REAL TIME CHAT WEBSITE

Web Technologies and Applications (IT254-Minor) Report Submitted in partial fulfillment of the requirements for the degree of B.Tech (Minor) In INFORMATION TECHNOLOGY

By
Abhi Krishna A (211EE203)

G Vishal Vardhan Reddy(211EE227)

Sumukh S K (211EE153)

DEPARTMENT OF INFORMATION TECHNOLOGY NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA SURATHKAL, MANGALORE -575025

NOVEMBER, 2023

DECLARATION

I hereby *declare* that the *Web Technologies and Applications (IT254-Minor) Project* titled **Real Estate Management Website**, which is being submitted to the National Institute of Technology Karnataka Surathkal, in partial fulfillment of the requirements for the award of the Degree of B.Tech (Minor) in the department of Information Technology, is a ***bonafide report of the work carried out by me.*** The material contained in this project has not been submitted to any University or Institution for the award of any degree.

Abhi Krishna A	G Vishal Vardhan Reddy	Sumukh S K
211EE203	211EE227	211EE153

Place : NITK, Surathkal
Date : 20th November 2023

CERTIFICATE

This is to certify that the Web Technologies and Applications (IT254-Minor) project titled **“Real Estate Management Website”** has been presented by Abhi Krishna A (211EE203), Vishal (211EE227), Sumukh S K (211EE153), students of V semester B.Tech. (Minor), Department of Information Technology, National Institute of Technology Karnataka, Surathkal on 20th November, during the odd semester of the academic year 2022 - 2023, in partial fulfillment of the requirements for the award of the degree of B.Tech(Minor) in Information Technology.

Guide: Dr. Janani
Signature of the Guide with Date

Place : NITK, Surathkal

Date : 20th November 2023

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

Welcome to our cutting-edge real-time chat web application, a modern marvel designed for unparalleled instant messaging experiences. This platform revolutionizes communication by offering live, real-time conversations and dynamic group chats, fostering seamless interaction among users. The application boasts a user-friendly interface coupled with a responsive design, ensuring effortless message sharing and user engagement.

Leveraging state-of-the-art technology, this chat application guarantees a smooth and interactive user experience. With an emphasis on speed and reliability, it redefines online connectivity, setting new standards for web-based communication. The user-centric design prioritizes simplicity, making it easy for individuals and groups to connect effortlessly in real-time.

Built with innovation at its core, our chat application represents the forefront of web-based communication solutions. Its dynamic features enable users to stay connected and engage effortlessly, enhancing the overall digital communication experience. Whether for personal or professional use, this platform provides a versatile and efficient way for individuals and groups to connect and communicate, fostering a sense of immediacy and connectivity in the digital landscape.

2 OBJECTIVE

The central aim of this ambitious project is to conceive and execute a state-of-the-art Real-Time Chat Website, distinguished by its user-friendly interface and robust functionality. Our primary objective is to forge a platform that streamlines the property search process, prioritizing user convenience amidst a vast array of real estate listings. Key functionalities encompass user authentication through login and registration mechanisms, comprehensive property listings, and detailed property information. In striving for excellence, we aspire to redefine the property listing experience, enhancing it to be more informative and engaging for both buyers and sellers alike.

In achieving this vision, the project will harness cutting-edge technologies to ensure seamless real-time communication through chat features, fostering immediate interaction between users. The integration of user-friendly interfaces and responsive design will contribute to an enhanced user experience, facilitating intuitive navigation through the diverse real estate offerings. Moreover, the inclusion of advanced search capabilities and informative property details aims to empower users with the knowledge they need to make informed decisions.

This Real-Time Chat Website stands as a testament to our commitment to innovation and user-centric design, with the ultimate goal of revolutionizing the property search landscape. By combining cutting-edge technology with a focus on user experience, we envision a platform that not only meets but exceeds the expectations of buyers and sellers in the dynamic real estate market.

3 METHODOLOGY

3.1 Node Js

In a real-time chat web application, Node.js serves as the foundational backend technology, managing server-side operations and facilitating instant communication. Leveraging its event-driven architecture, Node.js efficiently handles incoming requests from users, executes server-side logic, and integrates WebSocket technology to enable seamless real-time bidirectional communication. This framework's non-blocking I/O model ensures efficient handling of concurrent connections, vital for a chat app with multiple users engaged in simultaneous conversations. By integrating with Express.js, Node.js streamlines routing and HTTP request handling, enhancing backend efficiency. Overall, Node.js powers the application's backend, ensuring low-latency communication, scalability, and efficient server-side operations crucial for real-time messaging.

3.2 React Js

In a real-time chat web application, React.js functions as the client-side framework, responsible for building an interactive and dynamic user interface. Leveraging React's component-based architecture, the application's frontend is divided into reusable and modular components, facilitating a structured UI. React efficiently handles UI rendering and updates, providing a responsive and seamless user experience for instant messaging. Its virtual DOM (Document Object Model) efficiently manages and updates the UI, ensuring optimal performance and responsiveness, even in a dynamic real-time environment like a chat app. Additionally, React's state management capabilities enable the application to handle data changes, UI re-renders, and component updates efficiently, contributing to a smooth and engaging user experience.

3.3 Mongo DB

MongoDB acts as the robust database solution for a real-time chat web app, storing and managing the vast amounts of dynamic message data. Its NoSQL document-oriented structure allows flexible schema management, facilitating quick and scalable data storage. MongoDB's ability to handle high volumes of real-time data, coupled with its seamless integration with Node.js, ensures efficient storage, retrieval, and management of chat messages, essential for instant communication.

3.4 Express

Express.js, the minimalist web application framework, is instrumental in the real-time chat web application as it facilitates streamlined backend operations and routing. Integrated with Node.js, Express simplifies handling HTTP requests, defining routes, and executing middleware for the chat app. It manages routes for user authentication, message retrieval, and other backend functionalities. With its middleware capabilities, Express handles incoming requests efficiently, ensuring smooth data flow between the frontend (built with React.js) and the backend (powered by Node.js). Additionally, Express enables the creation of RESTful APIs, allowing seamless communication between the frontend and backend, optimizing the overall functionality and performance of the chat application..

3.5 Socket io

In the real-time chat web application, Socket.IO is pivotal, establishing a persistent connection between the Node.js server and clients' web browsers. This bidirectional communication protocol, built on WebSocket technology, ensures instant message delivery without continuous HTTP requests. Operating on an event-driven model, Socket.IO enables the emission and handling of specific events, such as message sending, ensuring real-time communication. Its functionality extends to broadcasting messages to designated rooms or users, error handling, and automatic reconnection, ensuring a seamless and engaging real-time chat experience.

4 FLOWCHART / BLOCK DIAGRAM

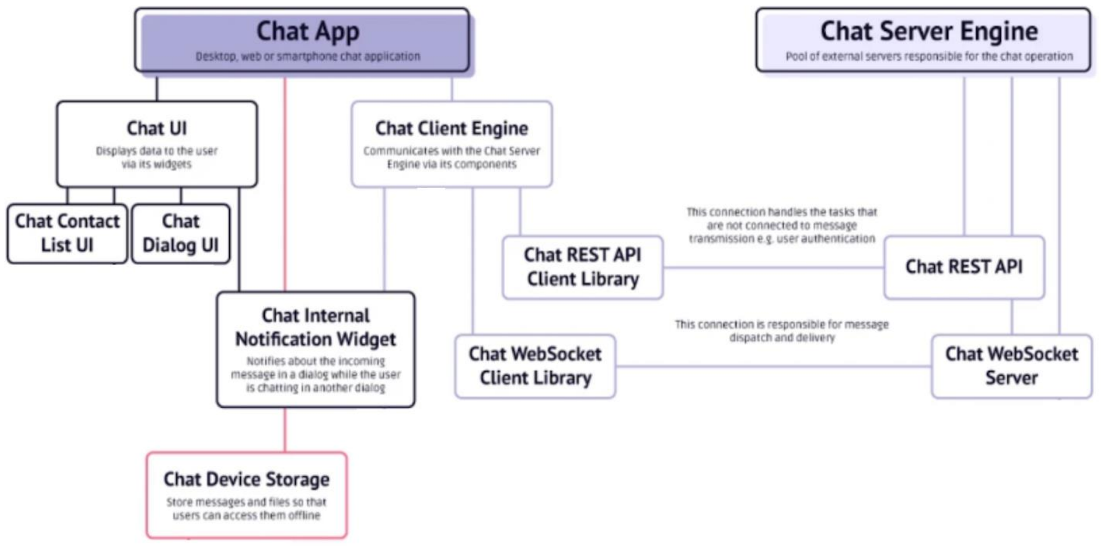

Chat App
Desktop, web or smartphone chat application
Chat Server Engine
Pool of external servers responsible for the chat operation
Chat UI
Displays data to the user via its widgets
Chat Client Engine
Communicates with the Chat Server Engine via its components
Chat Contact List UI
Chat Dialog UI
This connection handles the tasks that are not connected to message transmission e.g. user authentication
Chat REST API Client Library
Chat REST API
Chat Internal Notification Widget
Notifies about the incoming message in a dialog while the user is chatting in another dialog
This connection is responsible for message dispatch and delivery
Chat WebSocket Client Library
Chat WebSocket Server
Chat Device Storage
Store messages and files so that users can access them offline

Fig 4.1 – System design

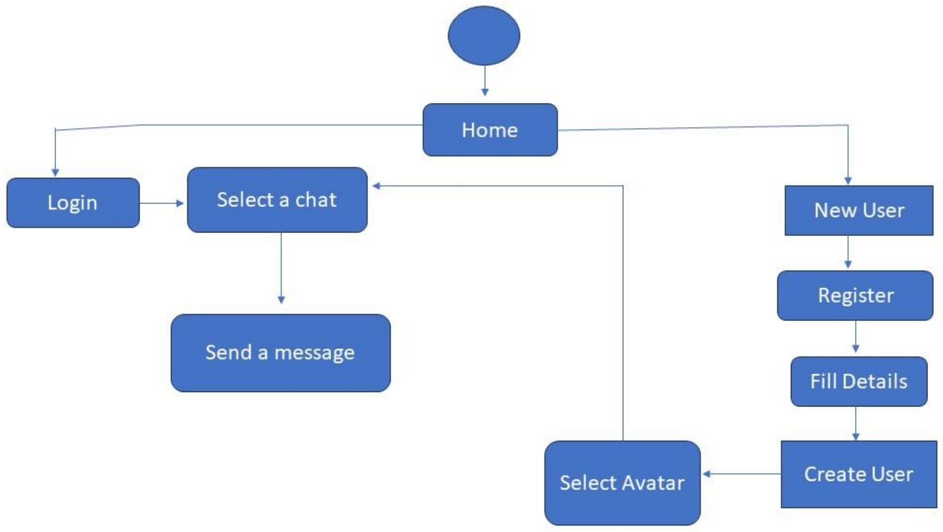


Fig 4.2 - UX Flowchart

5 IMPLEMENTATION

5.1 Register page (new user)

The implementation of a registration page involves establishing a connection between a React frontend and a MongoDB database on the backend. In React, a registration form is created to capture user details, including username, email, password, and confirm password. Using state management, the form handles user input and ensures password consistency. Upon submission, the React frontend communicates with a Node.js and Express backend. The backend, connected to MongoDB, employs Mongoose for schema definition and database operations. Before registering a new user, the backend checks for unique usernames and emails, enhancing data integrity. Successful registrations trigger responses to the React frontend, facilitating a seamless user registration experience.
Implementing avatar selection involves integrating an open-source random avatar generator API,. After obtaining a free API key, React can make HTTP requests to the API endpoint, specifying style and options. Upon retrieval, the avatar URL is displayed in the application. This approach provides users with unique and visually appealing avatars, enhancing personalization. Customization options, such as background color and style, are available through the API, allowing seamless integration into the user registration or profile creation process.

5.2 Login page

Creating a login page involves designing a React component that captures user credentials—typically, email or username and password. The component includes state management to handle user input and form submission. Utilizing an Axios library or Fetch API, the React frontend communicates with a Node.js and Express backend. The backend verifies user credentials against stored data, implementing secure authentication measures such as password hashing.Error handling and feedback mechanisms ensure a smooth user experience, providing informative messages for invalid credentials or other login issues.

5.3 Chat page

The implementation of a chat page involves creating a React component to facilitate real-time communication using Socket.IO. Users select a chat, initiating a WebSocket connection between the React frontend and a Node.js and Express backend. Socket.IO ensures bidirectional communication, enabling instant message updates. Messages exchanged in the chat are stored in MongoDB, providing a persistent record for future access. This integration allows users to review and retrieve chat history seamlessly. The backend employs MongoDB for efficient data storage and retrieval. Real-time features, including message notifications and dynamic updates, enhance the user experience. The utilization of Socket.IO ensures a responsive and interactive chat environment, fostering efficient communication in a dynamic, web-based setting.

6 RESULTS

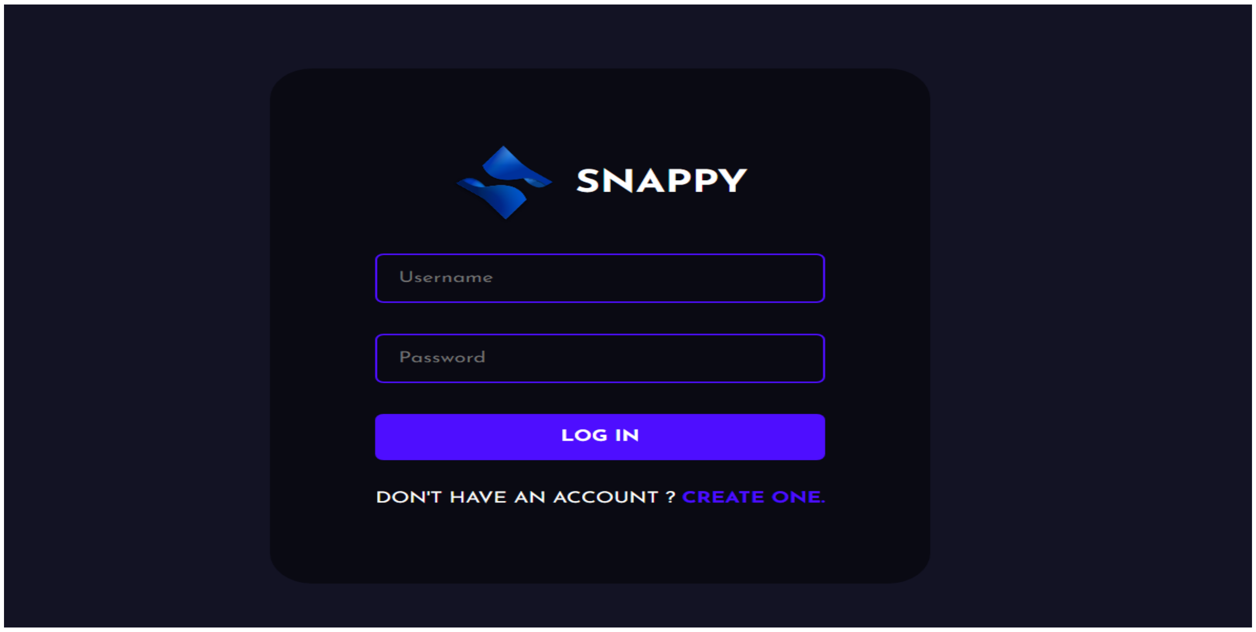


Fig 6.1 – user login page

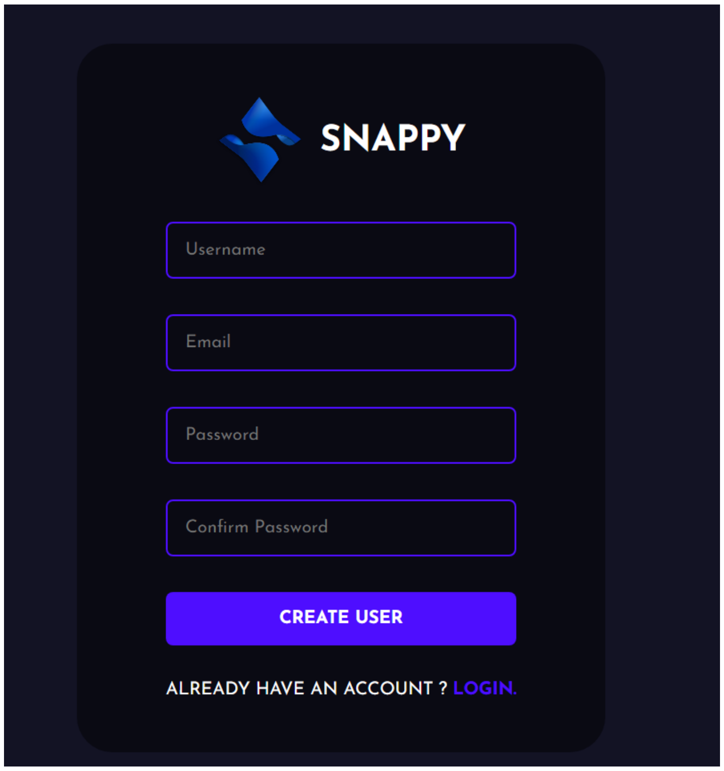
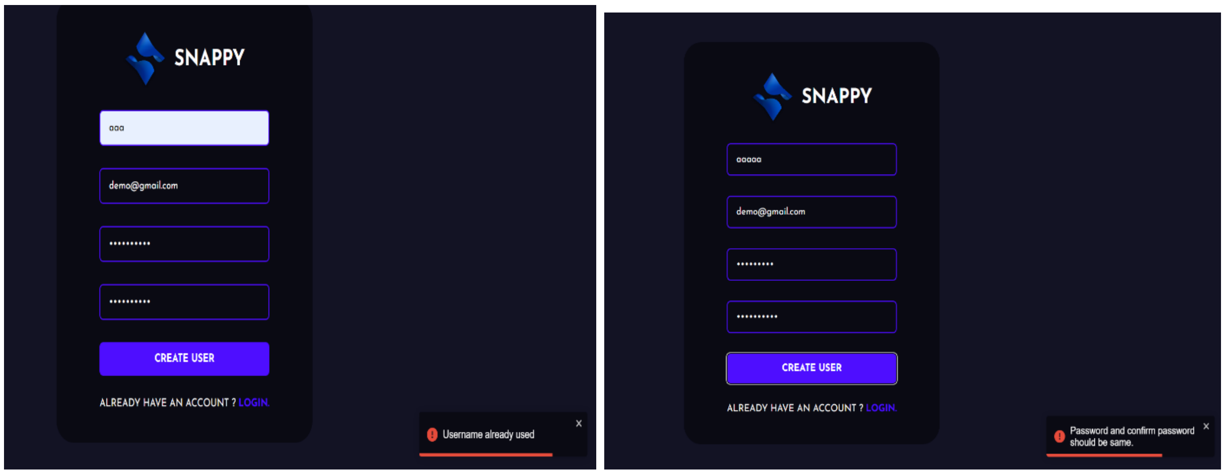


Fig 6.2 – Register page


SNAPPY
CREATE USER
ALREADY HAVE AN ACCOUNT ? LOGIN.
SNAPPY
CREATE USER
ALREADY HAVE AN ACCOUNT ? LOGIN.

aaa

demo@gmail.com

Username already used

aaaaa

demo@gmail.com

Password and confirm password should be same.

Fig 6.3 – New user errors

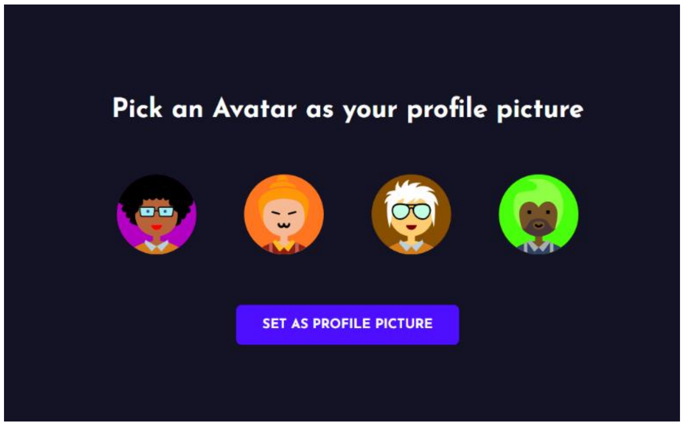


Fig 6.4 – Pick avatar

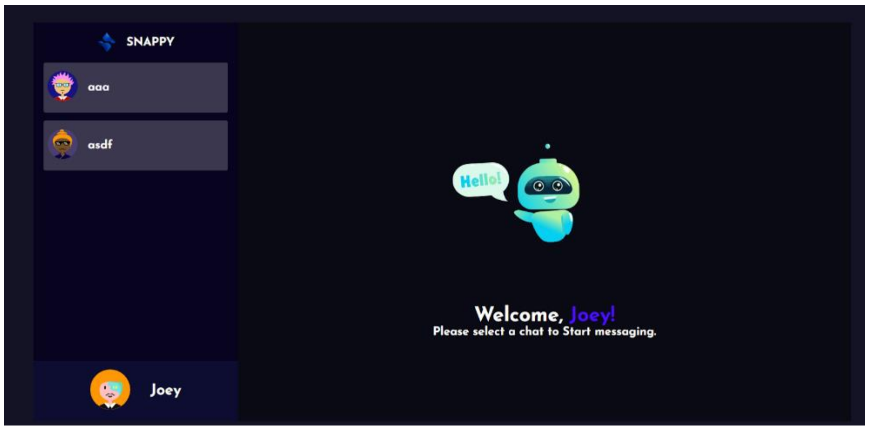


Fig 6.5 – Welcome page

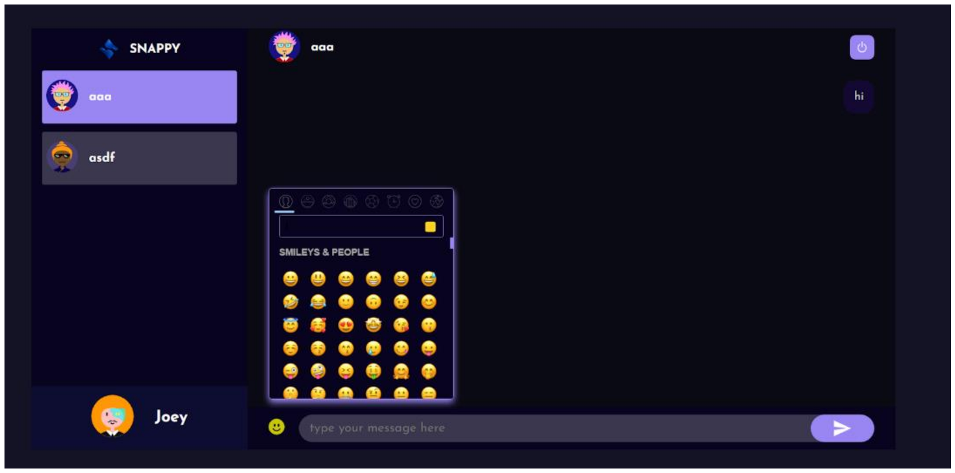
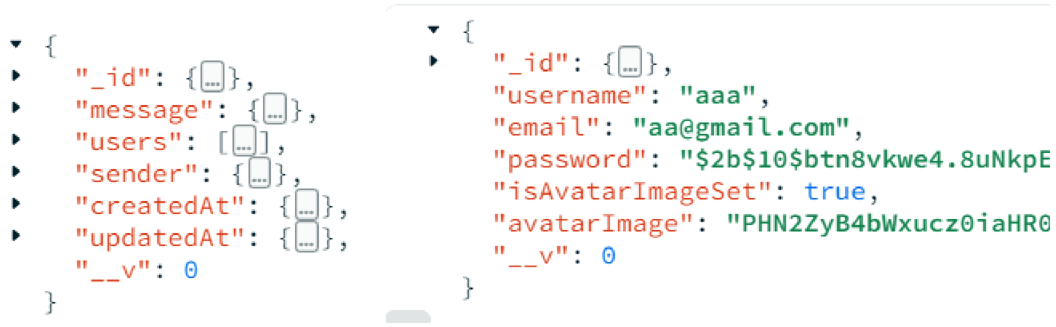


Fig 6.6 – Chat page



```
{
  "_id": {…},
  "message": {…},
  "users": […],
  "sender": {…},
  "createdAt": {…},
  "updatedAt": {…},
  "__v": 0
}
```

```
{
  "_id": {…},
  "username": "aaa",
  "email": "aa@gmail.com",
  "password": "$2b$10$btn8vkwe4.8uNkpE
  "isAvatarImageSet": true,
  "avatarImage": "PHN2ZyB4bWxucz0iaHR0
  "__v": 0
}
```

Fig 6.6 – MongoDB Json

7 FUTURE SCOPE

The future scope of our real-time chat web application extends beyond its current capabilities, anticipating advancements that will redefine digital communication. We envision incorporating artificial intelligence for personalized user experiences, intelligent chatbots for automated assistance, and enhanced security features to ensure privacy. Augmented reality integration may offer immersive communication environments, elevating user engagement. Furthermore, cross-platform compatibility and expanded integration with emerging technologies could broaden accessibility. Collaborative features such as file sharing and virtual meeting spaces are poised for expansion, facilitating seamless work and social interactions. As technology evolves, our chat application aims to stay at the forefront, embracing innovations that enhance communication efficiency, security, and user satisfaction, making it a pivotal tool in the evolving landscape of real-time digital connectivity.

8 REFERENCES

1. https://legacy.reactjs.org/docs/getting-started.html
2. https://www.mongodb.com/docs/
3. https://expressjs.com/
4. https://socket.io/docs/v4/
5. https://axios-http.com/docs/intro