EC 530

Software Engineering Principles

P2P Chat

Joel Franklin Stalin Vijayakumar

1) Tech Stack

The P2P chat application I've built uses the following tech stack:

HTML, **CSS**, and **JavaScript** for the **frontend**:

HTML is used for structuring the content and creating the user interface elements.
CSS is used for styling and applying visual effects to the HTML elements.
JavaScript is used for handling user interactions, managing WebSocket connections, and updating the UI based on received messages.

Node.js for the **backend**:

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine, which allows you to run JavaScript on the server-side.
In this P2P chat application, Node.js is used to create a WebSocket signaling server for exchanging messages between peers.

WebSocket protocol:

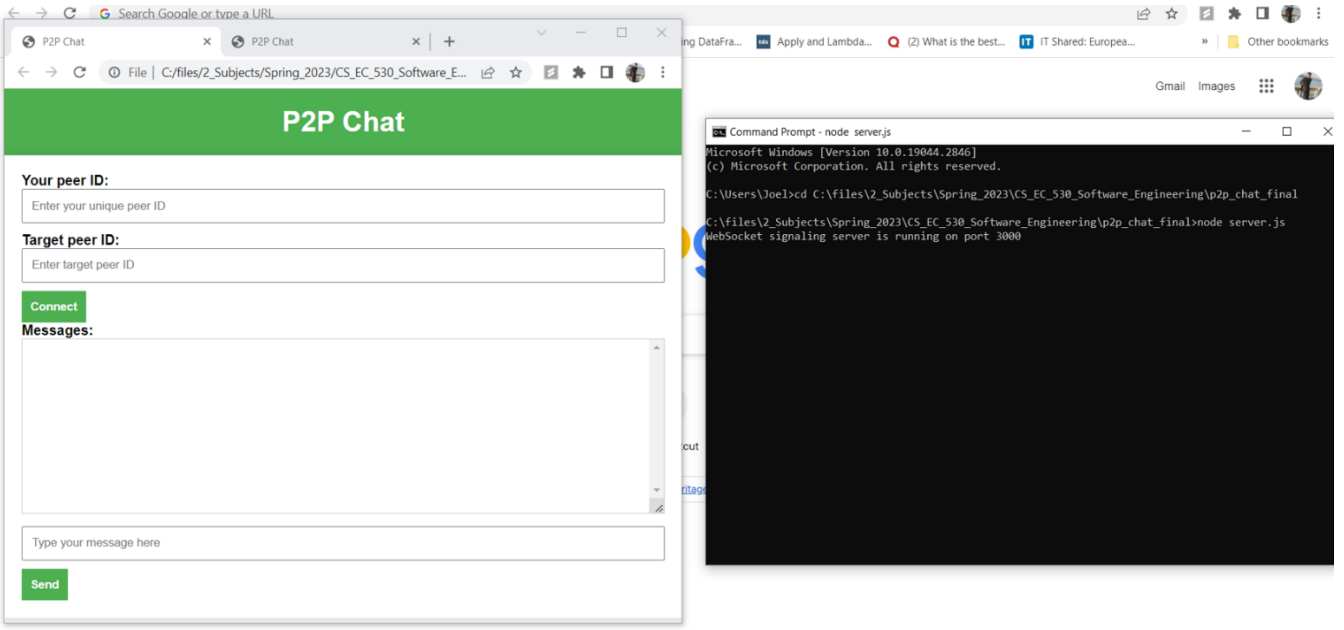
WebSocket is a communication protocol that provides full-duplex communication channels over a single TCP connection.
In this application, WebSocket is used for real-time communication between the frontend and the signaling server.
The signaling server acts as an intermediary that broadcasts messages to connected clients, enabling P2P communication between them.

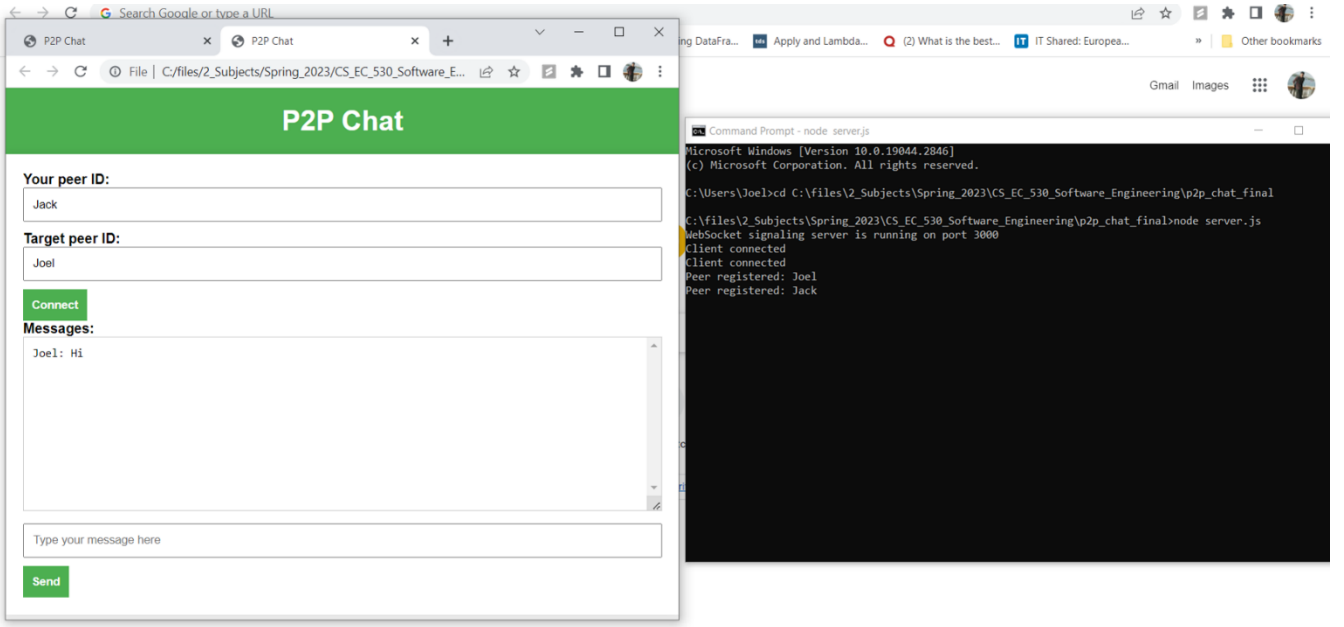
The 'ws' library:

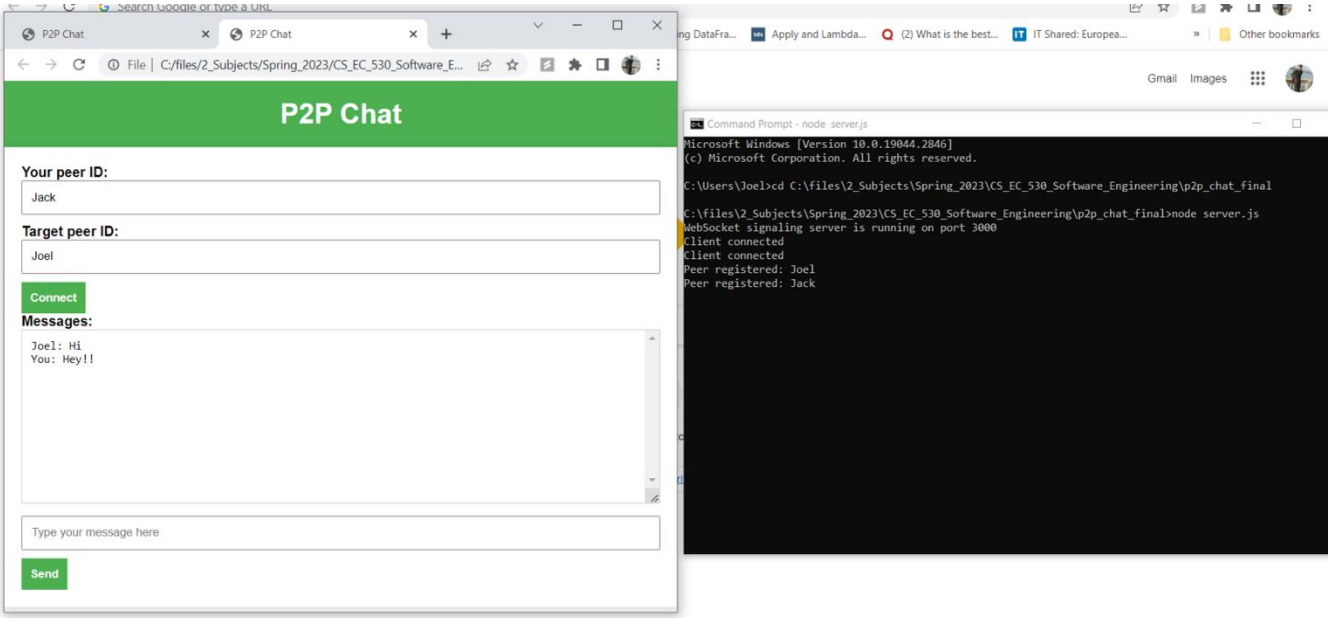
The 'ws' library is a WebSocket library for Node.js that simplifies creating, configuring, and handling WebSocket connections.
In this application, the 'ws' library is used to create a WebSocket server for the signaling server.

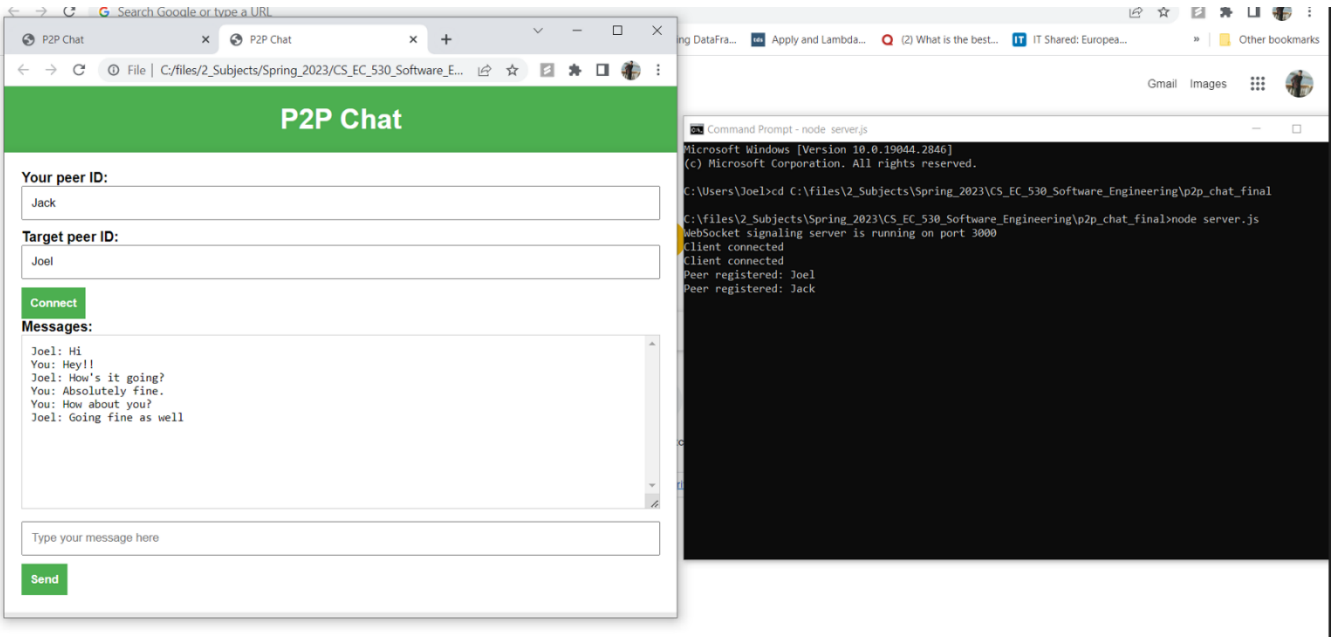
This tech stack allows you to create the P2P chat application where users can exchange text messages in real-time.

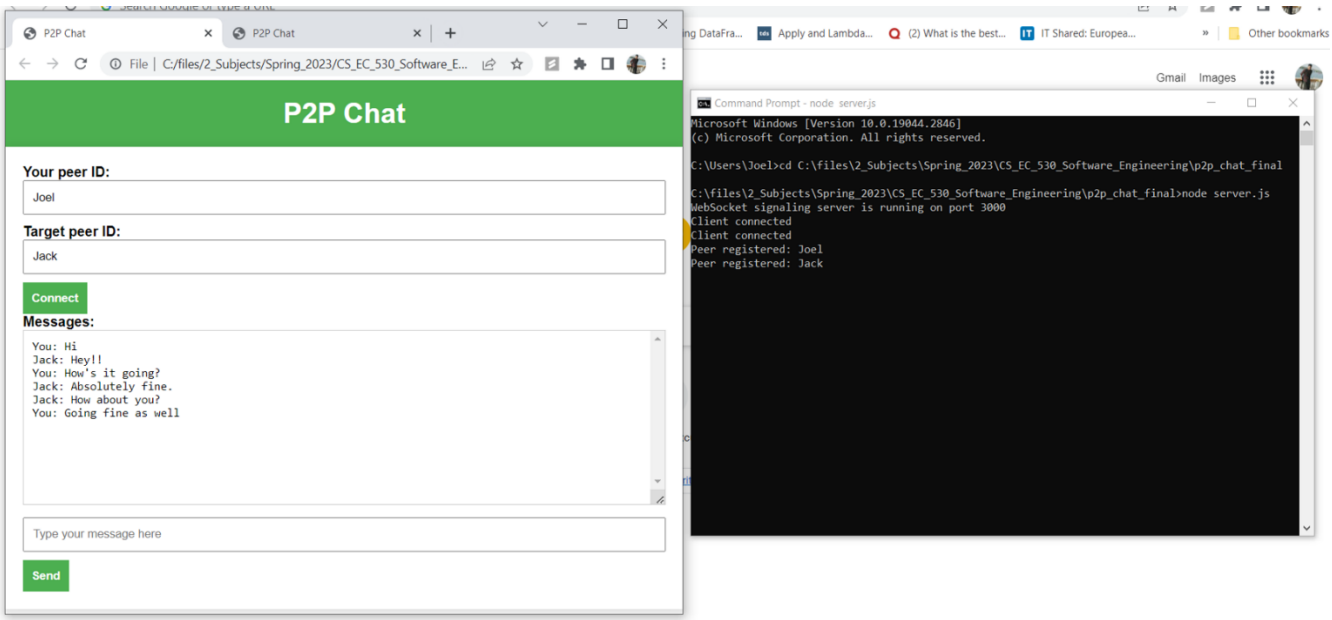
2) Screenshots of the P2P Chat application






P2P Chat
P2P Chat
File | C:/files/2_Subjects/Spring_2023/CS_EC_530_Software_E...
P2P Chat
Your peer ID:
Jack
Target peer ID:
Joel
Connect
Messages:
Joel: Hi
You: Hey!!
Type your message here
Send
Command Prompt - node server.js
Microsoft Windows [Version 10.0.19044.2846]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Joel>cd C:\files\2_Subjects\Spring_2023\CS_EC_530_Software_Engineering\p2p_chat_final
C:\files\2_Subjects\Spring_2023\CS_EC_530_Software_Engineering\p2p_chat_final>node server.js
WebSocket signaling server is running on port 3000
Client connected
Client connected
Peer registered: Joel
Peer registered: Jack


P2P Chat
P2P Chat
File | C:/files/2_Subjects/Spring_2023/CS_EC_530_Software_E...
P2P Chat
Your peer ID:
Jack
Target peer ID:
Joel
Connect
Messages:
Joel: Hi
You: Hey!!
Joel: How's it going?
You: Absolutely fine.
You: How about you?
Joel: Going fine as well
Type your message here
Send
Command Prompt - node server.js
Microsoft Windows [Version 10.0.19044.2846]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Joel>cd C:\files\2_Subjects\Spring_2023\CS_EC_530_Software_Engineering\p2p_chat_final
C:\files\2_Subjects\Spring_2023\CS_EC_530_Software_Engineering\p2p_chat_final>node server.js
WebSocket signaling server is running on port 3000
Client connected
Client connected
Peer registered: Joel
Peer registered: Jack



3) Key features of the P2P Chat Application

- **Unique Peer ID**: Each user can create a unique peer ID for themselves. This ID is used to identify and address the users within the chat system.

- **Connection establishment**: Users can input the target peer ID of the user they want to communicate with and click on the "Connect" button to establish a connection. The signaling server helps to exchange messages between peers and set up a connection.

- **Real-time text messaging**: Once a connection is established, users can send and receive text messages in real-time. Messages are displayed in the chatbox with the sender's name (either "You" or the sender's peer ID) and the message text.

- **Simple user interface**: The application has a minimalistic user interface that includes input fields for peer IDs, a chatbox for displaying messages, and buttons for connecting and sending messages.

- **WebSocket-based communication**: The application uses the WebSocket protocol for real-time communication between the client and the signaling server. This enables low-latency and efficient communication between peers.

4) Features that can be added

Here are some features that can be added to enhance the P2P chat application:

- **Chat history and persistence**: By saving chat history in a local or remote database, users can access previous conversations and continue where they left off. This feature could also include searching and filtering capabilities to help users find specific messages.

- **File sharing**: Allow users to share files, such as images, documents, and videos, within the chat. This can be achieved by implementing file transfers using WebRTC's data channels or a third-party file hosting service.

- **Group chats**: Enable multiple users to join a single chat room, allowing them to communicate simultaneously. This feature would require an updated UI to manage group chats and additional logic to handle multiple connections.

- **Audio and video calling**: Implement peer-to-peer audio and video calling using WebRTC. This would allow users to make voice or video calls directly from the chat application without relying on external services.

- **Typing indicators and read receipts**: Display typing indicators when a user is composing a message and show read receipts when a message has been seen by the recipient. This helps create a more interactive and engaging chat experience.

- **Customizable user profiles**: Allow users to personalize their profiles with information like profile pictures, display names, and status messages. This would create a more personalized and engaging experience for users.

- **Emojis and rich text formatting**: Enhance the messaging experience by supporting emojis and rich text formatting (e.g., bold, italics, and hyperlinks). This could be implemented using a rich text editor library or a custom input field.

- **Notifications**: Implement desktop and mobile notifications to alert users when they receive new messages or incoming calls. This ensures that users don't miss important updates, even when they're away from the chat application.

By adding these features, the P2P chat application can become a more robust and feature-rich communication platform that rivals popular messaging apps on the market.