

Web-ChatLine: An Innovative Chatting Platform [†]

Amit Kumar Goel ^{*}, Shivang Gupta, Chandan Kumar Singh and Krishna Kant Agrawal

School of Computing Science and Engineering, Galgotias University, Greater Noida 203201, India; dilip66gupta@gmail.com (S.G.); chandan.cse.singh@gmail.com (C.K.S.); kkagrawal@outlook.com (K.K.A.)

^{*} Correspondence: amit.goel@galgotiasuniversity.edu.in

[†] Presented at the International Conference on Innovative Research in Renewable Energy Technologies, West Bengal, India, 16–17 March 2022.

Abstract: With the turn of events and improvement in the web, an ever-increasing number of individuals have been choosing online chat platforms for correspondence. Their applications include being used for correspondence over significant distances. Along these lines, the application must both be continuous and multi-stage, and utilized by numerous clients. The online ongoing talking application need not bother with any extra outsider customer program, and visual correspondence could be set up advantageously. The programming instrument utilized to build this application is React.js, Node.js, which has an express structure and an in-memory data set. The text correspondence is moved to and from servers. Furthermore, the information transmission is achieved through highlight point association between servers. Because of the use of the response structure, a virtual space idea is carried out, which improves the presentation over existing applications created utilizing PHP by many times.

Keywords: real time; chat; web; message; room; group chat



Citation: Goel, A.K.; Gupta, S.; Singh, C.K.; Agrawal, K.K. Web-ChatLine: An Innovative Chatting Platform. *Mater. Proc.* **2022**, *10*, 6. <https://doi.org/10.3390/materproc2022010006>

Academic Editors: Sudipta Das, K. Vasu Babu, Samrat Paul and Kunal Chakraborty

Published: 22 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Web Chat application is an element or a program on the Internet used to impart information straightforwardly among Internet clients who are on the web or who are similarly utilizing the web. Talk applications permit clients to convey information from a distance. Thus, this talk application should be ongoing and multi-stage in order to be utilized by numerous clients. The improvement of data and correspondence innovations is quickly achieved through in-memory databases. Assembling this application starts with the assortment of significant information that will be shown on the web and versatile adaptations. The programming language used to construct workers is Node.js with express structure [1,2]. Users can chat in real time in a room by simply signing in and choosing which room they want to join. The client can share messages and files in real time within the room. Internal memory databases will store data that rely primarily on memory for data storage, in contrast with databases that store data on disks or Solid State Device (SSD) In-memory information stores are intended to empower negligible reaction times by eliminating the need for circles. Since all information is put away and overseen solely in the primary memory, in-memory data sets risk losing information upon an interaction or server disappointment [3]. In-memory data sets can maintain information on plates by putting away every activity in a log or by taking pictures.

2. Literature

2.1. Problem Statement

- The aim of this study is to create a talk application with a server and clients to empower clients to call one another.
- The study aims to foster a solution to empower clients to flawlessly speak with one another.

- The undertaking ought to be exceptionally simple to empower even a novice to utilize it.
- This undertaking can assume a significant role in authoritative fields where representatives can interface through LAN.
- The primary motivation behind this task is to provide multiple useful forms of communication through the network.

2.2. Innovative Ideas of the Project

- GUI: easy to utilize GUI (Graphical User Interface), henceforth any client with negligible information on working a framework can utilize the product.
- Stage freedom: the courier works on any framework heedless of the hidden working framework.
- Limitless customers: “N” number of clients can be associated with no sign of corruption of the server.

2.3. Project Objective

- Correspondence: to foster a texting platform to empower clients to consistently speak with one another.
- Ease of use: The venture ought to be exceptionally simple, allowing even a novice to utilize it.

2.4. Scope of The Project

- The broadcasting chat server application will be a text correspondence program; it will actually want to convey information between two PCs utilizing point-to-point correspondence.
- The restriction of Live Chat is it does not uphold voice chat. To overcome this limitation, we are simultaneously dealing with increasing advancements.
- Organizations might want to have a correspondence program wherein they can impart information right away inside their association [4].
- The way that the product utilizes an inward organization arrangement inside the association makes it extremely secure from outside assaults.

2.5. What Is Express.js?

- Express is an unimportant and adaptable Node.js web application system that gives a hearty arrangement of highlights for web and portable applications. It is an opensource structure created and maintained by the Node.js establishment.
- Express provides the apparatuses that are needed to fabricate our application, be it a single-page, multi-page or crossover web application. It is adaptable, as there are various modules accessible on npm (Node Package Manager), which can be straightforwardly connected to Express [5,6].
- Not at all like its rivals, such as Rails and Django, which have an obstinate method of building applications, Express has no “most ideal way” to accomplish something. It is entirely adaptable and pluggable.
- Pug (prior known as Jade) is a succinct language used to compose HTML formats. It produces HTML and upholds dynamic code and code reusability (DRY). It is perhaps the most well-known layout language utilized with Express.
- Express can be considered as a layer based on the highest point of the Node.js that deals with a server and courses. It permits clients to arrange middleware to react to HTTP Requests and characterizes a directing table which is utilized to perform various activities dependent on HTTP strategy and URL.
- Express permits powerful delivery of HTML pages dependent on passing contentions to formats.
- Express is offbeat and single strung and performs I/O tasks rapidly.

Why Utilize Express?

- It has super quick I/O.
- It is nonconcurrent and single strung.
- It has a MVC-like design.
- Its hearty API(Application Programming Interface) makes steering simple.

2.6. What Is React?

- ReactJS is an explanatory, effective, and adaptable JavaScript library for building reusable UI parts. It is an open source, part-based front-end library which is capable just for the view layer of the application. It was at first evolved and maintained by Facebook, and later utilized in applications such as WhatsApp and Instagram.
- A ReactJS application is comprised of different parts, and every part is answerable for yielding a little, reusable piece of HTML code. The parts are the basics of all React applications. These components can be settled with different parts to permit complex applications to be worked on via straightforward structure blocks. ReactJS utilizes a virtual DOM-based system to fill information in HTML DOM. The virtual DOM works quickly as it just changes individual DOM components as opposed to reloading total DOM without fail.
- Rather than utilizing customary JavaScript, React codes are sent in something many refer to as JSX (JavaScript Syntax Extension). JSX is fundamentally a sentence structure augmentation of ordinary JavaScript, and it is utilized to make React components. These components are then delivered to the React DOM. JSX is quicker than typical JavaScript as it performs improvements while using standard JavaScript.

Why Use React?

- It makes use of virtual DOM, which is a JavaScript object. This will improve applications execution, since JavaScript virtual DOM is quicker than the customary DOM.
- It can be utilized on the customer and server side, just as with different structures.
- Its part and information designs further develop comprehensibility, which assists with keeping up with bigger applications.

2.7. What Is Node.js?

- Node.js is an exceptionally amazing JavaScript-put together stage that works with respect to Google Chrome's JavaScript V8 Engine. It is utilized to foster I/O escalated web applications such as video web-based locales, single-page applications, and other web applications. Node.js is open source, totally free, and utilized by a large number of engineers all over the planet.
- Node.js is a server-side stage build on Google Chrome's JavaScript Engine (V8 Engine). Node.js was fabricated by Ryan Dahl in 2009.
- Node.js applications are written in JavaScript and can be run inside the Node.js runtime on OS X, Microsoft Windows, and Linux.
- Node.js additionally gives a rich library of many JavaScript screens which works on the advancement of web applications utilizing Node.js by and large.

Features of Node.js:

1. Incredibly quick: Node.js is based on Google Chrome's V8 JavaScript Engine, so its library is extremely quick at code execution.
2. I/O is Asynchronous and Event Driven: all APIs of Node.js library are offbeat; for example, non-hindering. Along these lines, a Node.js-based server never trusts that an API will bring information back. The server moves to the following API subsequent to calling it, and a warning instrument of Events of Node.js assists the server with getting a reaction from the past API call. It is additionally an explanation that it is exceptionally quick.

3. It is single strung: Node.js follows a solitary strung model with occasion circling.
4. It is exceptionally scalable: Node.js is profoundly versatile in light of the fact that occasionally, the instrument assists the server with reacting in a non-obstructing way.
5. There is no buffering: Node.js chops down the general handling time while transferring sound and video documents. Node.js applications never cushion any information. These applications just result the information in lumps.
6. It is open source: Node.js has an opensource local area that has delivered numerous incredible modules to add extra capacities to Node.js application.

3. Comparisons

The features of the app are compared in below Table 1 with the other chatting apps. This includes many changes in the app which make it different from others. The most interesting thing is the chat in real time with the in-memory database, through which we can accept this app in our daily life.

Table 1. Comparison between Web Chat line and other chatting app.

Title 1	Title 2	Title 3
Login Control	Can be accessed in a particular group.	Cannot be accessed by a particular group.
Live Users status	Users' access and exit from the group is known in real time.	Not possible to view in real time.
Data Base	An in-memory database is used, so that we can have backup instantly without the internet.	A cloud database is used so we cannot access data instantly without the internet.
Specific Group	In real time, the same user cannot access more than one group until he leaves the first group.	In real time, the same user can access more than one group, due to users not knowing the current user status.

4. Product Overview

The usefulness of the Chat Application is to enable visitation with whoever is online on the application. The clients and partners will gather until further notice, the utilization cases will be what is accessible to the client, and the useful/non-functional necessities will be covered, just like the achievements of the talk application.

The Architecture of Web Chat app shown in Figure 1 have the following features

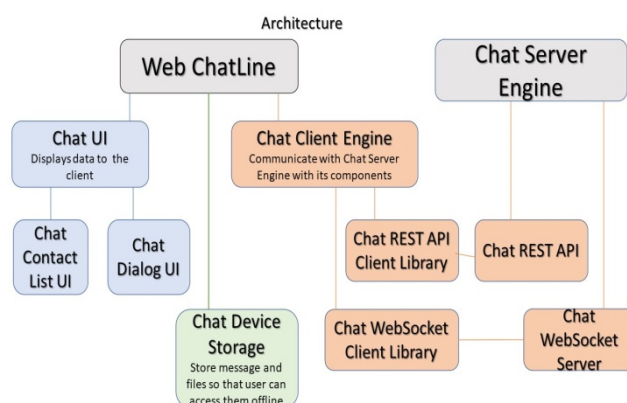


Figure 1. Architecture of Web Chat app.

Clients and Stakeholders. This segment will manage the clients and partners. The customers will use the discussion application and their accomplices will make, stay aware of, and test the visitation application.

The team and myself—We will be making, keeping up with, and testing the talk application through its periods of improvement.

Customers—The customer will be any person who has the visit application and registers for it.

- The framework to be created here is a Chat office. It is a merge framework. It is a Client-Server system with a united data base server. All nearby customers are associated with the incorporated server by means of LAN.
- There is a two-way correspondence between various customers and the server. This talk application can be utilized for group conversation. It allows customers to find other endorsements in the customer's interface.
- This application associates with the customer through G.U.I. The interface is straightforward, simple to deal with, and plain as day.
- Once opened, the client will effectively come into the stream with the application and effectively utilizes all interfaces appropriately. Be that as it may, the essential interface is accessible in our application.
- Title board.
- Message board.
- Contact list board.

Useful and Non-Functional Requirements:

Useful Requirements.

Client Registration: Users should have the option to enroll for the application through a Login Credential. When application opens, client/s should have the option to join themselves, or they can directly login if they have a record as of now. In the event that client avoids this progression, client should be ready to talk. The client's email will be the novel identifier of his/her record on the Chat Application [7,8].

Adding A New Member: The application should recognize all contacts from the server data base. Assuming any of the contacts have clients entered with Chat Application, those contacts should consequently be added to the clients contact list on Chat Application.

Send Message: The user should have the choice to send message to any contact on his/her Chat Application contact list. The customer should be told when the message is adequately passed on to the recipient by concealing the message.

Broadcast Message: The user ought to have the option to make groups of contacts. The customer should have the choice to convey messages to these social affairs.

Non-Functional Requirements.

Security: Messages divided among clients ought to be scrambled to keep up with protection.

Execution: The application should be lightweight and should send messages as shown in Figure 2.

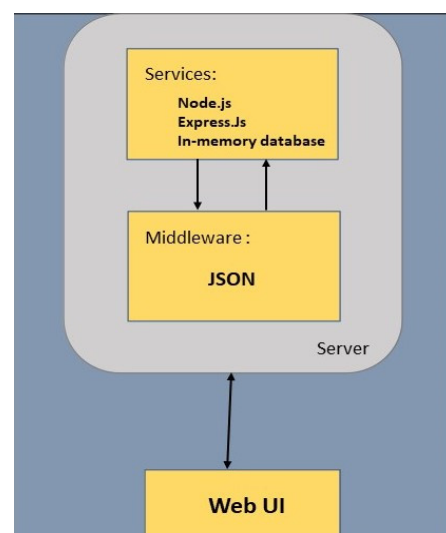


Figure 2. Application of Weblines chat app.

5. Product Design

A chat fuses two essential parts

- Visit App or client part, which is a Web talk application. Encourage React.
- The Talk Server Engine or server part, which is a pool of external servers responsible for the discussion advancement.

The two regions contain different parts that award to one another and bring the visit straightforwardly into it.

Talk App or Client Side.

The Visit App is the other basic piece of the conversation planning, the one that clients straightforwardly take an interest in.

It is secluded into two separate root parts:

- Visit Client Engine handles all the correspondence with the Chat Server Engine through its interior parts: a Chat REST API Client Library other than Chat WebSocket Client Library.
- Visit UI shows information to clients: Chat Contact List

UI, Chat Dialog UI.

Part:

Parts are the development squares of any React application.

Moreover, a standard React application will have an amazing piece of these. On a very basic level put, a segment is a JavaScript class or breaking point that then again perceives inputs, for example, properties(props), and returns a React part that portrays how a piece of the UI (User Interface) ought to show up. App.js is the beginning stage of our React application.

A package.json file:

- Records the social events your Endeavor depends on.
- Picks changes of a social event that your Endeavor can use.
- Makes your gathering reproducible, and consequently clearer to provide for different draftsmen.

A package.json record might give off an impression of being like this:

Visit Server Engine.

This is a point of convergence of the visit planning that handles message transport. In our variant of visit planning, it unites with components

Talk REST API handles the errands that are not obviously related to message forwarding and advancement, such as customer support, changing customer settings, welcoming friends, downloading sticker packs, etc.

- The Chat App (the conversation customer part) passes on with the Chat REST API through the Chat REST API Client library.
- Visit WebSocket Server is at risk of sharing messages between customers. The Chat App converses with the Chat WebSocket Server through the Chat WebSocket Client Library. This connection is open in two exceptional ways; it proposes clients do not need to make deals to the server in the event that there are any messages for them; they basically move them immediately.

6. Dependencies

The untouchable pack or modules introduced utilizing npm are displayed in this segment. The package.json record is the focal point of Node.js framework. It is the show record of any Node.js project and contains the metadata of the endeavor. The package.json report is the fundamental part to get a handle on, learn, and work with the Node.js. It is the fundamental development to find concerning progress in Node.js. Figure 3 shows sample coding of Server.js file and Figures 4 and 5 shows module and package coding.

```

JS server.js > io.on('connection') callback > socket.on('disconnect') callback
1  const path = require('path');
2  const http = require('http');
3  const express = require('express');
4  const socketio = require('socket.io');
5  const formatMessage = require('./utils/messages');
6  const {
7    userJoin,
8    getCurrentUser,
9    userLeave,
10   getRoomUsers
11 } = require('./utils/users');
12
13 const app = express();
14 const server = http.createServer(app);
15 const io = socketio(server);
16
17 // Set static folder
18 app.use(express.static(path.join(__dirname, 'public')));
19
20 const botName = 'ChatChord Bot';
21
22 // Run when client connects
23 io.on('connection', socket => {
24   socket.on('joinRoom', ({ username, room }) => {

```

Figure 3. Server .js file.

```

.
├── .vscode
│   └── settings.json
├── node_modules
├── public
│   ├── css
│   │   └── style.css
│   └── js
│       ├── main.js
│       ├── chat.html
│       └── index.html
├── utils
│   ├── messages.js
│   └── users.js
├── package-lock.json
├── package.json
└── server.js

```

Figure 4. Different module.

```

() package.json > {} dependencies > socket.io
1  {
2    "name": "chatchord",
3    "version": "1.0.0",
4    "description": "Realtime Chat app with rooms",
5    "main": "server.js",
6    "scripts": {
7      "start": "node server",
8      "dev": "nodemon server"
9    },
10   "author": "Shivang Gupta",
11   "license": "MIT",
12   "dependencies": {
13     "express": "^4.17.1",
14     "moment": "^2.29.1",
15     "socket.io": "^4.3.2"
16   },
17   "devDependencies": {
18     "nodemon": "^2.0.15"
19   }
20 }
21
22

```

Figure 5. Packages installed.

7. Screenshots

These are the screenshots of the web chatline app in Figure 6 and in Figure 7 interaction page is shown.

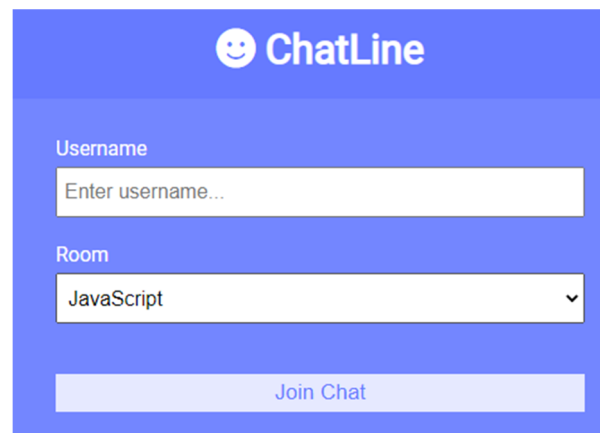


Figure 6. Web chatline—login page.



Figure 7. Web chatline—interaction page.

8. Conclusions

React JS is one of the most famous and incredible front-end innovations in the world today. It offers magnificent execution in any application. Furthermore, React JS offers great similarity in various stages, programs, and gadgets. Subsequent to fostering the Chat App, obviously React JS is simple to learn and helpful to execute. Firebase assumed an essential part in giving continuous information base backend administration. It is simple to utilize in light of the fact that additional code composition for the server is not needed while it is an instant API. Visual Studio Code was a superb IDE in this venture. I conclude by saying that the proposed application is a basic task; however, it needs improvement later on the off chance that the engineer has a chance to invest energy into running the tests. It very well may be construed that the talk application created utilizing Node.js, React and in-memory is quicker progressively, with a speed under a second looked at in the application created utilizing PHP and MySQL. Node.js is many times quicker than PHP (by framework time) Furthermore, it is more proficient than PHP in terms of RAM use.

9. Future Plans

We want to add more features such as Whatsapp, Telegram, etc. We want to add videocall and voice call features. We want to add a chat filtration facility to our project, by which we can filter the chats of a particular person. Our chat app is unique when we add chat filtration because popular chatting apps such as Whatsapp, and Telegram do not have chat filtration option.

This post is intended to give a stop to their pursuit, as I take care of the multitude of required key parts of how to make an ongoing informing application for portable and web applications with highlights and functionalities.

Author Contributions: Conceptualization, A.K.G. and K.K.A.; Methodology and Software, S.G. and C.K.S. All authors have read and agreed to the published version of the manuscript.

Funding: No External Funding.

Institutional Review Board Statement: Study not required approval.

Informed Consent Statement: No need of any consent in study.

Data Availability Statement: Study did not report any data.

Conflicts of Interest: No conflict of interest.

References

1. Kiessling, M. *The Node Beginner Book*; Lulu Press: Morrisville, NC, USA, 2012.
2. Teixeira, P. *Professional Node.js: Building Javascript Based Scalable Software*; John Wiley & Sons: Hoboken, NJ, USA, 2012.
3. Sidik, B.; Pohan, H.I. *Pemrograman Web Dengan HTML*, 2nd ed.; Informatika: Bandung, Indonesia, 2010.
4. Purnomosidi, B. *Penbangan Sistem Informasi Penegelolaan Inventaris Barang Divisi Pustekin Berbasis Web*; Bandung: Politeknik-Telkom: Bandung, Indonesia, 2013.
5. Heitkötter, H.; Majchrzak, T.A.; Ruland, B.; Weber, T. Evaluating Frameworks for Creating Mobile Web Apps. In Proceedings of the 9th International Conference on Web Information Systems and Technologies (WEBIST 2013), Aachen, Germany, 8–10 May 2013; pp. 209–221.
6. Brodsky, I. *Wireless Computing: A Manager's Guide to Wireless Networking*; Van Nostrand Reinhold Company: New York, NY, USA, 1997.
7. Rosa, A.S.; Shalahuddin, M. *Pemrograman J2ME Belajar Cepat Pemrograman Perangkat Telekomunikasi Mobile*; Informatika Publishing: Bandung, Indonesia, 2008.
8. Goel, A.K.; Pengoria, A.; Kumar, A.; Agrawal, K.K. Composite Movie Recommendation System. In Proceedings of the 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS 2022), Coimbatore, India, 25–26 March 2022; pp. 1273–1278.