Experiment 8

Aim - Enable **real-time communication** via WebSockets.

Code -

db.js -

whatsapp-mern > backend > config > JS db.js > ...

```
// backend/config/db.js
import mongoose from "mongoose";

const connectDB = async () => {
  try {
    await mongoose.connect(process.env.MONGO_URI);
    console.log("MongoDB connected");
  } catch (err) {
    console.error("MongoDB connection failed:", err.message);
    process.exit(1);
  }
};

export default connectDB;

```

models/Message.js -

whatsapp-mern > backend > models > JS Message.js > ...

```
// backend/models/Message.js
import mongoose from "mongoose";

const messageSchema = new mongoose.Schema({
  room: { type: String, required: true },
  sender: { type: String, required: true },
  text: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
});

export default mongoose.model("Message", messageSchema);

```

routes/rooms.js -

whatsapp-mern > backend > routes > JS rooms.js > ...

```
// backend/routes/rooms.js
import express from "express";
import Message from "../models/Message.js";

const router = express.Router();

// get message history for a room
router.get("/:room/messages", async (req, res) => {
  try {
    const { room } = req.params;
    const msgs = await Message.find({ room }).sort({ createdAt: 1 }).limit(200);
    res.json(msgs);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

export default router;
```

server.js -

whatsapp-mern > backend > JS server.js > ...

```
import express from "express";
import http from "http";
import { Server } from "socket.io";
import cors from "cors";

const app = express();
app.use(cors());

const server = http.createServer(app);
const io = new Server(server, {
  cors: { origin: "http://localhost:5173", methods: ["GET", "POST"] },
});

io.on("connection", (socket) => {
  console.log("User connected:", socket.id);

  socket.on("sendMessage", (data) => {
    console.log("Message received:", data);
    // Broadcast to all users except sender
    socket.broadcast.emit("receiveMessage", data);
  });

  socket.on("disconnect", () => {
    console.log("User disconnected:", socket.id);
  });
});

server.listen(5000, () => console.log("✅ Server running on port 5000"));
```

App.jsx -

whatsapp-mern > frontend > src > App.jsx > ...

```jsx
import React, { useState, useEffect } from "react";
import { io } from "socket.io-client";
// Connect to backend Socket.io server
const socket = io("http://localhost:5000");
export default function App() {
  const [messages, setMessages] = useState([]);
  const [message, setMessage] = useState("");
  const [username, setUsername] = useState("");
  const [isLoggedIn, setIsLoggedIn] = useState(false);
  // Listen for incoming messages once
  useEffect(() => {
    const handleReceive = (data) => {
      setMessages((prev) => {
        if (prev.find((m) => m.id === data.id)) return prev; // avoid duplicates
        return [...prev, data];
      });
    };
    socket.on("receiveMessage", handleReceive);
    return () => socket.off("receiveMessage", handleReceive);
  }, []);
  const sendMessage = (e) => {
    e.preventDefault();
    if (message.trim() === "") return;
    const newMsg = {
      id: Date.now(), // unique ID for each message
      sender: username,
      text: message,
    };
    // Add message locally
    setMessages((prev) => [...prev, newMsg]);
    // Send to backend for broadcast
    socket.emit("sendMessage", newMsg);
    setMessage("");
  };
  // --- LOGIN SCREEN ---
  if (!isLoggedIn) {
    return (
      <div className="flex justify-center items-center h-screen bg-gray-900 text-white">···
      </div>
    );
  }
  // --- CHAT SCREEN ---
  return (
    <div className="flex flex-col h-screen bg-gray-900 text-white">···
    </div>
  );
}
```

Home.jsx -

whatsapp-mern > frontend > src > pages > Home.jsx > Home

```jsx
import React, { useEffect, useState, useRef } from "react";
import io from "socket.io-client";
import axios from "axios";
import Sidebar from "../components/Sidebar";
import ChatWindow from "../components/ChatWindow";
// set backend URL
const API_URL = "http://localhost:5000";
const SOCKET_URL = "http://localhost:5000";
const socket = io(SOCKET_URL);

export default function Home() {
  // sample users/rooms for demo
  const contacts = [···
  ];
  const [currentRoom, setCurrentRoom] = useState("room-alex"); // default
  const [username, setUsername] = useState("You"); // your display name
  const [messages, setMessages] = useState([]);
  useEffect(() => {
    // join room on change
    socket.emit("joinRoom", currentRoom);
    // fetch history
    const fetchHistory = async () => {
      const roomKey = currentRoom;
      const res = await axios.get(`${API_URL}/api/rooms/${roomKey}/messages`);
      setMessages(res.data || []);
    };
    fetchHistory();
    // listen for new messages
    const handler = (msg) => {
      if (msg.room === currentRoom) {
        setMessages((prev) => [...prev, msg]);
      }
    };
    socket.on("newMessage", handler);
    // cleanup
    return () => {
      socket.off("newMessage", handler);
    };
  }, [currentRoom]);
  const sendMessage = (text) => {
    if (!text) return;
    socket.emit("sendMessage", { room: currentRoom, sender: username, text });
    // optimistic UI already handled by server emitting back to room
  };
  return (
    <div className="min-h-screen bg-gray-100 dark:bg-gray-900 text-gray-900 dark:text-gray-100">
      <div className="max-w-5xl mx-auto grid grid-cols-4 gap-4 p-4">
        <Sidebar
```

Output -

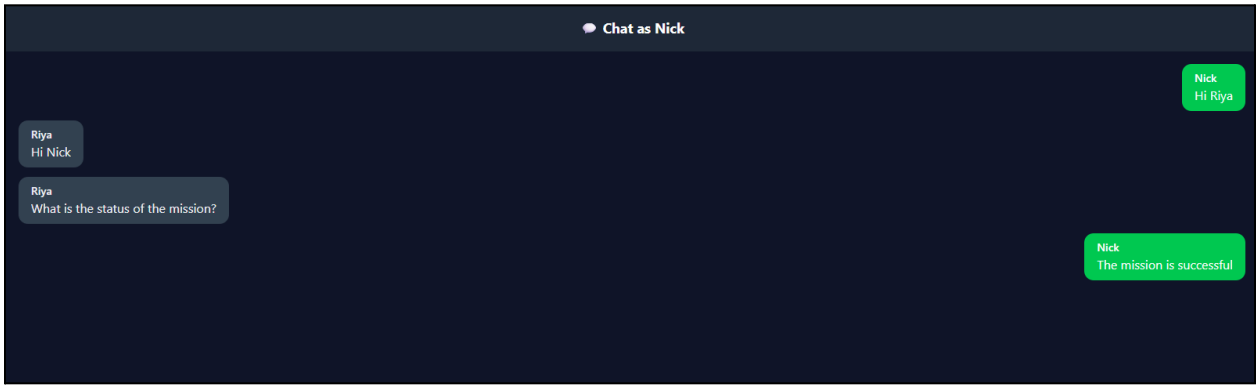

Fig 1.1 - Chat screen for User1

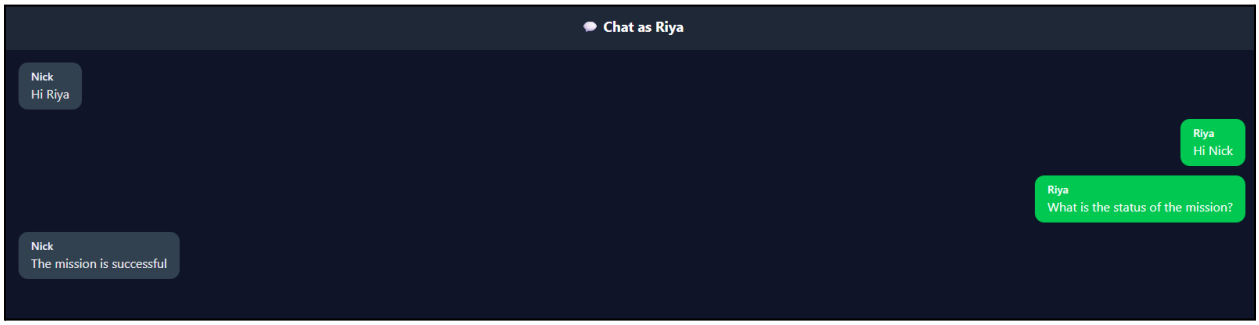

Fig 1.2 - Chat screen for user2

```
User connected: GhhUlAoGrMOpnM_fAAAQ
User connected: 64kqVUWwSL_f-jhiAAAR
Message received: { id: 1759983042305, sender: 'Nick', text: 'Hi Riya' }
Message received: { id: 1759983048177, sender: 'Riya', text: 'Hi Nick' }
Message received: {
  id: 1759983083032,
  sender: 'Riya',
  text: 'What is the status of the mission?'
}
Message received: {
  id: 1759983097281,
  sender: 'Nick',
  text: 'The mission is successful'
}
User disconnected: GhhUlAoGrMOpnM_fAAAQ
User disconnected: 64kqVUWwSL_f-jhiAAAR
```

Fig 2 - Terminal Logs