Summary of Web Sockets

🔌 What are Web-sockets?

- WebSockets are a way for our website (frontend) and our server (backend) to communicate with each other **in real-time**—like a two-way conversation.
- In simple words, WebSockets will help us to build real-time conversation.
- Think of a WebSocket connection like a **phone call** between your browser and the server. Once the call (connection) is made:
  - 📞 **Browser → Server**: You can talk to the server.
  - 📞 **Server → Browser**: The server can also talk back anytime.

🔗 Connect to the socket

- For building the real-time connection between frontend and backend, frontend needs to connect with socket. After that connection only that real-time connection will be establish. For that we use socket.io library on both side(Frontend & Backend).

```
const server = http.createServer(app); //this app is express app
const io = new Server(server, {
  cors: {
    origin: "*",
    methods: ["GET", "POST"],
  },
});

io.on("connection", () => {
  console.log("A user connected");
});

server.listen(PORT, () => console.log(`Server is running on port ${PORT}...
`));
```

⚙️ Socket main methods

- There are main two methods of socket which **emit & on**.
  - **emit** is used to send event (frontend to backend or backend to frontend)
  - **on** is used to receive event (frontend to backend or backend to frontend)

```
// Frontend (Sending Message using emit method)
socket.emit("sendMessage", messageData);

// Backend (Receiving Message using on method)
socket.on("sendMessage", (messageData) => {
  // store message in database
  console.log("New message from frontend:", messageData);
});
```

🌀 Variations of socket methods

- There are 4 variations of socket emit method.
  - `io.emit("showTyping")` - send event to all connected users(include sender)
  - `socket.emit("showTyping")` - send event to sender only
  - `io.to("room").emit("showTyping")` - send event to all users only in that room(include sender)
  - `socket.to("room").emit("showTyping")` - send event to all users only in that room(but not sender)

💬 Logic of Chat Features

- Suppose Harley wants to chat with Mike. First Harley join a room in which they have chat(with chat_id).
- After that Harley send the message from frontend using emit method and in the backend we receive event and store the message in the database.

- After storing message to database, we will send the message to frontend using socket.
- Now here is two options:
  - Mike also available in the room
  - Mike is not available in the room
- Option 01 - if mike is available then we update message status to seen
- Option 02 - if mike in not available in the room then we set message status to sent(if mike is offline) or delivered(if mike is online). After that when mike join the room, then we update those message status to seen.
- Group Chat has also the same logic, only difference is in the deliveryStatus field. We have to store each person status.

The ultimate Node JS Course ~ Code Bless You❤️