**socket-chat-app Guide**

Building a simple real-time chat app with Node.js and Socket.IO

**Part 1 Instructions**

Traditional web applications primarily used the HTTP request-response model, where clients sent requests to servers, and servers responded with data. However, implementing real-time features like live chat, notifications, collaborative tools, etc, was challenging. Developers had to resort to workarounds like long polling (repeatedly sending requests) or plugins such as Flash, to achieve real-time communication.

WebSockets changed the game by enabling constant, low-delay communication between clients and servers, breaking away from the old request-response model.

Socket.IO was introduced with the aim of simplifying real-time communication between servers and clients on the web. Socket.IO is built on top of WebSockets and allows developers to create real-time applications without worrying about low-level networking details.

In this article, we’ll explore the concept of using Socket.IO while creating a real-time chat application using Node.js + Socket.IO, that can be connected to any client-side application of our choice.

**Part 2 Project setup**

download the project folder, on /server folder, run the following command:

|  |
| --- |
| npm install |

**Part 3 Test the APIs**

To test our application, we need to start both the server and client.

Run the Node.js script to start the Socket.IO server:

|  |
| --- |
| node server.js |

Next, Open the index.html file in a web browser. It will connect to our Socket.IO server running on http://localhost:5000.

To create another client for exchanging chat messages, open a new browser window in incognito mode. This will initiate our application and create a new client user. Now, both clients can exchange messages.

