SOCKET SYNC STUDIO

RANJITH KUMAR P[1], MUHAMED ABID S[2], PRABHU P S[3]

[1]Student, Information Technology, Bannari Amman institute of Technology, Tamil Nadu, India
[2]Student, Information Technology, Bannari Amman institute of Technology, Tamil Nadu, India
[3]Assistant professor, Information Technology, Bannari Amman Institute of Technology, Tamil Nadu, India

ABSTRACT

Socket.IO provides a reliable and scalable framework that enables clients and servers to establish persistent connections, ensuring a seamless flow of data and reducing latency to imperceptible levels. But at present it is hard to provide support the customizable broadcast messaging service with appropriate text suggestion. The application's architecture is based on the Client-Server model, where clients connect to a central server that handles data transmission between participants. Each user interacts with the workspace through a web-based interface, accessing real-time updates on shared documents, collaborative whiteboards, and instant messaging features. The main feature of the application is to provide customizable broadcast message and text auto generation based on Natural Language Processing. Socket.IO ensures that any changes made by one user are propagated instantly to all connected users, fostering a collaborative environment where teams can work together as if they were in the same physical location. These features are seamlessly integrated using Socket.IO's event-driven paradigm, allowing users to communicate and collaborate in a natural and intuitive manner. Additionally, the application incorporates user authentication and access controls to maintain data privacy and restrict unauthorized access.

Keyword: - *Broadcast messaging, Socket.io, React, JavaScript, npm, customizable broadcast messages, text auto-generation, NLP*

1. INTRODUCTION

The project aimed to leverage Google Colab's collaborative workspace environment to implement a real-time communication system using SocketSync Studio. The report outlines the successful integration of SocketSync Studio within the Colab environment, enabling seamless bidirectional communication between multiple users. Through Socket.io's event-driven architecture, users could exchange messages, share data, and collaborate in real-time within Colab notebooks. This integration facilitated collaborative coding sessions, interactive data analysis, and distributed computing tasks. Furthermore, the project highlighted the versatility and accessibility of Colab as a platform for collaborative software development and experimentation, demonstrating the potential for innovative solutions within cloud-based collaborative environments. Overall, the project underscored the efficacy of combining SocketSync Studio with Colab, opening up new avenues for collaborative research, development, and learning. SocketSync Studio allows you to push updates to connected clients in real-time. This is essential for applications where immediate information dissemination is crucial, such as chat applications, live notifications, or collaborative tools. This new facility can be useful in scenarios where different types of messages need to be delivered with specific formatting or styling. Node-NLP package is used for showing text auto generation with the user's inputs. The input's changes will be continuously watched and their respective auto generated text will be displayed above the textbox.

The application is designed in such a way that it follows Client-server method. NLP is used for giving text auto generation based on user's prompt input. React JS is used for building a very good front-end and node server will be built and MongoDB is used to store the information. Socket.io is used for providing real-time connection. The react application is built using NPX command. he application is designed in such a way that it follows Client-server method. NLP is used for giving text auto generation based on user's prompt input. React JS is used for building a very good front-end and node server will be built and MongoDB is used to store the information. Socket.io is used for providing real-time connection.

The react application is built using NPX command. The component based architecture and Virtual DOM in React JS provides a very good advantages for the application as it does not require to refresh for updating chats. React works based on JSX syntax which combines HTML, CSS and JavaScript together. This application aims to enhance Communication within the workspace by offering chat and notification features. It allows users to send real time messages, updates and notifications promoting collaboration. Node JS server which uses single thread and event based architecture will be very effective in this situation. Working with web sockets provides stateful connection between client and server. Socket.io does not have any built in facility for providing customizable broadcast messages. We will be building a package which will help to provide this facility to be integrated with the applications.

2. SCOPE

The real-time communication capabilities offered by SocketSync Studio enable seamless collaboration among multiple users working on Colab notebooks simultaneously. This opens up possibilities for team projects, pair programming sessions, and remote collaborative coding workshops, fostering a more dynamic and engaging learning or working environment. Additionally, the integration of Socket.io could facilitate interactive data analysis and visualization, allowing users to share insights and collaborate on complex datasets in real-time. Furthermore, the combination of Colab and SocketSync Studio could support distributed computing tasks, enabling users to harness the power of multiple computing resources for parallel processing and accelerating computations. Overall, the scope for Workspace Colab using SocketSync Studio extends to enhancing colla00boration, interactivity, and productivity in various domains, including education, research, software development, and data science.

3. LITERATURE REVIEW

This chapter is the summary of all the literature surveys that are referred keenly for developing this project. These literatures are about web sockets, text auto generation (about NLP), TCP protocol and openai. We referred, studied and review all the relevant cases and discussed with our guide and identified the problem statement.

3.1 SECURE WIRELESS COMMUNICATION IN BROADCAST CHANNELS

Mohammed Adil Abbas, Hojin Song and Jun-Pyo Hong made a study on "Secure Wireless Communications in Broadcast Channels with Confidential Messages". This paper contains the details about how to control the explicit visibility of insecure endpoints and how to secure them. Also got that https encryption will help in security. This survey gives many information on the methods to encrypt the information that are transferred through the network. This broadcast messages send information to all the sockets at the same time. The payloads which are transferred from each sockets should be encrypted and transferred to other sockets securely. These payloads will be decoded at each socket when they receive. They prepared, presented and published this Journal of Software Engineering in ieee in vol: 7 with page no 170525 - 170533, They published it in 2019. There are some methods that were mentioned in the journal to secure the socket connection. Use HTTPS for your Socket.IO server to encrypt data in transit. This helps prevent eavesdropping on the Web Socket communication.

Jianli Sun, the author of the paper "TCP/IP Socket I/O Multiplexing Using a Hybrid Polling Event System", have given many details on how TCP/IP works. It is a stateful networking protocol which keeps the connection live until the connection is lost or aborted. This stable connection is helpful for two communications in the same network. A hybrid polling event system likely refers to a combination of different I/O multiplexing mechanisms to handle network communication. This approach can help in optimizing network applications by choosing the most suitable I/O multiplexing method for a particular situation. This is a Journal of Networking Engineering presented in 29th ieee conference paper, published in 2009.

3.2 TABLE TO DIALOG

Haihong E, Zecheng Zhan and Meina Song are the authors for the journal "Table-to Dialog: Building Dialog Assistants to Chat with People on Behalf of You". This literature gave many information on how to train the model using the correct prompt messages. This suggests that the dialog assistant is designed to represent and communicate on behalf of a user. It might be used in scenarios where a user wants the assistant to handle interactions or tasks with other individuals or entities, such as customer support inquiries, scheduling appointments, or making reservations. The correct prompt messages are mandatory for a good working text generator. Concepts like Machine learning and Natural Language Processing will be used for giving the texts. These texts will be generated by giving the respective prompts. This is a Journal in the field of Software Engineering published as vol: 8 in the year 2020.

3.3 Encryption

Libraries like OpenSSL or Node.js's built-in crypto module can be used to encrypt and decrypt messages. Only authorized clients should possess the necessary decryption keys to read broadcasted messages else the intruder can decode the message that are transmitted.

4. OBJECTIVES AND METHODOLOGY

This chapter contains the information about the objective of this project, how the project will be implemented and the methodology followed to implement the same. Multiple ideas were innovated and discussions were made with the guide, then decisions were made to use the specific methodology.

4.1 OBJECTIVES OF THE PROPOSED WORK

The goals of this study have been carefully developed based on a thorough analysis of the literature that has already been published and the methods were improvised, the gaps are filled in this project. One of the main objective of the project is to design and develop a robust application that enables users to
customizable broadcast messages to specific target audiences or groups, facilitating effective communication, engagement, and information dissemination. Also, the text autogeneration will be developed. The user just need to give the situation of keyword, then he will get the required result/results (sentences) to send to the channels/sockets.
The project also concentrates on multiple fields like, user-friendly interface, customization-options for messages, message templates, security privacy, scalability with testing and quality assurance. Finally, after developing the application, documentation will also be developed so that its users can identify the new features easily. To create an intuitive and user-friendly interface that allows users to easily compose and customize broadcast messages. Implement a range of customization options, including message content, formatting, media attachments, and scheduling, to cater to various communication needs. Offer pre-designed message templates that users can customize to save time and maintain consistency in branding and communication.

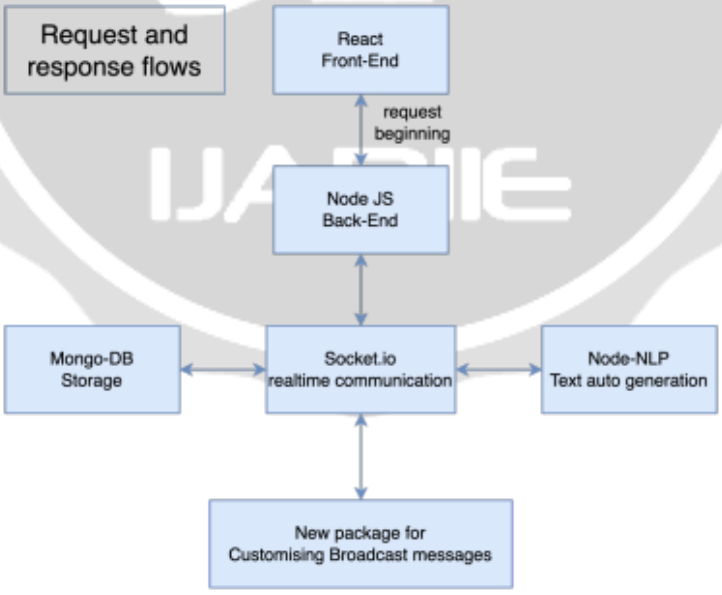

Figure 4.1 Request and Response Flows in Application Architecture

4.2 DEVELOPMENT OF REAL-TIME CHATTING

To establish a stable stateful connection, we have used socket.io package. The socket.io provides a web socket communication support i.e., it uses TCP connection to provide stateful connections. Once the connection is established, it will be terminated on our request. This library provides socket connections for clients and servers. When a user is connected to server, the server would be connected to other users. This is how the chatting is made live using web sockets. The following are the reasons for selecting Socket.io for providing real-time chats.

Real-Time Communication: Socket.io is a powerful library for enabling real-time bidirectional communication between the server and the client. It is specifically designed for scenarios where data needs to be pushed to clients instantly, making it ideal for applications like chat, online gaming, live notifications, and collaborative tools. This module aims to enhance communication within the workspace by offering chat and notification features. It allows users to send real time messages, updates and notifications promoting collaboration. Socket.io abstracts away the complexities of Web Sockets and provides a unified API that works across different browsers and devices. It gracefully falls back to other transport mechanisms like long polling when Web Sockets are not available, ensuring broad compatibility. Socket.io uses an event-driven architecture, similar to Node.js itself. This makes it intuitive to work with, as you can define custom events and event handlers on both the server and client, allowing for seamless and organized communication. Unlike traditional HTTP requests, which are typically initiated by the client, Socket.io allows for both the server and the client to initiate data transmission. This bidirectional data flow is essential for real-time applications where both parties need to exchange information at any given time. Socket.io can be easily scaled horizontally to accommodate a growing number of clients and connections. Setting up multiple instances of the server and using load balancing to distribute incoming connections, make it suitable for large-scale applications. Socket.io provides support for middleware, which allows to perform tasks like authentication, logging, and data validation before processing Web Socket messages. This adds security and custom logic to the real-time application. Socket.io offers a concept called "rooms," which allows us to group clients into specific channels or rooms. This feature is valuable for scenarios where we want to broadcast messages to a specific subset of clients, such as chat rooms or private conversations. Socket.io includes features for ensuring the reliability of messages, including acknowledgments and reconnection mechanisms. This helps maintain the integrity of data transmission even in less-than-ideal network conditions.

4.3 DEVELOPMENT OF NLP MODEL

The NLP model is used for providing custom text sentences based on the user's input. This feature is very useful for the user to generate automatic messages based on some keywords. In this project we are using node-NLP package to provide NLP support to the application. This package can help to easily integrate NLP features to the application and it takes huge amount of data to get trained accurately. After it being trained, when we give the input, the respective output will be obtained as output. The following are the advantages of using Node-NLP module in our project. Node-NLP offers a wide range of NLP features, including tokenization, part-of-speech tagging, sentiment analysis, entity recognition, and more. These features can be useful when analysing and processing of text before generating new content is required. Before generating text, it's often necessary to pre-process and clean the input text. Node-NLP can assist in cleaning and normalizing text data, making it more suitable for text generation algorithms. If the text auto-generation task requires generating text with a specific sentiment (e.g., positive, negative, neutral), Node-NLP's sentiment analysis capabilities can help ensure that the generated content aligns with the desired sentiment. Node-NLP can extract keywords and key phrases from text. This can be valuable when we want to ensure that the generated text is relevant to specific topics or keywords. If the text generation task involves incorporating named entities (e.g., names of people, places, organizations), Node-NLP can identify and extract these entities from the input text, helping us generate contextually relevant content. Node-NLP can detect the language of a given text, which can be helpful if we want to generate content in multiple languages or adapt the generated text to the user's preferred language. Node-NLP allows developer to train custom models for various NLP tasks. This means he can fine-tune the library to better suit your specific text auto-generation requirements. Since we are already using Node.js for our backend, integrating Node NLP can be relatively straightforward since it is designed to work seamlessly with Node.js. Node-NLP benefits from the Node.js and NLP communities, which means we can find documentation, tutorials, and community support to help us get started and address any issues we encounter while developing.

4. CONCLUSIONS

From the literature surveys, we have identified that most of the applications do not support customisable broadcast messages and we have created a new package to do the same. This package is published in npmjs.com as

open source project. The project also contains NLP model which is trained to get the inputs (keywords or situation) and provide the respective sentences as message to be sent. React JS is used for front-end and Node JS is used for back-end. This application also has the push-notification facility which will make the Workspace Collab user informed about the events happening in the environment. File sharing support is also present in the application which will enable the user to share files of almost every format to the other user and in the other end, second user can download the same file. The application will be hosted in the AWS EC2 instances and it will be configured in such a way that it will scale vertically and horizontally automatically. When the users count increases, it will automatically scale up to the requirement and do the necessary without any decrease in performance.

5. REFERENCES

[1]. Mohammed Adil Abbas; Hojin Song; Jun-Pyo Hong, (2019) "Secure Wireless Communications in Broadcast Channels With Confidential Messages" Journal of Software Engineering, vol : 7, https://doi.org/10.1109/ACCESS.2019.2955603

[2]. Jianli Sun, (2009), "TCP/IP Socket I/O Multiplexing Using a Hybrid Polling Event System" Journal of Networking Engineering, 29th ieee conference paper. https://doi.org/10.1109/ICDCSW.2009.24

[3]. Haihong E, Zecheng Zhan, Meina Song, (2020), "Table-to-Dialog: Building Dialog Assistants to Chat With People on Behalf of You", vol : 8. Journal of Software Engineering, https://doi.org/10.1109/ACCESS.2020.2998432

[4]. Hongzhou Yue, Shuilong He, Zhenghui Liu, (2020), "Social Media Users Send Promotional Links to Strangers: Legitimate Promotion or Security Vulnerability?" Journal Article, vol : 8, https://doi.org/10.1109/ACCESS.2020.2977101

[5]. Mengmeng Gong, Hui Song, Haoran Zhou, Bo Xu,(2022),"Enhancing Response Relevance and Emotional Consistency for Dialogue Response Generation", pp. 618-626, https://doi.org/10.1109/iSAI-NLP56921.2022.9960275

[6]. Ms. Kavita, Harveer Singh, (2023), "Utilizing Mixture Methods for Classifier in NLP: An Essential Consideration", International Conference on Artificial Intelligence and Smart Communication (AISC), https://doi.org/10.1109/AISC56616.2023.10085077

[7]. Raghav Jain, Tulika Saha, Souhitya Chakraborty, Sriparna Saha, (2022), "Domain Infused Conversational Response Gen, International Joint Conference on Neural Networks (IJCNN)", https://doi.org/10.1109/IJCNN55064.2022.9892890

[8]. Rui Yan, Juntao Li, Zhou Yu, (2022), "Deep Learning for Dialogue Systems: Chit-Chat and Beyond", https://ieeexplore.ieee.org/document/9800171