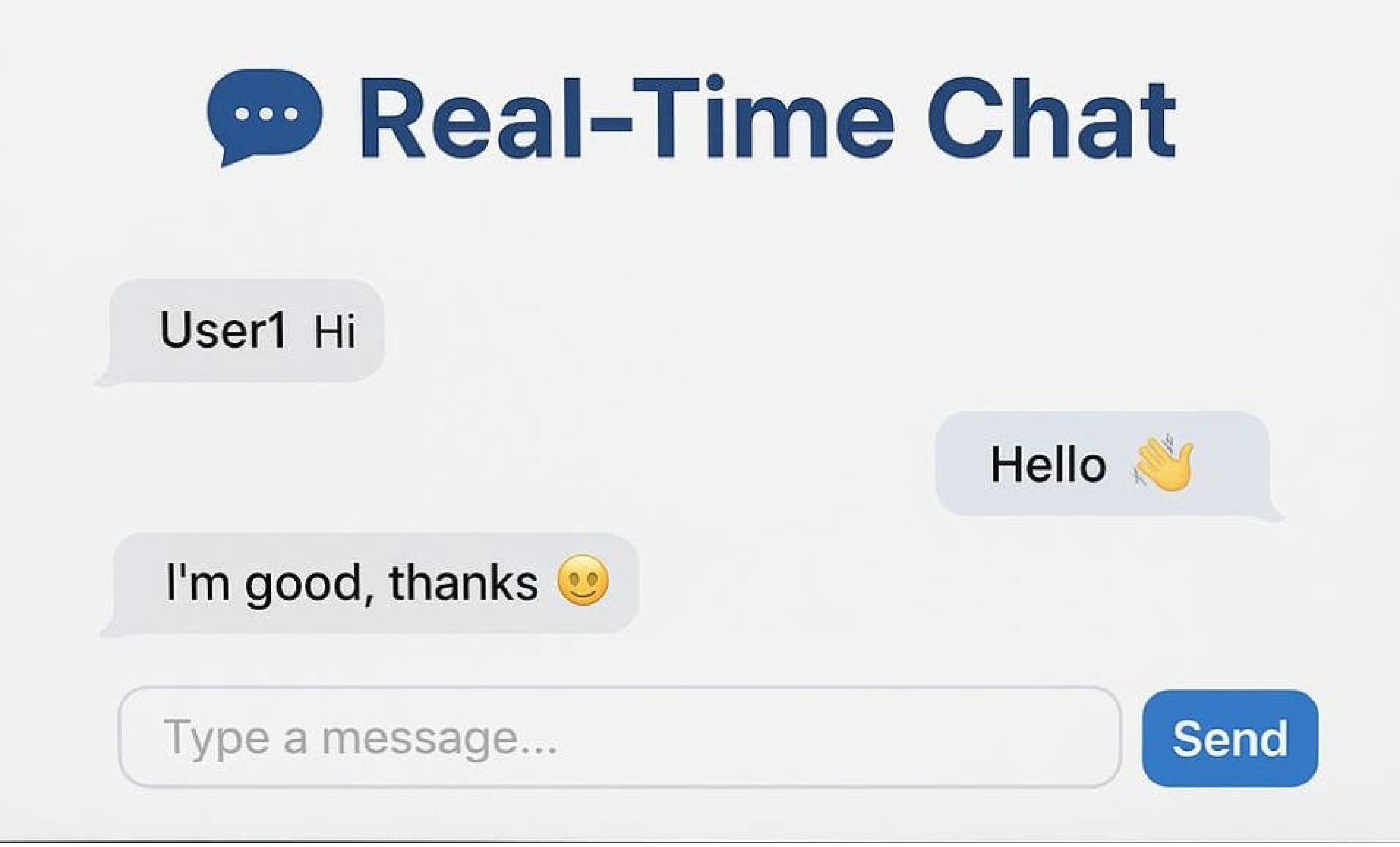
Frontend Code (React.js) :-

Javascript

```
import React . { use state , use effect } from "react";
import io from "Socket.io-client";
Const Socket = io("http://localhost:4000");
function App(){
 const [Message , set Message] = use state ("");
 Const    [chat , set chat ]= use State ([]):
 use effect (() => {
Socket.on ("chat History", (messages) => set chat (messages));
Socket.on("recieve message", (msg) => set chat ((prev) =>
 [.... prev, msg]));
  }, []):
  Const Send message = () => {
   if (message.trim ()! == "") {
    Const msg Data = { user : "User". text : message };
   Socket.emit ("Send message", msg Data):
    Set message("");
     }
   };

  return (
 <div ClassName = "flex flex-col items - Center justify-
-center min-h-Screen bg-gray-100")>
<h1 class Name = "text - 3xl font - bold Mb - 4">
  Real - time Chat App</h1>
<div class Name = "w - 96 bg - white rounded - 2xl shadow p-4">
<div class Nam = "h - 64 overflow - y - auto border p - 2 Mb - 3">
{ Chat . Map (Msg . index) => (
  <div key = {index} class Name = "p - 1">
<strong>{ Msg, user }: </strong> { Msg . text }
  </div>
)) }
   </div>
 <div Class Name = "flex">
  < input
  type = "text"
  classname = "flex-grow border rounded - l - xl p-2"
  Value = { Message }
  on change = { (e) => Set Message (e. target , value ) }
   place holder = " Type a Message... "
   />

 < button
 Class Name = " big - blue - 500 text - white px-4
 rounded - r - xl"
  On Click = { Send Message }
 >
    Send
 </button>
```

```
</div>

</div>
</div>
);
}
```

Output :-



💬 Real-Time Chat

User1 Hi

Hello 👋

I'm good, thanks 🙂