Training Day 3 Report

Date: 27 June 2025

Topic Covered: Real-Time Chat Basics with Socket.io

On Day3 of MERN-Stack chat app training, the focus shifted to implementing **real-time message communication** using WebSockets via **Socket.io**. This foundational step connects the frontend React app with the backend Node/Express server for instant chat updates.

Socket.io Overview

Socket.io enables real-time, bidirectional communication between client and server using WebSockets under the hood. Most MERN-based chat apps rely on it to deliver messages live without page reloads :contentReference[oaicite:1]index=1.

Essential concepts learned:

- **Server-side socket setup** – emit and listen for events.
- **Client-side socket setup** – connect and handle chat events.
- **Rooms or channels** – grouping users for individual or group chats.

Server-side Implementation

```
// server.js
const io = require('socket.io')(server);
io.on('connection', socket => {
  console.log('New client connected');
  socket.on('sendMessage', ({ userId, message }) => {
```

```
    io.to(userId).emit('receiveMessage', { message });
  });
  socket.on('disconnect', () => {
    console.log('Client disconnected');
  });
});
```

Client-side Implementation

```
// Chat.jsx (React component)
import io from 'socket.io-client';
const socket = io('http://localhost:5000');

socket.emit('sendMessage', { userId, message });
socket.on('receiveMessage', data => {
  setMessages(prev => [...prev, data.message]);
});
```

State Management Updates

Implement ‘useState‘ and ‘useEffect‘ to maintain chat state:

- ‘messages‘: a state array updated as ‘receiveMessage‘ fires.
- ‘useEffect‘ dependency on socket events for real-time rendering.

Connecting with Backend (Integration)

React app now actively communicates with Express backend:

- Sent messages are stored in MongoDB via REST API.
- Received messages are displayed live via realtime socket updates.

Outcome of the Day

- Established live message communication using Socket.io.
- Built foundational chat component in React with state updates.
- Set up backend to broadcast chat events to connected clients.
- Handled client disconnects and basic error logging.