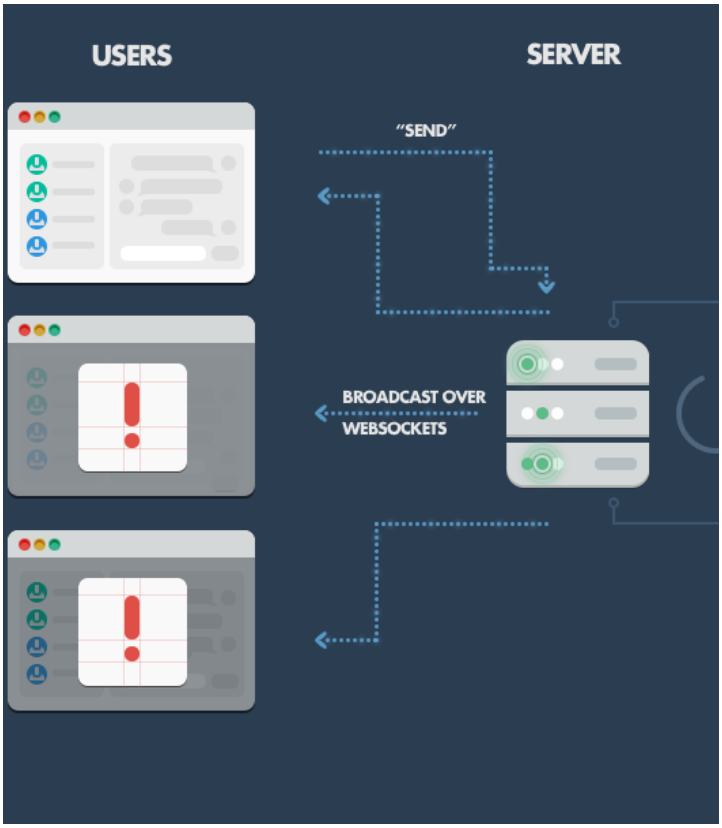
When Node.js Should be Used

CHAT

Chat is the most typical real-time, multi-user application. From IRC (back in the day), through many proprietary and open protocols running on non-standard ports, to the ability to implement everything today in Node.js with websockets running over the standard port 80.



The chat application is really the sweet-spot example for Node.js: it's a lightweight, high traffic, data-intensive (but low processing/computation) application that runs across distributed devices

Depiction

In the simplest example, we have a single chatroom on our website where people come and can exchange messages in one-to-many (actually all) fashion. For instance, say we have three people on the website all connected to our message board.

On the server-side, we have a simple Express.js application which implements two things:

1. A **GET /** request handler which serves the webpage containing both a message board and a 'Send' button to initialize new message input, and
2. A websockets server that listens for new messages emitted by websocket clients

On the client-side, we have an HTML page with a couple of handlers set up, one for the 'Send' button click event, which picks up the input message and sends it down the websocket, and another that listens for new incoming messages on the websockets client (i.e., messages sent by other users, which the server now wants the client to display).

When one of the clients posts a message, here's what happens:

1. Browser catches the 'Send' button click through a JavaScript handler, picks up the value from the input field (i.e., the message text), and emits a websocket message using the websocket client connected to our server (initialized on web page initialization).
2. Server-side component of the websocket connection receives the message and forwards it to all other connected clients using the broadcast method.
3. All clients receive the new message as a push message via a websockets client-side component running within the web page. They then pick up the message content and update the web page in-place by appending the new message to the board.

Source: https://www.toptal.com/nodejs/why-the-hell-would-i-use-node-js