

**PROJECT REPORT**

## On

BuzzChat

Submitted in partial fulfilment of the requirement for the Course BEE (22CS026) of

**COMPUTER SCIENCE AND ENGINEERING**

**B.E. Batch-2022 in**

**Dec -2024**

|  |  |
| --- | --- |
| **Under the Guidance of:** - | **Submitted By:** - |
| Rahul Singh Rajput | Piyush Arora |
| Project Supervisor | Roll No.: 2210992040 |
|  | Paramveer |
|  | Roll No.: 2210992019 |
|  | Partha Pratim Tamuly |
|  | Roll No.: 2210992031 |
|  | Nikhil Verma |
|  | Roll No.: 2210991983 |

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CHITKARA UNIVERSITY**

**PUNJAB**

# CERTIFICATE

This is to be certified that the project entitled “BuzzChat” has been submitted for the Bachelor of Computer Science Engineering at Chitkara University, Punjab during the academic semester July 2024 – December 2024 is a bona fide piece of project work carried out by “Piyush(2210992040), Paramveer(2210992031), Partha(2210992031) and Nikhil(2210991983)” towards the partial fulfilment for the award of the course Integrated Project (CS 203) under the guidance of “Rahul Singh Rajput” and supervision.

**Signature:**

Rahul Singh Rajput Project Supervisor (BE-CSE)



# CANDIDATE’S DECLARATION

We, Piyush(2210992040), Paramveer(2210992019), Partha(2210992031) and Nikhil(2210991983), B.E.-2022 of the Chitkara University, Punjab hereby declare that the Integrated Project Report entitled “BuzzChat” is an original work and data provided in the study is authentic to the best of our knowledge. This report has not been submitted to any other Institute for the award of any other course.

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature:** | **Signature:** | **Signature:** | **Signature:** |
| Piyush Arora | Paramveer | Partha Pratim Tamuly | Nikhil Verma |
| 2210992040 | 2210992019 | 2210992031 | 221099183 |

Place:

Date:

# ACKNOWLEDGEMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behavior and acts during the course of study.

We express our sincere gratitude to all for providing me an opportunity to undergo Integrated Project as the part of the curriculum.

We are thankful to “Rahul Sir” for his support, cooperation, and motivation provided to us during the training for constant inspiration, presence and blessings.

Lastly, we would like to thank the almighty and our parents for their moral support and friends with whom we shared our day-to day experience and received lots of suggestions that improve our quality of work.

|  |  |  |  |
| --- | --- | --- | --- |
| **Piyush Arora** | **Paramveer** | **Partha Pratim Tamuly** | **Nikhil Verma** |
| 2210992040 | 2210992019 | 2210992031 | 2210991983 |



# Table of Contents

|  |  |  |
| --- | --- | --- |
| **S. L. No.** | **Topics** | **Page No.** |
| 1 | Abstract | 6 |
| 2 | Introduction | 7 |
| 2.1 | Background | 7 |
| 2.2 | Problem Statement | 7 |
| 3 | Software and Hardware Requirement Specification | 8 |
| 3.1 | Methods | 8 |
| 3.2 | Programming/Working Environment | 8 |
| 3.3 | Requirements to Run the Application | 8 |
| 4 | Database Analysing, Design and Implementation | 9 |
| 5 | Program’s Structure Analysing and GUI Constructing | 9-11 |
| 6 | Code-Implementation and Database Connections | 12-15 |
| 7 | System Testing | 16 |
| 8 | Limitations | 16-17 |
| 9 | Conclusion | 17 |
| 10 | Future Scope | 17 |
| 11 | Bibliography/References | 17 |



1. **Abstract**

In the digital age, real-time communication has become integral to both personal and professional interactions. Despite numerous existing platforms, challenges such as scalability, security, and user experience persist, affecting the efficiency and satisfaction of users. This report presents BuzzChat, a MERN stack-based chat application designed to address these critical challenges by offering a secure, scalable, and user-friendly platform for instant messaging and collaboration.

BuzzChat empowers users by providing seamless real-time communication, enabling them to send and receive messages instantly, share multimedia content, and create group conversations with ease. For businesses and organizations, the application facilitates efficient team collaboration, enhancing productivity and fostering a connected work environment.

By offering a comprehensive solution suitable for both individual users and enterprise environments, BuzzChat aims to improve overall communication practices. Drawing inspiration from user-centric applications like Slack and WhatsApp, the interface is designed to be intuitive, ensuring that users can navigate the platform effortlessly.

The implementation of BuzzChat not only streamlines communication processes but also contributes to improved user engagement by fostering better interaction and collaboration among users. This report details the background, objectives, software and hardware requirements, design and implementation strategies, testing methodologies, limitations, and future enhancements of the BuzzChat project, emphasizing its potential to revolutionize real-time communication in the digital landscape.



# Introduction

## Background

Effective communication is the backbone of both personal relationships and organizational success. Traditional communication methods, while still relevant, often fall short in providing the immediacy and interactivity required in today's fast-paced environment. Existing chat applications, though numerous, frequently encounter issues such as limited scalability, security vulnerabilities, and suboptimal user experiences, which can hinder effective communication and collaboration.

Digital technology offers a promising solution to these challenges by enabling the development of integrated platforms for real-time communication. However, there remains a lack of a unified, secure, and highly scalable solution that meets the diverse needs of both individual users and enterprises. **BuzzChat** aims to bridge this gap by providing a web and mobile application for seamless instant messaging and collaboration. By leveraging the MERN (MongoDB, Express.js, React.js, Node.js) stack, **BuzzChat** enhances the scalability, security, and responsiveness of chat applications, empowering users and supporting organizations to improve their communication strategies and outcomes.

## Problem Statement

Despite the proliferation of chat applications, users often encounter challenges related to scalability, security, and user experience. Individual users may face issues such as message delays, limited customization, and inadequate privacy controls, which can diminish the overall communication experience. Concurrently, businesses and organizations struggle with integrating chat solutions that are secure, scalable, and capable of supporting large teams with diverse communication needs.

The absence of a comprehensive, secure, and easily scalable chat platform creates barriers to effective communication and collaboration. **BuzzChat** aims to address these issues by providing a seamless solution that enables both individual users and organizations to manage their real-time communications efficiently. By bridging this gap, **BuzzChat** enhances interaction and collaboration within various environments, ultimately improving user satisfaction and organizational productivity.

.



# Software and Hardware Requirement Specification

## Methods

**BuzzChat** employs a cloud-based architecture utilizing the MERN stack to enable scalable and secure real-time communication. The application is designed to be accessible through web portals and mobile applications, ensuring user-friendly interaction across multiple devices.

## Programming/Working Environment

The development environment for **BuzzChat** includes:

* **Frontend**: HTML, CSS, JavaScript, and React.js
* **Backend**: Node.js with Express framework
* **Database**: MongoDB for data storage
* **Real-Time Communication**: Socket.io for real-time messaging
* **Cloud Services**: AWS or Azure for hosting and storage
* **IDE**: Visual Studio Code
* **Version Control**: Git for source code management

## Requirements to Run the Application

**Hardware Requirements**:

* A computer or server with at least 8GB RAM and 4 CPU cores
* Stable internet connection

**Software Requirements**:

* Node.js and npm installed
* MongoDB database server
* Web browser (latest version)
* Mobile platforms (iOS and Android) for mobile app deployment



# Database Analysing, Design and Implementation

**BuzzChat** utilizes MongoDB for its database, which allows flexible storage of user data and messages. The database schema includes collections for users, conversations, messages, and notifications. Each document contains fields such as:

* **Users Collection**:
  + User ID
  + Username
  + Password (hashed)
  + Profile information (e.g., avatar, status)
  + Contact list
* **Conversations Collection**:
  + Conversation ID
  + Participants (User IDs)
  + Conversation type (e.g., private, group)
  + Creation date
* **Messages Collection**:
  + Message ID
  + Sender ID
  + Conversation ID
  + Content (text, images, files)
  + Timestamp
  + Read receipts
* **Notifications Collection**:
  + Notification ID
  + User ID
  + Type (e.g., new message, mention)
  + Content
  + Timestamp
  + Read status

# Program’s Structure Analysing and GUI Constructing

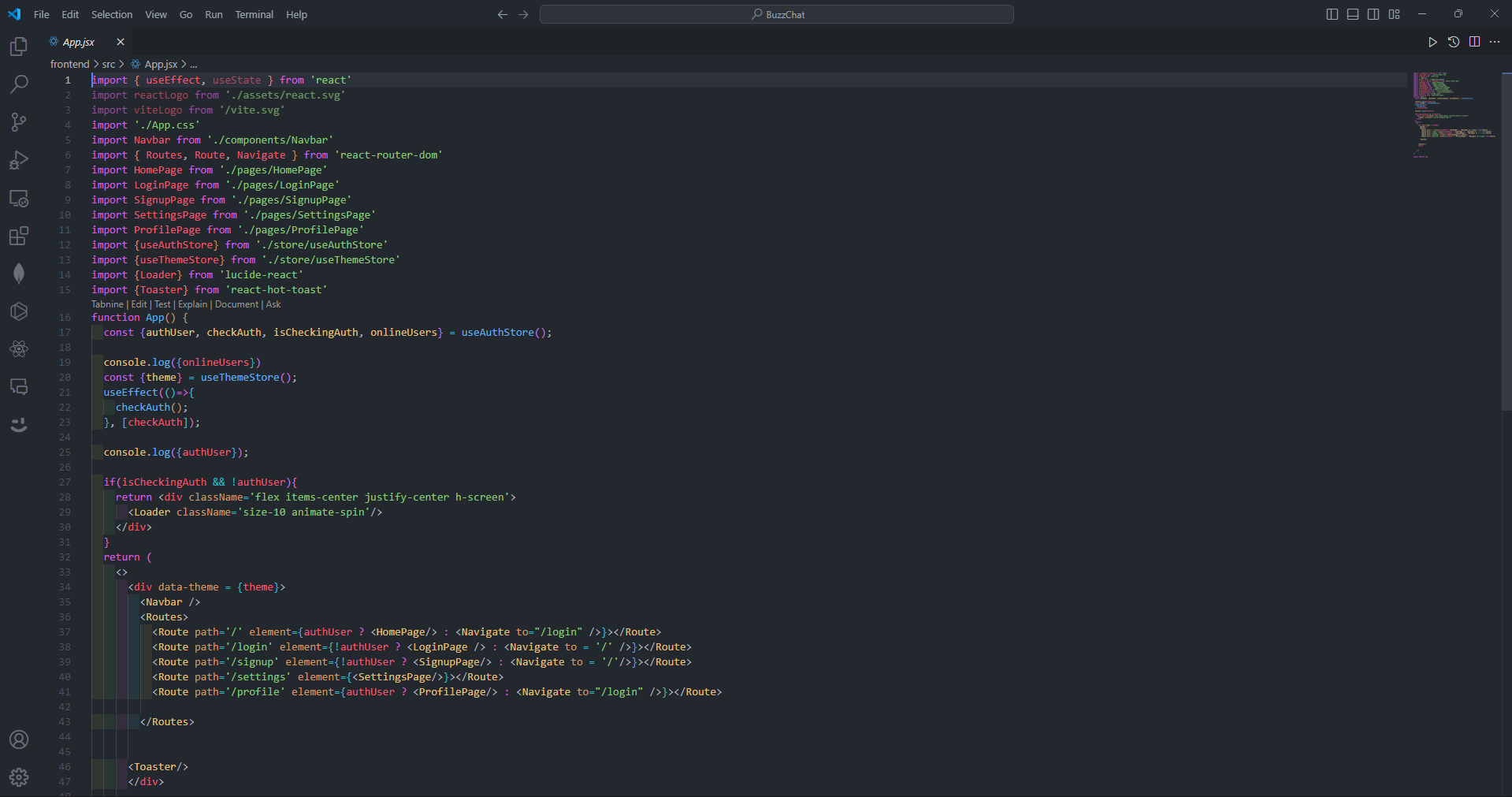
The application is structured into several modules:

* **User Authentication**: Handles login, registration, and authentication for users.
* **Real-Time Messaging**: Facilitates sending and receiving messages in real-time using Socket.io.
* **Conversation Management**: Allows users to create, join, and manage conversations and groups.
* **User Profile**: Provides an overview of user information and allows customization of profile settings.
* **Notifications**: Manages real-time notifications for incoming messages and other alert.

# Code-Implementation and Database Connections

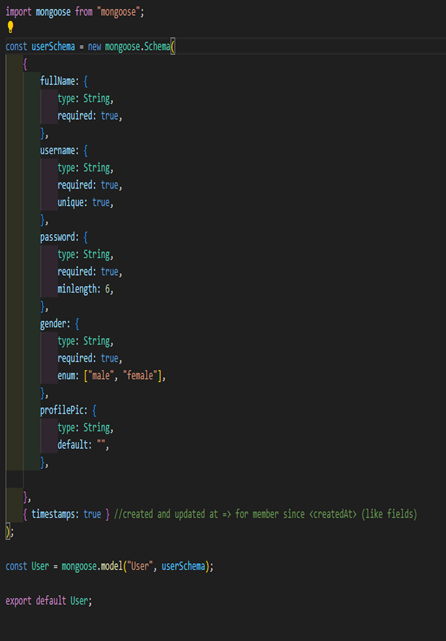
The backend uses Node.js and Express.js to handle API requests and manage real-time communication with Socket.io. MongoDB is accessed using the Mongoose library, which simplifies interactions with the database.

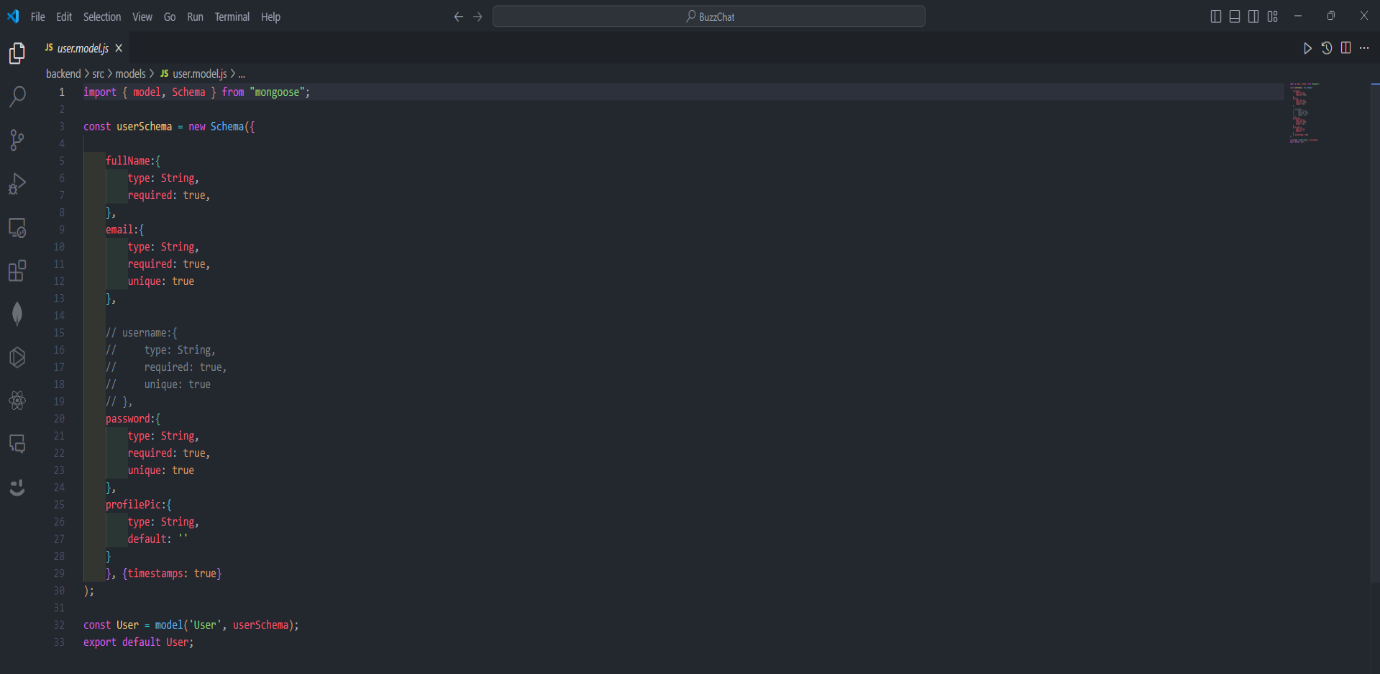
Snapshots of code:

* App.jsx
* 

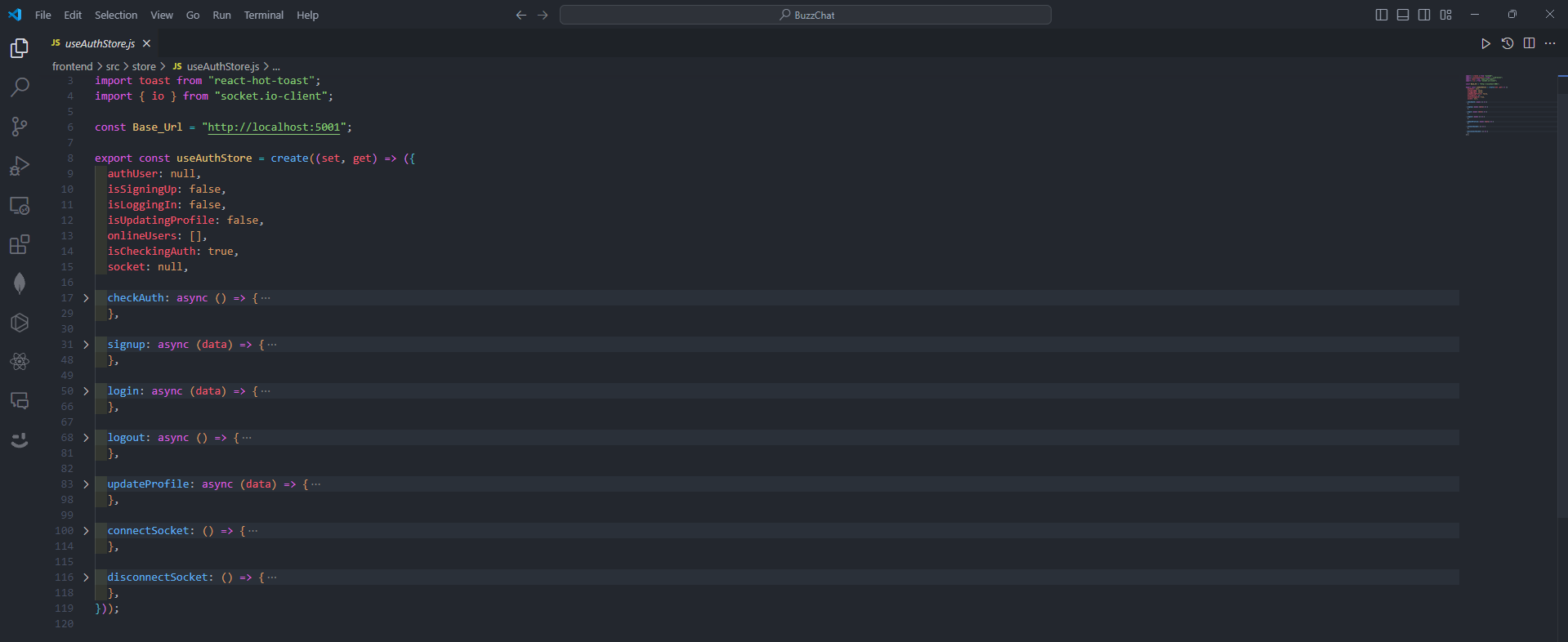


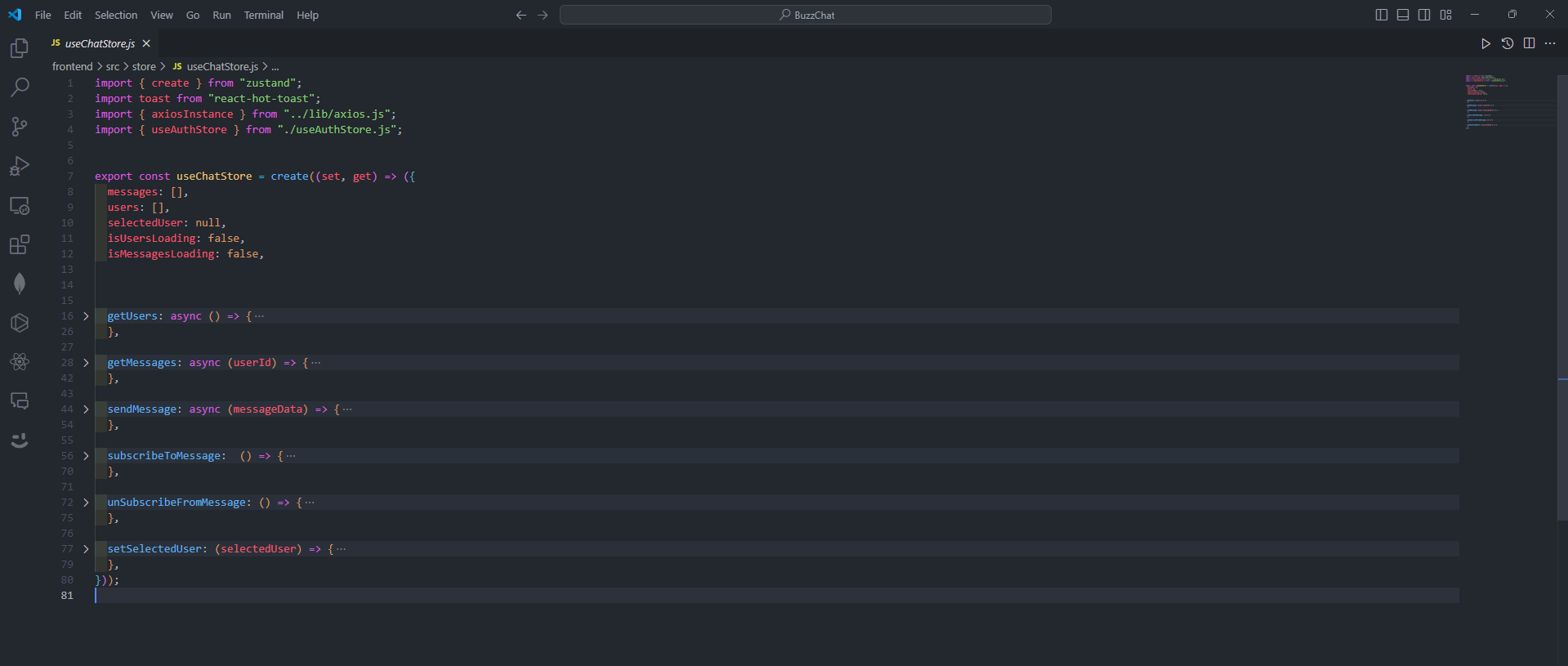
* Database Connection



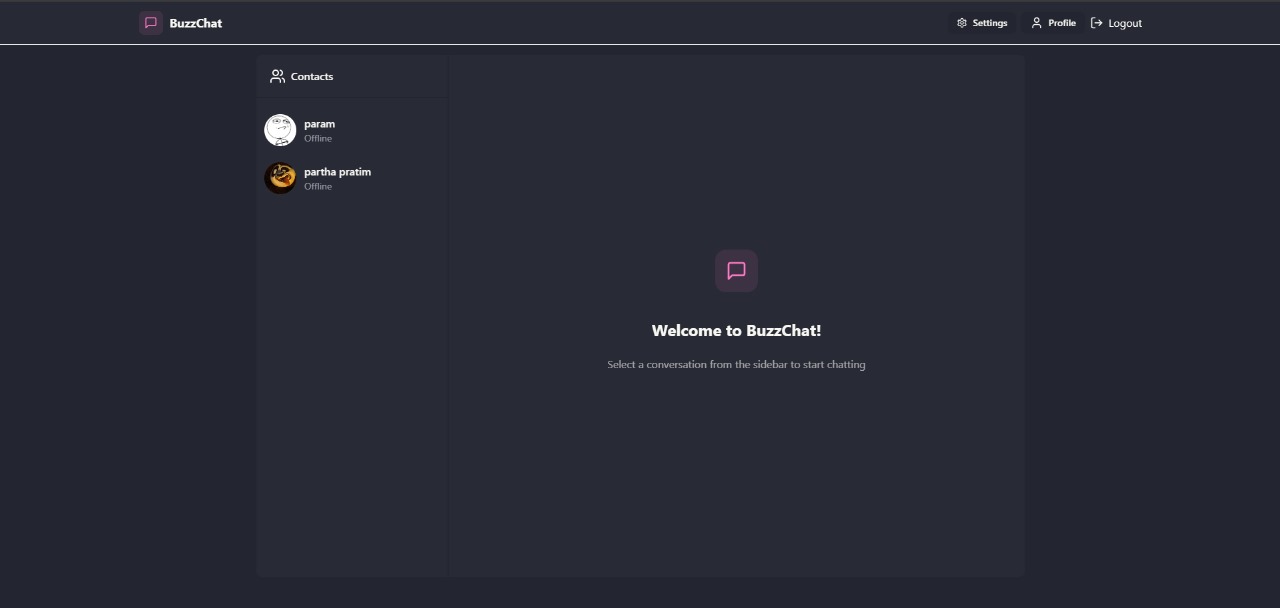
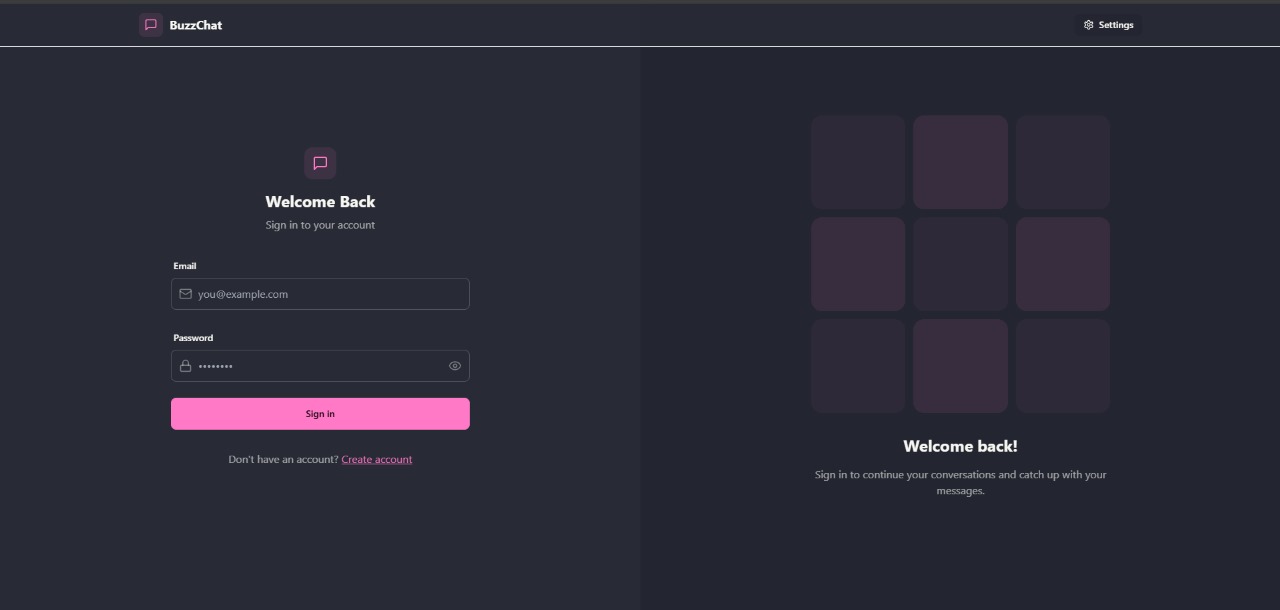
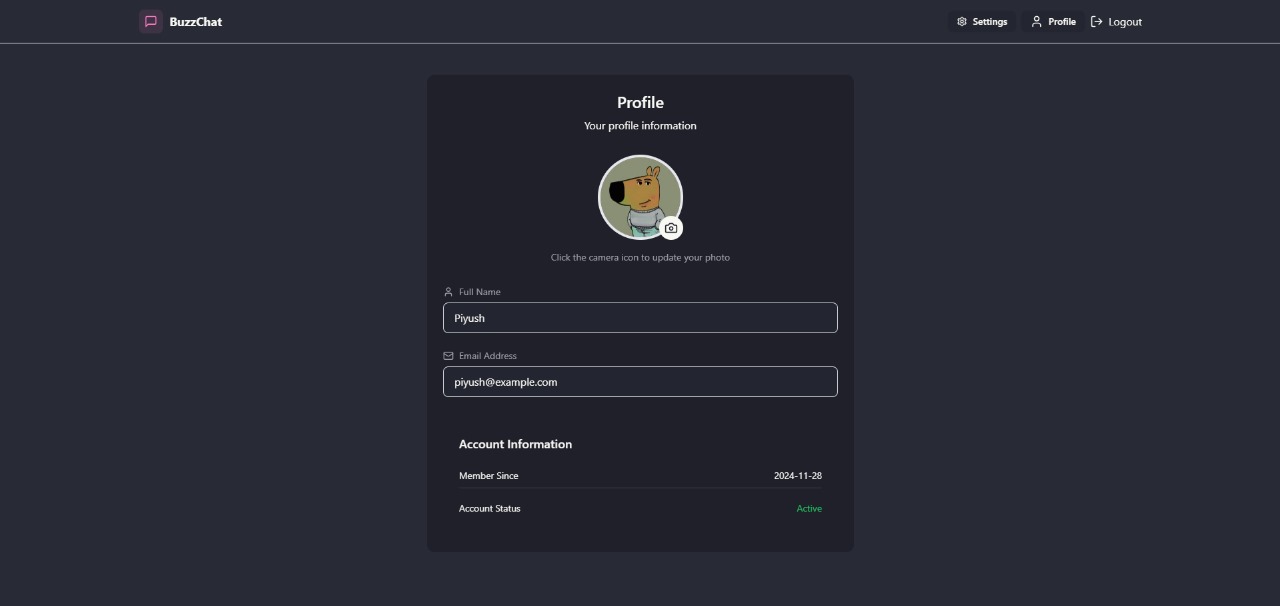
* Mongoose Setup



* Authentication
* UseChatStore



o UI





# System Testing

Testing of **BuzzChat** included three main types: unit testing, integration testing, and user acceptance testing (UAT).

* **Unit Testing** focused on verifying individual components of the application, such as user authentication, message handling, and database operations, to ensure each part functions correctly.
* **Integration Testing** assessed how well different components work together within the overall system, ensuring seamless interaction between the frontend, backend, and real-time communication modules.
* **User Acceptance Testing (UAT)** involved gathering feedback from real users to ensure the application meets their needs and expectations. This phase included testing the user interface, ease of use, responsiveness, and overall user experience.

This comprehensive testing process helped identify and resolve issues, ensuring a smooth and effective user experience. Tools such as Jest for unit testing, Postman for API testing, and manual testing for UAT were utilized to ensure the application's reliability and performance.

# Limitations

While **BuzzChat** presents a robust solution for real-time communication, certain limitations include:

* **Scalability Constraints**: Although built on the MERN stack, handling an extremely high number of concurrent users may require additional optimization and infrastructure scaling.
* **Data Security**: Despite implementing standard security measures, the platform remains susceptible to potential cyber threats such as hacking attempts and data breaches.
* **Latency Issues**: Real-time messaging relies heavily on network stability. Users with unstable internet connections may experience delays or message delivery failures.
* **Feature Limitations**: Current functionalities may not include advanced features like video conferencing, file encryption, or extensive customization options, which some users might require.
* **User Adoption**: Encouraging users and organizations to switch from existing platforms to **BuzzChat** may require significant marketing efforts and demonstrations of superior value.
* **Technical Dependencies**: Dependence on third-party services (e.g., cloud providers, libraries) can introduce vulnerabilities if these services experience downtime or other issues.
* **Cross-Platform Compatibility**: Ensuring seamless functionality across all devices and browsers can be challenging, potentially leading to inconsistent user experiences.
* **Data Privacy Concerns**: Users may have concerns about how their data is stored and managed, necessitating transparent privacy policies and robust data protection measures.

# Conclusion

**BuzzChat** provides a robust solution to the challenges of real-time communication in both personal and professional settings. By offering a secure, scalable, and user-friendly platform, it enhances user engagement and empowers individuals and organizations to communicate efficiently and effectively.

This project showcases the transformative potential of the MERN stack in developing zmodern communication applications. By streamlining message delivery, enabling real-time interactions, and facilitating seamless collaboration, **BuzzChat** improves the efficiency of communication processes, contributing to better user satisfaction and organizational productivity.

# Future Scope

Future enhancements for **BuzzChat** could include:

* **Integration of AI for Smart Messaging**: Implementing AI-driven features such as predictive text, chatbots, and sentiment analysis to enhance user interaction.
* **Development of Native Mobile Applications**: Creating fully native mobile apps for iOS and Android to improve performance and user experience on mobile devices.
* **Implementation of End-to-End Encryption**: Enhancing data security by incorporating end-to-end encryption to ensure that only intended recipients can read the messages.
* **Video and Voice Calling Features**: Adding capabilities for video and voice calls to provide a more comprehensive communication solution.
* **File Sharing and Collaboration Tools**: Enabling users to share files, collaborate on documents, and integrate with other productivity tools like calendars and task managers.
* **Customization and Theming Options**: Allowing users to customize the interface with different themes, layouts, and notification settings to better suit their preferences.
* **Advanced Analytics and Reporting**: Providing users and organizations with insights into communication patterns, message statistics, and user engagement metrics.
* **Multi-Language Support**: Expanding the application's reach by supporting multiple languages, catering to a global user base.
* **Integration with Third-Party Services**: Allowing seamless integration with other platforms and services such as CRM systems, email clients, and social media platforms.

# References

 **W3Schools**: <https://www.w3schools.com>

 **GeeksforGeeks**: <https://www.geeksforgeeks.org>

 **GitHub**: <https://github.com>

 **MERN Stack Documentation**: <https://mern.io>

 **Socket.io Documentation**: https://socket.io/docs/v4

 **React.js Documentation**: https://reactjs.org/docs/getting-started.html

 **Node.js Documentation**: https://nodejs.org/en/docs/

 **Express.js Documentation**: https://expressjs.com/en/4x/api.html

 **MongoDB Documentation**