IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Real Time Chat Application

Mr.Sachin Bansal, Siddarth Dutt Sharma, Shahil Kumar Jha Sakshi Tomar, Roopali Pandey
[IIMT COLLEGE OF ENGINEERING, GREATER NOIDA]

Abstract

This project aims to build a real-time chat application using ReactJS as the front-end framework and Firebase as the back-end platform. The application will enable users to create accounts, log in, and join chat rooms to communicate with other users in real-time. The chat rooms will be implemented using Firebase's real-time database, which allows for quick and efficient updates as messages are sent and received.

The user interface will be designed using HTML and CSS, with a responsive design that adapts to different screen sizes and devices. The chat functionality will be implemented using JavaScript and ReactJS, with components such as message input fields, message displays, and user lists. The application will support features such as message deletion, user blocking, and notification alerts for new messages.

This project aims to provide a seamless and user-friendly chat experience for users, with real-time updates and minimal lag times. The application will be developed using modern web development tools and frameworks, with an emphasis on clean code, efficient database usage, and responsive design.

Introduction

Real-time chat applications have become increasingly popular in recent years, especially with the growing need for remote communication and collaboration. These applications provide users with the ability to communicate instantly with each other, regardless of their location. In this context, a real-time chat application built using technologies such as React.js, Firebase, HTML, CSS, and JavaScript can offer a highly responsive and dynamic user experience. In recent years, several technologies have emerged to support the development of these applications, including React.js, Firebase, HTML, CSS, and JavaScript.

- React.js, a JavaScript library for building user interfaces, provides developers with a powerful toolset to create complex applications with ease. Its declarative programming model and virtual DOM make it possible to efficiently update the user interface in real- time, without the need for manual DOM manipulation.
- Firebase, a mobile and web application development platform, provides developers with a wide range of tools and services to build real-time applications. It includes features like real-time database, authentication, cloud storage, and hosting, which makes it an excellent choice for building real-time chat applications.
- HTML and CSS provide the basic structure and styling of the application, while JavaScript is used for implementing various features and functionalities.

Together, these technologies offer a robust foundation for building a real-time chat application that is scalable, efficient, and user-friendly. Whether it's for personal or professional use, a real- time chat application can help people stay connected, collaborate more effectively, and streamline communication.

FACTS & STATISTICS

- Real-time chat applications are popular among businesses and organizations as a way to connect with customers and clients.
- React and Firebase are popular technologies for building real-time chat applications.
- Real-time chat applications can improve customer satisfaction and engagement, and can save time and money for businesses.
- Real-time chat applications are used in a variety of industries, including healthcare, e-commerce, and finance.
- Building real-time chat applications requires a strong understanding of web development technologies and user experience design.

Scope

1. User registration and login: Users will be able to create accounts and log in to the chat application. The application will provide secure authentication and session management.
2. Chat room creation: Users will be able to create chat rooms based on their interests or groups. The application will enable users to set privacy settings for chat rooms, such as password protection and invitation-only access.
3. Real-time messaging: Users will be able to send and receive messages in real-time without refreshing the page. The application will use Firebase Realtime Database to store and retrieve chat messages.
4. User blocking: Users will be able to block other users from messaging them. Blocked users will not be able to send messages to the user who has blocked them.
5. Notification alerts: Users will receive notifications when new messages are received. The application will provide different notification settings, such as sound and vibration alerts.
6. Message deletion: Users will be able to delete their own messages from the chat rooms. Deleted messages will be removed from the chat history and will not be visible to other users.
7. Message formatting: The application will support basic message formatting, such as bold, italic, and underline. Users will be able to customize the color and font of their messages.
8. User profiles: Users will be able to create profiles with their name, profile picture, and other details. The application will display user profiles when messages are sent and received.
9. Admin panel: The application will have an admin panel that enables administrators to manage users, chat rooms, and messages. The admin panel will provide features such as user ban and chat room deletion.
10. Responsive design: The user interface will be designed using HTML and CSS with a responsive design that adapts to different screen sizes and devices. The application will provide a seamless user experience across different platforms and devices.
11. Multi-language support: The application will support multiple languages, allowing users to select their preferred language for the user interface.
12. Hosting and deployment: The application will be hosted on Firebase Hosting and will be deployed using Firebase CLI. The application will be scalable, secure, and reliable.

Identification of Need /Background

The old manual system was suffering from a series of drawbacks. Since whole of the system was to be maintained with hands the process of keeping, maintaining and retrieving the information was very tedious and lengthy. The records were never used to be in a systematic order. there used to be lots of difficulties in associating any particular transaction with a particular context. If any information was to be found it was required to go through the different registers, documents there would never exist anything like report generation. There would always be unnecessary consumption of time while entering records and retrieving records. One more problem was that it was very difficult to find errors while entering the records. Once the records were entered it was very difficult to update these records.

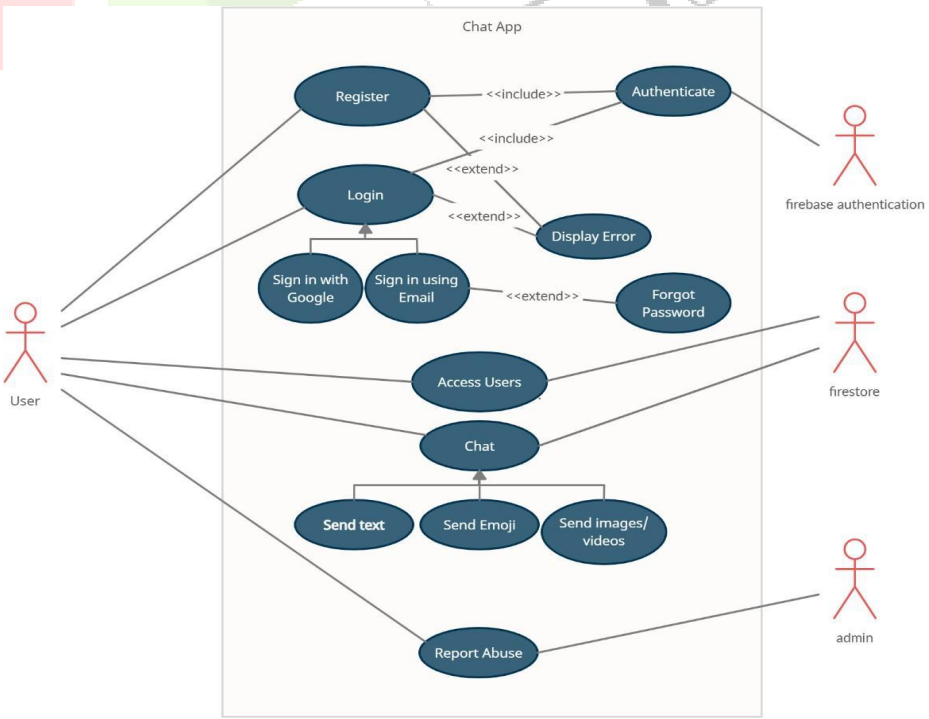
The reason behind it is that there is lot of information to be maintained and have to be kept in mind while running the business. For this reason we have provided features Present system is partially automated (computerized), actually existing system is quite laborious as one has to enter the same information at three different places.

Following points should be well considered:

- Documents and reports that must be provided by the new system: there can also be few reports, which can help management in decision-making and cost controlling, but since these reports do not get required attention, such kind of reports and information were also identified and given required attention.
- Details of the information needed for each document and report.
- The required frequency and distribution for each document.
- Probable sources of information for each document and report.

System Design

The application is built in two sides one is client and other is server. Client and Server connected bidirectional, when the client sends the message it goes to the server and when the server receives that message, server cannot do anything it could just print on terminal or do something else like dump it file into, but what we want to do is that make sure that everyone is connected to the chat room actually sees the message that this person has sent and at which time. We will bring other clients into mix and this time we send data from server to client. Server will send the client1 message to other clients that client 1 has sent the message, server will send that across to client and client can render that message through browser. In short when any client send the message it goes to server and then after receiving the message server sends it to all the other clients present in the room at that time.



1. When the user enter into the chat application, user login credentials is required. If the user doesn't have an account, then the user should signup first, In both login and
signup page user have option to sign up using Google or Facebook account.

2. After login, user have to enter the room name in which user can start a chat. Therefore, if user doesn't have the room name, user can create its own room name and invite theirfriends to chat with them in that room. After entering the room name user will be redirected to chat page.

3. There are two sides in the chat application in left side sidebar is given where user can see the room name in which they have joined and they can also see the connected online user at that time in chat room. Also, there is an option to join other room and option to logout.

Module of Real Time Chat Application

1. User authentication module: This module will handle the user registration and login process using Firebase Authentication. The module will allow users to create an account, log in with their email and password, and log out of the application. The module will also handle the user session management, ensuring that users remain logged in until they explicitly log out or their session expires.

2. Chat room module: This module will handle the creation and management of chat rooms. The module will allow users to create new chat rooms, join existing chat rooms, and view the list of available chat rooms. The module will also handle the privacy settings of chat rooms, such as password protection and invitation-only access.

3. Real-time messaging module: This module will handle the sending and receiving of messages in real-time using Firebase Realtime Database. The module will allow users to send text messages, images, and other media files. The module will also handle the formatting of messages, such as bold, italic, and underline.

4. User blocking module: This module will handle the blocking of users from messaging other users. The module will allow users to block and unblock other users, ensuring that blocked users cannot send messages to the user who has blocked them.

5. Notification module: This module will handle the notification alerts for new messages. The module will provide different notification settings, such as sound and vibration alerts. The module will also handle the display of notifications on different devices and platforms.

6. User profile module: This module will handle the creation and management of user profiles. The module will allow users to create a profile with their name, profile picture, and other details. The module will also handle the display of user profiles when messages are sent and received.

7. Admin panel module: This module will handle the administration of the chat application. The module will provide an admin panel that allows administrators to manage users, chat rooms, and messages. The module will provide features such as user ban and chat room deletion.

8. Responsive design module: This module will handle the responsive design of the chat application. The module will ensure that the user interface adapts to different screen sizes and devices, providing a seamless user experience across different platforms.

9. Multi-language support module: This module will handle the multi-language support of the chat application. The module will provide different language options for the user interface, allowing users to select their preferred language.

10. Hosting and deployment module: This module will handle the hosting and deployment of the chat application. The module will ensure that the application is hosted on Firebase Hosting and is deployed using Firebase CLI. The module will also ensure that the application is scalable, secure, and reliable.

METHODOLOGY USED FOR DATA COLLECTION:

- Research: Conduct research on real-time chat application development using React, Firebase, HTML, CSS, and JavaScript. This can include reading books, articles, online tutorials, and other relevant resources.
- Define objectives: Define the objectives of the real-time chat application, including the features, functionality, and user experience that you want to achieve.
- Design: Create a design document that outlines the architecture, user interface, and functionality of the chat application. This should include wireframes, mockups, and flowcharts.
- Develop: Use the React library, Firebase database, and HTML, CSS, and JavaScript to develop the chat application. This may involve coding, testing, and debugging the application.
- Test: Test the chat application to ensure that it functions as intended and meets the defined objectives. This can involve unit testing, integration testing, and user acceptance testing.
- Collect data: During the testing phase, collect data on the performance and user experience of the chat application. This can include metrics such as response time, message delivery rate, and user satisfaction.
- Analyze data: Analyze the data collected to identify any issues or areas for improvement. This may involve using statistical tools, data visualization, or other analytical techniques
.
- Refine: Use the insights gained from the data analysis to refine the chat application. This may involve making changes to the design, code, or database structure.
- Deploy: Once the chat application is fully tested and refined, deploy it to a production environment for use by end-users.
- Monitor: Monitor the chat application in the production environment to ensure that it continues to function correctly and meets user needs. This may involve collecting additional data and making further refinements as needed.

Conclusion

This scope outlines the main features and functionalities of a real-time chat application using ReactJS, Firebase, HTML, CSS, and JavaScript.
The application will enable users to create accounts, join chat rooms, and communicate in real-time. The application will provide a user-friendly interface with advanced features such as message deletion, user blocking, and notification alerts.
The application will be designed with a responsive layout that adapts to different devices and will be hosted on Firebase Hosting for scalability and reliability.

REFERENCES

1. "Building a Real-Time Chat Application with React and Firebase" by Leigh Halliday
2. "React Native Chat: Build a Chat App and Learn React Native and Firebase" by Konstantin Shkut.
3. “PL/SQL”. By Ivan Bayross
4. http://en.wikipedia.org/wiki/Real_Time_Chat_Application