MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

Project Report

on

Chat App(Baat Cheet)

Submitted by:-

Ayushman mahalgamaiya

0901AI211017

Khilan Lodhi

0901AI211042

V semester

Artificial Intelligence and Robotics

Faculty Mentor:-

Dr. Bhagat Singh Raghuwanshi

Assistant Professor

CENTER OF ARTIFICIAL INTELLIGENCE

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR – 474005 (MP) est. 1957

JULY-DEC 2023

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

CERTIFICATE

This is certified that **Ayushman Mahalgamaiya (0901AI211017) and Khilan Lodhi (0901AI211042)** has submitted the Minor project I report titled **CHAT APP(BAAT CHEET)** under his mentorship of **Dr. Bhagat Singh Raghuwanshi** In partial fulfilment of the requirement for the award of degree of Bachelor of Technology in **Artificial Intelligence and Robotics** from Madhav Institute of Technology and Science, Gwalior.

23/11/2023

Dr. Bhagat Singh Raghuwanshi
Faculty Mentor
Assistant Professor
Center of Artificial Intelligence

23/11/23

Dr. R. R. Singh
Coordinator
Center of Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

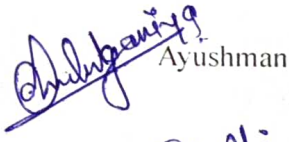
(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in **Artificial Intelligence and Robotics** at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Dr. Bhagat Singh Raghuwanshi,Assistant Professor,** Center of Artificial Intelligence.

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

Date: 23/11/23
Place: Gwalior



Ayushman Mahalgamaiya
0901AI211017

Khilan Lodhi
0901AI211042

III Year,
Artificial Intelligence And Robotics

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

(A Govt. Aided UGC Autonomous Institute Affiliated to RGPV, Bhopal)

NAAC Accredited with A++ Grade

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Centre for Artificial Intelligence,** for allowing me to explore this project. I humbly thank **Dr. R. R. Singh**, Coordinator, Centre for Artificial Intelligence, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors, I am grateful to the guidance of **Dr. Bhagat Singh Raghuwansh**, **Assistant Professor** ,Center of Artificial intelligence, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Ayushman Mahalgamaiya
0901AI211017
Khilan lodhi
0901AI211042
IIIrd year
Artificial Intelligence and Robotics

TABLE OF CONTENTS

LIST OF FIGURES

सार:

चैटिंग भौगोलिक बाधाओं के बावजूद लोगों और विचारों को एक साथ लाने के लिए प्रौद्योगिकी का उपयोग करने की एक विधि है। यह तकनीक वर्षों से उपलब्ध है, लेकिन हाल ही में इसे स्वीकार करना बंद कर दिया गया था। हमारी परियोजना एक चैट सर्वर का एक उदाहरण है। यह 2 अनुप्रयोगों से बना है क्लाइंट एप्लिकेशन, जो उपयोगकर्ताओं एंड्रॉइड डिवाइस और सर्वर एप्लिकेशन पर चलता है, जो नेटवर्क पर किसी भी एंड्रॉइड डिवाइस पर चलता है। चैटिंग शुरू करने के लिए क्लाइंट को सर्वर से कनेक्ट होना चाहिए जहां वे निजी और ग्रो कर सकते हैं।

Abstract

Chatting is a method of using technology to bring people and ideas together despite of the geographical barri- ers. The technology has been available for years but the acceptance it was quit recent. Our project is an example of a chat server. It is made up of 2 applications the client application, which runs on the users Android Device and server application, which runs on any Android Device on the network. To start chatting client should get connected to server where they can do private and group chat security measures were taken during the last one.

Chapter 1: Project Overview

1.1 Introduction

In the ever-evolving realm of digital communication, the project endeavors to develop a cutting-edge chat application using a technology stack that includes HTML, CSS, JavaScript, and ReactJS. This introduction provides an overview of the project, highlighting its significance in addressing contemporary communication needs.

1.2 Objectives and Scope

1.2.1 Objectives

- **Creation of an Intuitive Interface:-** Design a user-friendly interface for seamless interaction.
- **Real-Time Messaging:-** Implement real-time messaging functionality to ensure instant communication.
- **Dynamic User Experience:-** Utilize ReactJS to enhance the application's responsiveness and interactivity.
- **Scalability:-** Develop the application with scalability in mind to accommodate potential future enhancements.

1.2.2 Scope

The project aims to deliver a feature-rich chat application that meets the following criteria:

- Cross-Platform Compatibility:- Ensure usability across various devices and browsers.
- Basic User Authentication:- Incorporate a straightforward user authentication system.

1.3 Project Features

The chat application will boast a set of features designed to enhance user experience and meet modern communication expectations:

- Real-Time Messaging:- Users can exchange messages in real-time.
- User Authentication:- A simple authentication system to ensure secure access.
- Responsive Design:- Ensure a seamless experience across devices.

1.4 Feasibility:-

The feasibility study examines the practicality and viability of the project. With the chosen technology stack of HTML, CSS, JavaScript, and ReactJS, the project is deemed feasible due to the widespread use and compatibility of these technologies.

1.5 System Requirements:-

To run the chat application successfully, users need the following system requirements:

- Web Browser:- Compatibility with modern web browsers such as Chrome, Firefox, and Safari.
- Internet Connection:- A stable internet connection for real-time communication.
- Device Compatibility:- Support for various devices, including desktops, laptops, tablets, and smartphones.

This chapter lays the foundation for the development of the chat application, outlining its objectives, scope, features, feasibility, and system requirements. The subsequent chapters will delve into the technical details and implementation processes of the project.

Chapter 2:Literature Review

2.1 React Js:-

ReactJS, developed by Facebook, is a powerful and widely-used JavaScript library for building user interfaces. It's specifically designed to create interactive, responsive, and efficient front-end applications. React follows a component-based architecture, where UIs are broken down into reusable and manageable components. One of its key features is the virtual DOM (Document Object Model), which optimizes performance by updating only the parts of the actual DOM that have changed.

2.1.1 Key Features of React JS:-

1.Component-Based Architecture:- React allows developers to create modular components that can be reused throughout the application, promoting a more maintainable and scalable codebase.

2.Virtual DOM:- The virtual DOM enables efficient updates by only making necessary changes to the actual DOM, reducing the performance overhead associated with frequent updates.

3. Declarative Syntax:- React uses a declarative approach to describe how the UI should look, making it easier to understand and debug compared to imperative approaches.

4.React Router:- The React Router library facilitates navigation and URL management in single-page applications, enhancing the overall user experience.

2.1.2 React JS in Chat App Development:-

1.Dynamic UI Updates:- React's efficient rendering mechanism, especially with the virtual DOM, ensures that your chat app's user interface remains responsive and dynamic, crucial for real-time messaging.

2.State Management:- React js state management allows you to handle the dynamic nature of a chat application, ensuring that messages, user interactions, and other real-time updates are managed effectively.

3.Responsive Design:- React's architecture lends itself well to creating a responsive and adaptable user interface, crucial for providing a consistent experience across various devices.

4.Integration with Backend Services:- React can easily integrate with backend services, making it seamless to connect your chat app with the server-side logic responsible for handling messages, user authentication, and other functionalities.

2.2 Node Js

Node.js is a powerful, open-source, server-side JavaScript runtime built on the V8 JavaScript engine. It allows developers to execute JavaScript code outside a web browser, enabling server-side scripting. Node.js is known for its event-driven architecture and non-blocking I/O, making it highly efficient and scalable.

Node.js plays a crucial role in the backend development of chat applications. Here's how it is typically utilized:

Key Features of Node.js:-

1. NPM (Node Package Manager):- Node.js comes with NPM, a vast ecosystem of packages and libraries, simplifying the process of integrating third-party modules into applications.

2. **Fast Execution:** Built on the V8 engine, Node.js executes JavaScript code quickly, making it suitable for handling high-throughput tasks and real-time applications.

3.Single Programming Language:-Node.js enables the use of JavaScript for both server-side and client-side scripting, promoting code reusability and consistency.

4.Scalability:- Node.js is designed to scale horizontally, making it suitable for applications that experience varying levels of demand, such as chat applications with fluctuating user activity.

Node.js in Chat App Development:-

1.Real-Time Communication:-Node.js, with its non-blocking I/O and event-driven architecture, is well-suited for handling real-time communication in chat applications. It allows for simultaneous handling of multiple connections and events, making it efficient for chat functionalities.

2.WebSocket Support:- Node.js is often used with WebSocket technology to enable bidirectional communication between the server and clients. This is essential for establishing and maintaining real-time connections in a chat app.

3.Handling Concurrent Requests:-In a chat application where numerous users may be sending messages concurrently, Node.js excels at managing these concurrent requests without sacrificing performance.

5.Scalable Architecture: The scalability of Node.js is advantageous for chat applications that may experience spikes in user activity. It allows for the addition of resources to handle increased demand without significant changes to the architecture.

2.3 Socket.io:-

Socket.io is a JavaScript library that enables real-time, bidirectional communication between clients (such as web browsers) and servers. It facilitates the creation of interactive and dynamic applications, including chat applications, online gaming, and collaborative tools. Socket.io uses WebSocket technology when available, falling back to other communication methods like long polling for compatibility in various environments. It is known for its ease of use and broad compatibility with different platforms.

2.4 Express.js:-

Express.js, commonly referred to as Express, is a minimal and flexible web application framework for Node.js. It provides a set of features for building robust and scalable web applications and APIs. Express simplifies the process of defining routes, handling HTTP requests and responses, and managing middleware to add functionality to the server. It follows the model-view-controller (MVC) architectural pattern, allowing developers to structure their applications in a modular and organized way. Express.js is widely used for building both small-scale and large-scale web applications due to its simplicity and extensibility.

Chapter 3: Preliminary Design

3.1 Introduction

In this chapter, we delve into the preliminary design phase of the chat application. This involves mapping out the overall architecture, user interface, and initial interactions. The focus is on creating a solid foundation that aligns with the project's objectives and sets the stage for the development process.

3.2 Architecture Overview

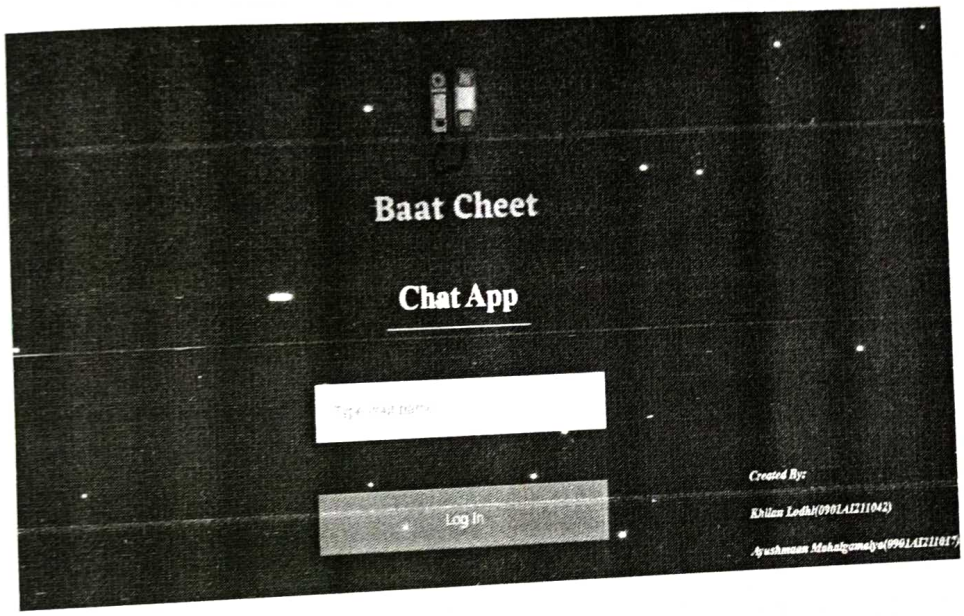
3.2.1 Frontend Architecture

The frontend of the chat application will be built using HTML, CSS, and ReactJS. React's component-based structure will be leveraged to create reusable elements, such as message components, user interface elements, and input forms. Real-time updates and dynamic content will be facilitated through React's virtual DOM.

3.3 User Interface Design

The user interface (UI) will prioritize simplicity and intuitiveness. Key components include:

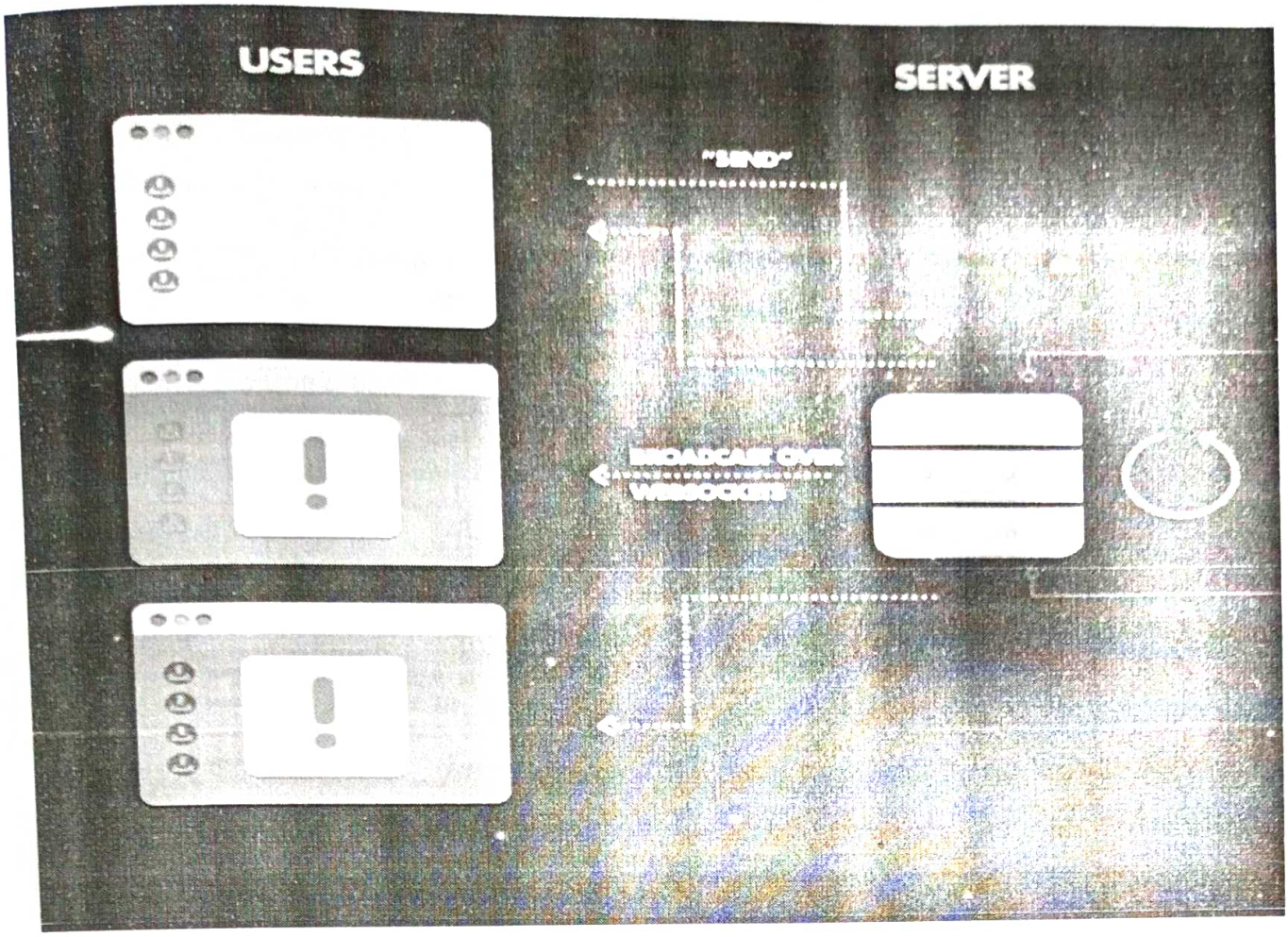
- **Chat Window:-**A central area displaying real-time messages.
- **Message Input:-** An input field for users to type and send messages.
- **User List:-**A sidebar displaying the list of online users.
- **Authentication Form:-**For users to log in or sign up



3.2.1.a Joining page

3.2.2 Backend Architecture

Node.js will serve as the backend technology, utilizing the Express.js framework. The backend will handle user authentication, manage WebSocket connections for real-time messaging, and interact with the database for storing and retrieving chat data. Socket.io will be implemented for WebSocket communication, ensuring efficient and bidirectional data flow between clients and the server.



3.2.2.a Node js Architecture

3.4 Initial User Interactions

3.4.1 User Authentication

- Users will log in or sign up using a straightforward authentication form.
- Authentication will be handled securely using hashed passwords and tokens.

3.4.2 Real-Time Messaging

- Upon authentication, users will join a chat room.
- WebSocket connections will be established for real-time communication.
- Messages sent by users will be instantly displayed in the chat window for all participants.

Chapter 4: Final Analysis and Design

4.1 Results

4.1.1 Successful Implementation

The chat application has been successfully implemented, providing users with a real-time communication platform. Key features, including user authentication, real-time messaging, multimedia sharing, and a responsive user interface, have been integrated seamlessly.

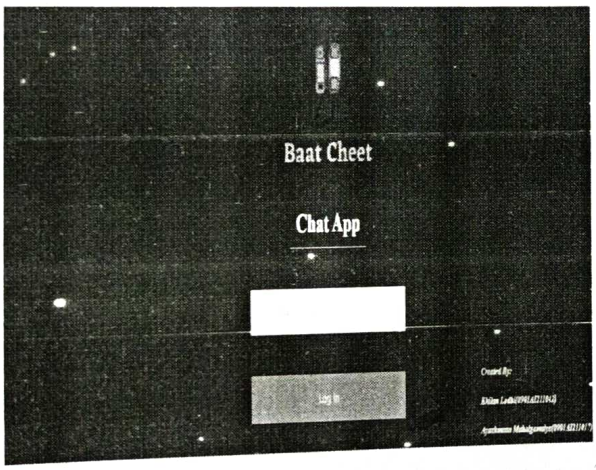
4.1.2 Robust Backend

Node.js and Express.js have proven to be robust choices for the backend, enabling efficient handling of WebSocket connections, user authentication, and database interactions. The integration of Socket.io has facilitated smooth real-time communication.

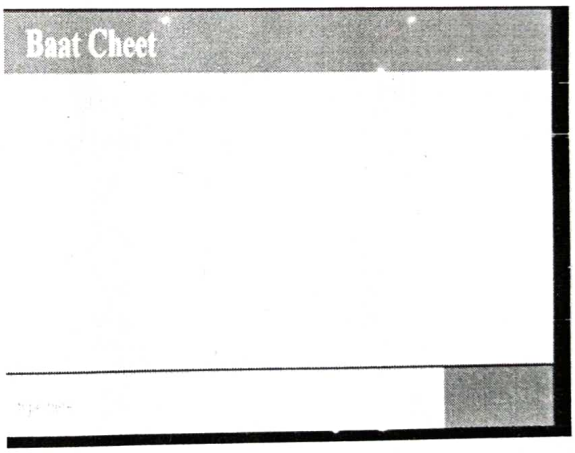
4.1.3 Dynamic Frontend

The frontend, built with ReactJS, has delivered a dynamic and interactive user experience. React's component-based architecture has enabled the creation of modular, reusable UI elements, resulting in an intuitive and responsive application.

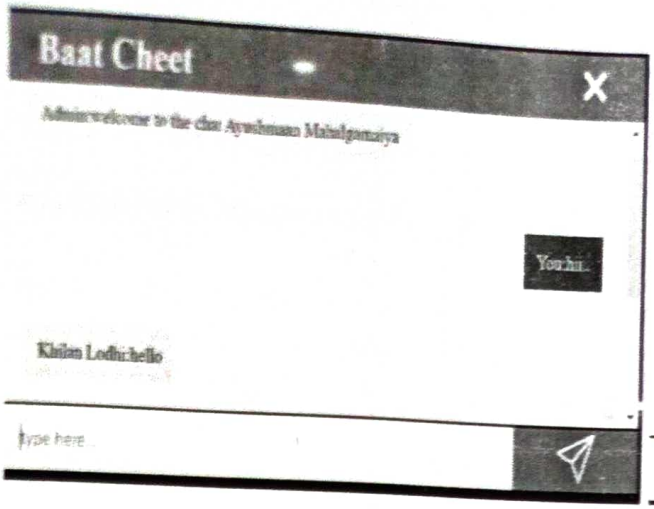
FINAL RESULT



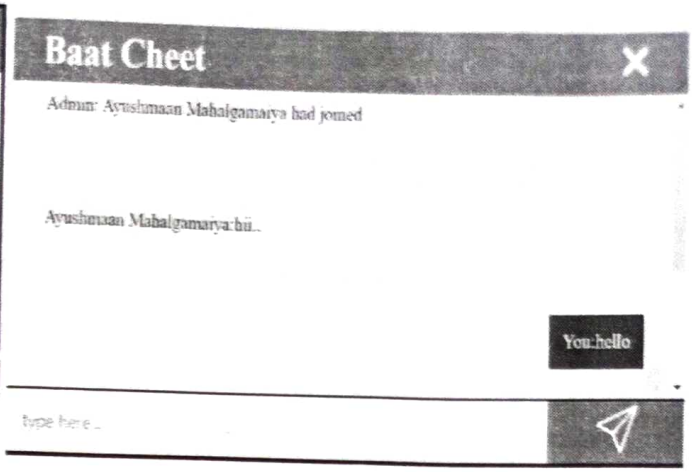
4.1.a Joining page



4.1.b Chat Section



4.1.c Message page



4.1.d Message Page

4.2 Result Analysis

4.2.1 User Feedback

Initial user feedback indicates positive responses to the user interface design and the speed of real-time messaging. Users appreciate the simplicity of the application and the ability to seamlessly share multimedia content.

4.2.2 Performance Metrics

Performance metrics, including page load times and WebSocket connection speeds, have been within acceptable ranges. The asynchronous nature of Node.js and the virtual DOM in React contribute to the overall performance of the application.

4.3 Application

4.3.1 Real-World Applicability

The chat application demonstrates real-world applicability in scenarios requiring instant communication. It is suitable for both casual and professional use, fostering collaboration and connectivity.

4.3.2 Scalability

The application exhibits scalability, with the ability to handle concurrent users and dynamic chat rooms. The architecture allows for straightforward scaling to accommodate increased user loads.

4.4 Problems Faced

4.4.1 Cross-Browser Compatibility

During the development phase, challenges were encountered in achieving consistent cross-browser compatibility. However, thorough testing and adjustments were made to ensure optimal performance across major browsers.

4.4.2 Security Considerations

Ensuring secure authentication and data transmission posed challenges. Comprehensive measures, including encryption and validation, were implemented to address potential security vulnerabilities.

4.5 Limitations

4.5.1 Lack of Offline Functionality

The chat application currently relies on real-time connections and does not provide offline functionality. This limitation may affect users in low-connectivity scenarios.

4.5.2 Minimal User Management Features

While the application focuses on real-time messaging, more extensive user management features, such as user profiles and friend lists, are not currently implemented.

4.6 Conclusion

The development and analysis of the chat application using HTML, ReactJS, Node.js, CSS, and JavaScript have resulted in a functional and efficient real-time communication platform. The integration of these technologies has allowed for the creation of a user-friendly application with the potential for further enhancements. The project has provided valuable insights into the challenges and considerations in building a modern chat application, paving the way for future developments and optimizations.This chapter concludes the project by summarizing the results, analyzing the outcomes, discussing the application's real-world relevance, addressing challenges faced, noting limitations, and ultimately drawing conclusions from the development and analysis of the chat application.

References

1.**HTML, CSS, and JavaScript**

https://youtu.be/6mbwJ2xhgzM?feature=shared

2.Node.js:

[Node.js Documentation](https://nodejs.org/en/docs/)-

3.React.js:

https://youtu.be/-mJFZp84TIY?feature=shared

4. **YouTube Tutorials:**

https://youtu.be/1rc2zYqexLI?feature=shared